G

Gasoline Blending and Distribution Scheduling: An MILP Model

ZHENYA JIA, MARIANTHI IERAPETRITOU
Department of Chemical and Biochemical
Engineering, Rutgers University,
Piscataway, USA

MSC2000: 90B35, 93A30

Article Outline

Synonyms

Indices

$i =$ orders
$j =$ product-stock tanks
$s =$ products
$k =$ components
$l =$ component tanks
$n =$ event points

Sets

$I =$ orders
$I_j =$ orders which can be performed in product-stock tank j
$I_s =$ orders which order product s
$J =$ product-stock tanks
$J_i =$ product-stock tanks which are suitable for performing order i
$J_s =$ product-stock tanks which can store product s
$N =$ event points within the time horizon
$S =$ products
$S_j =$ products which can be stored in product-stock tank j
$K =$ components
$K_l =$ components which can be stored in component-stock tank l
$L =$ component stock tanks
$L_k =$ component-stock tanks which can store component k

Parameters

$Vmax(j) =$ maximum capacity of product-stock tank j
$Vmin(j) =$ minimum amount of product stored in tank j if tank j is utilized

Vinitial(j,s) = amount of product *s* stored in tank *j* initially

Vin(l,k) = amount of component *k* stored in component tank *l* initially

Vcomp(l) = maximum capacity of component tank *l*

Recipe(s,k) = the proportion of component *k* to in product *s*

l(i) = lifting rate of order *i*

Bflow = flow rate of product being produced and transferred to product-stock tanks

Prod_srt(i) = time by which order *i* can start

Prod_end(i) = time by which order *i* is due

U1 = lower bound on the amount of product lifted

U2 = upper bound on the amount of product lifted

U3 = upper bound of a small-sized order

U4 = upper bound of a medium-sized order

U5 = lower bound of a large-sized order

flowmin = minimum flow rate of component tanks

flowmax = maximum flow rate of component tanks

H = time horizon

Variables

uv(i,j,n) = binary variables that assign the beginning of order *i* in tank *j* at event point *n*

y(s,j,n) = binary variables that assign product *s* being stored in tank *j* at event point *n*

sv(s,j,n) = binary variables that assign product *s* being produced and transferred to tank *j* at event point *n*

xv(s,n) = 0-1 continuous variables that assign product *s* being produced at event point *n*

yv(k,l,n) = binary variables that assign component *k* being extracted from component-stock tank *l* at event point *n*

Ts(i,j,n) = starting time of order *i* in tank *j* at event point *n*

Te(i,j,n) = finishing time of order *i* in tank *j* while it starts at event point *n*

lift(i,j,n) = amount of product being lifted for order *i* from tank *j* at event point *n*

Pst(s,j,n) = amount of product *s* in tank *j* at event point *n* before new product is transferred from the blender

Tbs(s,j,n) = starting time of product *s* being produced and transferred to product-stock tank *j* at event point *n*

Tbf(s,j,n) = finishing time of product *s* being produced and transferred to product-stock tank *j* at event point *n*

Blnd(s,j,n) = amount of product *s* being transferred from blender to tank *j* at event point *n*

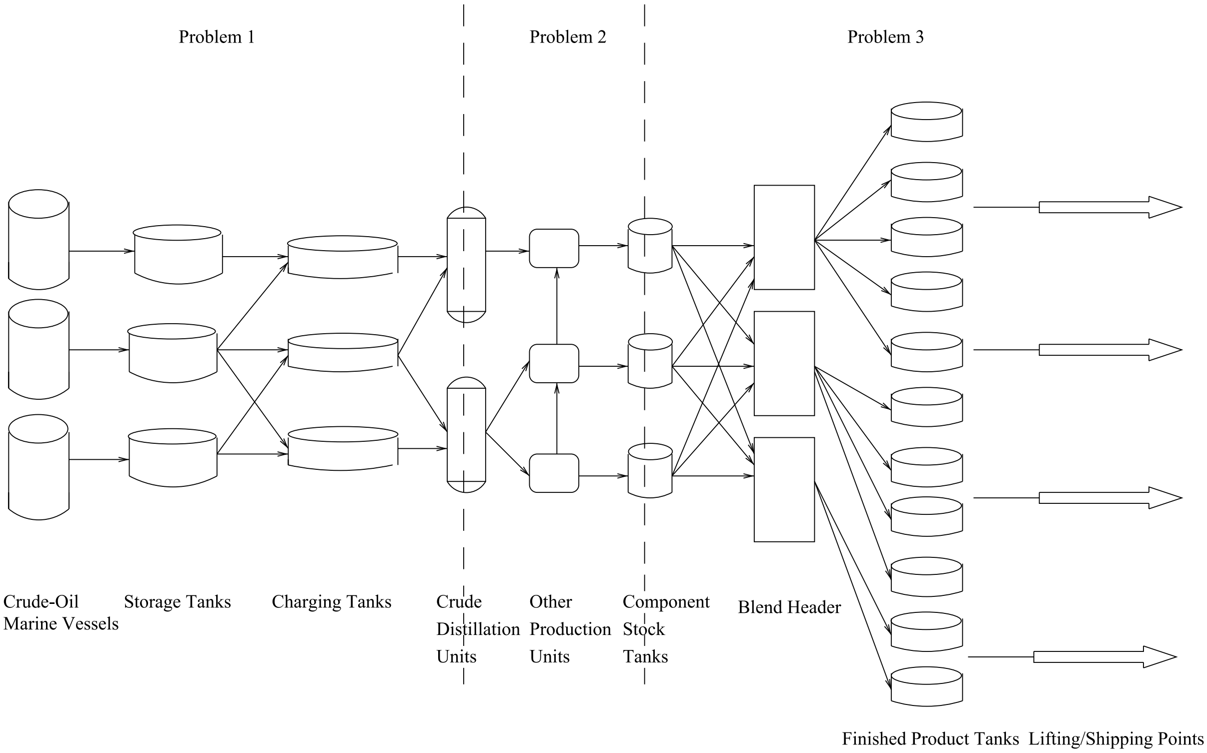
comp(k,l,n) = amount of component *k* being transferred to the blender at event point *n*

bc(k,l,n) = amount of component *k* in component tank *l* at event point *n*

cracking(k,l,n) = amount of component *k* being transferred from separation units to component tank *l* at event point *n*

Introduction

Gasoline blending is a crucial step in refinery operation as gasoline can yield 60–70% of a refinery's profit. The process involves mixing various stocks, which are the intermediate products from the refinery, along with some additives, such as antioxidants and corrosion inhibitors, to produce blends with certain qualities [1]. In the past few decades, a substantial amount of work has been dedicated to process operations [3,4,7,8,9]. A variety of support systems have been developed to address planning and scheduling of blending operations. StarBlend [13], for example, which is developed by Texaco, uses a multiperiod blending model written in GAMS that facilitates the incorporation of future requirements into current blending decisions. Glismann and Gruhn [5,6] proposed a mixed-integer linear model (MILP), which is based on a resource-task network representation, to solve the task of short-term scheduling of blending processes. The recipe optimization problem is then formulated as a nonlinear program and the results are returned to the scheduling problem, so that an overall optimization can be achieved. A fuzzy linear formulation was applied to the blending facilities by Djukanovic et al. [2], in order to address the problem of uncertainty of input information within the fuel scheduling optimization. Singh et al. [14] addressed the



Gasoline Blending and Distribution Scheduling: An MILP Model, Figure 1
Graphic overview of the gasoline blending and distribution system

problem of blending optimization for in-line blending for the case of stochastic disturbances in feedstock qualities. They presented a real-time optimization method that can provide significantly improved profitability.

The objective of this work is to propose a new mathematical model that addresses the simultaneous optimization of the short-term scheduling problem of gasoline blending and distribution as described in the following section.

Definition

The overall oil-refinery system is decomposed into three parts as depicted in Fig. 1. The first part (problem 1, Fig. 1) involves the crude-oil unloading, mixing and inventory control (Jia et al. [10]), the second part (problem 2, Fig. 1) consists of the production unit scheduling, which includes both fractionation and reaction processes, and the third part (problem 3, Fig. 1), which is addressed in this work, depicts the finished product blending and shipping end of the refinery. The gasoline blending system consists of four pieces of equipment all linked together through various piping segments, flow meters and valves. They are component-stock tanks, blend header, product-stock tanks and lifting ports. Components from the component-stock tanks are fed to the blend header according to the recipes. Thus, different products can be produced and then stored in their suitable product-stock tanks. The final step is to lift those products during the specified time periods in order to satisfy all the orders. The objective is to determine the following variables: (1) starting and finishing time of orders taking place in each product-stock tank; (2) the amount and type of product being lifted for each order from tanks; (3) starting and finishing times of the product being transferred from the blender to the tanks; (4) the amount and type of component being transferred from component tanks to the blender, so as to process all the orders in specific time periods.

The scheduling problem as described above is modeled in the next section following a continuous-time representation. It gives rise to an MILP formulation that can be efficiently solved using commercially available solvers.

Formulation

It is assumed that perfect mixing is achieved at the blend header and that the changeover time between different products in the storage tanks is negligible.

Material Balance Constraints for Product-Stock Tank *j*

Constraint (1a) expresses that the amount of product *s* in tank *j* at event point *n+1* (*Pst(s,j,n+1)*) is equal to that at event point n adjusted by any amounts transferred from the blender (*Blnd(s,j,n)*) or lifted at event point *n* ($\sum_{i\in I_s} lift(i,j,n)$). Constraint (1b) states that the amount of product *s* being lifted from tank *j* at the last event point *N* should not exceed the amount of product *s* stored in tank *j*.

$$Pst(s,j,n+1) = Pst(s,j,n) + Blnd(s,j,n) - \sum_{i\in I_s} lift(i,j,n), \quad \forall s \in S, j \in J_s, n \in N, n \neq N \tag{1a}$$

$$Pst(s,j,n) + Blnd(s,j,n) \geq \sum_{i\in I_s} lift(i,j,n)\,, \quad \forall s \in S, j \in J_s, n = N \tag{1b}$$

Capacity Constraints

Constraint (2) imposes a volume capacity limitation of product *s* in tank *j* at event point *n*.

$$Vmin(j) * y(s,j,n) \leq Pst(s,j,n) + Blnd(s,j,n) \leq Vmax(j) * y(s,j,n)\,, \quad \forall s \in S, j \in J_s, n \in N \tag{2}$$

Allocation Constraints

According to constraint (3a), *uv(i,j,n)* is equal to 1 if the amount of product being lifted from tank *j* for order *i* is not zero at event point *n*, that is, $lift(i,j,n) \neq 0$; *uv(i,j,n)* equals 0 otherwise. U1 and U2 correspond to lower and upper bounds on the amount of product lifted, respectively, and are chosen according to the smallest order and the maximum capacities of the tanks.

$$U1 * uv(i,j,n) \leq lift(i,j,n) \leq U2 * uv(i,j,n)\,, \quad \forall i \in I, j \in J_i, n \in N \tag{3a}$$

To avoid task splitting, constraints (3b)–(3d) state that order *i* should be processed only once if it is a small order and at most twice if it is a medium-sized order. Otherwise, it can be processed at most three times. For different problems, U3 and U4 are chosen accordingly to define small and medium-sized orders. Constraint (3e) expresses that for large orders which are defined as greater than or equal to U5, the minimum order splitting is 25 Mbbl.

$$\sum_n \sum_{j\in J_i} uv(i,j,n) = 1\,, \quad \forall \sum_s Prod\_ord(i,s) \leq U3,\ i \in I,\ n \in N \tag{3b}$$

$$\sum_n \sum_{j\in J_i} uv(i,j,n) \leq 2\,, \quad \forall \sum_s Prod\_ord(i,s) \leq U4,\ i \in I,\ n \in N \tag{3c}$$

$$\sum_n \sum_{j\in J_i} uv(i,j,n) \leq 3\,, \forall i \in I,\ n \in N \tag{3d}$$

$$25 * uv(i,j,n) \leq lift(i,j,n)\,, \quad \forall \sum_s Prod\_ord(i,s) \geq U5, i \in I, j \in J_i, n \in N \tag{3e}$$

Constraint (4) forces *sv(s,j,n)* to be equal to 1 when Blnd(s,j,n) is not zero; otherwise sv(s,j,n) equals 0.

$$Vmin(j) * sv(s,j,n) \leq Blnd(s,j,n) \leq Vmax(j) * sv(s,j,n)\,, \quad \forall s \in S, j \in J_s, n \in N \tag{4}$$

Demand Constraints

Constraints (5a) and (5b) state that order *i* can be processed at most once in one tank during the time horizon under consideration and that the amount of product being lifted from all the product-stock tanks should be equal to the amount ordered ($\sum_s Prod\_ord(i,s)$).

$$\sum_n uv(i,j,n) \leq 1\,, \quad \forall i \in I, j \in J_i, n \in N \tag{5a}$$

$$\sum_n \sum_{j\in J_i} lift(i,j,n) = \sum_s Prod\_ord(i,s)\,, \quad \forall s \in S, i \in I, n \in N \tag{5b}$$

Sequence Constraints

Constraints (6a)–(6c) state that order i starting in tank *j* at event point *n+1* should start after the finishing time of the same order processed in the same tank which has started at event point *n*. Constraints (6d) and (6e) express that order *i* should start and finish during the specific time period based on the order requirement. These constraints are relaxed if *uv(i,j,n)* is zero, which means order *i* is not executed in tank *j* at event point n.

$$Ts(i,j,n+1) \geq Te(i,j,n) - H * (1 - uv(i,j,n)) , \quad \forall\, i \in I_j, j \in J, n \in N, n \neq N \tag{6a}$$

$$Ts(i,j,n+1) \geq Ts(i,j,n) , \quad \forall i \in I_j, j \in J, n \in N, n \neq N \tag{6b}$$

$$Te(i,j,n+1) \geq Te(i,j,n) , \quad \forall i \in I_j, j \in J, n \in N, n \neq N \tag{6c}$$

$$Ts(i,j,n) \geq Prod\_srt(i) * uv(i,j,n) , \quad \forall i \in I_j, j \in J, n \in N \tag{6d}$$

$$Te(i,j,n) \leq Prod\_end(i) + H * (1 - uv(i,j,n)) , \quad \forall i \in I_j, j \in J, n \in N \tag{6e}$$

Duration Constraints

If order *i* is processed in tank *j* at event point n, that is, $uv(i,j,n) = 1$, then both ends of constraint (7a) are equal, so the duration is given by $lift(i,j,n)/l(i)$, where $l(i)$ is the lifting rate of order i. If $uv(i,j,n) = 0$, then the duration is zero according to constraint (7b).

$$\frac{lift(i,j,n) - \sum_s Prod\_ord(i,s) * (1 - uv(i,j,n))}{l(i)} \leq Te(i,j,n) - Ts(i,j,n) \leq \frac{lift(i,j,n)}{l(i)} , \quad \forall i \in I_j, j \in J, n \in N \tag{7a}$$

$$Te(i,j,n) - Ts(i,j,n) \leq \frac{\sum_{s \in S_j} Prod\_ord(i,s) * uv(i,j,n)}{l(i)} , \quad \forall i \in I_j, j \in J, n \in N \tag{7b}$$

Blending Stage Consideration

The consideration of the blending stage requires the incorporation of the constraints described in the following constraints.

Material Balance Constraints for the Blender

To avoid the introduction of bilinear terms in the mass-balance equations and to keep the model linear, the idea of component mixing used by Quesada and Grossmann [12] together with the assumption of constant production recipe is used. On the basis of these assumptions, constraint (8) is introduced to express that the required amount of component *k* to produce product *s* at event point *n* $(\sum_s (Recipe(s,k) * \sum_{j \in J_s} Blnd(s,j,n)))$ should be equal to the total amount of component *k* being transferred from all the component tanks at that event point $(\sum_{l \in L_k} comp(k,l,n))$.

$$\sum_s (Recipe(s,k) * \sum_{j \in J_s} Blnd(s,j,n)) = \sum_{l \in L_k} comp(k,l,n) , \quad \forall s \in S, k \in K, n \in N \tag{8}$$

Material Balance Constraints for Component Tank *l*

The amount of component *k* in tank *l* at event point *n+1* *(bc(k,l,n+1))* is equal to that at event point *n (bc(k,l,n))* adjusted by any amounts transferred from separation units (*cracking(k,l,n)*) or delivered to the blender at event point $n(comp(k,l,n))$. This relation is expressed by constraint (9a). Constraint (9b) imposes the upper and the lower bounds on the flow rates of component *k* transferred from tank *l* to the blender.

$$bc(k,l,n+1) = bc(k,l,n) + cracking(k,l,n) - comp(k,l,n), \forall k \in K_l, n \in N \tag{9a}$$

$$flowmin * yv(k,l,n) \leq comp(k,l,n) \leq flowmax * yv(k,l,n) , \quad \forall k \in K, l \in L_k, n \in N \tag{9b}$$

Allocation Constraints for Product-Stock Tank *j*

Constraint (10) states that product *s* cannot be transferred to product-stock tank *j* and distributed at the same event point *n*.

$$\sum_{s \in S_j} sv(s,j,n) + uv(i,j,n) \leq 1 , \quad \forall i \in I_j, j \in J, n \in N \tag{10}$$

Gasoline Blending and Distribution Scheduling: An MILP Model, Table 1
Distribution data for an example with ten orders

Order	o1	o2	o3	o4	o5	o6	o7	o8	o9	o10
Product and amount (Mbbl)	N411	W43	W43	N411	W43	N411	W43	N4132	W43	N5175
Time by which an order can start (hr)	0	0	24	24	48	48	96	118	144	150.5
Due date (hr)	24	24	48	48	72	72	120	190	168	185.5
Lifting rate (Mbbl/hr)	50	50	50	50	50	50	50	8	50	5

Time horizon (hr)	192										
Product-stock tank	pt1	pt2	pt3	pt4	pt5	pt6	pt7	pt8	pt9	pt10	pt11
Products that can be stored	E4W4	E4W4	E4W4	W4 E4N5	E4W4	E4W4	N4N5	N4N5	N4N5	N4N5	N4N5
Initial product and amount (Mbbl)	E490.20	–	W414.08	N587.51	W428.49	W457.59	N413.79	N412.36	N523.96	N485.11	N412.36
Maximum capacity (Mbbl)	92	92	94	91	92	84	94	92	92	91	82
Minimum capacity (Mbbl)	0.92	0.92	0.94	0.91	0.92	0.84	0.94	092	0.92	0.91	0.82

Allocation Constraints for Blender

According to constraint (11a), *xv(s,n)* equals 1 if product s is produced and transferred to at least one tank at event point n, whereas *xv(s,n)* equals 0 if product s is not transferred to any of the tanks at event point n. Constraint (11b) expresses that only one product can be produced in the blender at the same event point n.

$$sv(s,j,n) \le xv(s,n) \le \sum_{j\in J_s} sv(s,j,n)\,, \quad \forall s \in S, n \in N \tag{11a}$$

$$\sum_{s} xv(s,n) \le 1\,, \quad \forall s \in S, n \in N \tag{11b}$$

Sequence Constraints

Similar to constraints (6a)–(6c), constraints (12a)–(12c) state that product s should start being transferred to tank *j* at event point *(n+1)* after the finishing time for the same product transferred to the same tank which started at event point n, whereas constraints (12d) and (12e) represent the requirement of all the transfers to happen within the time horizon H.

$$Tbs(s,j,n+1) \ge Tbe(s,j,n) - H * (1 - sv(s,j,n))\,, \quad \forall s \in S_j, j \in J, n \in N, n \ne N \tag{12a}$$

$$Tbs(s,j,n+1) \ge Tbs(s,j,n)\,, \quad \forall s \in S_j, j \in J, n \in N, n \ne N \tag{12b}$$

$$Tbe(s,j,n+1) \ge Tbe(s,j,n)\,, \quad \forall s \in S_j, j \in J, n \in N, n \ne N \tag{12c}$$

$$Tbs(s,j,n) \le H\,, \quad \forall s \in S_j, j \in J, n \in N \tag{12d}$$

$$Tbe(s,j,n) \le H\,, \quad \forall s \in S_j, j \in J, n \in N \tag{12e}$$

If the blender provides product s for more than one product-stock tank at event point n, then the starting and finishing times for all the tanks should be the same.

$$Tbs(s,j,n) + H * (1 - sv(s,j,n)) \ge Tbs(s,j',n) - H * (1 - sv(s,j',n))\,, \quad \forall s \in S, j \in J_s, j' \in J_s, j \ne j', n \in N \tag{13a}$$

Gasoline Blending and Distribution Scheduling: An MILP Model, Table 2
Blending data for an example with ten orders

Component		A	C7	C6	M	C4	C5	CR	AR	CG
Tanks that can be stored in		ct10	ct9	ct8	ct53,54 ct15,52	ct51	ct57,58 ct60	ct4 ct13	ct55 ct11	ct7,12,17 ct56,59
Recipe of products	N4	0	0.0767	0	0	0.14	0.2742	0.4018	0	0.1073
	N5	0	0	0.0419	0	0.0121	0.5178	0	0.0443	0.384
	E4	0	0	0	0	0.2729	0	0.3897	0	0.3078
	W4	0.6527	0	0	0	0.1591	0	0.1882	0	0
Amount of component (Mbbl) and tank that it is and initially stored in		26.46 ct10	67.90 ct9	59.44 ct8	7.30 ct15 5.75 ct52 3.10 ct53 28.29 ct54	0.59 ct51	0.29 ct57 8.90 ct58 1.64 ct60	19.35 ct13 27.38 ct4	13.84 ct55 25.63 ct11	4.25 ct59 53.41 ct56 49.34 ct51 34.58 ct7
Blending rate (Mbbl/hr)		50								

$$\begin{aligned}&Tbs(s,j,n) - H*(1-sv(s,j,n))\\&\le Tbs(s,j',n) + H*(1-sv(s,j',n))\,,\\&\qquad \forall s \in S, j \in J_s, j' \in J_s, j \neq j', n \in N \qquad (13b)\end{aligned}$$

$$\begin{aligned}&Tbe(s,j,n) + H*(1-sv(s,j,n))\\&\ge Tbe(s,j',n) - H*(1-sv(s,j',n))\,,\\&\qquad \forall s \in S, j \in J_s, j' \in J_s, j \neq j', n \in N \qquad (13c)\end{aligned}$$

$$\begin{aligned}&Tbe(s,j,n) - H*(1-sv(s,j,n))\\&\le Tbe(s,j',n) + H*(1-sv(s,j',n))\,,\\&\qquad \forall s \in S, j \in J_s, j' \in J_s, j \neq j', n \in N \qquad (13d)\end{aligned}$$

Constraints (14a) and (14b) express that product transfer and distribution should be performed consecutively in the same product-stock tank j.

$$\begin{aligned}&Ts(i,j,n+1) \ge Tbe(s,j,n) - H*(1-sv(s,j,n))\,,\\&\qquad \forall i \in I_j, s \in S_j, j \in J, n \in N, n \neq N \qquad (14a)\end{aligned}$$

$$\begin{aligned}&Tbs(s,j,n+1) \ge Te(i,j,n) - H*(1-uv(i,j,n))\,,\\&\qquad \forall i \in I_j, s \in S_j, j \in J, n \in N, n \neq N \qquad (14b)\end{aligned}$$

According to constraint (15), two different products s and s' being transferred to the same or different product-stock tanks have to be transferred consecutively according to the allocation constraint for the blender.

$$\begin{aligned}&Tbs(s,j,n+1) \ge Tbe(s',j',n) - H*(1-sv(s',j',n))\,,\\&\forall s \in S_j, s' \in S_j, s \neq s', j \in J, j' \in J, n \in N, n \neq N\\&\qquad (15)\end{aligned}$$

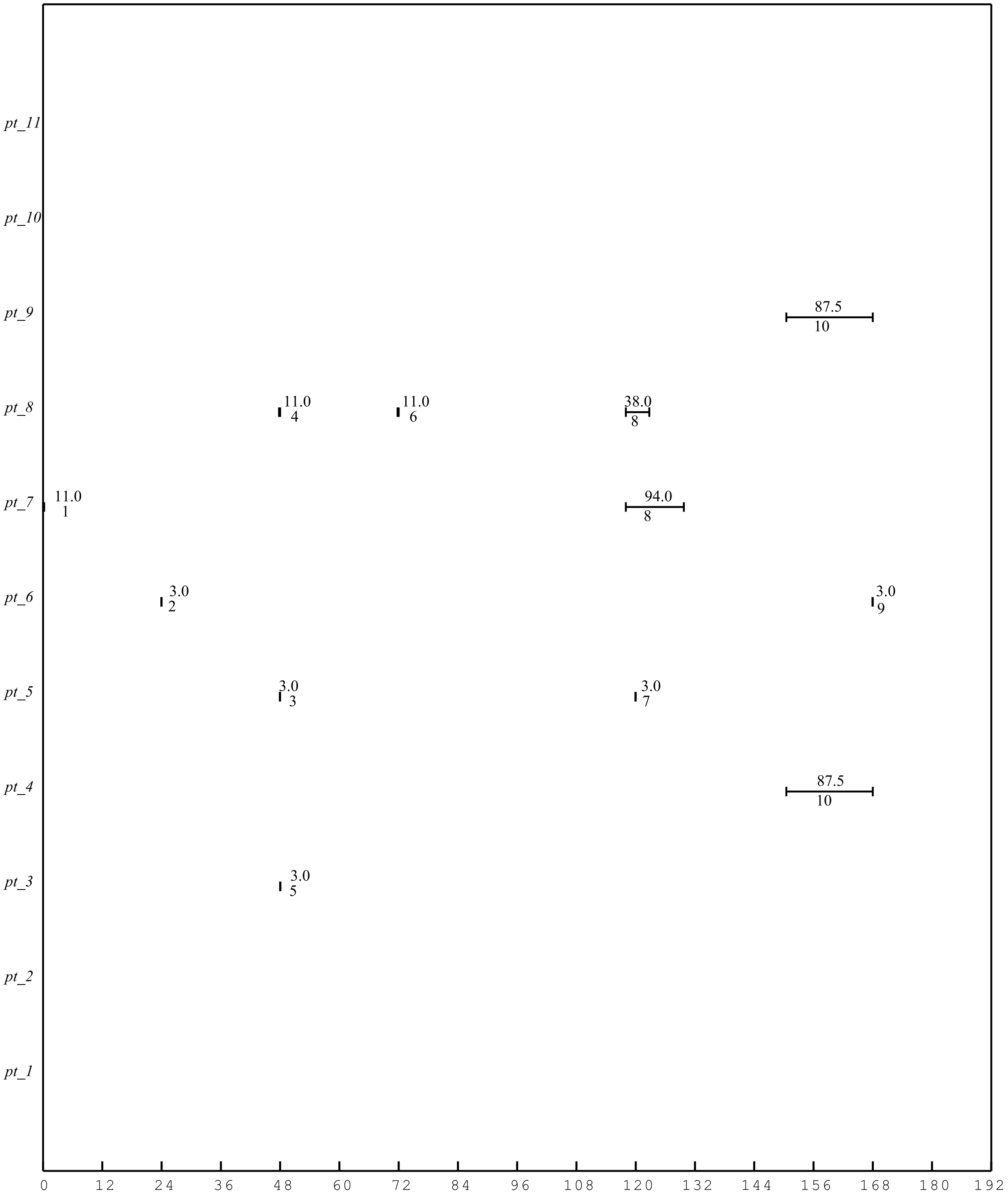
Duration Constraints

The minimum run length of 6h is imposed on the blender by constraint (16a):

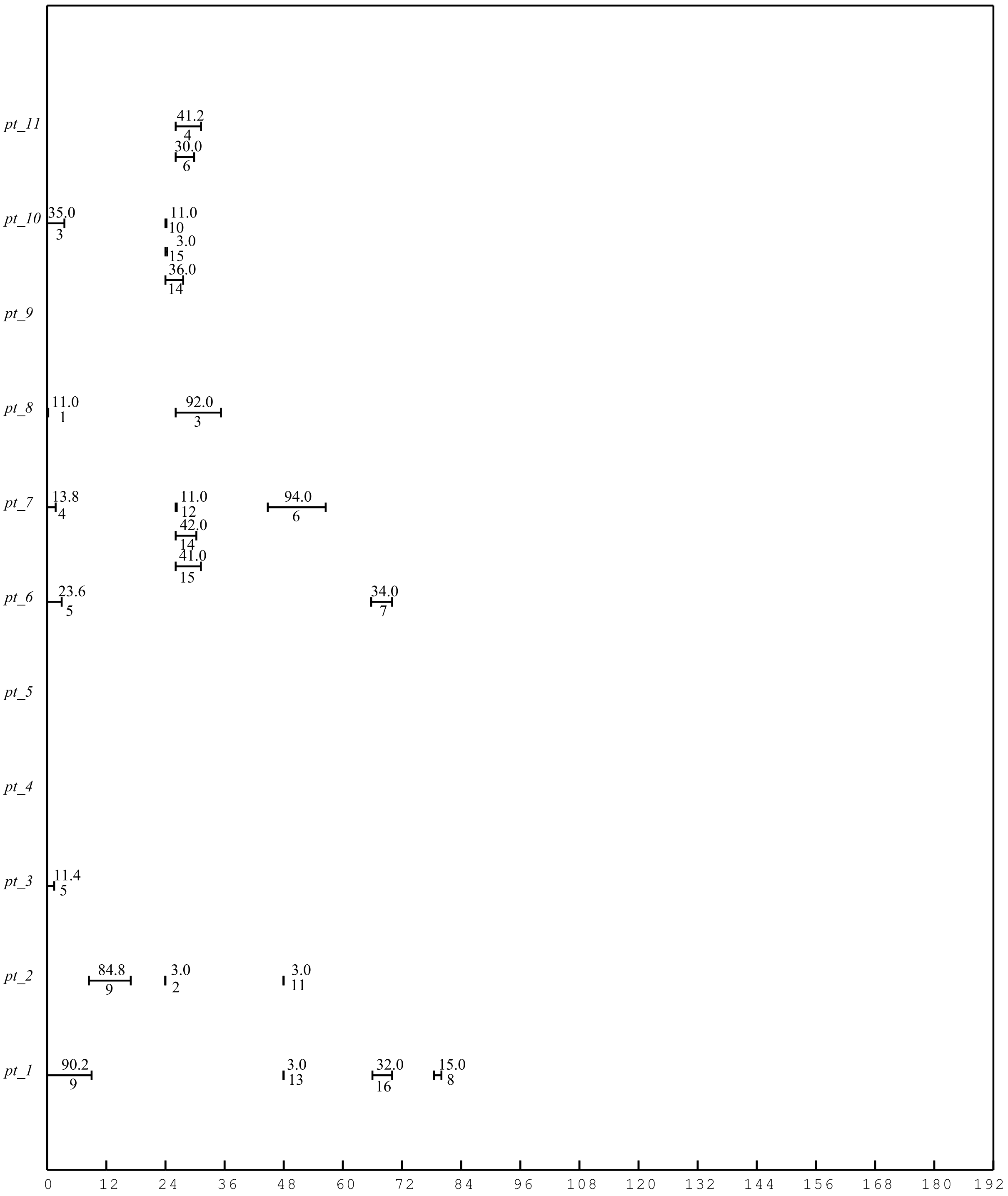
$$\sum_{j \in J_s} Blnd(s,j,n) \ge 6 * Bflow\,, \quad \forall s \in S, n \in N \quad (16a)$$

Constraint (16b) defines the duration of product s being transferred to the tanks at event point n as the difference between the finishing time ($Tbe(s,j,n)$) and the starting time ($Tbs(s,j,n)$), if it takes place in tank j. Constraint (16c) expresses that the duration of transferring product s from the blender to tank j corresponds to the amount of product s being transferred divided by the flow rate. The purpose of having an artificial variable ($arti(s,n)$) is to find a feasible solution in case a larger flow rate is required.

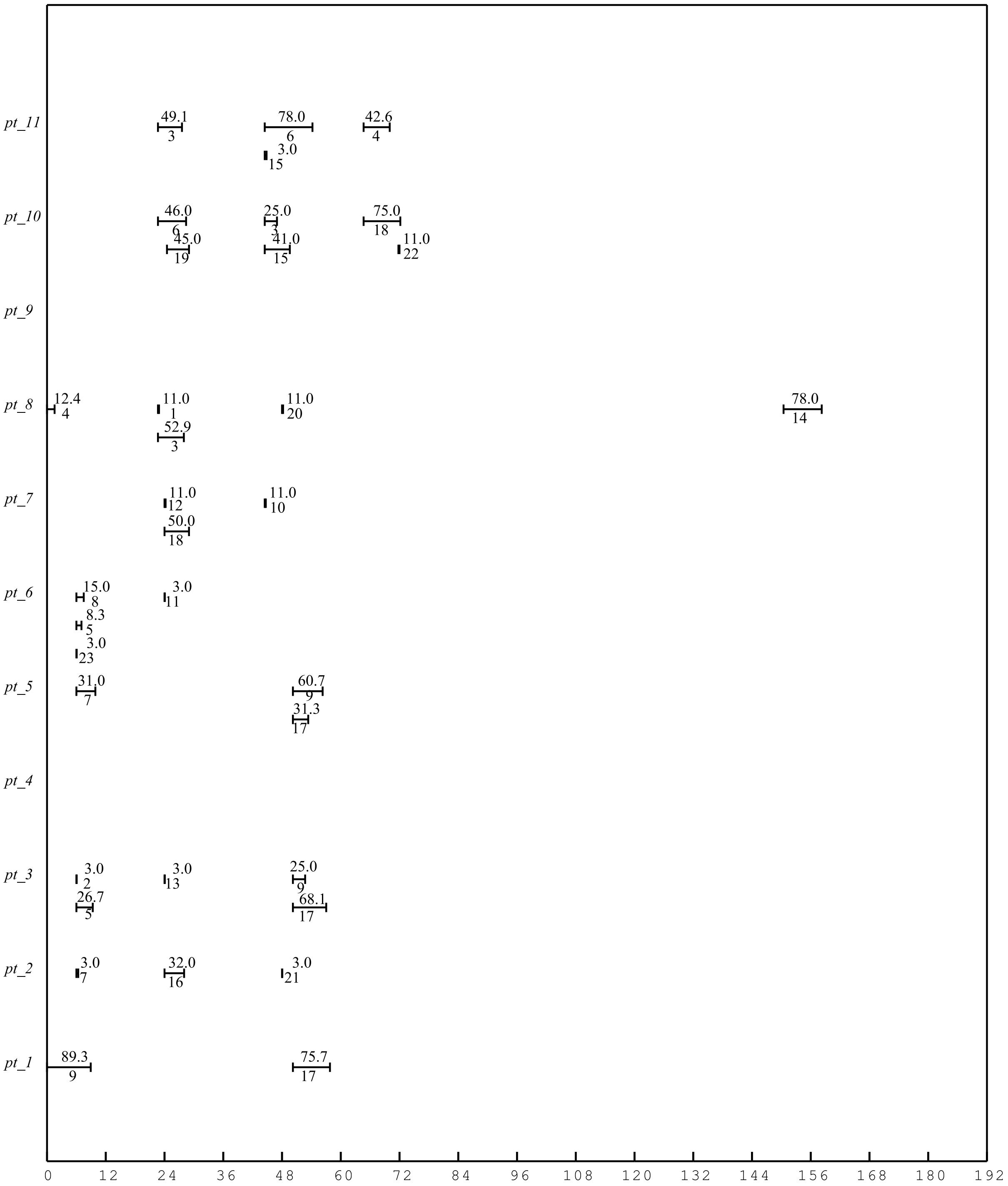
$$\begin{aligned}&(Tbe(s,j,n) - Tbs(s,j,n)) - H*(1-sv(s,j,n))\\&\le duration(s,n)\\&\le (Tbe(s,j,n) - Tbs(s,j,n)) + H*(1-sv(s,j,n))\,,\\&\qquad \forall s \in S_j, j \in J, n \in N \qquad (16b)\end{aligned}$$



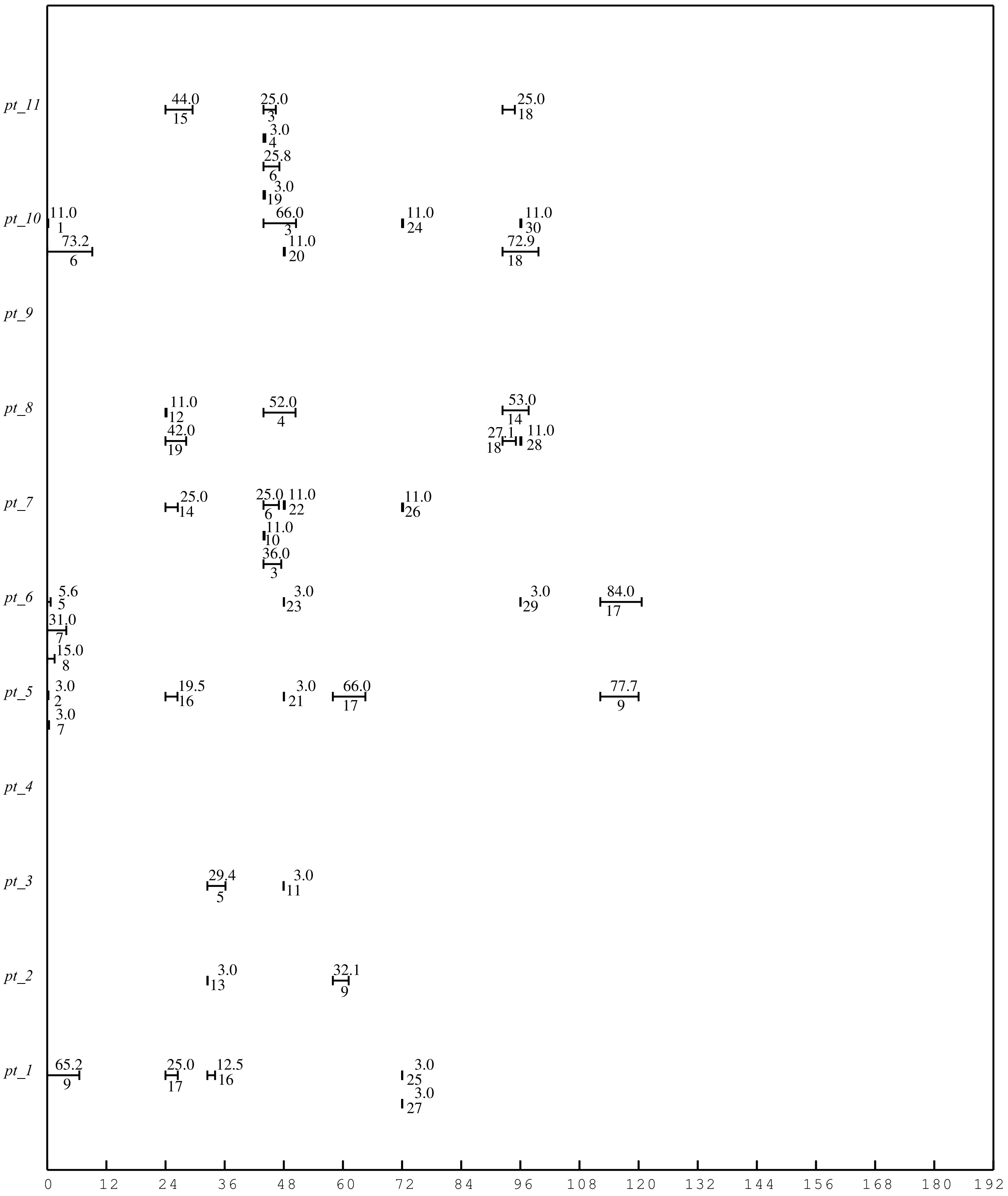
Gasoline Blending and Distribution Scheduling: An MILP Model, Figure 2
Gantt chart for the example with ten orders



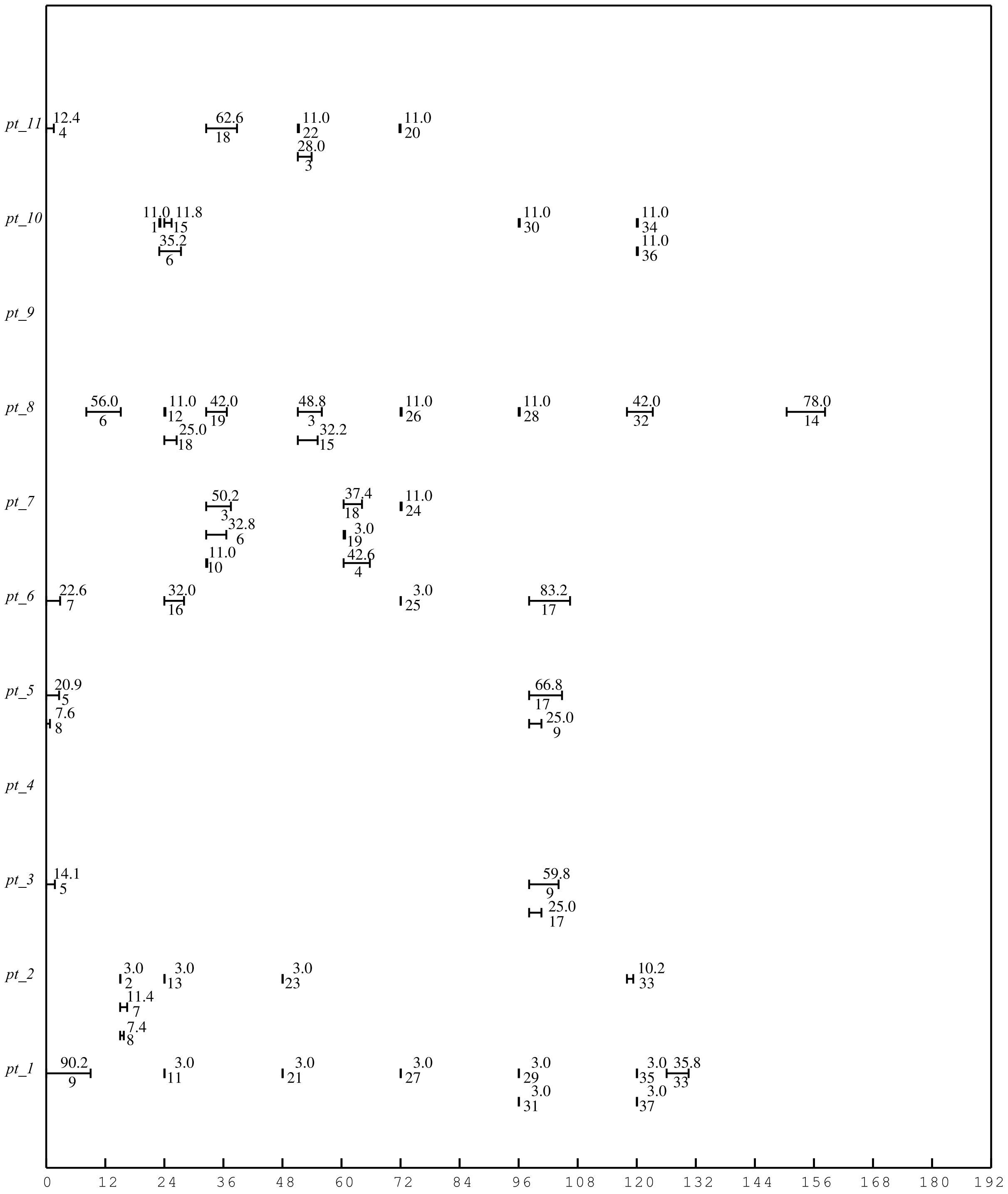
Gasoline Blending and Distribution Scheduling: An MILP Model, Figure 3
Gantt chart for the example with 16 orders



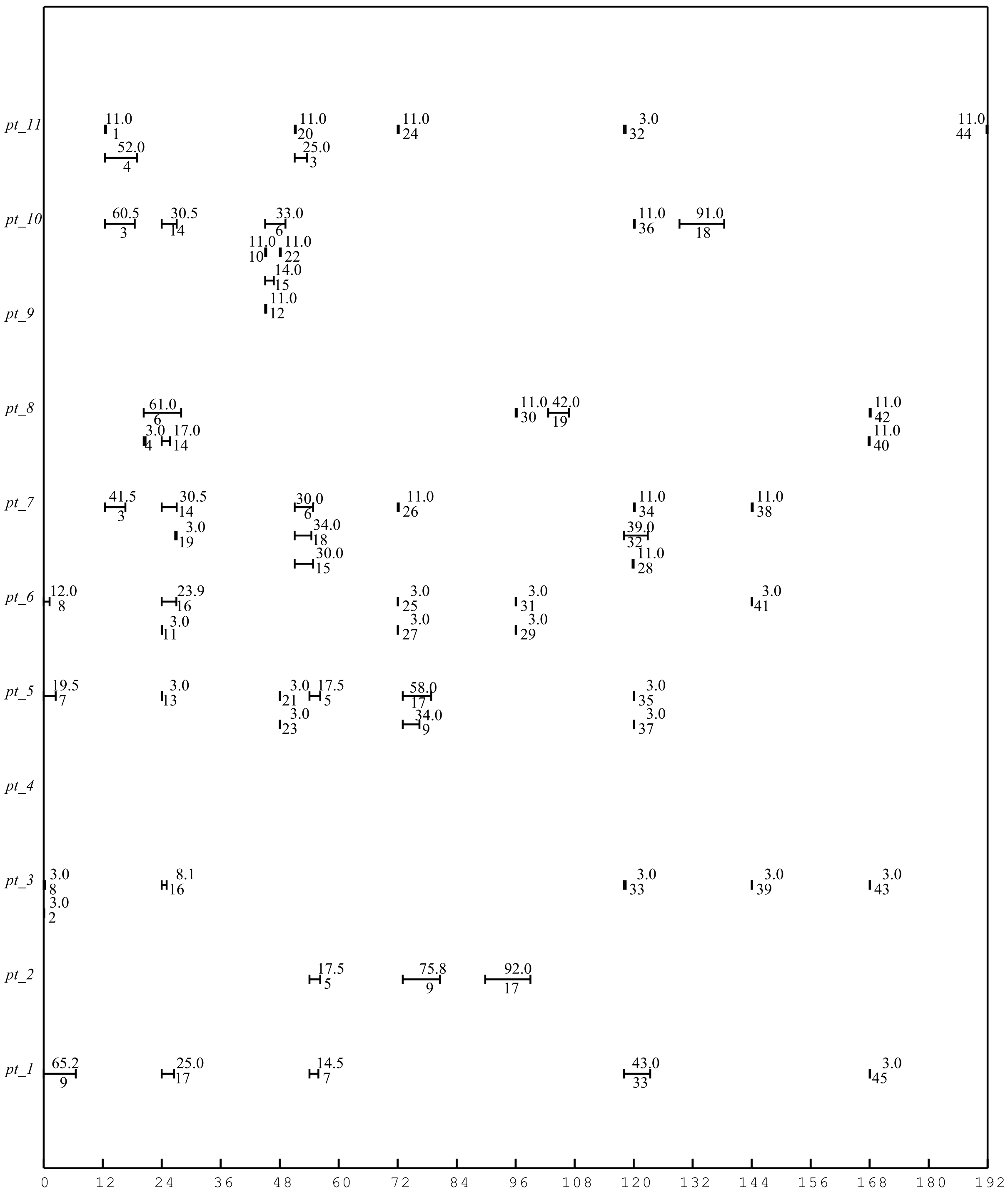
Gasoline Blending and Distribution Scheduling: An MILP Model, Figure 4
Gantt chart for the example with 23 orders



Gasoline Blending and Distribution Scheduling: An MILP Model, Figure 5
Gantt chart for the example with 30 orders



Gasoline Blending and Distribution Scheduling: An MILP Model, Figure 6
Gantt chart for the example with 37 orders



Gasoline Blending and Distribution Scheduling: An MILP Model, Figure 7
Gantt chart for the example with 45 orders

Gasoline Blending and Distribution Scheduling: An MILP Model, Table 3
Computational results for the blending and distribution system

Orders	Continuous variables	0-1 variables	Constraints	1st integer solution				2nd integer solution			Optimal solution		
				Nodes	Iterations	CPU time (s)	Objective value	Nodes	Iterations	Objective value	Nodes	Iterations	CPU time (s)
10	1706	420	7130	21	1495	6.15	0	N/A	N/A	N/A	21	1495	6.15
16	4205	1032	18737	20	3614	29.03	0	N/A	N/A	N/A	20	3614	29.03
23	5974	1470	26746	40	13474	210.24	0	N/A	N/A	N/A	40	13474	210.24
30	9056	2232	40793	80	24906	627.13	0	N/A	N/A	N/A	80	24906	627.13
37	13955	3444	63308	828	177746	4081.49	4.934	1338	244258	0.793	1353	246176	5016.51
45	25454	6289	116452	361	194954	7406.48	6.645	4138	838195	5.094	4280	874051	20351.18

$$duration(s,n) = \frac{\sum_{s \in S_j} Blnd(s,j,n)}{Bflow} - arti(s,n)\,, \quad \forall s \in S_j, j \in J, n \in N \qquad (16c)$$

Objective Function

The objective of the scheduling problem is to minimize the sum of artificial variables in the duration constraints on the blender so as to determine a feasible solution with a flow rate as close to Bflow as possible. The formulation, however, is general to accommodate different objective functions targeting the optimization of production. However, in most realistic cases [11] the objective of this stage of refinery operation is to satisfy all the orders without any delays.

$$objective = \sum_s \sum_n arti(s,n)\,, \quad \forall s \in S, n \in N \quad (17)$$

Case

The case study considered here is based on realistic data provided by Honeywell Hi-Spec Solutions. The distribution problem consists of 45 orders of four different products that are stored in 11 product-stock tanks. The incorporation of the blending stage adds the consideration of nine components and 20 component tanks. Smaller-scale instances of the problem are constructed to test the proposed formulation involving the consideration of 10, 16, 23, 30, and 37 orders. The detailed data for the case often orders are presented in Tables 1 and 2. GAMS/CPLEX 7.0 was used for the solution of the resulting MILP formulation. The computational characteristics of the models are tabulated in Table 3. The optimal solution with zero integrality gap as well as the first and second integer solutions are shown. Note that since the objective corresponds to the summation of artificial variables used to relax the flow-rate constraints, if a solution has a nonzero objective this indicates that one of these constraints has been violated at the cost of the objective function. For the case study examined, however, as shown in Table 3, even the full-scale problem involving 45 orders converged to a feasible solution requiring 4280 nodes in approximately 5h CPU time which is a reasonable time for the solution of the integrated scheduling of blending and distribution problem with a time horizon of 8 days. The resulting Gantt–charts of the six cases examined are shown in Figs. 2–7. Compared with the commonly used Gantt chart for scheduling purposes, the difference here is that the number below the line corresponds to the order number, whereas the number above the line corresponds to the amount of product lifted from this particular tank. Note that different orders can be performed in the same tank at the same time as shown in Figs. 3–7.

Conclusions

In this work, a continuous-time formulation was presented for the short-term scheduling of a gasoline blending and distribution system. It was shown that the resulting model can be solved efficiently even for realistic large-scale problems. The main advantage of the proposed approach is the full utilization of the time continuity. This results in smaller models in terms of variables and constraints since only the real events have to be modeled.

References

1. DeWitt CW, Lasdon LS, Waren AD, Brenner DA, Melhem SA (1989) Omega: an improved gasoline blending system for Texaco. Interfaces 19:85
2. Dujkanovic M, Babic B, Milosevic B, Sobajic DJ, Pao YH (1996) Fuzzy linear programming based optimal fuel scheduling incorporating blending/transloading facilities. IEEE Trans Power Syst 11:1017
3. Floudas CA, Lin X (2004) Continuous-Time versus Discrete-Time Approaches for Scheduling of Chemical Processes: A Review. Comput Chem Eng 28:2109
4. Floudas CA, Lin X (2005) Mixed Integer Linear Programming in Process Scheduling: Modeling, Algorithms, and Applications. Ann Oper Res 139:131
5. Glismann K, Gruhn G (2001) Short-term planning of blending processes: scheduling and nonlinear optimization of recipes. Chem Eng Tech 24:246
6. Glismann K, Gruhn G (2001) Short-term scheduling and recipe optimization of blending processes. Comput Chem Eng 25:627
7. Ierapetritou MG, Floudas CA (1998) Effective Continuous-Time Formulation for Short-Term Scheduling. 1. Multipurpose Batch Processes. Ind Eng Chem Res 37:4341
8. Ierapetritou MG, Floudas CA (1998) Effective Continuous-Time Formulation for Short-Term Scheduling. 2. Continuous and Semicontinuous Processes. Ind Eng Chem Res 37:4360
9. Ierapetritou MG, Hene TS, Floudas CA (1999) Effective Continuous-Time Formulation for Short Term Scheduling. 3. Multiple Intermediate Due Dates. Ind Eng Chem Res 38:3446
10. Jia Z, Ierapetritou MG, Kelly JD (2003) Refinery short-term scheduling using continuous time formulation – crude oil operations. Ind Eng Chem Res 42:3085
11. Kelly JD. Honeywell Hi-Spec Solutions. Personal communication
12. Quesada I, Grossmann IE (1995) Global optimization of bilinear process network with multicomponent flows. Comput Chem Eng 19:1219
13. Rigby B, Lasdon LS, Waren AD (1995) The evolution of Texaco blending systems - from omega to starblend. Interfaces 25:64
14. Singh A, Forbes JF, Vermeer PJ, Woo SS (2000) Model-based real-time optimization of automotive gasoline blending operations. J Process Control 10:43

Gauss, Carl Friedrich

DUKWON KIM
University Florida, Gainesville, USA

MSC2000: 01A99

Article Outline

Keywords

Fundamental theorem of algebra; Method of least squares; Gaussian elimination

C.F. Gauss (1777–1855) worked in a wide variety of fields in both mathematics and physics including number theory, group theory, analysis, differential geometry, geodesy, magnetism, astronomy, and optics. His work has had an immense influence in many areas.

In 1788, Gauss began his education at the Gymnasium with the help of L. Büttner and R. Bartels, where he learned High German and Latin. After receiving a stipend from the Duke of Brunswick–Wolfenbüttel, Gauss entered Brunswick Collegium Carolinum in 1792. At the academy, Gauss independently discovered Bode's law, the binomial theorem and the arithmetic-geometric mean, as well as the law of quadratic reciprocity and the prime number theorem [1,4].

Gauss left Göttingen in 1798 without a diploma, but by this time he had made one of his most important discoveries: the construction of a regular 17-gon by ruler and compasses [2,3]. This was the most major advance in this field since the time of Greek mathematics and was published in his famous work 'Disquisitiones Arithmeticae' [1, Sect. VII].

On July 16, 1799, in his absence, he was awarded his Doctor of Philosophy degree at the university in Helmstedt. His dissertation is a proof of the *fundamental theorem of algebra* (FTA) [2,3]. The fundamental theorem of algebra states that

Theorem 1 *Every polynomial equation of degree n has n roots in the complex numbers.*

Gauss is usually credited with the first proof of the FTA. He is undoubtedly the first to spot the fundamental flaw in earlier proofs, namely the fact that they were assuming the existence of roots and then trying to deduce properties of them. His proof of 1799 is topological in nature and has some rather serious gaps. It does not meet our present-day standards required for a rigorous proof. He published the book 'Disquisitiones Arithmeticae' in the summer of 1801. There were seven sections, all but the last section, referred to above, being devoted to number theory.

In 1814, the Swiss accountant J.R. Argand published a proof of the FTA which may be the simplest of all the proofs. His proof is based on d'Alembert's idea in 1746. Argand simplifies d'Alembert's idea using a general theorem on the existence of a minimum of a continuous function.

Two years after Argand's proof appeared Gauss published in 1816 a second proof of the FTA. Gauss uses Euler's approach but instead of operating with roots which may not exist, Gauss operates with indeterminates. This proof is complete and correct. A third proof by Gauss also in 1816 is, like the first, topological in nature. Gauss introduced in 1831 the term 'complex number'.

In 1849 Gauss produced the first proof that a polynomial equation of degree n with complex coefficients has n complex roots. The proof is similar to the first proof given by Gauss. However it adds little since it is straightforward to deduce the result for complex coefficients from the result about polynomials with real coefficients.

It is worth noting that despite Gauss's insistence that one could not assume the existence of roots which were then to be proved reals he did believe, as did everyone at that time, that there existed a whole hierarchy of imaginary quantities of which complex numbers were the simplest. Gauss called them a *shadow of shadows*.

The different proofs of the FTA are Gauss's most important contributions as a rigorist, that is to say, as a representative of logical strictness in method of proof [1]. Since this theorem has great significance in both algebra and function theory, it influenced many other related areas, including mathematical optimization.

Gauss used infinite sequences and series in his daily work, not only in mathematics but in astronomy, geodesy, and physics. As an eleven-year-old, Gauss was already studying Newton's binomial theorem, which includes the infinite geometric series as a special case. He investigated the conditions under which an infinite binomial series has a logical meaning. He also thought about the theoretical formulation of the notion of limiting value [3]. In an unfinished article written around 1800, 'Fundamental concepts in the principles of series', he formulated the notion of the limit of a sequence in a fashion far ahead of the times.

Gauss introduced there the notions of upper bound and least upper bound G; he also introduced the notions of lower bound and greatest lower bound g. Furthermore he introduced the 'final upper bound' H and the 'final lower bound' h. If $H = h$, then their common value was called the *absolute limit* (limiting value) of the sequence. His definitions nearly agree with the present-day definitions of upper bound G, lower bound g, limit superior H, limit inferior h, and the condition $H = h$ for the existence of the *limiting value* [3,4].

Gauss's great interest in astronomy, and his later interest in geodesy, compelled him to seek a rational method for determining the magnitude of observational errors. In turn, the theory of observational errors forced him to deal with the modes of thought and concepts of the calculus of probabilities. This work had great significance in the development of numerous areas in both the calculus of probabilities and mathematical statistics. Furthermore this theory forced researchers to make clear the conditions under which the *law of the normal distribution* is applicable. This law is often called *Gauss's distribution law*.

In 1823 Gauss published his great work 'Theoria combinationis observationum erroribus minimus obnoxiae' ('A theory for the combination of observations, which is connected with least possible error'). It is a systematic and generalized presentation of his earlier theory of observational errors. Here he develops the *method of least squares* [3,4] with mathematical rigor as, in general, the most suitable way of combining observations, independent of any hypothetical law concerning the probability of error.

The term 'determinant' was first introduced by Gauss in 'Disquisitiones Arithmeticae' (1801) while discussing quadratic forms [3]. He used the term because the determinant determines the properties of the quadratic form. However the concept is not the same as that of our determinant. In the same work Gauss lays out the coefficients of his quadratic forms in rectangular arrays. He describes matrix multiplication (which he thinks of as composition so he has not yet reached the concept of matrix algebra) and the inverse of a matrix in the particular context of the arrays of coefficients of quadratic forms.

Gaussian elimination, which first appeared in the text 'Nine Chapters of the Mathematical Art' written in 200 BC, was used by Gauss in his work which studied the orbit of the asteroid Pallas. Using observations of Pallas taken between 1803 and 1809, Gauss obtained a system of six linear equations in six unknowns. Gauss gave a systematic method for solving such equations which is precisely Gaussian elimination on the coefficient matrix [1].

Gauss's career was marked by distinct periods during which he immersed himself first in astronomy, then in geodesy, and then in physics. Yet he regarded himself first and last as 'entirely a mathematician'. More Gauss was an outstanding example of the few creative thinkers who were equally at home in both pure mathematics and applied mathematics. Gauss was always trying to find new applications of mathematics. He kept many little notebooks in which he wrote down ideas and suggestions as they occurred to him. Always alert to possibilities of applying mathematical theories to practical problems, he foresaw the use of mathematics not only in science and technology, but also in such fields as economics, statistics, finance, and so on.

During his long and active career, Gauss published a considerable number of books and articles in journals. But upon his death in 1855, many unpublished articles, notes, and manuscripts were found in his desk. When his complete 'Collected Works' were finally published later, it had taken a group of German scientists nearly seventy years to edit his writings. Even today the name of Gauss occurs throughout mathematics and related areas over and over again. We have the Gaussian equations in spherical trigonometry; the hypergeometric series is also called the Gaussian series; the normal probability curve is known as the Gaussian curve; Gaussian period is a period of congruent roots in the division of the circle; addition and subtraction logarithms are also known as Gaussian logarithm; in higher geometry we speak of Gauss's theorem and Gauss curvature; certain formulas for approximations are known as *Gaussian approximation methods*.

To appreciate the genius of a man like Gauss we must also see him in perspective, through the eyes of his colleagues, his students, his friends, and in terms of posterity's verdict. No other mathematician of the nineteenth century ever received as much acclaim and recognition as that given to Gauss.

See also

- ▶ Gauss–Newton Method: Least Squares, Relation to Newton's Method
- ▶ Least Squares Problems
- ▶ Linear Programming
- ▶ Symmetric Systems of Linear Equations

References

1. Bühler WK (1981) Gauss: A biographical study. Springer, Berlin
2. Dunnington GW (1955) Carl Friedrich Gauss: Titan of science. Exposition Press, New York
3. Hall T (1970) Carl Friedrich Gauss. MIT, Cambridge, MA
4. Schaaf WL (1964) Carl Friedrich Gauss: Prince of mathematicians. Franklin Watts, London

Gauss–Newton Method: Least Squares, Relation to Newton's Method

WILLIAM R. ESPOSITO,
CHRISTODOULOS A. FLOUDAS
Department Chemical Engineering,
Princeton University, Princeton, USA

MSC2000: 90C30, 90C30, 90C52, 90C53, 90C55

Article Outline

Keywords

Gauss–Newton method; Least squares; Gradient methods

Least squares optimization appears most often in *parameter estimation* problems involving nonlinear models. In this problem the object is to minimize the squared distance between an observed and a fitted value from a model with adjustable parameters. For a single equation model the formulation becomes

$$\min_{\theta} S(\theta) = \sum_{\mu=1}^{n} \left[y_\mu - f(\theta, \mathbf{x}_\mu)\right]^2 , \tag{1}$$

where θ are the adjustable model parameters, y_μ is the observed value of the the dependent variable (assumed to contain error) at the μ data point, $\boldsymbol{x}_\mu$ are the observed values of the independent variables (assumed error free) at the μ data point, and n is the total number of data points observed.

This is a very common and well studied problem. As a result many different solution methods exist. In particular two of the earlier developed methods, Newton's method and the Gauss–Newton approach will be discussed and the relationship between the two will be presented.

Newton's Method

Newton's method is derived based on a second order *Taylor series* expansion of the objective function around the current 'guess' of the solution θ_i:

$$\begin{aligned} Q_i(\theta) &= S(\theta_i) + \mathbf{q}^\top(\theta - \theta_i) \\ &\quad + \frac{1}{2}(\theta - \theta_i)^\top \mathbf{H}(\theta - \theta_i) \end{aligned} \tag{2}$$

with

$$\mathbf{q}_l = \frac{\partial S}{\partial \theta_l} = -2 \sum_{\mu=1}^{n} e_\mu \frac{\partial f_\mu}{\partial \theta_l}, \tag{3}$$

$$\begin{aligned} \mathbf{H}_{lk} &= \frac{\partial^2 S}{\partial \theta_l \partial \theta_k} \\ &= -2 \sum_{\mu=1}^{n} e_\mu \frac{\partial^2 f_\mu}{\partial \theta_l \partial \theta_k} + 2 \sum_{\mu=1}^{n} \frac{\partial f_\mu}{\partial \theta_l} \frac{\partial f_\mu}{\partial \theta_k}, \end{aligned} \tag{4}$$

where $e_\mu = y_\mu - f_\mu$ and $f_\mu = f(\mathbf{x}_\mu, \theta)$. In order to find a stationary point of (2) the first order derivatives are equated to zero:

$$\frac{\partial Q_i}{\partial \theta} = \mathbf{q}_i + \mathbf{H}_i(\theta - \theta_i) = 0. \tag{5}$$

If $\mathbf{H}$ is nonsingular, then the solution of (5) for θ can be written as:

$$\theta = \theta_i - \mathbf{H}_i^{-1}\mathbf{q}_i . \tag{6}$$

The method is implemented in a iterative fashion where the value of θ from (6) is used as the next 'guess' of the solution. The iterations continue until a convergence criterion is reached. Theoretically this should be based on the first order derivatives being equal to zero. But for practically purposes and numerical reasons the criterion is most often based on the change in the parameter values. For example:

$$\frac{|\theta_{i+1} - \theta_i|}{|\theta_i| + \epsilon_1} \le \epsilon_2 , \tag{7}$$

where ϵ_1 and ϵ_2 are arbitrary small constants.

Properties of Newton's Method

Newton's method has the following properties [11]:

- Converges in one iteration if $S(\theta)$ is quadratic, as is the case when the model $f(\theta, \mathbf{x})$ is linear in the parameters.
- Requires that both the first and second derivatives of $S(\theta)$ are computed.
- Inversion of the Hessian matrix of $S(\theta)$ is required at each iteration ($O(n^3)$ operation).
- The iteration is undefined when $\mathbf{H}$ is singular.
- $\mathbf{H}$ is required to be positive definite for the step to reduce the value of the objective function.
- Outside the neighborhood of the minimum, convergence is not guaranteed.

Many of these properties, especially the requirement of second derivatives, makes this method impractical for most physically significant problems.

Gauss–Newton Method

The method developed by C.F. Gauss [7] attempts to overcome some of the drawbacks to the original Newton approach. A closer look at (4) shows that for small errors (e_μ) the first term in the equation is approximately zero:

$$-2 \sum_{\mu=1}^{n} e_\mu \frac{\partial^2 f_\mu}{\partial \theta_l \partial \theta_k} \approx 0 \quad \text{for } e_\mu \ll 1 . \tag{8}$$

Therefore the Hessian matrix $\mathbf{H}_i$ can be approximated as:

$$\mathbf{H}_i \approx \mathbf{H}_i^* = 2 \sum_{\mu=1}^{n} \frac{\partial f_\mu}{\partial \theta_l} \frac{\partial f_\mu}{\partial \theta_k} . \tag{9}$$

A step in the solution method then takes the form:

$$\theta_{i+1} = \theta_i - \mathbf{H}_i^{*-1} \mathbf{q}_i . \tag{10}$$

This method can be viewed as linearizing the nonlinear model, and then solving the resulting linear regression to determine the starting point for the next iteration [4]. The Gauss–Newton method has the following properties [12]:

- Only first derivatives of $S(\theta)$ need to be computed at each iteration.
- The approximated Hessian matrix $\mathbf{H}^*$ is intrinsically positive definite and due to the structure, inversion is much easier.
- The approximation is exact if the errors e_μ tend to zero at the minimum.
- Outside the neighborhood of the minimum, convergence is not guaranteed.

These properties offer improvements over the Newton method especially in the computational effort required.

Comparisons Between Newton and Gauss–Newton Method

Various comparisons have been made between these two methods:

1) If the model fits the data well (i. e., all e_μ are small at the solution), then the Gauss–Newton method often requires no more iterations than the Newton method [1].
2) If the model does not fit the data well (i. e., some e_μ do not tend to zero at the solution), then the Newton method will require fewer iterations than the Gauss–Newton, but the computation times will be similar [6].

Both of these methods are similar in that they fall under the category of *gradient based approaches*. In general, a gradient method is iterative in which the step at each iteration is defined as:

$$\theta_{i+1} = \theta_i - \rho_i \mathbf{R}_i \mathbf{q}_i , \tag{11}$$

where $\mathbf{q}_i$ is defined earlier, ρ_i is the steplength, and $\mathbf{R}_i$ is a matrix which should be positive definite. In the Newton method $\mathbf{R}_i$ is the inverse Hessian $\mathbf{H}^{-1}$, while Gauss–Newton uses the approximation $\mathbf{H}^{*-1}$. As mentioned earlier, the inverse Hessian is not always positive definite, while the approximation is, except in the case that the Jacobian matrix, $\mathbf{q}$, is rank deficient. In the implementation of both methods, the steplength ρ_i is taken as 1.

Variable Steplength

One of the obvious extensions of the method involves a selection of the steplength other than one. At each iteration, the search direction given by the Gauss–Newton step is downhill due to the positive definiteness of the approximate Hessian. But the step does not necessarily result in a reduction of the objective function S, since overshooting the minimum is possible. Therefore a steplength ρ should be chosen such that at least:

$$S(\theta_{i+1}) \le S(\theta_i) . \tag{12}$$

One such method can be found in [3]. First define the function $\Psi_i(\rho)$ as:

$$\Psi_i(\rho) \equiv S(\theta_i - \rho \mathbf{R}_i \mathbf{q}_i) . \tag{13}$$

The value of $\Psi_i(0)$ is defined as $S(\theta_i)$. An initial value of ρ^o is chosen and the value of $\Psi_i(\rho^o)$ is calculated. If $\Psi_i(\rho^o)$ is greater than $\Psi_i(0)$, then obviously this value of ρ is not acceptable. Even if the value of ρ is acceptable, the following process may still offer an improvement.

The function $\Psi_i(\rho)$ can be approximated by a quadratic function which matches at $\rho = 0$, $\rho = \rho^0$, and the slope at $\rho = 0$. The function takes the form:

$$\Psi_i(\rho) \approx a + b\rho + c\rho^2 \tag{14}$$

with the coefficients defined as:

$$\begin{aligned} a &= \Psi_i(0) = S(\theta_i) , \\ b &= \left. \frac{d\Psi_i}{d\rho} \right|_{\rho=0} = -\mathbf{q}_i^\top \mathbf{R}_i \mathbf{q}_i , \\ c &= \frac{\Psi_i(\rho^o) - a - b\rho^o}{(\rho^o)^2} . \end{aligned}$$

The object is to minimize this approximation over ρ. A stationary point occurs at:

$$\rho^* = \frac{-b}{2c} . \tag{15}$$

This calculation can be used in an iterative fashion until an acceptable value of ρ is found which reduces the objective function. Reference [3] contains a detailed implementation of this iterative calculation.

Gauss–Newton Example

This example of the Gauss–Newton approach with a variable steplength is found in [3]. This example consists of a two parameter single equation model of the form:

$$y = \exp\left\{-\theta_1 x_1 \exp\left[-\frac{\theta_2}{x_2}\right]\right\} . \tag{16}$$

The parameters, θ, represent the *Arrhenius constants* for a first order irreversible reaction:

$$A \xrightarrow{k} B$$

with x_1 representing the reaction time, x_2 the reaction temperature, and y the fraction of A remaining. The data for the example can be found in the table below.

μ	x_1(hr)	x_2(K)	y
1	0.10	100	0.980
2	0.20	100	0.983
3	0.30	100	0.955
4	0.40	100	0.979
5	0.50	100	0.993
6	0.05	200	0.626
7	0.10	200	0.544
8	0.15	200	0.455
9	0.20	200	0.255
10	0.25	200	0.167
11	0.02	300	0.566
12	0.04	300	0.317
13	0.06	300	0.034
14	0.08	300	0.016
15	0.10	300	0.066

The objective is to minimize the least squares function:

$$\min_{\theta} S(\theta) = \sum_{\mu=1}^{15} \left[y - f_\mu(\theta)\right]^2 . \tag{17}$$

The gradients, $\mathbf{q}$, of the objective function take the form:

$$q_1 = 2\sum_{\mu=1}^{15} e_\mu f_\mu \exp\left[-\frac{\theta_2}{x_{\mu 2}}\right] x_{\mu 1}, \tag{18}$$

$$q_2 = -2\sum_{\mu=1}^{15} e_\mu f_\mu \frac{\theta_1 x_{\mu 1}}{x_{\mu 2}} \exp\left[-\frac{\theta_2}{x_{\mu 2}}\right], \tag{19}$$

and the approximate Hessian matrix is given by:

$$H^*_{lk} = 2\sum_{\mu=1}^{15} \frac{\partial f_\mu}{\partial \theta_l}\frac{\partial f_\mu}{\partial \theta_k}, \qquad l, k = 1, 2, \tag{20}$$

where:

$$\frac{\partial f_\mu}{\partial \theta_1} = f_\mu \exp\left[-\frac{\theta_2}{x_{\mu 2}}\right] x_{\mu 1}, \tag{21}$$

$$\frac{\partial f_\mu}{\partial \theta_2} = f_\mu \frac{\theta_1 x_{\mu 1}}{x_{\mu 2}} \exp\left[-\frac{\theta_2}{x_{\mu 2}}\right]. \tag{22}$$

The initial guess for the parameter values is taken as:

$$\theta_1 = \begin{pmatrix} \theta_{1,1} \\ \theta_{1,2} \end{pmatrix} = \begin{pmatrix} 750 \\ 1200 \end{pmatrix} .$$

Using this initial guess the value of the objective function, gradients, and approximated Hessian were calculated.

$$S(\theta_1) = 1.090441,$$

$$\mathbf{q}_1 = \begin{pmatrix} -0.002230450 \\ 0.006863795 \end{pmatrix},$$

$$\mathbf{H}^*_1 = \begin{pmatrix} 0.2689478 & -0.7730614 \\ -0.7730614 & 2.310325 \end{pmatrix} \times 10^{-5}.$$

The search step direction ν_1, is calculated from $-\mathbf{H}^{*-1}_1 \mathbf{q}_1$. This is generally accomplished by solving the linear system:

$$-\mathbf{H}^*_1 \nu_1 = \mathbf{q}_1 . \tag{23}$$

Many different numerical techniques exist for the solution of (23), see [5] or [13] for examples. The calculation results in:

$$\nu_1 = \begin{pmatrix} -644.9785 \\ -512.9099 \end{pmatrix} .$$

Initially using a stepsize $\rho^0 = 1$, the following values for the parameters are:

$$\theta^0 = \begin{pmatrix} 105.0215 \\ 687.0901 \end{pmatrix}.$$

An objective value of $S(\theta^0) = 0.9133969$ results, which is less than $S(\theta_1)$. Even though this is an acceptable value, still a different stepsize may give a better result. Using the approximation given in (14) with the following values of the parameters for the fit:

$$\begin{aligned} &\Psi_i(\rho = 0) = 1.090441, \\ &\Psi_i(\rho = 1) = 0.9133969, \\ &\left.\frac{d\Psi_i}{d\rho}\right|_{\rho=0} = -2.081916. \end{aligned}$$

The parabola has a minimum, given by (15), at a steplength $\rho^* = 0.5464714$. The resulting values of the parameters using this steplength are:

$$\theta^1 = \begin{pmatrix} 397.5376 \\ 919.7092 \end{pmatrix}.$$

An objective value of $S(\theta^1) = 0.3345645$ results, which is a large improvement over $S(\theta^0)$. This value of the parameter set, θ^1, is accepted as θ_2, and the iterations continue. The results of the iterations can be found in the table below.

i	$S(\theta_i)$	$\theta_{i,1}$	$\theta_{i,2}$
1	1.090411	750	1200
2	0.3345645	397.5376	919.7092
3	0.05765885	646.0847	938.5288
4	0.04038005	810.6260	965.7625
5	0.03980731	818.3628	962.1228
6	0.03980599	813.4583	960.9063

The value of the parameters and the objective function at the sixth iteration are accepted as the solution to the problem. The final values of the gradients and the approximate Hessian are:

$$\mathbf{q} = \begin{pmatrix} -0.218524 \\ 0.631308 \end{pmatrix} \times 10^{-6},$$

$$\mathbf{H}^* = \begin{pmatrix} 0.271890 & -0.957336 \\ -0.957336 & 3.50371 \end{pmatrix} \times 10^{-5}.$$

The above calculation benefited from that fact that the initial guess for the parameter values was relatively close to the solution. Take now the same example, but using the following parameter values as the starting point of the calculation:

$$\theta_1 = \begin{pmatrix} 100 \\ 2000 \end{pmatrix}.$$

This is obviously a 'worse' starting point than the previous calculation. Using these parameter values the following results:

$$\begin{aligned} S(\theta_1) &= 5.299502, \\ \mathbf{q}_1 &= \begin{pmatrix} -0.0007098080 \\ 0.0002442936 \end{pmatrix}, \\ \mathbf{H}_1^* &= \begin{pmatrix} 0.7036033 & -0.2354773 \\ -0.2354773 & 0.07896382 \end{pmatrix} \times 10^{-7}, \\ \nu_1 &= \begin{pmatrix} -134608.0 \\ -432361.0 \end{pmatrix}, \\ \theta^0 &= \begin{pmatrix} -134508.0 \\ -430361.0 \end{pmatrix}. \end{aligned}$$

Using the value of θ^0, calculated with $\rho^0 = 1$, it is not possible to calculate the value of the objective function since the resulting exponentials are very large. The value of ρ was repeatedly halved until a reasonable value of the objective function was obtained. The value $\rho^0 = 2^{-8} = 0.00390625$ resulted in:

$$\theta^0 = \begin{pmatrix} -425.8140 \\ 311.0039 \end{pmatrix}.$$

An objective value of $S(\theta^0) = 0.3366272 \times 10^{20}$ results, which is not acceptable. The stepsize needs to be adjusted such that the objective function decreases. This is accomplished in the same way as outlined previously. The parabolic approximation reaches a minimum at $\rho^* \approx 5 \times 10^{-25}$. This is too small to be practical, so a value of $\rho^1 = \rho^0/4$ will be used. This results in $S(\theta^1) = 5.471375$. Again this is not acceptable since it is larger than $S(\theta_1)$. The value of ρ is iterated on until an acceptable value is determined. Finally after three more iterations, $\rho^4 = 0.0000619701$, which produces:

$$\theta^4 = \begin{pmatrix} 91.65955 \\ 1973.211 \end{pmatrix}.$$

An objective value of $S(\theta^4) = 5.299135$ results, which is just less than the original value of 5.299502, but given the criterion in (12) is acceptable. θ^4 is accepted as θ_2 and the iterations continue.

The solution, in this case, is obtained after 25 iterations. This illustrates the major downfall of the Gauss–Newton method, that without a 'good' initial guess convergence to the solution is slow at best and not guaranteed. In fact without using a variable stepsize, the algorithm would have blown up after just one iteration.

Modifications and Applications

A very large number of different variations on the basic Gauss–Newton algorithm exist. For the most part, these variations include methods to determine the stepsize, and approaches which actually improve the accuracy of the approximated Hessian matrix. For examples of different variations see [10] or [8]. Others have done comparisons and numerical experiments with popular variations to test their applicability to a wide range of problems [2,15]. The algorithm has also been applied to what is referred to as *weighted least squares* (WLS) in which each term in the objective function receives a different coefficient:

$$\min_{\theta} \overline{S}(\theta) = \sum_{\mu=1}^{n} w_{\mu} \left[y_{\mu} - f(\theta, \mathbf{x}_{\mu}) \right]^2 , \qquad (24)$$

where w_{μ} is the weighting for the μth data point, see [14] and [9] for examples.

See also

- ▶ ABS Algorithms for Linear Equations and Linear Least Squares
- ▶ ABS Algorithms for Optimization
- ▶ Gauss, Carl Friedrich
- ▶ Generalized Total Least Squares
- ▶ Least Squares Orthogonal Polynomials
- ▶ Least Squares Problems
- ▶ Nonlinear Least Squares: Newton-type Methods
- ▶ Nonlinear Least Squares Problems
- ▶ Nonlinear Least Squares: Trust Region Methods

References

1. Bard Y (1967) A function minimization method with application to parameter estimation. In: New York Sci Center Report 3220902, IBM
2. Bard Y (1970) Comparison of gradient methods for the solution of nonlinear parameter estimation problems. SIAM J Numer Anal 7(1):157–186
3. Bard Y (1974) Nonlinear parameter estimation. Acad. Press, New York
4. Bard Y, Lapidus L (1968) Kinetics analysis by digital parameter estimation. Catalysis Rev 2(1):67–112
5. Coleman TF, Van Loan C (1988) Handbook of matrix computations. SIAM, Philadelphia
6. Flanagan PD, Vitale PA, Mendelsohn J (1969) A numerical investigation of several one-dimensional search procedures in nonlinear regression problems. Technometrics 11(2):265–284
7. Gauss KF (1809) Theoria motus corporum coelestium. Werke 7:240–254
8. Gill PE, Murray W (1978) Algorithms for the solution of the nonlinear least-squares problem. SIAM J Numer Anal 15(5):977–992
9. Guillaume P, Pintelon R (1996) A Gauss–Newton-like optimization algorithm for "weighted" nonlinear least-squares problems. IEEE Trans Signal Processing 44(9):2222–2228
10. Hartley HO (1960) The modified Gauss–Newton method for the fitting on non-linear regression functions by least squares. Technometrics 3(2):269–280
11. Mckeowen JJ (1975) On algorithms for sums of squares problems. In: Dixon LCW, Szego GP (eds) Toward Global Optimization. North-Holland, Amsterdam, 229–257
12. Meyer RR (1970) Theoretical and computational aspects of nonlinear regression. In: Rosen J, Mangasarian O, Ritter K (eds) Nonlinear Programming. Acad. Press, New York, pp 466–487
13. Press WH, Teukolsky SA, Vetterling WT, Flannery BP (1995) Numerical recipes in C, 2nd edn. Cambridge Univ. Press, Cambridge
14. Reedy PV, Niranjan, Sridharan K, Rao PV (1996) WLS method for parameter estimation in water distribution networks. J Water Resources Planning and Management 122(3):157–164
15. Spedicato E, Vespucci MT (1988) Numerical experiments with variations of the Gauss–Newton algorithm for nonlinear least squares. J Optim Th Appl 57(2):323–339

Gene Clustering: A Novel Decomposition-Based Clustering Approach: Global Optimum Search with Enhanced Positioning

MENG PIAO TAN, CHRISTODOULOS A. FLOUDAS
Department of Chemical Engineering,
Princeton University, Princeton, USA

MSC2000: 91C20, 90C11, 90C26

Article Outline

Introduction

The aim of cluster analysis is to establish a set of clusters such that the data points in a cluster are more similar to one another than they are to those in other clusters. The clustering problem is old, can be traced back to Aristotle, and has already been studied quite extensively by 18th century naturalists such as Buffon, Cuvier, and Linne [19]. Since then, clustering has been used in many disciplines, such as market research, social network analysis, and geology, thus reflecting its broad appeal and utility as a key step in exploratory data analysis [26]. In market research for instance, cluster analysis is widely used when working with multivariate data from surveys and test panels. Market researchers use cluster analysis methods to segment and determine target markets, and position new products. Cluster analysis is also used in the service of market approaches to the establishment of business enterprise value. Johnson [28] addresses the potential role and utility of cluster analysis in transfer pricing practices. Given the importance of clustering, a substantial number of books, such as [11,20,27,39], as well as review papers, such as [58] have been published on this subject.

In biology, clustering provides insights into transcriptional networks, physiological responses, gene identification, genome organization, and protein structure. Genome-wide measurement of mRNA expression levels is an efficient way of gathering comprehensive information on genetic functions and transcriptional networks. However, extracting useful information from the resulting data sets first involves organizing genes by their pattern and/or intensity of expression in order to define those that are co-regulated. Such information provides a basis for extracting regulatory motifs for transcription factors driving the diverse expression patterns, allowing assembly of predictive transcriptional networks [2]. This information also provides insights into the functions of unknown genes, since functionally related genes are often co-regulated [55]. Furthermore, clustered array data provides identification of distinct categories of otherwise indistinguishable cell types, which can have profound implications in processes such as disease progression [50]. In sequence analysis, clustering is used to group homologous sequences into gene families. Examining characteristic DNA fragments helps in the identification of gene structures and reading frames. In protein structure prediction, clustering the ensemble of low energy conformers is used to identify the top suggested protein structures.

Two common similarity metrics are correlation and Euclidean distance. The latter is often popular, since it is intuitive, can be described by a familiar distance function, and satisfies the triangular inequality. Clustering methods that employ asymmetric distance measures [33,41] are probably more difficult to intuitively comprehend even though they may be highly suited to their intended applications. The earliest work on clustering emphasized visual interpretations for the ease of study, resulting in methods that utilize dendograms and color maps [5]. Other examples of clustering algorithms include: (a) Single-Link and Complete-Link Hierarchical Clustering [27,49], (b) K-Means Algorithm and its family of variants, such as the K-Medians [21,34,37,60,61], (c) Reformulation Linearization-based Clustering [1,46], (d) Fuzzy Clustering [3,9,44,47], (e) Quality Cluster Algorithm (QTClust) [23], (f) Graph-Theoretic Clustering [17,57,59], (g) Mixture-Resolving Clustering Method [7,26], (h) Mode Seeking Algorithms [26], (i) Artificial Neural Networks for Clustering [4,31] such as the Self-Organizing Map (SOM) [32] and a variant that combines the SOM with hierarchical clustering, the Self-Organizing Tree Algorithm (SOTA) [22] (j) Information-Based Clustering [8,48,54], (k) Stochastic Approaches [30,36,38]. Some of these methods, such as the K-Means and Information Clustering, are optimization-based approaches, in which the clustering is represented as an unknown parameter vector of a cost function. The process then seeks to obtain the best clustering by minimizing this cost function. Other classes of clustering methods such as competitive learning may not have a straightforward cost function. For instance, in the SOM, cluster centers are arbitrarily chosen initially, after which random data points are selected and placed into the nearest cluster, whose center is updated accordingly after each selection. Clustering ceases when the cluster centers become stationary.

Recently, Tan et al. [51,52] presents a novel optimization-based Mixed-Integer Nonlinear Programming (MINLP) clustering algorithm, the Global Optimal Search with Enhanced Positioning (EP_GOS_Clust), which is robust yet intuitive. This algorithm is significant in that it is able to progressively identify and weed out outlier data points. In addition, it involves a pre-clustering process that is rigorous and has a clearly-defined decision criterion. This is notable as the results of many clustering methods based on function optimization schemes often vary depending on the random initialization or starting heuristics. The EP_GOS_Clust also contains a convenient method to predict the optimal cluster number. The algorithm is compared with several approaches commonly used in clustering biological microarray data, namely K-methods, QTClust, SOM, and SOTA. By comparing the intra-cluster and inter-cluster error sums, as well as the strength of biological coherence based on Gene Ontology resources and expression pattern correlation, the EP_GOS_Clust is shown to compare favorably against other methods. The following sections will describe this novel clustering approach in more detail.

Formulations

Notation and Pre-Clustering

The measure of distance for a gene i, for $i = 1, \dots, n$ having k features (or dimensions), for $k = 1, \dots, s$ is defined as a_{ik}. Each gene is to be assigned to only one (hard clustering) of c possible clusters, each with center z_{jk}, for $j = 1, \dots, c$. The binary variables w_{ij} indicates whether gene i falls within cluster j ($w_{ij} = 1$, if yes; $w_{ij} = 0$, if no).

Pre-clustering the data is important to expedite the computational resources required to solve the hard clustering problem by (i) identifying genes with similar experimental responses, and (ii) removing outliers deemed not to be significant to the clustering process. A straightforward pre-clustering approach to provide just the adequate amount of discriminatory characteristics so that the genes can be pre-clustered properly is to reduce the quantities represented in the k-dimensional expression vectors into a set of representative variables $\{+, o, -\}$. The $(+)$ variable represents an increase in expression level compared to the previous time point, the $(-)$ variable represents a decrease in expression level from the previous time point, and the (o) variable represents an expression level that does not vary significantly across the time points. The expression data can also be pre-clustered by creating a rank-ordered list of gene proximities based on Euclidean distance or correlation. Genes that demonstrate an obvious level of proximity, such as a separation of only at most 1% of the maximum inter-gene distances, are then grouped together. The pre-clusters are the proximity genes that form a complete clique, that is, there is a link between every gene within the same pre-cluster. With this choice, a maximal clique search can be performed by using various levels of pre-clustering criteria. Clearly, when the criterion is overly lenient, a large number of pre-clusters are formed, but most of the genes will belong to multiple pre-clusters, and the number of maximal cliques formed is small. On the other hand, an unnecessarily strict cut-off results in a small number of pre-clusters, thus not accurately reflecting the extent of relatedness between the data. In pre-clustering over a range of cut-off values, we can then select the appropriate criterion as the point where the maximum number of complete cliques is formed [53].

Hard Clustering by Global Optimization The global optimization approach seeks to minimize the Euclidean distances between the data points and the centers of their assigned clusters as:

$$\begin{aligned} &\underset{w_{ij}, z_{jk}}{\text{Minimize}} && \sum_{i=1}^{n} \sum_{j=1}^{c} \sum_{k=1}^{s} w_{ij} \left(a_{ik} - z_{jk}\right)^2 \\ &\text{s.t.} && \sum_{j=1}^{c} w_{ij} = 1\,, \quad \forall i = 1, \dots, n \\ & && w_{ij} \text{ are binary variables, } z_{jk} \text{ are} \\ & && \text{continuous variables}\,. \end{aligned}$$

(Problem 1)

There are two sets of variables in the problem, w_{ij} and z_{jk}. While the bounds of w_{ij} are clearly 0 and 1, that of

z_{jk} is obtained by observing the range of a_{ik} values.

$$z^L_{jk} = \min\{a_{ik}\}\ , \quad \forall k = 1, \ldots, s$$
$$z^U_{jk} = \max\{a_{ik}\}\ , \quad \forall k = 1, \ldots, s\ .$$

The pre-clustering work suggests that some of the genes need only be restricted to some number of known clusters, since it can be determined (for instance by distance and correlation metrics) that certain genes are exceedingly dissimilar from some of the pre-clusters and thus have virtually zero probability of being clustered there. This restriction can be described by introducing an additional binary parameter suit_{ij}. A data point deemed to belong uniquely to just one cluster will only have $\text{suit}_{ij} = 1$ for only one value of j and zero for the others, whereas a data point restricted to a few clusters will have $\text{suit}_{ij} = 1$ for only those clusters. This reduces the computational demands of the problem. The introduction of the suit_{ij} parameters also obviates the need for constraints that prevent the redundant re-indexing of clusters. Together with the necessary first-order optimality condition (i. e., the vector distance sum of all genes within a cluster to the cluster center should be intuitively zero), the formulation becomes:

$$\begin{aligned}
\underset{w_{ij}, z_{jk}}{\text{Minimize}} \quad & \sum_{i=1}^{n}\sum_{k=1}^{s} a^2_{ik} \\
& - \sum_{i=1}^{n}\sum_{j=1}^{c}\sum_{k=1}^{s} (\text{suit}_{ij})(a_{ik} w_{ij} z_{jk}) \\
\text{s.t.} \quad & (\text{suit}_{ij}) \left(z_{jk} \sum_{i=1}^{n} w_{ij} - \sum_{i=1}^{n} a_{ik} w_{ij} \right) \\
& = 0\ , \quad \forall j, \forall k \\
& \sum_{j=1}^{c} (\text{suit}_{ij}) w_{ij} = 1\ , \quad \forall i \\
& 1 \le \sum_{j=1}^{n} (\text{suit}_{ij}) w_{ij} \le n - c + 1 \\
& w_{ij} = 0 - 1\ , \quad \forall i, \forall j \\
& z^L_{jk} \le z_{jk} \le z^U_{jk}\ , \quad \forall j, \forall k\ .
\end{aligned}$$

(Problem 2)

The first set of constraints are the necessary optimality conditions, the second demand that each gene can belong to only one cluster, and the third state that there is at least one and no more than $(n - c + 1)$ data points in a cluster. Note also that the $\sum_{i=1}^{n}\sum_{k=1}^{s} a^2_{ik}$ term in the objective function of Problem 2 is a constant and can be dropped, though for the sake of completeness we will retain the term throughout the subsequent formulations in the paper. Problems 1 and 2 are Mixed Integer Nonlinear Programming (MINLP) problems with bilinear terms in the objective function and the first set of constraints. To handle the nonlinearities formed by the product of variables w_{ij} and z_{jk}, new variables y_{ijk} along with additional constraints [12] are defined as follows:

$$y_{ijk} = w_{ij} z_{jk} \tag{1}$$

$$z_{jk} - z^U_{jk}\left(1 - w_{ij}\right) \le y_{ijk} \le z_{jk} - z^L_{jk}\left(1 - w_{ij}\right) \tag{2}$$

$$z^L_{jk} w_{ij} \le y_{ijk} \le z^U_{jk} w_{ij}, \quad \forall i, \forall j, \forall k\ . \tag{3}$$

The introduction of y_{ijk} and the additional constraints reduces the formulation to an equivalent Mixed-Integer Linear Programming (MILP) problem, but results in an inordinately large number of variables. Thus, there is a need for new approaches to address large datasets.

The GOS Algorithm for Clustering The introduction of the bilinear variable y_{ijk} results in a large number of variables to be considered. In a problem with over 2000 data points, each having 24 features, to be placed into over 380 clusters, the number of variables to be considered numbers over 18 million. Without introducing the y_{ijk} variables will leave the problem in a nonlinear form. Mixed-integer nonlinear programming (MINLP) problems are considered extremely difficult. Theoretical advances and prominent algorithms for solving MINLP problems are addressed in [12,13,15].

The MINLP clustering formulation described in Problem 2 can be solved by a variant of the Generalized Benders Decomposition (GBD) algorithm [14], denoted as the Global Optimum Search (GOS). The primal problem results from fixing the binary variables to a particular 0-1 combination. Here, w_{ij} is fixed and z_{jk} is solved from the resultant linear programming (LP) problem. In addition, the solution also includes the relevant Lagrange multipliers. The master problem is essentially the problem projected onto the y-space (i. e., that of the binary variables). To expedite the solution of this projection, the dual representation of the master is used. This dual representation is in terms of the supporting Lagrange functions of the projected problem. It is assumed that the optimal solution of the primal problem as well as its Lagrange multipliers can be used for the determination of the support function. In the master problem, the z_{jk} solution from the accompanying primal is taken and the master is solved for the w_{ij} variables.

The two sequences of upper and lower bounds are then iteratively updated until they converge in a finite number of iterations. With each successive iteration, a new support function is added to the list of constraints for the master problem. Thus in a sense, the support functions for the master problem build up with each iteration, forming a progressively tighter envelope and gradually pushing up the lower bound solution until it converges with the upper bound solution.

With fixed starting values for w_{ij}, the primal problem becomes:

$$\begin{aligned} \underset{z_{jk}}{\text{Minimize}} \quad & \sum_{i=1}^{n}\sum_{k=1}^{s} a_{ik}^2 - \sum_{i=1}^{n}\sum_{j=1}^{c}\sum_{k=1}^{s} a_{ik} w_{ij}^{*} z_{jk} \\ \text{s.t.} \quad & z_{jk}\sum_{i=1}^{n} w_{ij}^{*} - \sum_{i=1}^{n} a_{ik} w_{ij}^{*} = 0\,, \quad \forall j, \forall k \\ & z_{jk}^{L} \le z_{jk} \le z_{jk}^{U}\,, \quad \forall j, \forall k\,. \end{aligned}$$

(Problem 3.1)

The primal problem is a Linear Programming (LP) problem. All the other constraints drop out since they do not involve z_{jk}, which are the variables to be solved in the primal problem. Besides z_{jk}, the Lagrange multipliers λ_{jk}^{m} for each of the constraints above is obtained. The objective function is the upper bound solution. These are inputted into the master problem, which becomes:

$$\begin{aligned} & \min_{w_{ij},\mu_B} \ \mu_B \\ \text{such that} \quad & \mu_B \ge \sum_{i=1}^{n}\sum_{k=1}^{s} a_{ik}^2 - \sum_{i=1}^{n}\sum_{j=1}^{c}\sum_{k=1}^{s} a_{ik} w_{ij} z_{jk}^{*} \\ & \quad + \sum_{j=1}^{c}\sum_{k=1}^{s} \lambda_{jk}^{m*}\left(z_{jk}^{*}\sum_{i=1}^{n} w_{ij} \right. \\ & \quad \left. - \sum_{i=1}^{n} a_{ik} w_{ij} \right)\,, \quad m = 1, M \\ & \sum_{j=1}^{c} w_{ij} = 1\,, \quad \forall i \\ & 1 \le \sum_{j=1}^{n} w_{ij} \le n - c + 1\,, \quad \forall j \\ & w_{ij} = 0 - 1\,, \quad \forall i, \forall j\,. \end{aligned}$$

(Problem 3.2)

The master problem solves for w_{ij} and μ_B, and results in a lower bound solution (i. e., the objective function). The master problem is a Mixed Integer Linear Programming (MILP) problem. The w_{ij} solutions are cycled back into the primal problem and the process is repeated until the solution converges. Thus, there is no longer a need for the variables y_{ijk}, which substantially reduces the number of variables to be solved. Also, after every solution of the master problem, where a solution set for w_{ij} is generated, an integer cut is added for subsequent iterations to prevent redundantly considering that particular solution set again. The cut is expressed as:

$$\sum_{i\in\{n|w_{ij}=1\}}^{n} w_{ij} - \sum_{i\in\{n|w_{ij}=0\}}^{n} w_{ij} \le n - 1\,. \tag{4}$$

Determining the Optimal Number of Clusters Most clustering algorithms do not contain screening functions to determine the optimal number of clusters. Yet this is important to evaluate the results of cluster analysis in a quantitative and objective fashion. On the other hand, while it is relatively easy to propose indices of cluster validity, it is difficult to incorporate these measures into clustering algorithms and appoint thresholds on which to define key decision values [18,27]. Some of the indices used to compute cluster validity include the Dunn's validity index [10], the Davis–Bouldin validity index [6], the Silhouette validation technique [43], the C index [24], the Goodman–Kruskal index [16], the Isolation index [39], the Jaccard index [25], and the Rand index [42]. We note that the optimal number of clusters occurs when the inter-cluster distance is maximized and the intra-cluster distance is minimized. We adapt the concept of a clustering balance [29], where it has been shown to have a minimum value when intra-cluster similarity is maximized and inter-cluster similarity is minimized. This provides a measure of how optimal is a certain number of clusters used for a particular clustering algorithm. We introduce the following:

$$\text{Global Center, } z_k^o = \frac{1}{n}\sum_{i=1}^{n} a_{ik} \,, \quad \forall k \tag{5}$$

Intra-cluster error sum,

$$\Lambda = \sum_{i=1}^{n}\sum_{j=1}^{c}\sum_{k=1}^{s} w_{ij} \left\| a_{ik} - z_{jk} \right\|_2^2 \tag{6}$$

Inter-cluster error sum,

$$\Gamma = \sum_{j=1}^{c}\sum_{k=1}^{s} \left\| z_{jk} - z_k^o \right\|_2^2 \,. \tag{7}$$

Jung et al. [29] proposed a clustering balance parameter, which is the α-weighted sum of the two error sums.

$$\text{Clustering Balance, } \varepsilon = \alpha\Lambda + (1-\alpha)\,\Gamma \,. \tag{8}$$

We note here that the rightful α-ratio is 0.5. There are two ways to come to this conclusion. We note that the factor α should balance the contributive weights of the two error sums to the clustering balance. At extreme cluster numbers, that is, the largest and smallest number possible, the sum of the intra-cluster and inter-cluster error sums at both cluster numbers should be balanced. In the minimal case, all the data points can be placed into a single cluster, in the case of which the inter-cluster error sum is zero and the intra-cluster error sum can be calculated with ease. In the maximal case, each data point forms its own cluster, in the case of which the intra-cluster error sum is zero and the inter-cluster error sum can be easily found. Obviously the intra-cluster error sum in the minimal case and inter-cluster error sum in the maximal case are equal, suggesting that the most appropriate weighting factor to use is in fact 0.5. The second approach uses a clustering gain parameter proposed by Jung et al. [29], which is given by:

$$\Delta = \sum_{j=1}^{c}\sum_{k=1}^{s} \left(n_j - 1\right) \left\| z_k^o - z_{jk} \right\|_2^2 \,. \tag{9}$$

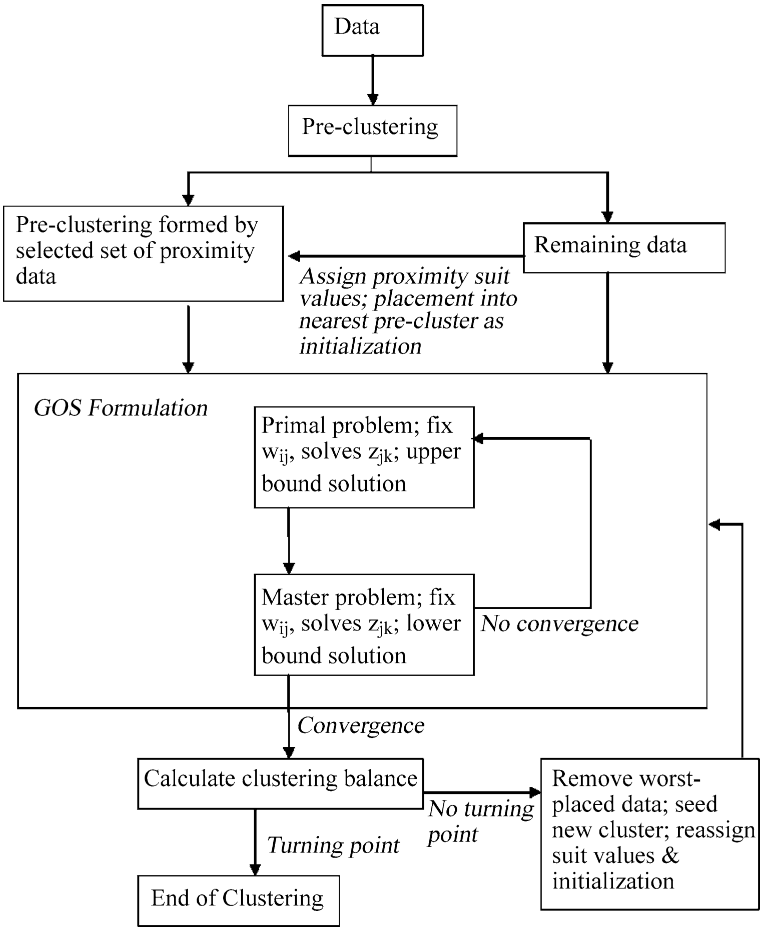
Jung et al. [29] showed the clustering gain to have a maximum value at the optimal number of clusters, and demonstrated that the sum total of the clustering gain and balance parameters is a constant. This is only shown to be only possible if the α-ratio is 0.5 [51]. These derivations suggest that for any clustering algorithm including that using the GOS algorithm, one can deduce the optimal number of clusters by performing multiple repetitions of the clustering process over a suitably large range of cluster numbers and watching for the clustering gain or clustering balance turning points.

Proposed Algorithm

The GOS formulation appears to be a suitable clustering algorithm. But for it to be effective, the formulation must be provided with a good initialization point. Also, we want to expeditiously incorporate the approach to predict the optimal number of clusters into a clustering algorithm. With these considerations in mind, we propose the following GOS clustering algorithm with enhanced data point positioning (EP_GOS_Clust).

Gene Pre-Clustering We pre-cluster the original data by proximity studies to reduce the computational demands by (i) identifying genes with very similar responses, and (ii) removing outliers deemed to be insignificant to the clustering process. To provide just adequate discriminatory characteristics, pre-clustering can be done by reducing the expression vectors into a set of representative variables or by pre-grouping genes that are close to one another by correlation or some other distance function.

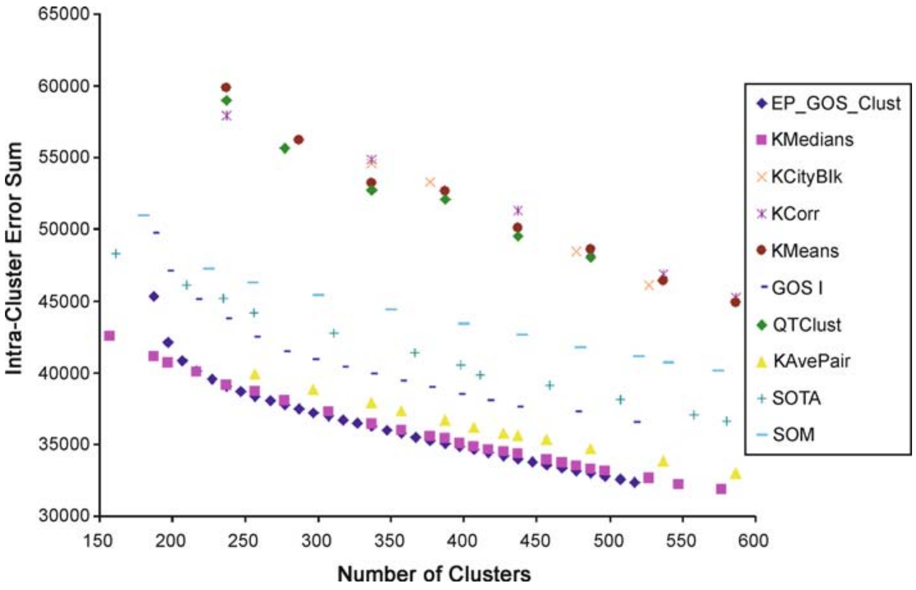
Iterative Clustering We let the initial clusters be defined by the genes pre-clustered previously, and find the distance between each of the remaining genes and these initial clusters and as a good initialization point placed these genes into the nearest cluster. For each gene, we allow its suitability in a limited number of clusters based on the proximity study. In the primal problem of the GOS algorithm, we solve for z_{jk}. These, together with the Lagrange multipliers, are used in the master problem to solve for w_{ij}. The primal gives an upper bound solution and the master a lower bound. The optimal solution is obtained when both bounds converge. Then, the worst-placed gene is removed and used as a seed for a new cluster. This gene has already been subjected to a membership search so there is no reason for it to belong to any one of the older clusters. The iterative steps are repeated and the clusters build up gradually until the optimal number is attained. Figure 1 shows a schematic of EP_GOS_Clust.


Data
Pre-clustering
Pre-clustering formed by selected set of proximity data
Remaining data
Assign proximity suit values; placement into nearest pre-cluster as initialization
GOS Formulation
Primal problem; fix w_{ij}, solves z_{jk}; upper bound solution
Master problem; fix w_{ij}, solves z_{jk}; lower bound solution
No convergence
Convergence
Calculate clustering balance
No turning point
Remove worst-placed data; seed new cluster; reassign suit values & initialization
Turning point
End of Clustering

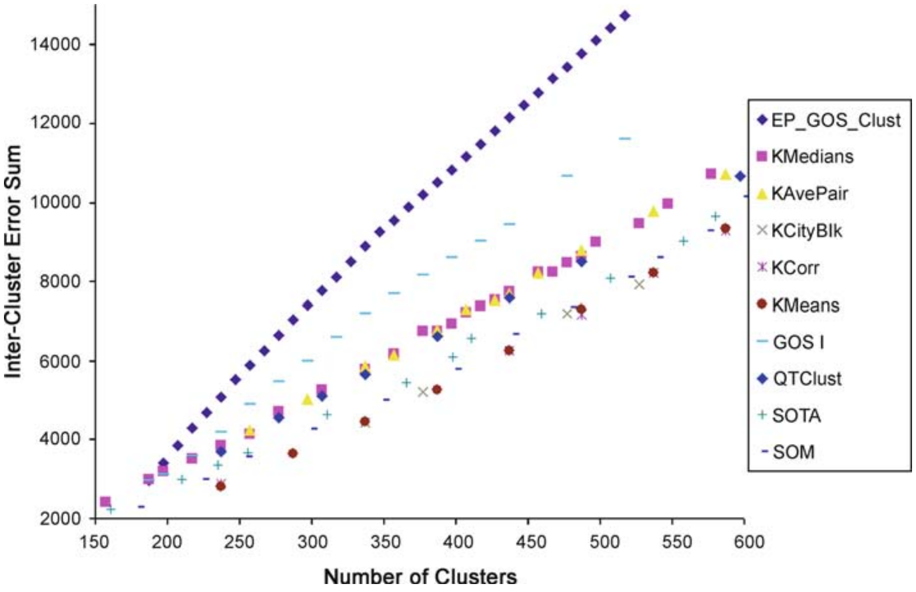
Gene Clustering: A Novel Decomposition-Based Clustering Approach, Figure 1
Schematic of EP_GOS_Clust algorithm

Gene Clustering: A Novel Decomposition-Based Clustering Approach, Table 1
Comparison of cluster correlation. The shaded row contains the results for EP_GOS_Clust and the top three performers in each column are marked with an asterisk

			Correlation coefficient			
		Optimal Cluster Number	Average	Maximum	Minimum	Standard deviation
Clustering Method	EP_GOS_Clust	237	0.617*	0.938*	0.264*	0.128*
	KMedians	445	0.615	0.937	0.197	0.134
	KCityBlk	665	0.398	0.760	-0.159	0.149
	KCorr	665	0.630*	0.931	0.239*	0.119*
	KMeans	775	0.614	0.959*	0.072	0.131
	GOS I	295	0.590	0.933	0.202	0.148
	KAvePair	452	0.567	0.909	0.156	0.141
	SOTA	540	0.604	0.925	0.378*	0.122*
	SOM	485	0.623*	0.968*	0.202	0.156



Gene Clustering: A Novel Decomposition-Based Clustering Approach, Figure 2
Intra-cluster error sum

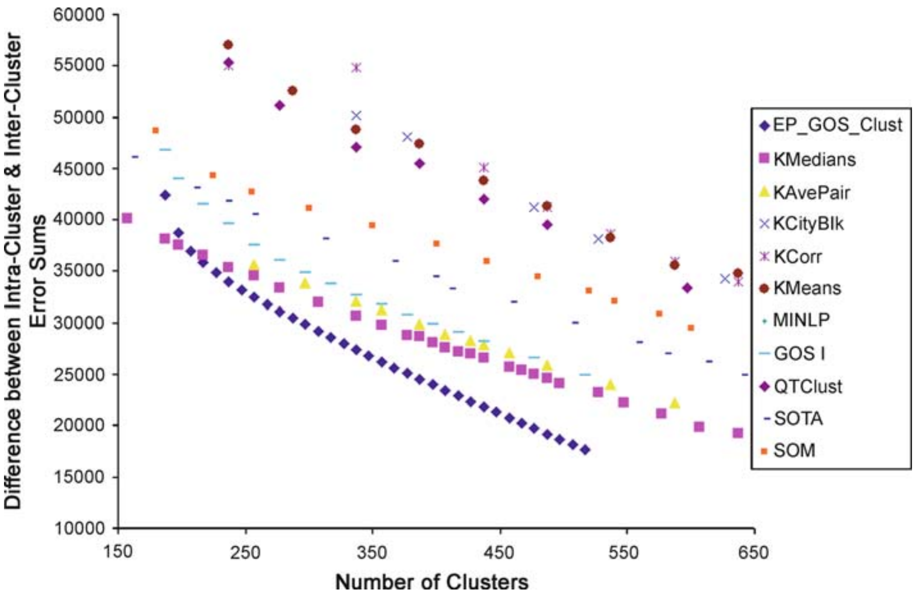


Gene Clustering: A Novel Decomposition-Based Clustering Approach, Figure 3
Inter-cluster error sum

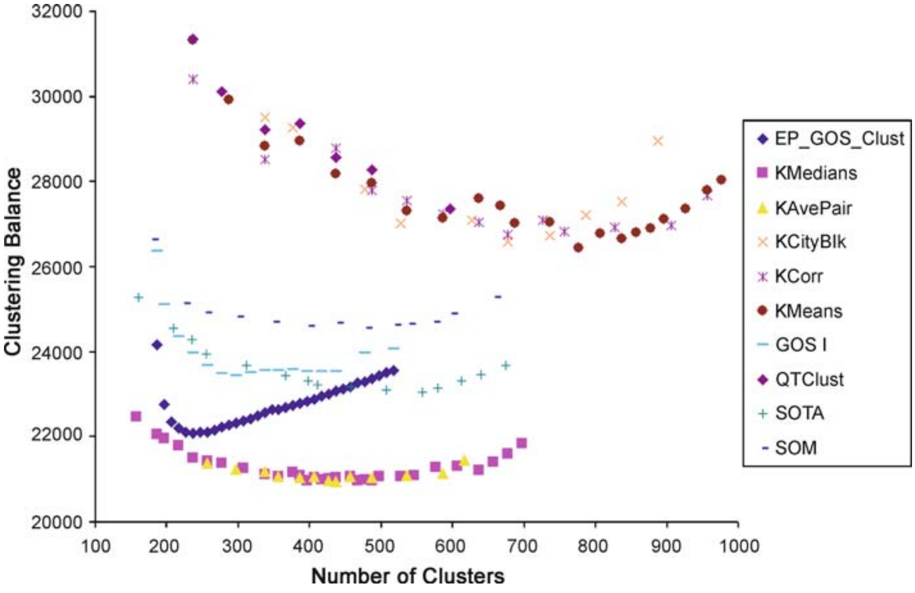
Case Study

Experimental Data

As a study, we use experimental microarray data derived from a study in the role of the Ras/protein kinase A pathway (PKA) on glucose signaling in yeast [56]. These experiments analyzed mRNA levels in samples extracted from cells at various times following stimulation by glucose or following activation of either Ras2 or Gpa2, which are small GTPases involved in the metabolic and transcriptional response of yeast cells to glucose [45]. These experiments were performed in wild type cells and cells defective in PKA activity. Clustering these microarray data has proven to be a critical



Gene Clustering: A Novel Decomposition-Based Clustering Approach, Figure 4
Error sum difference



Gene Clustering: A Novel Decomposition-Based Clustering Approach, Figure 5
Optimal cluster number

step in using the data to develop a predictive model of a topological map of the signaling network surrounding the Ras/PKA pathway [35].

Levels of RNA for each of the 6237 yeast genes in each of the RNA samples from the above experiments were measured using Affymetrix microarray chips and analyzed by the Affymetrix software. We used the Affymetrix MicroArray Suite 5.0, which analyzes the consensus of intensities of hybridization of an RNA to the collection of perfect match probes for a gene on the array, relative to the intensities of hybridization to single mismatch probes, to further determine whether

Gene Clustering: A Novel Decomposition-Based Clustering Approach, Table 2
Gene Ontology comparison. The table compares the $-\log_{10}(P)$ values of the clusters, which reflect the level of annotative richness, as well as the proportion of yeast genes that fall into biologically significant clusters. The latter is important in 'presenting' the maximal amount of relevant genetic information for follow-up work in areas such as motif recognition and regulatory network studies

		$-\log_{10}(P)$ Comparison			
		Average	Standard deviation	In clusters with $-\log_{10}(P)$ values ≥ 4	In clusters with $-\log_{10}(P)$ values ≥ 3
Clustering Method	EP_GOS_Clust	4.40*	0.37	32.82*	64.92*
	KMedians	4.27*	0.34*	30.83*	62.23*
	KCityBlk	3.69	0.49	27.53	56.68
	KCorr	4.15*	0.39	32.59*	60.08*
	KMeans	3.45	0.41	25.11	55.20
	GOS I	3.84	0.42	28.19	57.75
	KAvePair	3.77	0.48	25.18	54.43
	SOTA	3.67	0.31*	30.20	58.86
	SOM	3.94	0.35*	30.47	59.24

a signal for a specific RNA in a sample was reliable (P or present), unreliably low (A or absent), or ambiguous (M). Before clustering the array data, we filtered the data to remove unreliable data. In particular, we retained all genes for which all the time points were present (4105 genes), all the genes for which greater than 50% of the time points were present, and all the genes for which the present/absent calls exhibited a biologically relevant pattern (e. g. PAAA for the four time points in the experiment, suggesting repression of gene expression over the course of the experiment). In all, we retained 5652 genes.

Description of Comparative Study

The clustering algorithms to be compared are (a) K-Means, (b) K-Medians, (c) K-Corr, where the Pearson correlation coefficient is the distance metric, (d) K-CityBlock, where the distance metric is the city block distance, or the 'Manhattan' metric, which is akin to the north-south or east-west walking distance in a place like New York's Manhattan district, (e) K-AvePair, where the cluster metric is the average pair-wise distance between members in each cluster, (f) QTClust, (g) SOM, (h) SOTA, (i) GOS I, where genes with up to 7 different feature points are pre-clustered, initial clusters are defined by uniquely-placed genes, and each gene is placed into its nearest cluster as the initialization point, and (j) EP_GOS_Clust, for which genes are pre-clustered if they have 2 or less different feature points and can be uniquely clustered. Since the K-family approaches are sensitive to the initialization point, we run each 25 times and use only the best result.

Results and Discussion

A good clustering procedure should minimize the intra-cluster error sum and maximize the inter-cluster error sum. We look also at the difference between error sums, which is somewhat indicative of the efficacy of a particular clustering algorithm, since methods using intra-cluster error sum as the cost function would probably outperform methods using inter-cluster error sum as a performance indicator. From Fig. 2, 3 and 4, we can see that EP_GOS_Clust compares very favorably compared to the other clustering algorithms. Also, as seen from Fig. 5, EP_GOS_Clust predicts the lowest number of optimal clusters. Together with the quality of the error sum comparisons, we infer the superior 'economy' of EP_GOS_Clust in producing tighter data groupings by utilizing a lower number of clusters, as it is actually possible to achieve tight groupings by using a large number of clusters, even with an inferior clustering algorithm.

EP_GOS_Clust is also capable of uncovering strongly correlated clusters with high levels of biological coherence. Tables 1 and 2 shows that it performs consistently well when compared against the significance of cluster biological coherence uncovered by the other clustering methods. We find our clusters to exhibit good correlation and a high level of functional coherence strength across all cluster sizes, which indicates that EP_GOS_Clust shows good consistency and lack of size-bias. Also, it can be seen that EP_GOS_Clust compares very well with other clustering methods in producing highly correlated clusters, even against methods such as K-Corr that already explicitly uses correlation as a metric for clustering and the correlation hunting SOM. In addition, EP_GOS_Clust conveniently isolates errant data points and refines the existing groupings as the clustering progresses.

References

1. Adams WP, Sherali HD (1990) Linearization Strategies for a Class of Zero-One Mixed Integer Programming Problems. Oper Res 38(2):217–226
2. Beer M, Tavazoie S (2004) Predicting Gene Expression from Sequence. Cell 117:185–198
3. Bezdek JC (1981) Pattern Recognition with Fuzzy Objective Function Algorithms. Plenum Press, New York
4. Carpenter G, Grossberg S (1990) ART3: Hierarchical Search using Chemical Transmitters in Self-Organizing Patterns Recognition Architectures. Neural Netw 3:129–152
5. Claverie J (1999) Computational Methods for the Identification of Differential and Coordinated Gene Expression. Hum Mol Genet 8:1821–1832
6. Davis DL, Bouldin DW (1979) A Cluster Separation Measure. IEEE Trans Pattern Anal Mach Intell 1(4):224–227
7. Dempster AP, Laird NM, Rudin DB (1977) Maximum Likelihood from Incomplete Data via the EM Algorithm. J Royal Stat Soc B 39(1):1–38
8. Dhillon IS, Guan Y (2003) Information Theoretic Clustering of Sparse Co-Occurrence Data. In: Proceedings of the Third IEEE International Conference on Data Mining (ICDM), Melbourbe, November 2003
9. Dunn JC (1973) A Fuzzy Relative of the ISODATA Process and Its Use in Detecting Compact Well-Separated Clusters. J Cybern 3:32–57
10. Dunn JC (1974) Well Separated Clusters and Optimal Fuzzy Partitions. J Cybern 4:95–104
11. Duran MA, Odell PL (1974) Cluster Analysis: A Survey. Springer, New York
12. Floudas CA (1995) Nonlinear and Mixed-Integer Optimization: Fundamentals and Applications. Oxford University Press, Oxford
13. Floudas CA (2000) Deterministic Global Optimization: Theory, Algorithms, and Applications. Kluwer, Dordrecht
14. Floudas CA, Aggarwal A, Ciric AR (1989) Global Optimum Search for Non Convex NLP and MINLP Problems. Comp Chem Eng 13(10):1117–1132
15. Floudas CA, Akrotirianakis IG, Caratzoulas S, Meyer CA, Kallrath J (2005) Global Optimization in the 21st Century: Advances and Challenges. Comput Chem Eng 29:1185–2002
16. Goodman L, Kruskal W (1954) Measures of Associations for Cross-Validations. J Am Stat Assoc 49:732–764
17. Gower JC, Ross GJS (1969) Minimum Spanning Trees and Single-Linkage Cluster Analysis. Appl Stat 18:54–64
18. Halkidi M, Batistakis Y, Vazirgiannis M (2002) Cluster Validity Methods: Part 1. SIGMOD Rec 31(2):40–45
19. Hansen P, Jaumard B (1997) Cluster Analysis and Mathematical Programming. Math Program 79:191–215
20. Hartigan JA (1975) Clustering Algorithms. Wiley, New York
21. Hartigan JA, Wong MA (1979) Algorithm AS 136: A K-Means Clustering Algorithm. Appl Stat-J Roy St C 28:100–108
22. Herrero J, Valencia A, Dopazo J (2001) A Hierarchical Unsupervised Growing Neural Network for Clustering Gene Expression Patterns. Bioinformatics 17(2):126–136
23. Heyer LJ, Kruglyak S, Yooseph S (1999) Exploring Expression Data: Identification and Analysis of Co-Expressed Genes. Genome Res 9:1106–1115
24. Hubert L, Schultz J (1976) Quadratic Assignment as a General Data-Analysis Strategy. Br J Math Stat Psychol 29:190–241
25. Jaccard P (1912) The Distribution of Flora in the Alpine Zone. New Phytol 11:37–50
26. Jain AK, Murty MN, Flynn PJ (1999) Data Clustering: A Review. ACM Comput Surv 31(3):264–323
27. Jain AK, Dubes RC (1988) Algorithms for Clustering Data. In: Prentice-Hall Advanced Reference Series. Prentice, New Jersey.
28. Johnson RE (2001) The Role of Cluster Analysis in Assessing Comparability under the US Transfer Pricing Regulations. Bus Econ
29. Jung Y, Park H, Du D, Drake BL (2003) A Decision Criterion for the Optimal Number of Clusters in Hierarchical Clustering. J Glob Optim 25:91–111
30. Kirkpatrick S, Gelatt CD, Vecchi MP (1983) Optimization by Simulated Annealing. Science 220(4598):671–680
31. Kohonen T (1989) Self Organization and Associative Memory. In: Springer Information Science Series. Springer, New York
32. Kohonen T (1997) Self-Organizing Maps. Springer, Berlin
33. Leisch F, Weingessel A, Dimitriadou E (1998) Competitive Learning for Binary Valued Data. In: Niklasson L, Bod'en M, Ziemke T (eds) Proceedings of the 8th International Conference on Artificial Neural Networks (ICANN 98) vol 2. Springer, Skövde, pp 779–784
34. Likas A, Vlassis N, Vebeek JL (2003) The Global K-Means Clustering Algorithm. Pattern Recognit 36:451–461
35. Lin X, Floudas C, Wang Y, Broach JR (2003) Theoretical and Computational Studies of the Glucose Signaling Pathways in Yeast Using Global Gene Expression Data. Biotechnol Bioeng 84(7):864–886
36. Lukashin AV, Fuchs R (2001) Analysis of Temporal Gene Expression Profiles: Clustering by Simulated Annealing and

Determining the Optimal Number of Clusters. Bioinform 17(5):405–414

37. McQueen J (1967) Some Methods for Classification and Analysis of Multivariate Observations. In: Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, January 1966. University of California, Berkely, pp 281–297
38. Metropolis N, Rosenbluth A, Rosenbluth M, Teller A, Teller EJ (1953) Equations of state calculations by fast computing machines. J Chem Phys 21:1087
39. Pardalos PM, Boginski V, Vazakopoulos A (Co-Ed.) (2007) Data Mining in Biomedicine. Springer, Berlin
40. Pauwels EJ, Fregerix G (1999) Finding Salient Regions in Images: Non-parametric Clustering for Image Segmentation and Grouping. Comput Vis Image Underst 75:73–85
41. Pipenbacher P, Schliep A, Schneckener S, Schonhuth A, Schomburg D, Schrader R (2002) ProClust: Improved Clustering of Protein Sequences with an Extended Graph-Based Approach. Bioinform 18(Supplement 2):S182–191
42. Rand WM (1971) Objective Criteria for the Evaluation of Clustering Methods. J Am Stat Assoc 846–850
43. Rousseeuw PJ (1987) Silhouettes: A Graphical Aid to the Interpretation and Validation of Cluster Analysis. J Comp Appl Math 20:53–65
44. Ruspini EH (1969) A New Approach to Clustering. Inf Control 15:22–32
45. Schneper L, Düvel K, Broach JR (2004) Sense and Sensibility: Nutritional Response and Signal Integration in Yeast. Curr Opin Microbiol 7(6):624–630
46. Sherali HD, Desai J (2005) A Global Optimization RLT-Based Approach for Solving the Hard Clustering Problem. J Glob Optim 32(2):281–306
47. Sherali HD, Desai J (2005) A Global Optimization RLT-Based Approach for Solving the Fuzzy Clustering Approach. J Glob Optim 33(4):597–615
48. Slonim N, Atwal GS, Tkačik G, Bialek W (2005) Information Based Clustering. Proc Natl Acad Sci USA 102(51):18297–18302
49. Sokal RR, Michener CD (1958) A Statistical Method for Evaluating Systematic Relationships. Univ Kans Sci Bull 38:1409–1438
50. Sorlie T, Tibshirani R, Parker J, Hastie T, Marron JS, Nobel A, Deng S, Johnsen H, Pesich R, Geisler S, Demeter J, Perou CM, Lonning PE, Brown PO, Borresen-Dala AL, Botstein D (2003) Repeated Observations of Breast Tumor Subtypes in Independent Gene Expression Data Sets. Proc Natl Acad Sci USA 100:8418–8423
51. Tan MP, Broach JR, Floudas CA (2007) A Novel Clustering Approach and Prediction of Optimal Number of Clusters: Global Optimum Search with Enhanced Positioning. J Glob Optim 39:323–346
52. Tan MP, Broach JR, Floudas CA (2007) Evaluation of Normalization and Pre-Clustering Issues in a Novel Clustering Approach: Global Optimum Search with Enhanced Positioning. J Bioinform Comput Biol 5(4):895–913
53. Tan MP, Broach JR, Floudas CA (2007) Microarray Data Mining: A Novel Optimization-Based Iterative Clustering Approach to Uncover Biologically Coherent Structures. (submitted for publication)
54. Tishby N, Pereira F, Bialek W (1999) The Information Bottleneck Method. In: Proceedings of the 37th Annual Allerton Conference on Communication, Monticello, September 1999. Control and Computing, pp 368–377
55. Troyanskaya OG, Dolinski K, Owen AB, Altman RB, Botstein D (2003) A Bayesian Framework for Combining Heterogeneous Data Sources for Gene Function Prediction (in Saccharomyces Cerevisiae). Proc Natl Acad Sci USA 100:8348–8353
56. Wang Y, Pierce M, Schneper L, Guldal CG, Zhang X, Tavazoie S, Broach JR (2004) Ras and Gpa2 Mediate One Branch of a Redundant Glucose Signaling Pathway in Yeast. PLoS Biol 2(5):610–622
57. Wu Z, Leahy R (1993) An Optimal Graph Theoretic Approach to Data Clustering: Theory and Its Application to Image Segmentation. IEEE Trans Pattern Recognit Mach Intell 15(11):1101–1113
58. Xu R, Wunsch IID (2005) Survey of Clustering Algorithms. IEEE Trans Neural Netw 16(3):645–678
59. Zahn CT (1971) Graph Theoretical Methods for Detecting and Describing Gestalt Systems. IEEE Trans Comput C-20:68–86
60. Zhang B, Hsu M, Dayal U (1999) K-Harmonic Means – A Data Clustering Algorithm. Hewlett-Packard Research Laboratory Technical Report HPL-1999-124
61. Zhang B (2000) Generalized K-Harmonic Means: Boosting in Unsupervised Learning. Technical Report, Hewlett-Packard Research Laboratory

Generalizations of Interior Point Methods for the Linear Complementarity Problem

LAURA DI GIACOMO
Dipartimento di Statistica, Probabilità e Statistiche Applicate, Università di Roma "La Sapienza",
Rome, Italy

MSC2000: 90C33, 90C51, 65K10

Article Outline

Introduction

Some methods are reviewed to solve the resulting complementarity problem and two novel algorithms are described. The use of complementarity problems provides more flexibility to solve optimization problems, as well as a number of other advantages [10].

The existence of a general solution procedure for the linear complementarity problem (LCP) permits the incorporation of this algorithm recursively in an optimization algorithm and so avoids the use of active set strategies to handle inequality constraints and the use of second-order information on the objective function. This is often beneficial in the presence of nonconvex functions [10].

There exist many traditional approaches to solve the LCP. An algorithm was formulated early for the solution of LCPs [5,6]. Later, it was shown that if a LCP has a solution, then there exists a linear program, which, for a suitable objective function, will have an optimal solution that is also a solution to the LCP [2]. This was further generalized [7,8,9] so that for certain classes of LCPs the problem could be specified and solved as a linear program. A characterization of LCP was formulated [11] showing the equivalence of its solution to a solution of an appropriate parametric linear program with one scalar parameter.

A number of interior point algorithms to solve the LCPs have been presented, such as an interior point potential reduction algorithm [4] with P-matrices, positive semidefinite matrices and skew-symmetric matrices, an interior point algorithm which uses the affine scaling algorithm, to solve nonconvex (indefinite or negative definite) quadratic programming problems [14]. A fully polynomial-time approximation algorithm for computing a solution of the LCP with row-sufficient matrices can also be formulated [15]. This algorithm is a fully polynomial-time approximation scheme for finding an ϵ-approximate stationary point of the general LCP.

Here we shall briefly describe some particular methods and indicate two extensions of these algorithms which apply to more general matrices.

Definitions

In this section some definitions will be given and they will be used in the next sections [3].

Definition 1 Given M, an $n \times n$ matrix, and q, an n-dimensional vector. Let N be the index set of the variables, i. e., $N = 1, 2, \ldots, n$; the formulation of the LCP, $LCP(q, M)$, is then as follows:

$$Mx + q \geq 0\,, \tag{1}$$

$$x \geq 0\,, \tag{2}$$

$$x^T(Mx + q) = 0\,. \tag{3}$$

Definition 2 A matrix $M \in R^{n\times n}$ is said to be a P-matrix (P_0-matrix) if all its principal minors are positive (nonnegative). The class of such matrices is denoted P (P_0).

Definition 3 A square matrix is called a Z-matrix if its off-diagonal entries are all nonpositive. A Z-matrix which is also a P-matrix (P_0-matrix) is called a K-matrix (K_0-matrix).

Definition 4 A matrix $M \in R^{n\times n}$ is said to be column-sufficient if it satisfies the implication

$$[z_i(Mz)_i \leq 0 \text{ for all } i] \rightarrow [z_i(Mz)_i = 0 \text{ for all } i]\,. \tag{4}$$

The matrix M is called row-sufficient if its transpose is column -sufficient. If M is both column-sufficient and row-sufficient, then it is called sufficient.

Definition 5 A square matrix M is a skew-symmetric matrix if its transpose is also its negative:

$$A^T = -A\,. \tag{5}$$

Definition 6 If M is a positive definite matrix, then there exists a vector z such that

$$Mz > 0\,, \quad z > 0 \tag{6}$$

Definition 7 If M is a positive semidefinite matrix, then there exists a vector z such that

$$Mz \geq 0\,, \quad z > 0\,. \tag{7}$$

Definition 8 A potential function is

$$P(x, \Omega) = n \log(c^T x) - \sum_{j=1}^{n} \log x_j, \quad x \in \operatorname{int}(\chi_\Omega) \tag{8}$$

where $\operatorname{int}(\chi_\Omega)$ indicates the interior of the set χ_Ω which is the set of all feasible solutions of the dual.

Formulation

The aim of this section is to describe two modern implementations with interior point methods. In the first subsection an interior reduction algorithm to solve the LCP is presented, with particular matrix classes, [4], while in the following subsection an interior point potential algorithm to solve the general LCP is presented.

An Interior Point Reduction Algorithm to Solve the LCP

There exist many interior point algorithms to solve LCPs. A particularly interesting approach is an interior point potential reduction algorithm for the LCP [4]. The complementarity problem is viewed as a minimization problem, where the objective function is the product of the solution vector x and the slack vector of the inequalities y.

The objective of the algorithm formulated is to find an ϵ-complementarity solution in time bounded by a polynomial in the input size. This algorithm is formulated to solve LCP(q,M) which will have a solution, such as when the matrix M is a P-matrix. It is then extended to matrices M which are only positive semidefinite and to skew-symmetric matrices.

Consider a LCP, that is, given a rational matrix $M \in R^{n \times n}$ and a rational vector $q \in R^n$, find vectors $x, y \in R^n$ such that

$$y = Mx + q\,, \tag{9}$$

$$x, y \geq 0\,, \tag{10}$$

$$x^T y = 0\,, \tag{11}$$

which can be regarded as a quadratic programming problem

$$\text{Minimize } x^T y \tag{12}$$

$$\text{subject to } y = Mx + q \tag{13}$$

$$x, y \geq 0\,. \tag{14}$$

Given the problem Eqs. (12)–(14) the aim is to find a point with $x^T y < \epsilon$ for a given $\epsilon > 0$.

The algorithm proceeds by iteratively reducing the potential function:

$$f(x, y) = \rho \ln(x^T y) - \sum_j \ln(x_j y_j)\,. \tag{15}$$

Apply a linear scaling transformation to make the coordinates of the current point all equal to 1 and then take a gradient step in the transformed space using the gradient of the transformed potential function. The step size can be determined either by the algorithm or by line search to minimize the value of the potential function. Finally transform the solution point back to the original space.

Consider the potential function Eq. (15) under scaling of x and y, given any feasible interior point (x^0,y^0) if the matrices X and Y are diagonal matrices with the elements on the diagonal given by the values of (x^0,y^0).

Define a linear transformation of the space by

$$\bar{x} = X^{-1}x, \quad \bar{y} = Y^{-1}y\,. \tag{16}$$

and let $W = XY$, $w_j = (x_j^0)^T(y_j^0)$ so that ($w = w_1, w_2, \cdots, w_n$) and $\overline{M} = Y^{-1}MX$. Consider the transformed problem as follows:

$$\text{Minimize } \bar{x}^T W \bar{y} \tag{17}$$

$$\text{subject to } \bar{y} = \bar{M}\bar{x} + \bar{q} \tag{18}$$

$$\bar{x}, \bar{y} \geq 0\,. \tag{19}$$

Feasible solutions of the original problem are mapped into feasible solutions of the transformed problem:

$$\bar{y} = Y^{-1}(Mx + q) = \bar{M}\bar{x} + \bar{q}\,. \tag{20}$$

Assume that the current point is indeed (*e*,*e*) and the potential function has the form

$$f(\overline{x},\overline{y}) = \rho \ln(\overline{x}^T W \overline{y}) - \sum_{j=1}^{n} \ln(\overline{x}_j w_j \overline{y}_j)\,. \tag{21}$$

The gradient of f is given by

$$\nabla_x f(x,y) = \frac{\rho}{x^T W y} W y - X^{-1} e\,, \tag{22}$$

$$\nabla_y f(x,y) = \frac{\rho}{x^T W y} W x - Y^{-1} e\,, \tag{23}$$

and indicate by g the gradient vector evaluated at the current point (*e*,*e*).

Denote by $(\Delta x, \Delta y)$ the projection of $\nabla f(e,e)$ on the linear space Ω defined by $\Delta y = M \Delta x$.

Thus we define the following problem:

$$\text{Minimize } \|\Delta x - g\|^2 + \|\Delta y - g\|^2 \tag{24}$$

$$\text{subject to } \Delta y = M \Delta x\,. \tag{25}$$

It follows that [4]

$$\Delta x = (I + M^T M)^{-1} (I + M^T) g\,, \tag{26}$$

$$\Delta y = M (I + M^T M)^{-1} (I + M^T) g\,. \tag{27}$$

It is possible determine the reduction Δf in the value of f in moving from $x = y = e$ to a point of the form $\tilde{x} = e - t\Delta x$, $\tilde{y} = e - t\Delta y$, where $t > 0$. It is desired to choose t so as to achieve a reduction of at least n^{-k} for some $k > 0$, at every iteration. Since this is shown to be possible, [4], the result follows if the matrix is positive definite, positive semidefinite or skew-symmetric.

An Interior Point Potential Algorithm to Solve General LCPs

In this subsection a "condition-based" iteration complexity will be formulated regarding the solution of various LCPs. This parameter will characterize the degree of difficulty of the problem when a potential reduction algorithm is used. The condition number derived will of course depend on the data of the problem (*M*,*q*).

Consider the primal–dual potential function of a LCP as stated in Eqs: (9)–(11), for any interior feasible point, $(x,y) \in F$, and $\rho > 0$, which may be represented so:

$$\Psi(x,y) = \Psi_{n+\rho}(x,y) = (n+\rho)\ln(x^T y) - \sum_{j=1}^{n} \ln(x_j y_j). \tag{28}$$

Suppose the iterations have started from an interior feasible point (x_0,y_0), with $\Psi(x_0,y_0) = \Psi^0$ a sequence of interior feasible points can be generated $\{x^k, y^k\}, (k = 0, 1, \ldots)$ terminating at a point such that $(x^k)^T (y^k) \le \epsilon$. Such a point is found when

$$\Psi(x^k, y^k) \le \rho \ln(\epsilon) + n \ln(n) \tag{29}$$

since by the arithmetic–geometric inequality $n \ln((x^k)^T (y^k)) - \sum_{j=1}^{n} \ln(x_j y_j) \ge n \ln(n) \ge 0$.

The fact that $\Psi(x^T y) \le \Psi^0$ implies that $x^T y \le \Psi^0/\rho$ and therefore the boundedness of $\{(x,y) \in F \mid x^T y \le \Psi^0/\rho\}$ guarantees the boundedness of $\{(x,y) \in \text{int}(F) \mid x^T y \le \Psi^0\}$, where int() indicates the relative interior of its argument.

To obtain a reduction in the potential function the scaled gradient projection method may be used. The gradient vectors of the potential function with respect to x and y are

$$\nabla \Psi_x = \left(\frac{n+\rho}{x^T y}\right) y - X^{-1} e\,, \tag{30}$$

$$\nabla \Psi_y = \left(\frac{n+\rho}{x^T y}\right) x - Y^{-1} e\,. \tag{31}$$

At the kth iteration the following linear program is solved, subject to an ellipsoid constraint:

$$\text{Minimize } Z = \nabla^T \Psi_{x^k} d_x + \nabla^T \Psi_{y^k} d_y \tag{32}$$

$$\text{subject to } d_y = M d_x \tag{33}$$

$$1 > \alpha^2 \ge \|(X^k)^{-1} d_x\|^2 + \|(X^k)^{-1} d_x\|^2\,. \tag{34}$$

Denote by $(d_x^T, d_y^T)^T$ the minimal solution of Eqs. (32)–(34) and let

$$p^k = \begin{pmatrix} p_x^k \\ p_y^k \end{pmatrix} = \begin{pmatrix} \frac{n+\rho}{(x^k)^T (y^k)} X^k (y^k + M^T \pi) - e \\ \frac{n+\rho}{(x^k)^T (y^k)} Y^k (x^k - \pi) - e \end{pmatrix} \tag{35}$$

$$\pi = \left((Y^k)^2 + M (X^k)^2 M^T\right)^{-1} \left(Y^k - M X^k\right) \cdot \left(X^k y^k - \left(\frac{(x^k)^T (y^k)}{n+\rho}\right) e\right) \tag{36}$$

then there results

$$\begin{pmatrix}(X^k)^{-1}d_x\\(Y^k)^{-1}d_y\end{pmatrix} = \alpha\frac{p^k}{\|p^k\|}\,. \tag{37}$$

By the concavity of the log function and certain elementary results it can be shown [17] that

$$\Psi(x^k+d_x, y^k+d_y) - \Psi(x^k, y^k) \le \\ -\alpha\|p^k\| + \frac{\alpha^2}{2}\left(n+\rho+\frac{1}{(1-\alpha)}\right)\,. \tag{38}$$

Letting

$$\alpha = \min\left\{\frac{||p^k||}{n+\rho+2}, \frac{1}{n+\rho+2}\right\} \le \frac{1}{2} \tag{39}$$

results in

$$\Psi(x^k+d_x, y^k+d_y) - \Psi(x^k, y^k) \le \\ -\min\left\{\frac{||p^k||^2}{(2n+\rho+2)}, \frac{1}{2(n+\rho+2)}\right\}\,. \tag{40}$$

The expression for $\|p^k\|$ is indicated by (35) and can be considered the potential reduction at the kth iteration of the objective function. For any x,y let

$$g(x,y) = \frac{n+\rho}{x^T y}Xy - e \tag{41}$$

$$H(x,y) = 2I - (XM^T - Y)(Y^2 + MX^2M^T)^{-1}(MX - Y) \tag{42}$$

which is a positive semidefinite matrix. Thus

$$\|p^k\| = g^T(x^k, y^k)H(x^k, y^k)g(x^k, y^k) \tag{43}$$

which may also be indicated as $\|g(x,y)\|_H^2 = g^T(x,y)$ $H(x,y)g(x,y)$.

Define a condition number for the LCP(q,M) as

$$\gamma(M,q,\epsilon) = \inf\{||g(x,y)||_H^2 \mid x^T y \\ > \epsilon, \Psi(x,y) \le \Psi^0, (x,y) \in \text{int}(F)\}\,. \tag{44}$$

The condition number $\gamma(M,q,\epsilon)$ represents the degree of difficulty for the potential reduction algorithm in solving the LCP(q,M). The larger the condition number that results, the easier can the problem be solved. The condition number for LCPs provides a criterion to subdivide given instances of LCP(q,M) into classes and those that can be solved in polynomial time may be indicated.

Corollary 1 *An instance of a LCP(q,M) is solvable in polynomial time if $\gamma(M,q,\epsilon) > 0$ and $1/\gamma(M,q,\epsilon)$ is bounded above by a polynomial in* $\ln(1/\epsilon)$ *and n.*

This corollary is slightly different to corollary 1 in [16]. Further the following definitions are important:

$$\sum^{+}(M,q) = \{\pi \mid x^T y - q^T\pi < 0, x - \pi > 0, \\ y + M^T\pi > 0 \quad \text{for some} \quad (x,y) \in \text{int}(F)\} \tag{45}$$

Definition 9 Let G be a set of LCP(q,M) such that the following conditions are satisfied:

$$G = \{(M,q) \mid \text{int}(F) \ne \emptyset, \sum^{+}(M,q) = \emptyset\}\,. \tag{46}$$

Lemma 1 *Let $\sum^{+}(M,q)$ be empty for a LCP(q,M). Then for $\rho \ge n + \sqrt{2n}$, $\gamma(M,q,\epsilon) \ge 1$.*

Lemma 2 *Let $\{\pi \mid x^T y - q^T\pi > 0, x - \pi > 0, y + M^T\pi > 0$ for some $(x,y) \in int(F)\}$ be empty for a LCP(q,M). Then for $0 < \rho \le n - \sqrt{(2n)}$, there results $\gamma(M,q,\epsilon) \ge 1$.*

With these properties it can be shown that for many classes of matrices $\gamma(M,q,\epsilon) > 0$ or that the conditions indicated in the lemmas are satisfied, so the LCP is solvable in polynomial time.

Further, the potential reduction algorithm will solve, under general conditions, the LCP(q,M) when M is a $\mathcal{P}$-matrix and when M is a row-sufficient matrix. Thus,

Theorem 1 *Let $\Psi(x^0, y) \le O(n\ln(n))$ and M be a $\mathcal{P}$-matrix. Then the potential reduction algorithm terminates at $x^T y < \epsilon$ in $O(n^2\max\{|\lambda|/\theta(n), 1\}\ln(1/\epsilon))$ iterations and each iteration uses at most $O(n^3)$ arithmetic operations.*

The bound indicates that the algorithm is a polynomial-time algorithm if $|\lambda|/\theta(n)$ is bounded above by a polynomial in $\ln(1/\epsilon)$ and n.

Theorem 2 *Let $\rho > 0$ and be fixed. For a row-sufficient matrix M and $\{(x,y) \in F \mid \Psi(x,y) \le \Psi^0\}$ bounded, then $\gamma(M,q,\epsilon) > 0$.*

Since for the LCP(q,M) defined by this class of matrices the condition number is bounded away from zero, the potential reduction algorithm will solve this class of problems.

Methods and Applications

Depending on the algorithm proposed, any penalty function algorithm or any linear programming algorithm will ensure, given the conditions imposed on the problem, a polynomial-time solution is achieved.

Often computationally, the most efficient method is the Newton method with a penalty or a barrier parameter. However, the actual method of solution is left to the interested reader, who can refer to the original contributions, since too many problem -dependent factors are involved.

Models

The aim of this section is to treat the methods described in "Formulation" under some more general conditions.

An Interior Point Newton Method for the General LCP

This algorithm finds a Karush–Kuhn-Tucker point for a nonmonotone LCP with a primal interior point method using Newton's method with a convex barrier function, under some mild assumptions.

Consider a bounded LCP:

$$Mu + q - v = 0 \tag{47}$$

$$u, v \geq 0 \tag{48}$$

$$u^T v = 0 \tag{49}$$

and suppose that the LCP solution set $S = \{u, v | Mu + q - v = 0, u, v \geq 0, u^T v = 0\}$ is bounded above by a vector $(m_1^T, m_2^T)^T \in R^{2n}$. Define two diagonal positive matrices

$$D_1 > Diag(2m_1) \tag{50}$$

$$D_2 > Diag(2m_2) \tag{51}$$

to obtain the following LCP

$$\begin{aligned} y = D_2^{-1} v = D_2^{-1}(Mu + q) \\ = (D_2^{-1} M D_1)x + D_2^{-1} q \end{aligned} \tag{52}$$

$$\frac{1}{2} e \geq x, y \geq 0 \tag{53}$$

$$x^T y = 0 \tag{54}$$

which without loss of generality will be indicated as

$$Mx + q - y = 0 \tag{55}$$

$$x, y \geq 0 \tag{56}$$

$$x^T y = 0\,. \tag{57}$$

Assume that there exists an approximate interior point solution, as is usual with interior point methods, with variables $0 < x_i, y_i \leq \epsilon,\ \epsilon < n^{-2}\ \forall i = 1, 2, \ldots, n$ and consider the following barrier function for the optimization problem for the LCP (55)–(57).

$$\text{Minimize } \psi(x, y, \mu) = x^T y - \mu \sum_{i=1}^{n} \ln(x_i y_i) \tag{58}$$

$$\text{subject to } Mx - y + q = 0 \tag{59}$$

$$x, y < \frac{1}{2} e \tag{60}$$

$$x, y > 0 \tag{61}$$

where $e \in R^n$ is the vector of unit elements and $\beta > 0$ is an arbitrary small parameter.

To convert the optimization problem (58)–(61) into a convex programming problem, consider as a barrier parameter, which is successively reduced, then the gradient of this function is:

$$(\nabla_x \psi(x,\ y))_i = \frac{x_i y_i^2 - (\beta - \mu) y_i}{x_i y_i + \beta}\,, \tag{62}$$

$$(\nabla_x \psi(x,\ y))_i = \frac{x_i^2 y_i - (\beta - \mu) x_i}{x_i y_i + \beta}\,. \tag{63}$$

It is easy to show that if the barrier parameter at any iteration k will satisfy the following inequality

$$\mu > \frac{(x_i y_i + \beta)^2}{y_i^2 + \beta}\,, \tag{64}$$

then the Hessian matrix of the function (58) is positive definite for the conditions imposed. Thus the optimization problem (58)–(61) is a convex programming problem and it may be solved by one of the methods above, which is also suitable to a further generalization [1]. Here it will be solved as a convex quadratic programming [12]. Rewrite the optimization problem (58)–(61) as:

$$\text{Min } \psi(x,\ y,\ \mu) = x^T y - \mu \sum_{i=1}^{n} \ln(x_i y_i + \beta)\,, \tag{65}$$

$$\text{subject to} \begin{pmatrix} M & -I \\ I & 0 \\ 0 & I \\ -I & 0 \\ 0 & -I \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + b \geq 0 \,. \qquad (66)$$

Where $b^T = (q^T, 0, 0, \frac{1}{2}e^T, \frac{1}{2}e^T)$.

Indicate the constraint matrix as the matrix A of dimension $5n \times 2n$. Also, idicate with $z^T = (x^T, y^T) \in R^{2n}$.

The algorithm considered is a primal method with a log barrier function. It will follow a central path and will take small steps [12] and it can be shown that from an approximate global minimum, an exact global minimum can be simply derived [12].

Let Π denote the feasible region of Eq. (66) and denote the interior of this feasible region by int(Π), i. e., $Az > b$ by relaxing as is usual in the Interior point algorithms, the equality constraints.

Make the following assumptions:

- rank$(A) = 2n$,
- Π is compact,
- int$(\Pi) \neq \emptyset$.
- $x_i y_i > \varepsilon \quad \forall i = 1, 2, \dots, n$.

Define the potential function

$$h(z, \mu) = \psi(x, y, \mu) - \mu \sum_{i=1}^{m} \ln(a_i^T z - b_i) \,. \qquad (67)$$

The following lemmas are straight forward adaptations of the original results.

Lemma 3 *For any fixed choice of $\mu > 0$, that meets the condition (64), the function (67) is strictly convex on int(Π).*

Lemma 4 *For any fixed choice of $\mu > 0$, that meets the condition (64), the function (67) has a unique minimum.*

Let $\zeta(\mu)$ be the minimum of $h(z, \mu)$ for a fixed μ. As $\mu \to 0$ there must be an accumulation point by compactness. This point must be an approximate global minimum.

Lemma 5 *Let $\hat{z}$ be an accumulation point of $\zeta(\mu)$. As $\mu \to 0$ then $\hat{z}$ is an approximate global minimum for problem (65)–(66).*

Generalization of an Interior Point Reduction Algorithm to Solve General LCPs

The condition number for LCPs provides a criterion to subdivide given instances of LCP(q,M) into classes. These results will now be extended.

Consider a LCP(q,M) Eqs. (9)–(11) with a nonsingular coefficient matrix M, for which, moreover, (I–M) is nonsingular and the solution set of LCP(q,M) is bounded from above. This LCP can be indicated so:

$$Mu + q - v = 0 \,, \qquad (68)$$

$$u, v \geq 0 \,, \qquad (69)$$

$$u^T v = 0 \,, \qquad (70)$$

where $u, v, q \in R^n$. Suppose that the LCP solution set $S = \{u, v| \quad Mu + q - v = 0, u, v \geq 0, u^T v = 0\}$ is bounded above by a vector $(m_1^T, m_2^T)^T \in R^{2n}$.

Apply the transformation defined by Eqs. (50) and (51), so that there results

$$y = D_2^{-1} v = D_2^{-1}(Mu + q) = (D_2^{-1} M D_1) x + D_2^{-1} q \,, \qquad (71)$$

$$\frac{1}{2} e \geq x, y \geq 0 \,, \qquad (72)$$

$$x^T y = 0 \,, \qquad (73)$$

which will be indicated as

$$Mx + q - y = 0 \,, \qquad (74)$$

$$x, y \geq 0 \,, \qquad (75)$$

$$x^T y = 0 \,. \qquad (76)$$

For the potential reduction algorithm to solve general LCPs, it is required that $x > 0$ and $y > 0$.

Lemma 6 *For a nonsingular M the matrices $\hat{M} = D_1 M D_2$, $(I - \hat{M})$, $(I - XY\hat{M})$ and $(-Y + \hat{M}X)$ are all nonsingular.*

Corollary 2 *Under the conditions of Lemma 3 $(Y + MX)$ is nonsingular.*

The following additional lemma is also required.

Lemma 7 *For all LCP(q,M) with nonsingular matrices M and (I–M) transformed to the form given by Eqs. (71)–(73) so that for any feasible solution* $(x, y) \in int(F)$ *so that* $0 < X < I,\ 0 < Y < I$, *there results* $g(x, y) = \frac{n+\rho}{x^T y} Xy - e \neq 0$.

Theorem 3 *For all LCP(q,M) with nonsingular matrices M and (I–M) transformed to the form given by Eqs. (71)–(73) so that for any feasible solution* $(x, y) \in int(F)$ *there results* $0 < X < I,\ 0 < Y < I$, *the condition number for the LCP* $\gamma(M, q, \epsilon) > 0$ *for some* $\rho > 0$.

For notational simplicity assume that the transformed matrix $\hat{M}$ is indicated by M without loss of generality. $\gamma(M, q, \epsilon) = 0$ if $\|g(x, y)\|_H^2 = 0$. Assume that $\|g(x, y)\|_H^2 = 0$ and expand it in terms of its factors.

$$\begin{aligned} &2g(x, y)^T g(x, y) - g(x, y)^T \\ &[(XM^T - Y)(Y^2 + MX^2M^T)^{-1}(MX - Y)] \\ &\qquad\qquad\qquad \cdot g(x, y) = 0 \qquad (77) \end{aligned}$$

It is easy to show that this will never happen under the conditions of the theorem. Hence, for any matrix that satisfies the assumed conditions the condition number is strictly positive and so a solution to the LCP may be obtained straightforwardly by this method. This provides a partial characterization and extension of the matrix class $\mathcal{G}$ defined in [16].

Cases

Algorithms should be tested extensively for their computational efficiency on a wide series of cases, so that suitable comparisons can be made.

One hundred and forty random instances of LCPs were solved for four different sizes (30, 50, 100, 250), with three types of matrices: positive semidefinite, negative semidefinite and indefinite. In Table 1 the number of problems solved for each type of matrix with the parametric LCP algorithm [11] and with an interior point algorithm with the Newton method are indicated.

The instances with positive (semi)definite matrices are easy to solve in fact. The instances with negative (semi)definite and indefinite classes are considered hard to solve, but both algorithms have no trouble with these classes, except that the first seems to be more haphazard, rather than being subject to numerical difficulties.

Generalizations of Interior Point Methods for the Linear Complementarity Problem, Table 1
Results for 140 linear complementarity problems (LCPs) of different matrix classes and sizes

Type	PSD		NSD		INDF	
Size	PLCP	IPNM	PLCP	IPNM	PLCP	IPNM
30	6	6	12	12	28	28
50	3	3	3	3	26(3)	29
100	6	6	6	6	16	16
250	5	5	7(4)	11	15	15
Total	20	20	28(32)	32	85(88)	88

PSD positive semidefinite matrix, *NSD* negative semidefinite matrix, *INDF* indefinite matrix, *PLCP* parametric LCP algorithm, *IPMN* interior point algorithm with the Newton method.

Generalizations of Interior Point Methods for the Linear Complementarity Problem, Table 2
Timing results for 140 LCPs of different matrix classes and sizes (seconds)

Type	PSD		NSD		INDF	
Size	PLCP	IPNM	PLCP	IPNM	PLCP	IPNM
30	0.06	0.04	0.08	0.06	0.07	0.07
50	0.28	0.18	0.38	0.32	0.33	0.32
100	3.47	1.42	7.00	3.37	5.18	2.78
250	109.37	22.56	121.51	95.12	111.99	87.45

Both routines seem to be only slightly affected by the type of matrix, but the interior point algorithm with the Newton method is more efficient, as confirmed in Table 2, where the average time for solving the instances is given in seconds.

Conclusions

Interior point methods to solve the LCP are now well established and allow polynomial solutions to be obtained for such problems with suitable matrix classes. Moreover these routines can be used as a subroutine in general iterative optimization problems.

Evidently research is being actively conducted to generalize the applicable matrix classes for which solutions can be obtained in polynomial time and space.

See also

- ▶ Complementarity Algorithms in Pattern Recognition
- ▶ Mathematical Programming Methods in Supply Chain Management
- ▶ Simultaneous Estimation and Optimization of Nonlinear Problems

References

1. Boyd S, Vandenberghe L (2004) Convex Optimization. Cambridge University Press, Cambridge
2. Cottle RW, Dantzig G (1968) Complementarity pivot theory of mathematical programming. Lin Algebra Appl 1:103–125
3. Cottle RW, Pang J-S, Stone RE (1992) The Linear Complementarity Problem. Academic Press, Inc., San Diego
4. Kojima M, Megiddo N, Ye Y (1992) An Interior point potential reduction algorithm for the linear complementarity problem. Math Programm 54:267–279
5. Lemke CE (1965) Bimatrix Equilibrium Points and Mathematical Programming. Manag Sci 11:123–128
6. Lemke CE, Howson JT (1964) Equilibrium points of bimatrix games. SIAM J Appl Math 12:413–423
7. Mangasarian OL (1979) Simplified characterizations of linear complementarity problems solvable as linear programs. Math Programm 10(2):268–273
8. Mangasarian OL (1976) Linear complementarity problems solvable by a single linear program. Math Programm 10:263–270
9. Mangasarian OL (1978) Characterization of linear co complementarity problems as linear program. Math Programm 7:74–87
10. Ferris MC, Sinapiromsaran K (2000) Formulating and Solving Nonlinear Programs as Mixed Complementarity Problems. In: Nguyen VH, Striodot JJ, Tossing P (eds) Optimization. Springer, Berlin, pp 132–148
11. Patrizi G (1991) The Equivalence of an LCP to a Parametric Linear program with a Scalar Parameter. Eur J Oper Res 51:367–386
12. Vavasis S (1991) Nonlinear Optimization: Complexity Issues. Oxford University Press, Oxford
13. Ye Y (1991) An $O(n^3L)$ Potential Reduction Algorithm for linear Programming. Math Programm 50:239–258
14. Ye Y (1992) On affine scaling algorithms for nonconvex quadratic programming. Math Programm 56:285–300
15. Ye Y (1993) A fully polynomial-time approximation algorithm for computing a stationary point of the general linear complementarity problem. Math Oper Res 18:334–345
16. Ye Y, Pardalos PM (1991) A Class of Linear Complementarity Problems Solvable in Polynomial Time. Lin Algebra Appl 152:3–17
17. Ye Y (1997) Interior Point Algorithms: Theory and Analysis. Wiley, New York

Generalized Assignment Problem

O. Erhun Kundakcioglu, Saed Alizamir
Department of Industrial and Systems Engineering,
University of Florida, Gainesville, USA

MSC2000: 90-00

Article Outline

Introduction

The generalized assignment problem (GAP) seeks the minimum cost assignment of n tasks to m agents such that each task is assigned to precisely one agent subject to capacity restrictions on the agents.

The formulation of the problem is:

$$\min \quad \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij} \tag{1}$$

$$\text{subject to} \quad \sum_{j=1}^{n} a_{ij}x_{ij} \leq b_i \quad i=1,\dots,m \tag{2}$$

$$\sum_{i=1}^{m} x_{ij} = 1 \quad j=1,\dots,n \tag{3}$$

$$x_{ij} \in \{0,1\} \quad i=1,\dots,m; \quad j=1,\dots,n \tag{4}$$

where c_{ij} is the cost of assigning task j to agent i, a_{ij} is the capacity used when task j is assigned to agent i, and b_i is the available capacity of agent i. Binary variable x_{ij} equals 1 if task j is assigned to agent i, and 0 otherwise. Constraints 3 are usually referred to as the *semi-assignment constraints*.

The formulation above was first studied by Srinivasan and Thompson [80] to solve a transportation problem. The term *generalized assignment problem* for this setting was introduced by Ross and Soland [74]. This model is a generalization of previously proposed model by DeMaio and Roveda [17] where the capacity absorption is agent independent (i. e., $a_{ij} = a_j, \ \forall i$).

The classical assignment problem, which provides a one to one pairing of agents and tasks, can be solved in polynomial time [47]. However, in GAP, an agent may be assigned to multiple tasks ensuring each task is performed exactly once, and the problem is $\mathcal{NP}$-hard [28]. Even the GAP with agent-independent requirements is an $\mathcal{NP}$-hard problem [23,53].

The GAP has a wide spectrum of application areas ranging from scheduling (see [19,84]) and computer networking (see [5]) to lot sizing (see [31]) and facility location (see [7,30,74,75]). Nowakovski et al. [64] study the ROSAT space telescope scheduling where the problem is formulated as a GAP and heuristic methods are proposed. Multiperiod single-source problem (MPSSP) is reformulated as a GAP by Freling et al. [25]. Janak et al. [38] reformulate the NSF panel-assignment problem as a multiresource preference-constrained GAP. Other applications of GAP include lump sum capital rationing, loading in flexible manufacturing systems (see [45]), p-median location (see [7,75]), maximal covering location (see [42]), cell formation in group technology (see [79]), refueling nuclear reactors (see [31]), R & D planning (see [92]), and routing (see [22]). A summary of applications and assignment model components can be found in [76].

Extensions

Multiple-Resource Generalized Assignment Problem

Proposed by Gavish and Pirkul [29], multi-resource generalized assignment problem (MRGAP) is a special case of *the multi-resource weighted assignment model* that is previously studied by Ross and Zoltners [76]. In MRGAP a set of tasks has to be assigned to a set of agents in a way that permits assignment of multiple tasks to an agent subject to a set of resource constraints. This problem differs from the GAP in that, an agent consumes a variety of resources in performing the tasks assigned to it. Although most of the problems can be modeled as GAP, multiple resource constraints are frequently required in the effective modeling of real life problems. MRGAP may be encountered in large models dealing with processor and database location in distributed computer systems, trucking industry, telecommunication network design, cargo loading on ships, warehouse design and work load planning in job shops.

Gavish and Pirkul [29] introduce and compare various Lagrangian relaxations of the problem and suggest heuristic solution procedures. They design an exact algorithm by incorporating one of these heuristics along with a branch-and-bound procedure.

Mazzola and Wilcox [58] modify Gavish and Pirkul heuristic and develop a hybrid heuristic for MRGAP. Their algorithm defines a three phase heuristic which first constructs a feasible solution and then systematically tries to improve the solution. As an enhanced version of MRGAP, Janak et al. [38] study the NSF panel-assignment problem. In this setting, each task (i. e., proposal) has a specific number of agents (i. e., reviewers) assigned to it and each agent has a lower and upper bound on the number of tasks that can be done. The objective is to optimize the sum of a set of preference criteria for each agent on each task while ensuring that each agent is assigned to approximately the same number of tasks.

Multilevel Generalized Assignment Problem

The Multilevel Generalized Assignment Problem (MGAP) is first introduced by Glover et al. [31] to provide a model for the allocation of tasks in a manufacturing environment. MGAP differs from the classical GAP in that, agents can perform tasks at different efficiency levels, implying both different costs and different resource requirements. Each task must be assigned to one and only one agent at a level and each agent has limited amount of single resource. Important manufacturing problems, such as lot sizing, can be formulated as MGAP.

Laguna et al. [46] use a neighborhood structure for defining moves based on ejection chains and develop a Tabu Search (TS) algorithm for this problem. French and Wilson [26] develop two heuristic solution methods for MGAP from the solution methods for GAP. Procedures for deriving an upper bound on the solution of the problem are also described. Ceselli and Righini [11] present a branch-and-price algorithm based on decomposition of the MGAP into a master problem and a pricing sub-problem, where the former is a set-partitioning problem and the latter is a multiple-choice knapsack problem. This algorithm is the first exact method proposed in the literature for the MGAP. To provide a flexible assignment tool to the decision maker, Hajri-Gabouj [37] develops a fuzzy genetic multi-objective optimization algorithm to solve a nonlinear MGAP.

Dynamic Generalized Assignment Problem

In The Gap Model, the sequence in which the agent performs the tasks is not considered. This sequence is essential when each task is performed to meet a demand and earliness or tardiness incurs additional cost. Dynamic generalized assignment problem (DGAP) is suggested to track customer demand while assigning tasks to agents. Kogan et al. [44], for the first time, add the impact of time to the GAP model assuming that each task has a due date. They formulate the continuous-time optimal control model of the problem and derive analytical properties of the optimal behavior of such a dynamic system. Based on those properties, an efficient time-decomposition procedure is developed.

Kogan et al. [43] extend the DGAP to cope with stochastic environment and multiple agent-task relationships. They prove that this stochastic, continuous-time generalized assignment problem is strongly $\mathcal{NP}$-hard and reduce the model to a number of classical deterministic assignment problems stated at discrete time points. A pseudo-polynomial time combinatorial algorithm is developed to approximate the solution. The well-known application of such a generalization is found in the stochastic environment of the flow shop scheduling of parallel workstations and flexible manufacturing cells as well as dynamic inventory management.

Bottleneck Generalized Assignment Problem

Bottleneck generalized assignment problem (BGAP), is the min-max version of the well-known (min-sum) generalized assignment problem. In the BGAP, the maximum penalty incurred by assigning each task to an agent is minimized. Min-sum objective functions are commonly used in private sector applications, while min-max objective function can be applied to the public sector. BGAP has several important applications in scheduling and allocation problems. Mazzola and Neebe [57] propose two min-max formulations for the GAP: the Task BGAP and the Agent BGAP. Martello and Toth [56] present an exact branch-and-bound algorithm as well as approximate algorithms for BGAP. They introduce relaxations and produce, as sub-problems, min-max versions of the multiple-choice knapsack problem which can be solved in polynomial time.

Generalized Assignment Problem with Special Ordered Set

GAP is further generalized to include cases where items may be shared by a pair of adjacent knapsacks. This problem is called the generalized assignment problem with special ordered sets of type 2 (GAPS2). In other words, GAPS2 is the problem of allocating tasks to time-periods, where each task must be assigned to a time-period, or shared between two consecutive time-periods. Farias et al. [15] introduce this problem which can also be applied to production scheduling. They study the polyhedral structure of the convex hull of the feasible space, develop three families of facet-defining valid inequalities, and show that these inequalities cut off all infeasible vertices of the LP relaxation. A branch-and-cut procedure is described and facet-defining valid inequalities are used as cuts. Wilson [86] modifies and extends a heuristic algorithm developed previously for the GAP problem to solve GAPS2. He argues that, any feasible solution to GAP is a feasible solution to GAPS2, hence a heuristic algorithm for GAP can also be used as a heuristic algorithm to GAPS2. A solution produced by a GAP heuristic will be close to GAPS2 optimality if it is close to the LP relaxation bound of GAP. The heuristic uses a series of moves starting from an infeasible, but in some senses *optimal* solution and then attempts to restore feasibility with minimal degradation to the objective function value. An existing upper bound for GAP is also generalized to be used for GAPS2.

French and Wilson [27] develop an LP-based heuristic procedure to solve GAPS2. They modify a heuristic for GAP to be used for GAPS2 and show that, while Wilson [86] heuristic is straightforward for large instances of the problem, and Farias et al. [15] solve smaller instances of the problem by an exact method, their heuristic solves fairly large instances of the problem rapidly and with a consistently high degree of solution quality.

Stochastic Generalized Assignment Problem

In GAP, stochasticity may arise because the actual amount of resource needed to process the tasks by the different agents may not be known in advance or the presence or absence of individual tasks may be uncertain. In such cases, there is a set of potential tasks in which, each task may or may not require to be processed. Dyer and Frieze [20], analyze the generalized assignment problem under the assumption that all coefficients are drawn uniformly and independently from $[0, 1]$ interval. Romeijn and Piersma [72] analyze a probabilistic version of GAP as the number of tasks goes to infinity while the number of machines remains fixed. Their model is different from Dyer and Frieze [20] since it doesn't have the additional assumptions that the cost and resource requirement parameters are independent of each other and among machines. They first derive a tight condition on the probabilistic model of the parameters under which, the corresponding instances of the GAP are feasible with probability one. Next, under an additional sufficient condition, the optimal solution value of the GAP is characterized through a limiting value. It is shown that the optimal solution value, normalized by dividing by the number of tasks, converges with probability one to this limiting value. Toktas et al. [82], consider the uncertain capacities situation and derive two alternative approaches to utilize deterministic solution strategies while addressing capacity uncertainty. Albareda-Sambola et al. [1] assume that a random subset of the tasks would require to be actually processed. Tasks are interpreted as customers that may or may not require a service. They construct a convex approximation of the objective function and present three versions of an exact algorithm to solve this problem based on branch-and-bound techniques, optimality cuts, and a special purpose lower bound. An assignment of tasks can be modified once the actual demands are known. Different penalties are paid for reassigning tasks and for leaving unprocessed tasks with positive demand.

Bi-Objective Generalized Assignment Problem

Zhang and Ong [91] consider the GAP from a multi-objective point of view, and propose an LP-based heuristic to solve the bi-objective generalized assignment problem (BiGAP). In BiGAP, each assignment has two attributes that are to be considered. For example, in production planning, these attributes may be the cost and the time caused by assigning jobs to machines.

Generalized Multi-Assignment Problem

Proposed by Park Et Al. [66], the generalized multi-assignment problem (GMAP) consists of tasks that may be required to be duplicated at several agents. In other words, each task is assigned to r_j agents instead of one. Park et al. [66] develop a Lagrangian dual ascent algorithm for the GMAP that is combined with the subgradient search and used as a lower bounding scheme for the branch-and-bound procedure.

Methods

Determining whether an instance of a GAP has a feasible solution is an $\mathcal{NP}$-complete problem. Hence, unless $\mathcal{P} = \mathcal{NP}$, GAP admits no polynomial-time approximation algorithm with fixed worst-case performance ratio. Nevertheless there are numerous *approximation algorithms for GAP* in the literature which actually address a different setting where the available agent capacities are not fixed and the weighted sum of cost and available agent capacities is minimized. For some of these algorithms, a feasible solution is required as an input. For details, see [14,24,65,78]. Excluding this setting for GAP, the solution approaches proposed in the literature are either exact algorithms or heuristics. For expository surveys on the algorithms, see [10,54,60].

Exact Algorithms

The optimal solution to the GAP is obtained using an implicit enumerative procedure either via branch-and-bound scheme or branch-and-price scheme in the literature. Branch-and-bound method consists of an upper bounding procedure, a lower bounding procedure, a branching strategy, and a searching strategy. It is known that good bounding procedures are crucial steps in branch-and-bound method. Branch-and-price proceeds similar to branch-and-bound but obtains the bounds by solving the LP-relaxations of the subproblems by column generation. For more details on the valid inequalities and facets for the GAP that are used in the solution procedures, see [16,32,33,40,55,67].

The first branch-and-bound algorithm for the GAP is proposed by Ross and Soland [74]. Considering a minimization problem, they obtain the lower bounds by relaxing the capacity constraints. Martello and Toth [53] propose removing the semi-assignment constraints where the problem decomposes into a series of knapsack problems. Due to the quality of the bounds obtained, this algorithm is frequently used in the literature for benchmarking purposes. Chalmet and Gelders [12] introduce the Lagrangian relaxation of the semi-assignment constraints. Fisher et al. [23] use this technique with multipliers set by a heuristic adjustment method to obtain the lower bounds in the branch-and-bound procedure. Tighter bounds resulted from this method, significantly reduce the solution time. Guignard and Rosenwein [34] design a branch-and-bound algorithm with an enhanced Lagrangian dual ascent procedure that solves a Lagrangian dual at each enumeration node and adds a surrogate constraint to the Lagrangian relaxed model. This algorithm effectively solves generalized assignment problems with up to 500 variables. Drexl [19] presents a hybrid branch-and-bound/dynamic programming algorithm where the upper bounds are obtained via an efficient Monte Carlo type heuristic. Numerous lower bounds are proposed and their benchmark results are presented. Nauss [62] proposes a branch-and-bound algorithm where linear programming cuts, Lagrangian relaxation, and subgradient optimization are used to derive good lower bounds; feasible-solution generators with the heuristic proposed by Ronen [73] are used to derive good upper bounds. Nauss [63] uses similar branch-and-bound techniques to solve the elastic generalized assignment problem (EGAP) as well.

The first branch-and-price algorithm for the generalized assignment problem is proposed by Savelsbergh [77]. A combination of the algorithms proposed by Martello and Toth [53] and Jörnsten and Nasberg [39] is used to calculate the upper bound and the pricing problem is proved to be a knapsack problem. Barnhart et al. [6] reformulate the GAP by applying Dantzig-Wolfe decomposition to obtain a tighter LP relaxation. In order to solve the LP relaxation of the reformulated problem, pricing is done by solving a series of knapsack problems. Pigatti et al. [67] propose a branch-and-cut-and-price algorithm with a stabilization mechanism to speed up the pricing convergence. Ceselli and Righini [11] present a branch-and-price algorithm for *multilevel generalized assignment problem* that is based on decomposition and a pricing subproblem that is a multiple-choice knapsack problem.

Heuristics

Large instances of the GAP are computationally intractable due to the $\mathcal{NP}$-hardness of the problem. This calls for heuristic approaches whose benefits are twofold; they can be used as stand-alone algorithms to obtain good solutions within reasonable time and they can be used to obtain the upper bounds in exact solution methods such as the branch-and-bound procedure. Although the variety among the heuristics is high, they mostly fall into one of the following two categories: greedy heuristics and meta-heuristics.

Klastorin [41] proposes a two phase heuristic algorithm for solving the GAP. In phase one, the algorithm employs a modified subgradient algorithm to search for the optimal dual solution and in phase two, a branch-and-bound approach is used to search the neighborhood of the solution obtained in phase one.

Cattrysse et al. [9] use column generation techniques to obtain upper and lower bounds. In their method, a column represents a feasible assignment of a subset of tasks to a single agent. The master problem is formulated as a set partitioning problem. New columns are added to the master problem by solving a knapsack problem for each agent. LP relaxation of the set partitioning problem is solved by a dual ascent procedure.

Martello and Toth [54] present a greedy heuristic that assigns the jobs to machines based on a desirability factor. This factor is defined as the difference between the largest and second largest weight factors. The algorithm iteratively considers, among the unassigned jobs, the one having the highest desirability factor (or regret factor) and assigns it to its maximum profit agent. This iterative process establishes an initial solution which would be improved in the next step of the algorithm by simple interchange arguments. This heuristic can be used in a problem size reduction procedure by fixing variables to one or to zero.

Relaxation heuristics are developed by Lorena and Narciso [49] for maximization version of GAP. Feasible solutions are obtained by a subgradient search in a Lagrangian or surrogate relaxation. Six different heuristics are derived particularizing relaxation, the step size in the subgradient search and the method used to obtain the feasible solution. In a Lagrangian heuristic for GAP, Haddadi [35] introduces a substitution variable in the model which is defined as the multiplication of the original variables by their corresponding constraint coefficients. The constraints defining these new variables are then dualized in the Lagrangian relaxation of the problem and the resulted relaxation is decomposed into two subproblems: the knapsack problem and the transportation problem. Narciso and Lorena [61] use relaxation multipliers with efficient constructive heuristics to find good feasible solutions.

A breadth-first branch-and-bound algorithm is described by Haddadi and Ouzia [36] in which a standard subgradient approach is used in each node of the decision tree to solve the Lagrangian dual and to obtain an upper bound. The main contribution in this study is a new heuristic that is applied to exploit the solution of the relaxed problem by solving a GAP of smaller size.

Romeijn and Romero Morales [70] study the optimal value function from a probabilistic point of view and develop a class of greedy algorithms. A family of weight functions is designed to measure desirability of assigning each job to a machine which is used by the greedy algorithms. They derive conditions under which their algorithm is asymptotically optimal in a probabilistic sense.

Meta-heuristics are widely used to solve GAP in the literature. They are either adapted by themselves for GAP or are used in combination with other heuristics and meta-heuristics.

Variable depth search heuristic (VDSH) is a generalization of local search in which the size of the neighborhood adaptively changes to traverse a larger search space. VDSH is a two phase algorithm. In the first phase, an initial solution is developed and a lower bound is obtained. In the second phase, a nested iterative refinement process is applied to improve the quality of the solution. VDSH is introduced by Amini and Racer [2] to solve the GAP. In their method, the improvement phase consists of a two level nested loop. The major iteration creates an action set corresponding to each neighborhood structure alternative. Possible neighborhood structures for GAP are: reassign (shift) a task from one agent to another, swap the assignment of two tasks, and permute the assignment of a subset of the tasks. Then, a subsequence of operations that achieves the highest saving is obtained through performing some minor iterations. A new solution is established based on that and another major operation starts.

Amini and Racer [3] develop a hybrid heuristic (HH) around the two well known heuristics: VDSH (see [2,69]) and Heuristic GAP (HGAP) (see [54]). Previous studies show that HGAP dominates VDSH in terms of solution time, while VDSH obtains solutions of better quality within reasonable time. A computational comparison is conducted with the leading alternative heuristic approaches. Another hybrid approach is by Lourenço and Serra [52] where a MAX-MIN Ant System (MMAS) (see [81]) is applied with GRASP for the GAP.

Yagiura et al. [90] propose a variable depth search (VDS) method for GAP. Their method alternates between shift and swap moves to explore the solution space. The main aspect of their method is that, infeasible solutions are allowed to be considered. However in some of the problem instances, the feasible space is small or contains many small separate regions and the efficiency of the algorithm is affected. In another study, Yagiura et al. [89] improve VDS by incorporating branching search processes to construct the neighborhoods. They show that appropriate choices of branching strategies can improve the performance of VDS. Lin et al. [48] make further observations on the VDSH method through a series of computational experiments. They consider six greedy strategies for generating the initial feasible solution and designed several simplified strategies for the improvement phase of the method.

Osman [68] develops a hybrid heuristic which combines simulated annealing and tabu search. This algorithm takes advantage of the non-monotonic oscillation strategy of tabu search as well as the simulated annealing philosophy.

Yagiura et al. [87] propose a tabu search algorithm for GAP which utilizes an ejection chain approach. An ejection chain is an embedded neighborhood construction that compounds simple moves to create more complex and powerful moves. The chain considered in their study is a sequence of shift moves in which every two successive moves share a common agent. Searching into the infeasible region is allowed incurring a penalty proportional to the degree of infeasibility. An adaptive adjustment mechanism is incorporated for determining appropriate values of the parameters to control their influence on the problem. Yagiura et al. [88] improve their previous method by adding a path relinking approach which is a mechanism for generating new solutions by combining two or more reference solutions. The main difference of this method with the previous one is the way it generates starting solutions for ejection chains. It is shown that, by this simple change in the algorithm, the improvement in its performance is drastic.

Asahiro et al. [4] develop two parallel heuristic algorithms based on the ejection chain local search (EC) presented by Yagiura et al. [87]. One is a simple parallelization called multi-start parallel EC (MPEC) and the other one is cooperative parallel EC (CPEC). In MPEC, each search process independently explores search space while in CPEC search processes share partial information to cooperate with each other. They show that their proposed algorithms outperform EC by Yagiura [87].

Diaz and Fernandez [18], devise a flexible tabu search algorithm for GAP. Allowing the search to explore infeasible region and adaptively modification of the objective function are the sources of flexibility. The modification of the objective function is caused by the dynamic adjustment of the weight of the penalty incurred for violating feasibility. The main difference of this method with the tabu search method of Yagiura et al. [87,88] in exploring the infeasible region is that, in this method, no solution is qualitatively preferred to others in terms of its structure.

Chu and Beasley [13] develop a genetic algorithm for GAP that incorporates a fitness-unfitness pair evaluation function as a representation scheme. This algorithm uses a heuristic to improve the cost and feasibility. Feltl and Raidl [21] add new features to this algorithm including two alternative initialization heuristics, a modified selection and replacement scheme for handling infeasible solutions more appropriately and a heuristic mutation operator.

Wilson [85] proposes another algorithm for GAP which is operating in a dual sense. Instead of genetically improving a set of feasible solutions as in a regular GA, this algorithm tries to genetically restore feasibility to a set of near optimal ones. The method starts with potentially optimal but infeasible solutions and then improves feasibility while keeping optimality. When the feasible solution is obtained, the algorithm uses local search procedures to improve the solution.

Lorena et al. [50] propose a constructive genetic algorithm (CGA) for GAP. In CGA, unlike classical GA, problems are modeled as bi-objective optimization problems, which consider the evaluation of two fitness functions. The evolution process is conducted to attain the two objectives conserving schemata that survive to an adaptive threshold test. The CGA algorithm has some new features compared to GA including population formation by schemata, recombination among schemata, dynamic population, mutation in structure and the possibility of using heuristics in schemata and/or structure representation.

Lourenço and Serra [51] present two metaheuristic algorithms for GAP. One is a MIN-MAX ant system which is combined with local search and tabu search heuristics. The other one is a greedy randomized adaptive search heuristic (GRASP) studied with several neighborhoods. Both of these algorithms consist of three main steps: *(i)* constructing a solution by either a greedy randomized or an ant system approach, *(ii)* improving these initial solutions by applying local search and a tabu search, *(iii)* updating the parameters. These three steps are repeated until a stopping criterion is verified.

Monfared and Etemadi [59] use a neural network based approach for solving the GAP. They investigate four different methods to structure the energy function of the neural network: exterior penalty function, augmented Lagrangian, dual Lagrangian and interior penalty function. They show that augmented Lagrangian can produce superior results with respect to feasibility and integrality while maintaining feasibility and stability measures.

Problem generators and benchmark instances play an important role in comparing/developing new methods. Romeijn and Romero Morales [71] propose a new stochastic model for the GAP which can be used to analyze the random generators in the literature. They compare the random generators by Ross and Soland [74], Martello and Toth [53], Trick [83], Chalmet and Gelders [12], Racer and Amini [69] and conclude these random generators are not adequate because they tend to generate easier problem instances when the number of machines increases. Cario et al. [8] compare GAP instances generated under two correlation-induction strategies. Using two exact and four heuristic algorithms from the literature, they show how solutions are affected by the correlation between costs and the resource requirements.

Conclusions

This review presents the applications, extensions, and solution methods for the generalized assignment problem. As the GAP receives more attention, it will be more likely to see large sets of classical benchmark instances and comparative results on solution approaches.

References

1. Albareda-Sambola M, van der Vlerk MH, Fernandez E (2006) Exact solutions to a class of stochastic generalized assignment problems. Eur J Oper Res 173:465–487
2. Amini MM, Racer M (1994) A rigorous computational comparison of alternative solution methods for the generalized assignment problem. Manag Sci 40(7):868–890
3. Amini MM, Racer M (1995) A hybrid heuristic for the generalized assignment problem. Eur J Oper Res 87(2):343–348
4. Asahiro Y, Ishibashi M, Yamashita M (2003) Independent and cooperative parallel search methods for the generalized assignment problem. Optim Method Softw 18:129–141
5. Balachandran V (1976) An integer generalized transportation model for optimal job assignment in computer networks. Oper Res 24(4):742–759
6. Barnhart C, Johnson EL, Nemhauser GL, Savelsbergh MWP, Vance PH (1998) Branch-and-price: column generation for solving huge integer programs. Oper Res 46(3):316–329
7. Beasley JE (1993) Lagrangean heuristics for location problems. Eur J Oper Res 65:383–399
8. Cario MC, Clifford JJ, Hill RR, Yang J, Yang K, Reilly CH (2002) An investigation of the relationship between problem characteristics and algorithm performance: a case study of the gap. IIE Trans 34:297–313
9. Cattrysse DG, Salomon M, Van LN Wassenhove (1994) A set partitioning heuristic for the generalized assignment problem. Eur J Oper Res 72:167–174
10. Cattrysse DG, Van LN Wassenhove (1992) A survey of algorithms for the generalized assignment problem. Eur J Oper Res 60:260–272
11. Ceselli A, Righini G (2006) A branch-and-price algorithm for the multilevel generalized assignment problem. Oper Res 54:1172–1184
12. Chalmet L, Gelders L (1976) Lagrangean relaxation for a generalized assignment type problem. In: Advances in OR. EURO, North Holland, Amsterdam, pp 103–109
13. Chu EC, Beasley JE (1997) A genetic algorithm for the generalized assignment problem. Comput Oper Res 24:17–23
14. Cohen R, Katzir L, Raz D (2006) An efficient approximation for the generalized assignment problem. Inf Process Lett 100:162–166
15. de Farias Jr, Johnson EL, Nemhauser GL (2000) A generalized assignment problem with special ordered sets: a polyhedral approach. Math Program, Ser A 89:187–203
16. de Farias Jr, Nemhauser GL (2001) A family of inequalities for the generalized assignment polytope. Oper Res Lett 29:49–55
17. DeMaio A, Roveda C (1971) An all zero-one algorithm for a class of transportation problems. Oper Res 19:1406–1418
18. Diaz JA, Fernandez E (2001) A tabu search heuristic for the generalized assignment problem. Eur J Oper Res 132:22–38
19. Drexl A (1991) Scheduling of project networks by job assignment. Manag Sci 37:1590–1602
20. Dyer M, Frieze A (1992) Probabilistic analysis of the generalised assignment problem. Math Program 55:169–181
21. Feltl H, Raidl GR (2004) An improved hybrid genetic algorithm for the generalized assignment problem. In: SAC '04; Proceedings of the 2004 ACM symposium on Applied computing. ACM Press, New York, pp 990–995
22. Fisher ML, Jaikumar R (1981) A generalized assignment heuristic for vehicle routing. Netw 11:109–124
23. Fisher ML, Jaikumar R, van Wassenhove LN (1986) A multiplier adjustment method for the generalized assignment problem. Manag Sci 32:1095–1103
24. Fleischer L, Goemans MX, Mirrokni VS, Sviridenko M (2006) Tight approximation algorithms for maximum general assignment problems. In SODA '06: Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm. ACM Press, New York, pp 611–620
25. Freling R, Romeijn HE, Morales DR, Wagelmans APM (2003) A branch-and-price algorithm for the multiperiod single-sourcing problem. Oper Res 51(6):922–939
26. French AP, Wilson JM (2002) Heuristic solution methods for the multilevel generalized assignment problem. J Heuristics 8:143–153
27. French AP, Wilson JM (2007) An lp-based heuristic procedure for the generalized assignment problem with special ordered sets. Comput Oper Res 34:2359–2369
28. Garey MR, Johnson DS (1990) Computers and Intractability; A Guide to the Theory of NP-Completeness. Freeman, New York
29. Gavish B, Pirkul H (1991) Algorithms for the multi-resource generalized assignment problem. Manag Sci 37:695–713

30. Geoffrion AM, Graves GW (1974) Multicommodity distribution system design by benders decomposition. Manag Sci 20(5):822–844
31. Glover F, Hultz J, Klingman D (1979) Improved computer based planning techniques, part ii. Interfaces 4:17–24
32. Gottlieb ES, Rao MR (1990) $(1, k)$-configuration facets for the generalized assignment problem. Math Program 46(1):53–60
33. Gottlieb ES, Rao MR (1990) The generalized assignment problem: Valid inequalities and facets. Math Stat 46:31–52
34. Guignard M, Rosenwein MB (1989) An improved dual based algorithm for the generalized assignment problem. Oper Res 37(4):658–663
35. Haddadi S (1999) Lagrangian decomposition based heuristic for the generalized assignment problem. Inf Syst Oper Res 37:392–402
36. Haddadi S, Ouzia H (2004) Effective algorithm and heuristic for the generalized assignment problem. Eur J Oper Res 153:184–190
37. Hajri-Gabouj S (2003) A fuzzy genetic multiobjective optimization algorithm for a multilevel generalized assignment problem. IEEE Trans Syst 33:214–224
38. Janak SL, Taylor MS, Floudas CA, Burka M, Mountziaris TJ (2006) Novel and effective integer optimization approach for the nsf panel-assignment problem: a multiresource and preference-constrained generalized assignment problem. Ind Eng Chem Res 45:258–265
39. Jörnsten K, Nasberg M (1986) A new lagrangian relaxation approach to the generalized assignment problem. Eur J Oper Res 27:313–323
40. Jörnsten KO, Varbrand P (1990) Relaxation techniques and valid inequalities applied to the generalized assignment problem. Asia-P J Oper Res 7(2):172–189
41. Klastorin TD (1979) An effective subgradient algorithm for the generalized assignment problem. Comp Oper Res 6:155–164
42. Klastorin TD (1979) On the maximal covering location problem and the generalized assignment problem. Manag Sci 25(1):107–112
43. Kogan K, Khmelnitsky E, Ibaraki T (2005) Dynamic generalized assignment problems with stochastic demands and multiple agent task relationships. J Glob Optim 31:17–43
44. Kogan K, Shtub A, Levit VE (1997) Dgap – the dynamic generalized assignment problem. Ann Oper Res 69:227–239
45. Kuhn H (1995) A heuristic algorithm for the loading problem in flexible manufacturing systems. Int J Flex Manuf Syst 7:229–254
46. Laguna M, Kelly JP, Gonzfilez-Velarde JL, Glover F (1995) Tabu search for the multilevel generalized assignment problem. Eur J Oper Res 82:176–189
47. Lawler E (1976) Combinatorial Optimization: Networks and Matroids. Holt, Rinehart, Winston, New York
48. Lin BMT, Huang YS, Yu HK (2001) On the variable-depth-search heuristic for the linear-cost generalized assignment problem. Int J Comput Math 77:535–544
49. Lorena LAN, Narciso MG (1996) Relaxation heuristics for a generalized assignment problem. Eur J Oper Res 91:600–610
50. Lorena LAN, Narciso MG, Beasley JE (2003) A constructive genetic algorithm for the generalized assignment problem. J Evol Optim
51. Lourenço HR, Serra D (1998) Adaptive approach heuristics for the generalized assignment problem. Technical Report 288, Department of Economics and Business, Universitat Pompeu Fabra, Barcelona
52. Lourenço HR, Serra D (2002) Adaptive search heuristics for the generalized assignment problem. Mathw Soft Comput 9(2–3):209–234
53. Martello S, Toth P (1981) An algorithm for the generalized assignment problem. In: Brans JP (ed) Operational Research ’81, 9th IFORS Conference, North-Holland, Amsterdam, pp 589–603
54. Martello S, Toth P (1990) Knapsack Problems: Algorithms and Computer Implementations. Wiley, New York
55. Martello S, Toth P (1992) Generalized assignment problems. Lect Notes Comput Sci 650:351–369
56. Martello S, Toth P (1995) The bottleneck generalized assignment problem. Eur J Oper Res 83:621–638
57. Mazzola JB, Neebe AW (1988) Bottleneck generalized assignment problems. Eng Costs Prod Econ 14(1):61–65
58. Mazzola JB, Wilcox SP (2001) Heuristics for the multiresource generalized assignment problem. Nav Res Logist 48(6):468–483
59. Monfared MAS, Etemadi M (2006) The impact of energy function structure on solving generalized assignment problem using hopfield neural network. Eur J Oper Res 168:645–654
60. Morales DR, Romeijn HE (2005) Handbook of Combinatorial Optimization, supplement vol B. In: Du D-Z, Pardalos PM (eds) The Generalized Assignment Problem and extensions. Springer, New York, pp 259–311
61. Narciso MG, Lorena LAN (1999) Lagrangean/surrogate relaxation for generalized assignment problems. Eur J Oper Res 114:165–177
62. Nauss RM (2003) Solving the generalized assignment problem: an optimizing and heuristic approach. INFORMS J Comput 15(3):249–266
63. Nauss RM (2005) The elastic generalized assignment problem. J Oper Res Soc 55:1333–1341
64. Nowakovski J, Schwarzler W, Triesch E (1999) Using the generalized assignment problem in scheduling the rosat space telescope. Eur J Oper Res 112:531–541
65. Nutov Z, Beniaminy I, Yuster R (2006) A $(1 - 1/e)$-approximation algorithm for the generalized assignment problem. Oper Res Lett 34:283–288
66. Park JS, Lim BH, Lee Y (1998) A lagrangian dual-based branch-and-bound algorithm for the generalized multi-assignment problem. Manag Sci 44(12S):271–275
67. Pigatti A, de Aragao MP, Uchoa E (2005) Stabilized branch-and-cut-and-price for the generalized assignment prob-

lem. In: Electronic Notes in Discrete Mathematics, vol 19 of 2nd Brazilian Symposium on Graphs, Algorithms and Combinatorics, pp 385–395,

68. Osman IH (1995) Heuristics for the generalized assignment problem: simulated annealing and tabu search approaches. OR-Spektrum 17:211–225
69. Racer M, Amini MM (1994) A robust heuristic for the generalized assignment problem. Ann Oper Res 50(1):487–503
70. Romeijn HE, Morales DR (2000) A class of greedy algorithms for the generalized assignment problem. Discret Appl Math 103:209–235
71. Romeijn HE, Morales DR (2001) Generating experimental data for the generalized assignment problem. Oper Res 49(6):866–878
72. Romeijn HE, Piersma N (2000) A probabilistic feasibility and value analysis of the generalized assignment problem. J Comb Optim 4:325–355
73. Ronen D (1992) Allocation of trips to trucks operating from a single terminal. Comput Oper Res 19(5):445–451
74. Ross GT, Soland RM (1975) A branch and bound algorithm for the generalized assignment problem. Math Program 8:91–103
75. Ross GT, Soland RM (1977) Modeling facility location problems as generalized assignment problems. Manag Sci 24:345–357
76. Ross GT, Zoltners AA (1979) Weighted assignment models and their application. Manag Sci 25(7):683–696
77. Savelsbergh M (1997) A branch-and-price algorithm for the generalized assignment problem. Oper Res 45:831–841
78. Shmoys DB, Tardos E (1993) An approximation algorithm for the generalized assignment problem. Math Program 62:461–474
79. Shtub A (1989) Modelling group technology cell formation as a generalized assignment problem. Int J Prod Res 27:775–782
80. Srinivasan V, Thompson GL (1973) An algorithm for assigning uses to sources in a special class of transportation problems. Oper Res 21(1):284–295
81. Stützle T, Hoos H (1999) The Max-Min Ant System and Local Search for Combinatorial Optimization Problems. In: Voss S, Martello S, Osman IH, Roucairol C (eds) Meta-heuristics; Advances and trends in local search paradigms for optimization. Kluwer, Boston, pp 313–329
82. Toktas B, Yen JW, Zabinsky ZB (2006) Addressing capacity uncertainty in resource-constrained assignment problems. Comput Oper Res 33:724–745
83. Trick M (1992) A linear relaxation heuristic for the generalized assignment problem. Nav Res Logist 39:137–151
84. Trick MA (1994) Scheduling multiple variable-speed machines. Oper Res 42(2):234–248
85. Wilson JM (1997) A genetic algorithm for the generalised assignment problem. J Oper Res Soc 48:804–809
86. Wilson JM (2005) An algorithm for the generalized assignment problem with special ordered sets. J Heuristics 11:337–350
87. Yagiura M, Ibaraki T, Glover F (2004) An ejection chain approach for the generalized assignment problem. INFORMS J Comput 16:133–151
88. Yagiura M, Ibaraki T, Glover F (2006) A path relinking approach with ejection chains for the generalized assignment problem. Eur J Oper Res 169:548–569
89. Yagiura M, Yamaguchi T, Ibaraki T (1998) A variable depth search algorithm with branching search for the generalized assignment problem. Optim Method Softw 10:419–441
90. Yagiura M, Yamaguchi T, Ibaraki T (1999) A variable depth search algorithm for the generalized assignment problem. In: Voss S, Martello S, Osman IH, Roucairol C (eds) Meta-heuristics; Advances and Trends in Local Search paradigms for Optimization, Kluwer, Boston, pp 459–471
91. Zhang CW, Ong HL (2007) An efficient solution to biobjective generalized assignment problem. Adv Eng Softw 38:50–58
92. Zimokha VA, Rubinshtein MI (1988) R & d planning and the generalized assignment problem. Autom Remote Control 49:484–492

Generalized Benders Decomposition

GBD

CHRISTODOULOS A. FLOUDAS
Department Chemical Engineering,
Princeton University, Princeton, USA

MSC2000: 49M29, 90C11

Article Outline

Keywords

Decomposition; Duality; Global optimization

The generalized Benders decomposition, GBD, [7] is a powerful theoretical and algorithmic approach for addressing *mixed integer nonlinear optimization* problems, as well as problems that require exploitation of their inherent mathematical structure via *decomposition* principles. A comprehensive analysis of the Generalized Benders Decomposition approach along with a variety of other approaches for mixed integer nonlinear optimization problems and their applications are presented in [3].

Formulation

[7] generalized the approach proposed by [1], for exploiting the structure of mathematical programming problems stated as:

$$\begin{cases} \min\limits_{\mathbf{x},\mathbf{y}} & f(\mathbf{x},\mathbf{y}) \\ \text{s.t.} & \mathbf{h}(\mathbf{x},\mathbf{y}) = \mathbf{0} \\ & \mathbf{g}(\mathbf{x},\mathbf{y}) \leq \mathbf{0} \\ & \mathbf{x} \in \mathbf{X} \subseteq \mathbf{R}^n \\ & \mathbf{y} \in \{0,1\}\,, \end{cases}$$

under the following conditions:

C1) $\mathbf{X}$ is a nonempty, convex set and the functions

$$f\colon\ \mathbf{R}^n \times \mathbf{R}^q \to \mathbf{R},$$
$$\mathbf{g}\colon\ \mathbf{R}^n \times \mathbf{R}^q \to \mathbf{R}^p$$

are convex for each fixed $\mathbf{y} \in \mathbf{Y} = \{0, 1\}^q$, while the functions $\mathbf{h}$: $\mathbf{R}^n \times \mathbf{R}^l \to \mathbf{R}^m$ are linear for each fixed $\mathbf{y} \in \mathbf{Y} = \{0, 1\}^q$.

C2) The set

$$\mathbf{Z}_\mathbf{y} = \left\{ \mathbf{z} \in \mathbf{R}^p\colon \begin{array}{c} \mathbf{h}(\mathbf{x},\mathbf{y}) = \mathbf{0}, \\ \mathbf{g}(\mathbf{x},\mathbf{y}) \leq \mathbf{0} \\ \text{for some } \mathbf{x} \in \mathbf{X} \end{array} \right\}$$

is closed for each fixed $\mathbf{y} \in \mathbf{Y}$.

C3) For each fixed $\mathbf{y} \in \mathbf{Y} \cap \mathbf{V}$, where

$$\mathbf{V} = \left\{ \mathbf{y}\colon \begin{array}{c} \mathbf{h}(\mathbf{x},\mathbf{y}) = \mathbf{0}, \\ \mathbf{g}(\mathbf{x},\mathbf{y}) \leq \mathbf{0}, \\ \text{for some } \mathbf{x} \in \mathbf{X} \end{array} \right\}$$

one of the following two conditions holds:

i) the resulting problem has a finite solution and has an optimal multiplier vector for the equalities and inequalities.

ii) the resulting problem is unbounded, that is, its objective function value goes to $-\infty$.

It should be noted that the above stated formulation is, in fact, a subclass of the problems for which the GBD of [7] can be applied. This is due to the specification of $\mathbf{y} \in \{0, 1\}$, while [7] investigated the more general case of $\mathbf{Y} \subseteq \mathbf{R}^q$, and defined the vector of $\mathbf{y}$ variables as 'complicating' variables in the sense that if we fix $\mathbf{y}$, then:

a) the problem may be decomposed into a number of independent problems, each involving a different subvector of $\mathbf{x}$; or

b) the problem takes a well known special structure for which efficient algorithms are available; or

c) the problem becomes convex in $\mathbf{x}$ even though it is nonconvex in the joint $\mathbf{x}$-$\mathbf{y}$ domain, that is, it creates special structure.

Case a) may lead to parallel computations of the independent subproblems. Case b) allows the use of special-purpose algorithms (e. g., generalized network algorithms), while case c) invokes special structure from the convexity point of view that can be useful for the decomposition of nonconvex optimization problems. (e. g., [4]).

In the sequel, we concentrate on $\mathbf{Y} = \{0, 1\}^q$ due to our interest in (MINLP; cf. also ▶ Mixed integer nonlinear programming) models. Note also that the analysis includes the equality constraints $\mathbf{h}(\mathbf{x}, \mathbf{y}) = \mathbf{0}$ which are not treated explicitly in [7].

Condition C2) is not stringent and it is satisfied if one of the following holds (in addition to C1), C3)):

i) $\mathbf{x}$ is bounded and closed and $\mathbf{h}(\mathbf{x}, \mathbf{y})$, $\mathbf{g}(\mathbf{x}, \mathbf{y})$ are continuous on $\mathbf{x}$ for each fixed $\mathbf{y} \in \mathbf{Y}$.

ii) there exists a point $\mathbf{z}_\mathbf{y}$ such that the set

$$\left\{\mathbf{x} \in \mathbf{X}\colon\ \mathbf{h}(\mathbf{x},\mathbf{y}) = \mathbf{0},\ \mathbf{g}(\mathbf{x},\mathbf{y}) \leq \mathbf{z}_\mathbf{y}\right\}$$

is bounded and nonempty.

Note though that mere continuity of $\mathbf{h}(\mathbf{x}, \mathbf{y})$, $\mathbf{g}(\mathbf{x}, \mathbf{y})$ on $\mathbf{X}$ for each fixed $\mathbf{y} \in \mathbf{Y}$ does not imply that condition C2) is satisfied. For instance, if $\mathbf{X} = [1, \infty]$ and $h(x, y) = x + y$, $g(x, y) = -1/x$, then $z_y = (-\infty, 0)$ which is not closed since for $x \to \infty$, $g(x, y) \to -\infty$.

Note that the set $\mathbf{V}$ represents the values of $\mathbf{y}$ for which the resulting problem is feasible with respect to $\mathbf{x}$. In others words, $\mathbf{V}$ denotes the values of $\mathbf{y}$ for which there exists a feasible $\mathbf{x} \in \mathbf{X}$ for $\mathbf{h}(\mathbf{x}, \mathbf{y}) = \mathbf{0}$, $\mathbf{g}(\mathbf{x}, \mathbf{y}) \leq \mathbf{0}$. Then the intersection of $\mathbf{y}$ and $\mathbf{V}$, $\mathbf{Y} \cap \mathbf{V}$, represents the *projection* of the feasible region of the original problem onto the $\mathbf{y}$-space.

Condition C3) is satisfied if a first order constraint qualification holds for the resulting problem after fixing $\mathbf{y} \in \mathbf{Y} \cap \mathbf{V}$.

The basic idea in generalized Benders decomposition, GBD, is the generation, at each iteration, of an upper bound and a lower bound on the sought solution of the MINLP model. The upper bound results from the *primal problem*, while the lower bound results form the *master problem*. The primal problem corresponds to the original problem with fixed $\mathbf{y}$-variables (i. e., it is in the $\mathbf{x}$-space only) and its solution provides information about the upper bound and the Lagrange multipliers associated with the equality and inequality constraints. The master problem is derived via nonlinear *duality theory*, makes use of the Lagrange multipliers obtained in the primal problem, and its solution provides information about the lower bound, as well as the next set of fixed $\mathbf{y}$-variables to be used subsequently in the primal problem. As the iterations proceed, it is shown that the sequence of updated upper bounds is nonincreasing, the sequence of lower bounds is nondecreasing, and that the sequences converge in a finite number of iterations.

Theoretical Development

This Section presents the theoretical development of the generalized Benders decomposition, GBD. The primal problem is analyzed first for the feasible and infeasible cases. Subsequently, the theoretical analysis for the derivation of the master problem is presented.

The Primal Problem

The primal problem results from fixing the $\mathbf{y}$ variables to a particular 0–1 combination, which we denote as $\mathbf{y}^k$ where k stands for the iteration counter. The formulation of the primal problem $\mathrm{P}(\mathbf{y}^k)$, at iteration k is:

$$\mathrm{P}(\mathbf{y}^k) \qquad \begin{cases} \min\limits_{\mathbf{x}} & f(\mathbf{x}, \mathbf{y}^k) \\ \text{s.t.} & \mathbf{h}(\mathbf{x}, \mathbf{y}^k) = \mathbf{0} \\ & \mathbf{g}(\mathbf{x}, \mathbf{y}^k) \leq \mathbf{0} \\ & \mathbf{x} \in \mathbf{X} \subseteq \mathbf{R}^n. \end{cases}$$

Note that due to conditions C1) and C3i), the solution of the primal problem $\mathrm{P}(\mathbf{y}^k)$ is its global solution.

We will distinguish the two cases 'feasible primal' and 'infeasible primal', and describe the analysis for each case separately.

- Feasible primal.
  If the primal problem at iteration k is feasible, then its solution provides information on $\mathbf{x}^k$, $f(\mathbf{x}^k, \mathbf{y}^k)$ which is the upper bound, and the optimal multiplier vectors λ^k, μ^k for the equality and inequality constraints. Subsequently, using this information we can formulate the Lagrange function as

$$\begin{aligned} L(\mathbf{x}, \mathbf{y}, \lambda^k, \mu^k) &= f(\mathbf{x}, \mathbf{y}) \\ &+ \lambda^{k\top} \mathbf{h}(\mathbf{x}, \mathbf{y}) + \mu^{k\top} \mathbf{g}(\mathbf{x}, \mathbf{y}). \end{aligned}$$

- Infeasible primal.
  If the primal is detected by the NLP solver to be infeasible, then we consider its constraints

$$\begin{aligned} &\mathbf{h}(\mathbf{x}, \mathbf{y}^k) = \mathbf{0}, \\ &\mathbf{g}(\mathbf{x}, \mathbf{y}^k) \leq \mathbf{0}, \\ &\mathbf{x} \in \mathbf{X} \subseteq \mathbf{R}^n, \end{aligned}$$

  where the set $\mathbf{X}$, for instance, consists of lower and upper bounds on the $\mathbf{x}$ variables. To identify a feasible point we can minimize an l_1 or l_∞ sum of constraint violations. An l_1-minimization problem can be formulated as:

$$\begin{cases} \min\limits_{\mathbf{x} \in \mathbf{X}} & \sum\limits_{i=1}^{p} \alpha_i \\ \text{s.t.} & \mathbf{h}(\mathbf{x}, \mathbf{y}^k) = \mathbf{0} \\ & g_i(\mathbf{x}, \mathbf{y}^k) \leq \alpha_i, \quad i = 1, \dots, p, \\ & \alpha_i \geq 0, \quad i = 1, \dots, p, \end{cases}$$

  Note that if $\sum_{i=1}^{p} \alpha_i = 0$, then a feasible point has been determined.

Also note that by defining as

$$\alpha^{+} = \max(0, \alpha)$$

and

$$g_i^{+}(\mathbf{x}, \mathbf{y}^k) = \max\left(0, g_i(\mathbf{x}, \mathbf{y}^k)\right),$$

the l_1-minimization problem is stated as:

$$\begin{cases} \min\limits_{\mathbf{x}\in\mathbf{X}} & \sum\limits_{i=1}^{p} g_i^{+} \\ \text{s.t.} & \mathbf{h}(\mathbf{x}, \mathbf{y}^k) = \mathbf{0}. \end{cases}$$

An l_∞-minimization problem can be stated similarly as:

$$\begin{cases} \min\limits_{\mathbf{x}\in\mathbf{X}} \max\limits_{1,\dots,p} & g_i^{+}(\mathbf{x}, \mathbf{y}^k) \\ \text{s.t.} & \mathbf{h}(\mathbf{x}, \mathbf{y}^k) = \mathbf{0}. \end{cases}$$

Alternative feasibility minimization approaches aim at keeping feasibility in any constraint residual once it has been established. An l_1-minimization in these approaches takes the form:

$$\begin{cases} \min\limits_{\mathbf{x}\in\mathbf{X}} & \sum\limits_{i\in\mathbf{I}'} g_i^{+}(\mathbf{x}, \mathbf{y}^k) \\ \text{s.t.} & \mathbf{h}(\mathbf{x}, \mathbf{y}^k) = \mathbf{0} \\ & g_i(\mathbf{x}, \mathbf{y}^k) \le 0, \quad i \in \mathbf{I}, \end{cases}$$

where $\mathbf{I}$ is the set of feasible constraints and $\mathbf{I}'$ is the set of infeasible constraints. Other methods seek feasibility of the constraints one at a time while maintaining feasibility for inequalities indexed by $i \in \mathbf{I}$. This feasibility problem is formulated as:

$$\begin{cases} \min\limits_{\mathbf{x}\in\mathbf{X}} & \sum\limits_{i\in\mathbf{I}'} w_i g_i^{+}(\mathbf{x}, \mathbf{y}^k) \\ \text{s.t.} & \mathbf{h}(\mathbf{x}, \mathbf{y}^k) = \mathbf{0} \\ & g_i(\mathbf{x}, \mathbf{y}^k) \le 0, \quad i \in \mathbf{I}, \end{cases}$$

and it is solved at any one time.

To include all mentioned possibilities [2] formulated a general feasibility problem (FP) defined as:

$$(FP) \begin{cases} \min\limits_{\mathbf{x}\in\mathbf{X}} & \sum\limits_{i\in\mathbf{I}'} w_i g_i^{+}(\mathbf{x}, \mathbf{y}^k) \\ \text{s.t.} & \mathbf{h}(\mathbf{x}, \mathbf{y}^k) = \mathbf{0} \\ & g_i(\mathbf{x}, \mathbf{y}^k) \le 0, \quad i \in \mathbf{I}. \end{cases}$$

The weights w_i are nonnegative and not all are zero. Note that with $w_i = 1$, $i \in \mathbf{I}'$, we obtain the l_1-minimization. Also in the l_∞-minimization, there exist nonnegative weights at the solution such that

$$\sum w_i = 1$$

and $w_i = 0$ if $g_i(\mathbf{x}, \mathbf{y}^k)$ does not attain the maximum value.

Note that infeasibility in the primal problem is detected when a solution of (FP) is obtained for which its objective value is greater than zero.

The solution of the feasibility problem (FP) provides information on the Lagrange multipliers for the equality and inequality constraints which are denoted as $\overline{\lambda}^k$, $\overline{\mu}^k$ respectively. Then, the Lagrange function resulting from on infeasible primal problem at iteration k can be defined as:

$$\overline{L}^k(\mathbf{x}, \mathbf{y}, \overline{\lambda}^k, \overline{\mu}^k) = \overline{\lambda}^{k\top}\mathbf{h}(\mathbf{x}, \mathbf{y}) + \overline{\mu}^{k\top}\mathbf{g}(\mathbf{x}, \mathbf{y}).$$

It should be noted that two different types of Lagrange functions are defined depending on whether the primal problem is feasible or infeasible. Also, the upper bound is obtained only from the feasible primal problem.

The Master Problem

The derivation of the master problem in the GBD makes use of nonlinear duality theory, and is characterized by the following three key ideas:

i) projection onto the $\mathbf{y}$-space;
ii) dual representation of $\mathbf{V}$; and
iii) dual representation of the projection of the original problem on the $\mathbf{y}$-space.

In the sequel, the theoretical analysis involved in these three key ideas is presented.

Projection Onto the y-Space

The original problem can be written as:

$$\begin{cases} \min\limits_{\mathbf{y}} \inf\limits_{\mathbf{x}} & f(\mathbf{x}, \mathbf{y}) \\ \text{s.t.} & \mathbf{h}(\mathbf{x}, \mathbf{y}) = \mathbf{0} \\ & \mathbf{g}(\mathbf{x}, \mathbf{y}) \le \mathbf{0} \\ & \mathbf{x} \in \mathbf{X} \\ & \mathbf{y} \in \mathbf{Y} = \{0, 1\}^q, \end{cases} \qquad (1)$$

where the min operator has been written separately for $\mathbf{y}$ and $\mathbf{x}$. Note that it is infimum with respect to $\mathbf{x}$ since for given $\mathbf{y}$ the inner problem may be unbounded. Let us define $\nu(\mathbf{y})$ as:

$$\nu(\mathbf{y}) = \begin{cases} \inf\limits_{\mathbf{x}} & f(\mathbf{x}, \mathbf{y}) \\ \text{s.t.} & \mathbf{h}(\mathbf{x}, \mathbf{y}) = \mathbf{0} \\ & \mathbf{g}(\mathbf{x}, \mathbf{y}) \le \mathbf{0} \\ & \mathbf{x} \in \mathbf{X}. \end{cases} \tag{2}$$

Note that $\nu(\mathbf{y})$ is parametric in the $\mathbf{y}$ variables and therefore, from its definition corresponds to the optimal value of the original problem for fixed $\mathbf{y}$ (i. e., the primal problem $\mathrm{P}(\mathbf{y}^k)$ for $\mathbf{y} = \mathbf{y}^k$).

Let us also define the set $\mathbf{V}$ as:

$$\mathbf{V} = \left\{ \mathbf{y}: \begin{array}{c} \mathbf{h}(\mathbf{x}, \mathbf{y}) = \mathbf{0}, \\ \mathbf{g}(\mathbf{x}, \mathbf{y}) \le \mathbf{0} \\ \text{for some } \mathbf{x} \in \mathbf{X} \end{array} \right\}. \tag{3}$$

Then, problem (1) can be written as:

$$\begin{cases} \min\limits_{\mathbf{y}} & \nu(\mathbf{y}) \\ \text{s.t.} & \mathbf{y} \in \mathbf{Y} \cap \mathbf{V}, \end{cases} \tag{4}$$

where $\nu(\mathbf{y})$ and $\mathbf{V}$ are defined by (2) and (3) respectively.

Problem (4) is the projection of the original problem onto the $\mathbf{y}$-space. Note also that in (3) $\mathbf{y} \in \mathbf{Y} \cap \mathbf{V}$ since the projection needs to satisfy the feasibility considerations.

Having defined the projection problem onto the $\mathbf{y}$-space, we can now state the theoretical result of [7].

Theorem 1 (Projection)

i) If $(\boldsymbol{x}^*, \boldsymbol{y}^*)$ is optimal in the original problem, then $\boldsymbol{y}^*$ is optimal in (4).

ii) If the original problem is infeasible or has unbounded solution, then the same is true for (4) and vice versa.

Note that the difficulty in the original problem is due to the fact that $\nu(\mathbf{y})$ and $\mathbf{V}$ are known only implicitly via (2) and (3).

To overcome the aforementioned difficulty we have to introduce the dual representation of $\mathbf{V}$ and $\nu(\mathbf{y})$.

Dual of V

The dual representation of $\mathbf{V}$ will be invoked in terms of the intersection of a collection of regions that contain it, and it is described in the following theorem, due to [7].

Theorem 2 (Dual of V) *Assuming conditions C1) and C2), a point $\boldsymbol{y} \in \boldsymbol{Y}$ belongs also to the set $\boldsymbol{V}$ if and only if it satisfies the (finite) system:*

$$0 \ge \inf \bar{L}(\mathbf{x}, \mathbf{y}, \bar{\lambda}, \bar{\mu}), \quad \forall \bar{\lambda}, \bar{\mu} \in \Lambda,$$
$$\Lambda = \left\{ \bar{\lambda} \in \mathbf{R}^m, \bar{\mu} \in \mathbf{R}^p : \ \bar{\mu} \ge \mathbf{0}, \ \sum_{i=1}^{p} \bar{\mu}_i. = 1 \right\} \tag{5}$$

Note that (5) is an infinite system because it has to be satisfied for all $\bar{\lambda}, \bar{\mu} \in \Lambda$. The dual representation of the set $\mathbf{V}$ needs to be invoked so as to generate a collection of regions that contain it (i. e., system (5) and system (5) corresponds to the set of constraints that have to be incorporated for the case of infeasible primal problems.

Note that if the primal is infeasible and we make use of the l_1-minimization of the type:

$$\begin{cases} \min\limits_{\mathbf{x}} & \sum\limits_{i \in \mathbf{I}} \alpha_i \\ \text{s.t.} & \mathbf{h}(\mathbf{x}, \mathbf{y}^k) = \mathbf{0} \\ & g_i(\mathbf{x}, \mathbf{y}^k) \le \alpha_i, \quad i \in \mathbf{I}, \\ & \mathbf{x} \in \mathbf{X}, \end{cases} \tag{6}$$

then the set Λ results from a straightforward application of the KKT gradient conditions to problem (6) with respect to α_i.

Having introduced the dual representation of the set $\mathbf{V}$, which corresponds to infeasible primal problems, we can now invoke the dual representation of $\nu(\mathbf{y})$.

Dual Representation of $N(\mathbf{y})$

The dual representation of $\nu(\mathbf{y})$ will be in terms of the pointwise infimum of a collection of functions that support it, and it is described in the following theorem, due to [7].

Theorem 3 (Dual of $\nu(\mathbf{y})$)

$$\begin{aligned}\nu_{\mathbf{y}} &= \begin{cases} \inf\limits_{\mathbf{x}} & f(\mathbf{x},\mathbf{y}) \\ s.t. & \mathbf{h}(\mathbf{x},\mathbf{y}) = \mathbf{0} \\ & g(\mathbf{x},\mathbf{y}) \le \mathbf{0} \\ & \mathbf{x} \in \mathbf{X} \end{cases} \\ &= \sup_{\lambda,\mu\ge 0} \inf_{\mathbf{x}\in\mathbf{X}} L(\mathbf{x},\mathbf{y},\lambda,\mu), \\ &\quad \forall \mathbf{y} \in \mathbf{Y} \cap \mathbf{V}, \\ &\quad L(\mathbf{x},\mathbf{y},\lambda,\mu) \\ &= f(\mathbf{x},\mathbf{y}) + \lambda^{\top}\mathbf{h}(\mathbf{x},\mathbf{y}) + \mu^{\top}\mathbf{g}(\mathbf{x},\mathbf{y}). \end{aligned} \tag{7}$$

The equality of $\nu(\mathbf{y})$ and its dual is due to having the strong duality theorem satisfied because of conditions C1), C2) and C3).

Substituting (7) for $\nu(\mathbf{y})$ and (5) for $\mathbf{y} \in \mathbf{Y} \cap \mathbf{V}$ into problem (4), (which is equivalent to (1)), we obtain:

$$\begin{cases} \min\limits_{\mathbf{y}\in\mathbf{Y}} \sup\limits_{\lambda,\mu\ge 0} \inf\limits_{\mathbf{x}\in\mathbf{X}} & L(\mathbf{x},\mathbf{y},\lambda,\mu) \\ \text{s.t.} & 0 \ge \inf\limits_{\mathbf{x}\in\mathbf{X}} \overline{L}(\mathbf{x},\mathbf{y},\overline{\lambda},\overline{\mu}). \end{cases}$$

Using the definition of supremum as the lowest upper bound and introducing a scalar μ_B we obtain:

$$(M)\ \begin{cases} \min\limits_{\mathbf{y}\in\mathbf{Y},\mu_B} & \mu_B \\ \text{s.t.} & \mu_B \ge \inf\limits_{\mathbf{x}\in\mathbf{X}} L(\mathbf{x},\mathbf{y},\lambda,\mu), \\ & \quad \forall\lambda, \forall\mu \ge 0, \\ & 0 \ge \inf\limits_{\mathbf{x}\in\mathbf{X}} \overline{L}(\mathbf{x},\mathbf{y},\overline{\lambda},\overline{\mu}), \\ & \quad \forall \left(\overline{\lambda},\overline{\mu}\right) \in \Lambda, \end{cases}$$

where

$$\begin{aligned} & L(\mathbf{x},\mathbf{y},\lambda,\mu) = f(\mathbf{x},\mathbf{y}) \\ & \quad + \lambda^{\top}\mathbf{h}(\mathbf{x},\mathbf{y}) + \mu^{\top}\mathbf{g}(\mathbf{x},\mathbf{y}), \\ & \overline{L}(\mathbf{x},\mathbf{y},\overline{\lambda},\overline{\mu}) = \overline{\lambda}^{\top}\mathbf{h}(\mathbf{x},\mathbf{y}) + \overline{\mu}^{\top}\mathbf{g}(\mathbf{x},\mathbf{y}), \end{aligned}$$

which is called the *master problem*.

If we assume that the optimum solution of $\nu(\mathbf{y})$ in (2) is bounded for all $\mathbf{y} \in \mathbf{Y} \cap \mathbf{V}$, then we can replace the infimum with a minimum. Subsequently, the master problem will be as follows:

$$\begin{cases} \min\limits_{\mathbf{y}\in\mathbf{Y},\mu_B} & \mu_B \\ \text{s.t.} & \mu_B \ge \min\limits_{\mathbf{x}\in\mathbf{X}} L(\mathbf{x},\mathbf{y},\lambda,\mu), \\ & \quad \forall\lambda,\mu \ge 0, \\ & 0 \ge \min\limits_{\mathbf{x}\in\mathbf{X}} \overline{L}(\mathbf{x},\mathbf{y},\overline{\lambda},\overline{\mu}), \\ & \quad \forall \left(\overline{\lambda},\overline{\mu}\right) \in \Lambda, \end{cases}$$

where $L(\mathbf{x}, \mathbf{y}, \lambda, \mu)$ and $\overline{L}(\mathbf{x},\mathbf{y},\overline{\lambda},\overline{\mu})$ are defined as before.

Note that the master problem involves, an infinite number of constraints and hence we would need to consider a relaxation of the master (e. g., by dropping a number of constraints) which will represent a lower bound on the original problem. Note also that the master problem features an outer optimization problem with respect to $\mathbf{y} \in Y$ and inner optimization problems with respect to $\mathbf{x}$ which are in fact parametric in $\mathbf{y}$. It is this outer-inner nature that makes the solution of even a relaxed master problem difficult.

The inner minimization problems

$$\begin{aligned} & \min_{\mathbf{x}\in\mathbf{X}} L(\mathbf{x},\mathbf{y},\lambda,\mu), \quad \forall\lambda,\ \forall\mu \ge 0, \\ & \min_{\mathbf{x}\in\mathbf{X}} \overline{L}(\mathbf{x},\mathbf{y},\overline{\lambda},\overline{\mu}), \quad \forall \left(\overline{\lambda},\overline{\mu}\right) \in \Lambda, \end{aligned}$$

are functions of $\mathbf{y}$ and can be interpreted as support functions of $\nu(\mathbf{y})$. ($\xi(\mathbf{y})$ is a support function of $\nu(\mathbf{y})$ at point $\mathbf{y_o}$ if and only if $\xi(\mathbf{y}) = \nu(\mathbf{y})$ and $\xi(\mathbf{y}) \le \nu(\mathbf{y})$, $\forall \mathbf{y} \ne \mathbf{y_o}$.) If the support functions are linear in $\mathbf{y}$, then the master problem approximates $\nu(\mathbf{y})$ by tangent hyperplanes and we can conclude that $\nu(\mathbf{y})$ is convex in $\mathbf{y}$. Note that $\nu(\mathbf{y})$ can be convex in $\mathbf{y}$ even though the original problem is nonconvex in the joint $\mathbf{x}$-$\mathbf{y}$ space (see [5]).

In the sequel, we will define the aforementioned minimization problems in terms of the notion of support functions, that is:

$$\begin{aligned} & \xi(\mathbf{y};\lambda,\mu) = \min_{\mathbf{x}\in\mathbf{X}} L(\mathbf{x},\mathbf{y},\lambda,\mu), \\ & \forall\lambda, \quad \forall\mu \ge 0, \\ & \overline{\xi}(\mathbf{y};\overline{\lambda},\overline{\mu}) = \min_{\mathbf{x}\in\mathbf{X}} \overline{L}(\mathbf{x},\mathbf{y},\overline{\lambda},\overline{\mu}), \\ & \forall \left(\overline{\lambda},\overline{\mu}\right) \in \Lambda. \end{aligned}$$

Algorithmic Development

In the previous Section we discussed the primal and master problem for the GBD. We have the primal problem being a (linear or) nonlinear programming, NLP, problem that can be solved via available local NLP solvers (e. g., MINOS 5.3). The master problem, however, consists of outer and inner optimization problems, and approaches towards attaining its solution are discussed in the following.

How to Solve the Master Problem

The master problem has as constraints the two inner optimization problems (i. e., for the case of feasible primal and infeasible primal problems) which however need to be considered for all λ and all $\mu \geq 0$ (i.e feasible primal) and all $(\overline{\lambda}, \overline{\mu}) \in \Lambda$ (i. e., infeasible). This implies that the master problem has a very large number of constraints.

The most natural approach for solving the master problem is *relaxation* [7]. The basic idea in the relaxation approach consists of the following:

i) ignore all but a few of the constraints that correspond to the inner optimization problems (e. g., consider the inner optimization problems for specific or fixed multipliers (λ^1, μ^1) or $(\overline{\lambda}^1, \overline{\mu}^1)$);

ii) solve the relaxed master problem and check whether the resulting solution satisfies all of the ignored constraints. If not, then generate and add to the relaxed master problem one or more of the violated constraints and solve the new relaxed master problem again;

iii) continue until a relaxed master problem satisfies all of the ignored constraints, which implies that an optimal solution at the master problem has been obtained or until a termination criterion indicates that a solution of acceptable accuracy has been found.

General Algorithmic Statement of GBD

Assuming that the problem has a finite optimal value, [7] stated the general algorithm for GBD listed below.

Note that a feasible initial primal is needed in Step 1. However, this does not restrict the GBD since it is possible to start with an infeasible primal problem. In this case, after detecting that the primal is infeasible, Step 3b is applied in which a support function $\overline{\xi}$ is employed.

Note that Step 1 could be altered, that is instead of solving the primal problem we could solve a continuous relaxation of the original problem in which the $\mathbf{y}$ variables are treated as continuous bounded by zero and one:

$$\begin{cases} \min\limits_{\mathbf{x},\mathbf{y}} & f(\mathbf{x},\mathbf{y}) \\ \text{s.t.} & \mathbf{h}(\mathbf{x},\mathbf{y}) = \mathbf{0} \\ & \mathbf{g}(\mathbf{x},\mathbf{y}) \leq \mathbf{0} \\ & \mathbf{x} \in \mathbf{X} \\ & \mathbf{0} \leq \mathbf{y} \leq \mathbf{1}. \end{cases} \tag{8}$$

If the solution of (8) is integral, then we terminate. If there exist fractional values of the $\mathbf{y}$ variables, then these can be rounded to the closest integer values and subsequently these can be used as the starting $\mathbf{y}^1$ vector with the possibility of the resulting primal problem being feasible or infeasible.

Note also that in Step 1, Step 3a and Step 3b a rather important assumption is made, that is we can find the support functions ξ and $\overline{\xi}$ for the given values of the multiplier vectors (λ, μ) and $(\overline{\lambda}, \overline{\mu})$. The determination of these support functions can not be achieved in general since these are parametric functions of $\mathbf{y}$ and result from the solution of the inner optimization problems.

Their determination in the general case requires a global optimization approach as the one proposed by [5,6]. There exist however, a number of special cases for which the support functions can be obtained explicitly as functions of the $\mathbf{y}$ variables. We will discuss these special cases in the next Section. If however, it is not possible to obtain explicitly expressions of the support functions in terms of the $\mathbf{y}$ variables, then assumptions need to be introduced for their calculation. These assumptions, as well as the resulting variants of GBD will be discussed in the next Section. The point to note here is that the validity of lower bounds with these variants of GBD will be limited by the imposed assumptions.

Note that the relaxed master problem (see Step 2) in the first iteration will have as a constraint one support function that corresponds to feasible primal and will be of the form:

$$\begin{cases} \min\limits_{\mathbf{y}\in\mathbf{Y},\mu_B} & \mu_B \\ \text{s.t.} & \mu_B \geq \xi(\mathbf{y}; \lambda^1, \mu^1). \end{cases} \tag{9}$$

1 Let an initial point $\mathbf{y}^1 \in \mathbf{Y} \cap \mathbf{V}$ (i.e., by fixing $\mathbf{y} = \mathbf{y}^1$, we have a feasible primal). Solve the resulting primal problem $P(\mathbf{y}^1)$ and obtain an optimal primal solution $\mathbf{x}^1$ and optimal multipliers; vectors λ^1, μ^1. Assume that you can find, somehow, the support function $\xi(\mathbf{y}; \lambda^1, \mu^1)$ for the obtained multipliers λ^1, μ^1. Set the counters $k = 1$ for feasible and $l = 1$ for infeasible and the current upper bound UBD $= v(\mathbf{y}^1)$. Select the convergence tolerance $\epsilon \geq 0$.

2 Solve the relaxed master problem:

$$(\mathrm{RM}) \begin{cases} \min\limits_{\mathbf{y}\in\mathbf{Y},\mu_B} & \mu_B \\ \text{s.t.} & \mu_B \geq \xi(\mathbf{y}; \lambda^k, \mu^k), \\ & \quad k = 1, \dots, K, \\ & 0 \geq \xi(\mathbf{y} : \overline{\lambda}^l, \overline{\mu}^l), \\ & \quad l = 1, \dots, \Lambda. \end{cases}$$

Let $(\hat{\mathbf{y}}, \hat{\mu}_B)$ be an optimal solution of the above relaxed master problem. $\hat{\mu}_B$ is a lower bound on the original problem, that is the current lower bound is LBD $= \hat{\mu}_B$. If UBD $-$ LBD $\leq \epsilon$, then terminate.

3 Solve the primal problem for $\mathbf{y} = \hat{\mathbf{y}}$, that is the problem $P(\hat{\mathbf{y}})$. Then we distinguish two cases: feasible and infeasible primal:

3a Feasible Primal $P(\hat{\mathbf{y}})$.
The primal has $v(\hat{\mathbf{y}})$ finite with an optimal solution $\hat{\mathbf{x}}$ and optimal multiplier vectors $\hat{\lambda}, \hat{\mu}$. Update the upper bound UBD $= \min\{\text{UBD}, v(\hat{\mathbf{y}})\}$. If UBD $-$ LBD $\leq \epsilon$, then terminate. Otherwise, set $k = k + 1$, $\lambda^k = \hat{\lambda}$, and $\mu^k = \hat{\mu}$. Return to Step 2, assuming we can somehow determine the support function $\xi(\mathbf{y}; \lambda^{k+1}, \mu^{k+1})$.

3b Infeasible Primal $P(\hat{\mathbf{y}})$.
The primal does not have a feasible solution for $\mathbf{y} = \hat{\mathbf{y}}$. Solve a feasibility problem (e.g., then l_1-minimization) to determine the multiplier vectors $\overline{\hat{\lambda}}, \overline{\hat{\mu}}$ of the feasibility problem. Set $l = l + 1, \overline{\lambda}^l = \overline{\hat{\lambda}}$, and $\overline{\mu}^l = \overline{\hat{\mu}}$. Return to Step 2, assuming we can somehow determine the support function $\xi(\mathbf{y}; \overline{\lambda}^{l+1}, \overline{\mu}^{l+1})$.

In the second iteration, if the primal is feasible and (λ^2, μ^2) are its optimal multiplier vectors, then the relaxed master problem will feature two constraints and will be of the form:

$$\begin{cases} \min\limits_{\mathbf{y}\in\mathbf{Y},\mu_B} & \mu_B \\ \text{s.t.} & \mu_B \geq \xi(\mathbf{y}; \lambda^1, \mu^1) \\ & \mu_B \geq \xi(\mathbf{y}; \lambda^2, \mu^2). \end{cases} \tag{10}$$

Note that in this case the relaxed master problem (10), will have a solution that is greater or equal to the solution of (9). This is due to having the additional constraint. Therefore, we can see that the sequence of lower bounds that is created from the solution of the relaxed master problems is nondecreasing. A similar argument holds true in the case of having infeasible primal in the second iteration.

Note that since the upper bounds are produced by fixing the $\mathbf{y}$ variables to different 0–1 combinations, there is no reason for the upper bounds to satisfy any monotonicity property. If we consider however the updated upper bounds (i. e., UBD $= \min_k v(\mathbf{y}^k)$), then the sequence for the updated upper bounds is monotonically nonincreasing since by their definition we always keep the best (least) upper bound.

The termination criterion for GBD is based on the difference between the updated upper bound and the current lower bound. If this difference is less than or equal to a prespecified tolerance $\varepsilon \geq 0$ then we terminate. Note though that if we introduce in the relaxed master integer cuts that exclude the previously found 0–1 combinations then the termination criterion can be met by having found an infeasible master problem (i. e., there is no 0–1 combination that makes it feasible).

Finite Convergence of GBD

[7] proved finite convergence of the GBD algorithm which is as follows:

Theorem 4 (Finite convergence) *If C1), C2) and C3) hold and* $\mathbf{Y}$ *is a discrete set, then the GBD algorithm terminates in a finite number of iterations for any given* $\epsilon > 0$ *and even for* $\epsilon = 0$.

Variants of GBD

In the previous Section we discussed the general algorithmic statement of GBd and pointed out a key assumption made with respect to the calculation of the support functions $\xi(\mathbf{y};\lambda,\mu)$ and $\bar{\xi}(\mathbf{y};\bar{\lambda},\bar{\mu})$ from the feasible and infeasible primal problems respectively. In this section, we will discuss a number of *variants of GBD* that result from addressing the calculation of the aforementioned support functions either rigorously for special cases or making assumptions that may not provide valid lower bounds in the general case.

Variant 1 of GBD: V1-GBD

This variant of GBD is based on the following assumption that was denoted by [7] as Property (P):

Theorem 5 (Property (P)) *For every λ and $\mu \geq 0$, the infimum of $L(\boldsymbol{x}, \boldsymbol{y}, \lambda, \mu)$ with respect to $\boldsymbol{x} \in \boldsymbol{X}$ (i. e., the support $\xi(\boldsymbol{y};\lambda, \mu)$) can be taken independently of $\boldsymbol{y}$ so that the support function $\xi(\boldsymbol{y};\lambda, \mu)$ can be obtained explicitly with little or no more effort than is required to evaluate it at a single value of $\boldsymbol{y}$. Similarly, the support function $\bar{\xi}(\mathbf{y};\bar{\lambda},\bar{\mu})$, $(\bar{\lambda},\bar{\mu}) \in \Lambda$ can be obtained explicitly.*

[7] identified the following two important classes of problems where Property (P) holds:

- Class 1: f, $\mathbf{h}$, $\mathbf{g}$ are linearly separable in $\mathbf{x}$ and $\mathbf{y}$.
- Class 2: Variable factor programming.

In class-1 problems, we have

$$\begin{aligned} f(\mathbf{x},\mathbf{y}) =& f_1(\mathbf{x}) + f_2(\mathbf{y}),\\ \mathbf{h}(\mathbf{x},\mathbf{y}) =& \mathbf{h}_1(\mathbf{x}) + \mathbf{h}_2(\mathbf{y}),\\ \mathbf{g}(\mathbf{x},\mathbf{y}) =& \mathbf{g}_1(\mathbf{x}) + \mathbf{g}_2(\mathbf{y}). \end{aligned}$$

In class-2 problems, we have

$$\begin{aligned} f(\mathbf{x},\mathbf{y}) &= -\sum_i f_i(\mathbf{x}^i)\mathbf{y}_i,\\ \mathbf{g}(\mathbf{x},\mathbf{y})_j &= \sum_i \mathbf{x}^i\mathbf{y}_i - c. \end{aligned}$$

In [8] problems, we have

$$\begin{aligned} f(\mathbf{x},\mathbf{y}) &= \sum_k\sum_i f_i(\mathbf{x}_i(k))\mathbf{y}_i + \sum_i g_i(\mathbf{y}_i),\\ \mathbf{g}(\mathbf{x},\mathbf{y})_j &= -\sum_i \mathbf{x}_i(k)\mathbf{y}_i - L(k). \end{aligned}$$

In the sequel, we will discuss the v1-GBD for class-1 problems since this by itself defines an interesting mathematical structure for which other algorithms (e. g., outer approximation) has been developed.

V1-GBD Under Separability

Under the *separability assumption*, the support functions $\xi(\mathbf{y};\lambda^k,\mu^k)$ and $\bar{\xi}(\mathbf{y};\bar{\lambda}^l,\bar{\mu}^l)$ can be obtained as explicit functions of $\mathbf{y}$ since:

$$\begin{aligned} &\xi(\mathbf{y};\lambda^k,\mu^k) = \min_{\mathbf{x}\in\mathbf{X}}\ L(\mathbf{x},\mathbf{y},\lambda^k\mu^k)\\ &= \min_{\mathbf{x}\in\mathbf{X}}\{f(\mathbf{x},\mathbf{y})) + \lambda^{k\top}\mathbf{h}(\mathbf{x},\mathbf{y}) + \mu^{k\top}\mathbf{g}(\mathbf{x},\mathbf{y})\}\\ &= \min_{\mathbf{x}\in\mathbf{X}}\{f_1(\mathbf{x}) + f_2(\mathbf{y})\\ &\quad + \lambda^{k\top}(h_1(\mathbf{x}) + h_2(\mathbf{y})) + \mu^{k\top}(\mathbf{g}_1(\mathbf{x}) + \mathbf{g}_2(\mathbf{y}))\}\\ &= f_2(\mathbf{y}) + \lambda^{k\top}\mathbf{h}_2(\mathbf{y}) + \mu^{k\top}g_2(\mathbf{x})\\ &\quad + \min_{\mathbf{x}\in\mathbf{X}}[f_1(\mathbf{x}) + \lambda^{k\top}h_1(\mathbf{x}) + \mu^{k\top}\mathbf{g}_1(\mathbf{x})]. \end{aligned}$$

Note that due to separability we end up with an explicit function of $\mathbf{y}$ and a problem only in $\mathbf{x}$ that can be solved independently.

Similarly, the support function $\bar{\xi}(\mathbf{y};\bar{\lambda}^l,\bar{\mu}^l)$ is

$$\begin{aligned} &\bar{\xi}(\mathbf{y};\bar{\lambda}^l,\bar{\mu}^l) = \min_{\mathbf{x}\in\mathbf{X}} \bar{L}(\mathbf{x},\mathbf{y},\bar{\lambda}^l,\bar{\mu}^l)\\ &= \min_{\mathbf{x}\in\mathbf{X}}\{\bar{\lambda}^{l\top}\mathbf{h}(\mathbf{x},\mathbf{y}) + \bar{\mu}^{l\top}\mathbf{g}(\mathbf{x},\mathbf{y})\}\\ &= \min_{\mathbf{x}\in\mathbf{X}}\{\bar{\lambda}^{l\top}(h_1(\mathbf{x},\mathbf{y}) + h_2(\mathbf{x},\mathbf{y}))\\ &\quad + \bar{\mu}^{l\top}\left(g_1(\mathbf{x},\mathbf{y}) + g_2(\mathbf{x},\mathbf{y})\right)\}\\ &= \bar{\lambda}^{l\top}h_2(\mathbf{y}) + \bar{\mu}^{l\top}g_2(\mathbf{y})\\ &\quad + \min_{\mathbf{x}\in\mathbf{X}}\left[\bar{\lambda}^{l\top}h_1(\mathbf{x}) + \bar{\mu}^{l\top}g_1(\mathbf{x})\right]. \end{aligned}$$

Note that to solve the independent problems in $\mathbf{x}$, we need to know the multiplier vectors (λ^k, μ^k) and $(\bar{\lambda}^l,\bar{\mu}^l)$ from feasible and infeasible primal problems respectively.

Under the separability assumption, the primal problem for fixed $\mathbf{y} = \mathbf{y}^k$ takes the form

$$\begin{cases} \min\limits_{\mathbf{x}\in\mathbf{X}} & f_1(\mathbf{x}) + f_2(\mathbf{y}^k)\\ \text{s.t.} & h_1(\mathbf{x}) = -h_2(\mathbf{y}^k)\\ & g_1(\mathbf{x}) \leq -g_2(\mathbf{y}^k). \end{cases}$$

Now, we can state the algorithmic procedure for the v1-GBD under the separability assumption.

Note that if in addition to the separability of $\mathbf{x}$ and $\mathbf{y}$, we assume that $\mathbf{y}$ participates linearly (i. e., conditions

```
1  Let an initial point $\mathbf{y}^1 \in \mathbf{Y} \cap \mathbf{V}$. Solve the primal $P(\mathbf{y}^1)$ and obtain an optimal solution $\mathbf{x}^1$, and multiplier vectors $\lambda^1, \mu^1$. Set the counters $k = 1, l = 1$, and UBD $= v(\mathbf{y}^1)$. Select the convergence tolerance $\epsilon \geq 0$.
2  Solve the relaxed master problem
   $$\left\{\begin{array}{ll} \min\limits_{\mathbf{y}\in Y,\mu_B} & \mu_B \\ \text{s.t.} & \mu_B \geq f_2(\mathbf{y}) + \lambda^{k\top}\mathbf{h}_2(\mathbf{y}) \\ & \quad +\mu^{k\top}\mathbf{g}_2(\mathbf{y}) + L_1^k, \\ & \quad k = 1, \dots, K, \\ & 0 \geq \mu_B\overline{\lambda}^{l\top} h_2\mathbf{y} + \overline{\mu}^{l\top} g_2(\mathbf{y}) + L_1^l, \\ & \quad l = 1, \dots, \Lambda, \end{array}\right.$$
   where
   $$L_1^k = \min_{\mathbf{x}\in\mathbf{X}}\{f_1(\mathbf{x}) + \lambda^{k\top}\mathbf{h}_1(\mathbf{x}) + \mu^{k\top}\mathbf{g}_1(\mathbf{x})\},$$
   $$\overline{L}_1^k = \min_{\mathbf{x}\in\mathbf{X}}\{f_1(\mathbf{x}) + \overline{\lambda}^{l\top}\mathbf{h}_1(\mathbf{x}) + \overline{\mu}^{l\top}\mathbf{g}_1(\mathbf{x})\}$$
   are solutions of the above stated independent problems.
   Let $(\hat{\mathbf{y}}, \hat{\mu}_B)$ be an optional solution. $\hat{\mu}_B$ is a lower bound, that is LBD $= \hat{\mu}_B$. If UBD $-$ LBD $\leq \epsilon$, then terminate.
3  As in GBD.
```

Algorithm for v1-GBD

for outer approximation algorithm), then we have

$$\begin{aligned} f_2(\mathbf{y}) &= c^\top \mathbf{y}, \\ \mathbf{h}_2(\mathbf{y}) &= A\mathbf{y}, \\ \mathbf{g}_2(\mathbf{y}) &= B\mathbf{y}, \end{aligned}$$

in which case the relaxed master problem of Step 2 of v1-GBD will be a linear 0–1 programming problem with an additional scalar μ_B, which can be solved with available solvers (e. g., CPLEX, ZOOM, SCICONIC).

If the $\mathbf{y}$ variables participate separably but in a nonlinear way, then the relaxed master problem is of 0–1 nonlinear programming type.

Note that due to the strong duality theorem we do not need to solve the problems for $L_1^k, \overline{L}_1^l$ since their optimum solutions are identical to the ones of the corresponding feasible and infeasible primal problems with respect to $\mathbf{x}$ respectively.

Variant 2 of GBD: V2-GBD

This variant of GBD is based on the assumption that we can use the optimal solution x^k of the primal problem $P(y^k)$ along with the multiplier vectors for the determination of the support function $\xi(\mathbf{y};\lambda^k, \mu^k)$.

Similarly, we assume that we can use the optimal solution of the feasibility problem (if the primal is infeasible) for the determination of the support function $\xi(\mathbf{y}; \lambda^k, \mu^k)$.

The aforementioned assumption fixes the $\mathbf{x}$ vector to the optimal value obtained from its corresponding primal problem, and therefore eliminates the inner optimization problems that define the support functions. It should be noted that fixing $\mathbf{x}$ to the solution of the corresponding primal problem may not necessarily produce valid support functions in the sense that there would be no theoretical guarantee for obtaining lower bounds can be claimed in general.

The v2-GBD algorithm can be stated as follows:

```
1  Let an initial point $\mathbf{y}^1 \in \mathbf{Y} \cap \mathbf{V}$.
   Solve the primal problem $P(\mathbf{y}^1)$ and obtain an optimal solution $\mathbf{x}^1$ and multiplier vectors $\lambda^1$, $\mu^1$. Set the counters $k = 1$, $l = 1$, and UBD $= v(y^1)$. Select the convergence tolerance $\epsilon \geq 0$.
2  Solve the relaxed master problem:
   $$\left\{\begin{array}{ll} \min\limits_{\mathbf{y}\in\mathbf{Y}\mu_B} & \mu_B \\ \text{s.t.} & \mu_B \geq L(\mathbf{x}^k, \mathbf{y}, \lambda^k, \mu^k), \\ & \quad k = 1, \dots, K, \\ & 0 \geq \overline{L}(\mathbf{x}^l, \mathbf{y}, \overline{\lambda}^l, \overline{\mu}^l), \\ & \quad l = 1, \dots, \Lambda, \end{array}\right.$$
   $$\begin{aligned} & L(\mathbf{x}^k, \mathbf{y}, \lambda^k, \mu^k) \\ &= f(\mathbf{x}^k, \mathbf{y}) + \lambda^{k\top}\mathbf{h}(\mathbf{x}^k, \mathbf{y}) + \mu^{k\top}\mathbf{g}(\mathbf{x}^k, \mathbf{y}), \\ & \overline{L}(\overline{\mathbf{x}}^l, \mathbf{y}, \overline{\lambda}^l, \overline{\mu}^k) \\ &= \overline{\lambda}^{k\top}\mathbf{h}(\mathbf{x}^l, \mathbf{y}) + \overline{\mu}^{k\top}\mathbf{g}(\mathbf{x}^l, \mathbf{y}) \end{aligned}$$
   are the Lagrange functions evaluated at the optimal solution $x^k$ of the primal problem.
   Let $(\hat{\mathbf{y}}, \hat{\mu}_B)$ be an optimal solution. $\hat{\mu}_B$ is a lower bound, that is LBD $= \hat{\mu}_B$. If UBD $-$ LBD $\leq \epsilon$, then terminate.
3  As in GBD.
```

Algorithm for v2-GBD

Note that since $\mathbf{y} \in \mathbf{Y} = \{0 - 1\}$, the master problem is a 0–1 programming problem with one scalar variable μ_B. If the $\mathbf{y}$ variables participate linearly, then it is a 0–1 linear problem which can be solved with standard branch and bound algorithms. In such a case, we can introduce integer cuts of the form:

$$\sum_{i\in B} y_i - \sum_{i\in NB} y_i \le |B| - 1,$$

where $B = \{i: y_i = 1\}$, $NB = \{i: y_i = 0\}$, $|B|$ is the cardinality of B, which eliminate the already found 0–1 combinations. If we employ such a scheme, then an alternative termination criterion is that of having infeasible relaxed master problems. This of course implies that all 0–1 combinations have been considered.

It is of considerable interest to identify the conditions which if satisfied make the assumption in v2-GBD a valid one. The assumption in a somewhat different restated form is that:

$$\begin{aligned}
&\xi(\mathbf{y}; \lambda^k, \mu^k) = \min_{\mathbf{x}\in\mathbf{X}} L(\mathbf{x}, \mathbf{y}, \lambda^k, \mu^k)\\
&\quad\ge L(\mathbf{x}^k, \mathbf{y}, \lambda^k, \mu^k), \quad k = 1, \dots, K,\\
&\overline{\xi}(\mathbf{y}; \overline{\lambda}^l, \mu^l) = \min_{\mathbf{x}\in\mathbf{X}} \overline{L}(\mathbf{x}, \mathbf{y}, \overline{\lambda}^l, \overline{\mu}^l)\\
&\quad\ge \overline{L}(\mathbf{x}^l, \mathbf{y}, \overline{\lambda}^l, \overline{\mu}^l), \quad l = 1, \dots, \Lambda,
\end{aligned}$$

that is, we assume that the Lagrange function evaluated at the solution of the corresponding primal are valid underestimators of the inner optimization problems with respect to $\mathbf{x} \in \mathbf{X}$.

Due to condition C1) the Lagrange functions $L(\mathbf{x}, \mathbf{y}, \lambda^k, \mu^k)$, $\overline{L}(\mathbf{x}, \mathbf{y}, \overline{\lambda}^l, \overline{\mu}^l)$ are convex in $\mathbf{x}$ for each fixed $\mathbf{y}$ since they are linear combinations of convex functions ix $\mathbf{x}$.

$L(\mathbf{x}, \mathbf{y}, \lambda^k, \mu^k)$, $\overline{L}(\overline{\mathbf{x}}^l, \mathbf{y}, \overline{\lambda}^l, \overline{\mu}^l)$ represent local linearizations around the points $\mathbf{x}^k$ and $\overline{\mathbf{x}}^k$ of the support functions $\xi(\mathbf{y}; \lambda^k, \mu^k)$, $\overline{\xi}(\mathbf{y}; \overline{\lambda}^l, \mu^l)$ respectively. Therefore, the aforementioned assumption is valid if the projected problem $\nu(\mathbf{y})$ is convex in $\mathbf{y}$. If however, the projected problem $\nu(\mathbf{y})$ is convex in $\mathbf{y}$. If however, the projected problem $\nu(\mathbf{y})$ is nonconvex, then the assumption does not hold, and the algorithm may terminate at a local (not global) solution or even at a nonstationary point.

Note that in the above analysis we did not assume that $\mathbf{Y} = \{0, 1\}^l$, and hence the argument is applicable even when the $\mathbf{y}$-variables are continuous.

It is also very interesting to examine the validity of the assumption made in v2-GBD under the conditions of separability of $\mathbf{x}$ and $\mathbf{y}$ and linearity in $\mathbf{y}$ (i. e., OA conditions). In this case we have:

$$\begin{aligned}
f(\mathbf{x}, \mathbf{y}) &= \mathbf{c}^\top \mathbf{y} + f_1(\mathbf{x}),\\
\mathbf{h}(\mathbf{x}, \mathbf{y}) &= A\mathbf{y} + \mathbf{h}_1(\mathbf{x}),\\
\mathbf{g}(\mathbf{x}, \mathbf{y}) &= B\mathbf{y} + \mathbf{g}_1(\mathbf{x}).
\end{aligned}$$

Then, the support function for feasible primal becomes

$$\begin{aligned}
&\xi(\mathbf{y}; \lambda^k, \mu^k) = \mathbf{c}^\top \mathbf{y} + \lambda^{k\top}(A\mathbf{y})\\
&\quad + \mu^{k\top}(B\mathbf{y}) + \min_{\mathbf{x}\in\mathbf{X}} f_1(\mathbf{x}) + \lambda^{k\top}\mathbf{h}_1(\mathbf{x}) + \mu^{k\top}\mathbf{g}_1(\mathbf{x}),
\end{aligned}$$

which is linear in $\mathbf{y}$ and hence convex in $\mathbf{y}$. Note also that since we fix $\mathbf{x} = \mathbf{x}^k$, the $\min_{\mathbf{x}\in\mathbf{X}}$ is in fact an evaluation at $\mathbf{x}^k$. Similarly the case for $\overline{\xi}(\mathbf{y}; \overline{\lambda}^k, \mu^k)$ can be analyzed.

Therefore, the assumption in v2-GBD holds true if separability and linearity hold which covers also the case of linear 0–1 $\mathbf{y}$ variables. This way under conditions C1), C2), C3) the v2-GBD determined the global solution for separability in $\mathbf{x}$ and $\mathbf{y}$ and linearity in $\mathbf{y}$ problems.

Variant 3 of GBD: V3-GBD

This variant was proposed in [4] and denoted as *global optimum search*, GOS, and was applied to continuous as well as 0–1 set $\mathbf{Y}$. It uses the same assumption as the one in v2-GBD but in addition assumes that:

i) $\mathbf{f}(\mathbf{x}, \mathbf{y})$, $\mathbf{g}(\mathbf{x}, \mathbf{y})$ are convex functions in $\mathbf{y}$ for every fixed $\mathbf{x}$; and
ii) $\mathbf{h}(\mathbf{x}, \mathbf{y})$ are linear functions in $\mathbf{y}$ for every $\mathbf{x}$.

This additional assumption was made so as to create special structure not only in the primal but also in the relaxed master problem. The type of special structure in the relaxed master problem has to do with its convexity characteristics.

The basic idea in GOS is to select the $\mathbf{x}$ and $\mathbf{y}$ variables in a such a way that the primal and the relaxed master problem of the v2-GBD satisfy the appropriate convexity requirements and hence attain their respective global solutions.

We will discuss v3-GBD first under the separability of $\mathbf{x}$ and $\mathbf{y}$ and then for the general case.

V3-GBD Under Separability

Under the separability assumption we have:

$$\begin{aligned} f(\mathbf{x},\mathbf{y}) &= f_1(\mathbf{x}) + f_2(\mathbf{y}),\\ \mathbf{h}(\mathbf{x},\mathbf{y}) &= \mathbf{h}_1(\mathbf{x}) + \mathbf{h}_2(\mathbf{y}),\\ \mathbf{g}(\mathbf{x},\mathbf{y}) &= \mathbf{g}_1(\mathbf{x}) + \mathbf{g}_2(\mathbf{y}). \end{aligned}$$

The additional assumption that makes v3-GBD different from v2-GBD implies that

i) $\mathbf{f}_2(\mathbf{y})$, $\mathbf{g}_2(\mathbf{y})$ are convex in $\mathbf{y}$; and
ii) $\mathbf{h}_2(\mathbf{y})$ are linear in $\mathbf{y}$.

Then, the relaxed master problem will be:

$$\begin{cases} \min\limits_{\mathbf{y},\mu_B} & \mu_B\\ \text{s.t.} & \mu_B \geq f_2(\mathbf{y}) + \lambda^{k\top} h_2(\mathbf{y}) + \mu^{k\top} g_2(\mathbf{y})\\ & \quad + \left[f_1(\mathbf{x}^k) + \lambda^{k\top} h_1(\mathbf{x}^k) + \mu^{k\top} g_1(\mathbf{x}^k)\right],\\ & \qquad k = 1,\dots,K,\\ & 0 \geq \overline{\lambda}^{l\top} h_2(\mathbf{y}) + \overline{\mu}^{l\top} g_2(\mathbf{y})\\ & \quad + \left[\overline{\lambda}^{l\top} h_1(\overline{\mathbf{x}}^l) + \overline{\mu}^{l\top} g_1(\overline{\mathbf{x}}^l)\right],\\ & \qquad l = 1,\dots,L. \end{cases}$$

Note that the additional assumption makes the problem convex in $\mathbf{y}$ if $\mathbf{y}$ represent continuous variables. If $\mathbf{y} \in \mathbf{Y} = \{0, 1\}$, and the $\mathbf{y}$-variables participate linearly (i. e., $\mathbf{f}_2$, g_2 are linear in $\mathbf{y}$), then the relaxed master is convex. Therefore, this case represents an improvement over v3-GBD, and application of v3-GBD will result in valid support functions, which implies that the global optimum of the original problem will be obtained.

V3-GBD Without Separability

The global optimum search, GOS, aimed at exploiting and invoking special structure for nonconvex nonseparable problems

$$\begin{cases} \min & f(\mathbf{x},\mathbf{y})\\ \text{s.t.} & \mathbf{h}(\mathbf{x},\mathbf{y}) = 0\\ & \mathbf{g}(\mathbf{x},\mathbf{y}) \leq 0\\ & \mathbf{x} \in X \subseteq \mathbf{R}^n\\ & \mathbf{y} \in Y \subseteq \mathbf{R}^q, \end{cases}$$

under the conditions C1), C2), C3) and the additional condition:

i) $\mathbf{f}(\mathbf{x}, \mathbf{y})$, $\mathbf{g}(\mathbf{x}, \mathbf{y})$ are convex functions in $\mathbf{y}$ for every fixed $\mathbf{x}$;
ii) $\mathbf{h}(\mathbf{x}, \mathbf{y})$ are linear functions in $\mathbf{y}$ for every $\mathbf{x}$.

Hence both the primal and the relaxed problems attain their respective global solutions.

Note that since $\mathbf{x}$ and $\mathbf{y}$ are not separable, then the GOS cannot provide theoretically valid functions in the general case, but only if the $v(\mathbf{y})$ is convex (see the Section v2-GBD).

The global optimization approach (GOP) of [5,6] overcomes this fundamental difficulty and guarantees ϵ-global optimality for several classes of nonconvex problems.

GBD in Continuous and Discrete-Continuous Optimization

We mentioned in the Section Formulation that the original problem represents a sub-class of the problems for which the generalized Benders decomposition, GBD, can be applied. This is because we considered the $\mathbf{y} \in \mathbf{Y}$ set to consist of 0–1 variables, while [7] proposed an analysis for $\mathbf{Y}$ being a continuous, discrete or continuous-discrete set.

The main objective in this section is to present the modifications needed to carry on the analysis for continuous $\mathbf{Y}$ and discrete-continuous $\mathbf{Y}$ set.

The analysis presented for the primal problem remains the same. The analysis though for the Master problem changes only in the dual representation of the projection of the original problem (i. e., $v(\mathbf{y})$) on the $\mathbf{y}$-space. In fact, Theorem 3 is satisfied if in addition to the two conditions mentioned in C3) we have that:

iii) for each fixed $\mathbf{y}$, $v(\mathbf{y})$ is finite, $\mathbf{h}(\mathbf{x}, \mathbf{y})$, $\mathbf{g}(\mathbf{x}, \mathbf{y})$ and $\mathbf{f}(\mathbf{x}, \mathbf{y})$ are continuous on $\mathbf{X}$, $\mathbf{X}$ is closed and the ε-optimal solution of the primal problem $P(\mathbf{y})$ is nonempty and bounded for some $\varepsilon \geq 0$.

Hence, Theorem 3 has as assumptions: C1) and C3), which now has i), ii) and iii). The algorithmic procedure remains the same, while the theorem for the finite convergence becomes finite ε-convergence and requires additional conditions, which are described in the following theorem:

Theorem 6 (Finite ε-convergence) *Let*

i) ***Y*** *be a nonempty subset of* ***V****;*
ii) ***X*** *be a nonempty convex set;*
iii) f, $\boldsymbol{g}$ *be convex on* ***X*** *for each fixed* $\boldsymbol{y} \in \boldsymbol{Y}$*;*

iv) $\boldsymbol{h}$ *be linear on* $\boldsymbol{X}$ *for each fixed* $\boldsymbol{y} \in \boldsymbol{Y}$*;*

v) $f, \boldsymbol{g}, \boldsymbol{h}$ *be continuous on* $\boldsymbol{X} \times \boldsymbol{Y}$*;*

vi) the set of optimal multiplier vectors for the primal problem be nonempty for all $\boldsymbol{y} \in \boldsymbol{Y}$*, and uniformly bounded in some neighborhood of each such point.*

Then, for any given $\epsilon > 0$ *the GBD terminates in a finite number of iterations.*

Assumption i) (i. e., $\mathbf{Y} \subseteq \mathbf{V}$) eliminates the possibility of Step 3b, and there are many applications in which $\mathbf{Y} \subseteq \mathbf{V}$ holds (e. g., *variable factor programming*). If however, $\mathbf{Y} \not\subseteq \mathbf{V}$, then we may need to solve step 3b infinitely many successive times. In such a case, to preserve *finite* ϵ*-convergence*, we can modify the procedure so as to finitely truncate any excessively long sequence of successive executions of Step 3b and return to Step 3a with $\widehat{\mathbf{y}}$ equal to the extrapolated limit point which is assumed to belong to $\mathbf{Y} \cap \mathbf{V}$. If we do not make the assumption $\mathbf{Y} \subseteq \mathbf{V}$, then the key property to seek is that $\mathbf{V}$ has a representation in terms of a finite collection of constraints because if this is the case then Step 3b can occur at most a finite number of times. Note that if in addition to C1), we have that $\mathbf{X}$ represents bounds on the $\mathbf{x}$-variables or $\mathbf{X}$ is given by linear constraints, and $\mathbf{h}$, $\mathbf{g}$ satisfy the separability condition, then $\mathbf{V}$ can be represented in terms of a finite collection of constraints.

Assumption vi) requires that for all $\mathbf{y} \in \mathbf{Y}$ there exist optimal multiplier vectors and that these multiplier vectors do not go to infinity, that is they are uniformly bounded in some neighborhood of each such point. [7] provided the following condition to check the uniform boundedness:

If $\mathbf{X}$ is a nonempty, compact, convex set and there exists a point $\bar{\mathbf{x}} \in \mathbf{X}$ such that

$$\mathbf{h}(\bar{\mathbf{x}}, \bar{\mathbf{y}}) = 0,$$

$$\mathbf{g}(\bar{\mathbf{x}}, \bar{\mathbf{y}}) < 0,$$

then the set of optimal multiplier vectors is uniformly bounded in some open neighborhood of $\bar{\mathbf{y}}$.

See also

- ▶ Chemical Process Planning
- ▶ Decomposition Principle of Linear Programming
- ▶ Extended Cutting Plane Algorithm
- ▶ Generalized Outer Approximation
- ▶ MINLP: Application in Facility Location-allocation
- ▶ MINLP: Applications in Blending and Pooling Problems
- ▶ MINLP: Applications in the Interaction of Design and Control
- ▶ MINLP: Branch and Bound Global Optimization Algorithm
- ▶ MINLP: Branch and Bound Methods
- ▶ MINLP: Design and Scheduling of Batch Processes
- ▶ MINLP: Generalized Cross Decomposition
- ▶ MINLP: Global Optimization with αBB
- ▶ MINLP: Heat Exchanger Network Synthesis
- ▶ MINLP: Logic-based Methods
- ▶ MINLP: Outer Approximation Algorithm
- ▶ MINLP: Reactive Distillation Column Synthesis
- ▶ Mixed Integer Linear Programming: Mass and Heat Exchanger Networks
- ▶ Mixed Integer Nonlinear Programming
- ▶ Simplicial Decomposition
- ▶ Simplicial Decomposition Algorithms
- ▶ Stochastic Linear Programming: Decomposition and Cutting Planes
- ▶ Successive Quadratic Programming: Decomposition Methods

References

1. Benders JF (1962) Partitioning procedures for solving mixed-variables programming problems. Numer Math 4:238
2. Fletcher R, Leyffer S (1994) Solving mixed integer nonlinear programs by outer approximation. Math Program 66(3):327–349
3. Floudas CA (1995) Nonlinear and mixed integer optimization: Fundamentals and applications. Oxford Univ. Press, Oxford
4. Floudas CA, Aggarwal A, Ciric AR (1989) Global optimum search for nonconvex NLP and MINLP problems. Comput Chem Eng 13(10):1117–1132
5. Floudas CA, Visweswaran V (1990) A global optimization algorithm (GOP) for certain classes of nonconvex NLPs: I. Theory. Comput Chem Eng 14:1397–1417
6. Floudas CA, Visweswaran V (1993) A primal-relaxed dual global optimization approach. J Optim Th Appl 78(2):187–225
7. Geoffrion AM (1972) Generalized Benders decomposition. J Optim Th Appl 10:237–260
8. Geromel JC, Belloni MR (1986) Nonlinear programs with complicating variables: theoretical analysis and numerical experience. IEEE Trans Syst, Man Cybern SMC–16:231

Generalized Concavity in Multi-objective Optimization

ALBERTO CAMBINI, LAURA MARTEIN
Department Statistics and Applied Math.,
University Pisa, Pisa, Italy

MSC2000: 90C29

Article Outline

Keywords

Connectedness; Efficiency; Vector generalized concavity; Optimality conditions

In the context of economics and optimization, a fundamental role is nowadays recognized to generalized concavity which has been widely studied starting from the pioneering work of K. Arrow and A.C. Enthoven [1].

The study of generalized concavity of a vector valued function is not so deep as in the scalar case; nevertheless some classes with related properties have been suggested in order to obtain sufficient optimality conditions and the connectedness of the set of all efficient points.

In this order of ideas, since there are different ways in generalizing to the multi-objective case the definitions of generalized concave functions given in the scalar case, we introduce the following classes of generalized concave vector valued functions, referring to bibliography for further deepenings.

Let X be a convex subset of the n-dimensional space $\mathbf{R}^n$ and let F be a vector function from X to $\mathbf{R}^s$. Assume that $\mathbf{R}^s$ is partially ordered by the convex closed cone U with vertex at the origin $0 \in U$ and with nonempty interior (i. e. int$U \neq\emptyset$). Set $U^0 = U \setminus \{0\}$.

Definition 1 The function F is said to be U-*concave* if:

$$\begin{aligned}&F(x_1 + \lambda(x_2 - x_1))\\&\quad \in F(x_1) + \lambda(F(x_2) - F(x_1)) + U,\end{aligned}$$

for all $\lambda \in (0, 1)$ and all $x_1, x_2 \in S$.

Definition 2 The function F is said to be U-*quasiconcave* if:

$$x_1, x_2 \in S, \quad F(x_2) \in F(x_1) + U$$

imply

$$F(x_1 + \lambda(x_2 - x_1)) \in F(x_1) + U$$

for all $\lambda \in (0, 1)$.

Definition 3 The function F is said to be U^0-*quasiconcave* if:

$$x_1, x_2 \in S, \quad F(x_2) \in F(x_1) + U^0$$

imply

$$F(x_1 + \lambda(x_2 - x_1)) \in F(x_1) + U^0$$

for all $\lambda \in (0, 1)$.

Definition 4 The function F is said to be int$U$-*quasiconcave* if:

$$x_1, x_2 \in S, \quad F(x_2) \in F(x_1) + \operatorname{int} U$$

imply

$$F(x_1 + \lambda(x_2 - x_1)) \in F(x_1) + \operatorname{int} U$$

for all $\lambda \in (0, 1)$.

In [12], D.T. Luc suggests another class of quasiconcave functions which results less general than the one given in Definition 2, but which plays an important role in establishing the connectedness of the set of all efficient points.

Definition 5 The function F is said to be *Luc* U-*quasiconcave* if:

$$y \in \mathbf{R}^s, \quad x_1, x_2 \in S, \quad F(x_1), F(x_2) \in y + U$$

imply

$$F(x_1 + \lambda(x_2 - x_1)) \in y + U$$

for all $\lambda \in (0, 1)$.

In the scalar case, that is, when $s = 1$ and $U = \mathbf{R}_+$, Definitions 1, 2 and 5, 3 and 4 reduce to the ordinary definitions of concavity, quasiconcavity and semistrictly quasiconcavity, respectively.

Inclusion relationships among the previous classes of functions are given in the following Theorem:

Theorem 6

i) *if F is U-concave, then F is Luc U-quasiconcave;*

ii) *if F is Luc U-quasiconcave, then F is U-quasiconcave;*

iii) *if F is U-concave and U is a pointed cone, then F is $\text{int}U$-quasiconcave;*

iv) *if F is U-concave and U is a pointed cone, then F is U^0-quasiconcave.*

Proof i

i) Assume that $F(x_1), F(x_2) \in y + U$; it follows $(1 - \lambda)F(x_1) \in (1 - \lambda)y + U$ and $\lambda F(x_2) \in \lambda y + U$, so that $(1 - \lambda)F(x_1) + \lambda F(x_2) \in (1 - \lambda)y + \lambda y + U = y + U$.

ii) It is sufficient to choose $y = F(x_1)$.

iii) Assume that $F(x_2) \in F(x_1) + \text{int}U$, that is, $F(x_2) - F(x_1) \in \text{int}U$. Since F is U-concave we have $F(x_1 + \lambda(x_2 - x_1)) \in F(x_1) + \lambda(F(x_2) - F(x_1)) + U$. The thesis follows taking into account that for a pointed cone the property $\text{int}U + U = \text{int}U$ holds.

iv) The proof is similar to the one given in iii). □

Remark 7 When U is the *Paretian cone* $U = \mathbf{R}^s_+$, componentwise generalized concavity implies generalized concavity. For instance:

- if any component of F is quasiconcave then F is U-quasiconcave;
- if any component of F is strongly quasiconcave then F is either $\text{int}U$-quasiconcave or U^0-quasiconcave;
- if any component of F is upper semicontinuous and semistrictly quasiconcave then F is either $\text{int}U$-quasiconcave or U^0-quasiconcave.

It can be proven that F is $\mathbf{R}^s_+$-concave (Luc $\mathbf{R}^s_+$-quasiconcave) if and only if all its components are concave (quasiconcave); such a property does not hold for the other given classes of generalized concave functions, so that the inclusion relationships stated in i) and ii) of Theorem 6 are strict.

In the particular case of a continuous bicriteria function ($s = 2$, $U = \mathbf{R}^2_+$), the class of Luc U-quasiconcave functions collapses to the class of U-quasiconcave functions [8].

Remark 8 The following examples point out that the classes of $\text{int}U$-quasiconcave and U^0-quasiconcave functions are not comparable.

Consider the function F: $\mathbf{R} \to \mathbf{R}^3$, $F(x) = (x, x^2{-}x, -x^2{+}x)$ and the Paretian cone $U = \mathbf{R}^3_+$. F is $\text{int}U$-quasiconcave since there do not exist $x, y \in \mathbf{R}$ such that $F(y) > F(x)$; on the other hand, F is not U^0-quasiconcave since $F(1) = (1, 0, 0) \in F(0) + \mathbf{R}^3_+ \setminus \{0\}$, but $F(1/2) \notin F(0) + \mathbf{R}^3_+ \setminus \{0\}$.

Consider now the function F: $\mathbf{R} \to \mathbf{R}^2$, $F(x) = (x, f(x))$ with $f(x) = 0$ if $x \leq 1$, $f(x) = x - 1$ if $x > 1$ and the Paretian cone $U = \mathbf{R}^2_+$. It is easy to verify that F is U^0-quasiconcave, but F is not U-quasiconcave since $F(2) = (2, 1) \in F(0) + \text{int}\mathbf{R}^2_+$, and $F(1) = (1, 0) \notin F(0) + \text{int}\mathbf{R}^2_+$.

Remark 9 In the scalar case an upper semicontinuous and semistrictly quasiconcave function is also quasiconcave; this property is lost for a vector valued function as is shown in the following example, so that the two classes are not comparable: consider the function F: $\mathbf{R} \to \mathbf{R}^2$ defined as $F(x) = (x \sin 1/x, -x \sin 1/x)$ if $x \neq 0$; $F(x) = 0$ if $x = 0$. F is continuous and U^0-quasiconcave but it is not U-quasiconcave at $x = 0$.

Remark 10 As is known, in the scalar case there exists a characterization of quasiconcave functions in the differentiable case; unfortunately such a characterization cannot be extended in the vector case (for further developing see [7]).

Consider a differentiable vector valued function F. As for the quasiconcave case, there are different ways to extend the concept of pseudoconcavity introduced by O.L. Mangasarian [14]. With the aim to state some sufficient optimality conditions, we introduce the following two classes of functions, where $J_F(x)$ denotes the Jacobian matrix of F evaluated at x.

Definition 11 F is said to be *U-weakly pseudoconcave* if:

$$x_1, x_2 \in S, \quad F(x_2) \in F(x_1) + U^0$$

imply

$$J_F(x_1)d \in U^0, \qquad d = \frac{x_2 - x_1}{\|x_2 - x_1\|}.$$

Definition 12 F is said to be *U-pseudoconcave* if:

$$x_1, x_2 \in S, \quad F(x_2) \in F(x_1) + U^0$$

imply

$$J_F(x_1)d \in \operatorname{int} U, \qquad d = \frac{x_2 - x_1}{\|x_2 - x_1\|}.$$

When $s = 1$ and $U = \mathbf{R}_+$, Definitions 11, 12 reduce to the ordinary definition of a pseudoconcave function.

Obviously, a function which is U-pseudoconcave is U-weakly quasiconcave too; a linear function is U-concave and U-weakly pseudoconcave with respect to every cone U with vertex at the origin $0 \in U$ but it is not U-pseudoconcave. As a consequence the class of U-pseudoconcave functions is properly contained in the class of U-weakly pseudoconcave functions.

Remark 13 When U is the Paretian cone $U = \mathbf{R}^s_+$, we have:

- if any component of F is pseudoconcave then F is $\mathbf{R}^s_+$-weakly pseudoconcave;
- if any component of F is strictly pseudoconcave then F is either $\mathbf{R}^s_+$-weakly pseudoconcave or $\mathbf{R}^s_+$-pseudoconcave.

Efficiency

Consider the following vector optimization problem:

$$\text{(P)} \quad U - \max F(x), \quad x \in S \subseteq X,$$

where X is an open set of $\mathbf{R}^n$, F: $X \to \mathbf{R}^s$, and $U \in \mathbf{R}^s$ is a nontrivial cone with vertex at the origin $0 \in U$, int$U \neq \emptyset$.

A point $x_0 \in S$ is said to be:

- *weakly efficient* if $F(x) \notin F(x_0) + \operatorname{int} U$, for all $x \in S$;
- *efficient* if $F(x) \notin F(x_0) + U^0$, for all $x \in S$;
- *strictly efficient* if $F(x) \notin F(x_0) + U$, for all $x \in S$, $x \neq x_0$.

If the previous conditions are verified in $I \cap S$, where I is a suitable neighborhood of x_0, then x_0 is said to be a *local weakly efficient point* a *local efficient point* and a *local strictly efficient point*, respectively.

In the scalar case ($s = 1$, $U = \mathbf{R}_+$), the definitions of a (local) weakly efficient point and an (local) efficient point reduce to the ordinary definition of a (local) maximum point, while a (local) strictly efficient point reduces to the ordinary definition of a (local) strict maximum point. Obviously (local) strictly efficiency implies (local) efficiency and (local) efficiency implies (local) weakly efficiency.

The concept of efficiency was originally introduced by V. Pareto in the early 1900s when he used the positive orthant $\mathbf{R}^s_+$ to generate the order; therefore when $U = \mathbf{R}^s_+$ efficient points are often called *Pareto points*.

As in the scalar case, vector generalized concavity plays an important role in investigating relationships between local and global optima. Following [14], the assumption of convexity of the feasible region can be weakened requiring that S is star-shaped at the point x_0.

A set $S \subset X$ is said to be *star-shaped* at $x_0 \in S$ if for every $x \in S$ it results:

$$[x, x_0] = \{tx + (1 - t)x_0\colon \ t \in [0, 1]\} \subset S.$$

Since optimality results involve a feasible point, from now on we will consider generalized concavity at a point x_0; this means that all the given definitions hold with $x_1 = x_0$. The following theorem shows that, under suitable assumption of generalized concavity, local efficiency implies global efficiency.

Theorem 14 *Let us consider problem (P) where S is a star-shaped set at x_0.*

i) if x_0 is a local weakly efficient point and F is int$U$-quasiconcave at x_0, then x_0 is a weakly efficient point for (P);

ii) if x_0 is a local efficient point and F is U^0-quasiconcave at x_0, then x_0 is an efficient point for (P);

iii) if x_0 is a strict local efficient point and F is U-quasiconcave at x_0, then x_0 is a strictly efficient point for (P);

iv) if x_0 is a local efficient point and F is U-pseudoconcave at x_0, then x_0 is an efficient point for (P).

Proof i) Assume that there exists $x^* \in S$ such that $F(x^*) \in F(x_0) + \operatorname{int} U$. Since F is int$U$-quasiconcave at x_0, we have $F(x_0 + \lambda(x^* - x_0)) \in F(x_0) + \operatorname{int} U$ for all $\lambda \in (0, 1)$ and such a relation implies, choosing λ small enough, the non local weakly efficiency of x_0.

ii), iii) follow with similar arguments.

iv) Assume that there exists $x^* \in S$ such that $F(x^*) \in F(x_0) + U^0$. Since F is U-pseudoconcave at x_0, we have $J_F(x_0)d \in \operatorname{int} U$, $d = (x^* - x_0)/\|x^* - x_0\|$, that is

$$\lim_{t \to 0^+} \frac{F(x_0 + td) - F(x_0)}{t} \in \operatorname{int} U$$

and this implies the existence of a suitable $\epsilon > 0$ such that $F(x_0 + td) - F(x_0) \in \operatorname{int} U$ for all $t \in (0, \epsilon)$.

Set $t = \lambda \|x^* - x_0\|$; we have $F(x_0 + \lambda(x^* - x_0)) \in F(x_0) + \operatorname{int} U$ for all $\lambda \in (0, \epsilon/\|x^* - x_0\|)$ and this contradicts the local efficiency of x_0. □

Corollary 15 *Let us consider problem (P) where S is locally star shaped at x_0.*

i) If U is a pointed cone and F is U-concave at x_0, then a local efficient point x_0 is an efficient point too.

ii) If F is linear, then a local efficient point x_0 is an efficient point too.

Optimality Conditions

Now we point out the role played by vector generalized concavity in stating sufficient optimality conditions. With this aim consider the necessary optimality conditions stated in the following Theorem:

Theorem 16 *Let us consider problem (P) where F is differentiable at x_0.*

i) If x_0 is a local interior efficient point for (P), then

$$\exists \alpha \in U^* \setminus \{0\} : \alpha^\top J_{F_{x_0}} = 0, \tag{1}$$

where U^* denotes the positive polar cone of U.

ii) If x_0 is a local efficient point for (P) then

$$J_{F_{x_0}}(v) \notin \operatorname{int} U, \qquad \forall v \in T(S, x_0),\ v \neq 0. \tag{2}$$

Here, $T(S, x_0)$ is the *Bouligand tangent cone*, defined as:

$$T(S, x_0) = \left\{ v\colon \begin{array}{c} \exists \{\alpha_n\} \subset \mathbf{R},\ \{x_n\} \subset S, \\ \alpha_n \to \infty,\ x_n \to x_0, \\ \alpha_n(x_n - x_0) \to v \end{array} \right\}.$$

The following theorem points out the different roles played by weakly pseudoconcavity and pseudoconcavity:

Theorem 17 *Let us consider problem (P) where S is a star shaped set and F is differentiable at x_0.*

i) if (1) holds and F is U-pseudoconcave at x_0, then x_0 is an efficient point for (P);

ii) if (1) holds with $\alpha \in \operatorname{int} U^*$ and F is U-weakly pseudoconcave at x_0, then x_0 is an efficient point for (P);

iii) if (2) holds and F is U-pseudoconcave at x_0, then x_0 is an efficient point for (P);

iv) if $J_{F_{x_0}}(v) \notin U^0$, for all $v \in T(S, x_0)$ and F is U-weakly pseudoconcave at x_0, then x_0 is an efficient point for (P).

Proof i) Assume that there exists $x^* \in S$ such that $F(x^*) \in F(x_0) + U^0$. Since F is U-pseudoconcave at x_0, we have $J_{F_{x_0}}(d) \in \operatorname{int} U$, $d = (x^* - x_0)/\|x^* - x_0\|$, so that $\alpha^\top(J_{F_{x_0}}(d)) > 0$ and this contradicts (1).

ii) Assume that there exists $x^* \in S$ such that $F(x^*) \in F(x_0) + U^0$. Since F is U-weakly pseudoconcave at x_0, we have $J_{F_{x_0}}(d) \in U^0$, $d = (x^* - x_0)/\|x^* - x_0\|$, so that $\alpha^\top(J_{F_{x_0}}(d)) > 0$ and this contradicts (1).

iii), iv) follow immediately. □

When F is a linear vector valued function, Theorem 17ii) can be specified by means of the following theorem:

Theorem 18 *Consider problem (P) where F is linear and U is a pointed cone.*

An interior point x_0 is an efficient point for (P) if and only if there is $\alpha \in \operatorname{int} U^*$ such that $\alpha^\top J_{F_{x_0}} = 0$.

F. John Generalized Conditions

Now we stress the role of vector generalized concavity in stating the sufficiency of F. John condition.

With this aim consider the vector problem (P) in the following form:

$$(P) \begin{cases} U - \max & F(x), \\ & x \in S = \{x \in X\colon\ G(x) \in V\}, \end{cases}$$

where $X \subset \mathbf{R}^n$ is an open set, $F\colon X \to \mathbf{R}^s$, $G\colon X \to \mathbf{R}^m$ are differentiable functions and $U \subset \mathbf{R}^s$, $V \subset \mathbf{R}^m$ are closed, pointed, convex cones with vertices at the origin and nonempty interiors.

Denote with U^*, V^* the positive polar cones of U and V, respectively, and let x_0 be a feasible point such that $G(x_0) = 0$.

The following *F. John necessary optimality conditions* hold:

Theorem 19 *If x_0 is a local efficient point for (P), then*

$$\begin{aligned} &\exists (\alpha_F, \alpha_G) \neq 0, \alpha_F \in U^*, \alpha_G \in V^* : \\ &\alpha_F^\top J_{F_{x_0}} + \alpha_G^\top J_{G_{x_0}} = 0. \end{aligned} \tag{3}$$

The following theorem points out the role of generalized concavity in stating sufficient optimality conditions:

Theorem 20 *Let us consider the vector optimization problem (P) where S is a star shaped set at x_0 and F, G are differentiable at x_0.*

i) if F is U-weakly pseudoconcave at x_0, G is V-quasiconcave at x_0, and (3) holds with $\alpha_F \in \text{int}U^*$, then x_0 is an efficient point for (P).

ii) if F is U-pseudoconcave at x_0, G is V-quasiconcave at x_0, and (3) holds with $\alpha_F \in U^* \setminus \{0\}$, then x_0 is an efficient point for (P).

Proof i) Suppose that there exists $x^* \in S$ such that $F(x^*) \in F(x_0) + U^0$. Since F is U-weakly pseudoconcave at x_0 and G is V-quasiconcave at x_0 we have, respectively, $J_{F_{x_0}}(x^* - x_0) \in U^0$, $J_{G_{x_0}}(x^* - x_0) \in V$ and thus $\alpha_F^\top J_{F_{x_0}}(x^* - x_0) > 0$, $\alpha_G^\top J_{G_{x_0}}(x^* - x_0) \geq 0$ since $\alpha_F \in \text{int}U^*$ and $\alpha_G \in V^*$. Consequently $\alpha_F^\top J_{F_{x_0}}(x^* - x_0) + \alpha_G^\top J_{G_{x_0}}(x^* - x_0) > 0$ and this contradicts (3).

ii) similar to the one given in i). □

Connectedness of the Efficient Points Sets

A vector maximization problem normally has a continuum of optimal alternatives and it may be necessary to select one or several of these which are best with respect to some additional auxiliary criterion, so that a desirable property is connectedness since it provides a possibility of continuous moving from one efficient point to any other along optimal alternatives only. Consider problem (P) where $F = (f_1, \ldots, f_s)$ is a continuous function and U is the Paretian cone; denote with $S(a)$ the upper level set associated to the point $a \in \mathbf{R}^s$, that is $S(a) = \{x \in S: F(x) \in a + U\}$. The following fundamental result was given by A.R. Warburton [16].

Theorem 21

i) if $f_1, \ldots, f_s$ are quasiconcave functions on the closed convex set S and $S(a)$ is compact for each $a \in f_1(S) \times \cdots \times f_s(S)$, then the set of all weakly Pareto points is nonempty and connected;

ii) if $f_1, \ldots, f_s$ are strongly quasiconcave functions on the closed convex set S and $S(a)$ is compact for each $a \in f_1(S) \times \cdots \times f_s(S)$, then the set of all Pareto points is nonempty and connected.

Obviously the compactness of sets $S(a)$ is verified when S is a compact set; in this last case for a bicriteria and three criteria, Theorem 21ii) holds, requiring the weaker assumption of semistrictly quasiconcavity instead of strongly quasiconcavity [9,15].

In [12], Luc extends Theorem 21i) with respect to a pointed closed convex cone requiring that F is U-continuous.

F is said to be U-*continuous* at $x \in S$ if for any neighborhood H of $F(x)$, there exists a neighborhood I of x such that $F(y) \in H - U$ for all $y \in I \cap S$.

Theorem 22 *Assume that F is a U-continuous Luc U-quasiconcave function on S and the set of all weakly efficient points of $S(a)$ is compact for each $a \in \mathbf{R}^s$. Then the set of all weakly efficient points is nonempty and connected.*

See also

► Invexity and its Applications
► Isotonic Regression Problems

References

1. Arrow KJ, Enthoven AC (1961) Quasi-concave programming. Econometrica 29:779–800
2. Cambini A, Martein L (1993) An approach to optimality conditions in vector and scalar optimization. In: Diewert WE et al (eds) Mathematical modelling in Economics. Springer, Berlin, pp 345–358
3. Cambini A, Martein L (1994) Generalized concavity and optimality conditions in vector and scalar optimization. In: Komlosi S, Rapcsak T, Schaible S (eds) Generalized Convexity. vol. Lecture Notes Economics and Math Systems. Springer, Berlin, pp 337–357
4. Cambini A, Martein L (1998) Generalized concavity in multiobjective programming. In: Crouzeix JP, Martinez-Legaz JE, Volle M (eds) Generalized Convexity, Generalized Monotonicity: Recent Results. Kluwer, Dordrecht, pp 453–467
5. Cambini A, Martein L, Cambini R (1997) Some optimality conditions in multiobjective programming. In: Climaco JN (ed) Multicriteria Analysis. Springer, Berlin, pp 168–178
6. Cambini R (1996) Generalized concavity and optimality conditions in vector optimization. In: Du D-Z, Zhang X-S, Cheng K (eds) Oper. Res. Appl. World Publ. Corp., pp 172–180
7. Cambini R (1996) Some new classes of generalized concave vector-valued functions. Optim 36:11–24
8. Cambini R (1998) Generalized concavity for bicriteria functions. In: Crouzeix JP, Martinez-Legaz JE, Volle M (eds) Generalized Convexity, Generalized Monotonicity: Recent Results. Kluwer, Dordrecht, pp 439–451
9. Daniilidis A, Hadjisavvas N, Schaible S (1997) Connectedness of the efficient set for three- objective quasiconcave maximization problems. J Optim Th Appl 93(3):517–524
10. Jahn J (1986) Mathematical vector optimization in partially ordered linear spaces. P. Lang, Frankfurt am Main
11. Jahn J, Sach E (1986) Generalized quasiconvex mappings and vector optimization. SIAM J Control Optim 24(2):306–322

12. Luc DT (1987) Connectedness of the efficient point sets in quasiconcave vector maximization. J Math Anal Appl 122:346–354
13. Luc DT (1988) Theory of vector optimization. Lecture Notes Economics and Math Systems, vol 319. Springer, Berlin
14. Mangasarian OL (1969) Nonlinear programming. McGraw-Hill, New York
15. Schaible S (1983) Bicriteria quasiconcave programs. Cahiers CERO 25:93–101
16. Warburton AR (1983) Quasiconcave vector maximization: connectedness of the sets of Pareto-optimal and weak Pareto-optimal alternatives. J Optim Th Appl 40(4):537–557

Generalized Disjunctive Programming

IGNACIO E. GROSSMANN
Department of Chemical Engineering,
Carnegie Mellon University, Pittsburgh, USA

Article Outline

Keywords and Phrases

Generalized disjunctive programming; Disjunctive programming; Convex hull; Mixed-integer programming; Outer-approximation method; Generalized benders decomposition

Synonyms

Boolean variables; Convex functions; Disjunctions; Convex hull disjunctions; Disjunctive programming; Generalized disjunctive programming; Hull relaxation; Mixed-integer programming; MILP; MINLP

Introduction

Generalized Disjunctive Programming (GDP) [13] is an extension of disjunctive programming [1,2] that provides an alternate way of modeling mixed-integer linear programming (MILP) and mixed-integer nonlinear programming (MINLP) problems. The general formulation of a (GDP) is as follows:

$$\begin{aligned}
&\min Z = \sum_{k\in K} c_k + f(x)\\
&s.t.\quad r(x) \le 0\\
&\bigvee_{j\in J_k}\begin{bmatrix} Y_{jk}\\ g_{jk}(x) \le 0\\ c_k = \gamma_{jk}\end{bmatrix} \quad k\in K \qquad\qquad \text{(GDP)}\\
&\qquad \Omega(Y) = True\\
&x\in R^n,\quad c\in R^m,\quad Y\in\{true, false\}^m
\end{aligned}$$

where Y_{jk} are the Boolean variables that decide whether a given term j in a disjunction $k \in K$ is true or false, and x are the continuous variables. The objective function involves the term $f(x)$ for the continuous variables and the charges c_k that depend on the discrete choices in each disjunction $k \in K$. The constraints $r(x) \le 0$ must hold regardless of the discrete choices, and $g_{jk}(x) \le 0$ are conditional constraints that must hold when Y_{jk} is true in the j-th term of the k-th disjunction. The cost variables c_k correspond to the fixed charges, and their value equals to γ_{jk} if the Boolean variable Y_{jk} is true. $\Omega(Y) = True$ are logical relations for the Boolean variables expressed as propositional logic. An important particular case is the one where the functions $f(x)$, $r(x)$ and $g_{jk}(x)$ are all linear. For the nonlinear case it is assumed for the derivation of basic methods that the functions are convex, although in practical applications these often correspond to nonconvex functions.

Mixed-Integer Programming Reformulations

Problem (GDP) can be reformulated as the following "big-M"MINLP problem,

$$\begin{aligned}
&\min Z = \sum_{k\in K}\sum_{j\in J_k} \gamma_{jk} y_{jk} + f(x)\\
&s.t.\quad r(x) \le 0\\
&g_{jk}(x) \le M_{jk}(1 - y_{jk})\,,\quad j\in J_k,\quad k\in K \qquad \text{(BM)}\\
&\sum_{j\in J_k} y_{jk} = 1,\quad k\in K\\
&Ay \le a\\
&0 \le x \le x^U,\quad y_{jk}\in\{0,1\},\quad j\in J_k,\quad k\in K
\end{aligned}$$

where the Boolean variables are replaced by binary variables y_{jk}, the disjunctions are replaced by "Big-M" constraints which involve a parameter M_{jk} and binary variables y_{jk}. The propositional logic statements $\Omega(Y)$ = True are replaced by the linear constraints $Ay \leq a$ as described by Williams [19]. Here we assume that x is a non-negative variable with finite upper bound x^U. An important issue in model (BM) is how to specify a valid value for the Big-M parameter M_{jk}. If the value is too small, then feasible points may be cut off. If M_{jk} is too large, then the continuous relaxation might be too loose yielding weak lower bounds. Therefore, finding the smallest valid value for M_{jk} is the desired selection. For linear constraints one can use the upper and lower bound of the variable x to calculate the maximum value of each constraint, which then can be used to calculate a valid value of M_{jk}. For nonlinear constraints one can in principle maximize each constraint over the feasible region, which is a non-trivial calculation. It is also important to note that if the binary variables y_{jk} are specified as continuous, $0 \leq y_{jk} \leq 1$, and the functions $f(x)$, $r(x)$ and $g_{jk}(x)$ are assumed to be convex, the relaxation of problem (BM) reduces to a convex NLP problem, that provides a valid lower bound to the solution of problem (GDP).

The MINLP hull reformulation of problem (GDP) is based on the following proposition by Lee and Grossmann [11]:

Proposition 1 *The convex hull of each disjunction $k \in K$ in problem (GDP),*

$$\bigvee_{j\in J_k} \begin{bmatrix} Y_{jk} \\ g_{jk}(x) \leq 0 \\ c = \gamma_{jk} \end{bmatrix} \qquad (\mathrm{D}_k)$$

$$0 \leq x \leq x^U, \quad c \geq 0$$

where $g_{jk}(x) \leq 0$ are convex inequalities, is a convex set and is given by,

$$\begin{aligned} & x = \sum_{j\in J_k} \nu^{jk}, c = \sum_{j\in J} y_{jk}\gamma_{jk} \\ & 0 \leq \nu^{jk} \leq y_{jk}x^U_{jk}, j \in J_k \\ & \sum_{j\in J_k} y_{jk} = 1, 0 \leq y_{jk} \leq 1, j \in J_k \qquad (\mathrm{CH}_k) \\ & y_{jk}g_{jk}(\nu^{jk}/y_{jk}) \leq 0, j \in J_k \\ & x, c, \nu^{jk} \geq 0, j \in J_k \end{aligned}$$

The proof is based on an extension of the work by Stubbs and Mehrotra [16]. In (CH$_k$), ν^{jk} are disaggregated variables that are assigned to each term of the disjunction $\{k \in K\}$, and y_{jk} can be regarded as the weight factors that determine the feasibility of the disjunctive term. Note that when y_{jk} is 1, then the j'th term in the k'th disjunction is enforced and the other terms are ignored. The constraints $y_{jk}g_{jk}(\nu^{jk}/y_{jk})$ are convex if $g_{jk}(x)$ is convex as discussed in Hiriart-Urruty and Lemaréchal [8]. Formal proofs can be found in [15] and [16]. Note that the convex hull (CH$_k$) reduces to the result by Balas [2] if the constraints are linear. Based on the convex hull relaxation (CH$_k$), Lee and Grossmann [11] proposed the following MINLP hull reformulation of (GDP):

$$\begin{aligned} & \min Z = \sum_{k\in K}\sum_{j\in J_k} \gamma_{jk}y_{jk} + f(x) \\ & s.t. \quad r(x) \leq 0 \\ & x = \sum_{j\in J_k} \nu^{jk}, \quad \sum_{j\in J_k} y_{jk} = 1\,, \qquad k \in K \qquad (\mathrm{HR}) \\ & 0 \leq \nu^{jk} \leq y_{jk}x^U_{jk}\,, \quad j \in J_k\,, \quad k \in K \\ & y_{jk}g_{jk}(\nu^{jk}/y_{jk}) \leq 0, j \in J_k\,, \quad k \in K \\ & Ay \leq a \\ & 0 \leq x, \nu^{jk} \leq x^U, \quad y_{jk} = 0.1, j \in J_k\,, \quad k \in K\,. \end{aligned}$$

The relaxation of problem (HR) where $0 \leq y_{jk} \leq 1$, reduces to a convex NLP problem that yields a valid lower bound to the optimal solution of problem (GDP). Also, this relaxation, which can also be regarded as a generalization of the disjunctive problem studied by Ceria and Soares [4], can be interepreted as one where the convex hulls of each of the disjunctions are interesected.

The following proposition holds for problems (PR) and (BM) as proved by Grossmann and Lee [7].

Proposition 2 *LetZ^R_{HR} be the optimal value of problem (HR) where the binary variables are relaxed as $0 \leq y_{jk} \leq 1$, and let Z^R_{BM} be the optimal value of problem (BM) where the binary variables are relaxed as $0 \leq y_{jk} \leq 1$. Then, $Z^R_{BM} \leq Z^R_{HR}$.*

Hence, problem (HR) has the useful property that the lower bound of its relaxation is greater than or equal to the lower bound predicted from the relaxation of problem (BM). In some problems this translates into a significantly tighter formulations [13,14]). The trade-off, however, is that in the reformulation (HR) the number of constraints and variables is larger than the one in the reformulation (BM).

It is also important to point out that for the computer implementation of the constraint $y_{jk} g_{jk}(\nu_{jk}/y_{jk}) \leq 0$ in problem (HR), an approximation is required for the nonlinear functions, $g_{jk}(x)$ in order to avoid the division by zero when $y_{jk} = 0$. Furman et al. [5] have proposed the following approximation, which has the interesting feature that it is exact for $y_{jk} = 0$ and $y_{jk} = 1$,

$$((1-\varepsilon)y_{jk} + \varepsilon)(g_{jk}(\nu_{jk}/((1-\varepsilon)y_{jk} + \varepsilon))) \\ - \varepsilon g_{jk}(0)(1 - y_{jk}) \leq 0 \,.$$

Furthermore, it can be shown that this inequality is convex for any value of ε. Note also that this expression reduces to the original one as $\varepsilon \to 0$.

Solution Algorithms for GDP

The most direct way of solving problem (GDP) is by reformulating it as an MINLP (or MILP for the linear case). In both cases the big-M and hull reformulation are the two extreme choices. The latter generally yields tighter relaxations, but involves solving a larger problem. For the linear case LP-based branch and cut methods can be used [10], including special cutting plane techniques [14]. For the nonlinear case, MINLP methods such as branch and bound, outer-approximation, Generalized Benders, extended cutting plane or hybrid methods can be used [6].

Logic-based method for solving linear problems (GDP) include the branch and bound method by Beaumont [3], which branches on the constraints of the disjunctions. Raman and Grossmann [13] developed a branch and bound method which solves GDP problem in hybrid form, by exploiting the tight relaxation of the disjunctions and the tightness of the well-behaved mixed-integer constraints. For the nonlinear case a disjunctive branch and bound method based on the hull relaxation has been proposed by Lee and Grossmann [11] that is coupled with logic inference techniques [9]. Also, for the special case of two-term disjunctions in (GDP), which typically arise in process network problems, Türkay and Grossmann [17] have proposed outer-approximation and Generalized Benders Decomposition algorithms. Some of these algorithms have been implemented in LOGMIP, a computer code based on GAMS [18]. Finally, for the nonconvex case a disjunctive branch and bound method coupled with a spatial branch and bound search has been reported in [12].

References

1. Balas E (1979) Disjunctive programming. Ann Discret Math 5:3–51
2. Balas E (1985) Disjunctive programming and a hierarchy of relaxations for discrete optimization problems. SIAM J Alg Disc Meth 6:466–486
3. Beaumont N (1991) An algorithm for disjunctive programs. Eur J Oper Res 48:362–371
4. Ceria S, Soares J (1999) Convex programming for disjunctive optimization. Math Program 86(3):595–614
5. Furman K, Sawaya NW, Grossmann IE (2007) A robust MINLP reformulation for the implementation of nonlinear disjunctive programming. manuscript under preparation
6. Grossmann IE (2002) Review of nonlinear mixed-integer and disjunctive programming techniques. Optim Eng 3:227–252
7. Grossmann IE, Lee S (2003) Generalized disjunctive programming: nonlinear convex hull relaxation and algorithms. Comput Optim Appl 26:83–100
8. Hiriart-Urruty J, Lemaréchal C (1993) Convex analysis and minimization algorithms. Springer, Berlin, New York
9. Hooker JN (1999) Logic-based methods for optimization. Wiley, New York
10. Johnson EL, Nemhauser GL, Savelsbergh MWP (2000) Progress in linear programming based branch-and-bound algorithms: an exposition. INFORMS J Comput 12:2–23
11. Lee S, Grossmann IE (2000) New algorithms for nonlinear generalized disjunctive programming. Comput Chem Eng 24:2125–2141
12. Lee S, Grossmann IE (2001) A global optimization algorithm for nonconvex generalized disjunctive programming and applications to process systems. Comput Chem Eng 25:1675–1697
13. Raman R, Grossmann IE (1994) Modelling and computational techniques for logic based integer programming. Comput Chem Eng 18(7):563–578
14. Sawaya NW, Grossmann IE (2005) A cutting plane method for solving linear generalized disjunctive programming problems. Comput Chem Eng 29:1891–1913
15. Sawaya NW (2006) Reformulations, relaxations and cutting planes for generalized disjunctive programming, Ph.D. thesis. Carnegie Mellon University, Pittsburgh
16. Stubbs R, Mehrotra S (1999) A branch-and-cut method for 0-1 mixed convex programming. Math Program 86(3): 515–532
17. Türkay M, Grossmann IE (1996) Logic-based MINLP algorithms for the optimal synthesis of process networks. Comput Chem Eng 20(8):959–978

18. Vecchietti A, Grossmann IE (1999) LOGMIP: A disjunctive 0-1 nonlinear optimizer for process systems models. Comput Chem Eng 23:555–565
19. Williams HP (1985) Mathematical building in mathematical programming. Wiley, Chichester

Generalized Eigenvalue Proximal Support Vector Machine Problem

MARIO R. GUARRACINO, SALVATORE CUCINIELLO, DAVIDE FEMINIANO
High Performance Computing and Networking Institute, Italian Research Council, Napoli, Italy

MSC2000: 68Q32, 68T10

Article Outline

Problem

Consider two matrices $A \in \mathbb{R}^{n \times m}$ and $B \in \mathbb{R}^{k \times m}$, each row being a point in one of two classes in the feature space. The *generalized eigenvalue proximal support vector machine* (GEPSVM) consists in finding two hyperplanes each one being closer to one set of points and farther from another set of points. Let $x'w - \gamma = 0$ be a hyperplane in $\mathbb{R}^m$. In order to satisfy the previous condition for all points in A, the hyperplane can be obtained by solving the following optimization problem:

$$\min_{w,\gamma \neq 0} \frac{\|Aw - e\gamma\|^2}{\|Bw - e\gamma\|^2} . \tag{1}$$

The hyperplane for B can be obtained by minimizing the inverse of the objective function in (1). Now, let

$$\begin{aligned} G &= [A \quad -e]'[A \quad -e] , \\ H &= [B \quad -e]'[B \quad -e] \end{aligned} \tag{2}$$

and

$$z = [w' \quad \gamma]' . \tag{3}$$

Then (1) becomes

$$\min_{z \in \mathbb{R}^m} \frac{z'Gz}{z'Hz} . \tag{4}$$

The expression in (4) is the Raleigh quotient of the generalized eigenvalue problem $Gz = \lambda Hz$. When H is positive definite, the stationary points are obtained at and only at the eigenvectors of (4), where the value of the objective function is given by the eigenvalues. The Raleigh quotient is bounded, and it ranges over the interval determined by minimum and maximum eigenvalues [4]. H is positive definite under the assumption that the columns of $[B \quad -e]$ are linearly independent. The reciprocal of the objective function in (4) has the same eigenvectors and reciprocal eigenvalues. Let $z_{\min} = [w_1' \quad \gamma_1]'$ and $z_{\max} = [w_2' \quad \gamma_2]'$ be the eigenvectors related to the eigenvalues of the smallest and largest modulo, respectively. Then $x'w_1 - \gamma_1 = 0$ is the closest hyperplane to the set of points in A and the furthest from those in B and $x'w_2 - \gamma_2 = 0$ is the closest hyperplane to the set of points in B and the furthest from those in A. GEPSVM finds application in many supervised learning problems [3]. For example, a bank prefers to classify customer loan requests as “good” or “bad” depending on their ability to pay back the loan. The Internal Revenue Service tries to discover tax evaders starting from the characteristics of known evaders. As another example, a built-in system in a car could detect if a walking pedestrian is going to cross the street. More applications can be found in biology and medicine. The tissues that are prone to cancer can be detected with high accuracy, or new DNA sequences or proteins can be tracked down to their origins. Given its amino acid sequence, finding out how a protein folds provides important information about its expression level. An unlabeled point x is associated to the class y_i related to the closest hyperplane P_i. Therefore, a point x is classified using its distance for the corresponding hyperplane:

$$y_i = \operatorname{argmin}_{i=1,2}\{\operatorname{dist}(x, P_i)\} , \tag{5}$$

where

$$\operatorname{dist}(x, P_i) = \frac{|x'w_i - \gamma_i|}{\|w_i\|} . \tag{6}$$

Kernel Formulation

To obtain greater separability between classes, nonlinear embedding of data to a higher-dimensional space is required. This nonlinear mapping can be done implicitly by kernel functions, which represent the inner product of the elements in a nonlinear space. Kernel functions can be described as follows:

$$K(x_i, x_j) = \langle \phi(x_i) - \phi(x_j), \phi(x_i) - \phi(x_j) \rangle \,, \tag{7}$$

where $\phi(x)$ is the embedding function.

Using kernels, we can express the problem in terms of inner products between elements, and therefore the computationally expensive calculation of the feature, in the embedded space, is avoided. Some commonly used kernel functions are

$$\begin{aligned} \text{Linear} \quad & K(x_i, x_j) = x_i' \cdot x_j \\ \text{Polynomial} \quad & K(x_i, x_j) = (x_i' \cdot x_j + 1)^d \\ \text{Gaussian} \quad & K(x_i, x_j) = \exp\left(- \frac{\| x_i - x_j \|^2}{\sigma} \right) . \end{aligned}$$

Using the kernel function, each element of the kernel matrix is

$$K(A, B)_{i,j} = K(A^i, B^j) \,. \tag{8}$$

Let

$$C = \begin{bmatrix} A \\ B \end{bmatrix} .$$

Then problem (1) becomes

$$\min_{u,\gamma \neq 0} \frac{\|K(A, C)u - e\gamma\|^2}{\|K(B, C)u - e\gamma\|^2} \,. \tag{9}$$

A point x is classified using its distance for the corresponding hyperplane in the feature space:

$$y_i = \operatorname{argmin}_{i=1,2}\{\operatorname{dist}(x, P_i)\} \,, \tag{10}$$

where

$$\operatorname{dist}(x, P_i) = \frac{|K(x, C)u_i - \gamma_i|}{\|u_i\|} \,. \tag{11}$$

The associated eigenvalue problem has matrices of order $n + k + 1$ and rank at most m. This means a regularization technique is needed since the problem can be singular.

Algorithm

Let G and H be as defined in (2). Note that even if A and B are full rank, matrices G and H are always rank-deficient. The reason is that G and H are matrices of order $m + 1$, and their rank can be at most m. The added complexity due to the singularity of the matrices means that special care must be given to the solution of the generalized eigenvalue problem. Indeed, if the null spaces of G and H have a nontrivial intersection, i. e., $\operatorname{Ker}(A) \bigcap \operatorname{Ker}(B) \neq 0$, then the problem is ill posed and a regularization technique is needed to solve the eigenvalue problem. Mangasarian et al. [2] proposes to use Tikhonov regularization applied to a twofold problem:

$$\min_{w,\gamma \neq 0} \frac{\|Aw - e\gamma\|^2 + \delta\|z\|^2}{\|Bw - e\gamma\|^2} \tag{12}$$

and

$$\min_{w,\gamma \neq 0} \frac{\|Bw - e\gamma\|^2 + \delta\|z\|^2}{\|Aw - e\gamma\|^2} \,, \tag{13}$$

where δ is the regularization parameter and the new problems are still convex. The minimum eigenvalues-eigenvectors of these problems are approximations of the minimum and maximum eigenvalues-eigenvectors of (4). The solutions (w_i, γ_i), $i = 1, 2$ to (12) and (13) represent the two hyperplanes approximating the two classes of training points. The same regularization technique can be applied to the nonlinear formulation.

Another Algorithm

It is possible to solve the problem without regularization. In practice, if $\beta G - \alpha H$ is nonsingular for every α and β, it is possible to transform the problem into another problem that is nonsingular and that has the same eigenvectors of the initial one. We start with the following theorem whose proof can be found in [5], p. 288.

Theorem 1 *Consider the generalized eigenvalue problem $Gx = \lambda Hx$ and the transformed $G^* x = \lambda H^* x$ defined by*

$$G^* = \tau_1 G - \delta_1 H, \quad H^* = \tau_2 H - \delta_2 G \tag{14}$$

for each choice of scalars τ_1, τ_2, δ_1, and δ_2 such that the 2×2 matrix

$$\Omega = \begin{pmatrix} \tau_2 & \delta_1 \\ \delta_2 & \tau_1 \end{pmatrix} \tag{15}$$

is nonsingular. Then the problem $G^*x = \lambda H^*x$ has the same eigenvectors of the problem $Gx = \lambda Hx$. An associated eigenvalue λ^* of the transformed problem is related to an eigenvalue λ of the original problem by

$$\lambda = \frac{\tau_2\lambda^* + \delta_1}{\tau_1 + \delta_2\lambda^*} .$$

In the linear case, Theorem 1 can be applied. By setting $\tau_1 = \tau_2 = 1$ and $\hat{\delta}_1 = -\delta_1$, $\hat{\delta}_2 = -\delta_2$, the regularized problem becomes

$$\min_{w,\gamma\neq 0} \frac{\|Aw - e\gamma\|^2 + \hat{\delta}_1\|Bw - e\gamma\|^2}{\|Bw - e\gamma\|^2 + \hat{\delta}_2\|Aw - e\gamma\|^2} . \qquad (16)$$

If $\hat{\delta}_1$and $\hat{\delta}_2$ are nonnegative, Ω is nondegenerate. The spectrum is now shifted and inverted so that the minimum eigenvalue of the original problem becomes the maximum of the regularized one, and the maximum becomes the minimum eigenvalue. Choosing the eigenvectors related to the new minimum and maximum eigenvalue, we obtain the same solution of the original problem.

This regularization works for the linear case if we suppose that in each class of the training set there is a number of linearly independent rows that is at least equal to the number of the features. This is often the case and, if the number of points in the training set is much greater than the number of features, Ker(G) and Ker(H) have both dimension 1. In this case, the probability of a nontrivial intersection is zero.

In the nonlinear case the situation is different. Guarracino et al. [1] propose to generate the two proximal surfaces

$$K(x, C)u_1 - \gamma_1 = 0, \quad K(x, C)u_2 - \gamma_2 = 0 \qquad (17)$$

by solving the following problem

$$\min_{u,\gamma\neq 0} \frac{\|K(A, C)u - e\gamma\|^2 + \delta\|\tilde{K}_B u - e\gamma\|^2}{\|K(B, C)u - e\gamma\|^2 + \delta\|\tilde{K}_A u - e\gamma\|^2} , \qquad (18)$$

where $\tilde{K}_A$ and $\tilde{K}_B$ are diagonal matrices with the diagonal entries from the matrices $K(A,C)$ and $K(B,C)$. The perturbation theory of eigenvalue problems [6] provides an estimation of the distance between the original and the regularized eigenvectors. If we call z an eigenvector of the initial problem and $z(\delta)$ the corresponding one in the regularized problem, then $|z - z(\delta)| = \mathcal{O}(\delta)$, which means their closeness is in the order of δ.

References

1. Guarracino MR, Cifarelli C, Seref O, Pardalos PM (2007) A classification algorithm based on generalized eigenvalue problems. Optim Methods Softw 22(1):73–81
2. Mangasarian OL, Wild EW (2004) Multisurface Proximal Support Vector Classification via Generalized Eigenvalues. Data Mining Institute
3. Mitchell T (1997) Machine Learning, McGraw-Hill Education (ISE Editions) http://www.amazon.co.uk/exec/obidos/ASIN/0071154671/citeulike-21
4. Parlett BN (1998) The Symmetric Eigenvalue Problem. SIAM, Philadelphia, pp 343–345
5. Saad Y (1992) Numerical Methods for Large Eigenvalue Problems. Halsted, New York
6. Wilkinson J (1965) The Algebraic Eigenvalue Problem. Clarendon, Oxford

Generalized Geometric Programming: Mixed Continuous and Discrete Free Variables

HAN-LIN LI[1], JUNG-FA TSAI[2]

[1] Institute of Information Management,
National Chiao Tung University, Hsinchu, Taiwan

[2] Department of Business Management, National
Taipei University of Technology, Taipei, Taiwan

MSC2000: 49M37, 90C11, 90C30

Article Outline

Keywords and Phrases

Global optimization; Generalized geometric programming; Free variables; Convexification

Introduction

Generalized geometric programming (GGP) problems with continuous and discrete variables occur quite frequently in various fields such as civil and material engineering design, chemical engineering, location-allocation, inventory control, production planning, and scheduling etc. These applications are extensively surveyed in Floudas and Pardalos [11] and Floudas [9]. Biegler and Grossmann [3] provided a retrospective on optimization techniques that have been applied in process systems engineering. They indicated that design and synthesis problems have been dominated by nonlinear programming (NLP) and mixed-integer nonlinear programming (MINLP) models. Although MINLP programs appear in many chemical engineering problems, they are often nonconvex and no direct optimization method is available to guarantee global optimality [21]. With the increasing reliance on mathematical programming based approaches in chemical engineering problems, the need for finding global optimum is paramount.

The developed methods for GGP problems with continuous and discrete variables can be divided into two approaches.

(i) Stochastic methods: The stochastic methods involve random elements in their search and rely on a statistical argument to prove their convergence. For instance, Salcedo et al. [23] proposed an improved random search algorithm for solving nonlinear optimization problems. Cardoso et al. [5] solved nonconvex nonlinear integer programming problems with simulated annealing. Yiu et al. [30] developed a hybrid descent approach based on a simulated annealing algorithm and a gradient-based method to solve multidimensional nonconvex continuous optimization problems. Hussain and Al-Sultan [15] proposed a hybrid algorithm for nonconvex function minimization by utilizing the genetic technique to generate search directions. These stochastic methods mentioned above cannot guarantee to find the global optimum. Therefore, the quality of the solution is not ensured. Moreover, the probability of finding the global solution decreases when the problem size increases.

(ii) Deterministic methods: Mathematical methods that generate convex underestimators for twice differentiable constrained nonconvex optimization problems are of primary importance in deterministic global optimization [9]. The α BB global optimization algorithm [1,2,9] is a power approach for constructing such convex underestimators for nonconvex functions [10]. In a general survey of optimization techniques ([3,13,14]), many deterministic methods for convex MINLP problems have been reviewed. The methods include Branch and Bound (BB) ([17,24]), Generalized Benders Decomposition (GBD) [12], Outer-Approximation (OA) ([6,7,22]), Extended Cutting Plane Method (ECP) [28], and Generalized Disjunctive Programming (GDP) [16]. One possible approach to circumvent the nonconvex objective function or the nonconvex constraints in MINLP models is transformation. Floudas ([8,9]), Floudas and Pardalos [11] and Maranas and Floudas [20] proposed exponential transformation methods to treat GGP problems with continuous and discrete variables. The core concept of their methods is to convert the problem into a new problem where both the constraints and the objective are decomposed into the difference of two convex functions. By utilizing exponential variable transformations, each signomial term $z = x_1^{\alpha} x_2^{\beta}$, where x_1 and x_2 are positive, can be transferred into an exponential term $z' = e^{\alpha \ln x_1 + \beta \ln x_2}$. However, the exponential transformation technique can only be applied to strictly positive variables and is thus unable to deal with nonconvex GGP problems with continuous and discrete free variables.

Although positive variables are adopted frequently to represent engineering and scientific systems, it is also common to introduce free variables to model the system behavior, such as stresses, temperatures, electrical currents, velocities and accelerations, etc. In general, the values accepted by the machines are under a discrete space. For instance, a controller can only increase temperature from a fixed initial point to a set of fixed points at a fixed interval. Consequently, deriving a global optimum for the GGP problem with continuous and discrete free variables is essential for real applications. Li and Tsai [18] proposed a technique for treating free continuous variables in GGP problems. Pörn et al. [21] introduced different convexification strategies for MINLP problems with both polynomial and negative binomial terms in the constraints. They suggested a simple translation, $x + \tau = e^x$, to treat a free variable x. However, inserting the transformed result into the original signomial term will bring additional signomial terms and therefore increasing the computation burden. This study proposes a method for solving a GGP problem with continuous and discrete free variables to obtain a global optimal solution. The GGP problem is first transformed into another one containing only positive variables. Then the transformed problem is reformulated as a convex mixed-integer program. A global optimum of the GGP problem with continuous and discrete free variables can finally be found within the tolerable error. Furthermore, this study develops several convexification strategies for signomial terms so that the efficiency of the optimization approach can be enhanced. The right choice of transformation for convexification of nonconvex signomial terms might significantly decrease the solution time [4]. By employing the proposed rules, certain classes of signomial terms can be determined as convex terms and do not require any transformation. Moreover, some nonconvex signomial terms with specific features can be transformed into convex terms in accordance with the proposed rules by replacing some variables, thereby making the resulting problem a computationally efficient model.

Formulation

The mathematical formulation of a GGP problem with continuous and discrete free variables is expressed as follows:

GGP:

$$\begin{aligned} &\text{Minimize} && f(X,Y) \\ &\text{subject to} && g_t(X,Y) \le 0 \quad t=1,\dots,T, \\ &&& X=(x_1,\dots,x_p,x_{p+1}\dots,x_n), \\ &&& \underline{x}_i \le x_i \le \overline{x}_i, \\ &&& Y=(y_1,\dots,y_q,y_{q+1}\dots,y_m), \\ &&& \underline{y}_j \le y_j \le \overline{y}_j, \end{aligned}$$

where $x_i \in \Re^+$ for $1 \le i \le p$, x_i are bounded free variables for $p+1 \le i \le n$, y_j are positive integer/discrete variables for $1 \le j \le q$, y_j are bounded free variables for $q+1 \le j \le m$, $f(X,Y)$ and $g_t(X,Y)$ are mixed-integer signomial functions, $\underline{x}_i$ and $\overline{x}_i$ are lower and upper bounds of the continuous variable x_i, and $\underline{y}_j$ and $\overline{y}_j$ are lower and upper bounds of the integer/ discrete variable y_j, respectively.

Methods

Treating Free Variables. Li and Tsai [18] proposed a technique for treating free continuous variables in GGP problems. By integrating Li and Tsai method with the approach of dealing with free discrete variables described below, a GGP problem with continuous and discrete free variables can be equivalently transform into a mixed-integer GGP program with positive variables. The following illustrates how to convert free discrete variables into non-positive discrete variables.

$$\text{Let: } y_j = y_j^+ - y_j^-,\ \ y_j^+, y_j^- \ge 0, \quad \text{for } \ j=q+1,\cdots,m\,.$$

And a nonlinear term $y_j^{\beta_j}$ is expressed as

$$y_j^{\beta_j} = (y_j^+)^{\beta_j} + (-1)^{\beta_j}(y_j^-)^{\beta_j}, \quad \beta_j \in \text{ integer, for } j=q+1,\dots,m.$$

If $y_j^+ > 0$ and $y_j^- = 0$, then y_j is positive. Otherwise, if $y_j^- > 0$ and $y_j^+ = 0$, then y_j is negative. To prohibit from yielding positive values for y_j^+ and y_j^- simultaneously, we have the following remark.

Remark 1 A free discrete variable y_j can be expressed as $y_j = y_j^+ - y_j^-$, $y_j^+, y_j^- \ge 0$, and y_j^+ and y_j^- will not be positive concurrently by the following inequalities.

$$\begin{aligned} &(i)\ -y_j^- \le y_j \le M\theta_j - y_j^-, \\ &(ii)\ M(\theta_j - 1) + y_j^+ \le y_j \le y_j^+. \end{aligned}$$

M is a sufficiently large positive number and $\theta_j \in \{0,1\}$.

By means of changing variables, the GGP problem with free variables can be equivalently solved with another one having non-negative variables. The next is to deal with discrete variables containing zero, consider the following propositions:

Proposition 1 *[21] For positive discrete variables $y_j \in \{d_{j1}, d_{j2}, \cdots, d_{jm_j}\}$ where $d_{j,i+1} > d_{ji} > 0$ for $i = 1,2,\cdots,m_j - 1$, a product term $y_1^{\alpha_1} y_2^{\alpha_2} \cdots y_m^{\alpha_m}$ where $\alpha_1, \alpha_2, \cdots, \alpha_m$ are real constants can be transformed into a function $e^{\alpha_1 z_1 + \cdots + \alpha_m z_m}$ where $z_j = \ln d_{j1} + \sum_{i=1}^{m_j-1} u_{ji}(\ln d_{j,i+1} - \ln d_{j1})$, $\sum_{i=1}^{m_j-1} u_{ji} \le 1$ for $u_{ji} \in \{0,1\}$.*

Proof Let $y_j = e^{z_j}$ and $z_j = \ln y_j$, expressing y_j as $y_j = d_{j1} + \sum_{i=1}^{m_j-1} u_{ji}(d_{j,i+1} - d_{j1})$, $\sum_{i=1}^{m_j-1} u_{ji} \le 1$, where $u_{ji} \in \{0,1\}$.

We then have $y_1^{\alpha_1} y_2^{\alpha_2} \cdots y_m^{\alpha_m} = e^{\alpha_1 z_1 + \cdots + \alpha_m z_m}$ and $z_j = \ln d_{j1} + \sum_{i=1}^{m_j-1} u_{ji}(\ln d_{j,i+1} - \ln d_{j1})$, $\sum_{i=1}^{m_j-1} u_{ji} \leq 1$, for $u_{ji} \in \{0,1\}$. □

Because some variables y_j in Proposition 1 may have zero value, Proposition 1 needs to be modified as the following proposition:

Proposition 2 *For positive discrete variables $y_j \in \{d_{j1}, d_{j2}, \cdots, d_{jm_j}\}$ where $d_{j,i+1} > d_{ji} > 0$ for $i = 1, 2, \cdots, m_j - 1$, $1 \leq j \leq q$, and non-negative discrete variables $y_j \in \{0, d_{j1}, d_{j2}, \cdots, d_{jm_j}\}$ where $d_{j,i+1} > d_{ji} > 0$ for $i = 1, 2, \cdots, m_j - 1$, $q+1 \leq j \leq m$, a product term $s = y_1^{\alpha_1} y_2^{\alpha_2} \cdots y_q^{\alpha_q} y_{q+1}^{\alpha_{q+1}} \cdots y_m^{\alpha_m}$ can be expressed as*

$$(i)\ 0 \leq s \leq \bar{s}\left(\sum_{i=1}^{m_j} u_{ji}\right), \textit{ for } q+1 \leq j \leq m,$$

$$(ii)\ \bar{s}\left(\sum_{j=q+1}^{m}\sum_{i=1}^{m_j} u_{ji} - (m-q)\right) + e^{\alpha_1 z_1 + \cdots + \alpha_m z_m} \leq s$$

$$\leq \bar{s}\left((m-q) - \sum_{j=q+1}^{m}\sum_{i=1}^{m_j} u_{ji}\right) + L(e^{\alpha_1 z_1 + \cdots + \alpha_m z_m}),$$

where $y_j = d_{j1} + \sum_{\mathrm{i=1}}^{\mathrm{m_j-1}} u_{ji}(d_{j,i+1} - d_{j1})$, $z_j = \ln d_{j1} + \sum_{\mathrm{i=1}}^{\mathrm{m_j-1}} u_{ji}(\ln d_{j,i+1} - \ln d_{j1})$, $\sum_{\mathrm{i=1}}^{\mathrm{m_j-1}} u_{ji} \leq 1$, $u_{ji} \in \{0,1\}$, for $1 \leq j \leq q$, and $y_j = \sum_{\mathrm{i=1}}^{\mathrm{m_j}} u_{ji} d_{ji}$, $z_j = \sum_{\mathrm{i=1}}^{\mathrm{m_j}} u_{ji}(\ln d_{ji})$, $\sum_{\mathrm{i=1}}^{\mathrm{m_j}} u_{ji} \leq 1$, $u_{ji} \in \{0,1\}$ for $q+1 \leq j \leq m$, $L(e^{\alpha_1 z_1 + \cdots + \alpha_m z_m})$ is a piecewisely linearized expression of $e^{\alpha_1 z_1 + \cdots + \alpha_m z_m}$, and $\bar{s}$ is the upper bound of s.

Proof If there is $y_j = 0$ for some j $(q+1 \leq j \leq m)$, then $\sum_{\mathrm{i=1}}^{\mathrm{m_j}} u_{ji} = 0$ and $s = 0$ by (i).

If $y_j > 0$ for all $j = q+1, \cdots, m$, then $\sum_{\mathrm{i=1}}^{\mathrm{m_j}} u_{ji} = 1$ for $j = q+1, \cdots, m$. Therefore we have $\sum_{\mathrm{j=q+1}}^{\mathrm{m}} \sum_{\mathrm{i=1}}^{\mathrm{m_j}} u_{ji} - (m-q) = 0$ if all variables in the signomial term are not zero, and this implies $s = e^{\alpha_1 z_1 + \cdots + \alpha_m z_m}$ according to (ii). □

Remark 2 For a non-negative discrete variable y, $y \in \{d_1, d_2, \cdots, d_m\}$, $0 \leq d_1 < d_2 < \cdots < d_m$, the exponential term y^α where α is a real constant can be represented as

$$y^\alpha = d_1^\alpha + \sum_{i=1}^{m-1} u_i(d_{i+1}^\alpha - d_1^\alpha) \quad \text{where} \quad \sum_{i=1}^{m-1} u_i \leq 1,$$
$$u_i \in \{0,1\}.$$

According to the above discussions, free discrete variables in GGP can be converted into positive discrete variables. In addition, Li and Tsai method [18] can deal with the free continuous variables. Consequently, the GGP program with continuous and discrete free variables can be transformed into a GGP program with only positive variables. In order to obtain a global optimum of the transformed GGP program, it is required to be converted into a convex mixed-integer problem which is solvable by the conventional convex mixed-integer techniques to derive a globally optimal solution.

Convexification Strategies. Convexification strategies for signomial terms are important techniques for global optimization problems. Sun et al. [25] proposed a convexification method for a class of global optimization problems with monotone functions under some restrictive conditions. Wu et al. [29] developed a more general convexification and concavification transformation for solving a general global optimization problem with certain monotone properties. With different convexification approaches, an MINLP problem can be reformulated into another convex mixed-integer program solvable to obtain an approximately global optimum. Björk et al. [4] proposed a global optimization technique based on convexifying signomial terms. They discussed that the right choice of transformation for convexifying nonconvex signomial terms has a clear impact on the efficiency of the optimization approach. Tsai et al. [26] also suggested convexification techniques for the signomial terms with three variables. This study presents generalized convexification techniques and rules to transform a nonconvex GGP program with continuous and discrete variables into a convex mixed-integer program. Consider the following propositions:

Lemma 1 *For a twice-differentiable function $f(X) = c\prod_{i=1}^{n} x_i^{\alpha_i}$, $X = (x_1, x_2, \cdots, x_n)$, $c, x_i, \alpha_i \in \Re$, $\forall i$, let $H_i(X)$ be the ith principal minor of a Hessian matrix $H(X)$ of $f(X)$. The determinant of $H_i(X)$ can be expressed as* $\det H_i(x) =$

$$(-c)^i\left(\prod_{j\in J_i}^{i} \alpha_j x_j^{i\alpha_j - 2}\right)\left(\prod_{\substack{j\notin J_i\\ J_i\neq\Phi}}^{n} x_j^{i\alpha_j}\right)\left(1 - \sum_{j\in J_i}\alpha_j\right).$$

Remark 3 If $c \geq 0$, $x_i \geq 0$ and $\alpha_i \leq 0$ (for all i), then $\det H_i(x) \geq 0$.

Remark 4 If $c < 0$, $x_i \geq 0$, $\alpha_i \geq 0$ (for all i), and, $1 - \sum_{i=1}^{n} \alpha_j \geq 0$, then $\det H_i(x) \geq 0$.

Proposition 3 *A twice-differentiable function* $f(X) = c \prod_{i=1}^{n} x_i^{\alpha_i}$ *is convex for* $c, x_i \geq 0, \alpha_i \leq 0, i = 1, 2, \cdots, n$.

Proof By Lemma 1 and Remark 3, $\det H_i(x) =$

$$(-c)^i \left(\prod_{j \in J_i}^{i} \alpha_j x_j^{i\alpha_j - 2} \right) \left(\prod_{\substack{j \notin J_i \\ J_i \neq \Phi}}^{n} x_j^{i\alpha_j} \right) \left(1 - \sum_{j \in J_i} \alpha_j \right)$$

≥ 0 for $i = 1, 2, \cdots, n$, when $c, x_i \geq 0$, $\alpha_i \leq 0$, $i = 1, 2, \cdots, n$. Since $\det H_i(x) \geq 0$ for all i, $H_i(X)$ is positive semi-definite and $f(X)$ is convex. □

Proposition 4 *A twice-differentiable function* $f(X) = c \prod_{i=1}^{n} x_i^{\alpha_i}$ *is convex for* $c < 0$, $x_i, \alpha_i \geq 0$ *(for* $i = 1, 2, \cdots, n$*), and* $1 - \sum_{\mathrm{i}=1}^{\mathrm{n}} \alpha_i \geq 0$.

Proof By Lemma 1 and Remark 4, $\det H_i(x) =$

$$(-c)^i \left(\prod_{j \in J_i}^{i} \alpha_j x_j^{i\alpha_j - 2} \right) \left(\prod_{\substack{j \notin J_i \\ J_i \neq \Phi}}^{n} x_j^{i\alpha_j} \right) \left(1 - \sum_{j \in J_i} \alpha_j \right)$$

≥ 0 for $i = 1, 2, \cdots, n$, when $c < 0$, $x_i, \alpha_i \geq 0$, and $1 - \sum_{\mathrm{i}=1}^{\mathrm{n}} \alpha_i \geq 0$. Since $\det H_i(x) \geq 0$ for all i, $H_i(X)$ is positive semi-definite and $f(X)$ is convex. □

For a given signomial term s, if s can be converted into a set of convex terms satisfying Proposition 3 and 4, then the whole solution process is more computationally efficient. Under this condition, s does not necessitate the exponential transformation. For instance, $s = x_1^{-1} x_2^{-2} x_3^{-1}$ with $x_1, x_2, x_3 \geq 0$ is a convex term requirbreaking no transformation by Proposition 3, and $s = -x_1^{0.2} x_2^{0.7}$ with $x_1, x_2 \geq 0$ is also a convex term by Proposition 4.

Remark 5 A product term $z = uf(x)$ is equivalent to the following linear inequalities:

$$(i)\ M(u-1) + f(x) \leq z \leq M(1-u) + f(x),$$
$$(ii)\ -Mu \leq z \leq Mu,$$

where $u \in \{0, 1\}$, z is an unrestricted in sign variable, and $M = \max f(x)$ is a large constant.

Remark 6 The product term $u_1 u_2 \cdots u_m$ where $u_i \in \{0, 1\}$ for $i = 1, 2, \cdots, m$ can be replaced by a variable u expressed as

$$(i)\ 0 \leq u \leq u_i, \text{ for } i = 1, 2, \cdots, m,$$
$$(ii)\ u \geq \sum_{i=1}^{m} u_i - m + 1.$$

Following the above discussions, herein we take a signomial term with three variables for instance to describe the strategy of convexification. The strategy can also be extended to convexity a signomial term containing n variables.

Consider a signomial term $cx_1^{\alpha} x_2^{\beta} x_3^{\gamma}$ composed of three positive variables, the term $cx_1^{\alpha} x_2^{\beta} x_3^{\gamma}$ can be convexified by the following rules:

Rule 1 If $c > 0$, $\alpha, \beta, \gamma < 0$, then $cx_1^{\alpha} x_2^{\beta} x_3^{\gamma}$ is already a convex term by Proposition 3.

Rule 2 If $c > 0$, $\alpha, \beta < 0$, and $\gamma > 0$, then let $cx_1^{\alpha} x_2^{\beta} x_3^{\gamma} = cx_1^{\alpha} x_2^{\beta} z_1^{-\gamma}$ where $z_1 = x_3^{-1}$. The term $cx_1^{\alpha} x_2^{\beta} z_1^{-\gamma}$ is convex by Rule 1.

Rule 3 If $c > 0$, $\alpha < 0$, and $\beta, \gamma > 0$, then let $cx_1^{\alpha} x_2^{\beta} x_3^{\gamma} = cx_1^{\alpha} z_1^{-\beta} z_2^{-\gamma}$ where $z_1 = x_2^{-1}, z_2 = x_3^{-1}$. The term $cx_1^{\alpha} z_1^{-\beta} z_2^{-\gamma}$ is convex by Rule 1.

Rule 4 If $c > 0$ and $\alpha, \beta, \gamma > 0$, then let $cx_1^{\alpha} x_2^{\beta} x_3^{\gamma} = ce^{\alpha \ln x_1 + \beta \ln x_2 + \gamma \ln x_3}$.

Rule 5 If $c < 0, \alpha, \beta, \gamma \geq 0$, and $\alpha + \beta + \gamma \leq 1$, then $cx_1^{\alpha} x_2^{\beta} x_3^{\gamma}$ is already a convex term by Proposition 4.

Rule 6 If $c < 0$, $\alpha, \beta > 0$, $\alpha + \beta < 1$, then let $cx_1^{\alpha} x_2^{\beta} x_3^{\gamma} = cx_1^{\alpha} x_2^{\beta} z_1^{1-\alpha-\beta}$ where $z_1 = x_3^{\gamma/(1-\alpha-\beta)}$. The term $cx_1^{\alpha} x_2^{\beta} z_1^{1-\alpha-\beta}$ is convex by Rule 5.

Rule 7 If $c < 0$, $0 < \alpha < 1$, then let $cx_1^{\alpha} x_2^{\beta} x_3^{\gamma} = cx_1^{\alpha} z_1^{(1-\alpha)/2} z_2^{(1-\alpha)/2}$ where $z_1 = x_2^{2\beta/(1-\alpha)}$ and $z_2 = x_3^{2\gamma/(1-\alpha)}$. The term $cx_1^{\alpha} z_1^{(1-\alpha)/2} z_2^{(1-\alpha)/2}$ is convex by Rule 5.

Rule 8 If $c < 0$ and "$\alpha, \beta, \gamma < 0$ or $\alpha, \beta, \gamma \geq 1$", then let $cx_1^{\alpha} x_2^{\beta} x_3^{\gamma} = cz_1^{\frac{1}{3}} z_2^{\frac{1}{3}} z_3^{\frac{1}{3}}$ where $z_1 = x_1^{3\alpha}, z_2 = x_2^{3\beta}$, and $z_3 = x_3^{3\gamma}$. The term $cz_1^{\frac{1}{3}} z_2^{\frac{1}{3}} z_3^{\frac{1}{3}}$ is convex by Rule 5.

Rule 9 If $\alpha, \beta > 0$, $x_1 \in Z$, $x_3 = 1$ and $\alpha + \beta > 1$, then let $cx_1^{\alpha} x_2^{\beta} = c\big[d_{11}^{\alpha} + \sum_{i=1}^{m_1-1} u_{1i}(d_{1,i+1}^{\alpha} - d_{11}^{\alpha})\big]x_2^{\beta}$ for $i \in \{1, 2, \cdots, m_1 - 1\}$. By Remark 5, the product term $u_{1i} x_2^{\beta}$ can be transformed into linear inequalities.

By applying the proposed rules, we can determine certain classes of signomial terms are convex and do not necessitate any transformation. Besides, we can transform a nonconvex signomial term into a convex term accordance with the proposed rules by replacing some variables, thereby decreasing the number of concave functions requiring to be estimated and making the resulting problem a computationally efficient model.

In order to be a valid transformation in the global optimization procedure, the transformation should be selected such that the signomial terms are not only convexified but also underestimated [4,21,27]). If the transformations are appropriately selected, the corresponding approximate signomial term will underestimate the original convexified signomial term by applying piecewise linear approximations to the inverse transformation functions. We examine the proposed rules can satisfy the underestimating condition as follows:

In Rule 2, let $\hat{z}_1$ be the approximate transformation variable obtained from piecewise linear function of $z_1 = x_3^{-1}$. The inverse transformation $z_1 = x_3^{-1} (x_3 > 0)$ is convex and z_1 will be overestimated $(\hat{z}_1 > z_1)$. When inserting the approximate variable in the signomial term, we find the underestimating property $cx_1^{\alpha} x_2^{\beta} \hat{z}_1^{-\gamma} \leq cx_1^{\alpha} x_2^{\beta} z_1^{-\gamma}$ is fulfilled since $c > 0$ and z_1 has a negative power in the convexified term. Similarly, Rules 3 and 4 meet the underestimating condition.

In Rule 6, let $\hat{z}_1$ be the approximate transformation variable obtained from piecewise linear function of $z_1 = x_3^{\gamma/(1-\alpha-\beta)}$. The inverse transformation $z_1 = x_3^{\gamma/(1-\alpha-\beta)}$ $(x_3 > 0, \frac{\gamma}{1-\alpha-\beta} > 1$ or $\frac{\gamma}{1-\alpha-\beta} \leq 0)$ is convex and z_1 will be overestimated $(\hat{z}_1 > z_1)$. When inserting the approximate variable in the signomial term, we find the underestimating property $cx_1^{\alpha} x_2^{\beta} \hat{z}_1^{1-\alpha-\beta} \leq cx_1^{\alpha} x_2^{\beta} z_1^{1-\alpha-\beta}$ is fulfilled since $c < 0$ and z_1 has a positive power in the convexified term. Similarly, Rules 7 and 8 satisfy the underestimating property.

From above discussions, we observe the proposed rules not only convexity but underestimate the convexified signomial term. Consequently, utilizing the transformations in the global optimization of a GGP problems, the feasible region of the convexified problem overestimates the feasible region of the original nonconvex problem.

Case Studies

Case1 Minimize $x_1^3 x_2^{1.5} x_3^3 + x_2^{5.5} x_3 + x_1^5$

subject to

$$3x_1 + 2x_2 - x_3 \leq 7,$$
$$-5 \leq x_1 \leq 2,\ 0 \leq x_2 \leq 4,\ -5 \leq x_3 \leq -1,$$
$$x_1, x_2 \in \mathrm{Z}, \quad x_3 \in \Re.$$

This problem is a nonconvex GGP program with continuous and discrete free variables. Current exponential transformation methods [8,9,11,20]) developed for solving mixed-integer GGP problems can not be adopted to treat this kind of problems. By employing the proposed method, we first utilize a straightforward substitution for the free variables to make the GGP problem with only non-negative variables. By Li and Tsai [18], let the free continuous variable $x_3 = -x_3^-$, $x_3^- \geq 0$.

The free discrete variable can be transformed by Remark 1, $x_1 = x_1^+ - x_1^-$, $x_1^+, x_1^- \geq 0$. The original problem becomes as follows:

$$\begin{aligned}
\text{Minimize}\quad & -(x_1^+)^3 x_2^{1.5} (x_3^-)^3 + (x_1^-)^3 x_2^{1.5} (x_3^-)^3 - \\
& x_2^{5.5} x_3^- + (x_1^+)^5 - (x_1^-)^5 \\
\text{subject to}\quad & 3x_1^+ - 3x_1^- + 2x_2 + x_3^- \leq 7, \\
& 0 \leq x_1^+ \leq 2,\ 0 \leq x_1^- \leq 5.0 \leq x_2 \leq 4, \\
& 1 \leq x_3^- \leq 5, \quad x_1^+, x_1^-, x_2 \in \mathrm{Z}, \quad x_3^- \in \Re.
\end{aligned}$$

Then, we use the proposed convexification rules to transform all signomial terms into convex terms as follows:

(i) $z_1 = (x_1^+)^3 x_2^{1.5} (x_3^-)^3$ and $z_2 = (x_1^-)^3 x_2^{1.5} (x_3^-)^3$ are transformed by Rule 4 and Proposition 2.

(ii) $-x_2^{5.5} x_3^-$ is convexified as $-x_2^{5.5} x_3^- = -(u_{21} + 2^{5.5} u_{22} + 3^{5.5} u_{23} + 4^{5.5} u_{24}) x_3^- = -z_3 - 2^{5.5} z_4 - 3^{5.5} z_5 - 4^{5.5} z_6$ by Rule 9.

(iii) $(x_1^+)^5$ and $-(x_1^-)^5$ are treated directly as $(x_1^+)^5 = u_{11}^+ + 2^5 u_{12}^+, -(x_1^-)^5 = -u_{11}^- - 2^5 u_{12}^- - 3^5 u_{13}^- - 4^5 u_{14}^- - 5^5 u_{15}^-$ by Remark 2.

Subsequently, the transformed program is presented as a convex mixed-integer program below:

Minimize

$-z_1 + z_2 - z_3 - 2^{5.5} z_4 - 3^{5.5} z_5 - 4^{5.5} z_6 + z_7 - z_8$

subject to

$3x_1^+ - 3x_1^- + 2x_2 + x_3^- \leq 7,$

$-x_1^- \leq x_1 \leq 5\theta_1 - x_1^-,$

$5(\theta_1 - 1) + x_1^+ \leq x_1 \leq x_1^+,$

$x_1^+ = u_{11}^+ + 2u_{12}^+, \quad y_1^+ = u_{12}^+ \cdot \ln 2,$

$x_1^- = u_{11}^- + 2u_{12}^- + 3u_{13}^- + 4u_{14}^- + 5u_{15}^-,$

$y_1^- = u_{12}^- \cdot \ln 2 + u_{13}^- \cdot \ln 3 + u_{14}^- \cdot \ln 4 + u_{15}^- \cdot \ln 5,$

$u_{11}^+ + u_{12}^+ \leq 1, u_{11}^- + u_{12}^- + u_{13}^- + u_{14}^- + u_{15}^- \leq 1,$

$x_2 = u_{21} + 2u_{22} + 3u_{23} + 4u_{24},$

$y_2 = u_{22} \cdot \ln 2 + u_{23} \cdot \ln 3 + u_{24} \cdot \ln 4,$

$u_{21} + u_{22} + u_{23} + u_{24} \leq 1,$

$y_3^- = L(\ln x_3^-),$

$0 \leq z_1 \leq \bar{z}(u_{11}^+ + u_{12}^+),$

$0 \leq z_1 \leq \bar{z}(u_{21} + u_{22} + u_{23} + u_{24}),$

$\bar{z}(u_{11}^+ + u_{12}^+ + u_{21} + u_{22} + u_{23} + u_{24} - 2)$
$+ e^{3y_1^+ + 1.5y_2 + 3y_3^-} \leq z_1,$

$z_1 \leq \bar{z}\,(2 - (u_{11}^+ + u_{12}^+ + u_{21} + u_{22} + u_{23} + u_{24}))$
$+ L(e^{3y_1^+ + 1.5y_2 + 3y_3^-}),$

$0 \leq z_2 \leq \bar{z}(u_{11}^- + u_{12}^- + u_{13}^- + u_{14}^- + u_{15}^-),$

$0 \leq z_2 \leq \bar{z}(u_{21} + u_{22} + u_{23} + u_{24}),$

$\bar{z}(u_{11}^- + u_{12}^- + u_{13}^- + u_{14}^- + u_{15}^- + u_{21} + u_{22} + u_{23}$
$+ u_{24} - 2) + e^{3y_1^- + 1.5y_2 + 3y_3^-} \leq z_2,$

$z_2 \leq \bar{z}(2 - (u_{11}^- + u_{12}^- + u_{13}^- + u_{14}^- + u_{15}^- + u_{21}$
$+ u_{22} + u_{23} + u_{24})) + L(e^{3y_1^- + 1.5y_2 + 3y_3^-}),$

$5(u_{21}^- 1) + x_3^- \leq z_3 \leq x_3^-, \quad 0 \leq z_3 \leq 5u_{21},$

$5(u_{22}^- 1) + x_3^- \leq z_4 \leq x_3^-, \quad 0 \leq z_4 \leq 5u_{22},$

$5(u_{23}^- 1) + x_3^- \leq z_5 \leq x_3^-, \quad 0 \leq z_5 \leq 5u_{23},$

$5(u_{24}^- 1) + x_3^- \leq z_6 \leq x_3^-, \quad 0 \leq z_6 \leq 5u_{24},$

$z_7 = u_{11}^+ + 2^5 u_{12}^+,$

$z_8 = u_{11}^- + 2^5 u_{12}^- + 3^5 u_{13}^- + 4^5 u_{14}^- + 5^5 u_{15}^-,$

$(0, 0, 0, 1, 0, 0, 0) \leq (x_1^+, x_1^-, x_2, x_3^-, y_1^+, y_1^-, y_2, y_3^-)$
$\leq (2, 5, 4, 5, \ln 2, \ln 5, \ln 4, \ln 5),$

where $u_{ij}, u_{ij}^+, u_{ij}^-, \theta_1 \in \{0, 1\},$ and $\bar{z} = 125{,}000.$

Solving the original problem without any variable transformation and convexification by LINGO [19], a local optimum obtained is $(x_1, x_2, x_3) = (-5, 0, -5)$ and the objective value is –3125. However, solving the above transformed convex mixed-integer program within the tolerable error 0.001, the globally optimal solution obtained is $(x_1, x_2, x_3) = (-2, 4, -3.266)$ and the objective value found is –4491.16.

Case2 Minimize $x_1^{0.5} x_2 + 3 \ln x_1$ subject to

$-x_1 + x_2 \leq 5$

$x_1^{0.5} y - x_2 \leq 6,$

$x_1 \in \{0.1, 0.5, 0.7, 1.2\}, -6 \leq x_2 \leq 4, \quad y \in \{0, 1\}.$

This problem contains a discrete variable, a free continuous variable and a binary variable which cannot be treated by the exponential-based methods. The nonlinear terms $x_1^{0.5} x_2$, $3 \ln x_1$ and $x_1^{0.5} y$ are nonconvex functions. By Remarks 2, 5 and 6, the problem can be equivalently transformed into a linear mixed-integer programming problem as follows.

Minimize

$0.1^{0.5} x_2 + (0.5^{0.5} - 0.1^{0.5}) s_1 + (0.7^{0.5} - 0.1^{0.5}) s_2 +$
$(1.2^{0.5} - 0.1^{0.5}) s_3 + 3(\ln 0.1 + (\ln 0.5 - \ln 0.1) u_1 +$
$(\ln 0.7 - \ln 0.1) u_2 + (\ln 1.2 - \ln 0.1) u_3)$

subject to

$-x_1 + x_2 \leq 5, \quad x_1 = 0.1 + (0.5 - 0.1) u_1 +$
$(0.7 - 0.1) u_2 + (1.2 - 0.1) u_3,$

$u_1 + u_2 + u_3 \leq 1, \quad 0.1^{0.5} y + (0.5^{0.5} - 0.1^{0.5}) z_1 +$
$(0.7^{0.5} - 0.1^{0.5}) z_2 + (1.2^{0.5} - 0.1^{0.5}) z_3 - x_2 \leq 6,$

$0 \leq z_i, \quad z_i \leq u_i, \quad z_i \leq y, \quad z_i \geq u_i + y - 1,$

$i = 1, 2, 3, \quad -6u_i \leq s_i \leq 6u_i, \quad 6(u_i - 1) +$
$x_2 \leq s_i \leq 6(1 - u_i) + x_2, \quad i = 1, 2, 3,$

s_1, s_2, s_3 are unrestricted in sign variables,

$u_1, u_2, u_3 \in \{0, 1\}, -6 \leq x_2 \leq 4.$

The transformed program can be solved to locate the globally optimal solution $(x_1, x_2, y) = (0.1, -6, 0)$. The objective value is –8.805.

Conclusions

This paper proposes a generalized method to solve the globally optimal solutions of GGP problems with continuous and discrete free variables. The techniques of dealing with free variables aim to change variables and to convert the logical relationship among the variables in a product term into a set of linear inequalities, which can be merged conveniently into the GGP models. Compared with current GGP methods, the proposed method is capable of dealing with free variables of a GGP problem and is guaranteed to converge to a global optimum. In addition, several computationally efficient convexification rules for signomial terms are presented to enhance the efficiency of the optimization approach.

References

1. Adjiman CS, Androulakis IP, Floudas CA (1998) A global optimization method, αBB, for general twice-differentiable NLPs–II. Implementation and computational results. Comput Chem Eng 22:1159–1179
2. Adjiman CS, Dallwig S, Floudas CA, Neumaier A (1998) A global optimization method, αBB, for general twice-differentiable NLPs–I. Theoretical advances. Comput Chem Eng 22:1137–1158
3. Biegler LT, Grossmann IE (2004) Retrospective on optimization. Comput Chem Eng 28:1169–1192
4. Björk KM, Lindberg PO, Westerlund T (2003) Some convexifications in global optimization of problems containing signomial terms. Comput Chem Eng 27:669–679
5. Cardoso MF, Salcedo RL, Feyo de Azevedo S (1996) The simplex-simulated annealing approach to continuous nonlinear optimization. Comput Chem Eng 20:1065–1080
6. Duran M, Grossmann IE (1986) An outer-approximation algorithm for a class of mixed integer nonlinear programs. Math Program 36:307–339
7. Fletcher R, Leyffer S (1994) Solving Mixed Integer Nonlinear Programs by outer approximation. Math Program 66:327–349
8. Floudas CA (1995) Nonlinear and mixed integer optimization: Fundamentals and applications. Oxford Univ Press, New York
9. Floudas CA (2000) Deterministic global optimization: Theory, methods and applications. Kluwer, Boston
10. Floudas CA (2007) On the functional form of convex underestimators for twice continuously differentiable functions. Optim Lett 1:187–192
11. Floudas CA, Pardalos PM (1996) State of the art in global optimization: Computational methods and applications. Kluwer, Boston
12. Geoffrion AM (1972) Generalized Benders Decomposition. J Optim Theory Appl 10:237–260
13. Grossmann IE (2002) Review of nonlinear mixed-integer and disjunctive programming techniques. Optim Eng 3:227–252
14. Grossmann IE, Biegler LT (2004) Part II. Future perspective on optimization. Comput Chem Eng 28:1193–1218
15. Hussain MF Al-Sultan KS (1997) A hybrid genetic algorithm for nonconvex function minimization. J Global Optim 11:313–324
16. Lee S, Grossmann IE (2000) New algorithms for nonlinear generalized disjunctive programming. Comput Chem Eng 24:2125–2142
17. Leyffer S (2001) Integrating SQP and branch-and-bound for mixed integer nonlinear programming. Comput Optim Appl 18:295–309
18. Li HL, Tsai JF (2005) Treating free variables in generalized geometric global optimization programs. J Global Optim 33(1):1–13
19. Lingo Release 9.0 (2004) Lindo System Inc, Chicago
20. Maranas CD, Floudas CA (1997) Global optimization in generalized geometric programming. Comput Chem Eng 21:351–370
21. Pörn R, Harjunkoski I, Westerlund T (1999) Convexification of different classes of non-convex MINLP problems. Comput Chem Eng 23:439–448
22. Quesada I, Grossmann IE (1992) An LP/NLP based branch and bound algorithm for convex MINLP optimization problems. Comput Chem Eng 16:937–947
23. Salcedo RL, Goncalves MJ, Feyo de Azevedo S (1990) An improved random-search algorithm for nonlinear optimization. Comput Chem Eng 14:1111–1126
24. Stubbs RA, Mehrotra S (1999) A branch-and-cut method for 0–1 mixed convex programming. Math Program 86:515–532
25. Sun XL, Mckinnon K, Li D (2001) A convexification method for a class of global optimization problems with applications to reliability optimization. J Global Optim 21:185–199
26. Tsai JF, Li HL, Hu NZ (2002) Global optimization for signomial discrete programming problems in engineering design. Eng Optim 34:613–622
27. Westerlund T (2005) Some transformation techniques in global optimization. In: Liberti L, Maculan N (eds) From theory to implementations, pp 45–74. Kluwer, Boston
28. Westerlund T, Pettersson F (1995) An extended cutting plane method for solving convex MINLP problems. Comput Chem Eng 19:131–136
29. Wu ZY, Bai FS, Zhang LS (2005) Convexification and concavification for a general class of global optimization problems. J Global Optim 31:45–60
30. Yiu KFC, Liu Y, Teo KL (2004) A hybrid descent method for global optimization. J Global Optim 28:229–238

Generalized Monotone Multivalued Maps

GMMVM

Nicolas Hadjisavvas[1], Siegfried Schaible[2]
[1] Department Math., University Aegean,
Karlovassi, Greece
[2] A.G. Anderson Graduate School of Management,
University California, Riverside, USA

MSC2000: 90C26

Article Outline

Keywords

Generalized convexity; Generalized monotonicity; Quasiconvex function; Quasimonotone operator

The study of multivalued generalized monotone operators is a recent(as of 1999) subject. The first to introduce such a notion seem to have been L.G. Mitjuschin and W.M. Polterovich [16] who defined multivalued quasimonotone operators in demand theory. The same concept was also defined by A. Hassouni [10] and D.T. Luc [14]. Later, Luc [15] and J.P. Penot and P.H. Quang [20] proceeded to define new kinds of generalized monotonicity for multivalued operators. Alarge part of this effort has been devoted to the definition of appropriate concepts so that generalized convex nonsmooth functions are characterized by the generalized monotonicity of their subdifferentials [18].

As it stands today, the theory is not at the stage of development of the corresponding theory for single valued operators (see ▶ Generalized monotone single valued maps). More concepts have to be introduced and probably some of the already existing ones have to be modified so that a nice correspondence such as the one exhibited in the first theorem of ▶ Generalized monotone single valued maps can be established, without imposing any additional assumptions. This concerns both generalized monotonicity of multivalued operators and generalized convexity of nonsmooth functions, as some notions of generalized convexity involve subdifferentials.

This article presents various definitions of generalized monotonicity for multivalued operators and generalized convexity for nonsmooth functions. Also, various characterizations of generalized convexity of a function through the corresponding generalized monotonicity of the subdifferential are surveyed. Some characterizations have a 'mixed' form, i. e., they involve both the function and its subdifferential.

The next section contains the definition of the subdifferential for lower semicontinuous functions, along with the necessary notation. Then the less known correspondence between the convexity of a function and the monotonicity of its subdifferential is presented. In the main part of the article, this correspondence is extended to cover the various cases of generalized convexity and generalized monotonicity.

The Subdifferential

There is a host of nonequivalent subdifferentials for nonconvex functions. The interested reader may find a thorough exposition of the various concepts in [19]. The most common, the Clarke–Rockafellar subdifferential, is the one that will be used here, although many of the results hold also for a large number of other subdifferentials; see for instance [1,18,19]. Generalized monotonicity of bifunctions is used in [13] to characterize generalized convex functions through various generalized derivatives.

In this article, X denotes a Banach space, X^* its dual, and $f\colon X \to \mathbf{R} \cup \{+\infty\}$ a lower semicontinuous (lsc) function with nonempty domain $\operatorname{dom}(f) = \{x \in X\colon f(x) \neq +\infty\}$. The function f is called *radially continuous* if its restriction to line segments is continuous. The value of a functional $x^* \in X^*$ at a point $x \in X$ will be denoted by $\langle x^*, x\rangle$. Given $x, y \in X$, (x, y) is the open line segment $\{tx+ (1-t)y\colon t \in (0, 1)\}$. The line segments $[x, y]$, $[x, y)$ and $(x, y]$ aredefined analogously.

The *Clarke–Rockafellar generalized derivative* of f at $x_0 \in \operatorname{dom}(f)$ in the direction $d \in X$ is given by

$$f^{\uparrow}(x_0, d) = \sup_{\varepsilon > 0} \limsup_{\substack{x \to_f x_0 \\ t \searrow 0}} \inf_{d' \in B_\varepsilon(d)} \frac{f(x + td') - f(x)}{t}.$$

Here, $t \searrow 0$ is used to denote the fact that $t > 0$ and $t \to 0$, and $x \to_f x_o$ means that both $x \to x_o$ and $f(x) \to f(x_o)$.

The (Clarke–Rockafellar) *subdifferential* of f at $x_0 \in \operatorname{dom}(f)$ is defined by

$$\partial f(x_0) = \left\{x^* \in X^* \colon \ \langle x^*, d\rangle \le f^{\uparrow}(x_0, d), \ \forall d \in X\right\},$$

while for $x_0 \in X \setminus \operatorname{dom}(f)$, $\partial f(x_0) = \emptyset$.

Even for $x_0 \in \operatorname{dom}(f)$, the subdifferential $\partial f(x_0)$ may be empty. Whenever the function f is locally Lipschitz, one has $\partial f(x_0) \neq \emptyset$, for all $x_0 \in \operatorname{dom}(f)$. In this case $f^{\uparrow}$ coincides with the *Clarke generalized derivative*:

$$f^o(x_0; d) = \limsup_{\substack{x \to x_0 \\ t \searrow 0}} \frac{f(x + td) - f(x)}{t}.$$

In case f is convex, ∂f coincides with the classical *Fenchel–Moreau subdifferential*

$$\partial f(x_0) = \{x^* \in X^* \colon \ \langle x^*, d\rangle \le f(x_0 + d) - f(x_0)\}.$$

The Monotone Case

Let $T \colon X \to 2^{X^*}$ be a multivalued operator with domain

$$D(T) = \{x \in X \colon \ T(x) \neq \emptyset\}.$$

The operator T is called:

- *monotone*, if for all $x, y \in X$ and

  $$x^* \in T(x), y^* \in T(y)$$

  one has

  $$\langle y^* - x^*, y - x\rangle \ge 0; \qquad (1)$$

- *strictly monotone*, if for all $x \neq y$ the above inequality is strict.

It is well known that the subdifferential of a convex function is amonotone operator. However, the fact that convex functions are characterized by the monotonicity of their Clarke–Rockafellar subdifferentials is a relatively recent result. In addition, there exists a 'mixed' characterization of convexity, involving both the function and its subdifferential:

Theorem 1 *Let f be lsc. The following are equivalent:*

i) The function f is convex.

ii) For all $x, y \in \operatorname{dom}(f)$ and all $x^* \in \partial f(x)$ one has:

$$\langle x^*, y - x\rangle \le f(y) - f(x). \qquad (2)$$

iii) The subdifferential ∂f is a monotone operator.

The implication i)⇒ii) follows from the equality of the Clarke–Rockafellar and the Fenchel–Moreau subdifferential for convex functions. The implication ii)⇒iii) is shown in every textbook on monotone operators. Finally, the implication iii)⇒i) is shown in [4].

An analogous theorem holds for strictly convex functions (see ► Generalized monotone single valued maps for definitions of the various kinds of convexity and generalized convexity):

Theorem 2 *Let f be lsc. Consider the following assertions:*

i) The function f is strictly convex.

ii) For all distinct $x, y \in \operatorname{dom}(f)$ and

$$x^* \in \partial f(x),$$

one has

$$\langle x^*, y - x\rangle < f(y) - f(x).$$

iii) The subdifferential ∂f is a strictly monotone operator.

Then i)⇒ii)⇒iii). If, in addition, $\partial f(x) \neq \emptyset$ for all $x \in \operatorname{dom}(f)$, then iii)⇒i).

For the proof, see [8].

The Quasimonotone Case

The concepts of quasimonotone, semistrictly quasimonotone and strictly quasimonotone maps are direct generalizations of the corresponding concepts for single valued maps (see ► Generalized monotone single valued maps and [9,12]). A multivalued operator $T \colon X \to 2^{X*}$ is called:

- *quasimonotone* [14], if for all $x, y \in X$ and all $x^* \in T(x)$, $y^* \in T(y)$, the following implication holds:

  $$\langle x^*, y - x\rangle > 0 \ \Rightarrow\ \langle y^*, y - x\rangle \geq 0;$$

- *semistrictly quasimonotone* [5], if it is quasimonotone and for any distinct

  $$x, y \in D(T)$$

  one has the implication:

  $$\begin{aligned} &\exists x^* \in T(x)\colon\ \langle x^*, y - x\rangle > 0 \\ \Rightarrow &\exists z \in \left(\frac{x+y}{2}, y\right),\ \exists z^* \in T(z)\colon \\ &\langle z^*, y - x\rangle > 0; \end{aligned} \tag{3}$$

- *strictly quasimonotone* [5], if it is quasimonotone and for any distinct $x, y \in D(T)$, there exists $z \in (x, y)$ and $z^* \in T(z)$such that $\langle z^*, y{-}x\rangle \neq 0$.

It can be shown [5] that relation (3) is equivalentto the following: if $\langle x^*, y{-}\ x\rangle > 0$ for some $x^* \in T(x)$, then the set of all $z \in (x, y)$ for which there exists $z^* \in T(z)$ such that $\langle z^*, y{-}\ x\rangle > 0$, is dense in $[x, y]$.

In the single valued case, whenever T is a gradient, its quasimonotonicity, semistrict quasimonotonicity and strict quasimonotonicity is equivalent to quasiconvexity, semistrict quasiconvexity and strict quasiconvexity of the underlying function, respectively (see ► Generalized monotone single valued maps for the corresponding definitions, and [3] for properties of such functions). Analogous results hold for multivalued operators which are subdifferentials. The next theorem gives two equivalent characterizations of quasiconvexity: one 'mixed', andone through the quasimonotonicity of the subdifferential.

Theorem 3 *Let f be lsc. The following are equivalent:*
i) The function f is quasiconvex.
ii) For all x, y ∈ dom(f), the following implication holds:

$$\begin{aligned} &\exists x^* \in \partial f(x)\colon\ \langle x^*, y - x\rangle > 0 \\ &\Rightarrow \forall z \in [x, y]\colon\ f(z) \leq f(y). \end{aligned} \tag{4}$$

iii) The operator ∂f is quasimonotone.

The equivalence i)⇔iii) is shown in [14, Thm. 3.2], while the equivalence i)⇔ii) is shown in [1, Thm. 2.1]. In [1] it is also shown that, in case f is radially continuous, implication (4) is equivalent to the following implication:

$$\exists x^* \in \partial f(x)\colon\ \langle x^*, y - x\rangle > 0 \ \Rightarrow\ f(x) \leq f(y).$$

A 'mixed' characterization exists also for semistrictly quasiconvex functions [5], but a continuity assumption stronger than lower semicontinuity is needed:

Theorem 4 *Let f be lsc. If f is semistrictly quasiconvex, then for all x, y ∈ dom(f) one has:*

$$\begin{aligned} &\exists x^* \in \partial f(x)\colon\ \langle x^*, y - x\rangle > 0 \\ &\Rightarrow \forall z \in [x, y)\colon\ f(z) < f(y). \end{aligned} \tag{5}$$

The converse also holds if in addition f is radially continuous.

Radial continuity is an often used, weak continuity assumption. In fact, it is not as weak as it seems. Since X is a Banach space, it can be shown that a lsc quasiconvex function which is radially continuous is also continuous [8].

Characterization of strict or semistrict quasiconvexity via the generalized monotonicity of the subdifferential requires an even stronger continuity assumption:

Theorem 5 *A locally Lipschitz function f is strictly (respectively semistrictly) quasiconvex, if and only if its subdifferential is strictly (respectively semistrictly) quasimonotone.*

For the proof, see [5].

The Pseudomonotone Case

The definition of pseudomonotonicity for multivalued operators was given by J.C. Yao [21] and generalizes the corresponding definition for single valued operators (see ► Generalized monotone single valued maps and [11]). An operator

$$T\colon\ X \to 2^{X^*}$$

is called *pseudomonotone* if for all $x, y \in X$ one has:

$$\begin{aligned} &\exists x^* \in T(x)\colon\ \langle x^*, y - x\rangle \geq 0 \\ &\Rightarrow\ \forall y^* \in T(y)\colon\ \langle y^*, y - x\rangle \geq 0. \end{aligned}$$

Equivalently, an operator T is pseudomonotone if and only if the following implication holds:

$$\begin{aligned} &\exists x^* \in T(x)\colon\ \langle x^*, y - x\rangle > 0 \\ &\Rightarrow\ \forall y^* \in T(y)\colon\ \langle y^*, y - x\rangle > 0. \end{aligned} \tag{6}$$

Obviously, a pseudomonotone operator T is quasimonotone. If in addition the domain $D(T)$ is convex, then relation (6) implies that T is also semistrictly quasimonotone. Also, it is clear that a monotone operatoris pseudomonotone.

An operator $T : X \to 2^{X^*}$ is called *strictly pseudomonotone* [21], if for all distinct $x, y \in X$ one has:

$$\begin{aligned}&\exists x^* \in T(x)\colon\ \langle x^*, y - x\rangle \geq 0\\ &\ \Rightarrow\ \forall y^* \in T(y)\colon\ \langle y^*, y - x\rangle > 0.\end{aligned}$$

It is clear that a strictly pseudomonotone operator is pseudomonotone, and that a strictly monotone operator is strictly pseudomonotone. Finally, it can easily be shown [8] that a strictly pseudomonotone operator with convex domain is strictly quasimonotone.

In summary, between the various concepts of generalized monotonicity, the following implications hold (some of which assume convexity of the domain):

$$\begin{tikzcd}
 & & qm \\
m \arrow[r, Rightarrow] & pm \arrow[r, Rightarrow] & sstr.qm \arrow[u, Rightarrow] \\
str.m \arrow[u, Rightarrow] \arrow[r, Rightarrow] & str.pm \arrow[u, Rightarrow] \arrow[r, Rightarrow] & str.qm \arrow[u, Rightarrow]
\end{tikzcd}$$

Here, 'str.' and 'sstr.' stands for 'strictly' and 'semistrictly', respectively, and 'm', 'pm' and 'qm' for 'monotone', 'pseudomonotone' and 'quasimonotone', respectively. These implications are exactly the same as those holding for singlevalued operators (see ► Generalized monotone single valued maps).

In contrast to quasiconvex functions and their variants, pseudoconvex functions have to be redefined in the nonsmooth case. The reason is that the usual definition of pseudoconvexity makes explicit reference to the derivative of the function (however, there exists a definition which does not mention the derivative explicitly [17]; see also [3] for details).

A function f is called *pseudoconvex*, if for all $x, y \in \operatorname{dom}(f)$ the following implication holds:

$$\begin{aligned}&\exists x^* \in \partial f(x)\colon\ \langle x^*, y - x\rangle \geq 0\\ &\ \Rightarrow\ \forall z \in [x, y)\colon\ f(z) \leq f(y).\end{aligned} \tag{7}$$

Note that the above definition, expresses a 'mixed' property in the spirit of relation (4); actually, (7) is stronger than (4), and hence any pseudoconvex function is quasiconvex. In particular, a pseudoconvex function f has a convex domain. If in addition f is radially continuous, then it is semistrictly quasiconvex [8].

The definition of pseudoconvexity given here differs slightly from the definition introduced in [20]. There, a function f is called pseudoconvex if it satisfies the implication

$$\exists x^* \in \partial f(x)\colon\ \langle x^*, y - x\rangle \geq 0\ \Rightarrow\ f(x) \leq f(y). \tag{8}$$

A pseudoconvex function (as defined by relation (7)) obviously satisfies (8). The converse is not always true; however, if f is radially continuous, or if its domain is convex, then (8) implies that f is quasiconvex (see [20] and [6], respectively). It follows immediately that f satisfies (7), i. e., it is pseudoconvex.

The following theorem connects pseudoconvexity of a function to pseudomonotonicity of its subdifferential (see [8] and [20] for the proof of the first and the second assertion, respectively):

Theorem 6 *If f is pseudoconvex, then ∂f is pseudomonotone.Conversely, if ∂f is pseudomonotone and f is radially continuous, then f is pseudoconvex.*

A function f is called *strictly pseudoconvex* [8] if for all $x, y \in \operatorname{dom}(f)$ one has:

$$\begin{aligned}&\exists x^* \in \partial f(x)\colon\ \langle x^*, y - x\rangle \geq 0\\ &\ \Rightarrow\ \forall z \in [x, y)\colon\ f(z) < f(y).\end{aligned} \tag{9}$$

For radially continuous functions, relation (9) is equivalent to

$$\exists x^* \in \partial f(x)\colon\ \langle x^*, y - x\rangle \geq 0\ \Rightarrow\ f(x) < f(y). \tag{10}$$

Indeed, if relation (10) holds, then f is pseudoconvex, hence it is semistrictly quasiconvex. Consequently, if $\langle x^*, y - x\rangle \geq 0$ for some

$$x^* \in \partial f(x),$$

then $f(x) < f(y)$ implies that $f(z) < f(y)$ for all $z \in [x, y)$, i. e. (9) holds.

We have the following connection to strict pseudomonotonicity:

Theorem 7 *If f is strictly pseudoconvex, then ∂f is strictly pseudomonotone. Conversely, if ∂f is strictly pseudomonotone and its values are nonempty on $\operatorname{dom}(f)$, then f is strictly pseudoconvex.*

For the proof of the first assertion, see [20]; the second assertion is shown in [8].

As a corollary of the last theorem, it can be shown [8] that a locally Lipschitz, strictly pseudoconvex function f is strictly quasiconvex. Hence, between the various kinds of generalized convexity, the following implications hold (some implications need extra continuity assumptions):

$$\begin{tikzcd}
 & & qcx \\
cx \arrow[r, Rightarrow] & pcx \arrow[r, Rightarrow] & sstr.qcx \arrow[u, Rightarrow] \\
str.cx \arrow[u, Rightarrow] \arrow[r, Rightarrow] & str.pcx \arrow[u, Rightarrow] \arrow[r, Rightarrow] & str.qcx \arrow[u, Rightarrow]
\end{tikzcd}$$

Here, 'cx', 'pcx' and 'qcx' stands for 'convex', 'pseudoconvex' and 'quasiconvex', respectively. Thus, the same implications hold as those fordifferentiable functions (see the corresponding diagram in ► Generalized monotone single valued maps). In addition, each type of generalized convex function is characterized by the corresponding generalized monotonicity of the subdifferential, exactly as in the case of differentiable functions (the first theorem in ► Generalized monotone single valued maps).

See also

► Fejér Monotonicity in Convex Optimization
► Generalized Monotone Single Valued Maps
► Generalized Monotonicity: Applications to Variational Inequalities and Equilibrium Problems
► Set-valued Optimization

References

1. Aussel D (1998) Subdifferential properties of quasiconvex and pseudoconvex functions. J Optim Th Appl 97:29–45
2. Aussel D, Corvellec JN, Lassonde M (1995) Mean value property and subdifferential criteria for lower semicontinuous functions. Trans Amer Math Soc 347:4147–4161
3. Avriel M, Diewert WE, Schaible S, Zang I (1988) Generalized concavity. Plenum, New York
4. Correa R, Jofre A, Thibault L (1992) Characterization of lower semicontinuous convexfunctions. Proc Amer Math Soc 116:67–72
5. Daniilidis A, Hadjisavvas N (1999) Characterization of nonsmooth semistrictly quasiconvex and strictly quasiconvex functions. J Optim Th Appl 102:525–536
6. Daniilidis A, Hadjisavvas N (1999) On the subdifferentials of quasiconvex and pseudoconvex functions and cyclic monotonicity. J Math Anal Appl 237:30–42
7. Ellaia R, Hassouni A (1991) Characterization of nonsmooth functions through their generalized gradients. Optim 22:401–416
8. Hadjisavvas N (2001) The use of subdifferentials for studying generalized convex functions. J Statist Managem Systems no. March
9. Hadjisavvas N, Schaible S (1993) On strong pseudomonotonicity and (semi)strict quasimonotonicity. J Optim Th Appl 79:139–155
10. Hassouni A (1983) Sous-differentiels des fonctions quasiconvexes. Thése de 3ème cycle: Math Appliq Toulouse
11. Karamardian S (1976) Complementarityover cones with monotone and pseudomonotone maps. J Optim Th Appl 18:445–454
12. Karamardian S, Schaible S (1990) Seven kinds of monotone maps. J Optim Th Appl 66:37–46
13. Komlosi S (1995) Generalized monotonicity and generalized convexity. J Optim Th Appl 84:361–376
14. Luc DT (1993) Characterizations of quasiconvex functions. BullAustral Math Soc 48:393–405
15. Luc DT (1994) On generalized convex nonsmooth functions. Bull Austral Math Soc 49:139–149
16. Mitjuschin LG, Polterovich WM (1978) Criteria for monotonicity of demand functions. Ekonomika i Mat Metody 14:122–128 in Russian.
17. Ortega JM, Rheinboldt WC (1970) Iterative solution of nonlinear equations in several variables. Acad. Press, New York
18. Penot JP (1995) Generalized convexity in the light of nonsmooth analysis. In: Duvier R, Michelot C (eds) Recent Developments In Optimization. Lecture Notes Economics and Math Systems. Springer, Berlin, pp 269–290
19. Penot JP (1998) Are generalized derivatives useful for generalized convex functions? In: Crouzeix JP, Martinez-Legaz JE, Volle M (eds) Generalized Convexity, Generalized Monotonicity: Recent Developments. Proc. 5th Internat. Symp. Generalized Convexity, Kluwer, Dordrecht, pp 3–60
20. Penot JP, Quang PH (1997) Generalized convexity of functions and generalized monotonicity of set-valued maps. J Optim Th Appl 92:343–356
21. Yao JC (1994) Multivalued variational inequalities with K-pseudomonotone operators. J Optim Th Appl 83:391–403

Generalized Monotone Single Valued Maps

GMSVM

NICOLAS HADJISAVVAS[1], SIEGFRIED SCHAIBLE[2]

[1] Department Math., University Aegean,
Karlovassi, Greece

[2] A.G. Anderson Graduate School of Management,
University California, Riverside, USA

MSC2000: 90C26

Article Outline

Keywords

Generalized convexity; Generalized monotonicity; Quasiconvex function; Quasimonotone map

In the analysis and solution of complementarity problems and variational inequalities, it is commonly assumed that the defining map is *monotone*. This is not surprising since in the special case of an underlying optimization problem usually *convexity* is assumed, and convexity of the objective function corresponds to monotonicity of its gradient.

For several decades much effort has been devoted to generalizing convexity in various ways, often with the view of nonconvex optimization inmind [1]. On the other hand, only recently a systematic study of generalizations of monotonicity has emerged. Since the article [14] in 1990 about two hundred publications have appeared. They deal with either concepts and characterizations of generalized monotonicity or with uses in variational inequalities and related models [23].

In this survey characterizations of generalized monotonicity for different subclasses of maps are presented. The need for such criteria is obvious, given that the defining inequalities are often hard to verify.

The article is organized as follows. The next section provides a brief review of some basic generalized monotonicity concepts and their relationships. This is followed by a presentation of criteria for generalized-monotonicity in case of differentiable, affine and nondifferentiable (locally Lipschitz) maps in the subsequent sections.

This article on concepts and characterizations of generalized monotone maps in the single valued case is complemented by one on multi valuedmaps. In a third article in this volume the use of generalized monotonicity in variational inequalities and more general models is surveyed. For amore detailed survey of applications see [11].

Seven Kinds of (Generalized) Monotonicity

Seven basic kinds of convex/generalized convex functions are [1]:

- convex (cx), strictly convex (str.cx);
- pseudoconvex (pcx), strictly pseudoconvex (str.pcx);
- quasiconvex (qcx), semistrictly quasiconvex (sstr.qcx) and strictly quasiconvex (str.qcx).

Strongly convex and strongly pseudoconvex functions [1] are not considered here.

These functions are related to each other as follows:

$$\begin{tikzcd}
 & & qcx \\
cx \arrow[r, Rightarrow] & pcx \arrow[r, Rightarrow] & sstr.qcx \arrow[u, Rightarrow] \\
str.cx \arrow[u, Rightarrow] \arrow[r, Rightarrow] & str.pcx \arrow[u, Rightarrow] \arrow[r, Rightarrow] & str.qcx \arrow[u, Rightarrow]
\end{tikzcd}$$

For the sake of completeness, the related definitions are presentedbelow.

Consider $f\colon C \to \mathbf{R}$ where $C \subseteq \mathbf{R}^n$ is convex.

- f is *convex* (*cx*) if for all $x, y \in C$ and $t \in (0, 1)$,

$$f\left(tx + (1-t)y\right) \leq tf(x) + (1-t)f(y); \qquad (1)$$

- f is *strictly convex* (str.cx) if (1) is a strict inequality for $x \neq y$.
- f is *quasiconvex* (qcx) if for all $x, y \in C$ such that $f(x) \leq f(y)$, $t \in (0, 1)$,

$$f\left(tx + (1-t)y\right) \leq f(y); \qquad (2)$$

- f is *strictly quasiconvex* (str.qcx) if (2) is a strict inequality for $x \neq y$;
- f is *semistrictly quasiconvex* (sstr.qcx) if for all $x, y \in C$ such that $f(x) < f(y)$ the inequality (2) is strict.

For the remaining two types of generalized convex functions one assumes differentiability of f on the open convex set $C \subseteq R^n$, although more general definitions are available [1]:

- f is *pseudoconvex* (pcx) if for all $x, y \in C$

$$(y-x)^\top \nabla f(x) \geq 0 \;\Rightarrow\; f(y) \geq f(x); \qquad (3)$$

- f is *strictly pseudoconvex* (str.pcx) if for all $x, y \in C$, $x \neq y$ thesecond inequality in (3) is strict.

Different kinds of generalized convexity preserve different properties of convex functions. E.g., the characteristic of a pseudoconvex function is that a stationary point is a global minimum. Furthermore, for a semistrictly quasiconvex function a local is a global minimum and for a quasiconvex function the lower level sets are convex. The qualifier 'strictly' indicates that a global minimum is unique. In contrast to convex functions, inflection points are admissible for all types of generalized convex functions.

Note that in [1] the terminology of quasiconvex and pseudoconvex functions was harmonized, resulting in renaming former 'strongly quasiconvex' functions as strictly quasiconvex and 'strictly quasiconvex' functions as semistrictly quasiconvex.

It is well known that a differentiable convex function is characterized by a monotone gradient. Correspondingly, a strictly convex function is characterized by a strictly monotone gradient. Accordingly, *generalized monotonicity* concepts have been introduced in such a way that incase of a gradient map $F = \nabla f$ generalized monotonicity of F corresponds to some kind of generalized convexity ofthe underlying function f. The definitions of (generalized) monotone maps are listed below.

Consider $F: C \to \mathbf{R}^n$ where $C \subseteq \mathbf{R}^n$.

- F is *monotone* (m) on C if for all $x, y \in C$

$$(y-x)^\top \big(F(y) - F(x)\big) \geq 0; \qquad (4)$$

- F is *strictly monotone* (str.m) on C if for all $x, y \in C$, $x \neq y$

$$(y-x)^\top \big(F(y) - F(x)\big) > 0; \qquad (5)$$

- F is *pseudomonotone* (pm) on C if for all $x, y \in C$,

$$(y-x)^\top F(x) \geq 0 \;\Rightarrow\; (y-x)^\top F(y) \geq 0, \qquad (6)$$

which is equivalent to

$$(y-x)^\top F(x) > 0 \;\Rightarrow\; (y-x)^\top F(y) > 0;$$

- F is *strictly pseudomonotone* (str.pm) on C if for all $x, y \in C$, $x \neq y$,

$$(y-x)^\top F(x) \geq 0 \;\Rightarrow\; (y-x)^\top F(y) > 0; \qquad (7)$$

- F is *quasimonotone* (qm) if for all $x, y \in C$,

$$(y-x)^\top F(x) > 0 \;\Rightarrow\; (y-x)^\top F(y) \geq 0; \qquad (8)$$

- F is *strictly quasimonotone* (str.qm) on C if F is quasimonotone on C and for all $x, y \in C$, $x \neq y$ there exists $z = tx + (1-t)y$, $t \in (0, 1)$, such that

$$(y-x)^\top F(z) \neq 0; \qquad (9)$$

- F is *semistrictly quasimonotone* (sstr.qm) on C if F is quasimonotone on C and for $x, y \in C$, $x \neq y$,

$$(y-x)^\top F(x) > 0 \;\Rightarrow\; (y-x)^\top F(z) > 0 \qquad (10)$$

for some $z = tx + (1-t)y$, $t \in (0, 1/2)$.

If F is continuous, quasimonotonicity does not have to be required explicitly for strictly/semistrictly quasimonotone maps since it is implied by (9), (10), respectively. In terms of references for the concepts above, see [13] for pseudomonotone maps, [14] for quasimonotone and strictly pseudomonotone maps and [9] for strictly quasimonotone and semistrictly quasimonotone maps.

The following diagram was derived in [9,13,14] for general maps which are not necessarily gradient maps:

$$\begin{tikzcd}
 & & qm \\
m \arrow[r, Rightarrow] & pm \arrow[r, Rightarrow] & sstr.qm \arrow[u, Rightarrow] \\
str.m \arrow[r, Rightarrow] \arrow[u, Rightarrow] & str.pm \arrow[r, Rightarrow] \arrow[u, Rightarrow] & str.qm \arrow[u, Rightarrow]
\end{tikzcd}$$

Now consider the special case of a gradient map $F = \nabla f$, where f is differentiable on the open convex set $C \subseteq \mathbf{R}^n$. In analogy to monotone maps it can be shown [9,13,14]:

Theorem 1 *The map $F = \nabla f$ is quasimonotone (respectively, semistrictly quasimonotone, strictly quasimonotone, pseudomonotone, strictly pseudomonotone) if and only if the function f is quasiconvex (respectively, semistrictly quasiconvex, strictly quasiconvex, pseudoconvex, strictly pseudoconvex).*

Note that in the case of semistrictly quasiconvex functions Theorem 1 provides the first successful characterization in terms of the gradient. Before, the existence of such a characterization was doubted [17].

There are several studies where similar results are obtained for nondifferentiable functions in which the gradient is replaced by the subdifferential (see, e. g., ▶ Generalized monotone multivalued maps).

Given the geometric properties of generalized convex functions mentioned above [1], it is not difficult to derive the geometric properties describing generalized monotonicity of gradient maps; e. g. [2,3,15].

New generalized monotone maps can be constructed from existing ones. As an example from [20], consider $z = \mathrm{A}x + b$, where A is an $m \times n$ matrix and $b \in \mathbf{R}^m$. Let $D \subseteq \mathbf{R}^m$ and $C = \{x \in \mathbf{R}^n{:}Ax + b \in D\}$. Then the map $F(x) = A^\intercal G\ (Ax + b)$ is quasimonotone (pseudomonotone) on C if G is quasimonotone (pseudomonotone) on D. Moreover, F is strictly pseudomonotone on C if G is strictly pseudomonotone on D and A hasfull rank.

The Differentiable Case

In this section it is assumed that $F{:}C \to \mathbf{R}^n$ is differentiable and $C \subseteq \mathbf{R}^n$ is an open convex set. Let $J_F\,(x)$ be the Jacobian of F. First order characterizations of generalized monotone maps have been established in [15]. In case of gradient maps they extend classical second order characterizations of generalized convex functions.

Let $x \in C$, $v \in \mathbf{R}^n$, $v \neq 0$ and consider the following conditions:

A) $v^\intercal F(x) = 0$ implies $v^\intercal J_F(x)v \geq 0$;

A+) $v^\intercal F(x) = 0$ implies $v^\intercal J_F(x)v > 0$;

B) $v^\intercal F(x) = v^\intercal J_F(x)v = 0$ and the condition $v^\top F(x + \widehat{t}v) > 0$ for some $\widehat{t} < 0$ implies that there exists $\widetilde{t} > 0$ such that $x + \widetilde{t}v \in C$, $v^\top F(x + tv) \geq 0$ for all $0 \leq t \leq \widetilde{t}$;

C) $v^\intercal F(x) = v^\intercal J_F(x)\ v = 0$ implies that there exists $\widetilde{t} > 0$ such that $x + \widetilde{t}v \in Cv^\top F(x + tv) \geq 0$ for all $0 \leq t \leq \widetilde{t}$.

The following can be shown:

Theorem 2 *Let F: $C \to \boldsymbol{R}^n$ be differentiable on the open convex set $C \subseteq \boldsymbol{R}^n$.*

i) F is quasimonotone if and only if A) and B) hold for all $x \in C$ and $v \in \boldsymbol{R}^n$;

ii) F is pseudomonotone if and only if A) and C) hold for all $x \in C$ and $v \in \boldsymbol{R}^n$;

iii) F is strictly pseudomonotone if A+) holds for all $x \in C$ and $v \in \boldsymbol{R}^n$.

More recently, it was shown in [4] that for continuously differentiable maps $v^\intercal F(x) = 0$ in B) and C) can be replaced by the less restrictive assumption $F(x) = 0$, and i) and ii) are still true. An immediate consequence of this stronger characterization is that for a nonvanishing map on an open convex set there is no difference between quasimonotonicity and pseudomonotonicity. Both are characterized by condition A). However, this is no longer true in closed convex sets (see [10, Example 3.1]).

The Affine Case

In this section we focus on the special case of affine maps. Let $F(x) = Mx + q$ where M is an $n \times n$ matrix and $q \in \mathbf{R}^n$. Consider F on an open convex set $C \subseteq \mathbf{R}^n$. For general differentiable maps we have $F = \nabla f$ if and only if $J_F(x)$ is symmetric for all x. Hence for an affine map $F(x) = Mx + q$ we have $F = \nabla f$ if and only if M is symmetric. In this case $f(x) = (x^\intercal Mx)/2 + q^\intercal x$. Therefore first order characterizations of generalized monotone affine maps correspond to second order characterizations of generalized convex quadratic functions.

For affine maps conditions B) and C) are always satisfied. Hence, specializing Theorem 2 we have

Theorem 3 *The map $F(x) = Mx + q$ is quasimonotone on an open convex set $C \subseteq \boldsymbol{R}^n$ if and only if F is pseudomonotone on C if and only if for all $x \in C$ and $v \in \boldsymbol{R}^n$*

$$v^\top(Mx + q) = 0 \ \Rightarrow\ v^\top Mv \geq 0.$$

As a result, quasimonotonicity in a neighborhood of a point $\overline{x}$ such that $M\overline{x} + q = 0$ implies monotonicity on $\mathbf{R}^n$.

As mentioned earlier, one can construct new generalized monotone maps with the help of a given one as follows. Given the linear map $G(z) = Mz$, if G is quasimonotone (pseudomonotone) on the nonnegative orthant $\mathbf{R}^m_+$, then the map $F(x) = (A^\intercal MA)x$ is quasimonotone (pseudomonotone) on $\mathbf{R}^n_+$, for any nonnegative $m \times n$ matrix A.

Recently a matrix-theoretic characterization of generalized monotone affine maps was obtained [6]. The departure point for its derivation is Theorem 3. The following notation is needed to describe the results.

For the affine map $F(x) = Mx + q$ one considers

$$B = \frac{1}{2}(M + M^\top), \quad P = \frac{1}{2}M^\top B^\dagger M,$$

where $B^\dagger$ is the Moore–Penrose pseudo-inverse of B, n_+, n_- and n_0 is the number of positive, negative and zero eigenvalues of B, respectively,

$$\begin{aligned} r &= \dim(\ker(M)) , \\ f(x) &=(Mx+q)^{\top} B^{\dagger}(Mx+q), \\ S &= \{x \in \mathbf{R}^n :\ f(x) \leq 0\} , \\ T &= \left\{x \in \mathbf{R}^n :\ x^{\top} P x \leq 0\right\} , \end{aligned}$$

$C \subseteq \mathbf{R}^n$ is convex with $C \neq \emptyset$.

One has [6]:

Theorem 4 *F is quasimonotone on C (and pseudomonotone on (C)) if and only if one of the following conditions holds:*

i) *$n_- = 0$, i. e., B is positive semidefinite and F is monotone on $\boldsymbol{R}^n$;*

ii1) *$n_- = 1$, $r = n_0 + 1$, $-q \notin M$ (int C), $q \in B$ $(\boldsymbol{R}^n) \supseteq M$ $(\boldsymbol{R}^n)$, P is positive semidefinite, S isa closed convex set and $C \subseteq S$;*

ii2) *$n_- = 1$, $r = n_0$, $-q \notin M$(int C), $q \in B(\boldsymbol{R}^n) = M(\boldsymbol{R}^n)$, $T = T_+ \cup (-T_+)$ where T_+ is a closed convex cone, int $T_+ \neq \emptyset$, and for $\overline{x}$ such that $M\overline{x} = q$ either $C \subseteq -\overline{x} + T_+$ or $C \subseteq -\overline{x} - T_+$.*

Hence the maximal domain of quasimonotonicity is:

- $\mathbf{R}^n$ in case i);
- S in case ii1), and
- $-\overline{x} + T_+$ or $-\overline{x} - T_+$ in case ii2).

From Theorem 4 a characterization of quasimonotone (pseudomonotone)affine maps on convex cones can be derived, and further specialized to the nonnegative orthant [6].

It should be noted that in the special case $M^{\top} = M$, case ii1) does not occur and Theorem 4 reduces to classical characterizations of generalized convex quadratic functions [7,18,19,21,22]; see also [1]. Case ii1) does not occur either if M is nonsingular. Hence it arises only if M is not symmetric and singular.

Theorem 4 characterizes pseudomonotone affine maps on open convex sets. However in applications, e. g. in complementarity problems and variational inequalities, pseudomonotonicity on closed and convex sets is needed. Such characterizations have very recently been derived in [5] with an approach different from the one in [6]. It involves an extension of Martos' concept of positive subdefinite matrices [18] to the nonsymmetric case. Among others, [5] generalizes previous results on pseudomonotone matrices for linear complementarity problems, e. g. [8].

The Nondifferentiable Case

Finally, characterizations of certain nondifferentiable generalizedmonotone maps [16] are presented in this section.

Let F: $C \to \mathbf{R}^n$ be locally Lipschitz where $C \subseteq \mathbf{R}^n$ is open convex. The criteria below make use of the generalized Jacobian in the senseof Clarke. Given $x \in C$, let $L(x)$ be the set of all limits $DF(x_i)$ where $x_i \to x$, F is differentiable at $x_i \in C$ and $DF\ (x_i)$ is the Jacobian. Define $\partial F(x)$ to be the convex hull of $L(x)$. Finally, for $x \in C$ and $v \in \mathbf{R}^n$ set

$$\begin{aligned} D_+F(x;v) &= \sup\left\{v^{\top}Av :\ A \in \partial F(x)\right\} , \\ D_-F(x;v) &= \inf\left\{v^{\top}Av :\ A \in \partial F(x)\right\} . \end{aligned}$$

In generalization of Theorem 2i) one has:

Theorem 5 *The locally Lipschitz map F is quasimonotone on C if and only if for all $x \in C$, $v \in \boldsymbol{R}^n$*

A') *$v^{\top}F(x) = 0$ implies $D_+F(x;v) \geq 0$, and*

B') *$v^{\top}F(x) = 0$, $0 \in \{v^{\top}Av$: $A \in \partial F(x)\}$ and $v^{\top}F(x + \hat{t}v) > 0$ for some $\hat{t} < 0$ imply that there exists $\tilde{t} > 0$ such that $v^{\top}$ $F(x + tv) \geq 0$ for all $t \in [0, \tilde{t}]$.*

In light of [4], a stronger sufficient condition can be obtained which however is no longer necessary [16], in contrast to the differentiable case.

Theorem 6 *The map F is quasimonotone on C if for all $x \in C$, $v \in \boldsymbol{R}^n$, $\boldsymbol{v} \neq 0$*

A'') *$v^{\top}F(x) = 0$ implies $D_-F(x;v) \geq 0$, and*

B'') *$F(x) = 0$, $D_-(x;v) = 0$ and $v^{\top}F(x + \hat{t}v) > 0$ for some $\hat{t} < 0$ imply that there exists $\tilde{t} > 0$ such that $v^{\top}$ $F(x + tv) \geq 0$ for all $t \in [0, \tilde{t}]$.*

In analogy to the differentiable case (see Theorem 2), corresponding characterizations can be obtained for pseudomonotone maps, replacing B′), B″) by a stronger condition. Furthermore, criteria for strict pseudomonotonicity are derived in [16].

Very recently, generalized monotonicity criteria for locally Lipschitz maps have been extended to the class of general continuous maps [12]. In this study Clarke's generalized Jacobian is replaced by an 'approximate Jacobian'.

Conclusion

In this survey we have presented various characterizations of generalized monotone maps. Details are shown mainly for quasimonotone and pseudomonotone maps. In retrospect, it becomes clear how the main characterization in the differentiable case (Theorem 2) specializes in the affine case (Theorems 3, 4) and how it can be extended in the nondifferentiable case (Theorem 5).

See also

- ► Fejér Monotonicity Inconvex Optimization
- ► Generalized Monotone Multivalued Maps
- ► Generalized Monotonicity: Applications to Variational Inequalities and Equilibrium Problems
- ► Set-valued Optimization

References

1. Avriel M, Diewert WE, Schaible S, Zang I (1988) Generalized concavity. Plenum, New York
2. Castagnoli E, Mazzoleni P (1991) Order-preserving functionsand generalized convexity. Rivista di Mat per le Sci Economiche e Sociali 14:33–46
3. Crouzeix JP (1998) Characterizations of generalized convexity and generalized monotonicity, a survey. In: Crouzeix JP, Martinez-Legaz JE, Volle M (eds) Generalized Convexity, Generalized Monotonicity: Recent Developments. Kluwer, Dordrecht, 237–256
4. Crouzeix JP, Ferland JA (1996) Criteria for differentiable generalized monotone maps. Math Program 75:399–406
5. Crouzeix JP, Hassouni A, Lahlou A, Schaible S (1999) Positive subdefinite matrices, generalized monotonicity and linear complementarity problems. SIAM J Matrix Anal Appl
6. Crouzeix JP, Schaible S (1996) Generalized monotone affine maps. SIAM J Matrix Anal Appl 17:992–997
7. Ferland JA (1971) Quasi-convex and pseudo-convex functions on solid convex sets. Dept Oper Res Stanford Univ 71–4
8. Gowda MS (1990) Affine pseudomonotone maps and the linear complementarity problem. SIAM J Matrix Anal Appl 11:373–380
9. Hadjisavvas N, Schaible S (1993) On strong pseudomonotonicity and (semi)strict quasimonotonicity. J Optim Th Appl 79:139–155
10. Hadjisavvas N, Schaible S (1996) Quasimonotone variational inequalities in Banach spaces. J Optim Th Appl 90:95–111
11. Hadjisavvas N, Schaible S (1998) Quasimonotonicity and pseudomonotonicity in variational inequalities and equilibrium problems. In: Crouzeix JP, Martinez-Legaz JE, Volle M (eds) Generalized Convexity, Generalized Monotonicity: Recent Developments. Kluwer, Dordrecht, pp 257–275
12. Jeyakumar V, Luc DT, Schaible S (1998) Characterizations of generalized monotone nonsmooth continuous maps using approximate Jacobians. J Convex Anal 5:119–132
13. Karamardian S (1976) Complementarity over cones with monotone and pseudomonotone maps. J Optim Th Appl 18:445–454
14. Karamardian S, Schaible S (1990) Seven kinds of monotone maps. J Optim Th Appl 66:37–46
15. Karamardian S, Schaible S, Crouzeix JP (1993) Characterizations of generalized monotone maps. J Optim Th Appl 76:399–413
16. Luc DT, Schaible S (1996) Generalized monotone nonsmooth maps. J Convex Anal 3:195–205
17. Mangasarian OL (1969) Nonlinear programming. McGraw-Hill, New York
18. Martos B (1969) Subdefinite matrices and quadratic forms. SIAM J Appl Math 17:1215–1223
19. Martos B (1971) Quadratic programming with a quasiconvex objective function. Oper Res 19:82–97
20. Pini R, Schaible S (1994) Invariance properties of generalized monotonicity. Optim 28:211–222
21. Schaible S (1971) Beiträge zur quasikonvexen Programmierung. PhD Thesis Univ. Köln
22. Schaible S (1981) Quasiconvex, pseudoconvex and strictly pseudoconvex quadratic functions. J Optim Th Appl 35:303–338
23. Schaible S (1996) From generalized convexity to generalized monotonicity. In: Du D-Z, Zhang X-S, Cheng K (eds) Operations Research and its Applications, Proc. Second Internat. Symp. Oper. Res.and its Applications (ISORA) in Guilin, P.R. China, Beijing Word Publ. Corp., pp 134–143

Generalized Monotonicity: Applications to Variational Inequalities and Equilibrium Problems

GMVIPEP

NICOLAS HADJISAVVAS[1], SIEGFRIED SCHAIBLE[2]

[1] Department Math., University Aegean, Karlovassi, Greece

[2] A.G. Anderson Graduate School of Management, University California, Riverside, USA

MSC2000: 90C26, 49J40, 46N10

Article Outline

Keywords

Generalized monotonicity; Variational inequalities; Equilibrium problems

This article on generalized monotone maps focuses on some of their uses in variational inequalities and equilibrium problems. Definitions and properties of various types of generalized monotone maps are found in ▶ Generalized monotone single valued maps and ▶ Generalized monotone multivalued maps. These articles form the background of the present survey.

Variational inequalities appear in various forms and arise in a wide range of problems in the natural and social sciences, for example [22]. The simplest variational inequality problem (VIP) is the following: Given a nonempty closed convex subset K of $\mathbf{R}^n$ and a map F: $K \to \mathbf{R}^n$, find an element $x_0 \in K$ such that

$$(F(x_0))^\top (x - x_0) \geq 0 \text{ for all } x \in K. \tag{1}$$

The prime example of a variational inequality stems from a minimization problem. Given a differentiable function f: $K \to \mathbf{R}$, if $x_0 \in K$ minimizes f, then x_0 is a solution of the VIP (1) with $F = \nabla f$.

As shown by G.J. Hartman and G. Stampacchia [17], (1) has a solution if K is compact and F is continuous. This result found many applications and holds also, with the same assumptions, in infinite-dimensional Banach spaces (cf. [26, Prop. 77.8]). However, in infinite-dimensional problems this form of the theorem is not useful. The reason for this is that in almost all interesting applications the assumptions of (strong) compactness of the set K and of continuity of the operator F are too strong to be met. A decisive step forward was made by F. Browder who relaxed both assumptions, at the cost of imposing another assumption, namely monotonicity [7]. Specifically, let X be a real Banach space with dual X^*, and K a nonempty, weakly compact and convex subset of X. Given an operator T: $K \to X^*$, consider the following VIP: find $x_0 \in K$ such that

$$\langle Tx_0, x - x_0 \rangle \geq 0 \quad \text{for all } x \in K, \tag{2}$$

where $\langle \cdot, \cdot \rangle$ is the duality pairing between X^* and X. As shown by Browder, the VIP (2) has a solution if T is hemicontinuous and monotone. We recall that an operator T is called *hemicontinuous* if its restriction to line segments is continuous when X^* is equipped with the w^*-topology. The operator is called *monotone* if for all $x, y \in K$ one has

$$\langle Ty - Tx, y - x \rangle \geq 0.$$

It is interesting to note that in the standard example of a variational inequality problem where $X = \mathbf{R}^n$ and T is the gradient of a function f: $K \to \mathbf{R}$ the operator T is monotone if and only if f is convex. This shows that monotonicity is a natural assumption for VIP. But it also shows that it may be too rigid in many applications. This led to the consideration of variational inequality problems and their extensions with *generalized monotone operators*. The first to consider generalized monotonicity in connection with variational inequalities was H. Brezis [5]. Then S. Karamardian [19], coming from convex and generalized convex optimization [1], began a tradition of introducing concepts of generalized monotonicity which, unlike the one of Brezis, preserve the connection between monotonicity and convexity. They ensure that in case of a gradient map, the gradient is generalized monotone (for instance, pseudomonotone, strictly pseudomonotone, quasimonotone, strictly quasimonotone, semistrictly quasimonotone) if the underlying function is generalized convex (i. e., respectively, pseudoconvex, strictly pseudoconvex, quasiconvex, strictly quasiconvex, semistrictly quasiconvex [1]). For definitions and properties of these concepts see ▶ Generalized monotone single valued maps and ▶ Generalized monotone multivalued maps for single- and multivalued generalized monotone maps, respectively.

In the next section, results on the existence of solutions for the variational inequality problem with generalized monotone operators are presented. A generalization of these results to vector valued variational inequality problems is given in the third section. Finally, the last section surveys results on the existence of solutions for equilibrium problems, both in the scalar and in the vector case. To begin, consider the following notation and definitions.

Let X be a real Banach space. Given $x, y \in X$, $]x, y[$ and $[x, y]$ denote the open line segment and the closed line segment joining x and y, respectively; the segments $]x, y]$, and $[x, y[$ are defined analogously. A multivalued operator $T\colon K \to 2^{X^*}\backslash\{\emptyset\}$ is called *upper hemicontinuous* if for all $x, y \in K$, the restriction of T to $[x, y]$ is upper semicontinuous with respect to the w^*-topology on X^*.

For any nonempty subset D of X, a point $x_0 \in X$ is called an *inner point* of D [14,25] if for all $u \in X^*$ the following implication holds:

$$\langle x, u\rangle \leq \langle x_0, u\rangle\,, \ \forall x \in D$$
$$\Rightarrow \quad \langle x, u\rangle = \langle x_0, u\rangle\,, \quad \forall x \in D.$$

The set of inner points of D is denoted by inn D. The concept of an inner point is a generalization of the concept of a relative algebraic interior point. Indeed, in case X is finite dimensional, the two concepts coincide. In the general case, any relative algebraic interior point is an inner point; in case of a closed convex set, inner points have the following properties [14,25]:

Theorem 1 *Let $K \neq \emptyset$ be a closed and convex subset of X. Then one has:*

i) inn $K \subseteq K$;

ii) if K is separable, then inn $K \neq \emptyset$;

iii) if $x_1 \in K$, $x_0 \in$ inn K, then

$$]x_1, x_0] \subseteq \text{inn}\, K;$$

in particular, inn K is convex.

There are many important examples of closed convex subsets K which contain inner points, without containing any relative algebraic interior points [14].

Scalar Variational Inequalities

Let X be a real Banach space, and K a nonempty, closed, convex subset of X. Let further $T\colon K \to 2^{X^*}\backslash\ \{\emptyset\}$ be a multivalued operator with nonempty values. The VIP for such an operator is the following: find $x_0 \in K$ such that

$$\forall x \in K\, \exists x^* \in Tx_0: \quad \langle x^*, x - x_0\rangle \geq 0. \qquad (3)$$

This problem is closely related to the so-called *dual variational inequality problem* (DVIP), which is the following: find $x_0 \in K$ such that

$$\forall x \in K\, \forall x^* \in Tx: \langle x^*, x - x_0\rangle \geq 0. \qquad (4)$$

Indeed, it is well known that, if x_0 is a solution of DVIP, then it is also a solution of VIP, provided that T is upper hemicontinuous [20]. For this reason, most proofs of existence of a solution for VIP establish first the existence of a solution of DVIP.

R.W. Cottle and J.C. Yao [8] were the first to show an existence result for a solution of a VIP with a single valued pseudomonotone operator, hereby extending Karamardian's result [19] for complementarity problems in finite-dimensional spaces. Later, Yao [24] generalized this result to multi valued pseudomonotone operators; I.V. Konnov [20] generalized it further to include semistrictly quasimonotone operators; see ► Generalized monotone multivalued maps. The most general result in this direction with no assumptions (except coercivity) was derived for properly quasimonotone operators [12]. The operator T is called *properly quasimonotone* if for all $x_1, \dots, x_n \in K$ and all $y \in \text{co}\{x_1, \dots, x_n\}$ there exists $i \in \{1, \dots, \text{n}\}$ such that $\langle x^*, y - x_i\rangle \leq 0$ for all $x^* \in Tx_i$. The name of this property is justified by the fact that a lower semicontinuous function $f\colon K \to \mathbf{R}$ is quasiconvex if and only if its Clarke-Rockafellar subdifferential is properly quasimonotone [12]. For such operators, the following theorem holds [11]:

Theorem 2 *Let $T\colon K \to 2^{X^*}\backslash\{\emptyset\}$ be a properly quasimonotone operator. Suppose that K is weakly compact, or alternatively that the following coercivity condition holds: there exists a weakly compact subset W of K and $x_0 \in W$ such that*

$$\forall x \in K\backslash W\ \exists x_0^* \in Tx_0\colon \left\langle x_0^*, x_0 - x\right\rangle < 0. \qquad (5)$$

Then the DVIP (4) has a solution. Consequently, if T is upper hemicontinuous, then the VIP (3) also has a solution.

A semistrictly quasimonotone operator (or, a fortiori, a pseudomonotone operator) is properly quasimonotone [11]. Thus, the above result generalizes the corresponding results in [20] and [24].

For the still more general case of a quasimonotone operator, even for single valued operators, one needs a mild assumption on the domain [14]. For multivalued operators one needs still stronger assumptions [9]:

Theorem 3 *Let $T\colon K \to 2^{X^*}\backslash\ \{\emptyset\}$ be a quasimonotone operator. Suppose that:*

a) K is weakly compact, or alternatively that coercivity condition (5) holds;
b) inn K≠ ∅;
c) T has compact values.
d) T is upper hemicontinuous.
Then the VIP (3) has a solution.

Vector Variational Inequalities

The VIP has been generalized in various ways. One of these generalizations proposed by F. Giannessi [13] suggests to consider the variational inequality in a multidimensional space rather than the real number field. This is the so-called *vector variational inequality problem* (VVIP). The VVIP is closely related, just as its scalar counterpart, to the least element problem and the complementarity problem [23].

In the VVIP, apart from the Banach space X and its closed, convex subset K, one considers a Banach space Y and the space $L(X, Y)$ of all continuous linear operators from X to Y. The space Y is ordered by a cone C. In this case, the expression 'the element $x \in Y$ is nonnegative' can have two different meanings: either $x \in C$ or $x \notin -\text{int}\, C$. It further increases the applicability, especially to economics, without much additional effort if one allows this cone to 'move'; thus, instead of a cone one considers a multivalued mapping $C\colon K \to 2^Y$ such that for each $x \in K$, $C(x)$ is a closed convex cone with nonempty interior. Let further $T\colon K \to 2^{L(X,Y)} \setminus \{\emptyset\}$ be a multivalued operator. The VVIP is the following: find $x_0 \in K$ such that

$$\begin{aligned}&\forall y \in K\, \exists A \in Tx_0 :\\ &A(y - x_0) \notin -\operatorname{int} C(x_0).\end{aligned} \tag{6}$$

In the scalar case $Y = \mathbf{R}$, $C(x) = \mathbf{R}^+$ one has $L(X, Y) = X^*$, and VVIP becomes VIP. In the general case, monotonicity and generalized monotonicity have to be newly defined. The operator T is called:

- *monotone* if for all $x, y \in K$ one has:

$$\begin{aligned}&\forall A \in Tx\, \forall B \in Ty:\\ &(B - A)(y - x) \in C(x);\end{aligned}$$

- *pseudomonotone* if for all $x, y \in K$ the following implication holds:

$$\begin{aligned}&\exists A \in Tx :\ A(y - x) \notin -\operatorname{int} C(x)\\ &\Rightarrow\quad \forall B \in Ty\colon\ B(y - x) \notin -\operatorname{int} C(x);\end{aligned}$$

- *quasimonotone* if for all $x, y \in K$ the following implication holds:

$$\begin{aligned}&\exists A \in Tx :\ A(y - x) \notin -C(x)\\ &\quad\Rightarrow \forall B \in Ty\colon\ B(y - x) \notin -\operatorname{int} C(x).\end{aligned}$$

We now recall some topological notions. The *strong operator topology* (SOT) on $L(X, Y)$ is the weakest topology such that for each $x \in X$, the function $L(X, Y) \ni A \to Ax \in Y$ is continuous. An operator $A \in L(X, Y)$ is called *completely continuous* if it maps weakly convergent sequences into strongly convergent sequences. Examples of completely continuous operators are compact operators. The following result proved in [10] generalizes many existence results in the literature as well as Theorem 3:

Theorem 4 *Suppose that the following assumptions hold:*
i) the operator T is upper hemicontinuous with respect to the SOT topology;
ii) the graph of the multifunction

$$x \to Y \setminus (-\operatorname{int} C(x))$$

is sequentially closed in X × Y in the (weak)× (strong) topology;
iii) K is weakly compact;
iv) for each x ∈ K, Tx consists of completely continuous operators;
v) T is pseudomonotone, or
v') T is quasimonotone, its values are norm compact and inn K ≠ ∅.
Then the VVIP (6) has a solution.

As in the scalar case, the assumption 'K is compact' may be replaced by a coercivity condition.

Equilibrium Problems

The remainder of this article deals with problems more general than VIP. Given a nonempty set K and a bifunction $f\colon K \times K \to \mathbf{R}$, the equilibrium problem (EP) [4,6] for f is the following: find $x_0 \in K$ such that

$$f(x_0, y) \geq 0 \quad \text{for all } y \in K. \tag{7}$$

A great variety of problems can be formulated as an EP including problems of optimization, saddle point theory, game theory, fixed point theory and VIP [4]. For instance, if K is a nonempty closed, convex subset of a Banach space X and $T\colon K \to 2^{X^*} \setminus \{\emptyset\}$ is a multivalued operator with weakly compact values, let f be defined as

$$f(x, y) = \max\{\langle x^*, y - x\rangle : \ x^* \in Tx\}\,. \qquad (8)$$

It is easy to see that $x_0 \in K$ is a solution of the EP (7) if and only if it is a solution of the VIP (2). Because of this correspondence, one is led to define concepts of generalized monotonicity for bifunctions. A bifunction f is called:

- *monotone* [4] if for all $x, y \in K$ one has:

$$f(x, y) + f(y, x) \leq 0;$$

- *pseudomonotone* [3] if for all $x, y \in K$ the following implication holds:

$$f(x, y) \geq 0 \ \Rightarrow \ f(y, x) \leq 0;$$

- *quasimonotone* [3] if for all $x, y \in K$ the following implication holds:

$$f(x, y) > 0 \ \Rightarrow \ f(y, x) \leq 0.$$

It is easy to see that a multi valued operator is monotone (respectively, pseudomonotone, quasimonotone) if and only if the bifunction defined by relation (8) is monotone (respectively, pseudomonotone, quasimonotone). Equilibrium problems with generalized monotone bifunctions in the above sense were considered in [3]. There the following result was proved:

Theorem 5 *Let X be a real topological Hausdorff vector space and $K \subseteq X$ be nonempty, convex and closed. Let further $f\colon K \times K \to \mathbf{R}$ be a bifunction such that $f(x, x) \geq 0$ for all $x \in K$. Consider the following assumptions:*

i) $f(\cdot, y)$ is hemicontinuous (i. e., continuous on every line segment in K) for all $y \in K$;

ii) $f(x, \cdot)$ is semistrictly quasiconvex [1] and lower semicontinuous for all $x \in K$;

iii) there exists a compact subset $B \subseteq X$ and $y_0 \in B \cap K$ such that $f(x, y_0) < 0$ for all $x \in K \setminus B$ (coercivity);

iv) for all $x \in K$, if $f(x, y) = 0$ and $f(x, y_1) > 0$, then $f(x, z) > 0$ for all $z \in]y, y_1[$;

v) the algebraic interior of K is nonempty.

If f is pseudomonotone and assumptions (i–iii) hold, then the EP (7) has a solution. Likewise, if f is quasimonotone and all assumptions i)–v) hold, then (7) has a solution.

The above theorem generalizes older results by Brezis, L. Nirenberg and Stampacchia [6] and is related to more recent results with monotone bifunctions by E. Blum and W. Oettli [4].

Just like vector variational inequalities, vector equilibrium problems have also been considered where the bifunction takes values in a locally convex vector space ordered by a cone [2]. As shown by Oettli [21] for the pseudomonotone case, vector equilibrium problems can also be treated by considering two real valued bifunctions, rather than one vector valued one. Oettli's approach can even be applied to the quasimonotone case [15]. For this, let X be a real Hausdorff topological vector space, $K \subseteq X$ be nonempty and convex, and $f, g\colon K \times K \to \mathbf{R}$ be two bifunctions. The bifunction f is said to be *pseudomonotone* with respect to the bifunction g [21] if for all $(x, y) \in K \times K$ the following implication holds:

$$f(x, y) \geq 0 \ \Rightarrow \ g(y, x) \leq 0.$$

The bifunctions f, g are said to be a *quasimonotone pair* [15] if for all $(x, y) \in K \times K$ the following implication holds:

$$f(x, y) > 0 \ \Rightarrow \ g(y, x) \leq 0.$$

If $f = g$, then the above definitions reduce to those of pseudomonotone and quasimonotone bifunctions, respectively.

The following rather technical, but very useful result was proved in [21] for the pseudomonotone case and in [15] for the quasimonotone case:

Theorem 6 *Consider the following assumptions:*

i) $f(x, x) \geq 0$ for all $x \in K$;

ii) the set $\sigma(y) = \{x \in K : g(y, x) \leq 0\}$ is closed in K for all $y \in K$;

iii) for all $x, y, z \in K$, if $f(x, y) < 0$ and $f(x, z) \leq 0$, then $f(x, u) < 0$ for all $u \in]y, z[$;

iv) there exist a compact subset D of K and $y^* \in D$ such that for all $x \in K \backslash D$ one has $f(x, y^*) < 0$;

v) the set $\{u \in [x, z] : g(u, y) \leq 0\}$ is closed for all $x, z \in K$;

vi) the relative algebraic interior of K is nonempty.

Suppose that f is pseudomonotone with respect to g and assumptions i)–iv) hold, or that the bifunctions f, g are a quasimonotone pair and all assumptions i)–vi) hold.

Then at least one of the following problems has a solution $x_0 \in K$:

$$f(x_0, y) \geq 0 \quad \textit{for all } y \in K,$$
$$g(y, x_0) \leq 0 \quad \textit{for all } y \in K.$$

By choosing the bifunctions f and g appropriately, a variety of results can be produced. For instance, let X and K be as before, and let Z be a real Hausdorff locally convex space. Finally, let $C \subseteq Z$ be a proper, convex, closed cone with nonempty interior int C. Define the relations $\leq, <, \nleq$ and $\nless$ on Z by

$$\begin{aligned} x \leq y &\quad\Leftrightarrow\quad y - x \in C; \\ x < y &\quad\Leftrightarrow\quad y - x \in \operatorname{int} C; \\ x \nleq y &\quad\Leftrightarrow\quad y - x \notin C; \\ x \nless y &\quad\Leftrightarrow\quad y - x \notin \operatorname{int} C. \end{aligned}$$

Given a bifunction $H: K \times K \to Z$, consider the vector equilibrium problem (VEP): find $x_0 \in K$ such that

$$H(x_0, y) \nless 0 \quad \text{for all } y \in K. \tag{9}$$

Theorem 6 can now be applied to show the existence of a solution for VEP. This is done as follows. Since the cone C has a nonempty interior by assumption, the dual cone C^* has a w^*-compact base B. (Recall that a (closed) base B of a cone W is a convex subset of W such that $0 \notin B$ and $W = \cup_{t \geq 0} tB$.) Define the real valued bifunctions f and g on K as follows:

$$f(x, y) = \max_{\phi \in B} \phi\left(H(x, y)\right),$$
$$g(x, y) = \min_{\phi \in B} \phi\left(H(x, y)\right).$$

Applying Theorem 6 to these bifunctions, one arrives at the following result [15]:

Theorem 7 *Suppose that the bifunction H satisfies the following assumptions, for all x, y, z in K:*

i) $H(x, x) \nless 0$;

ii) the set $\{x \in K: H(y, x) \ngtr 0\}$ *is closed in K;*

iii) if $H(x, y) < 0$ *and* $H(x, z) \leq 0$, *then* $H(x, u) < 0$ *for all* $u \in]y, z[$;

iv) the sets $\{u \in]x, z[: H(u, y) \ngtr 0\}$ *and* $\{u \in]x, z[: H(u, y) \nless 0\}$ *are closed;*

v) there exist a compact subset D of K and $y^* \in D$ *such that for all* $x \in K \backslash D$ *we have* $H(x, y^*) < 0$ *(coercivity);*

vi) $H(x, y) > 0 \Rightarrow H(y, x) \leq 0$ *(quasimonotonicity of H);*

vii) if $H(u, y) < 0$ *for some* $u \in]x, y[$, *then* $H(u, x) > 0$.

Then the VEP (9) has a solution.

The above result considerably strengthens a corresponding result in [2].

As another example for using Theorem 6, consider the Banach spaces X, Y, the multivalued operator T and the cone-valued map C as in the previous section on VVIP. For each $x \in K$, choose a w^*-compact base $B(x)$ of the dual cone $C^*(x)$. Now define the bifunctions f and g as follows:

$$f(x, y) = \max_{\substack{\phi \in B(x) \\ A \in Tx}} \phi\left(A(y - x)\right),$$
$$g(x, y) = \min_{\substack{\phi \in B(y) \\ A \in Tx}} \phi\left(A(y - x)\right).$$

Then, applying Theorem 6, one can show Theorem 4 as a corollary, for a set K with nonempty relative algebraic interior. Other variants of Theorem 4 can also be deduced [15].

In conclusion, this article demonstrates that generalized monotonicity rather than monotonicity is sufficient to establish the existence of solutions for VIP, VVIP, EP and VEP. A more extensive survey can be found in [16].

Finally, the reader interested in recent results on the relevance of generalized monotone VIP for the general economic equilibrium is referred to [18].

See also

- ▶ Equilibrium Networks
- ▶ Fejér Monotonicity in Convex Optimization
- ▶ Financial Equilibrium
- ▶ Generalized Monotone Multivalued Maps
- ▶ Generalized Monotone Single Valued Maps
- ▶ Hemivariational Inequalities: Applications in Mechanics
- ▶ Hemivariational Inequalities: Eigenvalue Problems
- ▶ Hemivariational Inequalities: Static Problems
- ▶ Nonconvex Energy Functions: Hemivariational Inequalities
- ▶ Oligopolistic Market Equilibrium
- ▶ Quasidifferentiable Optimization
- ▶ Quasidifferentiable Optimization: Algorithms for Hypodifferentiable Functions

- ▶ Quasidifferentiable Optimization: Algorithms for QD Functions
- ▶ Quasidifferentiable Optimization: Applications
- ▶ Quasidifferentiable Optimization: Applications to Thermoelasticity
- ▶ Quasidifferentiable Optimization: Calculus of Quasidifferentials
- ▶ Quasidifferentiable Optimization: Codifferentiable Functions
- ▶ Quasidifferentiable Optimization: Dini Derivatives, Clarke Derivatives
- ▶ Quasidifferentiable Optimization: Exact Penalty Methods
- ▶ Quasidifferentiable Optimization: Optimality Conditions
- ▶ Quasidifferentiable Optimization: Stability of Dynamic Systems
- ▶ Quasidifferentiable Optimization: Variational Formulations
- ▶ Quasivariational Inequalities
- ▶ Sensitivity Analysis of Variational Inequality Problems
- ▶ Solving Hemivariational Inequalities by Nonsmooth Optimization Methods
- ▶ Spatial Price Equilibrium
- ▶ Traffic Network Equilibrium
- ▶ Variational Inequalities
- ▶ Variational Inequalities: F. E. Approach
- ▶ Variational Inequalities: Geometric Interpretation, Existence and Uniqueness
- ▶ Variational Inequalities: Projected Dynamical System
- ▶ Variational Principles
- ▶ Walrasian Price Equilibrium

References

1. Avriel M, Diewert WE, Schaible S, Zang I (1988) Generalized concavity. Plenum, New York
2. Bianchi M, Hadjisavvas N, Schaible S (1997) Vector equilibrium problems with generalized monotone bifunctions. J Optim Th Appl 92:527–542
3. Bianchi M, Schaible S (1996) Generalized monotone bifunctions and equilibrium problems. J Optim Th Appl 90:31–43
4. Blum E, Oettli W (1994) From optimization and variational inequalities to equilibrium problems. Math Student 63:123–145
5. Brezis H (1968) Equations et inéquations non linéaires dans les espaces vectoriels en dualité. Ann Inst Fourier 18:115–175
6. Brezis H, Nirenberg L, Stampacchia G (1972) A remark on Ky Fan's minimax principle. Boll Unione Mat Ital 6:293–300
7. Browder F (1966) Existence and approximations of solution of nonlinear variational inequations. Proc Nat Acad Sci USA 56:419–425
8. Cottle RW, Yao JC (1992) Pseudomonotone complementarity problems in Hilbert space. J Optim Th Appl 75:281–295
9. Daniilidis A, Hadjisavvas N (1995) Variational inequalities with quasimonotone multivalued operators. Preprint
10. Daniilidis A, Hadjisavvas N (1996) Existence theorems for vector variational inequalities. Bull Austral Math Soc 54:473–481
11. Daniilidis A, Hadjisavvas N (1999) Characterization of nonsmooth semistrictly quasiconvex and strictly quasiconvex functions. J Optim Th Appl 102:525–536
12. Daniilidis A, Hadjisavvas N (1999) On the subdifferentials of quasiconvex and pseudoconvex functions and cyclic monotonicity. J Math Anal Appl 237:30–42
13. Giannessi F (1980) Theorems of the alternative: quadratic programs and complementarity problems. In: Cottle RW, Giannessi F, Lions JL (eds) Variational inequalities and complementarity problems. Wiley, New York, pp 151–186
14. Hadjisavvas N, Schaible S (1996) Quasimonotone variational inequalities in Banach spaces. J Optim Th Appl 90:95–111
15. Hadjisavvas N, Schaible S (1998) From scalar to vector equilibrium problems in the quasimonotone case. J Optim Th Appl 96:297–309
16. Hadjisavvas N, Schaible S (1998) Quasimonotonicity and pseudomonotonicity in variational inequalities and equilibrium problems. In: Crouzeix JP, Martinez-Legaz JE, Volle M (eds) Generalized Convexity, Generalized Monotonicity: Recent Developments. Kluwer, Dordrecht, pp 257–275
17. Hartman GJ, Stampacchia G (1966) On some nonlinear elliptic differential functional equations. Acta Math 115:271–310
18. John R (1998) Variational inequalities and pseudomonotone functions: Some characterizations. In: Crouzeix JP, Martinez-Legaz JE, Volle M (eds) Generalized Convexity, Generalized Monotonicity: Recent Developments. Kluwer, Dordrecht, pp 291–301
19. Karamardian S (1976) Complementarity over cones with monotone and pseudomonotone maps. J Optim Th Appl 18:445–454
20. Konnov IV (1998) On quasimonotone variational inequalities. J Optim Th Appl 99:165–181
21. Oettli W (1997) A remark on vector-valued equilibria and generalized monotonicity. Acta Math Vietnam 22:213–221
22. Patriksson M (1999) Nonlinear programming and variational inequality problems. Kluwer, Dordrecht

23. Schaible S, Yao JC (1995) On the equivalence of nonlinear complementarity problems and least element problems. Math Program 70:191–200
24. Yao JC (1994) Multi-valued variational inequalities with K-pseudomonotone operators. J Optim Th Appl 83:391–403
25. Zarantonello EH (1971) Projections on convex sets in Hilbert space and spectral theory. In: Zarantonello EH (ed) Contributions to Nonlinear Functional Analysis. Acad. Press, New York, pp 237–424
26. Zeidler E (1988) Nonlinear functional analysis and its applications, vol IV. Springer, Berlin

Generalized Networks

GN

JEFFERY L. KENNINGTON, KAREN R. LEWIS
Southern Methodist University, Dallas, USA

MSC2000: 90C35

Article Outline

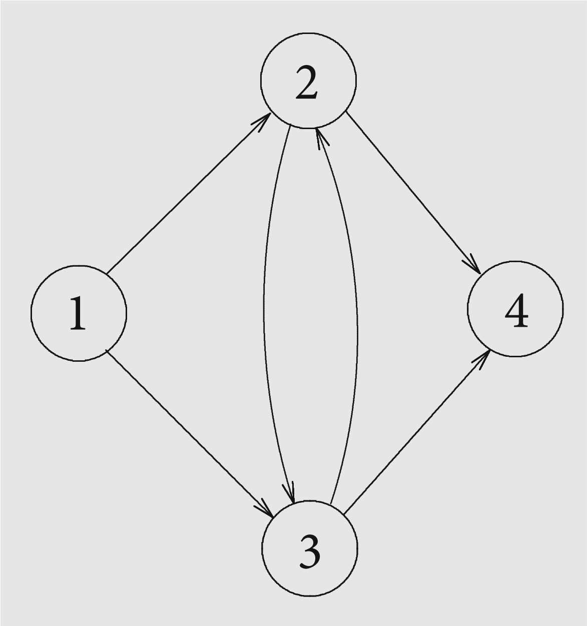
Keywords

Network flows; Generalized networks; Flows with gains

A *network* is composed of two types of entities: *arcs* and *nodes*. The nodes represent locations or terminals, and the arcs represent one-way links connecting pairs of nodes. The arc (i, j) links node i to node j and the flow is from i to j. The structure of a network can be displayed by a drawing, as illustrated in Fig. 1. The structure of a network may also be represented by a *node-arc incidence matrix* A, where A_{ik} is 1 if arc k is directed away from node i, A_{ik} is -1 if arc k is directed toward node i, and A_{ik} is 0 otherwise. Any matrix A in which each column has exactly two nonzero entries, $a+1$ and $a-1$, is called a node-arc incidence matrix. The *minimum cost network flow problem* is a linear program, say

$$\min_x \{c'x\colon\ Ax = b, l \le x \le u\},$$

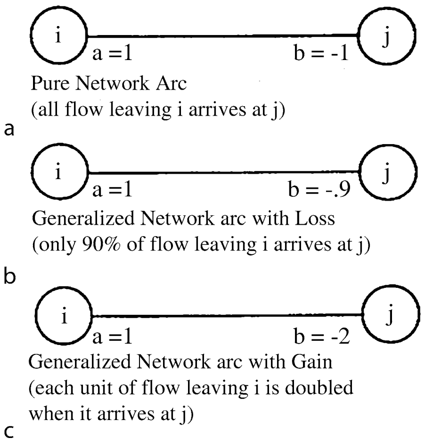
where A is a node-arc incidence matrix. The *generalized network problem*, as its name implies, is a generalization of the minimum cost network flow problem, also referred to as the *pure network problem*.



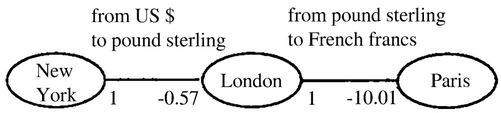
Generalized Networks, Figure 1
Example network with nodes 1, 2, 3, 4 and arcs (1, 2), (1, 3), (2, 3), (2, 4), (3, 2), (3, 4)

Let f denote the flow in arc (i, j) in a pure network. A characteristic of this model is that the f units which depart node i must arrive at node j. Many real applications violate this assumption. In a pipeline distribution network, liquid or gas will be lost due to leakage. In a network carrying a perishable commodity, a certain percentage of the commodity will be lost as it moves along the arcs. For these cases, flow may be lost as it traverses certain arcs. However, if an arc represents holding money in a savings account over a period of time, the value at the end of the period will equal the initial investment plus the interest earned. An arc in a generalized network permits flow to increase, decrease, or remain the same as it traverses the arc. This is illustrated in Fig. 2 for the arc (i, j). Each end of the arc has a constant (*multiplier*) associated with it, which determines the gain or loss during traversal. For the pure network arc, the +1 and -1 correspond to the coefficients in the node-arc incidence matrix.

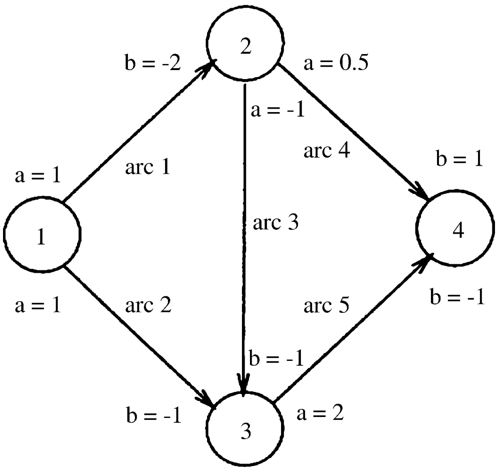
Generalized network models are also used to change units in a flow model. The arcs illustrated in Fig. 3 con-


i
j
a =1
b = -1
Pure Network Arc
(all flow leaving i arrives at j)
a
i
j
a =1
b = -.9
Generalized Network arc with Loss
(only 90% of flow leaving i arrives at j)
b
i
j
a =1
b = -2
Generalized Network arc with Gain
(each unit of flow leaving i is doubled
when it arrives at j)
c

Generalized Networks, Figure 2
Different types of generalized network arcs



Generalized Networks, Figure 3
Generalized network arcs to convert currency



Generalized Networks, Figure 4
Sample generalized network

vert from US dollars to pound sterling, and from pound sterling to French francs. That is, dollars which depart New York are converted to pounds when they arrive at London. Pounds leaving London are converted to francs when they arrive at Paris. This is also useful to convert from machine-hours to units of finished parts or pallets to truck loads.

In its most general form, the generalized network problem is a linear program with the special feature that each column of the constraint matrix has at most two nonzero entries. Let G be an $m \times n$ matrix with full row rank having this feature. Let c, l, and u be n-component vectors, and r an m-component vector. Let $Y = \{x\colon Gx = r, l \le x \le u\}$, and assume that $Y \neq \emptyset$. The generalized network problem is to find an n-component vector $\widetilde{x}$, such that $c\widetilde{x} = \min_x \{cx\colon\ Gx = r, l \le x \le u\}$. For the generalized network model illustrated in Fig. 4, G is

nodes\arcs	1	2	3	4	5
1	1	1	0	0	0
2	−2	0	−1	0.5	0
3	0	−1	−1	0	2
4	0	0	0	1	−1

For each arc, an arbitrary orientation has been assigned so that an arc is defined by the following tuple: (from node, to node, from-node multiplier, to-node multiplier, cost, lower bound, upper bound).

Some authors and computer codes require that the from-node multiplier be 1. The above model can be converted to this form via the variable substitution $\overline{x}_k = a_k x_k$ for $k = 1, \ldots, n$, where a_k is the from-node multiplier for arc k. However, this restriction causes some difficulty if the generalized network solver is ever adapted to solve the integer generalized network model. The code developed by J.L. Kennington and R.A. Mohamed [8] (RAMSES) allows for arbitrary multipliers on both ends of each arc. Other authors assume that the lower bounds are all zero. The above model can be converted to this form via the variable substitution $\overline{x}_k = x_k - l_k$ for $k = 1, \ldots, n$, where l_k is the lower bound for arc k.

Many of the computer codes that have been developed for the generalized network problem are specializations of the *primal simplex algorithm*. These specializations exploit the graphical structure of the basis and solve systems of equations on a graph rather than with standard matrix operations. Let B be a nonsingular $m \times m$ submatrix of G, and N be the submatrix composed of the remaining $n - m$ columns of G. By imposing similar partitions on c, x, and u, the generalized network problem is represented as

$$\begin{cases} \min & c^B x^B + c^N x^N \\ \text{s.t.} & B x^B + N x^N = r, \\ & l^B \le x^B \le u^B, \\ & l^N \le x^N \le u^N. \end{cases}$$

Any solution (x^B, x^N) in which $x_i^N \in \{l_i^N, u_i^N\}$ and $x^B = B^{-1}(r - Nx^N)$ is called a *basic solution* with respect to the basis B. A feasible solution that is also basic is called a *basic feasible solution* BFS. Each iteration of the primal simplex algorithm corresponds to moving from one BFS to another BFS so that the objective function value never increases, proceeding until an optimum is reached.

The *dual variables* associated with a BFS are given by $\pi = c^B B^{-1}$ and the *reduced costs* are given by $\lambda = c - \pi\ G$. The optimality conditions for a given primal-dual pair are

$$\begin{cases} \lambda_j > 0 & \Rightarrow x_j = l_j, \\ \lambda_j = 0 & \Rightarrow l_j \le x_j \le u_j, \\ \lambda_j < 0 & \Rightarrow x_j = u_j, \end{cases}$$

for each j. Using this notation, an iteration of the primal simplex algorithm is as in the table above.

```
PROCEDURE primal simplex iteration
1    Let $\pi$ be a solution to $\pi B = c^B$
2    Set $\lambda^N \leftarrow c^N - \pi N$
3    Select $q$ such that
         $\lambda_q^N < 0$ and $x_q^N = l_q^N$, or
         $\lambda_q^N > 0$ and $x_q^N = u_q^N$
4a   IF no such $q$ exists,
     THEN the solution is an optimum.
4b   IF $x_q^N = l_q^N$,
     THEN $\Delta \leftarrow 1$
     ELSE $\Delta \leftarrow -1$
5    Let $y$ be a solution to $By = N_{\cdot q}$, where $N_{\cdot q}$ is
     the $q$th column of $N$
6    Set
         $d_i \leftarrow \begin{cases} \left|\frac{(x_i^B - l_i^B)}{y_i}\right| & \text{for } \Delta y_i > 0, \\ \left|\frac{(x_i^B - u_i^B)}{y_i}\right| & \text{for } \Delta y_i < 0, \\ \infty & \text{otherwise} \end{cases}$
7    Let $s = \text{argmin}\{d_i : i = 1, \dots, n - m\}$.
8    IF $d_s > u_q^N - l_q^N$
     THEN DO Case 1
     ELSE DO Case 2.
     Case 1.
       $x^B \leftarrow x^B - \Delta(u_q^N - l_q^N)y$
       IF $\Delta = 1$,
       THEN DO $x_q^N \leftarrow u_q^N$
       ELSE DO $x_q^N \leftarrow l_q^N$
     Case 2.
       $x^B \leftarrow x^B - \Delta d_s y$
       $x_q^N \leftarrow x_q^N + \Delta d_s$.
       Interchange:
       the $q$th column of $N$ and
       the $s$th column of $B$.
```

By re-ordering the rows and columns of B, it can be displayed in block diagonal form as follows:

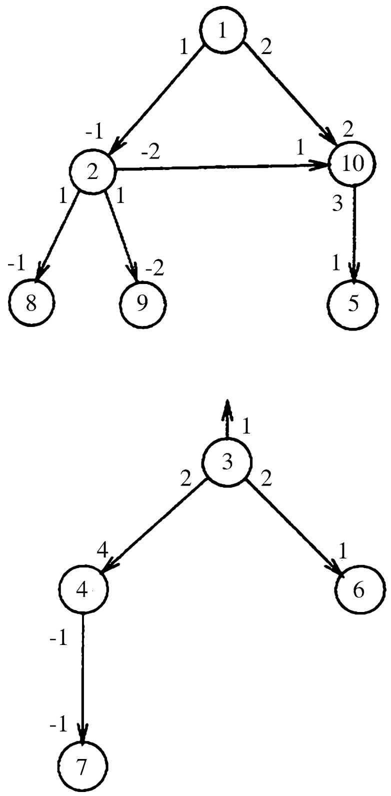
$$\overline{B} = \begin{pmatrix} \overline{B}^1 & & \\ & \ddots & \\ & & \overline{B}^p \end{pmatrix}.$$

For example, the basis

$$B = \begin{pmatrix} 1 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 1 & 1 & -2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 2 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 4 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -2 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 3 \end{pmatrix}$$

can be displayed as $\overline{B}$ equals

$$\begin{array}{cccccc|cccc} 1 & 2 & & & & & & & & \\ -1 & & 1 & 1 & -2 & & & & & \\ & 2 & & & 1 & 3 & & & & \\ & & -1 & & & & & & & \\ & & & -2 & & & & & & \\ & & & & & 1 & & & & \\ \hline & & & & & & 1 & & & \\ & & & & & & & -1 & & \\ & & & & & & & -1 & 4 & \\ & & & & & & 2 & & 2 & 1 \end{array}$$



Generalized Networks, Figure 5
A display of the basis *B*

Generalized Networks, Table 1
Label for the basis illustrated in Fig. 5

Node	Pred	Thrd	Card	Last Node
1	10	2	1	1
2	1	8	3	9
3	3	4	4	6
4	3	7	2	7
5	10	1	1	5
6	3	3	1	6
7	4	6	1	7
8	2	9	1	8
9	2	10	1	9
10	2	5	2	5

with $p = 2$ and row order 1, 2, 10, 8, 9, 5, 6, 7, 4, and 3. A display of the graph corresponding to B is illustrated in Fig. 5. The direction of the arcs was selected arbitrarily.

A connected network having exactly one cycle (such as the upper component in Fig. 5) is called a *one-tree*. An arc which is incident to a single node (such as the arc corresponding to the last column of $\overline{B}$) is called a *root arc*. A connected network on k nodes having $k - 1$ regular arcs and one root arc is called a *rooted tree* (such as the lower component in Fig. 5). It has been known from at least the 1960s that the connected components of a generalized network are either one-trees or rooted trees ([5,7]). This structure can be exploited in solving the systems $\pi B = c^B$ and $By = N{\cdot}q$ needed in the simplex algorithm, the details of which appear in [6].

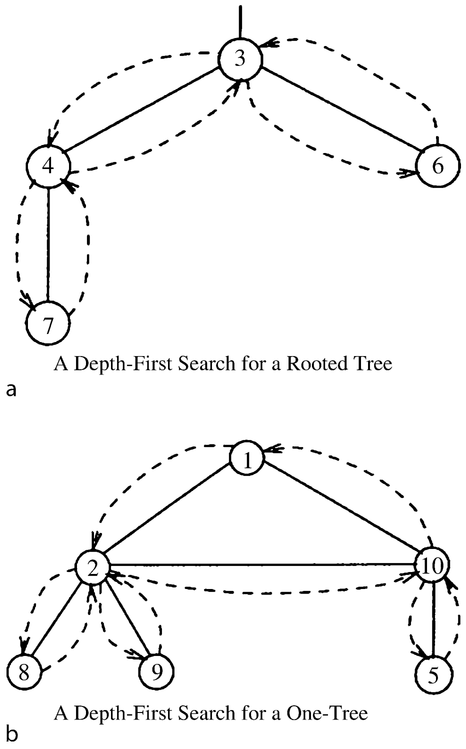
In software implementations of the primal simplex algorithm, the basis of a generalized network is maintained using a special data structure. Using the rooted tree illustrated in Fig. 5, one may imagine a line around the contours of the tree as illustrated in Fig. 6a, which is known as a *depth-first search*. For this example, the nodes in this search are ordered 3, 4, 7, 4, 3, 6, 3. An order called *pre-order* is obtained by eliminating all duplicate occurrences (i. e. 3, 4, 7, 6). The label which gives the next node in the pre-order is called the *thread*.

Three additional labels are generally used to maintain the basis. The *predecessor* of node v, denoted $p(v)$ is the first node encountered on the path from v to the root. For root nodes, we adopt the convention $p(v) = v$. If the arc linking v and $p(v)$ were deleted, then there would be two trees, one containing v and the other excluding v. The tree containing v is said to be rooted at v. The *cardinality* of v is defined to be the number of nodes in the tree rooted at v. The *last node* of v is defined to be the last node in the tree rooted at v when the nodes are taken in thread (pre-order) order.

The data structure used to represent a rooted tree is extended for the one-tree in an obvious way. The cycle in the one-tree plays the role of the root node. The predecessor label of the nodes in the cycle point to the next node in the cycle. That is, beginning with any node in the cycle, say v, the sequence v, $p(v)$, $p(p(v))$, … identifies all nodes in the cycle. The convention adopted for the thread is that traversal around the cycle using the thread is in the opposite direction to that using the predecessor. The pre-order for a one-tree is illustrated in Fig. 6b and the four labels for the basis illustrated in Fig. 5 are given in Table 1. Using these

Generalized Networks, Table 2
Survey of generalized network codes, where A stands for Assembly and F for FORTRAN

Code	Lang	Authors
NETG 1973	F	F. Glover, D. Klingman, J. Stutz
- 1973	F	W. Langley
- 1981	F	D. Adolphson, L. Heum
GENNET 1984	F	G. Brown, R. McBride
GWHIZNET 1984	A	J. Tomlin
GRNET 1985	F	M. Engquist M. Chang
LPNETG 1985	F	J. Mulvey, S. Zenios
- 1986	F	I. Ali, A. Charles, T. Song
GRNET-K (parallel) 1987	F	M. Chang, M. Engquist, M. Finkel, R. Meyer
PGRNET (parallel) 1987	F	R. Clark, R. Meyer, M. Chang
GNO/PC 1988	C	W. Nulty, M. Trick
GRNET-A 1988	A	M. Chang, M. Cheng, C. Chen
GENFLO 1989	F	M. Ramamurti
GRNET2 (serial) 1989	F	R. Clark, R. Meyer, M. Chang
TPGRNET (parallel) 1989	F	R. Clark, R. Meyer
NETPD 1994	F	N. Curet
RAMSES 1997	C	J. Kennington, R. Mohamed



A Depth-First Search for a Rooted Tree
a

A Depth-First Search for a One-Tree
b

Generalized Networks, Figure 6
Depth-first search illustrated

labels and the ideas presented in [2], all operations of the primal simplex algorithm can be performed directly on the *basis forest* composed of rooted trees and one-trees.

Since the generalized network problem is a specially structured linear program, any of the LP algorithms can be used to solve the network problem. By utilizing the structure of the network, however, any of the LP algorithms can be specialized to reduce the solution time. A specialization of the *dual simplex algorithm* may be found in [8] and a primal-dual procedure may be found in [4]. The relaxation method of Bertsekas has also been extended for the generalized network case (see [3]). The interior point algorithm (see [9]) could also be specialized for this problem.

The first specialized software for the generalized network problem was developed in the early 1970's. A partial list of codes which have been developed may be found in Table 2. An extensive list of applications of the generalized network model may be found in [1] and in [10].

See also

- ▶ Auction Algorithms
- ▶ Communication Network Assignment Problem
- ▶ Directed Tree Networks
- ▶ Dynamic Traffic Networks
- ▶ Equilibrium Networks
- ▶ Evacuation Networks
- ▶ Maximum Flow Problem
- ▶ Minimum Cost Flow Problem
- ▶ Network Design Problems
- ▶ Network Location: Covering Problems
- ▶ Nonconvex Network Flow Problems
- ▶ Piecewise Linear Network Flow Problems
- ▶ Shortest Path Tree Algorithms
- ▶ Steiner Tree Problems
- ▶ Stochastic Network Problems: Massively Parallel Solution
- ▶ Survivable Networks
- ▶ Traffic Network Equilibrium

References

1. Ahuja RK, Magnanti TL, Orlin JB (1993) Network flows: theory, algorithms and applications. Prentice-Hall, Englewood Cliffs, NJ
2. Barr R, Glover F, Klingman D (1979) Enhancement of spanning tree labeling procedures for network optimization. INFOR 17:16–34
3. Bertsekas D, Tseng P (1988) Relaxation methods for minimum cost ordinary and generalized network flow problems. Oper Res 36(1):93–114
4. Curet ND (1994) An incremental primal-dual method for generalized networks. Comput Oper Res 21(10):1051–1059
5. Dantzig GB (1963) Linear programming and extensions. Princeton Univ. Press, Princeton
6. Helgason RV, Kennington JL (1995) Primal simplex algorithms for minimum cost network flows. In: Ball MO, Magnanti TL, Monma CL, Nemhauser GL (eds) Handbook Oper. Res. and Management Sci. vol 7, Elsevier, Amsterdam
7. Johnson E (1966) Networks and basic solutions. Oper Res 14:619–623
8. Kennington JL, Mohamed RA (1997) An efficient dual simplex optimizer for generalized networks. In: Barr R, Helgason R, Kennington J (eds) Interfaces in Computer Sci. and Oper. Res.: Adv. in Metaheuristics, Optimization, and Stochastic Modeling Techniques. Kluwer, Dordrecht
9. Lustig IJ, Martsten RE, Shanno DF (1994) Interior point methods for linear programming: computational state of the art. ORSA J Comput 6(1):1–14
10. Rardin RL (1998) Optimization in operations research. Prentice-Hall, Englewood Cliffs, NJ

Generalized Nonlinear Complementarity Problem

Michael M. Kostreva
Department Math. Sci., Clemson University,
Clemson, USA

MSC2000: 90C33

Article Outline

Keywords

Nonlinear complementarity problem; Generalized complementarity

The *complementarity problem* and its generalizations are now established as an important class of applied mathematical problems. For these problems, there exists a body of theoretical results, algorithms for computing solutions and many applications from engineering to economics and from theoretical physics to computer science. A recent survey, [6], describes some of this progress, including applications in some major industrial research laboratories in the United States. Covered there are models for elasto-hydrodynamic lubrication of bearings (automotive industry) and spatial price equilibrium (telecommunications firm). The application of complementarity allowed engineers and analysts to comprehend and solve a new range of problems which had been out of reach. It is now well documented that other approaches do not adequately model these application problems while complementarity handles them.

Two main *generalizations of the nonlinear complementarity problem* were made:

- Generalization of the ordering to that of a cone (see [3]).
- Generalization to several functions per index (see [1], and [7]).

The first of these generalizations was applied to solve an elasto-hydrodynamic lubrication problem in [5], while the second was applied in [7] to solve a more complex mixed lubrication problem. These particular problems were studied in the past without complementarity models, but it is now recognized that the earlier attempts were incomplete, and failed to comprehend the main features of the physical situation.

In recent years, the second *generalized complementarity problem* above has been reconsidered and a related problem, the generalized order complementarity problem has been studied. It was known for some time that under certain conditions on the functions involved, there exists a solution to the linear generalized complementarity problem. See [1]. Recently, more extensive results have been obtained. For example, B.P. Szanc [8] developed a theory and algorithms for nonlinear functions of the class P, thereby extending the work of G.J. Habetler and M.M. Kostreva [2]. Results for the *infinite-dimensional version* of the generalized order complementarity problem are presented in [4].

The *nonlinear complementarity problem* is as follows: Given $f\colon \mathbf{R}^n \to \mathbf{R}^n$, find $x \in \mathbf{R}^n$ satisfying $x \geq 0$, $f(x) \geq 0$, and $x^\intercal f(x) = 0$. The most general set of conditions known for existence and uniqueness of solutions for this problem (even removing the requirement for continuity of f) are in [2].

Considering the generalized complementarity with cone ordering, let K be a pointed, solid cone in $\mathbf{R}^n$ and let

$$K^* = \left\{y \in \mathbf{R}^n\colon\ x^\top y \geq 0 \text{ for all } x \in K\right\},$$

and let $f\colon \mathbf{R}^n \to \mathbf{R}^n$. The *generalized complementarity problem* (f, K) is to find $x \in \mathbf{R}^n$ satisfying $x \in K$, $f(x) \in K^*$, and $x^\intercal f(x) = 0$.

Finally, the generalization with multiple functions per index is as follows: $f_{ij}\colon \mathbf{R}^n \to \mathbf{R}$, find $x \in \mathbf{R}^n$ satisfying $x_j \geq 0$, $f_{ij}(x) \geq 0$, $i \in I_j$, $j = 1, \dots, n$, $x_j \cdot \prod f^i_j(x) = 0$, $i \in I_j$, $j = 1, \dots, n$. Here the product of the variable x_j with the product of functions ($|I_j|$ of them), plays the role of the complementarity condition.

See also

▶ Convex-simplex Algorithm
▶ Equivalence Between Nonlinear Complementarity Problem and Fixed Point Problem
▶ Integer Linear Complementary Problem
▶ LCP: Pardalos–Rosen Mixed Integer Formulation
▶ Lemke Method
▶ Linear Complementarity Problem
▶ Linear Programming
▶ Order Complementarity
▶ Parametric Linear Programming: Cost Simplex Algorithm
▶ Principal Pivoting Methods for Linear Complementarity Problems
▶ Sequential Simplex Method
▶ Topological Methods in Complementarity Theory

References

1. Cottle RW, Dantzig GB (1970) A generalization of the linear complementarity problem. J Combin Th 8:79–90
2. Habetler GJ, Kostreva MM (1978) On a direct algorithm for nonlinear complementarity problems. SIAM J Control Optim 16:504–511
3. Habetler GJ, Price AL (1971) Existence theory for generalized nonlinear complementarity problems. J Optim Th Appl 7:223–239
4. Isac G, Kostreva M (1991) The generalized order complementarity problem. J Optim Th Appl 71:517–534
5. Kostreva MM (1984) Elasto-hydrodynamic lubrication: A non-linear complementarity problem. Internat J Numer Methods in Fluids 4:377–397
6. Kostreva MM (1990) Recent results on complementarity models for engineering and economics. INFOR 28:324–334
7. Oh KP (1986) The formulation of the mixed lubrication problem as a generalized nonlinear complementarity problem. Trans ASME, J Tribology 108:598–604
8. Szanc BP (1989) The generalized complementarity problem. Rensselaer Polytech. Inst., Troy, NY, PhD Diss.

Generalized Outer Approximation

S. Leyffer
Department Math., University Dundee, Dundee, UK

MSC2000: 90C11, 90C30, 49M20

Article Outline

Keywords

Mixed integer nonlinear programming; Outer approximation; Branch and bound

This article deals with the solution of *mixed integer nonlinear programming* (MINLP) problems of the form

$$(P)\begin{cases} \min\limits_{x,y} & f(x,y) \\ \text{s.t.} & g(x,y) \le 0 \\ & x \in X,\ y \in Y \text{ integer}. \end{cases}$$

Throughout the following general assumptions are made:

A1) f and g are twice continuously differentiable and convex functions;

A2) X and Y are nonempty compact convex (polyhedral) sets; and

A3) a constraint qualification holds at the solution of every NLP subproblem obtained by fixing the integer variables y.

MINLP problems arise in a range of engineering applications (see, e. g., [8] and [10] and references therein).

A class of methods for MINLP problems is discussed, which provide an alternative to nonlinear branch and bound (cf. ► MINLP: Branch and bound methods) [3]. These algorithms are based on the concept of defining an *MILP master problem*. Relaxations of such a master problem are then used in constructing algorithms for solving the MINLP problem.

The methods presented here are a generalization of outer approximation proposed by M.A. Duran and I.E. Grossmann [4] (see also [14]) and of LP/NLP based branch and bound of I. Quesada and Grossmann [13].

The next section presents the reformulation of (P) as an MILP master program. Based on this reformulation two algorithms are presented in the following sections which solve a finite sequence of NLP subproblems and MILP or *MIQP master problems*, respectively. The final section shows how the re-solution of these master problems can be avoided by updating their branch and bound trees.

Outer Approximation of (P)

In this section the MINLP model problem (P) is reformulated as an MILP problem using outer approximation. The reformulation employs projection onto the integer variables and linearization of the resulting NLP subproblems by means of supporting hyperplanes. The convexity assumption allows an MILP formulation to be given where all supporting hyperplanes are collected in a single MILP.

In order to improve the readability of the material, the reformulation is first done under the simplifying assumption that all integer assignments $y \in Y$ are feasible. Next a rigorous treatment of infeasible subproblems is outlined, correcting an inaccuracy in [4] and [14], which could cause the algorithm to cycle. Finally, the two parts are combined and the correctly reformulated MILP master program is presented.

The reformulation presented in the next section affords new insight into Outer Approximation. It can be seen, for example, that it suffices to add the linearizations of *strongly active* constraints to the master program. This is very important since it reduces the size of the MILP master program relaxations that are solved in the outer approximation algorithms.

When All $y \in Y$ Are Feasible

In this subsection the simplifying assumption is made that all $y \in Y$ are feasible. The first step in reformulating (P) is to define the *NLP subproblem*

$$NLP(y^j)\begin{cases} \min\limits_{x} & f(x,y^j) \\ \text{s.t.} & g(x,y^j) \le 0 \\ & x \in X \end{cases}$$

in which the integer variables are fixed at the value $y = y^j$. By defining $v(y^j)$ as the optimal value of the subproblem NLP(y^j) it is possible to express (P) in terms of a projection on to the y variables, that is

$$\min_{y^j \in Y}(v(y^j)). \tag{1}$$

The assumption that all $y \in Y$ are feasible implies that *all* subproblems are feasible. Let x^j denote an optimal solution of NLP(y^j) for $y^j \in Y$ (existence of x^j follows by the compactness of X). Because a constraint qualification holds at the solution of every subproblem NLP(y^j)

for every $y^j \in Y$, it follows that (1) has the same optimal value as the problem

$$\min_{y^j \in Y}(u(y^j)), \tag{2}$$

where $u(y^j)$ is the optimal value of the following LP

$$\begin{cases} \min\limits_{x} & f^j + (\nabla f^j)^\top \begin{pmatrix} x - x^j \\ 0 \end{pmatrix} \\ \text{s.t.} & 0 \geq g^j + [\nabla g^j]^\top \begin{pmatrix} x - x^j \\ 0 \end{pmatrix} \\ & x \in X. \end{cases} \tag{3}$$

In fact, it suffices to include those linearizations of constraints about (x^j, y^j) which are strongly active at the solution of the corresponding subproblem. Here $f^j = f(x^j, y^j)$ and $\nabla f^j = \nabla f(x^j, y^j)$, etc.

It is convenient to introduce a dummy variable $\eta \in \mathbf{R}$ into (3), giving rise to the equivalent problem

$$\begin{cases} \min\limits_{x,\eta} & \eta \\ \text{s.t.} & \eta \geq f^j + (\nabla f^j)^\top \begin{pmatrix} x - x^j \\ 0 \end{pmatrix} \\ & 0 \geq g^j + [\nabla g^j]^\top \begin{pmatrix} x - x^j \\ 0 \end{pmatrix} \\ & x \in X. \end{cases}$$

The convexity assumption A1) implies that (x^i, y^i) is feasible in the inner optimization problem above for all $y^i \in Y$, where x^i is an optimal solution to NLP(y^i). Thus an equivalent MILP problem

$$(M_Y) \begin{cases} \min\limits_{x,y,\eta} & \eta \\ \text{s.t.} & \eta \geq f^j + (\nabla f^j)^\top \begin{pmatrix} x - x^j \\ y - y^j \end{pmatrix} \\ & 0 \geq g^j + [\nabla g^j]^\top \begin{pmatrix} x - x^j \\ y - y^j \end{pmatrix} \\ & \forall y^j \in Y \\ & x \in X, \quad y \in Y \text{ integer} \end{cases}$$

is obtained. That is, (M_Y) has one set of linearizations of the objective and constraint functions per integer point $y^j \in Y$.

Infeasible Subproblems

Usually, not all $y \in Y$ give rise to feasible subproblems. Defining the sets

$$\begin{aligned} T &= \left\{j\colon\ x^j \text{ optimal solution to NLP}(y^j)\right\}, \\ V &= \{y \in Y\colon\ \exists x \in X \text{ with } g(x, y) \leq 0\}. \end{aligned}$$

Then V is the set of all integer assignments y that give rise to feasible NLP subproblems and T is the set of indices of these integer variables. Then (P) can be expressed as a projection on to the integer variables

$$\min_{y^j \in V}(v(y^j)).$$

In this projection the set V replaces Y in (1). The equivalent MILP problem is now given by

$$(M_V) \begin{cases} \min\limits_{x,y,\eta} & \eta \\ \text{s.t.} & \eta \geq f^j + (\nabla f^j)^\top \begin{pmatrix} x - x^j \\ y - y^j \end{pmatrix} \\ & 0 \geq g^j + [\nabla g^j]^\top \begin{pmatrix} x - x^j \\ y - y^j \end{pmatrix} \\ & \forall j \in T \\ & x \in X, \quad y \in V \text{ integer} \end{cases}$$

obtained from (M_Y) by replacing Y by V.

It remains to find a suitable representation of the constraint $y \in V$ by means of supporting hyperplanes. The master problem given in [4] is obtained from problem (M_V) by replacing $y \in V$ by $y \in Y$. Duran and Grossmann 1986 justify this step by arguing that a representation of the constraints $y \in V$ is included in the linearizations in problem (M_V). However, it is not difficult to derive an MINLP problem where this would lead to an incorrect master problem [5], [11, p. 79].

In order to derive a correct representation of $y \in V$ it is necessary to consider how NLP solvers detect infeasibility. Infeasibility is detected when convergence to an optimal solution of a feasibility problem occurs. At such an optimum, some of the nonlinear constraints will be violated and other will be satisfied and the norm of the infeasible constraints can only be reduced by making some feasible constraints infeasible. A suitable framework for *nonlinear feasibility problems* in the context of outer approximation is

$$F(y^k)\begin{cases}\min\limits_{x} & \sum\limits_{i\in J^{\perp}} w_i^k g_i^+(x,y^k)\\ \text{s.t.} & g_j(x,y^k)\le 0, \quad j\in J,\\ & x\in X.\end{cases}$$

The constraints in F(y^k) have been divided into two sets: one that can be satisfied and another that cannot be satisfied. Infeasible subproblems now correspond to solutions of F(y^k) with $\sum_{i\in J^{\perp}} w_i^k\ g_i^+\ (x, y^k) > 0$. Most common feasibility problems such as l_1 and l_∞ as well as the feasibility filter [6] fit into this framework. The following lemma shows how solutions of F(y^k) can be used to construct a representation of $y \in V$.

Lemma 1 *If NLP(y^k) is infeasible, so that x^k solves F(y^k) with*

$$\sum_{i\in J^{\perp}} w_i^k (g_i^k)^+ > 0, \tag{4}$$

then $y = y^k$ is infeasible in the constraints

$$0\ge g_i^k + (\nabla g_i^k)^\top \begin{pmatrix}x-x^k\\ y-y^k\end{pmatrix}, \quad \forall i\in J^{\perp}$$

$$0\ge g_j^k + (\nabla g_j^k)^\top \begin{pmatrix}x-x^k\\ y-y^k\end{pmatrix}, \quad \forall j\in J,$$

for all $x \in X$. The proof of this Lemma can be found in [5, Lemma 1].

The General Case

This subsection completes the derivation of the MILP master program by combining the developments of the previous two subsections. Let the integer assignment y^k produce an infeasible subproblem and denote

$$S=\left\{k:\ \text{NLP}(y^k)\text{ infeasible}, x^k \text{ solves F}(y^k)\right\}.$$

Note that S is the complement of the set T defined in the previous subsection. It then follows directly from Lemma 1 that the constraints

$$0\ge g^k + [\nabla g^k]^\top \begin{pmatrix}x-x^k\\ y-y^k\end{pmatrix}, \quad \forall k\in S,$$

exclude all integer assignments y^k for which NLP(y^k) is infeasible. Thus a general way to correctly represent the constraints $y \in V$ in (M_V) is to add linearizations from F(y^k) when infeasible subproblems are obtained, giving rise to the following MILP master problem:

$$(M)\begin{cases}\min\limits_{x,y,\eta} & \eta\\ \text{s.t.} & \eta\ge f^j + (\nabla f^j)^\top\begin{pmatrix}x-x^j\\ y-y^j\end{pmatrix}\\ & 0\ge g^j + [\nabla g^j]^\top\begin{pmatrix}x-x^j\\ y-y^j\end{pmatrix}\\ & \forall j\in T\\ & 0\ge g^k + [\nabla g^k]^\top\begin{pmatrix}x-x^k\\ y-y^k\end{pmatrix}\\ & \forall k\in S\\ & x\in X, \quad y\in Y \text{ integer.}\end{cases}$$

The development of the preceding two subsections provides a proof of the following result:

Theorem 2 *If assumptions A1), A2) and A3) hold, then (M) is equivalent to (P) in the sense that (x^*, y^*) solves (P) if and only if it solves (M).*

Problem (M) is an MILP problem, but it is not practical to solve (M) directly, since this would require all subproblems NLP(y^j) to be solved first. This would be a very inefficient way of solving problem (P). Therefore, instead of attempting to solve M directly, relaxations of (M) are used in an iterative process that is the subject of the next section.

Linear Outer Approximation

This section describes, how relaxations of the master program (M), developed in the previous section can be employed to solve the model problem (P). The resulting algorithm is termed *linear outer approximation*. It is shown to iterate finitely between NLP subproblems and MILP master program relaxations. This algorithm is seen to be less efficient if curvature information is present in the problem. A worst-case example, in which linear outer approximation visits all integer assignments has been derived in [5]. This example motivates the introduction of a second order term into the MILP master program relaxations, resulting in a *quadratic outer approximation* algorithm which is considered in the next section.

Each iteration of the linear outer approximation algorithm chooses a new integer assignment y^i and attempts to solve NLP(y^i). Either a feasible solution x^i is obtained or infeasibility is detected and x^i is the solution of a feasibility problem F(y^i) (other pathological cases are eliminated by the assumption that the set X is compact). The algorithm replaces the sets T and S in (M) by the sets

$$
\begin{aligned}
T^i &= \left\{ j \le i\colon\ x^j \text{ solution to } \mathrm{NLP}(y^j) \right\},\\
S^i &= \left\{ k \le i\colon\ x^k \text{ solution to } \mathrm{F}(y^k) \right\}.
\end{aligned}
$$

It is also necessary to prevent any $y^j, j \in T^i$, from becoming the solution of the relaxed master problem. This can be done by including a constraint

$$\eta < \mathrm{UBD}^i,$$

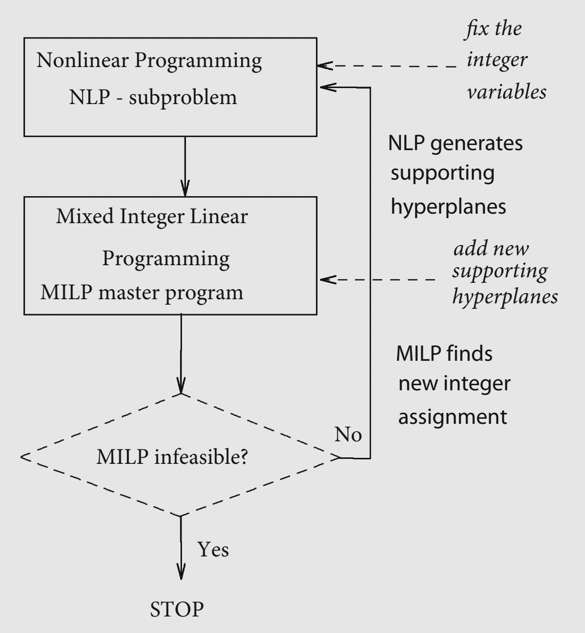
where

$$\mathrm{UBD}^i = \min \left\{ f^j\colon\ j \in T^i \right\}$$

is an upper bound on the optimum. Thus the following master problem is defined

$$
(M^i) \begin{cases}
\min\limits_{x,y,\eta} & \eta \\
\text{s.t.} & \eta < \mathrm{UBD}^i \\
& \eta \ge f^j + \nabla(f^j)^\top \begin{pmatrix} x - x^j \\ y - y^j \end{pmatrix} \\
& 0 \ge g^j + \nabla[g^j]^\top \begin{pmatrix} x - x^j \\ y - y^j \end{pmatrix} \\
& \forall j \in T^i \\
& 0 \ge g^k + \nabla[g^k]^\top \begin{pmatrix} x - x^k \\ y - y^k \end{pmatrix} \\
& \forall k \in S^i \\
& x \in X, \quad y \in Y \text{ integer.}
\end{cases}
$$

The algorithm solves (M^i) to obtain a new integer assignment y^{i+1}, and the whole process is repeated iteratively. A detailed description of the algorithm is as follows.



Generalized Outer Approximation, Figure 1
Illustration of linear outer approximation

```
Initialization: $y^0$ given:
set $i = 0,\ T^{-1} = \emptyset,\ S^{-1} = \emptyset,\ \mathrm{UBD}^{-1} = \infty$
REPEAT
  1. Solve NLP($y^i$) at F($y^i$) as appropiate. Let the solution be $x^i$.
  2. Liniarize objective and constraint functions about $(x^i, y^i)$. Set $T^i = T^{i-1} \cup \{i\}$ or $S^i = S^{i-1} \cup \{i\}$ as appropriate.
  3. IF (NLP($y^i$) feasible AND $f^i < \mathrm{UBD}^{i-1}$)
     THEN
     update current best point by setting $x^* = x^i$, $y^* = y^i$, $\mathrm{UBD}^i = f^i$
     ELSE Set $\mathrm{UBD}^i = \mathrm{UBD}^{i-1}$.
  4. Solve the current relaxation (M$^i$) of the master program (M), giving a new $y^{i+1}$. Set $i = i + 1$.
UNTIL ((M$^i$) is infeasible)
```

Algorithm 1: Linear outer approximation

The figure below illustrates the different stages of the algorithm.

The algorithm also detects whether or not (P) is infeasible. If UBD = ∞ on exit, then all integer assignments that are visited by the algorithm are infeasible

(i. e. Step 3 is not invoked). The use of upper bounds on η and the definition of the sets T^i and S^i ensure that no y^i is replicated by the algorithm. This enables a proof to be given that the algorithm terminates after a finite number of steps, provided that there is only a finite number of integer assignments.

Theorem 3 *If assumptions A1), A2) and A3) hold, and if $|Y| < \infty$, then Algorithm 1 terminates in a finite number of steps at an optimal solution of (P) or with an indication that (P) is infeasible.*

A proof of this theorem can be found in [5]. Below a brief outline of the proof is given: The optimality of x^i in NLP(y^i) implies that $\eta \geq f^i$ for any feasible point in (3). The upper bound $\eta < f^i$ therefore ensures that the choice $y = y^i$ in (M^k) has no feasible points $x \in X$. Therefore the algorithm is finite. The optimality of the algorithm follows from the convexity of f and g which ensures that the linearizations are supporting hyperplanes.

Quadratic Outer Approximation

Curvature can often play an important role in optimization. If this is the case, then an algorithm based on *linear* representations of the problem functions can be inefficient. In [5], a worst-case example is given for which linear outer approximation visits all integer points. This motivates the introduction of a curvature information into the master programs. In the remainder of this section it is shown how this can be achieved for linear outer approximation by including a second order Lagrangian term into the objective function of the MILP master programs.

These considerations have led to the development of a new algorithm based on the use of second order information. The development of such an algorithm seems contradictory at first sight, since quadratic functions do not provide underestimators of general convex functions. However, the derivation of the previous section allows the inclusion of a curvature term into the objective function of the MILP master problem. This quadratic term influences the choice of the next iterate by the algorithm without surrendering the finite convergence properties which rely on the fact that the feasible region of the master problem is an outer approximation of the feasible region of the MINLP problem P. The resulting algorithm is referred to as *quadratic outer approximation* and is obtained by replacing the relaxed master problem (M^i) by the MIQP problem (Q^i) in Step 4 of Algorithm 1. Introducing the *quadratic* term

$$q^i(x, y) = \frac{1}{2} \begin{pmatrix} x - x^i \\ y - y^i \end{pmatrix}^\top \nabla^2 \mathcal{L}^i \begin{pmatrix} x - x^i \\ y - y^i \end{pmatrix},$$

where

$$\mathcal{L}^i = \mathcal{L}(x^i, y^i, \lambda^i) = f(x^i, y^i) + (\lambda^i)^\top g(x^i, y^i)$$

is the usual Lagrangian function.

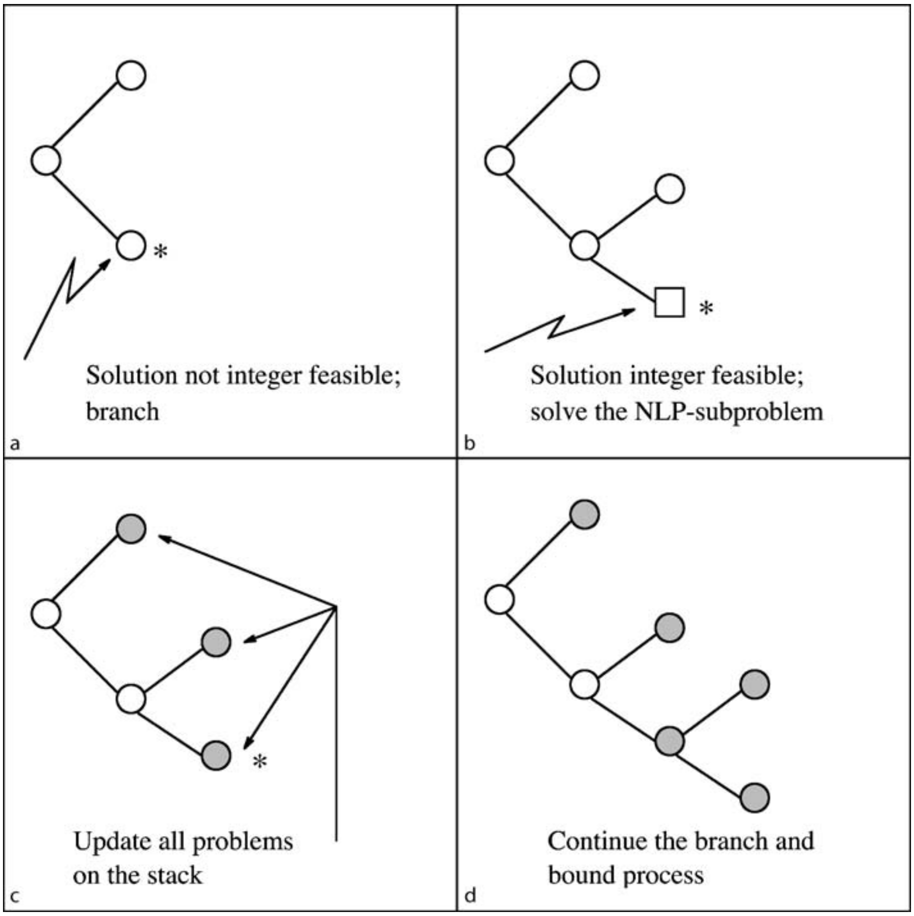
The new master problem (Q^i) can be defined as

$$(Q^i) \begin{cases} \min\limits_{x,y,\eta} & \eta + q^i(x, y) \\ \text{s.t.} & \eta < \text{UBD}^i \\ & \eta \geq f^j + (\nabla f^j)^\top \begin{pmatrix} x - x^j \\ y - y^j \end{pmatrix} \\ & 0 \geq g^j + [\nabla g^j]^\top \begin{pmatrix} x - x^j \\ y - y^j \end{pmatrix} \\ & \forall j \in T^i \\ & 0 \geq g^k + [\nabla g^k]^\top \begin{pmatrix} x - x^k \\ y - y^k \end{pmatrix} \\ & \forall k \in S^i \\ & x \in X, \quad y \in Y \text{ integer.} \end{cases}$$

Numerical experience in [11, Chapter 6] indicates that adding a curvature term reduces the number of iterations of outer approximation if general integer variables are present. However, the iteration count is not reduced for problems involving binary variables only. As a consequence these preliminary results indicate that quadratic outer approximation only improves the computation times for problems with general integer variables, as MIQP problems are usually more expensive to solve than MILP problems.

Avoiding Resolving the Master Problems

This section presents a new approach to the solution of successive master program relaxations. It has been proposed by Quesada and Grossmann [13] for a class of problems whose objective and constraint functions are linear in the integer variables and is called *LP/NLP based branch and bound* algorithm. Their approach is generalized here to cover problems with nonlinearities



Generalized Outer Approximation, Figure 2
Progress of LP/NLP based branch and bound

in the integer variables. In addition a new algorithm *QP/NLP based branch and bound* is proposed based on the quadratic master program (Q^i) which takes curvature information into account.

The motivation for the LP/NLP based branch and bound algorithm is that outer approximation usually spends an increasing amount of computing time in solving successive MILP master program relaxations. Since the MILP relaxations are strongly related to one another this means that a considerable amount of information is re-generated each time a relaxation is solved. The new approach avoids the re-solution of MILP master program relaxations by updating the branch and bound tree. This section makes extensive use of branch and bound terminology; see the extensive literature on branch and bound (e. g., [1,2,8,9,12]) for the relevant definitions.

Instead of solving successive relaxations of (M), the new algorithm solves only one MILP problem which is updated as new integer assignments are encountered *during* the tree search. Initially an NLP-subproblem is solved and the initial master program relaxation (M^0) is set up from the supporting hyperplanes at the solution of the NLP-subproblem. The MILP problem (M^0) is then solved by a branch and bound process with the exception that each time a node (corresponding to an LP problem) gives an integer feasible solution y^i, say, the process is interrupted and the corresponding NLP(y^i) subproblem is solved. New linearizations from NLP(y^i) are then added to every node on the stack, effectively updating the branch and bound tree. The branch and bound process continues in this fashion until no problems remain on the stack. At that moment all nodes are fathomed and the tree search is exhausted.

```
Initialization: $y^0$ given;
        set $i = 1,\ T^{-1} = \emptyset,\ S^{-1} = \emptyset$
Set up the initial master program:
        Solve NLP($y^0$). Let the solution be $x^0$.
        Linearize objective and constraint functions about $(x^0, y^0)$.
        Set $T^0 = \{0\}$.
        Set $x^* = x^0,\ y^* = y^0,\ \text{UBD}^0 = f^0$.
Place (M$^0$) with its integer restrictions relaxed on the stack.
WHILE (stack is not empty) DO BEGIN
  1.    Remove a problem (P′) from the stack and solve the LP giving $(x', y', \eta')$. $\eta'$ is a lower bound for all
        NLP child problems whose root is the current problem.
  2.    IF ($y'$ integer) THEN
        Set $y^i = y'$;
        Solve NLP($y^i$) or F($y^i$).
        Let the solution be $x^i$.
        Linearize objective and constraint functions about $(x^i, y^i)$.
        Set $T^i = T^{i-1} \cup \{i\}$ or $S^i = S^{i-1} \cup \{i\}$.
        Add linearizations to *all* pending problems on the stack.
        IF (NLP($y^i$)feasible AND $f^i < \text{UBD}^i$) THEN
        Update best point $x^* = x^i,\ y^* = y^i,\ \text{UBD}^{i+1} = f^i$.
        ELSE Set $\text{UBD}^{i+1} = \text{UBD}^i$.
        ENDIF
        Place (P′) back on stack; set $i = i + 1$.
        Pruning: Remove all problems from stack with $\eta' > \text{UBD}^{i+1}$.
        ELSE
        Branch on an integer variable and add two new problems to the stack.
        ENDIF
END (WHILE-LOOP)
```

Algorithm 2: LP/NLP based branch and bound

Unlike ordinary branch and bound a node cannot be assumed to have been fathomed, if it produces an integer feasible solution, since the previous solution at this node is cut out by the linearizations added to the master program. Thus only infeasible nodes can be assumed to be fathomed. In the case of MILP master programs there exists an additional opportunity for pruning. Since the LP nodes are outer approximations of the MINLP subproblem corresponding to their respective subtree a node can be regarded as fathomed if its lower bound is greater than or equal to the current upper bound UBD^i.

As in the two outer approximation algorithms the use of an upper bound implies that no integer assignment is generated twice during the tree search. Since both the tree and the set of integer variables are finite the algorithm eventually encounters only infeasible problems and the stack is thus emptied so that the procedure stops. This provides a proof of the following consequence to Theorem 3.

Corollary 4 *If assumptions A1), A2) and A3) hold, and if $|Y| < \infty$, then Algorithm 2 terminates in a finite number of steps at a solution of (P) or with an indication that (P) is infeasible.*

The figure below illustrates the progress of Algorithm 2. In i), the LP relaxation of the initial MILP has been solved and two branches added to the tree. The LP that is solved next (indicated by an $*$) does not give an integer feasible solution and two new branches are introduced. The next LP in ii) produces an integer feasible solution indicated by a box. The corresponding NLP subproblem is solved and in iii) all nodes on the stack are updated (indicated by the shaded circles) by adding the linearizations from the NLP subproblem including the upper bound UBD^i which cuts out the current assignment y^i. Then, the branch and bound process continues on the updated tree by solving the LP marked by a $*$.

If curvature information plays an important part in the problem (P), then it may be beneficial to add a quadratic term $q^i(x, y)$ to the master problem. This gives rise to *QP/NLP based branch and bound* algorithm. It differs from Algorithm 2 in two important aspects. The first difference is that QP rather than LP problems are solved in the tree search. The second difference is a consequence of the first. Since QP problems do not provide lower bounds on the MINLP problems (P), the pruning step in Algorithm 2 cannot be applied.

In preliminary numerical experiments in [11, Chapter 6] and [7] it has been observed that the LP and QP version of Algorithm 2 are usually faster than their counterparts based on Algorithm 1, often beating the latter by a factor of 2. A detailed numerical comparison of the two approaches is still outstanding.

See also

- ▶ Chemical Process Planning
- ▶ Extended Cutting Plane Algorithm
- ▶ Generalized Benders Decomposition
- ▶ MINLP: Application in Facility Location-allocation
- ▶ MINLP: Applications in Blending and Pooling Problems
- ▶ MINLP: Applications in the Interaction of Design and Control
- ▶ MINLP: Branch and Bound Global Optimization Algorithm
- ▶ MINLP: Branch and Bound Methods
- ▶ MINLP: Design and Scheduling of Batch Processes
- ▶ MINLP: Generalized Cross Decomposition
- ▶ MINLP: Global Optimization with αBB
- ▶ MINLP: Heat Exchanger Network Synthesis
- ▶ MINLP: Logic-based Methods
- ▶ MINLP: Outer Approximation Algorithm
- ▶ MINLP: Reactive Distillation Column Synthesis
- ▶ Mixed Integer Linear Programming: Mass and Heat Exchanger Networks
- ▶ Mixed Integer Nonlinear Programming

References

1. Beale EML (1978) Integer programming. In: Jacobs DAH (ed) The State of the Art in Numerical Analysis. Acad. Press, New York
2. Borchers B, Mitchell JE (1994) An improved branch and bound algorithm for mixed integer nonlinear programming. Comput Oper Res 21(4):359–367
3. Dakin RJ (1965) A tree search algorithm for mixed integer programming problems. Comput J 8:250–255
4. Duran M, Grossmann IE (1986) An outer-approximation algorithm for a class of mixed-integer nonlinear programs. Math Program 36:307–339
5. Fletcher R, Leyffer S (1994) Solving mixed integer nonlinear programs by outer approximation. Math Program 66:327–349
6. Fletcher R, Leyffer S (1997) Nonlinear programming without a penalty function. Numer Anal Report Univ Dundee, Dept Math NA/171
7. Fletcher R, Leyffer S (1998) Computing lower bounds for MIQP, branch-and-bound. SIAM J Optim 8:604–616
8. Floudas CA (1995) Nonlinear and mixed-integer optimization: Topics in chemical engineering. Oxford Univ. Press, Oxford
9. Garfinkel RS, Nemhauser GL (1972) Integer programming. Wiley, New York
10. Grossmann IE, Kravanja Z (1997) Mixed-integer nonlinear programming: A survey of algorithms and applications. In: Conn AR, Biegler LT, Coleman TF, Santosa FN (eds) Large-Scale Optimization with Applications, Part II: Optimal Design and Control. Springer, Berlin
11. Leyffer S (1993) Deterministic methods for mixed integer nonlinear programming. PhD Thesis Univ. Dundee
12. Nemhauser GL, Wolsey LA (1988) Integer and combinatorial optimization. Wiley, New York
13. Quesada I, Grossmann IE (1992) An LP/NLP based branch-and-bound algorithm for convex MINLP: optimization problems. Comput Chem Eng 16:937–947
14. Yuan X, Zhang S, Pibouleau L, Domenech S (1988) Une méthode d'optimization non linéaire en variables mixtes pour la conception de procédés. Oper Res 22:331–346

Generalized Primal-relaxed Dual Approach

GPRD

WENBIN LIU
University Kent, Canterbury, England

MSC2000: 90C26

Article Outline

Keywords

Global optimization; Primal-relaxed dual approach; GOP algorithm

Generalized primal-relaxed dual approach (GPRD) in the context of global optimization employs the Benders' idea of partitioning (see [2]) in order to exploit the structure of global optimization problems with *complicating variables* (variables which, when temporarily fixed, render the remaining optimization problem much simpler, see [4]). For the class of global optimization problem considered by the GPRD approach, fixing the values of the complicating variables reduces the given problem to a convex program, parameterized by the values of the complicating variables. In order to approximate the solution of this class of problems efficiently, the GPRD approach uses the primal and relaxed dual problems with fixed complicating variables to provide sharper upper and lower bounds of the solution respectively, following the original ideas in [2,4] and [9].

It however adopts a different way of constructing relaxed dual problems by generalizing the original method used in the GOP algorithms (see [3]) so that it can handle a wider range of global optimization problems including nonsmooth ones.

Let k, p, n, m be some positive integers. Let X and Y be two closed sets in $\mathbf{R}^n$ and $\mathbf{R}^m$ respectively. Let f, g_i, h_j ($1 \le i \le k$ and $1 \le j \le p$) be continuous functions on $\mathbf{R}^n \times \mathbf{R}^m$. Let $g = (g_1, \dots, g_k)^\top$ and $h = (h_1, \dots, h_p)^\top$.

Let us consider the following global optimization problem:

$$(OP)\ v = \begin{cases} \min\limits_{x,y} & f(x,y) \\ \text{s.t.} & g(x,y) \le 0, \\ & h(x,y) = 0, \\ & x \in X, \quad y \in Y, \\ & 1 \le i \le k, \quad 1 \le j \le p, \end{cases}$$

where X and Y are nonempty closed convex sets in $\mathbf{R}^n$ and $\mathbf{R}^m$ respectively. It is assumed that for any fixed $y \in Y$, or $x \in X$, $1 \le i \le k$, and $1 \le j \le p$, $f(\cdot, y)$, $g_i(\cdot, y)$, $f(x, \cdot)$, $g_i(x, \cdot)$ are convex functions, and $h_j(\cdot, y)$, $h_j(x, \cdot)$ are affine functions. It is also assumed that for any fixed $y \in V = \{y \in Y\colon$ there is an $x \in X\colon g(x, y) \le 0$ and $h(x, y) = 0\}$, the partial primal problem (OP) is stable in the sense that its perturbation function has a nonempty subgradient at zero point; see [1]. This assumption holds when, for instance, the Slater's constraint qualification holds for (OP) at every fixed $y \in V$, though it is more general than the Slater's.

Although the problem (OP) appears to address only a limited class of global optimization programs, it is shown in [5] that very broad mathematical programming problems can indeed be reformulated in this form by using a simple variable transformation. Furthermore, it is shown in [6] that for any fixed $y \in Y$ the reformulated problems are always stable.

It follows from the stability assumption that for any fixed $y_0 \in V$ there exist Lagrange multipliers $(\lambda_0, \mu_0) \in \mathbf{R}^p \times \mathbf{R}^k_+$ and $x_0 \in X$ such that the triplet (x_0, λ_0, μ_0) is the solution of the following saddle problem:

$$\begin{cases} g(x_0, y_0) \le 0, \\ h(x_0, y_0) = 0, \\ \mu_0^\top g(x_0, y_0) = 0, \end{cases}$$

and for any $(x, \lambda, \mu) \in X \times \mathbf{R}^p \times \mathbf{R}^k_+$

$$(SP) \quad \begin{aligned} L(x_0, y_0, \lambda, \mu) &\le L(x_0, y_0, \lambda_0, \mu_0) \\ &\le L(x, y_0, \lambda_0, \mu_0), \end{aligned}$$

where the Lagrange function of the primal problem (OP) is defined by

$$L(x, y, \lambda, \mu) = f(x, y) + \lambda^\top h(x, y) + \mu^\top g(x, y).$$

The solution (x_0, λ_0, μ_0) of (SP) can be found by solving the following partial primal problem:

$$(PP) \quad \min_{\substack{x \in X, \\ g(x, y_0) \le 0, \\ h(x, y_0) = 0}} \quad f(x, y_0),$$

which is a convex minimization problem.

For a given $y_0 \in V$, the GPRD approach finds an upper bound for the solution of (OP) by solving (PP):

$$\nu^+(y_0) = \min_{\substack{x\in X,\\ g(x,y_0)\le 0,\\ h(x,y_0)=0}} f(x, y_0).$$

The problem (PP) is in general easier to solve as it is convex. The GPRD approach then estimates a lower bound for the solution of (OP) by solving the following relaxed dual problem:

$$(RD) \quad \begin{aligned} &\nu^-(U, H)\\ &= \min_{y\in V} \max_{(\lambda_t,\mu_t)\in U} \min_{x\in X} H^{(\lambda_t,\mu_t)}(x, y), \end{aligned}$$

where $U = \{(\lambda_t, \mu_t) \in \mathbf{R}^p \times \mathbf{R}^m_+ : 1 \le t \le N\}$, and the mapping $H: U \to C^0(X \times Y)$ is such that the function $H^{(\lambda_t,\mu_t)}(\cdot,\cdot)$ is continuous and satisfies that for any fixed $(\lambda_t, \mu_t) \in U$,

$$\begin{aligned} &L(x, y, \lambda_t, \mu_t)\\ &= f(x, y) + \lambda_t^\top h(x, y) + \mu_t^\top g(x, y)\\ &\ge H^{(\lambda_t,\mu_t)}(x, y), \quad \forall (x, y) \in X \times Y. \end{aligned}$$

In the GPRD approach, the set U is usually constructed to include the multipliers (λ, μ) obtained from solving the problem (PP) above.

The *generalized primal-relaxed dual algorithm* is to construct, for $n = 0, 1, \ldots,$ a sequence of elements $y_n \in Y$, sets U_n, and functions $H_n^{(\lambda_t,\mu_t)}$ for each $(\lambda_t, \mu_t) \in U_n$ such that $\nu^+(y_n) - \nu^-(U_n, H_n) \to 0$ as $n \to 0$. The selections of U_n and $H_n^{(\lambda_t,\mu_t)}$ are clearly not unique but they have to be constructed so that the *global* solutions of the relaxed-dual problems (RD) can be solved efficiently. In the literature U_n is set to be the unit of all the Lagrange multipliers (λ, μ) of the partial primal problems (PP) with $y = y_m$ $(m = 1, \ldots, n)$. Assume that the selection $H_n^{(\lambda_t,\mu_t)}$ is given for any $(\lambda_t, \mu_t) \in U_n$. Then the generalized primal-relaxed dual algorithm reads:

```
1  Given $y_0 \in V$, and $\epsilon > 0$.
2  Given $y_n \in V$, solve (PP) for $y = y_n$ to obtain
   $x_n$ and Lagrange multiplies $(\lambda_n, \mu_n)$.
3  Solve (RD) to obtain $y_{n+1}$, where $U_n = \cup_{m=1}^{n}\{(\lambda_m, \mu_m)\}$.
4  Stop if $\nu^+(y_n) - \nu^-(U_n, H_n) < \epsilon$.
   Otherwise go to Step 2 with $n = n + 1$.
```

PRD Algorithm for (OP)

It is shown in [7] that the PRD algorithm converges to the global solutions of (OP) under some mild assumptions.

There are many possible choices for the mapping role H. In the literature the following results have been reported. In Geoffrion's original work in [4], $H_n^{(\lambda_m,\mu_m)}(x, y) = L(x, y, \lambda_m, \mu_m)$ $(1 \le m \le n)$. It is in general difficult to solve (RD) computationally with this choice of H_n. In the pioneer work [3], H_n takes the form of

$$\begin{aligned} &H_n^{(\lambda_m,\mu_m)}(x, y)\\ &\quad = L(x_m, y_m, \lambda_m, \mu_m)\\ &\qquad + \nabla_x L(x_m, y, \lambda_m, \mu_m)(x - x_m)\\ &\qquad + \nabla_y L(x_m, y_m, \lambda_m, \mu_m)(y - y_m)\\ &\qquad\qquad (m = 1, \ldots, n), \end{aligned}$$

where x_m, y_m, λ_m, μ_m are obtained from the previous iterations of the PRD algorithm and $\nabla_x L(x, y, \lambda, \mu)$ (or $\nabla_y L(x, y, \lambda, \mu)$) is the gradient of the Lagrange function L at x (or y) for a fixed (y, λ, μ) (or (x, λ, μ)). The resulting PRD algorithm has been referred to as *GOP algorithm* and has been widely used in various global optimization problems (see, e.g., [8] for a survey). Important progress has been made in developing efficient ways of solving (RD) for the GOP algorithm, see, also [8].

The GOP algorithm is only applicable to smooth optimization problems where the objective functions and the constraints are differentiable. Nonsmooth optimization problems occur in many important real applications. In [7] the GPRD approach is applied to a class of nonsmooth global optimization problems where

$$f = F + \max_{e\in E} F^e$$

and

$$g_i = G_i + \max_{e\in E} G_i^e, \quad i = 1, \ldots, k,$$

where $E = \{1, \ldots, d\}$, and the smooth C^1 functions F, F^e, $G = (G_1, \ldots, G_k)^\intercal$, $G^e = (G_1^e, \ldots, G_k^e)^\intercal$ satisfy all the conditions in (OP) for $e = 1, \ldots, d$. The resulting algorithm, referred to as *EGOP*, reads:

```
1 Given $y_0 \in V$, and $\epsilon > 0$.
2 Given $y_n \in V$, solve (PP) for $y = y_n$ to obtain $x_n$ and Lagrange multipliers $(\lambda_n, \mu_n)$.
3 Solve (RD) to obtain $y_{n+1}$, where $U_n = \cup_{m=1}^{n}\{(\lambda_m, \mu_m)\}$ and for any $(\lambda_m, \mu_m) \in U_n$,
    $$\begin{aligned} &H_n^{(\lambda_m,\mu_m)}(x, y)\\ &= LS(x_m, y_m, \lambda_m, \mu_m)\\ &+\nabla_x LS(x_m, y, \lambda_m, \mu_m)(x - x_m)\\ &+\nabla_y LS(x_m, y_m, \lambda_m, \mu_m)(y - y_m)\\ &+\max_{e\in E}(F^e(x_m, y_m)+\nabla_x F^e(x_m, y)(x-x_m)\\ &+\nabla_y F^e(x_m, y_m)(y - y_m))\\ &+\sum_{i=1}^{k}\mu_m^i \max_{e\in E}(G_i^e(x_m, y_m))\\ &+\nabla_x G_i^e(x_m, y)(x - x_m)\\ &+\nabla_y G_i^e(x_m, y_m)(y - y_m), \end{aligned}$$
    where the smooth part of the Lagrange is defined by $LS(x, y, \lambda, \mu) = F(x, y) + \lambda^\top h(x, y) + \mu^\top G(x, y)$.
4 Stop if $v^+(y_n) - v^-(U_n, H_n) < \epsilon$. Otherwise go to Step 2 with $n = n + 1$.
```

EGOP Algorithm for (OP)

This algorithm is identical with the GOP algorithm in the smooth case where $F^e = 0$, $G^e = 0$ for $e = 1, \ldots, d$. The EGOP covers a wider range of global optimization problems, and it is shown in [7] to be convergent under essentially the same assumptions which ensure the convergence of the GOP algorithm.

Penalty implementation of the PRD algorithm has also been considered in the literature to explore another way of coping with possible infeasible primal or relaxed dual subproblems in the algorithm. In [7], the EGOP algorithm is applied to the following two penalty problems:

$$(NPOP)_\rho \quad \min_{(x,y)\in X\times Y} P(x, y),$$

where

$$P(x, y) = f(x, y) + \rho\sum_{j=1}^{k}\max\left(0, g_j(x, y)\right) + \rho\sum_{j=1}^{p}\left|h_j(x, y)\right|, \quad \rho > 0\,.$$

and

$$(SPOP)_M \quad \min_{(x,y)\in X\times Y} P(x, y),$$

where

$$P(x, y) = f(x, y) + M\sum_{j=1}^{k}\max\left(0, g_j(x, y)\right)^2 + M\sum_{j=1}^{p}\left|h_j(x, y)\right|^2, \quad M > 0.$$

The convergence of the two penalty implementations of EGOP algorithm is established in [7], where it is shown that the sequences $\{(x_n, y_n)\}$ generated by the EGOP algorithm for the penalty problems $(\mathrm{NPOP})_\rho$ and $(\mathrm{SPOP})_M$ converge to the solutions of the (OP) when ρ is large enough or M tends to infinite.

See also

- ► αBB Algorithm
- ► Global Optimization in Phase and Chemical Reaction Equilibrium

References

1. Avriel M (1976) Nonlinear programming: Analysis and methods. Prentice-Hall, Englewood Cliffs, NJ
2. Benders JF (1962) Partitioning procedures for solving mixed-variables programing problems. Numerische Math 4:238–252
3. Floudas CA, Visweswaran V (1990) A global optimization algorithm (GOP) for certain class of nonconvex NLPs -I. Theory. Comput Chem Eng 14:1397–1417
4. Geoffrion AM (1972) Generalized Benders decomposition. J Optim Th Appl 10:237–260
5. Liu WB, Floudas CA (1993) A remark on the GOP algorithm for global optimization. J Global Optim 3:519–521
6. Liu WB, Floudas CA (1995) Convergence of the (GOP) algorithm for a large class of smooth optimization problems. J Global Optim 6:607–611
7. Liu WB, Floudas CA (1996) Generalized primal-relaxed dual approach for global optimization. J Optim Th Appl 14:416–434
8. Visweswaran V, Floudas CA (1996) New formulations and branching strategies for the GOP algorithm. In: Grossmann IE (ed) Global optimization in chemical engineering. Kluwer, Dordrecht, pp 75–100
9. Wolsey LA (1981) A resource decomposition algorithm for general mathematical programs. Math Program Stud 90:244–257

Generalized Semi-infinite Programming: Optimality Conditions

OLIVER STEIN
School of Economics and Business Engineering,
University of Karlsruhe, Karlsruhe, Germany

MSC2000: 90C34, 90C46, 90C31

Article Outline

Introduction

In generalized semi-infinite optimization problems, a finite-dimensional decision variable x is subject to infinitely many inequality constraints, that is, in

$$GSIP: \quad \text{minimize} \quad f(x) \quad \text{subject to} \quad x \in M\,,$$

the feasible set is described by

$$M = \{x \in \mathbb{R}^n | g(x, y) \leq 0 \text{ for all } y \in Y(x)\}\,,$$

with the index set

$$Y(x) = \{y \in \mathbb{R}^m | v_\ell(x, y) \leq 0, \ell \in L\}\,.$$

All defining functions $f, g, v_\ell, \ell \in L = \{1, \dots, s\}$ are assumed to be real valued and at least continuous on their respective domains. Moreover, the set-valued mapping $Y : \mathbb{R}^n \rightrightarrows \mathbb{R}^m$ is assumed to be locally bounded, that is, for each $\bar{x} \in \mathbb{R}^n$ there exists a neighborhood U of $\bar{x}$ such that $\bigcup_{x \in U} Y(x)$ is bounded in $\mathbb{R}^m$.

In applications such as design centering, robust optimization, and (reverse) Chebyshev approximation ([13,32]), often finitely many semi-infinite constraints $g_i(x, y) \leq 0,\ y \in Y_i(x),\ i \in I$, describe the feasible set M of *GSIP*, along with finitely many equality constraints in the definitions of M and $Y(x)$. In order to avoid technicalities this article focuses on the basic case of a single semi-infinite constraint (see [13,32] for more general formulations).

As opposed to a standard semi-infinite optimization problem, the possibly infinite index set $Y(x)$ of the semi-infinite inequality constraint is allowed to vary with x in a *GSIP*. For surveys and detailed studies about *standard* semi-infinite optimization see [10,15,25]. In contrast to standard semi-infinite programs, the feasible set of *GSIP* is not necessarily a closed set, and it might possess a stable disjunctive structure ([32]).

Powerful optimality conditions are based on a thorough analysis of these topological structures. This article mainly deals with first-order optimality conditions and will, thus, begin with a discussion of different first-order properties of the feasible set.

Definitions

The key to understanding the topological features in the feasible set of *GSIP* lies in the bilevel structure of semi-infinite programming ([27,32]). In fact, it is not hard to see that the semi-infinite constraint in *GSIP* is equivalent to

$$\varphi(x) := \max_{y \in Y(x)} g(x, y) \leq 0\,,$$

which means that the feasible set M of *GSIP* is the lower-level set of some optimal value function:

$$M = \{x \in \mathbb{R}^n | \varphi(x) \leq 0\}\,. \tag{1}$$

The usual convention "$\max_\emptyset = -\infty$" is consistent here, as an empty index set $Y(x)$ corresponds, loosely speaking, to "the absence of restrictions" at x and, hence, to the feasibility of x.

The function ϕ is the optimal value function of the so-called *lower-level problem*

$$Q(x): \quad \max_{y \in \mathbb{R}^m} g(x, y) \quad \text{subject to} \quad v_\ell(x, y) \leq 0,\quad \ell \in L.$$

In contrast to the upper-level problem, which consists in minimizing f over M, in the lower-level problem x plays the role of an n-dimensional parameter, and y is the decision variable. The main computational problem in semi-infinite programming is that the lower-level problem has to be solved to global optimality, even if, for example, only a stationary point of the upper-level problem is sought.

Topological Properties

The alternative description of the feasible set in (1) shows that the topological properties of M are determined by the continuity properties of ϕ, whereas first- and second-order optimality conditions will rely on the first- and second-order properties of ϕ. The properties of optimal value functions have been studied extensively in parametric optimization ([2]; for a brief introduction see [32]).

The optimal value function ϕ can be shown to be at least upper semicontinuous, so that points $x \in \mathbb{R}^n$ with $\varphi(x) < 0$ belong to the topological interior of M. On the other hand, for investigations of the local structure of M or of local optimality conditions one is only interested in feasible points from the boundary ∂M of M. Hence it suffices to consider the zeros of ϕ, that is, points $x \in \mathbb{R}^n$ for which $Q(x)$ has maximal value $\varphi(x) = 0$. We denote the globally maximal points of $Q(x)$ for arbitrary $x \in \mathbb{R}^n$ by

$$Y_\star(x) = \{y \in Y(x) | g(x, y) = \varphi(x)\}$$

and for the special case of $x \in M \cap \partial M$ by

$$Y_0(x) = \{y \in Y(x) | g(x, y) = 0\} .$$

The set $Y_0(x)$ is also called the upper-level *active index set* of *GSIP*.

Note that M is closed if for all $x \in \mathbb{R}^n$ the index set $Y(x)$ is nonempty and the *Mangasarian–Fromovitz constraint qualification (MFCQ)* holds at some element of $Y_0(x)$ ([13,32]). If M is not closed, there may exist infeasible boundary points $x \in \partial M$, that is, boundary points with $\varphi(x) > 0$.

The Reduction Ansatz

For theoretical as well as numerical purposes it is of crucial importance to keep track of the elements of $Y_\star(x)$ for varying x. These points solve the lower-level problem so that for functions g and $v_\ell, \ell \in L$, which are continuously differentiable with respect to y, they satisfy the first-order necessary optimality condition of Karush–Kuhn–Tucker: let

$$\mathcal{L}(x, y, \gamma) = g(x, y) - \gamma^\top v(x, y) ,$$

denote the Lagrangian of $Q(x)$ with multiplier vector $\gamma \in \mathbb{R}^s$. Then for $\bar{x} \in M$ and each $\bar{y} \in Y_\star(\bar{x})$ such that MFCQ holds at $\bar{y}$ in $Q(\bar{x})$, there exist multipliers $\bar{\gamma} \geq 0$ with $D_y\mathcal{L}(\bar{x}, \bar{y}, \bar{\gamma}) = 0$ and $\bar{\gamma}_\ell \cdot v_\ell(\bar{x}, \bar{y}) = 0$, $\ell \in L$. Here $D_y\mathcal{L}$ denotes the gradient of $\mathcal{L}$ with respect to y as a row vector. Note that the multiplier vector $\bar{\gamma}$ is uniquely determined if instead of MFCQ the stronger *linear independence constraint qualification (LICQ)* holds at $\bar{y}$.

Keeping track of the elements of $Y_\star(x)$ can now be achieved, for example, by means of the implicit function theorem, if the functions g and $v_\ell, \ell \in L$, are C^2 with respect to y. For $\bar{x} \in M$ a local maximizer $\bar{y}$ of $Q(\bar{x})$ is called *nondegenerate* in the sense of Jongen/Jonker/Twilt ([19]), if LICQ, strict complementary slackness and a second-order sufficiency condition are satisfied. The *Reduction Ansatz* ([14,16,35]) is said to hold at $\bar{x} \in M$ if all global maximizers of $Q(\bar{x})$ are nondegenerate. The set $Y_\star(\bar{x})$ can then only contain finitely many points, say, $Y_\star(\bar{x}) = \{\bar{y}^1, \dots, \bar{y}^p\}$ with $p \in \mathbb{N}$. By a result from [8] the local variation of these points with x can be described by the implicit function theorem.

In fact, for x locally around $\bar{x}$ there exist continuously differentiable functions $y^i(x)$, $1 \leq i \leq p$, with $y^i(\bar{x}) = \bar{y}^i$ such that $y^i(x)$ is the locally unique local maximizer of $Q(x)$ around $\bar{y}^i$. It turns out that the functions $\varphi_i(x) := g(x, y^i(x))$ are even C^2 in a neighborhood of $\bar{x}$. Their gradients are

$$D\varphi_i(\bar{x}) = D_x\mathcal{L}(\bar{x}, \bar{y}^i, \bar{\gamma}^i) , \tag{2}$$

where $\bar{\gamma}^i$ is the uniquely determined multiplier vector corresponding to $\bar{y}^i$.

A major consequence of the Reduction Ansatz is the so-called Reduction Lemma ([16]): if the Reduction Ansatz holds at $\bar{x}$, then for all x from a neighborhood U of $\bar{x}$ one has

$$\varphi(x) = \max_{1 \leq i \leq p} \varphi_i(x) .$$

In view of (1) this means that locally around a feasible boundary point $\bar{x} \in M \cap \partial M$, the feasible set M can be described by finitely many C^2-constraints, that is, *GSIP* locally looks like a smooth finite optimization problem:

$$M \cap U = \{x \in U \mid \varphi_i(x) \leq 0, i = 1, \dots, p\} . \quad (3)$$

In particular, locally around $\bar{x}$ set M is closed, and it does not possess a stable disjunctive structure at $\bar{x}$.

First-Order Properties of the Feasible Set

Since the Reduction Ansatz cannot be expected to hold generically *everywhere* in $M \cap \partial M$, the first-order structure of M is also studied under considerably weaker assumptions. For the first-order approximation of M one defines the *contingent cone* $\Gamma^\star(\bar{x}, M)$ to M at $\bar{x}$ as follows: $\bar{d} \in \Gamma^\star(\bar{x}, M)$ if and only if there exist sequences of scalars $(t^\nu)_{\nu\in\mathbb{N}}$ and of vectors $(d^\nu)_{\nu\in\mathbb{N}}$ such that

$$t^\nu \searrow 0, d^\nu \to \bar{d}(\nu \to \infty) \quad \text{and} \quad \bar{x} + t^\nu d^\nu \in M \quad \text{for all} \quad \nu \in \mathbb{N}.$$

The contingent cone is a closed cone, not necessarily convex, containing first-order information about M. In view of (1) the contingent cone to M at $\bar{x}$ should be related to a level set of a first order approximation of ϕ at $\bar{x}$. Unfortunately, the differentiability properties of ϕ can be very weak, so that *lower and upper directional derivatives* of ϕ at $\bar{x}$ in direction $\bar{d}$ in the Hadamard sense ([4]) come into play:

$$\varphi'_-(\bar{x}, \bar{d}) = \liminf_{t\searrow 0, d\to\bar{d}} \frac{\varphi(\bar{x}+td) - \varphi(\bar{x})}{t}$$

and

$$\varphi'_+(\bar{x}, \bar{d}) = \limsup_{t\searrow 0, d\to\bar{d}} \frac{\varphi(\bar{x}+td) - \varphi(\bar{x})}{t} .$$

ϕ is called *directionally differentiable* at $\bar{x}$ (in the Hadamard sense) if for each direction $d \neq 0$ one has $\varphi'_-(\bar{x}, d) = \varphi'_+(\bar{x}, d) =: \varphi'(\bar{x}, d)$. The *outer linearization cone* of M at $\bar{x}$ can now be defined as

$$L^\star(\bar{x}, M) = \{d \in \mathbb{R}^n | \varphi'_-(\bar{x}, d) \leq 0\}$$

and the *inner linearization cone* by

$$L(\bar{x}, M) = \{d \in \mathbb{R}^n | \varphi'_+(\bar{x}, d) < 0\} .$$

For $\bar{x} \in \partial M \cap M$ the chain of inclusions

$$L(\bar{x}, M) \subset \Gamma^\star(\bar{x}, M) \subset L^\star(\bar{x}, M) \quad (4)$$

holds ([22,33]). A good first-order description of M around $\bar{x}$ can thus be obtained if the linearization cones $L(\bar{x}, M)$ and $L^\star(\bar{x}, M)$ do not differ too much from each other.

For example, in standard semi-infinite programming the index set mapping $Y(x) \equiv Y$ is constant, and the theorem of Danskin ([6]) then says that ϕ is directionally differentiable with

$$\varphi'(\bar{x}, d) = \max_{y\in Y_0(\bar{x})} D_x g(\bar{x}, y) d$$

for all $d \in \mathbb{R}^n$. The linearization cones

$$L(\bar{x}, M) = \bigcap_{y\in Y_0(\bar{x})} \{d \in \mathbb{R}^n | D_x g(\bar{x}, y) d < 0\}$$

and

$$L^\star(\bar{x}, M) = \bigcap_{y\in Y_0(\bar{x})} \{d \in \mathbb{R}^n | D_x g(\bar{x}, y) d \leq 0\}$$

thus differ only by the strictness of inequalities, and they do not possess a disjunctive structure.

If in *GSIP* the Reduction Ansatz holds at $\bar{x}$, using (2) it is not hard to see that ϕ is directionally differentiable with

$$\varphi'(\bar{x}, d) = \max_{1\leq i\leq p} D_x \mathcal{L}(\bar{x}, \bar{y}^i, \bar{\gamma}^i) d$$

for all $d \in \mathbb{R}^n$. The linearization cones

$$L(\bar{x}, M) = \bigcap_{i=1}^{p} \{d \in \mathbb{R}^n | D_x \mathcal{L}(\bar{x}, \bar{y}^i, \bar{\gamma}^i) d < 0\}$$

and

$$L^\star(\bar{x}, M) = \bigcap_{i=1}^{p} \{d \in \mathbb{R}^n | D_x \mathcal{L}(\bar{x}, \bar{y}^i, \bar{\gamma}^i) d \leq 0\}$$

again differ only by the strictness of inequalities.

Under weaker assumptions than the Reduction Ansatz the situation in *GSIP* becomes more involved since ϕ does not even have to be directionally differentiable. The following estimates for the upper and lower directional derivatives from [9,23] are known to be tight: for $\bar{x} \in \partial M \cap M$ such that MFCQ is satisfied at each $y \in Y_0(\bar{x})$ one has for each $d \in \mathbb{R}^n$

$$\sup_{y \in Y_0(\bar{x})} \min_{\gamma \in KKT(\bar{x},y)} D_x\mathcal{L}(\bar{x}, y, \gamma)\, d \le \varphi'_-(\bar{x}, d)$$
$$\le \varphi'_+(\bar{x}, d) \le \max_{y \in Y_0(\bar{x})} \max_{\gamma \in KKT(\bar{x},y)} D_x\mathcal{L}(\bar{x}, y, \gamma) d\,.$$

Here

$$KKT(x, y) = \{\gamma \in \mathbb{R}^s | \gamma \ge 0, D_y\mathcal{L}(x, y, \gamma) = 0, \\ \gamma_\ell \cdot v_\ell(x, y) = 0, \ell \in L\}$$

denotes the set of Karush–Kuhn–Tucker multipliers at y in $Q(x)$.

At least this yields estimates for the linearization cones:

$$\bigcap_{y \in Y_0(\bar{x})} \bigcap_{\gamma \in KKT(\bar{x},y)} \{d \in \mathbb{R}^n | D_x\mathcal{L}(\bar{x}, y, \gamma)\, d \;<\; 0\}$$
$$\subset L(\bar{x}, M) \subset \Gamma^\star(\bar{x}, M) \subset L^\star(\bar{x}, M)$$
$$\subset \bigcap_{y \in Y_0(\bar{x})} \bigcup_{\gamma \in KKT(\bar{x},y)} \{d \in \mathbb{R}^n | D_x\mathcal{L}(\bar{x}, y, \gamma) d \le 0\}\,.$$

However, the estimate for the inner linearization cone is rather poor in many situations in which the problem data are endowed with a special structure. In [31] analogous estimates are given without the assumption of MFCQ in $Y_0(\bar{x})$.

A disjunctive structure of $\Gamma^\star(\bar{x}, M)$ is intimately related to the nonuniqueness of the lower-level Karush–Kuhn–Tucker multipliers. This becomes clearer under the assumption that the lower-level problems $Q(x)$, $x \in U$, are convex for some neighborhood U of $\bar{x}$, and that $Y(\bar{x})$ possesses a Slater point. Due to results from [11,18,26] the multiplier set $KKT(\bar{x})$ then does not depend on $y \in Y_0(\bar{x})$, and ϕ is directionally differentiable at $\bar{x}$ with

$$\varphi'(\bar{x}, d) = \min_{\gamma \in KKT(\bar{x})} \max_{y \in Y_0(\bar{x})} D_x\mathcal{L}(\bar{x}, y, \gamma)\, d$$

for all $d \in \mathbb{R}^n$. This yields

$$L(\bar{x}, M) = \bigcup_{\gamma \in KKT(\bar{x})} \bigcap_{y \in Y_0(\bar{x})} \{d \in \mathbb{R}^n |\, D_x\mathcal{L}(\bar{x}, y, \gamma)\, d \;<\; 0\}$$

and

$$L^\star(\bar{x}, M) = \bigcup_{\gamma \in KKT(\bar{x})} \bigcap_{y \in Y_0(\bar{x})} \{d \in \mathbb{R}^n | \\ D_x\mathcal{L}(\bar{x}, y, \gamma)\, d \;\le\; 0\}\,.$$

Now both the inner and outer linearization cones possess a disjunctive structure, and they only differ by the strictness of inequalities. Moreover it becomes obvious that the occurrence of a stable disjunctive structure in *GSIP* is caused by nonunique lower-level Karush–Kuhn–Tucker multipliers. For more details on lower-level problems with a special structure see [27,29,32].

Constraint Qualifications

In what follows let the functions f, g, and v_ℓ , $\ell \in L$, be continuously differentiable. It is well known ([3]) that at a local minimizer $\bar{x}$ of f on M the following primal first-order necessary optimality condition holds:

$$\{d \in \mathbb{R}^n | Df(\bar{x})d < 0\} \cap \Gamma^\star(\bar{x}, M) = \emptyset\,. \tag{5}$$

To obtain a more explicit condition from (5) one needs an explicit description of $\Gamma^\star(\bar{x}, M)$. A good candidate would be the outer linearization cone $L^\star(\bar{x}, M)$, which contains the contingent cone by (4). Even in finite optimization simple examples show, however, that $\Gamma^\star(\bar{x}, M)$ can be a proper subset of $L^\star(\bar{x}, M)$. In this case one cannot replace the contingent cone in (5) by the outer linearization cone.

On the other hand, in view of (4) it is always possible to replace the contingent cone in (5) by the inner linearization cone. However, the resulting optimality condition may be trivially satisfied since $L(\bar{x}, M)$ can be void itself.

These observations give rise to the following definitions.

Definition 1 The *extended Mangasarian–Fromovitz constraint qualification* (EMFCQ) holds at $\bar{x} \in M$ if $L(\bar{x}, M) \neq \emptyset$, and the *extended Abadie constraint qualification* (EACQ) holds at $\bar{x} \in M$ if $\Gamma^\star(\bar{x}, M) = L^\star(\bar{x}, M)$.

Note that EMFCQ coincides with MFCQ for finite differentiable optimization problems ([24]). Furthermore, it is obvious that EACQ coincides with the Abadie constraint qualification (ACQ, [1]) for finite differentiable optimization problems. Whereas in finite optimization MFCQ is stronger than ACQ, for *GSIP* this is not necessarily the case as an example in [33] shows (see, however, [31]). For extensions of the Karush–Kuhn–Tucker constraint qualification to *GSIP* see [12]. Explicit formulations of EMFCQ under different structural assumptions on the lower-level problem $Q(\bar{x})$ can easily be obtained from the descriptions of $L(\bar{x}, M)$ given above.

Formulation

An important difference to finite or standard semi-infinite programming is that, for *GSIP*, there does not exist a single first-order necessary optimality condition, but the explicit formulation of the condition heavily depends on the structure of the lower-level problem. In fact, from the abstract primal first-order optimality condition (5) one can derive explicit dual conditions by replacing the contingent cone by an appropriate linearization cone and then cast the resulting conditions on certain infinite inequality systems in a dual formulation by means of theorems of the alternative, like, for example, the lemma of Gordan ([5,17]).

First-Order Optimality Conditions. In what follows, such optimality conditions are given for the structures discussed above. Recall that optimality conditions are trivial at interior points of M.

Theorem 1 **([16])** *Let $\bar{x} \in \partial M \cap M$ be a local minimizer of GSIP, at which the Reduction Ansatz holds. Moreover, let there exist a $d_0 \in \mathbb{R}^n$ such that*

$$D_x \mathcal{L}(\bar{x}, \bar{y}^i, \bar{\gamma}^i)\, d_0 < 0 \quad \textit{for all} \quad 1 \le i \le p\,,$$

(i. e. EMFCQ holds at $\bar{x}$). Then there exist multipliers $\lambda_i \ge 0$, $i = 1, \dots, p$, with $|\{1 \le i \le p | \lambda_i > 0\}| \le n$ such that

$$Df(\bar{x}) + \sum_{i=1}^{p} \lambda_i\, D_x \mathcal{L}(\bar{x}, \bar{y}^i, \bar{\gamma}^i) = 0\,.$$

Theorem 2 **([29,32])** *Let $\bar{x} \in \partial M \cap M$ be a local minimizer of GSIP, let the lower-level problems $Q(x)$, $x \in U$, be convex for some neighborhood U of $\bar{x}$, and let $Y(\bar{x})$ possess a Slater point. Then for each choice $\gamma \in KKT(\bar{x})$ such that there exists a d_0 with*

$$D_x \mathcal{L}(\bar{x}, y, \gamma)\, d_0 < 0 \quad \textit{for all} \quad y \in Y_0(\bar{x})\,, \tag{6}$$

there exist $y^i \in Y_0(\bar{x})$ and multipliers $\lambda_i \ge 0$, $i = 1, \dots, p$, with $|\{1 \le i \le p | \lambda_i > 0\}| \le n$, such that

$$Df(\bar{x}) + \sum_{i=1}^{p} \lambda_i\, D_x \mathcal{L}(\bar{x}, y^i, \gamma) = 0\,.$$

If EMFCQ holds at $\bar{x}$, then at least one such choice γ exists.

Theorem 3 **([27,32])** *Let $\bar{x} \in \partial M \cap M$ be a local minimizer of GSIP, and let MFCQ hold at all $y \in Y_0(\bar{x})$. Moreover, let there exist a $d_0 \in \mathbb{R}^n$ such that*

$$D_x \mathcal{L}(\bar{x}, y, \gamma)\, d_0 < 0 \qquad \textit{for all} \quad \gamma \in KKT(\bar{x}, y)\,, \quad y \in Y_0(\bar{x})\,,$$

(which is sufficient for EMFCQ at $\bar{x}$). Then there exist $y^i \in Y_0(\bar{x})$, $\gamma^i \in KKT(\bar{x}, y^i)$, and multipliers $\lambda_i \ge 0$, $i = 1, \dots, p$, with $|\{1 \le i \le p | \lambda_i > 0\}| \le n$, such that

$$Df(\bar{x}) + \sum_{i=1}^{p} \lambda_i\, D_x \mathcal{L}(\bar{x}, y^i, \gamma^i) = 0\,.$$

Note that, under the convexity assumption on the lower-level problem, Theorem 2 provides a whole family of optimality conditions (parametrized by $\gamma \in KKT(\bar{x})$) and, thus, takes a possible disjunctive structure of M at $\bar{x}$ into account. On the other hand, in the absence of a nice lower-level structure, Theorem 3 yields a much weaker condition (which cannot be strengthened without further assumptions, as examples show).

First-order necessary optimality conditions for *GSIP* have been derived under several other structural assumptions and other theoretical approaches as well. In fact, without the assumption of EMFCQ, Fritz John-type optimality conditions can be derived ([32]), and there also exist optimality conditions without the assumption of *any* regularity condition, either in the upper- *or in the lower-level problem* ([20,31,32]). Conditions under other constraint qualifications are investigated in [12]. Furthermore, other theoretical approaches to optimality conditions are the linearization approach from [27] and conditions based on quasidifferentiable calculus ([7,27,30]). First-order *sufficient* optimality conditions for *GSIP* are examined in [32,34].

Second-Order Optimality Conditions

Second-order necessary and sufficient optimality conditions can be obtained in a straightforward manner under the Reduction Ansatz. One must simply write down the corresponding condition for the reduced finite optimization problem with the feasible set from

(3). Unfortunately, the Hessians of the optimal value functions $\varphi_i(x) = g(x, y^i(x)), 1 \leq i \leq p$, have a more complicated structure than the gradients from (2), due to the appearance of so-called *shift terms*. In fact, one has

$$D_x^2\varphi_i(\bar{x}) = D_x^2\mathcal{L}(\bar{x}, \bar{y}^i, \bar{\gamma}^i) - \begin{pmatrix} D^2_{yx}\mathcal{L}_i(\bar{x}, \bar{y}^i, \bar{\gamma}^i) \\ -D_x v_{L_0^i}(\bar{x}, \bar{y}^i) \end{pmatrix}^\top \cdot \begin{pmatrix} D_y^2\mathcal{L}_i(\bar{x}, \bar{y}^i, \bar{\gamma}^i) & -D_y^\top v_{L_0^i}(\bar{x}, \bar{y}^i) \\ -D_y v_{L_0^i}(\bar{x}, \bar{y}^i) & 0 \end{pmatrix}^{-1} \cdot \begin{pmatrix} D^2_{yx}\mathcal{L}_i(\bar{x}, \bar{y}^i, \bar{\gamma}^i) \\ -D_x v_{L_0^i}(\bar{x}, \bar{y}^i) \end{pmatrix},$$

where $D_x v_{L_0^i}$ stands for the matrix with rows $D_x v_\ell$, $\ell \in L_0^i := \{\ell \in L | \, v_\ell(\bar{x}, \bar{y}^i) = 0\}$.

Second-order conditions are also known under weaker assumptions, for example without the strict complementary slackness assumption of the Reduction Ansatz ([16]), and in connection with second-order epiregularity ([13,28], see also [4]). A second-order stability analysis for *GSIP* is given in [21].

Conclusions

First- and second-order optimality conditions are not only of theoretical importance, but also of high significance for the design of efficient numerical methods for *GSIP*. A review of such methods, including methods of feasible directions, KKT methods, and discretization methods, is given in [13].

See also

- ▸ Bilevel Optimization: Feasibility Test and Flexibility Index
- ▸ Parametric Optimization: Embeddings, Path Following and Singularities
- ▸ Second Order Constraint Qualifications

References

1. Abadie JM (1967) On the Kuhn–Tucker theorem. In: Abadie J (ed) Nonlinear Programming. Wiley, New York, pp 21–36
2. Bank B, Guddat J, Klatte D, Kummer B, Tammer K (1983) Non-linear Parametric Optimization. Birkhäuser, Basel
3. Bazaraa MS, Sherali HD, Shetty CM (1993) Nonlinear Programming. Theory and Algorithms. Wiley, New York
4. Bonnans JF, Shapiro A (2000) Perturbation Analysis of Optimization Problems. Springer, New York
5. Cheney EW (1966) Introduction to Approximation Theory. McGraw-Hill, New York
6. Danskin JM (1967) The Theory of Max-Min and Its Applications to Weapons Allocation Problems. Springer, New York
7. Demyanov VF, Vasilev LV (1985) Nondifferentiable Optimization, Optimization Software Inc. Publications Division, New York
8. Fiacco AV, McCormick GP (1968) Nonlinear Programming: Sequential Unconstrained Minimization Techniques. Wiley, New York
9. Gauvin J, Dubeau F (1982) Differential properties of the marginal function in mathematical programming. Math Programm Study 19:101–119
10. Goberna MA, López MA (1998) Linear Semi-infinite Optimization. Wiley, Chichester
11. Gol'stein EG (1972) Theory of Convex Programming, Translations of Mathematical Monographs, vol 36. American Mathematical Society, Providence, RI
12. Guerra Vázquez F, Rückmann J-J (2005) Extensions of the Kuhn–Tucker constraint qualification to generalized semi-infinite programming. SIAM J Optim 15(3):926–937
13. Guerra Vázquez F, Rückmann J-J, Stein O (2007) Still G Generalized semi-infinite programming: a tutorial. J Comput Appl Math, DOI: 10.1016/j.cam.2007.02.012
14. Hettich R, Jongen HT (1978) Semi-infinite programming: conditions of optimality and applications. In: Stoer J (ed) Optimization Techniques, Part 2, Lecture Notes in Control and Information Sciences, vol 7. Springer, Berlin, pp 1–11
15. Hettich R, Kortanek KO (1993) Semi-infinite programming: theory, methods, and applications. SIAM Rev 35:380–429
16. Hettich R, Still G (1995) Second order optimality conditions for generalized semi-infinite programming problems. Optim 34:195–211
17. Hettich R, Zencke P (1982) Numerische Methoden der Approximation und semi-infiniten Optimierung. Teubner, Stuttgart
18. Hogan WW (1973) Directional derivatives for extremal value functions with applications to the completely convex case. Oper Res 21:188–209
19. Jongen HT, Jonker P, Twilt F (1986) Critical sets in parametric optimization. Math Programm 34:333–353
20. Jongen HT, Rückmann J-J, Stein O (1998) Generalized semi-infinite optimization: a first order optimality condition and examples. Math Programm 83:145–158
21. Klatte D (1994) Stable local minimizers in semi-infinite optimization: regularity and second-order conditions. J Comp Appl Math 56:137–157
22. Laurent P-J (1972) Approximation et Optimisation. Hermann, Paris
23. Lempio F, Maurer H (1980) Differential stability in infinite-dimensional nonlinear programming. Appl Math Optim 6:139–152
24. Mangasarian OL, Fromovitz S (1967) The Fritz John necessary optimality conditions in the presence of equality and inequality constraints. J Math Anal Appl 17:37–47

25. Polak E (1997) Optimization. Algorithms and Consistent Approximations. Springer, Berlin
26. Rockafellar RT (1984) Directional differentiability of the optimal value function in a nonlinear programming problem. Math Program Stud 21:213–226
27. Rückmann J-J, Shapiro A (1999) First-order optimality conditions in generalized semi-infinite programming. J Optim Theory Appl 101:677–691
28. Rückmann J-J, Shapiro A (2001) Second-order optimality conditions in generalized semi-infinite programming. Set-Valued Anal 9:169–186
29. Rückmann J-J, Stein O (2001) On convex lower level problems in generalized semi-infinite optimization. In: Goberna MA, López MA (eds) Semi-infinite Programming – Recent Advances. Kluwer, Dordrecht, pp 121–134
30. Shapiro A (1984) On optimality conditions in quasidifferentiable optimization. SIAM J Control Optim 22:610–617
31. Stein O (2001) First order optimality conditions for degenerate index sets in generalized semi-infinite programming. Math Oper Res 26:565–582
32. Stein O (2003) Bi-level Strategies in Semi-infinite Programming. Kluwer, Boston
33. Stein O (2004) On constraint qualifications in non-smooth optimization. J Optim Theory Appl 121:647–671
34. Stein O, Still G (2000) On optimality conditions for generalized semi-infinite programming problems. J Optim Theory Appl 104:443–458
35. Wetterling W (1970) Definitheitsbedingungen für relative Extrema bei Optimierungs- und Approximationsaufgaben. Numerische Mathematik 15:122–136

Generalized Total Least Squares

CHENGXIAN XU
Xian Jiaotong University, Xian, China

MSC2000: 90C30

Article Outline

Keywords

Generalized nonlinear least squares; Gauss–Newton method; Separable optimization; separated Newton method

One important application of *nonlinear least squares* concerns with *data fitting* or *parameter estimations*. In ordinary least squares for data fitting, it is assumed that the errors in independent variables are either zero or negligible. Although there are situations in which errors in independent variables are zero or negligible, there exist many cases such as experiments and observations where this isnot so and use of the ordinary least squares may lead to bias in the estimated values of parameter vector and variance values [8]. *Generalized total least squares* problems are formulated from data fitting if errors in all variables are taken into account. Suppose that we have chosen a model function $y = \phi(x, t)$ to fit a set of data $y_1, \dots, y_m$ sampled at m points $t_1, \dots, t_m$, where $x \in \mathbf{R}^n$ is an adjustable parameter vector. The generalized total least squares problem concerning with this data fitting determines an optimal value of x and τ such that the function

$$\begin{aligned} f(x,\tau) &= \frac{1}{2}\sum_{j=1}^{m}[w_j(\phi(x,\tau_j) - y_j)^2 + v_j(\tau_j - t_j)^2] \\ &= \frac{1}{2}[r^\top Wr + e^\top Ve] \end{aligned}$$

is minimized, where $(\phi(x, \tau_j), \tau_j), j = 1, \dots, m$, are true but unknown values of pair (y, t), $W = \operatorname{diag}(w_1, \dots, w_m)$, $V = \operatorname{diag}(v_1, \dots, v_m)$, $w_j \geq 0$, $v_j \geq 0$, $j = 1, \dots, m$, are weighting factors, r and e are two m-vectors with components $r_j = \phi(x, \tau_j)- y_j$, $e_j = \tau_j- t_j$, $j = 1, \dots, m$, respectively.

Generalized total least squares problems can be solved by directly applying any method for ordinary nonlinear least squares or general minimization problems. Since these methods minimize the objective function $f(x, \tau)$ with respect to $(n+m)$ variables x and τ, and do not allow for the use of the special structureof the function, direct use of these methods will not be efficient. Assuming that the functions $r_j(x, \tau)$, $j = 1, \dots, m$, hence the function $f(x, \tau)$ is twice continuously differentiable, the first and the second order derivatives of $f(x, \tau)$ are defined by

$$\nabla f = \begin{bmatrix} \nabla_x f \\ \nabla_\tau f \end{bmatrix} = \begin{bmatrix} AWr \\ Ve + DWr \end{bmatrix},$$

$$\nabla^2 f = \begin{pmatrix} \nabla^2_{xx} f & \nabla^2_{x\tau} f \\ \nabla^2_{\tau x} f & \nabla^2_{\tau\tau} f \end{pmatrix},$$

where

$$\nabla^2_{xx} f = AWA^\top + \sum_{j=1}^m w_j r_j \nabla^2_{xx} r_j,$$

$$\nabla^2_{x\tau} f = AWD + \sum_{j=1}^m w_j r_j \nabla^2_{x\tau} r_j,$$

$$\nabla^2_{\tau\tau} f = V + DWD + \sum_{j=1}^m w_j r_j \nabla^2_{\tau\tau} r_j,$$

$$A = \left[\nabla_x r_1 \cdots \nabla_x r_m\right],$$

$$D = \operatorname{diag}\left[\tfrac{\partial r_1}{\partial \tau_1} \cdots \tfrac{\partial r_m}{\partial \tau_m}\right].$$

The $(m\times m)$-matrix $\nabla^2_{\tau\tau} f$ is a diagonal matrix with diagonal elements

$$v_j + w_j \left(\frac{\partial r_j}{\partial \tau_j}\right)^2 + w_j r_j \frac{\partial^2 r_j}{\partial \tau_j^2}.$$

In developing algorithms for generalized total least squares, it is important to exploit the special structures of the function $f(x, \tau)$ and its derivatives, and in particular, the fact that variables x and τ can be treated separately. W.E. Demming [2], M. O'Neill, I.G. Sinclair and J. Smith [5], D.R. Powell and J.R. Macdonald [6] proposed *approximate Newton methods* for polynomial data fitting. These methods evaluate the second order derivatives $\nabla^2_{xx} f$ and $\nabla^2_{\tau\tau} f$ analytically or numerically, but ignore the mixed partial derivatives $\nabla^2_{x\tau} f$ and $\nabla^2_{\tau x} f$. When analytical derivatives are used, approximate Newton methods are not very efficient because of the unreasonable approximations. When derivatives are evaluated from difference quotient and compensations for ignoring mixed parts are made, the behavior of these methods is improved, because in this case the methods are equivalent to using one Newton step to separate problem variables and then the separated problem is solved using Newton method.

An optimization problem is *separable* if the optimization with respect to some of the variables is easier than with respect to others. Generalized total least squares problems are a kind of separable optimization problems. W.H. Southwell [7] uses the first order necessary condition to separate the vector x and the vector τ and then the separated problem is solvedusing Newton method. Gauss-Newton and quasi-Newton methods can also be used to solve the separated problems.

When Newton method is applied to solve a generalized total least squares problem, the solution of the Newton equation

$$\begin{bmatrix} \nabla^2_{xx} f & \nabla^2_{x\tau} f \\ \nabla^2_{\tau x} f & \nabla^2_{\tau\tau} f \end{bmatrix} \begin{bmatrix} \delta x \\ \delta \tau \end{bmatrix} = -\begin{bmatrix} \nabla_x f \\ \nabla_\tau f \end{bmatrix}$$

gives a correction $(\delta x, \delta\tau)$ to (x, τ), that is,

$$x_+ = x + \delta x, \qquad \tau_+ = \tau + \delta\tau,$$

where x_+, τ_+ denote the new iterate. When the fitting function $\phi(x, t)$ is a polynomial in the form

$$\phi(x, t) = \sum_{i=1}^n x_i p_i(t),$$

where $p_i(t)$, $i = 1, \dots, n$, are a set of *orthogonal polynomials*, then off-diagonal elements of the $(n \times n)$-matrix $\nabla^2_{xx} f$ are all zeros. Thus both the matrices $\nabla^2_{xx} f$ and $\nabla^2_{\tau\tau} f$ are diagonal. By assuming the elements of matrices $\nabla^2_{x\tau} f$ and $\nabla^2_{\tau x} f$ are negligible, approximate Newton methodsapproximate the Hessian matrix $\nabla^2 f$ by the simple diagonal matrix

$$\begin{bmatrix} \nabla^2_{xx} f & \\ & \nabla^2_{\tau\tau} f \end{bmatrix}.$$

Since $\nabla^2_{xx} f$ and $\nabla^2_{\tau\tau} f$ are diagonal, the solution δx and $\delta\tau$ can be easily given by

$$\delta x_i = -\frac{\sum_{j=1}^m w_j r_j p_i(\tau_j)}{\sum_{j=1}^m w_j p_i(\tau_j)^2}, \quad i = 1, \dots, n,$$

$$\delta \tau_j = -\frac{v_j e_j + w_j \frac{\partial \phi(x,\tau_j)}{\partial \tau_j} r_j}{v_j + w_j \left(\frac{\partial \phi(x,\tau_j)}{\partial \tau_j}\right)^2 + w_j r_j \frac{\partial^2 \phi(x,\tau_j)}{\partial \tau_j^2}},$$

$$j = 1, \dots, m.$$

Polynomials $p_i(t)$, $i = 1, \dots, n$, orthogonal over a set of points τ_j, $j = 1, \dots, m$, can be generated using the *recurrence relation*

$$p_1(t) = 1, \quad p_2(t) = t - \alpha_1,$$

$$p_i(t) = (t - \alpha_{i-1}) p_{i-1}(t) - \beta_{i-1} p_{i-2}(t),$$

$$i = 3, \dots, n,$$

where

$$\alpha_{i-1} = \frac{\sum_{j=1}^m w_j \tau_j p_{i-1}(\tau_j)^2}{\sum_{j=1}^m w_j p_{i-1}(\tau_j)^2},$$

$$\beta_{i-1} = \frac{\sum_{j=1}^m w_j \tau_j p_{i-1}(\tau_j) p_{i-2}(\tau_j)}{\sum_{j=1}^m w_j p_{i-2}(\tau_j)^2}.$$

Approximate Newton methods begin iteration from the initial point $x^{(1)} = 0$ and $\tau_j^{(1)} = t_j, j = 1, \ldots, m$. At each iteration, the polynomials $p_i(t)$, $i = 1, \ldots, n$, orthogonal over the set of points $\tau_j^{(k)}$, $j = 1, \ldots, m$, are first calculated from the recurrence relation, then iteration

$$x^{(k+1)} = x^{(k)} + \delta x^{(k)}, \quad \tau^{(k+1)} = \tau^{(k)} + \delta\tau^{(k)}$$

is implemented to generate a new iterate. The process is repeated until convergence is reached. If the resulting fitting polynomial is required to express in the form of power series

$$\phi(x, t) = \sum_{i=1}^{n} c_i t^{i-1} = \sum_{i=1}^{n} x_i p_i(t),$$

the coefficients c_i can be calculated from

$$c_i = \sum_{k=i}^{n} x_k \sigma_{i+1k+1}, \quad 1 \leq i \leq n,$$

where

$$\sigma_{ik} = \begin{cases} 1 & \text{if } i = k, \\ 0 & \text{if } i > k \text{ or } i, k < 2, \end{cases}$$
$$\sigma_{i+1k+1} = \sigma_{ik} - \alpha_{k-1}\sigma_{i+1,k} - \beta_{k-1}\sigma_{i+1k-1},$$
$$i < k.$$

Powell and Macdonald extended the method to more general case where $\phi(x, t)$ is a general nonlinear function of both the variables x and t. In this case, the $(n \times n)$-matrix $\nabla_{xx}^2 f$ is no longer diagonal, and the correction δx needs the solution of the equations

$$\nabla_{xx}^2 f \delta x = -\nabla_x f.$$

By taking account of the omitted parts of the mixed partial derivatives $\nabla_{x\tau}^2 f$ and $\nabla_{\tau x}^2 f$, they use 'unconventional formulas', rather than analytical derivatives or usual difference approximations, to approximate derivatives in $\nabla_x f$ and $\nabla{}_{xx}^{2} f$ so that the omission parts can be compensated to some degree. In fact, their approximate Newton method is equivalent, in some sense, to the separated Newton method.

Approximate Newton methods require evaluations of second order derivatives for problem functions. Ignoring all the second order terms in $\nabla_{xx}^2 f$, $\nabla_{x\tau}^2 f$, $\nabla_{\tau x}^2 f$ and $\nabla_{\tau\tau}^2 f$, an approximation to $\nabla^2 f$ is directly obtained from the first order derivatives of functions r_j and e_j, $j = 1, \ldots, m$. The iteration scheme $x^{(k+1)} = x^{(k)} + \delta\, x^{(k)}$, $\tau^{(k+1)} = \tau^{(k)} + \delta\tau^{(k)}$ with $\delta x^{(k)}$ and $\delta\tau^{(k)}$ given by

$$\begin{bmatrix} A_k W_k A_k^\top & A_k W D_k \\ D_k W A_k^\top & V + D_k W D_k \end{bmatrix} \begin{bmatrix} \delta x \\ \delta\tau \end{bmatrix} = -\begin{bmatrix} A_k W r^{(k)} \\ V e^{(k)} + D_k W r^{(k)} \end{bmatrix}$$

gives the *Gauss–Newton method* for generalized total least squares. Special structure of the system can be exploited to get savings in finding its solution. Define $P_k = V + D_k \mathrm{W} D_k$. From the bottom part of the system we have

$$\delta\tau = -P_k^{-1}\left[V e^{(k)} + D_k W r^{(k)} + D_k W A_k^\top \delta x\right].$$

Since P_k is a diagonal matrix, once δx is obtained, $\delta\tau$ can be directly obtained by substitutions. Substituting $\delta\tau$ into the top part of the system we obtain

$$\left[A_k W A_k^\top - A_k W D_k P_k^{-1} D_k W A_k^\top\right]\delta x$$
$$= A_k W^{\frac{1}{2}} b^{(k)}$$

with

$$b^{(k)} = W^{\frac{1}{2}}$$
$$\times\left[-r^{(k)} + D_k P_k^{-1}(V e^{(k)} + D_k W r^{(k)})\right].$$

This equation can be expressed as

$$A_k W^{\frac{1}{2}} U_k W^{\frac{1}{2}} A_k^\top \delta x = A_k W^{\frac{1}{2}} b^{(k)}$$

where $U_k = I - W^{1/2} D_k P_k^{-1} D_k W^{1/2}$ is a diagonal matrix with diagonal elements $v_j/[v_j + w_j(\partial r_j^{(k)}/\partial\tau_j)^2] > 0$, $j = 1, \ldots, m$. The solution $\delta x^{(k)}$ can be generated by first performing a *QR factorization* to the matrix $U_k^{1/2} W^{1/2} A_k^\top$

$$U_k^{\frac{1}{2}} W^{\frac{1}{2}} A_k^\top = Q \begin{bmatrix} R \\ 0 \end{bmatrix}$$

and then back substitutions in

$$R\delta x = Q U_k^{-\frac{1}{2}} b^{(k)}.$$

The Gauss–Newton method is locally convergent and convergence behavior depends upon the closeness of the Gauss–Newton matrix to the true Hessian matrix

$\nabla^2 f$ at the solution. In order to introduce global convergence for Gauss–Newton method, line search technique or trust region strategy can be used. Let

$$J_k = \begin{bmatrix} A_k W^{\frac{1}{2}} & 0 \\ D_k W^{\frac{1}{2}} & V^{\frac{1}{2}} \end{bmatrix}$$

Then

$$\begin{bmatrix} A_k W A_k^\top & A_k W D_k \\ D_k W A_k^\top & V + D_k W D_k \end{bmatrix} = J_k J_k^\top$$

and the Gauss–Newton matrix is at least positive semidefinite, often positive definite, $(\delta x^{(k)}, \delta\tau^{(k)})$ is a descent direction of $f(x, \tau)$ at $(x^{(k)}), \tau^{(k)})$. A line searchalong the direction determines a steplength α_k satisfying some descent conditions and the new iteration point is

$$x^{(k+1)} = x^{(k)} + \alpha_k \delta x^{(k)},$$
$$\tau^{(k+1)} = \tau^{(k)} + \alpha_k \delta\tau^{(k+1)}.$$

P.T. Boggs, R.H. Byrd and R.B. Schnabel [1] use trust region technique in their modification of Gauss–Newton method for generalized total least squares problems. The modification is a generalization of the Levenberg–Marquardt method, in which the trust region subproblem

$$\begin{cases} \min & q_k(\delta z) = \left\| J_k^\top \delta z + h^{(k)} \right\|^2 \\ \text{s.t.} & \|\delta z\| \le \Delta_k \end{cases}$$

is solved, where $\mathit{\Delta}_k$ is the trust region radius,

$$z = \begin{pmatrix} x \\ \tau \end{pmatrix}, \qquad h^{(k)} = \begin{pmatrix} W^{\frac{1}{2}} r^{(k)} \\ V^{\frac{1}{2}} e^{(k)} \end{pmatrix}.$$

The solution, denoted by $\delta z(\mu)$, of the subproblem satisfies the system of equations

$$B_k \begin{bmatrix} \delta x \\ \delta\tau \end{bmatrix} = -\begin{bmatrix} A_k W r^{(k)} \\ V e^{(k)} + D_k W r^{(k)} \end{bmatrix},$$
$$\|\delta z(\mu)\| = \Delta_k,$$

$\mu > 0$, unless $\| \delta z(0) \| \le \mathit{\Delta}_k$, where B_k denotes the matrix

$$\begin{bmatrix} A_k W A_k^\top + \mu I & A_k W D_k \\ D_k W A_k^\top & V + D_k W D_k + \mu I \end{bmatrix}.$$

Let $\overline{P}_k = V + D_k W D_k + \mu I$. From the buttom part of the system, we get

$$\delta\tau = -\overline{P}_k^{-1} [V e^{(k)} + D_k W r^{(k)} + D_k W A_k^\top \delta x].$$

Substituting it into the top part we have

$$(A_k W^{\frac{1}{2}} \overline{U}_k W^{\frac{1}{2}} A_k^\top + \mu I)\delta x = A_k W^{\frac{1}{2}} \overline{b}^{(k)},$$
$$\overline{U}_k = I - W^{\frac{1}{2}} D_k \overline{P}_k^{-1} D_k W^{\frac{1}{2}},$$
$$\overline{b}^{(k)} = W^{\frac{1}{2}}$$
$$\times [-r^{(k)} + D_k \overline{P}_k^{-1} (V e^{(k)} + D_k W r^{(k)})].$$

Since this system is the *normal equation* of the linear least squares problem

$$\min \left\| \begin{bmatrix} \overline{U}_k^{\frac{1}{2}} W^{\frac{1}{2}} A_k^\top \\ \mu^{\frac{1}{2}} I \end{bmatrix} \delta x + \begin{bmatrix} \overline{U}_k^{-\frac{1}{2}} \overline{b}^{(k)} \\ 0 \end{bmatrix} \right\|,$$

the solution $\delta x^{(k)}$ can be obtained by performing a QR factorization to the matrix $\overline{U}_k^{\frac{1}{2}} W^{\frac{1}{2}} A_k^\top$, a sequence of plane rotations to eliminate $\mu^{1/2} I$ and back substitutions.

For a given value $\mu^{(\ell)}$, $\delta x(\mu^{(\ell)})$ is obtained from the solution of the system and then $\delta\,\tau(\mu^{(\ell)})$ from substitution. If

$$\left|\phi(\mu^{(\ell)})\right| = \left| \left\|\delta z(\mu^{(\ell)})\right\| - \Delta_k \right| \le \rho\Delta_k$$

is satisfied, $\delta z(\mu^{(\ell)})$ is accepted as an approximate solution of the trust region subproblem where $\rho \in (0, 1)$is a preset tolerance. Otherwise, $\mu^{(\ell)}$ is updated to give a new value $\mu^{(\ell+1)}$ and a solution $\delta z(\mu^{(\ell+1)})$ is recomputed from the system. *Moré's updating formula* [4]

$$\mu^{(\ell+1)} = \mu^{(\ell)} - \frac{\phi(\mu^{(\ell)})}{\nabla\phi(\mu^{(\ell)})} \frac{\left\|\delta z(\mu^{(\ell)})\right\|}{\Delta_k}$$

can be used to generate $\mu^{(\ell+1)}$, where $\nabla\phi(\mu^{(\ell)})$ is evaluated from difference approximation

$$\nabla\phi(\mu^{(\ell)}) = \frac{\phi(\mu^{(\ell)}) - \phi(\mu^{(\ell-1)})}{\mu^{(\ell)} - \mu^{(\ell-1)}}.$$

For generalized total least squares problems, the parameter vector x and the variable vector τ can be treated separately. The first order necessary condition for a point to be a solution of the problem can be used to eliminate the τ dependence in the function $f(x, \tau)$. Consider the system of equations

$$\nabla_\tau f = Ve + DWr = 0.$$

These contain m nonlinear equations with m unknowns, each of which only contains one unknown τ_j

for fixed value of x

$$v_j(\tau_j - t_j) + w_j(\phi(x, \tau_j) - y_j)\frac{\partial\phi(x, \tau_j)}{\partial\tau_j} = 0,$$
$$j = 1, \ldots, m.$$

When these equations can be algebraically solved to give an explicit solution expression $\tau(x)$, substitution it into the function $f(x, \tau)$ allows the parameter vector x to be determined by directly using any conventional method to minimize the function $f(x, \tau(x))$ which now is a function of the parameter vector x. However, in most cases, it is impossible or difficult to get an explicit form of the solution $\tau(x)$ and each equation mustbe solved numerically for each given value of x by minimizing the functions

$$\psi(x, \tau_j) = \frac{1}{2}[w_j(\phi(x, \tau_j) - y_j)^2 + v_j(\tau_j - t_j)^2],$$
$$j = 1, \ldots, m,$$

to get an approximate solution, $\overline{\tau}(x)$ say, to the solution $\tau(x)$ so that the values of function $f(x, \tau(x))$ and its derivatives with respect to x can be evaluated from the values x and $\overline{\tau}(x)$.

Assume that $\nabla^2_{\tau\tau} f(x^*, \tau^*)$ is positive definite, then it follows from the *implicit function theorem* [3] that there exist open neighborhoods $N(x^*)$, $N(\tau^*)$ of x^*, τ^* such that for any $x \in N(x^*)$, a unique τ satisfying the system exists in $N(\tau^*)$, this being the vector $\tau(x)$. Furthermore, $\tau(x)$ is continuously differentiable and $\nabla^2_{\tau\tau} f(x, \tau(x))$ is positive definite for all $x \in N(x^*)$. Substituting $\tau(x)$ into the function $f(x, \tau)$ we get a separable minimization problem

$$\min f(x, \tau(x)),$$

which is defined only in terms of x and reduces the problem dimension from $m + n$ to n. The separation is particularly efficient since in most cases, m is very large. Using the *chain rule*, the differentiability of $\tau(x)$ and the fact that $\nabla_\tau f = 0$ we get derivatives of the function $f(x, \tau(x))$

$$\begin{aligned} g(x) =& \nabla_x f + \nabla_x \tau \nabla_\tau f = \nabla_x f, \\ G(x) =& \nabla^2_{xx} f + \nabla^2_{x\tau} f \nabla_x \tau \\ =& \nabla^2_{xx} f - \nabla^2_{x\tau} f [\nabla^2_{\tau\tau} f]^{-1} \nabla^2_{\tau x} f. \end{aligned}$$

Since the positive definiteness of the matrix $G(x)$ is implied by that of the matrix $\nabla^2 f$, if $\nabla^2 f$ is positive definite at the solution (x^*, τ^*), the matrix $G(x^*)$ is positive definite, too.

The *separated Newton method* minimizes the function $f(x, \tau(x))$ using Newton iteration

$$G_k \delta x^{(k)} = -g^{(k)}, \quad x^{(k+1)} = x^{(k)} + \delta x^{(k)}$$

to generate a sequence $\{x^{(k)}\}$, where G_k and $g^{(k)}$ are evaluated at $x^{(k)}$ and $\tau(x^{(k)})$. $\tau(x^{(k)})$ is an approximate solution of the system $\nabla_\tau f = 0$ obtained using Newton iteration

$$\tau_j^{(s+1)} = \tau_j^{(s)} - \frac{\psi'(x^{(k)}, \tau_j^{(s)})}{\psi''(x^{(k)}, \tau_j^{(s)})},$$
$$s = 1, 2, \ldots, \quad j = 1, \ldots, m.$$

When

$$\left|\tau_j^{(s+1)} - \tau_j^{(s)}\right| \le \epsilon,$$

$\tau_j^{(s+1)}$ is accepted as $\tau_j(x^{(k)})$ where $\epsilon > 0$ isa preset small constant. The values t_j and $\tau_j(x^{(k-1)})$, $j = 1, \ldots, m$, can be used as starting values of these iterations for $k = 1$ and $k \ge 2$, respectively.

A careful observation shows that the difference between the Powell–Macdonald method and the separated Newton method is that for given value $x^{(k)}$, the former carries out only one Newton iteration for the system $\nabla_\tau f = 0$ while the later one solves the system quite exactly by repeated doing the iteration.

The separated Newton method still requires the evaluation of secondorder derivatives. Ignoring second order terms in all derivatives $\nabla^2_{xx} f$, $\nabla^2_{x\tau} f$, $\nabla^2_{\tau x} f$ and $\nabla^2_{\tau\tau} f$, we get an approximation to G

$$\begin{aligned} M_k &= A_k W^{\frac{1}{2}} U_k W^{\frac{1}{2}} A_k^\top, \\ U_k &= (I + V^{-1} D_k W D_k)^{-1}. \end{aligned}$$

Then the iteration

$$M_k \delta x^{(k)} = -g^{(k)}, \quad x^{(k+1)} = x^{(k)} + \delta x^{(k)}$$

is the separated Gauss–Newton method [8]. The property that the convergence of Gauss–Newton method for ordinary least squares depends on the closeness of the Gauss–Newton matrix to true Hessian matrix is applicable to the separated Gauss–Newton method. If $M(x^*) = G(x^*)$, the method is locally convergent and rate of convergence is quadratic. If $M(x^*) \neq G(x^*)$, the method may not converge and if it converges, the rate is at best linear. In order to force global convergence, line search or trust region techniques can be incorporated.

For *large residual problems*, the Gauss–Newton matrix M is not a good approximation to G and quasi-Newton updates can be used to generate better approximations. When quasi-Newton updates, for example BFGS update, are used, the separated problem is regarded as a general minimization problem, the special structure of the problem function is not exploited and approximations are not directly obtained from the first order derivatives. The vectors $\delta^{(k)}$ and $\gamma^{(k)}$ used in updating formulas can be defined by

$$\begin{aligned}\delta^{(k)} &= \delta x^{(k)} = x^{(k+1)} - x^{(k)},\\ \gamma^{(k)} &= g(x^{(k+1)}, \tau(x^{(k+1)})) - g(x^{(k)}, \tau(x^{(k)}))\end{aligned}$$

Alternative definitions for $\gamma(k)$ can be derived by using thespecial structure of the derivatives. Two common used definitions for $\gamma^{(k)}$ are

$$\begin{aligned}\gamma^{(k)} &= A_{k+1} W A_{k+1}^{\top} \delta x^{(k)} + A_{k+1} W D_{k+1} \delta\tau^{(k)}\\ &\quad + (A_{k+1} - A_k) r^{(k+1)},\\ \gamma^{(k)} &= A_{k+1} W (r^{(k+1)} - r^{(k)})\\ &\quad + (A_{k+1} - A_k) W r^{(k+1)},\end{aligned}$$

where $\delta\tau^{(k)} = \tau(x(k+1)) - \tau(x^{(k)})$. Numerical experiments favors the last definition of $\gamma^{(k)}$ [9].

Based on the separated Gauss–Newton method and the separated BFGS method, separated hybrid method is a simple generalization of the hybrid method for ordinary nonlinear least squares problems, where a test [9] is derived to determine what step should be chosen at each iteration. When the test chooses the Gauss–Newton step, the approximation B_k to G_k is set to the Gauss– Newton matrix M_k and when the test chooses the BFGS step, the matrix B_k is obtained from B_{k-1} using BFGS updating formula.

When separated methods are used to solve generalized total least squares problems, computational savings can be obtained if we initially ignore errors in t_j, $j = 1, \dots, m$, and just solve an ordinary nonlinear least squares problem. Whenreasonable reduction in the objective function has been achieved, errors in all variables are then considered and separated methods are applied. This modification of any separated method is effective in solving generalized total least squares problems.

See also

- ▶ ABS Algorithms for Linear Equations and Linear Least Squares
- ▶ ABS Algorithms for Optimization
- ▶ Gauss–Newton Method: Least Squares, Relation to Newton's Method
- ▶ Least Squares Orthogonal Polynomials
- ▶ Least Squares Problems
- ▶ Nonlinear Least Squares: Newton-type Methods
- ▶ Nonlinear Least Squares Problems
- ▶ Nonlinear Least Squares: Trust Region Methods

References

1. Boggs PT, Byrd RH, Schnabel RB (1987) A stable and efficient algorithm for nonlinear orthogonal distance regression. SIAM J Sci Statist Comput 8:1052–1078
2. Demming WE (1943) Statistics adjustment of data. Wiley, New York
3. Hestens MR (1966) Calculus of variations and optimal control problems. New York Wiley, New York
4. Moré JJ (1977) The Levenberg-Marquardt algorithm: Implementation and theory. In: Watson GA (ed) Numerical Analysis, Dundee. Lecture Notes Math. Springer, Berlin, pp 105–116
5. O'Neill M, Sinclair IG, Smith J (1969) Polynomial curve fitting when abscisses and ordinates are both subject to error. Comput J 12:52–56
6. Powell DR, Macdonld JR (1972) A rapidly convergent iterative method for the solution of the generalizednonlinear least squares problem. Comput J 15:148–155
7. Southwell WH (1975) Fitting data to nonlinear functions with uncertainties in all measurement variables. Comput J 19:67–73
8. Watson GA (1985) The solution of generalized least squares problems. Internat Ser Numer Math 75:388–400
9. Xu CX (1987) Hybrid methods for nonlinear least squares and related problems. PhD Thesis Univ. Dundee

Generalized Variational Inequalities: A Brief Review

Barbara Panicucci
Department of Applied Mathematics,
University of Pisa, Pisa, Italy

MSC2000: 49J53, 90C30

Article Outline

Keywords and Phrases

Generalized variational inequality; Optimization problem; Gap function

Introduction

The theory as well the applications of variational inequalities (VIs) and the nonlinear complementarity problem (NCP) have proved to be a very powerful tool for studying a wide range of problems arising in mechanics, physics, optimization, and applied sciences. A survey on the developments of VI and NCP is in [7]. In recent years, considerable interest has been shown in developing various extensions and generalizations of the VI problem. An important class of such generalizations, introduced in [2], is the so-called generalized variational inequality (GVI). This class has many important and significant applications in various fields such as mathematical physics and control theory, economics, and transportation equilibrium (see, e. g., [1,11]). For example, it is known that the traffic equilibrium problem can be formulated as a VI when the travel cost between any two given nodes for a given flow is fixed [4]. However, the traffic conditions may vary and the travel cost between two given nodes may not be fixed, but within a cost interval. In this case the corresponding problem can be formulated as a GVI. Moreover, GVI provides a unifying framework for many general problems such us fixed-point, optimization, and complementarity problems. In what follows we give an overview of recent developments concerning the issue of existence of a solution and equivalent reformulations.

Problem Formulation and Framework

In its general form, the GVI problem can be stated as follows:

$$\text{find } x^* \in X \text{ and } u^* \in F(x^*) \text{ such that}$$

$$\langle u^*, y - x^* \rangle \geq 0 \quad \forall\, y \in X\,,$$

where

- $\langle \cdot, \cdot \rangle$ denotes the usual inner product in $\mathbb{R}^n$,
- $X \subseteq \mathbb{R}^n$ is a nonempty closed and convex set,
- $\mathbb{R}^n \rightrightarrows \mathbb{R}^n$ is a set-valued map, i. e., an operator that associates with each $x \in \mathbb{R}^n$ a set $F(x) \subseteq \mathbb{R}^n$.

If F is a single valued function, then the GVI problem reduces to the classical VI, which is to find $x^* \in X$ such that

$$\langle F(x^*), y - x^* \rangle \geq 0 \quad \forall\, y \in X\,.$$

In connection with the set-valued map $F\colon \mathbb{R}^n \rightrightarrows \mathbb{R}^n$ a few definitions need to be recalled. First, F is characterized by its graph:

$$\operatorname{graph}(F) = \{(x, u) \in \mathbb{R}^n \times \mathbb{R}^n\colon\ u \in F(x)\}\,.$$

The image of X under F is

$$F(X) = \bigcup_{x \in X} F(x)\,,$$

the inverse of F is defined by

$$F^{-1}(u) = \{x\colon\ u \in F(x)\},$$

and the domain of F is the set

$$\operatorname{dom}(F) = \{x \in \mathbb{R}^n\colon F(x) \neq \emptyset\}\,.$$

Throughout we assume that $\operatorname{dom}(F) \supseteq X$. Over the past two decades, most effort has been concentrated on the question of the existence of solutions to GVI problems. The study of the existence of solutions of GVI involves several continuity properties of set-valued maps. We recall these conditions in the sequel.

- A set-valued map $F\colon \mathbb{R}^n \rightrightarrows \mathbb{R}^n$ is said to be upper semicontinuous (u.s.c.) at $x \in \mathbb{R}^n$ if for each open set $V \supseteq F(x)$ there exists a neighborhood U of x such that $F(U) \subseteq V$; F is u.s.c. on a set $X \subseteq \mathbb{R}^n$ if it is u.s.c. at every point in X.
- A set-valued map $F\colon \mathbb{R}^n \rightrightarrows \mathbb{R}^n$ is upper hemicontinuous on $X \subseteq \mathbb{R}^n$, if its restriction to line segments of X is upper semicontinuous.

The study of the existence of solutions of GVI involves also some monotonicity-type properties for set-valued maps. In what follows we recall the definitions.

(M1) F is quasimonotone on X if, for every pair of distinct points $x, y \in X$ and every $u \in F(x)$, $v \in F(y)$, we have:

$$\langle v, x - y \rangle > 0 \Longrightarrow \langle u, x - y \rangle \geq 0\,.$$

(M2) F is properly quasimonotone on X if, for any $x^1, \ldots, x^n \in X$ and any $\lambda_1, \ldots, \lambda_n > 0$ with $\sum_{i=1}^{n} \lambda_i = 1$, there exists $j \in \{1, \ldots, n\}$ such that for all $u^j \in F(x^j)$ and $x = \sum_{i=1}^{n} \lambda_i x^i$, we have:

$$\langle u^j, x - x^j \rangle \leq 0\,.$$

(M3) F is pseudomonotone on X if, for every pair of distinct points $x, y \in X$ and every $u \in F(x), v \in F(y)$, we have:

$$\langle v, x - y \rangle \geq 0 \Longrightarrow \langle u, x - y \rangle \geq 0\,.$$

(M4) F is monotone on X if, for every pair of distinct points $x, y \in X$ and every $u \in F(x), v \in F(y)$, we have:

$$\langle u - v, x - y \rangle \geq 0\,.$$

(M5) F is strictly monotone on X if, for every pair of distinct points $x, y \in X$ and every $u \in F(x), v \in F(y)$, we have:

$$\langle u - v, x - y \rangle > 0\,.$$

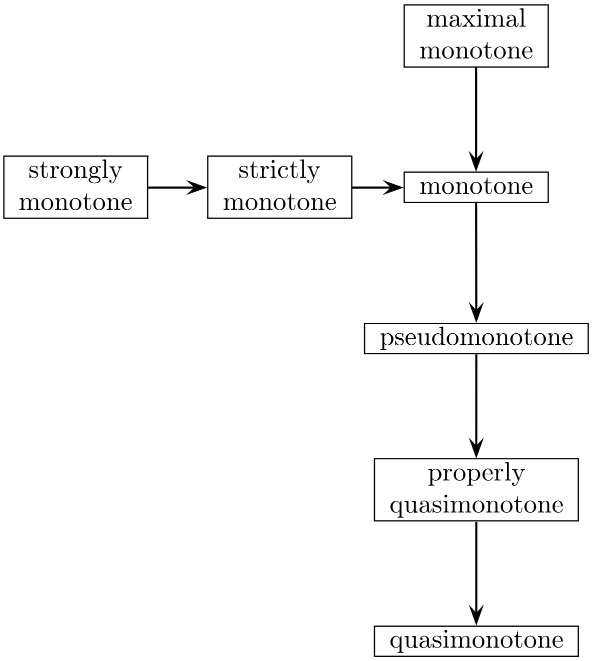
(M6) F is strongly monotone on X with constant $\beta > 0$ if, for every pair of distinct points $x, y \in X$ and every $u \in F(x), v \in F(y)$, we have:

$$\langle u - v, x - y \rangle \geq \beta \|x - y\|^2\,,$$

where $\|\cdot\|$ denotes the classical euclidean norm.

(M7) F is maximal monotone on X if it is monotone on X and its graph is not properly contained in the graph of any other monotone operator on X.

The relationships among these kinds of monotonicity are represented in Fig. 1.



Generalized Variational Inequalities: A Brief Review, Figure 1
Relationships among generalized monotonicity conditions

Existence and Uniqueness

In recent years the existence of solutions to GVIs has been investigated extensively. In what follows we provide some of the most fundamental results. The basic result on the existence of a solution to the GVI problem requires the set X to be compact and convex and the map F to be u.s.c. From this basic result many others can be derived by replacing the compactness of X with additional coercivity conditions on F.

Existence of Solutions: Bounded Domain

This section presents some existence results for solutions of GVI in the case of a compact domain. The following existence theorem exploits the formulation of GVI as a fixed-point problem.

Theorem 1 ([8]) *If X is compact and F is u.s.c. on X with compact and convex values, then GVI has a solution.*

Theorem 2 ([12]) *If X is compact and F is upper hemicontinuous and properly quasimonotone on X with compact and convex values, then GVI has a solution.*

Existence of Solutions: Unbounded Domain

The existence of solutions of GVI on unbounded domains is guaranteed by the same conditions as for bounded domains, together with a coercivity condition. In the literature various coercivity conditions have been considered. In particular (see [5]):

(C1)

$$\exists\, R > 0, \quad \forall x \in X \setminus X_R, \qquad \forall u \in F(x)\,,$$
$$\exists\, y \in X_R\colon \quad \langle u, y - x\rangle < 0\,;$$

(C2)

$$\exists\, R > 0, \quad \forall\, x \in X \setminus X_R, \quad \exists\, y \in X_R\,,$$
$$\forall\, u \in F(x)\colon \quad \langle u, y - x\rangle < 0\,;$$

(C3)

$$\exists\, R > 0, \quad \forall\, x \in X \setminus X_R, \quad \exists\, y \in X_R\,,$$
$$\exists\, v \in F(y)\colon \quad \langle v, y - x\rangle < 0\,;$$

(C4)

$$X_\infty \cap (F(X))^- = \{0\}\,,$$

where

$$X_R = \{x \in X\colon \|x\| \le R\}$$

and

$$(F(X))^- = \{d \in \mathbb{R}^n\colon \langle u, d\rangle \le 0, \forall u \in F(X)\}$$

is the polar cone of $F(X)$. Further, the recession cone X_∞, for X closed and convex, is defined by

$$X_\infty = \{d \in \mathbb{R}^n\colon x + t\,d \in X,\ \forall\, t \ge 0,\ x \in X\}\,.$$

Some basic relationships among these coercivity conditions are summarized in the following result.

Theorem 3 ([5])

- *(C2) $\Longrightarrow$ (C1).*
- *If F has convex values, then (C2) and (C1) are equivalent.*
- *If F is pseudomonotone on X, then (C3) $\Longrightarrow$ (C2).*
- *(C4) $\Longrightarrow$ (C3).*
- *If F is upper hemicontinuous and pseudomonotone on X, then (C2), (C3) and (C4) are equivalent.*
- *If F has convex values and it is upper hemicontinuous and pseudomonotone on X, then (C1), (C2), (C3), and (C4) are equivalent.*

The coercivity conditions allow us to exhibit a sufficiently large ball intersecting with X such that no point outside this ball is a solution of the GVI; then one can establish the existence of a solution stated below.

Theorem 4 ([5]) *If F is upper hemicontinuous and pseudomonotone on X with compact and convex values, then the following statements are equivalent:*

- *GVI has a nonempty and compact solution set.*
- *(C1) holds;*
- *(C2) holds.*
- *(C3) holds.*
- *(C4) holds.*

In what follows we state an existence theorem for which we require neither the upper semicontinuity of F, nor the compactness, nor the convexity of $F(x)$, but we need the maximal monotonicity of F.

Theorem 5 ([15]) *Assume that F is maximal monotone on $\mathbb{R}^n$. Then the solution set of GVI is nonempty and compact if and only if (C4) holds.*

In general, GVI can have more than one solution. The following theorem gives conditions under which GVI can have at most one solution.

Theorem 6

- *If F is strictly monotone on X, then GVI has at most one solution.*
- *If F is u.s.c., strongly monotone on X, and has nonempty convex and compact values, then GVI has a unique solution.*

GVI and Related Problems

As stated, the theory of GVI is a powerful unifying methodology that contains as special cases several well-known problems such as fixed-point, optimization, and complementarity problems. In what follows we describe these equivalent formulations of the GVI problem. Such formulations can be very beneficial for both analytical and computational purposes. Indeed we can apply classic results of these problems to treat the GVI.

GVI and Fixed-Point Problems

In what follows we exploit the formulation of GVI as a fixed-point problem. We recall that x^* is a fixed point of the set-valued map $F\colon X \rightrightarrows \mathbb{R}^n$ if

$$x^* \in X \quad \text{and} \quad x^* \in F(x^*) \,.$$

The fixed-point reformulation is very relevant for the GVI problem. Indeed we can apply Kakutani's fixed-point theorem, which is instrumental for proving the existence result on a bounded domain. We define the following set-valued map:

$$\begin{aligned}\Theta\colon X \times \operatorname{conv}(F(X)) &\rightrightarrows X \times \operatorname{conv}(F(X))\\ (x, u) &\mapsto \Psi(u) \times F(x)\,,\end{aligned}$$

where $\Psi(u) = \arg\min_{x\in X}\langle u, x\rangle$ is the set of constrained minimizers of the map $\langle u, x\rangle$ on X and $\operatorname{conv}(F(X))$ denotes the convex hull of $F(X)$. Assuming that X is compact, $\Psi(u)$ results in being nonempty. It easy to see that the problem of finding a fixed point (x^*, u^*) of Θ, i. e.,

$$x^* \in K, \quad u^* \in F(x^*), \quad x^* \in \arg\min_{x\in K}\langle u^*, x\rangle \,,$$

is equivalent to GVI.

It is worth noting that the GVI problem can also be formulated as an inclusion as follows:

$$\text{find } x^* \in K \text{ such that } 0 \in F(x^*) + N_K(x^*) \,,$$

i. e., finding a zero of the set-valued map $F + N_K$ in the domain X, where the normal cone $N_X(x)$ to the set X at point $x \in X$ is given by:

$$N_X(x) = \{d \in \mathbb{R}^n\colon \langle d, y - x\rangle \le 0 \quad \forall\, y \in X\} \,.$$

GVI and Optimization Problems

Let us consider the constrained optimization problem:

$$\begin{cases}\min f(x)\\ x \in X\,,\end{cases} \tag{1}$$

where

- X is a closed and convex subset of $\mathbb{R}^n$,
- The objective function f is defined on an open neighborhood of X, denoted Ω.

It is well known that if f is continuously differentiable, then the classical VI with $F = \nabla f$ is a necessary optimality condition for (1). The VI gives also a sufficient condition if f is pseudoconvex on X, i. e.,

$$f(x) > f(y) \Longrightarrow \langle \nabla f(x), y - x\rangle < 0 \,,$$

for all $x, y \in X$.

Therefore, if f is continuously differentiable and pseudoconvex on X, the VI with $F = \nabla f$ is equivalent to the optimization problem (1). In what follows we extend these results in terms of GVI when $f\colon \Omega \to \mathbb{R}$ is a locally Lipschitz continuous function, that is, for each point $x \in \Omega$ there exists a neighborhood U of x such that f is Lipschitz continuous on U. To this end we recall some basic facts about Clarke calculus for a locally Lipschitz continuous function, see [3]. The Clarke's generalized derivative of f at x in the direction v, denoted by $f^0(x;v)$, is given by

$$f^0(x; v) = \limsup_{\substack{y\to x\\ t\downarrow 0}} \frac{f(y + t\,v) - f(y)}{t} \,.$$

The generalized gradient of f at x, denoted by $\partial f(x)$, is defined as follows:

$$\partial f(x) = \{\xi \in \mathbb{R}^n\colon \langle \xi, v\rangle \le f^0(x; v) \quad \forall\, v \in \mathbb{R}^n\} \,.$$

A generalized derivative can be obtained from the generalized gradient:

$$f^0(x; v) = \max\{\langle \xi, v\rangle : \xi \in \partial f(x)\} \,.$$

We can extend the definition of pseudoconvexity for a locally Lipschitz continuous function $f\colon \Omega \to \mathbb{R}$, [16]: f is pseudoconvex on Ω if, for all $x, y \in \Omega$, there exists $\xi \in \partial f(x)$ such that

$$\langle \xi, y - x\rangle \ge 0 \quad \Longrightarrow \quad f(x) \le f(y) \,.$$

Let us now consider the GVI with Clarke gradient operator $F = \partial f$. We can state the following result.

Theorem 7 ([3]) *A GVI with $F = \partial f$ provides necessary optimality conditions for problem (1).*

In general, a GVI does not give sufficient optimality conditions. However, as shown in [16], when f is pseudoconvex on Ω, the GVI gives sufficient optimality conditions too. Consequently, as for the single-valued case, if f is pseudoconvex on Ω, a GVI with $F = \partial f$ is equivalent to the optimization problem (1). The above discussion focused on the GVI with gradient operator; however, an arbitrary set-valued map, in general, is not a gradient map. A powerful tool in dealing with the GVI problem by way of its equivalent optimization reformulation is given by the so-called gap functions. Specifically, we say that a function $\varphi\colon \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ is a gap function for GVI if

- $\varphi(x, u) \geq 0$ for all $(x, u) \in \operatorname{graph}(F)$,
- x^* is a solution of GVI if and only if $x^* \in X$ and there exists $u^* \in F(x^*)$ such that $\varphi(x^*, u^*) = 0$.

Hence, the GVI problem can be rewritten as the following constrained optimization problem:

$$\begin{cases} \min\ \varphi(x, u) \\ (x, u) \in \operatorname{graph}(F) . \end{cases}$$

An example of a gap function, proposed in [6], is:

$$\varphi(x, u) = \sup_{y \in X} \langle u, x - y \rangle , \qquad (x, u) \in \mathbb{R}^n \times \mathbb{R}^n . \quad (2)$$

The function $\varphi(x, \cdot)$ is convex and closed for every fixed $x \in \mathbb{R}^n$ and $\varphi(\cdot, u)$ is affine for every fixed $u \in \mathbb{R}^n$ (see [6]). It is worth noting that ϕ represents a duality gap in the Mosco duality scheme [14] for GVI. Let us consider this more general GVI problem: find $x^* \in \mathbb{R}^n$ and $u^* \in F(x^*)$ such that

$$\langle u^*, x - x^* \rangle \geq \phi(x^*) - \phi(x) \quad \forall\, x \in \mathbb{R}^n , \quad (3)$$

where $\phi\colon \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ is a proper, lower semicontinuous convex function. The dual problem of (3) is defined as: find $v^* \in \mathbb{R}^n$ and $y^* \in -F^{-1}(-v^*)$ such that

$$\langle y^*, v - v^* \rangle \geq \phi^*(v^*) - \phi^*(y) \quad \forall\, v \in \mathbb{R}^n ,$$

where $\phi^*(v) = \sup_{x \in \mathbb{R}^n} \{\langle v, x \rangle - \phi(x)\}$ is the Fenchel conjugate of φ.

Theorem 8 ([15]) *The gap function (2) measures the duality gap of Mosco's duality scheme:*

$$\phi(x) + \phi^*(-u) + \langle u, x \rangle = \begin{cases} \varphi(x, u) & \text{if } x \in X \\ +\infty & \text{otherwise.} \end{cases}$$

The gap function ϕ is not differentiable in general. Moreover, when graph (F) is unbounded, it is in general not finite valued. These drawbacks can be avoided by using a regularized gap function. Let us consider

$$\varphi_G(x, u) = \max_{y \in X} \left[\langle u, x - y \rangle - \frac{1}{2} \|x - y\|_G^2 \right] ,$$

where $(x, u) \in \mathbb{R}^n \times \mathbb{R}^n$, G is a symmetric positive definite matrix, and $\|\cdot\|_G$ is the norm in $\mathbb{R}^n$ defined by $\|x\|_G = \sqrt{\langle x, G\, x \rangle}$. This function, introduced in [6] for generalized quasivariational inequalities, i. e., GVIs where set X depends on solution x, is a gap function for GVI and is called a regularized gap function. Since

$$\psi_G(x, u, y) = \langle u, x - y \rangle - \frac{1}{2} \|x - y\|_G^2$$

is strongly concave with respect to y, there is a unique maximizer over X denoted by $y(x, u)$. If we denote the projection operator onto set X with respect to the norm $\|\cdot\|_G$ by $\Pi_{X,G}(\cdot)$, it is easy to check that this maximizer is

$$y(x, u) = \Pi_{X,G}(x - G^{-1}\, u) .$$

Therefore, the regularized gap function

$$\varphi_G(x, u) = \langle u, x - y(x, u) \rangle - \frac{1}{2} \|x - y(x, u)\|_G^2$$

is finite valued everywhere. Moreover, the regularized gap function is continuously differentiable, and its gradient is given by

$$\nabla_x \varphi_G(x, u) = u + G\, [y(x, u) - x] ,$$
$$\nabla_u \varphi_G(x, u) = x - y(x, u) .$$

Therefore, using the regularized gap function we obtain an equivalent differentiable optimization reformulation of the GVI problem. Gap functions can be used in the design of numerical algorithms for solving the GVI.

GVI and Complementarity Problems

It is well known that, when X is a closed convex cone and $F\colon X \to \mathbb{R}^n$, the VI problem is equivalent to the NCP problem, which consists in finding $x^* \in X$ such that

$$F(x^*) \in X^* \quad \text{and} \quad \langle F(x^*), x^* \rangle = 0 ,$$

where

$$X^* = \{d \in \mathbb{R}^n : \langle u, d \rangle \geq 0, \forall u \in X\}$$

is the negative polar cone of X. Such a relationship is preserved in the GVI problems. First, let us consider an extension of the NCP problem, see [17], that can be defined as follows.

Let X be a closed convex cone of $\mathbb{R}^n$ and F a set-valued map. The generalized complementarity problem (GCP) is to find $x^* \in X$ such that there exists $u^* \in F(x^*)$ satisfying the following properties:

$$u^* \in X^* \quad \text{and} \quad \langle u^*, x^* \rangle = 0\,.$$

As in the single-valued case, both problems GVI and GCP have the same solution set if the underlying set X is a closed convex cone.

References

1. Aubin JP (1984) L'Analyse Non Linéaire et Ses Motivations Economiques. Masson, Paris
2. Browder FE (1965) Multivalued Monotone Nonlinear Mappings and Duality Mappings in Banach Spaces. Trans Am Math Soc 71:780–785
3. Clarke FH (1990) Optimization and nonsmooth analysis, vol 5 of Classics in Applied Mathematics. SIAM, Philadelphia
4. Dafermos S (1980) Traffic Equilibrium and Variational Inequalities. Transp Sci 14:42–54
5. Daniilidis A, Hadjisavvas N (1999) Coercivity conditions and variational inequalities. Math Programm 86:433–438
6. Dietrich H (1999) A smooth dual gap function solution to a class of quasivariational inequalities. J Math Anal Appl 235:380–393
7. Facchinei F, Pang JS (2003) Finite-dimensional variational inequalities and complementarity problems, vol I, II. Springer, New York
8. Fang SC, Peterson EL (1982) Generalized variational inequalities. J Optim Theory Appl 38:363–383
9. Giannessi F, Maugeri A, Pardalos P (2001) Equilibrium Problems: Nonsmooth Optimization and Variational Inequality Models. Kluwer, Dordrecht
10. Giannessi F, Pardalos P, Rapcsák T (2001) Optimization theory. Recent developments. Kluwer, Dordrecht
11. Harker PT, Pang S (1990) Finite-Dimensional Variational Inequality and Nonlinear Complementarity Problems: a survey of theory, algorithms and applications. Math Programm 48:161–220
12. John R (2001) A note on Minty variational inequalities and generalized monotonicity. In: Generalized convexity and generalized monotonicity. Springer, Berlin, pp 240–246
13. Konnov I (2001) Combined relaxation methods for variational inequalities. Springer, Berlin
14. Mosco U (1972) Dual Variational Inequalities. J Math Anal Appl 202–206
15. Panicucci B, Pappalardo M, Passacantando M (2006) On finite-dimensional generalized variational inequalities. J Indust Manag Optim 2:43–53
16. Penot J-P, Quang PH (1997) Generalized convexity of functions and generalized monotonicity of set-valued maps. J Optim Theory Appl 92:343–356
17. Saigal R (1976) Extension of the generalized complementarity problem. Math Oper Res 1:260–266

General Moment Optimization Problems

GEORGE A. ANASTASSIOU
Department Math. Sci., The University Memphis,
Memphis, USA

MSC2000: 28-XX, 49-XX, 60-XX

Article Outline

Keywords

Geometric moment theory; Probability; Integral constraint; Optimal integral bounds subject to moment conditions; Finite moment problem; Convex moment problem; Convexity; Infinite moment problem

In this article we describe the main moment problems and their solution methods from theoretical to applied. In particular we present the standard moment problem, the convex moment problem, and the infinite many conditions moment problem. Optimization moment theory has a lot of important applications in many sciences and subjects, for a detailed list please see the final section.

The Standard Moment Problem

Let $g_1, \ldots, g_n$ and h be given real-valued Borel measurable functions on a fixed measurable space $X := (X, A)$. We would like to find the best upper and lower bound on

$$\mu(h) := \int_X h(t)\mu(dt),$$

given that μ is a probability measure on X with prescribed moments

$$\int g_i(t)\, \mu(dt) = y_i, \qquad i = 1, \ldots, n.$$

Here we assume μ such that

$$\int_X |g_i|\ \mu(dt) < +\infty, \qquad i = 1, \ldots, n,$$

and

$$\int_X |h|\ \mu(dt) < +\infty.$$

For each $y := (y_1, \ldots, y_n) \in \mathbf{R}^n$, consider the optimal quantities

$$L(y) := L(y|h) := \inf_\mu \mu(h),$$
$$U(y) := U(y|h) := \sup_\mu \mu(h),$$

where μ is a probability measure as above with

$$\mu(g_i) = y_i, \qquad i = 1, \ldots, n.$$

If there is no such probability measure μ we set $L(y) := +\infty$, $U(y) := -\infty$.

If $h := \chi_S$ the characteristic function of a given measurable set S of X, then we agree to write

$$L(y|\chi_S) := L_S(y), \quad U(y|\chi_S) := U_S(y).$$

Hence, $L_S(y) \leq \mu(S) \leq U_S(y)$. Consider $g\colon X \to \mathbf{R}^n$ such that $g(t) := (g_1(t), \ldots, g_n(t))$. Set also $g_0(t) := 1$, all $t \in X$. Here we basically present J.H.B. Kemperman's (1968) geometric methods for solving the above main moment problems [13] which were related to and motivated by [18,20,24]. The advantage of the geometric method is that many times is simple and immediate giving us the optimal quantities L, U in a closed-numerical form, on the top of this is very elegant. Here the σ-field A contains all subsets of X.

The next result comes from [22,23,25].

Theorem 1 *Let $f_1, \ldots, f_N$ be given real-valued Borel measurable functions on a measurable space Ω (such as $g_1, \ldots, g_n$ and h on X). Let μ be a probability measure on Ω such that each f_i is integrable with respect to μ. Then there exists a probability measure μ' of finite support on Ω (i. e., having nonzero mass only at a finite number of points) satisfying*

$$\int_\Omega f_i(t)\, \mu(dt) = \int_\Omega f_i(t)\, \mu'(dt),$$

all $i = 1, \ldots, N$.

One can even achieve that the support of μ' has at most $N+1$ points. So from now on we can talk only about finitely supported probability measures.

Call

$$V := \operatorname{conv} g(X)$$

(conv stands for convex hull), where $g(X) := \{z \in \mathbf{R}^n\colon z = g(t) \text{ for some } t \in X\}$ is a curve in $\mathbf{R}^n$ (if $X = [a, b] \subset \mathbf{R}$ or if $X = [a, b] \times [c, d] \subset \mathbf{R}^2$).

Let $S \subset X$, and let $M^+(S)$ denote the set of all probability measures on X whose support is finite and contained in S.

The next results come from [13].

Lemma 2 *Given $y \in \mathbf{R}^n$, then $y \in V$ if and only if $\exists \mu \in M^+(X)$ such that*

$$\mu(g) = y$$

(i. e. $\mu(g_i) := \int_X g_i(t)\, \mu(dt) = y_i$, $i = 1, \ldots, n$).

Hence $L(y|h) < +\infty$ if and only if $y \in V$ (note that by Theorem 1,

$$L(y|h) = \inf\left\{\mu(h)\colon\ \mu \in M^+(X), \mu(g) = y\right\}$$

and

$$U(y|h) = \sup\left\{\mu(h)\colon\ \mu \in M^+(X), \mu(g) = y\right\}).$$

Easily one can see that

$$L(y) := L(y|h)$$

is a convex function on V, i. e.

$$L(\lambda y' + (1-\lambda)y'') \leq \lambda L(y') + (1-\lambda)L(y''),$$

whenever $0 \leq \lambda \leq 1$ and $y', y'' \in V$. Also $U(y) := U(y|h) = -L(y|-h)$ is a concave function on V.

One can also prove that the following three properties are equivalent:

i) $\operatorname{int}(V) :=$ interior of $V \neq \phi$;

ii) $g(X)$ is not a subset of any hyperplane in $\mathbf{R}^n$;

iii) $1, g_1, \ldots, g_n$ are linearly independent on X.

From now on we assume that $1, g_1, \ldots, g_n$ are linearly independent, i. e. $\operatorname{int}(V) \neq \phi$.

Let D^* denote the set of all $(n+1)$-tuples of real numbers $d^* := (d_0, \ldots, d_n)$ satisfying

$$h(t) \geq d_0 + \sum_{i=1}^{n} d_i g_i(t), \qquad \text{all} \quad t \in X. \tag{1}$$

Theorem 3 *For each $y \in int\ (V)$ we have that*

$$L(y|h) \tag{2}$$
$$= \sup \left\{ d_0 + \sum_{i=1}^{n} d_i y_i : \ d^* = (d_0, \ldots, d_n) \in D^* \right\} .$$

Given that $L(y|\,h) > -\infty$, the supremum in (2) is even assumed by some $d^* \in D^*$.

If $L(y|h)$ is finite in $\operatorname{int}(V)$, then for almost all $y \in \operatorname{int}(V)$ the supremum in (2) is assumed by a unique $d^* \in D^*$. Thus $L(y|\,h) < +\infty$ in $\operatorname{int}(V)$ if and only if $D^* \neq \emptyset$. Note that $y := (y_1, \ldots, y_n) \in \operatorname{int}(V) \subset \mathbf{R}^n$ if and only if $d_0 + \sum_{i=1}^{n} d_i y_i > 0$ for each choice of the real constants d_i not all zero such that $d_0 + \sum_{i=1}^{n} d_i g_i(t) \geq 0$, all $t \in X$. (The last statement comes from [8 p. 5] and [12 p. 573].)

If h is bounded then $D^* \neq \emptyset$, trivially.

Theorem 4 *Let $d^* \in D^*$ be fixed and set*

$$B(d^*)$$
$$:= \left\{ z = g(t) : \ d_0 + \sum_{i=1}^{n} d_i g_i(t) = h(t), t \in X \right\} \tag{3}$$

Then for each point

$$y \in \operatorname{conv} B(d^*) \tag{4}$$

the quantity $L(y|h)$ is found as follows. Set

$$y = \sum_{j=1}^{m} p_j g(t_j)$$

with

$$g(t_j) \in B(d^*),$$

and

$$p_j \geq 0, \quad \sum_{j=1}^{m} p_j = 1. \tag{5}$$

Then

$$L(y|h) = \sum_{j=1}^{m} p_j h(t_j) = d_0 + \sum_{i=1}^{n} d_i y_i. \tag{6}$$

Theorem 5 *Let $y \in int(V)$ be fixed. Then the following are equivalent:*

i) $\exists \mu \in M^+\ (X)$ such that $\mu(g) = y$ and $\mu(h) = L(y|h)$, i. e. infimum is attained.

ii) $\exists d^* \in D^*$ satisfying (4).

Furthermore for almost all $y \in int(V)$ there exists at most one $d^* \in D^*$ satisfying (4).

In many situations the above infimum is not attained so that Theorem 4 is not applicable. The next theorem has more applications. For that, set

$$\eta(z) := \lim_{\delta \to 0} \inf \inf_{t} \{ h(t) : \ t \in X, |g(t) - z| < \delta \} . \tag{7}$$

If $\varepsilon \geq 0$ and $d^* \in D^*$, define

$$C_\varepsilon(d^*)$$
$$:= \left\{ z \in \overline{g(T)} : \ 0 \leq \eta(z) - \sum_{i=0}^{n} d_i z_i \leq \varepsilon \right\} , \tag{8}$$

and

$$G(d^*) := \bigcap_{N=1}^{\infty} \overline{\operatorname{conv}} C_{\frac{1}{N}}(d^*). \tag{9}$$

It is easily proved that $C_\varepsilon(d^*)$ and $G(d^*)$ are closed; furthermore $B(d^*) \subset C_0(d^*) \subset C_\varepsilon(d^*)$, where $B(d^*)$ is defined by (3).

Theorem 6 *Let $y \in int(V)$ be fixed.*

i) Let $d^* \in D^*$ be such that $y \in G(d^*)$. Then

$$L(y|h) = d_0 + d_1 y_1 + \cdots + d_n y_n. \tag{10}$$

ii) *Assume that g is bounded. Then there exists* $d^* \in D^*$ *satisfying*

$$y \in \operatorname{conv} C_0(d^*) \subset G(d^*)$$

and

$$L(y|h) = d_0 + d_1 y_1 + \cdots + d_n y_n. \tag{11}$$

iii) *We further obtain, whether or not g is bounded, that for almost all* $y \in int(V)$ *there exists at most one* $d^* \in D^*$ *satisfying* $y \in G(d^*)$.

The above results suggest the following practical simple geometric methods for finding $L(y|h)$ and $U(y|h)$, see [13].

The Method of Optimal Distance

Call

$$M := \operatorname{conv}_{t \in X}(g_1(t), \dots, g_n(t), h(t)).$$

Then $L(y|h)$ is equal to the *smallest* distance between $(y_1, \dots, y_n, 0)$ and $(y_1, \dots, y_n, z) \in \overline{M}$. Also $U(y|h)$ is equal to the *largest* distance between $(y_1, \dots, y_n, 0)$ and $(y_1, \dots, y_n, z) \in \overline{M}$. Here, $\overline{M}$ stands for the closure of M. In particular we see that $L(y|h) = \inf\{y_{n+1} : (y_1, \dots, y_n, y_{n+1}) \in M\}$ and

$$\begin{aligned} &U(y|h) \\ &\quad = \sup\{y_{n+1} \colon (y_1, \dots, y_n, y_{n+1}) \in M\}. \end{aligned} \tag{12}$$

Example 7 Let μ denote probability measures on $[0, a]$, $a > 0$. Fix $0 < d < a$. Find

$$L := \inf_{\mu} \int_{[0,a]} t^2 \, \mu(dt)$$

and

$$U := \sup_{\mu} \int_{[0,a]} t^2 \, \mu(dt)$$

subject to

$$\int_{[0,a]} t \, \mu(dt) = d.$$

So consider the graph $G := \{(t, t^2) \colon 0 \le t \le a\}$. Call $M := \overline{\operatorname{conv} G} = \operatorname{conv} G$.

A direct application of the optimal distance method here gives us $L = d^2$ (an optimal measure μ is supported at d with mass 1), and $U = da$ (an optimal measure μ here is supported at 0 and a with masses $(1 - d/a$ and d/a, respectively).

The Method of Optimal Ratio

We would like to find

$$L_S(y) := \inf \mu(S)$$

and

$$U_S(y) := \sup \mu(S),$$

over all probability measures μ such that

$$\mu(g_i) = y_i, \qquad i = 1, \dots, n.$$

Set $S' := X - S$. Call $W_S := \overline{\operatorname{convg}(S)}$, $W_{S'} := \overline{\operatorname{convg}(S')}$ and $W := \overline{\operatorname{convg}(X)}$, where $g := (g_1, \dots, g_n)$.

Finding $L_S(y)$.

1) Pick a boundary point z of W and 'draw' through z a hyperplane H of support to W.
2) Determine the hyperplane H' parallel to H which supports $W_{S'}$ as well as possible, and on the same side as H supports W.
3) Denote

$$A_d := W \cap H = W_S \cap H$$

and

$$B_d := W_{S'} \cap H'.$$

Given that $H' \neq H$, set $G_d := \overline{\operatorname{conv}}(A_d \cup B_d)$. Then we have that

$$L_S(y) = \frac{\Delta(y)}{\Delta}, \tag{13}$$

for each $y \in \operatorname{int}(V)$ such that $y \in G_d$. Here, $\Delta(y)$ is the distance from y to H' and Δ is the distance between the distinct parallel hyperplanes H, H'.

Finding $U_S(y)$. (Note that $U_S(y) = 1 - L_{S'}(y)$.)

1) Pick a boundary point z of W_S and 'draw' through z a hyperplane H of support to W_S. Set $A_d := W_S \cap H$.
2) Determine the hyperplane H' parallel to H which supports $g(X)$ and hence W as well as possible, and on the same side as H supports W_S. We are interested only in $H' \neq H$ in which case H is between H' and W_S.
3) Set $B_d := W \cap H' = W_{S'} \cap H'$. Let G_d as above. Then

$$U_S(y) = \frac{\Delta(y)}{\Delta}, \tag{14}$$

for each $y \in \operatorname{int}(V)$, where $y \in G_d$, assuming that H and H' are distinct. Here, $\Delta(y)$ and Δ are defined as above.

Examples here of calculating $L_S(y)$ and $U_S(y)$ tend to be more involved and complicated, however the applications are many.

The Convex Moment Problem

Definition 8 Let $s \geq 1$ be a fixed natural number and let $x_0 \in \mathbf{R}$ be fixed. By $m_s(x_0)$ we denote the set of probability measures μ on $\mathbf{R}$ such that the associated cumulative distribution function F possesses an $(s-1)$th derivative $F^{(s-1)}\ (x)$ over $(x_0, +\infty)$ and furthermore $(-1)^s\ F^{(s-1)}\ (x)$ is convex in $(x_0, +\infty)$.

Description of the Problem

Let g_i, $i = 1, \ldots, n$; h are Borel measurable functions from $\mathbf{R}$ into itself. These are assumed to be locally integrable on $[x_0, +\infty)$ relative to Lebesgue measure. Consider $\mu \in m_s(x_0)$, $s \geq 1$ such that

$$\mu(|g_i|) := \int_{\mathbb{R}} |g_i(t)|\ \mu(dt) < +\infty, \qquad i = 1, \ldots, n \qquad (15)$$

and

$$\mu(|h|) := \int_{\mathbb{R}} |h(t)|\ \mu(dt) < +\infty. \qquad (16)$$

Let $c := (c_1, \ldots, c_n) \in \mathbf{R}^n$ be such that

$$\mu(g_i) = c_i, \quad i = 1, \ldots, n, \quad \mu \in m_s(x_0). \qquad (17)$$

We would like to find $L(c) := \inf_\mu\ \mu\ (h)$ and

$$U(c) := \sup_\mu \mu(h), \qquad (18)$$

where μ is as above described.

Here, the method will be to transform the above convex moment problem into an ordinary one handled by the first section, see [14].

Definition 9 Consider here another copy of $(\mathbf{R}, \mathbf{B})$; $\mathbf{B}$ is the Borel σ-field, and further a given function $P(y, A)$ on $\mathbf{R} \times \mathbf{B}$.

Assume that for each fixed $y \in \mathbf{R}$, $P(y, \cdot)$ is a probability measure on $\mathbf{R}$, and for each fixed $A \in \mathbf{B}$, $P(\cdot, A)$ is a Borel-measurable real-valued function on $\mathbf{R}$. We call P a *Markov kernel*. For each probability measure ν on $\mathbf{R}$, let $\mu := T\nu$ denote the probability measure on $\mathbf{R}$ given by

$$\mu(A) := (T\nu)(A) := \int_{\mathbb{R}} P(y, A)\ \nu(dy).$$

T is called a *Markov transformation.*

In particular: Define the kernel

$$K_s(u, x) := \begin{cases} \frac{s(u-x)^{s-1}}{(u-x_0)^s} & \text{if } x_0 < x < u, \\ 0 & \text{elsewhere.} \end{cases} \qquad (19)$$

Notice $K_s\ (u, x) \geq 0$ and $\int_{\mathbf{R}} K_s\ (u, x) = dx = 1$, all $u > x_0$. Let δ_u be the unit (Dirac) measure at u. Define

$$P_s(u, A) := \begin{cases} \delta_u(A) & \text{if } u \leq x_0; \\ \displaystyle\int_A K_s(u, x)\ dx & \text{if } u > x_0. \end{cases} \qquad (20)$$

Then

$$(T\nu)(A) := \int_{\mathbb{R}} P_s(u, A)\nu(\ du) \qquad (21)$$

is a Markov transformation.

Theorem 10 *Let $x_0 \in \mathbf{R}$ and natural number $s \geq 1$ be fixed. Then the Markov transformation (21) $\mu = T\nu$ defines a 1-1 correspondence between the set m_* of all probability measures ν on $\mathbf{R}$ and the set $m_s(x_0)$ of all probability measures μ on $\mathbf{R}$ as in Definition 8. In fact T is a homeomorphism given that m^* and $m_s(x_0)$ are endowed with the weak*-topology.*

Let ϕ: $\mathbf{R} \to \mathbf{R}$ be a bounded and continuous function. Introducing

$$\phi^*(u) := (T\phi)(u) := \int_{\mathbb{R}} \phi(x) \cdot P_s(u, dx), \qquad (22)$$

then

$$\int \phi d\mu = \int \phi^*\ d\nu. \qquad (23)$$

Here ϕ^* is a bounded and continuous function from $\mathbf{R}$ into itself.

We obtain that

$$\phi^*(u) = \begin{cases} \phi(u) & \text{if } u \leq x_0; \\ \displaystyle\int_0^1 \phi((1-t)u + tx_0)st^{s-1}\ dt & \\ \qquad \text{if } u > x_0. & \end{cases} \qquad (24)$$

In particular

$$\frac{1}{s!}(u-x_0)^s\phi^*(u) = \frac{1}{(s-1)!}\int_{x_0}^{u}(u-x)^{s-1}\phi(x)\,dx. \tag{25}$$

Especially, if $r > -1$ we get for $\phi(u) := (u-x_0)^r$ that $\phi^*(u) = \binom{r+s}{s}^{-1}(u-x_0)^r$, for all $u > x_0$. Here $r\,! := 1 \cdot 2 \cdots r$ and

$$\binom{r+s}{s} := \frac{(r+1)\cdots(r+s)}{s!}.$$

Solving the Convex Moment Problem

Let T be the Markov transformation (21) as described above. For each $\mu \in m_s(x_0)$ corresponds exactly one $\nu \in m^*$ such that $\mu = T\nu$. Call $g_i^* := Tg_i$, $i = 1, \ldots, n$ and $h^* := Th$. We have

$$\int_{\mathbb{R}} g_i^*\,d\mu = \int_{\mathbb{R}} g_i\,d\mu$$

and

$$\int_{\mathbb{R}} h^*\,d\nu = \int_{\mathbb{R}} h\,d\mu.$$

Notice that we get

$$\nu(g_i^*) := \int_{\mathbb{R}} g_i^*\,d\nu = c_i; \quad i = 1, \ldots, n. \tag{26}$$

From (15), (16) we get that

$$\int_{\mathbb{R}} T|g_i|\,d\nu < +\infty, \quad i = 1, \ldots, n,$$

and

$$\int_{\mathbb{R}} T|h|\,d\nu < +\infty. \tag{27}$$

Since T is a positive linear operator we obtain $|Tg_i| \le T|g_i|$, $i = 1, \ldots, n$, and $|Th| \le T|h|$, i. e.

$$\int_{\mathbb{R}} |g_i^*|\,d\nu < +\infty, \quad i = 1, \ldots, n,$$

and

$$\int_{\mathbb{R}} |h^*|\,d\nu < +\infty.$$

That is, g_i^*, h^* are ν-integrable.

Finally

$$L(c) = \inf_{\nu} \nu(h^*) \tag{28}$$

and

$$U(c) = \sup_{\nu} \nu(h^*), \tag{29}$$

where $\nu \in m^*$ (probability measure on **R**) such that (26) and (27) are true.

Thus the convex moment problem is solved as a standard moment problem (see the first section).

Remark 11 Here we restrict our probability measures on $[0, +\infty)$ and we consider the case $x_0 = 0$. That is $\mu \in m_s(0)$, $s \ge 1$, i. e. $(-1)^s F^{(s-1)}(x)$ is convex for all $x > 0$ but $\mu(\{0\}) = \nu(\{0\})$ can be positive, $\nu \in m^*$. We have

$$\phi^*(u) = su^{-s} \cdot \int_0^u (u-x)^{s-1} \cdot \phi(x) \cdot dx, \quad u > 0. \tag{30}$$

Further $\phi^*(0) = \phi(0)$, $(\phi^* = T\phi)$. Especially,

$$\text{if} \quad \phi(x) = x^r \quad \text{then} \quad \phi^*(u) = \binom{r+s}{s}^{-1} \cdot u^r, \quad (r \ge 0). \tag{31}$$

Hence the moment

$$\alpha_r := \int_0^{+\infty} x^r \mu(\,dx) \tag{32}$$

is also expressed as

$$\alpha_r = \binom{r+s}{s}^{-1} \cdot \beta_r, \tag{33}$$

where

$$\beta_r := \int_0^{+\infty} u^r\,\nu(du). \tag{34}$$

Recall that $T\nu = \mu$, where ν can be any probability measure on $[0, +\infty)$.

Here we restrict our probability measures on $[0, b]$, $b > 0$ and again we consider the case $x_0 = 0$. Let $\mu \in$

$m_s(0)$ and

$$\int_{[0,b]} x^r \, \mu(dx) := \alpha_r , \tag{35}$$

where $s \geq 1$, $r > 0$ are fixed.

Also let ν be a probability measure on $[0, b]$ unrestricted, i. e. $\nu \in m^*$. Then $\beta_r = \binom{r+s}{s} \alpha_r$, where

$$\beta_r := \int_{[0,b]} u^r \, \nu(du) . \tag{36}$$

Let h: $[0, b] \to \mathbf{R}_+$ be an integrable function with respect to Lebesgue measure. Consider $\mu \in m_s\,(0)$ such that

$$\int_{[0,b]} h \, d\mu < +\infty . \tag{37}$$

i. e.

$$\int_{[0,b]} h^* \, d\nu < +\infty , \quad \nu \in m^* . \tag{38}$$

Here $h^* = Th$, $\mu = T\nu$ and

$$\int_{[0,b]} h \, d\mu = \int_{[0,b]} h^* \, d\nu .$$

Letting α_r be free, we have that the set of all possible $(\alpha_r, \mu(h)) = (\mu(x^r), \mu(h))$ coincides with the set of all

$$\left(\binom{r+s}{s}^{-1} \cdot \beta_r , \nu(h^*) \right) = \left(\binom{r+s}{s}^{-1} \cdot \nu(u^r), \nu(h^*) \right) ,$$

where μ as in (37) and ν as in (38), both probability measures on $[0, b]$. Hence, the set of all possible pairs $(\beta_r, \mu(h)) = (\beta_r, \nu(h^*))$ is precisely the convex hull of the curve

$$\Gamma := \{(u^r, h^*(u)) \colon \; 0 \leq u \leq b\} . \tag{39}$$

In order one to determine $L(\alpha_r)$ the infimum of all $\mu(h)$, where μ is as in (35) and (37), one must determine the lowest point in this convex hull which is on the vertical through $(\beta_r, 0)$. For $\mathrm{U}(\alpha_r)$ the supremum of all $\mu(h)$, μ as above, one must determine the highest point of above convex hull which is on the vertical through $(\beta_r, 0)$.

For more on the above see again §1.

Infinite Many Conditions Moment Problem

See also [16].

Definition 13 A finite nonnegative measure μ on a compact and Hausdorff space S is said to be *inner regular* when

$$\mu(B) = \sup \{\mu(K) \colon \; K \subseteq B; \; K \text{ compact}\} \tag{40}$$

holds for each Borel subset B of S.

Theorem 14 *See [16]. Let S be a compact Hausdorff topological space and a_i: $S \to \mathbf{R}(i \in I)$ continuous functions (I is an index set of arbitrary cardinality), also let α_i ($i \in I$) be an associated set of real constants. Call $M_0(S)$ the set of finite nonnegative inner regular measures μ on S which satisfy the moment conditions*

$$\mu(a_i) = \int_S a_i(s) \, \mu(ds) \leq \alpha_i , \quad \text{all} \quad i \in I . \tag{41}$$

Also consider the function b: $S \to \mathbf{R}$ which is continuous and assume that there exist numbers $d_i \geq 0$ ($i \in I$), all but finitely many equal to zero, and further a number $q \geq 0$ such that

$$1 \leq \sum_{i \in I} d_i a_i(s) - q b(s) , \quad \text{all} \quad s \in S . \tag{42}$$

Finally assume that $M_0(S) \neq \emptyset$ and call

$$U_0(b) = \sup \{\mu(b) \colon \; \mu \in M_0(S)\} . \tag{43}$$

($\mu(b) := \int_S b(s) \, \mu(ds)$). Then

$$U_0(b) = \inf \left\{ \sum_{i \in I} c_i \alpha_i \colon \; \begin{array}{c} c_i \geq 0; \\ b(s) \leq \sum_{i \in I} c_i a_i(s) \text{ all } s \in S \end{array} \right\} , \tag{44}$$

here all but finitely many c_i, $i \in I$, are equal to zero. Moreover, $U_0(b)$ is finite and the above supremum is assumed.

Remark 15 In general we have: let S be a fixed measurable space such that each 1-point set $\{s\}$ is measurable. Further let $M_0(S)$ denote a fixed nonempty set of finite nonnegative measures on S.

For f: $S \to \mathbf{R}$ a measurable function we denote

$$L_0(f) := L(f, M_0(S)) := \inf \left\{ \int_S f(s) \, \mu(ds) \colon \; \mu \in M_0(S) \right\} . \tag{45}$$

Then we have

$$L_0(f) = -U_0(-f). \qquad (46)$$

Now one can apply Theorem 14 in its setting to find $L_0(f)$.

Applications and Discussion

The above described moment theory optimization methods have a lot of applications in many sciences. To mention a few of them: physics, chemistry, statistics, stochastic processes and probability, functional analysis in mathematics, medicine, material science, etc. Optimization moment theory could be also considered the theoretical part of linear finite or semi-infinite programming (here we consider discretized finite nonnegative measures).

The above described methods have in particular important applications: in the marginal moment problems and the related transportation problems, also in the quadratic moment problem, see [17].

Other important applications are in tomography, crystallography, queueing theory, rounding problem in political science, and martingale inequalities in probability. At last, but not least, optimization moment theory has important applications in estimating the speeds: of the convergence of a sequence of positive linear operators to the unit operator, and of the weak convergence of nonnegative finite measures to the unit-Dirac measure at a real number, for that and the solutions of many other important optimal moment problems please see [2].

Final Conclusion

Optimization moment theory is a very active area of mathematical probability theory with a lot of applications in other subjects, and with a lot of researchers from around the world in it contributing new useful results, continuously during all of the 20th century.

See also

- ▶ Approximation of Extremum Problems with Probability Functionals
- ▶ Approximation of Multivariate Probability Integrals
- ▶ Discretely Distributed Stochastic Programs: Descent Directions and Efficient Points
- ▶ Extremum Problems with Probability Functions: Kernel Type Solution Methods
- ▶ Logconcave Measures, Logconvexity
- ▶ Logconcavity of Discrete Distributions
- ▶ L-shaped Method for Two-stage Stochastic Programs with Recourse
- ▶ Multistage Stochastic Programming: Barycentric Approximation
- ▶ Preprocessing in Stochastic Programming
- ▶ Probabilistic Constrained Linear Programming: Duality Theory
- ▶ Probabilistic Constrained Problems: Convexity Theory
- ▶ Simple Recourse Problem: Dual Method
- ▶ Simple Recourse Problem: Primal Method
- ▶ Stabilization of Cutting Plane Algorithms for Stochastic Linear Programming Problems
- ▶ Static Stochastic Programming Models
- ▶ Static Stochastic Programming Models: Conditional Expectations
- ▶ Stochastic Integer Programming: Continuity, Stability, Rates of Convergence
- ▶ Stochastic Integer Programs
- ▶ Stochastic Linear Programming: Decomposition and Cutting Planes
- ▶ Stochastic Linear Programs with Recourse and Arbitrary Multivariate Distributions
- ▶ Stochastic Network Problems: Massively Parallel Solution
- ▶ Stochastic Programming: Minimax Approach
- ▶ Stochastic Programming Models: Random Objective
- ▶ Stochastic Programming: Nonanticipativity and Lagrange Multipliers
- ▶ Stochastic Programming with Simple Integer Recourse
- ▶ Stochastic Programs with Recourse: Upper Bounds
- ▶ Stochastic Quasigradient Methods in Minimax Problems
- ▶ Stochastic Vehicle Routing Problems
- ▶ Two-stage Stochastic Programming: Quasigradient Method
- ▶ Two-stage Stochastic Programs with Recourse

References

1. Akhiezer NI (1965) The classical moment problem. Hafner, New York

2. Anastassiou GA (1993) Moments in probability and approximation theory. Res Notes Math, vol 287. Pitman, Boston, MA
3. Anastassiou GA, Rachev ST (1992) How precise is the approximation of a random queue by means of deterministic queueing models. Comput Math Appl 24(8-9):229–246
4. Anastassiou GA, Rachev ST (1992) Moment problems and their applications to characterization of stochastic processes, queueing theory, and rounding problems. In: Anastassiou G (ed) Proc. 6th S.E.A. Meeting, Approximation Theory. 1–77 M. Dekker, New York
5. Benes V, Stepan J (eds) (1997) Distributions with given marginals and moment problems. Kluwer, Dordrecht
6. Isii K (1960) The extreme of probability determined by generalized moments (I): bounded random variables. Ann Inst Math Statist 12:119–133
7. Johnson NL, Rogers CA (1951) The moment problems for unimodal distributions. Ann Math Stat 22:433–439
8. Karlin S, Shapley LS (1953) Geometry of moment spaces. Memoirs, vol 12. Amer Math. Soc., Providence, RI
9. Karlin S, Studden WJ (1966) Tchebycheff systems: with applications in analysis and statistics. Interscience, New York
10. Kellerer HG (1964) Verteilungsfunktionen mit gegebenen Marginalverteilungen. Z Wahrscheinlichkeitsth Verw Gebiete 3:247–270
11. Kellerer HG (1984) Duality theorems for marginal problems. Z Wahrscheinlichkeitsth Verw Gebiete 67:399–432
12. Kemperman JHB (1965) On the sharpness of Tchebycheff type inequalities. Indagationes Mathematicae 27:554–601
13. Kemperman JHB (1968) The general moment problem, a geometric approach. Ann MathStat 39:93–122
14. Kemperman JHB (1971) Moment problems with convexity conditions. In: Rustagi JS (ed) Optimizing Methods in Statistics. Acad. Press, New York, pp 115–178
15. Kemperman JHB (1972) On a class of moment problems. Proc. Sixth Berkeley Symp. Math. Stat. Prob. 2, pp 101–126
16. Kemperman JHB (1983) On the role of duality in the theory of moments. In: Fiacco AV, Kortanek KO (eds) Semi-Infinite Programming and Applications. of Lecture Notes Economics and Math Systems. Springer, Berlin, pp 63–92
17. Kemperman JHB (1987) Geometry of the moment problem. Moments in Math., of In: Short Course Ser, San Antonio, Texas, 1986, vol 34. Amer. Math. Soc., Providence, RI), pp 20–22
18. Krein MG (1959) The ideas of P.L. Cebysev and A.A. Markov in the theory of limiting values of integrals and their further development. AmerMathSoc Transl 2(12):1–121. ((1951) Uspekhi Mat Nauk 6:3–130)
19. Krein MG, Nudel'man AA (1977) The Markov moment problem and extremal problems. Amer. Math. Soc., Providence, RI
20. Markov A (1884) On certain applications of algebraic continued fractions. Thesis Univ St Petersburg
21. Mises R von (1939) The limits of a distribution function if two expected values are given. Ann Math Stat 10:99–104
22. Mulholland HP, Rogers CA (1958) Representation theorems for distribution functions. Proc London Math Soc 8:177–223
23. Richter H (1957) Parameterfreie Abschätzung und Realisierung von Erwartungswerten. Blätter Deutschen Gesellschaft Versicherungsmath 3:147–161
24. Riesz F (1911) Sur certaines systèmes singuliers d'équations intégrales. Ann Sci Ecole Norm Sup 28:33–62
25. Rogosinsky WW (1958) Moments of non-negative mass. Proc Royal Soc London Ser A 245:1–27
26. Rogosinsky WW (1962) Non-negative linear functionals, moment problems, and extremum problems in polynomial spaces. Stud. Math. Anal. and Related Topics. Stanford Univ Press, Palo Alto, CA, pp 316–324
27. Selberg HL (1940) Zwei Ungleichungen zur Ergänzung des Tchebycheffschen Lemmas. Skand Aktuarietidskrift 23:121–125
28. Shohat JA, Tamarkin JD (1983) The problem of moments. Math Surveys, vol 1. Amer. Math. Soc., Providence, RI
29. Shortt RM (1983) Strassen's marginal problem in two or more dimensions. Z Wahrscheinlichkeitsth Verw Gebiete 64:313–325

General Routing Problem

GRP

Richard Eglese, Adam Letchford
Lancaster University, Lancaster, UK

MSC2000: 90B20

Article Outline

Keywords

Routing

The *general routing problem* (GRP) is a routing problem defined on a graph or network where a minimum cost tour is to be found and where the route must include visiting certain required vertices and traversing certain required edges. More formally, given a connected, undirected graph G with vertex set V and (undirected) edge set E, a cost c_e for traversing each edge e

$\in E$, a set $V_R \subseteq V$ of *required vertices* and a set $E_R \subseteq E$ of *required edges*, the GRP is the problem of finding a minimum cost vehicle route, starting and finishing at the same vertex, passing through each $v \in V_R$ and each $e \in E_R$ at least once ([13]).

The GRP contains a number of other routing problems as special cases. When $E_R = \emptyset$, the GRP reduces to the *Steiner graphical traveling salesman problem* (SGTSP) ([4]), also called the *road traveling salesman problem* in [7]. On the other hand, when $V_R = \emptyset$, the GRP reduces to the *rural postman problem* (RPP) ([13]). When $V_R = V$, the SGTSP in turn reduces to the *graphical traveling salesman problem* or GTSP ([4]). Similarly, when $E_R = E$, the RPP reduces to the *Chinese postman problem* or CPP ([5,8]).

The CPP can be solved optimally in polynomial time by reduction to a matching problem ([6]), but the RPP, GTSP, SGTSP and GRP are all NP-hard. This means that the computational effort to solve such a problem increases exponentially with the size of the problem. Therefore exact algorithms are only practical for a GRP if it is not too large, otherwise a heuristic algorithm is appropriate. The GRP was proved to be NP-hard in [10].

In [3], an integer programming formulation of the GRP is given, along with several classes of valid inequalities which induce facets of the associated polyhedra under mild conditions. Another class of valid inequalities for the GRP is introduced in [11] and in [12] it is shown how to convert facets of the GTSP polyhedron into valid inequalities for the GRP polyhedron. These valid inequalities form the basis for a promising branch and cut style of algorithm described in [2] which can solve GRPs of moderate size to optimality.

In [9], a heuristic algorithm for the GRP is described. The author adapts Christofides' heuristic for the TSP to show that when the triangle inequality holds in the graph, the heuristic has a worst-case ratio of heuristic solution value to optimum value of 1.5.

There are many vehicle routing applications of the GRP. In these cases, the edges of the graph are used to represent streets or roads and the vertices represent road junctions or particular locations on a map. In any practical application there are likely to be many additional constraints which must also be taken into account such as the capacity of the vehicles, time-window constraints for when the service may be carried out, the existence of one-way streets and prohibited turns etc.

Many applications are for the special cases when either $E_R = \emptyset$ or $V_R = \emptyset$. However, there are some types of vehicle routing applications where the problem is most naturally modeled as a GRP with both required edges and required vertices. For example, in designing routes for solid waste collection services, collecting waste from all houses along a street could be modeled as a required edge and collecting waste from the foot of a multistory apartment block could be modeled as a required vertex. Other examples include postal delivery services where some customers with heavy demand might be modeled as required vertices, while other customers with homes in the same street might be modeled together as a required edge. School bus services are other examples of GRPs where a pick-up in a remote village could be modeled as a required vertex, but if the school bus must pick-up at some point along a street (and is not allowed to perform a U-turn in the street) then that may best be modeled as a required edge.

Further details about solution methods and applications for various network routing problems can be found in [1].

See also

- ▶ Stochastic Vehicle Routing Problems
- ▶ Vehicle Routing
- ▶ Vehicle Scheduling

References

1. Ball MO, Magnanti TL, Monma CL, Nemhauser GL (eds) (1995) Network routing. vol 8, Handbook Oper. Res. and Management Sci. North-Holland, Amsterdam
2. Corberáan A, Letchford AN, Sanchis JM (1998) A cutting-plane algorithm for the general routing problem. Working Paper
3. Corberáan A, Sanchis JM (1998) The general routing problem polyhedron: Facets from the RPP and GTSP polyhedra. Europ J Oper Res 108:538–550
4. Cornué;jols G, Fonlupt J, Naddef D (1985) The travelling salesman problem on a graph and some related integer polyhedra. Math Program 33:1–27
5. Edmonds J (1963) The Chinese postman problem. Oper Res 13:B73–B77
6. Edmonds J, Johnson EL (1973) Matchings, Euler tours and the Chinese postman. Math Program 5:88–124

7. Fleischmann B (1985) A cutting-plane procedure for the travelling salesman problem on a road network. Europ J Oper Res 21:307–317
8. Guan M (1962) Graphic programming using odd or even points. Chinese Math 1:237–277
9. Jansen K (1992) An approximation algorithm for the general routing problem. Inform Process Lett 41:333–339
10. Lenstra JK, Rinnooy Kan AHG (1976) On general routing problems. Networks 6:273–280
11. Letchford AN (1997) New inequalities for the general routing problem. Europ J Oper Res 96:317–322
12. Letchford AN (1999) The general routing polyhedron: A unifying framework. Europ J Oper Res 112:122–133
13. Orloff CS (1974) A fundamental problem in vehicle routing. Networks 4:35–64

Genetic Algorithms

GA

Richard S. Judson
Genaissance Pharmaceuticals, New Haven, USA

MSC2000: 92B05

Article Outline

Keywords

Optimization; Genetic algorithms; Evolution; Stochastic global optimization; Population; Fitness; Crossover; Mutation; Binary encoding; Individual; Chromosome; Generation; Elitism; Premature convergence; Gray code; Random walk search; Roulette wheel procedure; Population size; Schema theorem; Schema; Local minimum; Selection; Evolution strategy

Genetic algorithms (GAs) comprise a class of *stochasticglobal optimization* methods based on several strategies from biological evolution. The basic genetic algorithm was developed by J.H. Holland and his students ([5,6,7,8]), and was based on the observation that selection (either natural or artificial) can produce highly optimized individuals in a relatively short number of generations. This is true despite the fact that the space of all gene mutations through which a population must sort is astronomical. For instancethe genome of the yeast *Saccharomyces cerevisiae*, which is the simplest eukaryote, contains just over 6000 genes, each of which can occur in several mutant forms. Despite this, *S. cerevisiae* can reoptimize itself to survive and flourish in many new environments in a relatively short number of generations. This is equivalent to having a computer search for a near-optimal solution to a 6000-dimensional problem where each of the 6000 variables can take on any one of a large number of values.

The most important notion from natural systems that the GA employs is the use of a *population* of individuals which go through a *selection* step to produce offspring and pass on their genetic material.Optimality or *fitness* is measured by how many offspring an individual produces. A second notion is the use of *crossover* in which individuals share genetic information and pass the shared information onto their offspring. A third borrowing from nature is the idea of *mutation*, the consequence of which is that the transfer of genetic informationis prone to random errors. This helps maintain the level of genetic diversity in a population.

The implementation of a simple GA (SGA) which uses these ideas is straightforward. The description that follows uses a *binary encoding*, but all of the ideas follow identically for integer or even real number encodings. The most important idea is that one works with a population of *individuals* which will interact through genetic operators to carry out an optimization process. An individual is specified by a *chromosome* C which is a bit string of length N_c that can be decoded to give a set of N parameters x_i which are the natural parameters for the optimization application. Each parameter x_i is encoded by n_i bits so that $\sum_i^N n_i = N_c$. In what follows, chromosome and bit string are synonymous. A fitness function $f(x_1, \ldots, x_N)$, which is the function to be optimized, is used to rank the individual chromosomes. An initial population of N_{pop} individuals is formed by choosing N_{pop} bit strings at random, and evaluating each individual's fitness. (Decode $C \rightarrow (x_1, \ldots, x_N)$, calculate $f(x_1, \ldots, x_N)$.)Subsequent *generations* are formed as follows. All parents (members of the current generation) are ranked by fitness and the highest fitness individual is placed directly into the next generation with no change. (This step of keeping the most-fit individual intact is termed *elitism* and is a purely heuristic addition. It insures that good solutions to the problem at hand are not lost until better ones are found.) Next, pairs of parents are selected and their chromosomes are crossed over to form chromosomes of the remaining individuals in the next generation. A parent's probability of being selected increases with its fitness. So for a minimization application, the parent with the current lowest value of $f(x_1,\ldots,x_N)$ has the highest chance of being selected for mating. Crossover consists of taking some subset of the bits from parent 1 and the complementary set of bits from parent 2 and combining them to form the chromosome of child 1. A childis simply a member of the next generation. The remaining bits from the two parents are combined to form the chromosome of child 2. Additionally, during replication there is a small probability of a bit flip or mutation in a chromosome. This serves primarily to maintain diversity and prevent *premature convergence*. Convergence occurs when the population becomes largely homogeneous – most individuals have almost the same values for all of their parameters. Premature convergence occurs when the population converges early in a run, before significant amount of searching has been performed. The most common cause is a poor choice of the scaling of the fitness function. It should be noted that 'premature' and 'early' are loosely defined. To bound the magnitude of the effect of mutations, the binary chromosomes are usually *Gray coded*. An integer that is represented as a Gray coded binary number has the property that most single bit flips change the value of the decimal integer represented by the chromosome by ± 1. In sum, the algorithm consists of successively transforming one generation of individuals into the next using the operations of selection, crossover and mutation. Since the selection process is biased towards individuals with higher fitness, individuals are produced that come ever closer to being optimal solutions to the function of interest.

It is important to emphasize that crossover is the key feature that distinguishes the GA from other stochastic global search methods. If crossover is ineffective, GA degenerates into a *random walk search* being executed separately by each individual in the population. The random walk is generated by the mutation operator.

The GA is presented below as pseudocode:

```
PROCEDURE genetic algorithm()
    Initialize population;
    FOR ($g$ = 1 to $N_{\text{gen}}$ generations) DO
        FOR ($i$ = 1 to $N_{\text{pop}}$ individuals) DO
            Evaluate fitness of individual $i$: $f_i(g)$:
        END FOR;
        Save best individual to population $g + 1$;
        FOR ($i$ = 2 to $N_{\text{pop}}$) DO
            Select 2 individuals;
            Crossover: create 2 new individuals;
            Mutate the new individuals;
            Move new individuals to population g+1;
        END FOR;
    END FOR;
END genetic algorithm;
```

Pseudocode for the Simple Genetic Algorithm

Selection commonly uses a *roulette wheel procedure*. Each individual is assigned a slice of the unit circle proportional to its fitness $(f(x_1,\ldots,x_N))$.One then chooses pairs of random numbers to select the next two individuals to be mated. A typical crossover operator takes the chromosomes from apair of individuals and chooses a common cut point along them. One child gets the portion of the first parent's chromosome to the left of thecut point, and the portion of the second parent's chromosome to the right of the cut point. The chromosome of the second child is comprised of the remaining fragments of the two parent chromosomes. In the most common mutation operator each bit in the binary chromosome has an equal and low probability being flipped from 1 to 0 or vice versa.Many variants on these operators have been used.

The important variables in the GA method are the *population size*, N_{pop}, the total number of generations allowed, N_{gen}, the number of bits used to represent a real variable, and the mutation rate. The total CPU time used in an optimization run is proportional to $N_{\text{pop}} \times N_{\text{gen}} \times T(f)$, where $T(f)$ is the time required to evaluate the fitness function $f(x_1, \ldots, x_N)$. This leads to a trade-off between having large, diverse populations that explore parameter space widely, and having smaller populations that explore longer. In practice, the choice is problem dependent.

The simple GA and a large number of variants have been successfullyused to find near-optimal solutions to many engineering and scientific applications. ([2,3,4,6,9,10,11]) Although much effort has gone into formally analyzing the GA to understand why it is so robust, the most important formal result is the *Schema theorem* ([6,7,8]). *Schemata* are strings made up of the characters 1, 0 and $*$ which is the 'don't care' character. These schemata are building blocks out of which the strings representing individuals' chromosomes can be constructed. For instance the string 11100 contains schema such as 111, 1100 and $1*10$. The schema theorem provides a powerful statement about the behavior of schemata in a chromosome. Mathematically, it states

$$\begin{aligned} &m(H, g+1) \\ &\quad \geq m(H, g)\frac{f(H)}{\overline{f}}\left(1 - p_c\frac{\delta(H)}{l-1} - p_m\frac{o(H)}{p_m}\right), \end{aligned} \qquad (1)$$

where $m(H, g)$ is the number of examples of a schema H that exist in the population at generation g; $f(H)$ is the average fitness of chromosomes containing H; $\overline{f}$ is the average fitness of all chromosomes; p_c is the probability that crossover will occur at a particular mating; p_m is the probability that a particular bit will be mutated; l is the length of the chromosome; $\delta(H)$ is the length of the schema in bits; and o(H) is the order of the schema, defined to be the number of fixed (as opposed to don't care) positions in the schema.

The factors outside the brackets in (1) indicate that a particular schema will increase its representation in the population at a rate proportional to its fitness relative to the average fitness. Good schemata will increase their representation exponentially and bad schemata will decrease their representation likewise. The terms inside the bracket serve to decrease this exponential convergence by disrupting the selection-based pressure. Both crossover and mutation can disrupt good schemata. The longer a schema is, the more likely it is to be disrupted by crossover, and disappear from the population. In the same fashion, schemata with many fixed positions are more likely to be disrupted by mutations.

The competition between selection which drives the population towards convergence on a good solution and crossover and mutation which drive the population towards more diverse states are the keys to the GA. Crossover is especially important for keeping the method from being trapped in *local minima*. One consequence of the parameter shuffling brought about by the crossover operator is that the GA is most efficient at optimizing functions that are at least partially separable. One individual can find a state where half of the parameters of the fitness function are optimized and a second individual can find a state where the other half are optimized. If these individuals crossover at the correct point, one of theirchildren will have the parameter values that globally optimize the function.

As with most other heuristic global optimization methods, no definitive statements can be made about the global optimality of GA-generated solutions.

A family of algorithms that are very similar to the GA, called *evolution strategies* were developed independently and virtually simultaneously in Germany by I. Rechenberg ([1,12]).

See also

- ▶ Adaptive Simulated Annealing and its Application to Protein Folding
- ▶ Genetic Algorithms for Protein Structure Prediction
- ▶ Global Optimization in Lennard–Jones and Morse Clusters
- ▶ Global Optimization in Protein Folding
- ▶ Molecular Structure Determination: Convex Global Underestimation
- ▶ Monte-Carlo Simulated Annealing in Protein Folding
- ▶ Multiple Minima Problem in Protein Folding: αBB Global Optimization Approach
- ▶ Packet Annealing
- ▶ Phase Problem in X-ray Crystallography: Shake and Bake Approach
- ▶ Protein Folding: Generalized-ensemble Algorithms
- ▶ Simulated Annealing
- ▶ Simulated Annealing Methods in Protein Folding

References

1. Bäck T, Schwefel H-P (1993) An overview of evolutionalgorithms for parameter optimization. Evolutionary Computation 1(1)
2. Belew RK, Booker LB (eds) (1991) Proc. fourth Internat. Conf. Genetic Algorithms. Morgan Kaufmann, San Mateo
3. Davis L (ed) (1987) Genetic algorithms and simulated annealing. Pitman, Boston

4. Davis L (1991) Handbook of genetic algorithms. v. Nostrand Reinhold, Princeton, NJ
5. DeJong K (1976) An analysis of the behavior of a class of genetic adaptive systems. PhD Thesis Univ. Michigan
6. Goldberg D (1989) Genetic algorithms in search, optimization and learning. Addison-Wesley, Reading
7. Holland JH (1992) Adaptation in natural and artificial systems. MIT, Cambridge
8. Holland JH (1992) Genetic algorithms. Scientif Amer 267:66
9. Judson RS (1997) Genetic algorithms and their use in chemistry. In: Lipkowitz KB, Boyd DB (eds) Rev. Computational Chemistry, vol 10. Wiley-VCH, Weinheim, pp 1–73
10. Koza J (1992) Genetic programming. MIT, Cambridge
11. Rawlins GJE (1991) Foundations of genetic algoritms. Morgan Kaufmann, San Mateo
12. Rechenberg I (1973) Evolutionsstrategie – Optimierung technischer Systeme nach Prinzipien der biologischen Evolution. Frommann-Holzboog, Stuttgart-Bad Cannstatt

Genetic Algorithms for Protein Structure Prediction

RICHARD S. JUDSON
Genaissance Pharmaceuticals, New Haven, USA

MSC2000: 92B05

Article Outline

Keywords

Optimization; Evolution; Protein structure; Amino acid; Active site; Free energy; Conformation; Configuration; Primary structure; Tertiary structure; Cartesian coordinates; Internal coordinates; Bond distance; Bond angle; Dihedral angle; Rotamer library; Rotamer; Empirical potential; Nonbonded distance; Secondary structure

Genetic algorithms (GAs; cf. also ► Genetic algorithms) have been used for a large number of modeling applications in chemical and biological fields [5,9]. At least three factors contribute to this. First, GAs provide an easy-to-use global search and optimization approach. Second, they can easily handle noncontinuous functions. Finally, they are relatively robust even for moderately high-dimensional problems. All of these have contributed to the use of the GA for the important but computationally demanding field of protein structure prediction.

Proteins carry out a wide variety of functions in living cells, almost all of which require that the protein molecules assume precise 3-dimensional shapes [2,3]. Enzymes are typical examples. They generally consist of a large structure of 100–300 *amino acids* stabilizing a small *active site* which is designed to carry out a specific chemical reaction such as cleaving a bond in a target molecule. Even slight changes in the structure of the active site can destroy the protein's ability to function. Many drugs act by fitting snugly into enzymes' active sites, causing them to shut down. Therefore, a detailed understanding of the 3-dimensional structure of a protein can enhance our understanding of its function. This can in turn help understand related disease processes and can finally lead to disease cures. Unfortunately the experimental determination of protein structures, using *x*-ray crystallography or solution NMR is very difficult. Currently the structures of only a few thousand of the estimated 100,000 proteins that are used by the human body have been determined this way. The alternative is to predict the structures computationally.

The basic computational approach is simple to state, although many details have yet to be worked out. It relies on the experimental fact that a protein in solution (as well as any other molecule) will tend to find a state of low *free energy*. Free energy accounts for the internal energy (potential plus kinetic) of single molecules as well as the entropy of the ensemble of molecules of the same type. At absolute zero, the entropy contribution to the energy, as well as the kinetic energy, go to zero, leaving only the potential energy. Therefore, the most likely shape or state of a protein at absolute zero is the one of lowest potential energy. The simplest computational model then needs a method to search the space of conformations and an energy function (approximating the physical potential energy) which is minimized during the search. (A protein's *conformation* is the description of the 3-dimensional positions of all of the atoms for a fixed set of atoms and atom-atom connections. The *configuration* describes the atom-atom connectivity and only changes through chemical bond forming or breaking.) The conformation which yields the lowest value of the energy function is a best estimate of conformation of the natural protein. It is possible to extend this simple model to include the effects of finite temperature, but these extensions are beyond the scope of this article. In-depth discussions of molecular modeling, including energy functions for proteins and other molecules can be found in [6,8,10], and [1].

Because proteins possess many degrees of freedom, and the energy functions have many local minima, global optimization methods that search efficiently and are not prone to being caught in local minima are required. The GA is often used because it fits both of these criteria.

Proteins [2] are long linear polymers composed of well-conserved sequences of the 20 amino acids. Each amino acid is in turn made up of a backbone

$$- \quad (\mathrm{NH} \quad - \quad \overset{\overset{\displaystyle R}{|}}{\mathrm{C}_\alpha} \quad - \quad \mathrm{CO}) \quad -$$

where R stands for one of the 20 side groups that make the amino acids unique. These range from a single hydrogen atom to chains having many degrees of freedom. The *primary structure* of the protein is simply the sequence of amino acids. For many naturally occurring proteins, this sequence carries sufficient information to determine the final 3-dimensional or *tertiary structure* of the protein. Experimentally, proteins that have been denatured (caused to unfold by heating the solution or changing its chemical composition) will spontaneously refold to their active, or native conformation, when the solution is returned to its original state.

There are two sets of coordinates often used for specifying the conformation of a protein. The first are the standard *Cartesian coordinates* for each atom. For N atoms, this requires $3N - 6$ numbers. The alternative is to use *internal coordinates* which are the *bond distances* (distances between atoms bound together), the *bond angles* (angles formed by a given atom and two atoms bound to it), and the *dihedral angles* (the angle of rotation about a center bond for a set of 4 atoms bound as $\mathrm{A} - \mathrm{B} - \mathrm{C} - \mathrm{D}$). To a good first approximation, the bond distances and bond angles are fixed at values that are independent of the particular amino acid or protein. Therefore, the conformation of a protein is determined largely by the values of its dihedral angles. There are on average about 15 atoms and about 3 dihedrals per amino acid, requiring about $N/5$ degrees of freedom to describe the conformation of an N-atom protein. The dimension of conformation space for a moderate-size protein of 100 amino acids (≈ 1500 atoms) is ≈ 4500 when using Cartesian coordinates vs. ≈ 300 when using internal coordinates with fixed bond distances and angles.

In many protein structure prediction applications, the simple GA approach is used. For each generation, one calculates the fitness (energy) of each individual in the population, selects pairs of individuals based on their energy, performs crossover and mutation. The GA chromosome directly codes for the values of the dihedral angles. Both binary encoded and real number encoded chromosomes have been used with equal success. For binary encoded dihedrals, one must decide on the resolution of the GA search. The maximum one would use is 10 bits per angle which gives a resolution of about 1/3 degree. Often as few as 5 or 6 bits will be sufficient, especially if the GA-generated conformations will be subjected to local gradient minimization.

For each GA individual, the chromosome is decoded to give the values of the dihedrals which are passed to the energy function. This in turn returns an energy which is used as the fitness for the subsequent selection process.

Another encoding scheme that is often used is based on the idea of a *rotamer library*. It is known from studying the set of experimentally known structures that the dihedral angles in many amino acid side chains take on restricted sets of values. Also, the values of several neighboring dihedrals are often correlated. It has then been possible to develop libraries of preferred sidechain conformations (called *rotamers*) for each amino acid. This can be incorporated into the GA by having each word in the chromosome simply determine which of a set of rotamers to use for each amino acid in the sequence. The use of rotamer libraries in the GA framework is illustrated in references [7,12,13,14], and [11].

The other major ingredient needed for a protein structure prediction method is an energy function to be minimized. This is a huge area of research which is beyond the scope of this article, but two major approaches will be summarized. The first scheme uses physics-based *empirical potentials*. These are functions of the bond distances, bond angles, dihedral angles, and *non-bonded distances* (distances between atoms not directly bound together). The functional forms are derived from the results of accurate but computationally expensive quantum mechanical calculations that are performed on small molecular fragments such as individual amino acids. The results are fitted to simple functions with several free parameters. The parameter values are either taken from the original quantum calculations or from independent spectroscopic experiments. Various methods are used to approximate the effect of the water and salt environment around the protein. The advantage of these potentials is that they are continuous and very general. They can be constructed for any protein and give reasonable energies for any conformation requested. The disadvantage is that they are not yet sufficiently accurate to give reliable structure predictions. For many if not all of the proteins whose structure is known, there are conformations that have much lower calculated energy than that of the experimental conformation.

The second approach is to use potentials based on observations of known protein structures. Basically, more probable conformations (ones that look more like real proteins) will have lower energy values. For instance certain sequences of amino acids almost always assume a particular *secondary structure*. The secondary structure of a protein describes the presence of multi-amino acid helices, sheets and turns but not the exact placement of the atoms in the secondary structure elements or the spatial orientation of these elements. These potentials have the advantage that they build on our observations of proteins as entire molecules and incorporate long-range order. As with the empirical potentials, though, they suffer from accuracy problems. However, except for very small proteins (less than 20 amino acids) the structure-based potentials show the most promise.

A common feature of GA-based protein structure prediction methods is the use of hybrid approaches combining standard GA with a local search method. The GA is then used primarily to perform an efficient global search which is biased towards regions of conformation space with low energy. This is a pragmatic approach driven by the large number of degrees of freedom even when internal coordinates are used. A simple and often used approach [5] is to subject GA-generated conformations to gradient minimization. Another approach is to use a population of individuals which carry out independent Monte-Carlo or simulated annealing walks (cf. also ► Simulated annealing methods in protein folding; ► Monte-Carlo simulated annealing in protein folding) for a number of steps and then undergo selection, crossover and mutation [4,15,16].

See also

► Adaptive Simulated Annealing and its Application to Protein Folding
► Bayesian Global Optimization
► Genetic Algorithms
► Global Optimization Based on Statistical Models
► Monte-Carlo Simulated Annealing in Protein Folding
► Packet Annealing
► Random Search Methods
► Simulated Annealing Methods in Protein Folding
► Stochastic Global Optimization: Stopping Rules
► Stochastic Global Optimization: Two-phase Methods

References

1. Allen MP, Tildesley DJ (1996) Computer simulation of liquids. Oxford Sci. Publ., Oxford
2. Branden C, Tooze J (1991) Introduction to protein structure. Garland Publ., Oxford
3. Creighton TE (1993) Proteins: structure and molecular properties. Freeman, New York
4. Friesner JR, Gunn A, Monge RA, Marshall GH (1994) Hierarchical algorithms for computer modeling of protein tertiary structure: folding of myoglobin to 6.2Å resolution. J Phys Chem 98:702
5. Judson RS (1997) Genetic algorithms and their use in chemistry. In: Lipkowitz KB, Boyd DB (eds) Rev. Computational Chemistry, vol 10. Wiley-VCH, Weinheim, pp 1–73
6. Karplus CL, Brooks M, Pettitt BM (1988) Proteins: A theoretical perspective of dynamics, structure and thermodynamics. Wiley/Interscience, New York
7. LeGrand S, Merz K (1993) The application of the genetic algorithm to the minimization of potential energy functions. J Global Optim 3:49
8. McCammon JA, Harvey S (1987) Dynamics of proteins and nucleic acids. Cambridge Univ. Press, Cambridge
9. Pedersen J, Moult J (1996) Genetic algorithms for protein structure prediction. Curr Opin Struct Biol 227–231
10. Rapaport DC (1995) The art of molecular dynamics simulation. Cambridge Univ. Press, Cambridge
11. Ring CS, Cohen FE (1994) Conformational sampling of loop structures using genetic algorithms. Israel J Chem 34:245

12. Sun S (1993) Reduced representation model of, protein structure prediction: statistical potential and genetic algorithms. Protein Sci 2:762
13. Tuffery P, Etchebest C, Hazout S, Lavery R (1991) A new approach to the rapid determiniation of protein side chain conformations. J Biomol Struct Dynam 8:1267
14. Tuffery P, Etchebest C, Hazout S, Lavery R (1993) A critical comparison of search algorithms applied to the optimization of protein side chain conformations. J Comput Chem 14:790
15. Unger R, Moult J (1993) Effects of mutations on the performance of genetic algorithms suitable for protein folding simulations. In: Tanaka M, Doyoma M, Kihara J, Yamamoto R (eds) Computer-Aided Innovation In New Materials. Elsevier, Amsterdam, pp 1283–1286
16. Unger R, Moult J (1993) Genetic algorithms for protein folding simulations. J Mol Biol 231:638

Geometric Programming

YANJUN WANG
Department of Applied Mathematics, Shanghai University of Finance and Economics, Shanghai, China

MSC2000: 90C28, 90C30

Article Outline

Keywords and Phrases

Generalized geometric programming; Global optimization; Linear relaxation programming; Branch and bound

Introduction

Geometric programming is an important class of nonlinear optimization problems. Their source dates back to the 1960s when Zener began to study a special type of minimization cost problem for design in engineering, now known as geometric programming. The term geometric programming is adopted because of the crucial role that the arithmetic-geometric mean inequality plays in its initial development.

Actually, the early work in geometric programming was, for the most part, concerned with minimizing posynomial functions subject to inequality constraints on such functions, which was called posynomial geometric programming. In the past decade, because a number of models abstracted from application fields were not posynomial geometric programming, the theory had to be generalized to a much broader class of optimization problems called generalized geometric programming, which has spawned a wide variety of applications since its initial development. Its great impact has been in the areas of (1) engineering design [1,4,10,11]; (2) economics and statistics [2,3,6,9]; (3) manufacturing [8,17]; (4) chemical equilibrium [13,16]. Reference [19] focuses on solutions for generalized geometric programming.

Formulation

[19] provides a global optimization algorithm for the generalized geometric programming (GGP) problem stated as:

$$GGP \begin{cases} \min & G_0(x) \\ \text{s.t.} & G_m(x) \leq \delta_m,\ m = 1, \ldots, M \\ & x \in X = \{x\colon 0 < x_i^l \leq x_i \leq x_i^u \\ & i = 1, \ldots, N\} \end{cases}$$

where $G_m(x) = \sum_{t=1}^{T_m} \delta_{mt} c_{mt} \prod_{i=1}^{N} x_i^{\gamma_{mti}}$, $m = 0, 1, \ldots, M$, and c_{mt} are positive coefficients, T_m are the given number of the terms in the function $G_m(x)$, $\delta_{mt} = +1$ and -1; $\delta_m = +1$ or -1, γ_{mti} are arbitrary real constant exponents. In general, formulation GGP corresponds to a nonlinear optimization problem with a nonconvex objective function and constraint set. In $G_m(x)$, if $\delta_{mt} = +1$ for all $t, t = 1, \ldots, T_m$, and $x_i > 0, i = 1, \ldots, N$, then the function $G_m(x)$ is called a posynomial. Note that if we set $\delta_{mt} = +1$ for all $m = 0, 1, \ldots, M, t = 1, \ldots, T_m$ and $\delta_m = +1$ for all $m = 1, \ldots, M$, then the GGP formulation reduces to the classical posynomial geometric programming (PGP) formulation that laid the foundation for the theory of the GGP problem.

Local optimization approaches for solving the GGP problem include three kinds of methods in general. First, successive approximation by posynomials, called "condensation," is the most popular [14]. Second, Passy and Wilde [15] developed a weaker type of duality, called "pseudo-duality," to accommodate this class of nonlinear optimization. Third, some nonlinear programming methods are adopted to solve the GGP problem based on exploiting the characteristics of the GGP problem [12].

Though local optimization methods for solving the GGP problem are ubiquitous, global optimization algorithms based on the characteristics of the GGP problem are scarce. Maranas and Floudas [13] proposed such a global optimization algorithm based on the exponential variable transformation of GGP, the convex relaxation, and branch and bound on some hyperrectangle region. Reference [19] proposes a branch-and-bound optimization algorithm that solves a sequence of linear relaxations over partitioned subsets in order to find a global solution, and to generate the linear relaxation of each subproblem and to ensure convergence to a global solution, special strategies have been applied. (1) The equivalent reverse convex programming (RCP) formulation is considered. (2) A linear relaxation method for the RCP problem is proposed based on the arithmetic-geometric mean inequality and the linear upper bound of the reverse convex constraints; this method is more convenient with respect to computation than the convex relaxation method [13]. (3) A bound tightening method is developed that will enhance the solution procedure, and, based on this method, a branch-and-bound algorithm is proposed.

Methods and Applications

Transformation

In [5], Duffin and Peterson show that any GGP problem can be transformed into the following reverse posynomial geometric programming (RPGP):

$$\begin{cases}\min & x_0\\ \text{s.t.} & g_m(x)\le 1,\quad m=1,\dots,p\\ & g_m(x)\ge 1,\quad m=p+1,\dots,q\\ & x\in\Omega_0=\{x\colon 0<x_i^l\le x_i\le x_i^u<\infty\\ & i=0,\dots,n\}\end{cases}$$

where $g_m(x)$ are posynomials for $m=1,\dots,q$, and $n\ge N$.

To see how such a reformulation is possible, first consider the objective function in GGP. If the optimal value of GGP is positive, the GGP problem is equivalent to the following form:

$$(GGP1)\colon\begin{cases}\min & x_0\\ \text{s.t.} & x_0^{-1}G_0(x)\le 1,\\ & G_m(x)\le\delta_m,\, m=1,\dots,M\\ & x\in X\,.\end{cases}$$

And if the optimal value of GGP is negative, then GGP can be transformed into the following form:

$$(GGP2)\colon\begin{cases}\min & x_0\\ \text{s.t.} & x_0G_0(x)\le -1\,,\\ & G_m(x)\le\delta_m,\, m=1,\dots,M\\ & x\in X\,.\end{cases}$$

We can add a large constant to the objective function of GGP in order to ensure that the optimal value of (GGP) is positive, then derive the form GGP1. In this method a probably lower bound estimation for the optimal value of GGP is needed.

Secondly we turn to consider the constraints. If the primal constrained function $G_m(x)$ is either a posynomial or the negative of a posynomial, then it is obvious. So we only consider the following constrained function:

$$G_m(x)=h_1(x)-h_2(x)\le 1\,,$$

where each $h_i(x)(i=1,2)$ is a posynomial. Notice that x satisfies the above inequality if and only if there exists a single variable $s>0$ such that (x,s) satisfies

$$h_1(x)\le s\le h_2(x)+1\,.$$

Now note that the above formulation is equivalent to the following two constraints

$$s^{-1}h_1(x)\le 1\quad\text{and}\quad s^{-1}h_2(x)+s^{-1}\ge 1\,,$$

which are in a form consistent with the formulation RPGP.

By applying the following exponent transformation

$$x_i=\exp z_i\,,\quad i=0,\dots,n$$

to the formulation *RPGP*, we can obtain the following reverse convex programming (RCP) problem:

$$\begin{cases} \min & \exp(z_0) \\ \text{s.t.} & g_m(z) \le 1,\ m = 1, \dots, p \\ & g_m(z) \ge 1,\ m = p+1, \dots, q \\ & z \in \Omega = \{z\colon z_i^L \le z_i \le z_i^U\ , \\ & i = 0, 1, \dots, n\} \end{cases}$$

where

$$g_m(z) = \sum_{t=1}^{T_m} c_{mt} \exp\left\{\sum_{i=0}^{n} \gamma_{mti} z_i\right\}\ , \quad m = 1, \dots, q$$

Because each $\exp\{\sum_{i=0}^{n} \gamma_{mti} z_i\}$ is convex, both the objective and constrained functions are convex.

The main difficulty for solving the RCP problem is connected with the presence of the reverse convex constraints $g_m(z) \ge 1,\ m = p+1, \dots, q$, which destroy the convexity and possibly even the connectivity of the feasible set and give rise to a nonconvex feasible region.

Linear Relaxation Programming

The principal construct in the development of a solution procedure for solving the RCP problem is the construction of a linear relaxation programming of RCP for obtaining the lower bound for this problem, as well as for its partitioned subproblems [19] derives such a linear relaxation by applying the arithmetic-geometric mean inequality for the convex constraints and overestimating every reverse convex constraint in either the initial bounds on the variables of the problem or modified bounds as defined for some partitioned subproblem in a branch-and-bound scheme.

(1) Linear Relaxation for Convex Constraints The arithmetic-geometric mean inequality that played such a crucial role in developing the duality theory for posynomial programming is also used to obtain linear relaxation programming. Recall that this inequality states that for any vector $\omega > 0$ and any nonnegative weight vector ε whose components sum to one, we have

$$\sum_t \omega_t \ge \prod_t (\frac{\omega_t}{\varepsilon_t})^{\varepsilon_t}$$

provided $(\omega_t/\varepsilon_t)^{\varepsilon_t}$ is defined to be 1 when $\varepsilon_t = 0$. Give a posynomial

$$g_m(x) = \sum_t u_{mt}(x) = \sum_t c_{mt} \prod_i x_i^{mti}$$

and $\varepsilon_m \ge 0$ with $\sum_t \varepsilon_{mt} = 1$. Then a condensed posynomial $\bar{g}_m(x)$ is defined by

$$\bar{g}_m(x) = \bar{c}_m \prod_i x_i^{\bar{\gamma}_{mi}}$$

where $\bar{c}_m = \prod_t (c_{mt}/\varepsilon_{mt})^{\varepsilon_{mt}}$ and $\bar{\gamma}_{mi} = \sum_t \gamma_{mti}\varepsilon_{mt}$.

Thus the condensed posynomial $\bar{g}_m(x)$ is also a posynomial, and it has a single posynomial term. According to this method, the condensed single term for the convex constraints $g_m(z) \le 1$ of RCP, where $z_i = \ln x_i$, is of the following form:

$$\bar{g}_m(z) = \bar{c}_m \exp\left(\sum_i \bar{\gamma}_{mi} z_i\right) \tag{1}$$

where the definitions of $\bar{c}_m$ and $\bar{\gamma}_{mi}$ have been given in the former.

To illustrate how the condensed term can be used to obtain the linear relaxation, we consider the following convex constraints $g_m(z) \le 1, m = 1, \dots, p$ and select an arbitrary weight vector $\varepsilon_m \ge 0$ whose components sum to one. We use the condensed constrained functions to replace the above convex constraints:

$$\bar{g}_m(z) \le 1, \quad m = 1, \dots, p\ . \tag{2}$$

It follows that

$$\bar{g}_m(z) \le g_m(z)$$

for each $m = 1, \dots, p$. Thus if in RCP the convex constraints are replaced by the condensed constraints, the feasible region for RCP will be contained in the new feasible region. Notice that the condensed constraints (2) can be easily transformed into equivalent formulations as linear constraints:

$$L_m(z) = \sum_i \bar{\gamma}_{mi} z_i + \ln \bar{c}_m \le 0,\ m = 1, \dots, p\ .$$

(2) Linear Relaxation for Reverse Convex Constraints For reverse convex constraints such a linear relaxation can be obtained by overestimating every convex function $g_m(z)$ of the reverse convex constraint with a linear function $L_m(z)$ for every $m = p+1, \dots, q$. The method in [13] of underestimating a concave function with a linear function is adopted, and we describe the linear function as follows:

$$L_m(z) = \sum_{t=1}^{T_m} c_{mt} \left\{ A_{mt} + B_{mt} \left(\sum_{i=0}^{n} \gamma_{mti} z_i \right) \right\}$$

and

$$\begin{aligned} A_{mt} &= \frac{Y_{mt}^U \exp(Y_{mt}^L) - Y_{mt}^L \exp(Y_{mt}^U)}{Y_{mt}^U - Y_{mt}^L} , \\ B_{mt} &= \frac{\exp(Y_{mt}^U) - \exp(Y_{mt}^L)}{Y_{mt}^U - Y_{mt}^L} , \\ Y_{mt}^L &= \sum_{i=0}^{n} \min(\gamma_{mti} z_i^L, \gamma_{mti} z_i^U) , \\ Y_{mt}^U &= \sum_{i=0}^{n} \max(\gamma_{mti} z_i^L, \gamma_{mti} z_i^U) , \end{aligned}$$

and it follows that

$$L_m(z) \geq g_m(z), \; m = p+1, \dots, q .$$

Thus if in (RCP) the reverse convex constraints are replaced by the overestimation linear constraints, the feasible region for RCP will be contained in the new feasible region.

(3) Linear Relaxation Programming For the objective function of RCP, it is obvious that $\min \exp(z_0)$ is equivalent to $\min z_0$. From the above discussion for the two kinds of constraints respectively, [19] constructs the corresponding linear relaxation programming on the region Ω *LRP*(Ω) as follows:

$$\begin{cases} \min & z_0 \\ \text{s.t.} & L_m(z) \leq 0, \; m = 1, \dots, p \\ & L_m(z) \geq 1, \; m = p+1, \dots, q \\ & z \in \Omega = \{z \colon z_i^L \leq z_i \leq z_i^U , \\ & i = 0, 1, \dots, n\} . \end{cases}$$

The following results establish some salient properties of the linear relaxation programming *LRP*(Ω) that are essential in designing the proposed algorithm.

Lemma 1 *Assume the minimum of LRP(Ω) is LB*; then* $\exp(LB^*)$ *provides a lower bound of the optimal value of the RCP problem.*

Proof We denote the feasible region of RCP and *LRP*(Ω) D and P; then it is immediate that $P \supseteq D$ by the construction method. So based on the above assumption, $\exp(LB^*)$ is a lower bound of the minimum of the RCP problem. □

Branch-and-Bound Algorithm

Reference [19] develops a branch-and-bound algorithm to solve the RCP based on the former linear relaxation method. This algorithm needs to solve a sequence of linear relaxation programming problems over Ω or the subsets of Ω in order to find a global solution. Furthermore, to ensure convergence to a global solution, a new bound tightening method (BTM) is proposed and will be applied to enhance the solution procedure.

The critical element in guaranteeing convergence to a global minimum is the choice of a suitable branching rule. In [18] three kinds of branching methods are provided. Reference [19] chooses the first method, a simple and standard bisection rule. This method is sufficient to ensure convergence since it drives all the intervals to zero for the variables that are associated with the term that yields the greatest discrepancy in the employed approximation along any infinite branch of the branch-and-bound tree.

Branching rule:

Assume that the hyperrectangle Ω^q is going to be divided. Then the selection of the branching variable z_e, which possesses a maximum length in Ω^q and the partitioning of Ω^q are done using the following rules, where $\Omega^q = \{z \colon z_j^L(\Omega^q) \leq z_j \leq z_j^U(\Omega^q), \; j = 0, \dots, n\}$. Let

$$e = \arg\max \left\{ z_j^U(\Omega^q) - z_j^L(\Omega^q) \right\} ,$$

and partition Ω^q by bisecting the interval $[z_e^L(\Omega^q), z_e^U(\Omega^q)]$ into the subintervals $[z_e^L(\Omega^q), (z_e^L(\Omega^q) + z_e^U(\Omega^q))/2]$ and $[(z_e^L(\Omega^q) + z_e^U(\Omega^q))/2, z_e^U(\Omega^q)]$.

In what follows we describe the BTM strategy proposed by [19].

Assume that the subhyperectangle $\Omega^{q(s)}$ (s is the iteration counter) is selected for further consideration. If in the node $q(s)$ the corresponding solution $\hat{z}(\Omega^{q(s)})$ is not feasible in some convex constraint, let

$$l = \arg\max\{g_m(\hat{z}(\Omega^{q(s)}))|$$

$$g_m(\hat{z}(\Omega^{q(s)})) = \sum_{t=1}^{T_m} u_{mt}(\hat{z}(\Omega^{q(s)})) > 1\}\,.$$

Compute the weight vector $\bar{\varepsilon}_l$ by $\bar{\varepsilon}_{li} = u_{li}(\hat{z})/g_l(\hat{z})$, $i = 1, \dots, T_l$, and then condense the function $g_l(z)$ using this weight vector as described in Sect. "Linear Relaxation Programming." Then a new single term is obtained, and therefore a new linear constraint is added to the linear relaxation programming $LRP(\Omega^{q(s)})$. Denote this new linear relaxation programming and new added condensed single term $\overline{LRP}(\Omega^{q(s)})$ and $\bar{g}_l(z)$. And from the discussion in Sect. "Linear Relaxation Programming" we know $\bar{g}_l(z) = \bar{c}_l \exp(\sum_i \bar{\gamma}_{li} z_i)$, where $\bar{c}_l = \prod_t (c_{lt}/\bar{\varepsilon}_{lt})^{\bar{\varepsilon}_{lt}}$ and $\bar{\gamma}_{li} = \sum_t \gamma_{lti}\bar{\varepsilon}_{lt}$.

It is obvious that

$$\bar{g}_l(\hat{z}(\Omega^{q(s)})) = g_l(\hat{z}(\Omega^{q(s)}))\,,$$

and since $g_l(\hat{z}(\Omega^{q(s)})) > 1$, it follows that $\hat{z}(\Omega^{q(s)})$ does not satisfy the new added constraint $\bar{g}_l(z) \le 1$. From the arithmetic-geometric mean inequality, we have $\bar{g}_l(z) \le g_l(z)$. Of course, the new single-term constraint $\bar{g}_l(z) \le 1$ is equivalent to a linear constraint. Hence, if z is feasible for RCP, it is certainly feasible for $\overline{LRP}(\Omega^{q(s)})$, whose feasible region obviously does not contain the point $\hat{z}(\Omega^{q(s)})$. Clearly, this BTM technique will enhance the solution procedure.

Based on the previous BTM technique, [19] constructs the global optimization algorithm. The basic steps of the algorithm are summarized in the following statement.

Algorithm Statement

step 0: (Initialization)

0.1: Assume a convergence tolerance $\delta > 0$, and the initial weights $\varepsilon_m, m = 1, \dots, p$. Set the iteration counter $s = 0$, then $Q_s = Q_0 = \{1\}, q(s) = q(0) = 1$, $\Omega^{q(s)} = \Omega^1 = \Omega$. Set an initial upper bound $U^* = \infty$.

0.2: Solve the problem $LRP(\Omega^{q(s)})$, and denote the solution and the minimum $(\hat{z}(\Omega^{q(s)}), LB_{q(s)})$.

0.3: If $\hat{z}(\Omega^{q(s)})$ is feasible for RCP, then stop with $\hat{z}(\Omega^{q(s)})$ as the prescribed solution to the RCP problem, else let $LB(s) = LB_{q(s)}$;

0.4: If $\hat{z}(\Omega^{q(s)})$ is not feasible on some convex constraints, the BTM technique will be adopted.

step 1: (Partitioning step) Choose a branching variable z_e, then partition $\Omega^{q(s)}$ to get $\Omega^{q(s).1}$ and $\Omega^{q(s).2}$. Replace $q(s)$ by node indices $q(s).1, q(s).2$ in Q_s.

step 2: (Feasibility check for (RCP)) For each $q(s).w$, where $w = 1, 2$, compute:

$$g_m(w) = \bar{c}_m \exp\left(\sum_{i=0}^{n} \min(\bar{\gamma}_{mi} z_i^L, \bar{\gamma}_{mi} z_i^U)\right)\,,$$

$$\text{for } m = 1, \dots, p$$

$$g_m(w) = \sum_{t=1}^{T_m} c_{mt} \exp\left(Y_{mt}^U\right)\,,$$

$$\text{for } m = p+1, \dots, q$$

where $\bar{c}_m, \gamma_{mi}, Y_{mt}^U$ have been defined in Sect. "Linear Relaxation Programming." If for some $m \in \{1, \dots, p\}$, $g_m(z) > 1$, or for some $m \in \{p+1, \dots, q\}$, $g_m(z) < 1$, then the node indices $q(s).w$ will be eliminated. If $\Omega^{q(s).w} (w = 1, 2)$ are all eliminated, then go to step 5.

step 3: (Updating upper bound) For undeleted sub-hyperrectangle update

$$A_{mt}, B_{mt}, Y_{mt}^L, Y_{mt}^U\,.$$

Solve $LRP(\Omega^{q(s).w})$, where $w = 1$ or $w = 2$ or $w = 1, 2$, and denote the solutions and optimal values $(\hat{z}(\Omega^{q(s).w}), LB_{q(s).w})$. Then if $\hat{z}(\Omega^{q(s).w})$ is feasible for RCP, $U^* = \min\{U^*, LB_{q(s).w}\}$.

step 4: (Deleting step) If $LB_{q(s).w} > U^* + \delta$, then delete the corresponding node;

step 5: (Fathoming step) Fathom any nonimproving nodes by setting $Q_{s+1} = Q_s - \{q \in Q_s : LB_q \ge U^* - \delta\}$. If $Q_{s+1} = \emptyset$, then stop, and $\exp(U^*)$ is the optimal value, $z^*(\kappa)$ (where $\kappa \in \kappa_0$) are the global solutions, where $\kappa_0 = \{\kappa : z_0^*(\kappa) = U^*\}$. Otherwise, $s = s + 1$;

step 6: (Node-selection step) Set $LB(s) = \min\{LB_q : q \in Q_s\}$, then select an active node $q(s) \in \arg\min\{LB(s)\}$ for further considering;

step 7: (Bound tightening step) If in this node $q(s)$, $\hat{z}(\Omega^{q(s)})$ is feasible in all convex constraints of RCP, then return to step 1, else the BTM technique will be adopted, and then return to step 1.

Theorem 1 (convergence result) *The above algorithm either terminates finitely with the incumbent solution being optimal to RCP or it generates an infinite sequence of iterations such that along any infinite branch of the branch-and-bound tree, any accumulation point of the sequence LB(s) will be the global minimum of the RCP problem.*

Proof A sufficient condition for a global optimization to be convergent to the global minimum, stated in Horst and Tuy [7], requires that the bounding operation be consistent and the selection operation bound improving.

A bounding operation is called consistent if at every step any unfathomed partition can be further refined and if any infinitely decreasing sequence of successively refined partition elements satisfies:

$$\lim_{s\to+\infty}(U^* - LB(s)) = 0\,, \tag{3}$$

where $LB(s)$ is a lower bound inside some subhyperrectangle in stage s and U^* is the best upper bound at iteration s, not necessarily occurring inside the above same subhyperrectangle. In the following we will demonstrate that (3) holds.

Since the employed subdivision process is the bisection, the process is exhaustive. Consequently, from the discussion in [13] (3) holds, and this means that the employed bounding operation is consistent.

A selection operation is called bound improving if at least one partition element where the actual lower bound is attained is selected for further partition after a finite number of refinements. Clearly, the employed selection operation is bound improving because the partition element where the actual lower bound is attained is selected for further partition in the immediately following iteration.

In summary, it is shown that the bounding operation is consistent and that the selection operation is bound improving; therefore, according to Theorem IV.3. in Horst and Tuy [7], the employed global optimization algorithm is convergent to the global minimum. □

Applications

Reference [19] reports the numerical experiment for the deterministic global optimization algorithm described above to demonstrate its potential and feasibility. The experiment is carried out with the C programming language. The simplex method is applied to solve the linear relaxation programming problems.

To illustrate how the proposed algorithm works, first [19] gives a simple example to show the solving procedure of the proposed algorithm.

Example 1:

$$\begin{cases} \min & x_1^2 + x_2^2 \\ \text{s.t.} & 0.3x_1x_2 \ge 1 \\ & x \in X = \{2 \le x_1 \le 5; \\ & 1 \le x_2 \le 3\}\,. \end{cases}$$

First, transform the above problem into the RPGP form as follows:

$$\begin{cases} \min & x_0 \\ \text{s.t.} & g_1(x) = x_0^{-1}x_1^2 + x_0^{-1}x_2^2 \le 1 \\ & g_2(x) = 0.3x_1x_2 \ge 1 \\ & x \in \Omega_0 = \{x \mid 5 \le x_0 \le 10; \\ & 2 \le x_1 \le 5;\ 1 \le x_2 \le 3\}\,. \end{cases}$$

Let $x_i = \exp z_i\ (i = 0, 1, 2)$, then we can obtain the following reverse convex programming problem (P) of Example 1 :

$$\begin{cases} \min & \exp(z_0) \\ \text{s.t.} & f_1(z) = \exp(-z_0 + 2z_1) \\ & +\exp(-z_0 + 2z_2) \le 1 \\ & f_2(z) = 0.3\exp(z_1 + z_2) \ge 1 \\ & z \in \Omega = \{z \mid \\ & 1.6094 \le z_0 \le 2.3026; \\ & 0.6931 \le z_1 \le 1.6094; \\ & 0 \le z_2 \le 1.0986\}\,. \end{cases}$$

In step 0, set $\delta = 10^{-3}$, s=0, $U^* = \infty$. For the convex constraint function $f_1(z)$, choose the initial weight as $\varepsilon_1 = (1/2, 1/2)$ since it has two terms. Then $q(s) = 1$, $Q_s = Q_0 = \{1\}$, $\Omega^{q(s)} = \Omega^1 = \Omega$. According to the discussion in Sect. "Methods and Applications“, the $LRP(\Omega^1)$ of problem P is formulated below:

$$\begin{cases} \min & z_0 \\ \text{s.t.} & L_1(z) = -z_0 + z_1 + z_2 \leq -0.6931 \\ & L_2(z) = 1.9356z_1 + 1.9356z_2 \geq 1.7416 \\ & z \in \Omega^1 \,. \end{cases}$$

The solution and optimal value of $LRP(\Omega^1)$ are:

$$\hat{z}(\Omega^1) = (1.6094, 0.6931, 0.2231)\,,$$
$$LB_1 = 1.6094\,.$$

Since $\hat{z}(\Omega^1)$ is not feasible for problem P, then $LB(s) = LB(0) = 1.6094$. Since $\hat{z}(\Omega^1)$ is not feasible for $f_1(z) \leq 1$, then the BTM technique will be adopted. First, update the weight ε_1 according to the solution $\hat{z}(\Omega^1)$, and derive $\varepsilon_1 = (0.7191, 0.2809)$, then from formula (1) in Sect. "Methods and Applications", we obtain a new linear constraint:

$$L_3(z) = -z_0 + 1.4382z_1 + 0.5618z_2 \leq -0.5938\,.$$

The current linear relaxation programming denoted as $\overline{LRP}(\Omega^1)$ is:

$$\begin{cases} \min & z_0 \\ \text{s.t.} & L_1(z) = -z_0 + z_1 + z_2 \leq -0.6931 \\ & L_2(z) = 1.9356z_1 + 1.9356z_2 \geq 1.7416 \\ & L_3(z) = -z_0 + 1.4382z_1 + 0.5618z_2 \\ & \leq -0.5938 \\ & z \in \Omega^1 \,. \end{cases}$$

In step 1, divide the region Ω^1 into the following two regions:

$$\Omega^2 = \{z \,|\, 1.6094 \leq z_0 \leq 2.3026\,;$$
$$0.6931 \leq z_1 \leq 1.6094;\ 0 \leq z_2 \leq 0.5493\}\,,$$
$$\Omega^3 = \{z \,|\, 1.6094 \leq z_0 \leq 2.3026\,;$$
$$0.6931 \leq z_1 \leq 1.6094;\ 0.5493 \leq z_2 \leq 1.0986\}\,,$$

then the node set $Q_0 = \{2, 3\}$.

In step 2, the two nodes in Q_0 have not been deleted; then go to step 3. After updating the parameters according to the formula in Sect. "Linear Relaxation Programming" respectively, we can obtain the new function $L_2(z)$ in each node. Then we have $LRP(\Omega^2)$:

$$\begin{cases} \min & z_0 \\ \text{s.t.} & L_1(z) = -z_0 + z_1 + z_2 \leq -0.6931 \\ & L_2(z) = 1.3633z_1 + 1.3633z_2 \geq 1.3450 \\ & L_3(z) = -z_0 + 1.4382z_1 + 0.5618z_2 \\ & \leq -0.5938 \\ & z \in \Omega^2 \end{cases}$$

and we have $LRP(\Omega^3)$:

$$\begin{cases} \min & z_0 \\ \text{s.t.} & L_1(z) = -z_0 + z_1 + z_2 \leq -0.6931 \\ & L_2(z) = 2.3613z_1 + 2.3613z_2 \geq 2.8946 \\ & L_3(z) = -z_0 + 1.4382z_1 + 0.5618z_2 \\ & \leq -0.5938 \\ & z \in \Omega^3 \,. \end{cases}$$

The solutions and optimal values are respectively

$$\hat{z}(\Omega^2) = (1.7555, 0.6931, 0.2934)\,,$$
$$LB_2 = 1.7555$$
$$\hat{z}(\Omega^3) = (1.9356, 0.6931, 0.8427)\,,$$
$$LB_3 = 1.9356\,.$$

In step 4 the two nodes have not been deleted; then go to step 5. Compute

$$Q_1 = Q_0 - \{q \in Q_0 \colon LB_q \geq U^* - \delta\} = \{2, 3\}\,,$$

and set $s = 1$. In step 6, the current lower bound is

$$LB(s) = LB(1) = \min\{LB_q, q \in Q_s\}$$
$$= \min\{LB_2, LB_3\} = 1.7555\,.$$

So we will choose the active node as $q(1) = 2$ for further consideration.

In step 7 in the node $q(1)$, the BTM technique is adopted. From formula (1) in Sect. "Methods and Applications" we compute the new weight $\varepsilon_1 = (0.6899, 0.3101)$ according to the solution $\hat{z}(\Omega^2)$, and we obtain the following new linear constraint:

$$L_4(z) = -z_0 + 1.3797z_1 + 0.6203z_2 \le 1.2919\,.$$

The current linear relaxation programming denoted as $\overline{LRP}(\Omega^2)$ is:

$$\begin{cases} \min & z_0 \\ \text{s.t.} & L_1(z) = -z_0 + z_1 + z_2 \le -0.6931 \\ & L_2(z) = 1.3633z_1 + 1.3633z_2 \ge 1.3450 \\ & L_3(z) = -z_0 + 1.4382z_1 + 0.5618z_2 \\ & \le -0.5938 \\ & L_4(z) = -z_0 + 1.3797z_1 + 0.6203z_2 \\ & \le 1.2919 \\ & z \in \Omega^2\,. \end{cases}$$

Then return to step 1, divide the region Ω^2, and go into a new circle. After 22 iterations, the procedure stops. The global minimum of problem P is 1.9140, and the global solution is

$$z^* = (1.9140, 0.6933, 0.5107)\,.$$

Then the global minimum of example 1 is 6.7804, and the global solution is $x^* = (2.0003, 1.6664)$.

Additionally, to test the algorithm, [19] chooses five examples, all of which are taken from engineering, concerning the detailed application context, please refer to the releveant references.

Example 2 ([1]):

$$\begin{cases} \min & x_0 \\ \text{s.t.} & x_0^{-1}x_2^{-1}x_3^{-1}x_5 + 5x_0^{-1}x_1^{\frac{1}{2}}x_4x_5 \le 1 \\ & x_2^{\frac{1}{3}}x_3 - x_4^{\frac{1}{2}} \le -1 \\ & -x_5 - 2x_0x_1x_2x_3^4x_4^{-1}x_5 \le -1 \\ & x \in X = \{x \mid 30 \le x_0 \le 40; \\ & 0.01 \le x_1 \le 1; \\ & 0.0001 \le x_2 \le 1; \\ & 15 \le x_3 \le 20; \\ & 15 \le x_4 \le 20; \\ & 0.1 \le x_5 \le 1\}\,. \end{cases}$$

Example 3 ([11]):

$$\begin{cases} \min & x_0 \\ \text{s.t.} & 0.274x_3x_4^4 + 2520.66x_1x_4^5 + x_0x_3^2 \\ & -x_0x_1x_2x_3 + 1 \le 1 \\ & x_1x_2^{-1}x_3 \le 1 \\ & x_1x_4^4 \le 1 \\ & x_3x_4^3 \le 1 \\ & x \in X = \{x \mid 10^{-12} \le x_0 \le 2; \\ & 20 \le x_1 \le 35; \\ & 120 \le x_2 \le 160; \\ & 1 \le x_3 \le 10; \\ & 10^{-6} \le x_4 \le 1\}\,. \end{cases}$$

Example 4 ([20]):

$$\begin{cases} \min & x_0 \\ \text{s.t.} & 3.7x_0^{-1}x_1^{0.85} + 1.985x_0^{-1}x_1 \\ & +700.3x_0^{-1}x_2^{-0.75} \le 1 \\ & 0.7673x_2^{0.05} - 0.05x_1 \le 1 \\ & x \in X = \{x \mid 5 \le x_0 \le 15; \\ & 0.1 \le x_1 \le 5; \\ & 380 \le x_2 \le 450\}\,. \end{cases}$$

Example 5 ([20]):

$$\begin{cases} \min & x_0 \\ \text{s.t.} & 4x_1 - 4x_0^2 \le 1 \\ & -x_0 - x_1 \le -1 \\ & x \in X = \{x \mid 0.01 \le x_0 \le 15; \\ & 0.01 \le x_1 \le 15\}\,. \end{cases}$$

Example 6 ([5]):

$$\begin{cases} \min & x_3^{0.8}x_4^{1.2} \\ \text{s.t.} & x_1x_4^{-1} + x_2^{-1}x_4^{-1} \le 1 \\ & -x_1^{-2}x_3^{-1} - x_2x_3^{-1} \le -1 \\ & x \in X = \{x \mid 0.1 \le x_1 \le 1; \\ & 5 \le x_2 \le 10; \\ & 8 \le x_3 \le 15; \\ & 0.01 \le x_4 \le 1\}\,. \end{cases}$$

The following table summarizes the computational results on the above five examples. In the table *s* denotes the number of the iteration, L denotes the longest node number in Q_s described in the algorithm statement, and δ denotes the convergence tolerance. The results show that the algorithm of [19] can globally solve the GGP problem effectively.

No.	Solution
2	(37.0070,0.4489,0.0048,18.0348,16.0449,0.5667)
3	(0.0000, 32.7781,155.0000, 4.7288, 0.0027)
4	(11.9637, 0.8098, 442.0915)
5	(0.5, 0.5)
6	(0.1020, 7.0711, 8.3284, 0.2434)

no	s	L	δ	CPU time
2	131	28	10^{-3}	4s
3	191	74	10^{-6}	6s
4	138	39	10^{-6}	5s
5	96	10	10^{-9}	1s
6	146	42	10^{-6}	6s

References

1. Avriel M, Williams AC (1971) An extension of geometric programming with applications in engineering optimization. J Eng Math 5(3):187–199
2. Bricker DL, Kortanek KO, Xu L (1995) Maximum likelihood estimates with order restrictions on probabilities and odds ratios: a geometric programming approach. J Appl Math Decis Sci 1(1):53–65, The University of IA, Iowa City
3. Choi JC, Bricker DL (1996) Effectiveness of a geometric programming algorithm for optimization of machining economics models. Comput Oper Res 23(10):957–961
4. Das K, Roy TK, Maiti M (2000) Multi-item inventory model with under imprecise objective and restrictions: a geometric programming approach. Product Plann Control 11(8):781–788
5. Duffin RJ, Peterson EL (1973) Geometric programming with signomial. J Optim Theory Appl 11(1):3–35
6. El Barmi H, Dykstra RL (1994) Restricted multinomial maximum likelihood estimation based upon Fenchel duality. Statist Probab Lett 21:121–130
7. Horst R, Tuy H (1990) Global optimization, Deterministic Approaches. Springer, Berlin
8. Sonmez AI, Baykasoglu A, Dereli T, Flz IH (1999) Dynamic optimization of multipass milling operations via geometric programming. Int J Mach Tools Manufact 39:297–320
9. Jagannathan R (1990) A stochastic geometric programming problem with multiplicative recourse. Oper Res Lett 9:99–104
10. Jefferson TR, Scott CH (1978) Generalized geometric programming applied to problems of optimal control: I. Theory. JOTA 26:117–129
11. Jha NK (1995) Geometric programming based robot control design. Comput Ind Eng 29(1–4):631–635
12. Kortanek KO, Xu X, Ye Y (1996) An infeasible interior-point algorithm for solving primal and dual geometric programs. Math Program 76:155–181
13. Maranas CD, Floudas CA (1997) Global optimization in generalized geometric programming. Comput Chem Eng 21(4):351–369
14. Passy U (1971) Generalized weighted mean programming. SIAM J Appl Math 20:763–778
15. Passy U, Wilde DJ (1967) Generalized polynomial optimization. J Appl Math 15(5):1344–1356
16. Rijckaert MJ, Martens XM (1974) Analysis and optimization of the Williams–Otto process by geometric programming. AIChE J 20(4):742–750
17. Scott CH, Jefferson TR (1995) Allocation of resources in project management. Int J Syst Sci 26:413–420
18. Sherali HD (1998) Global optimization of nonconvex polynomial programming problems having ratilnal exponents. J Global Optim 12:267–283
19. Wang Y, Zhang K, Gao Y (2004) Global optimization of generalized geometric programming. Comput Math Appl 48:1505–1516
20. Zhang K (1990) Geometric Programming and Optimal Design. Xi'an Jiaotong University publishing house, China

Global Equilibrium Search

ERIKA J. SHORT, OLEG V. SHYLO
Center for Applied Optimization,
Department of Industrial and Systems Engineering,
University of Florida, Gainesville, USA

Article Outline

Abstract

Global equilibrium search is a method that can be applied to a variety of hard optimization problems. The algorithm utilizes ideas similar to those of the simulated annealing method. The algorithm accumulates information about the search space in order to generate new solutions for the subsequent stages. This method has been successfully applied to well-known problems such as the multidimensional knapsack problem, the job-shop scheduling problem, the unconstrained

quadratic programming problem, the maximum satisfiability problem, etc.

The numerous discrete optimization problems that arise in practice have such different characteristics that development of a general purpose solution method is clearly impracticable. One way of tackling this issue is to develop a library of suitable solution methods, allowing the practitioner to choose the most suitable for his problem under his time constraints and quality requirements. In recent decades, heuristic approaches, such as tabu search [1], simulated annealing (SA) [3], etc., have gained a considerable amount of attention from the scientific community for being the only practical tool that can be applied to a wide range of difficult problems. Global equilibrium search (GES) offers another highly effective tool for solving large-scale optimization problems.

The method was introduced by Shylo [7] in 1999. It shares ideas similar to those that inspired the SA technique, while providing, in practice, faster asymptotic convergence to the optimal solution on a wide class of optimization problems. Moreover, the GES method can be used in an ensemble with other techniques, which makes it more versatile than most of its predecessors.

Consider a discrete optimization problem of the following form:

$$\min\{f(x) : x \in S : S \subseteq \{0,1\}^n\} \tag{1}$$

where f is some quality function. Let us introduce a random binary vector ξ that takes a value from a feasible set S according to the Boltzmann distribution, with $\mu \geq 0$ being the temperature parameter:

$$P\{\xi(\mu) = x\} = \frac{\exp(-\mu f(x))}{\sum_{x \in S} \exp(-\mu f(x))} \,. \tag{2}$$

Consider the SA method applied to problem (1). It can be shown easily that under certain conditions (i. e., symmetric neighborhood structure) the stationary probabilities of the Markov chain associated with the SA method are given by (2).

Set S can be split into two subsets in such a way that one of them contains the feasible solutions for which the jth component is 1, and another set will contain the solution with the jth component equal to 0. Let us name these two sets S_j^1 and S_j^0. Obviously, $S_j^1 \cup S_j^0 = S$. Then the probability of the jth component of ξ being 1 can be expressed as

$$p_j(\mu) \equiv P\{\xi_j(\mu) = 1\} = \frac{\sum_{x \in S_j^1} \exp(-\mu f(x))}{\sum_{x \in S} \exp(-\mu f(x))} \,. \tag{3}$$

The idea of the GES method is to use some subset of known solutions $\widehat{S}$ to generate new solutions in the successive stages of the algorithm. The distribution (3) or any other equivalent formula [4] can be used for such a generation (substituting S with $\widehat{S}$ in the formula):

$$\widehat{p}_j(\mu) \equiv P\{\widehat{\xi}_j(\mu) = 1\} = \frac{\sum_{x \in \widehat{S}_j^1} \exp(-\mu f(x))}{\sum_{x \in \widehat{S}} \exp(-\mu f(x))} \,. \tag{4}$$

If $\arg\min\{f(x) : x \in \widehat{S}\}$ is unique, then the average Hamming distance between newly generated solutions and the best solution in the set $\widehat{S}$ converges to zero as μ goes to infinity. However, the speed of such convergence is not the same for different components of the solutions generated, i. e., the speed of convergence of the jth component depends on the quality of the solutions S_j^1 compared with the quality of solutions in S_j^0. Simply put, the temperature parameter in (3) controls the level of similarity of the newly generated solutions with high-quality solutions in $\widehat{S}$. The uniqueness of the best solution x^* in $\widehat{S}$ mentioned above should be maintained at all stages of the algorithm.

One of the limitations of the strategy described above is that in order to implement it, there should exist an easy way of generating random solutions from S with the distribution given by (4). Unfortunately, for some problems, the structure of set S would make this hard to achieve. For such cases, the local search based techniques (i. e., SA method, tabu method, GES method) are not easily applicable.

Another issue with generating the random solution x from S using (4) is that the components of the random solution x are not independent random variables. However, for the simplicity of an algorithm, this is usually ignored because the convergence property is more important for the performance of the algorithm.

Whenever the new solution is added to set $\widehat{S}$, it is easy to recalculate the probabilities $\widehat{p}_j$ if the denominator and numerator in (4) are stored separately. Therefore, there is no need to store the whole set $\widehat{S}$ in the

```
Input: $\mu$ – vector of temperature values, $K$ – number of temperature stages,
    $maxnfail$ – restart parameter, $ngen$ – # of solutions generated during each
    stage
Output:
x_best ← construct random solution; $\widehat{S}=E=\{x_{\text{best}}\}$
while stopping criterion = FALSE do
    if $\widehat{S} = \emptyset$ then
        $x$ ← construct random solution
        $x_{\text{max}} = x$
        $\widehat{S} = \{x_{\text{max}}\}$ (set of known solutions)
        $E = \{x_{\text{max}}\}$ (set of elite solutions)
    end if
    for $nfail = 0$ to $nfail^*$ do
        $x_{\text{old}} = x_{\text{max}}$
        for $k = 0$ to $K$ do
            calculate generation probabilities($p^k, \widehat{S}, \mu_k$)
            for $g = 0$ to $ngen$ do
                $x$ ← generate solution($x_{max}, p^k$)
                $R$ ← search method($x$) (R is some subset of encountered solutions)
                $\widehat{S} = \widehat{S} \cup R$
                $x_{\text{max}} = \arg\min \{f(x) : x \in \widehat{S}\}$
                if $f(x_{\text{max}}) < f(x_{\text{best}})$ then
                    $x_{\text{best}} = x_{\text{max}}$
                end if
                update_elite_set(E,R)
            end for
        end for
        if $f(x_{\text{old}}) > f(x_{\text{max}})$ then
            $nfail = 0$
        end if
        $\widehat{S} = E$
    end for
    $P = P \cup N(x_{\text{best}}, d_p)$
    $E = E - P$
    if RESTART-criterion= TRUE then
        $E = \emptyset$
    end if
    $\widehat{S} = E$;
    $x_{\text{max}} = \arg\min \{f(x) : x \in \widehat{S}\}$
end while
return $x_{\text{best}}$
```

Global Equilibrium Search, Figure 1
Global equilibrium search method (general scheme)

memory! The notion of $\widehat{S}$ is used below mainly for the simplicity of discussion.

The performance of any GES-based algorithm is dependent on the choice of the temperature schedule. As with the SA method, there is no basic recipe to provide an optimal schedule for the GES. The general advice here is to choose the sequence of increasing values $\mu_0 = 0, \mu_1, \mu_2 = \mu_1\alpha, \ldots, \mu_K = \mu_{K-1}\alpha$ (K is a number of temperature stages and $\alpha > 0$), in such a manner that the algorithm will find the best solution from set $\widehat{S}$ almost for sure when generating solutions with temperature parameter μ_K. However, there is no need to provide a separate cooling schedule for each problem solved. Simple scaling of the cost function ($f'(x) = C \cdot f(x)$, $C > 0$) can make one temperature schedule suitable for a wide range of problems from the same class. The choice of scaling factor can be made, for example, in the initial stage of the algorithm, when $\mu = 0$. Additionally, if we multiply the denominator and numerator of (4) by $\exp(\mu f(x^*))$, where x^* is the best solution from $\widehat{S}$, then the convergence to the best solution from $\widehat{S}$ is less dependent on the absolute values of solution costs.

The general scheme of the GES method is presented in Fig. 1. There are some elements that are included in the scheme, but that were not discussed above: elite solutions set, prohibition of certain solutions and restarting the search. These elements are not necessary for success of the GES method and can be easily excluded. However, for some classes of problems they can provide a significant performance improvement.

The main cycle (lines 2–36) is repeated until some stopping criterion is satisfied. The algorithm execution can be terminated when the best known record for the given problem is improved, or when the running time exceeds some limiting value. If the set of known solutions $\tilde{S}$ is empty, then the initialization of the data set is performed in lines 3–7. The cycle in lines 9–28 is executed until there is no improvement in *nfail** consecutive cycles. The main element of the GES method is the temperature cycle (lines 11–23). The probabilities that guide the search are estimated using expression (4) at the beginning of each temperature stage (line 12). For each probability vector, *ngen* solutions are generated (lines 13–22). These solutions are used as initial solutions for the local search procedure (line 15). The subset of encountered solutions R is used to update set $\widehat{S}$ (line 16).

Some set of the solutions can be stored in memory, in order to provide a fast initialization of the algorithm's memory structures (lines 27 and 34). Such a set is referred to as an elite set in the algorithm pseudocode. Certain solutions can be excluded from this set to avoid searching the same areas multiple times. In lines 29 and 30, the solutions for which the Hamming distance to x_{best} is less than parameter d_p are excluded from the elite set.

A number of successful applications of the GES method have been reported in recent years [6]. The application of the GES method for the multidimensional knapsack problem is described in [8].

The GES based method was presented in [5] for solving job-shop scheduling problems. To date, suitable exact solution methods are not able to find high-quality solutions with reasonable computational effort for the problems involving more than ten jobs and ten machines. The computational testing of the GES algorithm provided a set of new upper bounds for a wide set of challenging benchmark problems [2]. The comparison with existing techniques for job-shop scheduling asserts that the GES method has a great potential for solving scheduling problems.

The application of GES for the unconstrained quadratic programming problem was discussed in [4], where GES was used in a combination with a tabu algorithm. Such an ensemble proved to be an extremely efficient tool for large-scale problems, outperforming some of the best available solution techniques.

In conclusion, the universality of the GES method together with its flexibility make it an optimization tool worth considering.

References

1. Glover F, Laguna M (1993) Tabu search in Modern Heuristic Techniques for Combinatorial Problems. In: Reeves C (ed). Blackwell, Oxford, pp 70–141
2. Job Shop Scheduling webpage, http://plaza.ufl.edu/shylo/jobshopinfo.html. Accessed 14 Oct 2007
3. Kirkpatrick S, Gelatt CD, Vecchi MP (1983) Optimization by Simulated Annealing. Science 220(4598):671–680
4. Pardalos PM, Prokopyev OA, Shylo OV, Shylo VP (2007) Global equilibrium search applied to the unconstrained binary quadratic optimization problem. Optim Meth Softw. doi:10.1080/10556780701550083
5. Pardalos PM, Shylo OV (2006) An algortihm for the Job Shop Scheduling based on Global Equilibrium Search Techniques. Comput Manag Sci 3(4):331–348
6. Sergienko IV, Shylo VP (2006) Problems of discrete optimization: Challenges and main approaches to solve them. Cybernet Syst Anal 42:465–482
7. Shylo VP (1999) A global equilibrium search method. Kybernetika i Systemnuiy Analys 1:74–80 (in Russian)
8. Shylo VP (2000) Solution of multidimensional knapsack problems by global equilibrium search, Theory of Optimal Solutions. Glushkov VM Inst Cybern, NAS Ukraine Kiev, p 10

Globally Convergent Homotopy Methods

LAYNE T. WATSON
Virginia Polytechnic Institute and State University,
Virginia, USA

MSC2000: 65F10, 65F50, 65H10, 65K10

Article Outline

Keywords

Continuation; Globally convergent; Homotopy; Nonlinear equations; Probability-one homotopy

Probability-one homotopy methods are a class of algorithms for solving nonlinear systems of equations that are accurate, robust, and converge from an arbitrary starting point almost surely. These new globally convergent homotopy techniques have been successfully applied to solve Brouwer fixed point problems, polynomial systems of equations, constrained and unconstrained optimization problems, discretizations of nonlinear two-point boundary value problems based on shooting, finite differences, collocation, and finite elements, and finite difference, collocation, and Galerkin approximations to nonlinear partial differential equations.

Probability-One Globally Convergent Homotopies

A *homotopy* is a continuous map from the interval [0, 1] into a function space, where the continuity is with respect to the topology of the function space. Intuitively, a homotopy $\rho(\lambda)$ continuously deforms the function $\rho(0) = g$ into the function $\rho(1) = f$ as λ goes from 0 to 1. In this case, f and g are said to be *homotopic*. Homotopy maps are fundamental tools in topology, and provide a powerful mechanism for defining equivalence classes of functions.

Homotopies provide a mathematical formalism for describing an old procedure in numerical analysis, variously known as continuation, incremental loading, and embedding. The continuation procedure for solving a nonlinear system of equations $f(x) = 0$ starts with a (generally simpler) problem $g(x) = 0$ whose solution x_0 is known. The *continuation* procedure is to track the set of zeros of

$$\rho(\lambda, x) = \lambda f(x) + (1 - \lambda)g(x) \qquad (1)$$

as λ is increased monotonically from 0 to 1, starting at the known initial point $(0, x_0)$ satisfying $\rho(0, x_0) = 0$. Each step of this tracking process is done by starting at a point $(\widetilde{\lambda}, \widetilde{x})$ on the zero set of ρ, fixing some $\Delta\lambda > 0$, and then solving $\rho(\widetilde{\lambda} + \Delta\lambda, x) = 0$ for x using a locally convergent iterative procedure, which requires an invertible Jacobian matrix $D_x\rho(\widetilde{\lambda} + \Delta\lambda, x)$. The process stops at $\lambda = 1$, since $f(\overline{x}) = \rho(1, \overline{x}) = 0$ gives a zero $\overline{x}$ of $f(x)$. Note that continuation assumes that the zeros of ρ connect the zero x_0 of g to a zero $\overline{x}$ of f, and that the Jacobian matrix $D_x\rho(\lambda, x)$ is invertible along the zero set of ρ; these are strong assumptions, which are frequently not satisfied in practice.

Continuation can fail because the curve γ of zeros of $\rho(\lambda, x)$ emanating from $(0, x_0)$ may:

1) have turning points,
2) bifurcate,
3) fail to exist at some λ values, or
4) wander off to infinity without reaching $\lambda = 1$.

Turning points and bifurcation correspond to singular $D_x\rho(\lambda, x)$. Generalizations of continuation known as *homotopy methods* attempt to deal with cases 1) and 2) and allow tracking of γ to continue through singularities. In particular, continuation monotonically increases λ, whereas homotopy methods permit λ to both increase and decrease along γ. Homotopy methods can also fail via cases 3) or 4).

The map $\rho(\lambda, x)$ connects the functions $g(x)$ and $f(x)$, hence the use of the word 'homotopy'. In general the homotopy map $\rho(\lambda, x)$ need not be a simple convex combination of g and f as in (1), and can involve λ nonlinearly. Sometimes λ is a physical parameter in the original problem $f(x; \lambda) = 0$, where $\lambda = 1$ is the (nondimensionalized) value of interest, although 'artificial parameter' homotopies are generally more computationally efficient than 'natural parameter' homotopies $\rho(\lambda, x) = f(x; \lambda)$. An example of an artificial parameter homotopy map is

$$\rho(\lambda, x) = \lambda f(x; \lambda) + (1-\lambda)(x-a), \qquad (2)$$

which satisfies $\rho(0, a) = 0$. The name 'artificial' reflects the fact that solutions to $\rho(\lambda, x) = 0$ have no physical interpretation for $\lambda < 1$. Note that $\rho(\lambda, x)$ in (2) has a unique zero $x = a$ at $\lambda = 0$, regardless of the structure of $f(x; \lambda)$.

All four shortcomings of continuation and homotopy methods have been overcome by probability-one homotopies, proposed in 1976 by S.N. Chow, J. Mallet-Paret, and J.A. Yorke [2]. The supporting theory, based on differential geometry, will be reformulated in less technical jargon here.

Definition 1 Let $U \subset \mathbf{R}^m$ and $V \subset \mathbf{R}^p$ be open sets, and let ρ: $U\times[0, 1)\times V \to \mathbf{R}^p$ be a C^2 map. ρ is said to be *transversal to zero* if the $p\times(m+1+p)$ Jacobian matrix $D\rho$ has full rank on $\rho^{-1}(0)$.

The C^2 requirement is technical, and part of the definition of transversality. The basis for the probability-one homotopy theory is the *parametrized Sard's theorem*, [2]:

Theorem 2 *Let ρ: $U \times [0, 1) \times V \to \boldsymbol{R}^p$ be a C^2 map. If ρ is transversal to zero, then for almost all $a \in U$ the map*

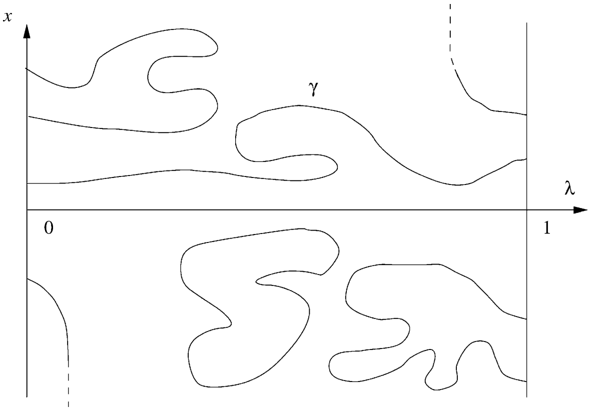
$$\rho_a(\lambda, x) = \rho(a, \lambda, x)$$

is also transversal to zero.

To discuss the importance of this theorem, take $U = \mathbf{R}^m$, $V = \mathbf{R}^p$, and suppose that the C^2 map ρ: $\mathbf{R}^m \times [0, 1) \times \mathbf{R}^p \to \mathbf{R}^p$ is transversal to zero. A straightforward application of the implicit function theorem yields that for almost all $a \in \mathbf{R}^m$, the zero set of ρ_a consists of smooth, nonintersecting curves which either:

1) are closed loops lying entirely in $(0, 1) \times \mathbf{R}^p$,
2) have both endpoints in $\{0\} \times \mathbf{R}^p$,
3) have both endpoints in $\{1\} \times \mathbf{R}^p$,
4) are unbounded with one endpoint in either $\{0\} \times \mathbf{R}^p$ or in $\{1\} \times \mathbf{R}^p$, or
5) have one endpoint in $\{0\} \times \mathbf{R}^p$ and the other in $\{1\} \times \mathbf{R}^p$.

Furthermore, for almost all $a \in \mathbf{R}^m$, the Jacobian matrix $D\rho_a$ has full rank at *every* point in $\rho_a^{-1}(0)$. The goal is to construct a map ρ_a whose zero set has an endpoint in $\{0\} \times \mathbf{R}^p$, and which rules out 2) and 4). Then 5) obtains, and a zero curve starting at $(0, x_0)$ is *guaranteed* to reach a point $(1, \bar{x})$. All of this holds for almost all $a \in \mathbf{R}^m$, and hence with probability one [2]. Furthermore, since $a \in \mathbf{R}^m$ can be almost any point (and, indirectly, so can the starting point x_0), an algorithm based on tracking the zero curve in 5) is legitimately called *globally convergent*. This discussion is summarized in the following theorem (and illustrated in Fig. 1).



Globally Convergent Homotopy Methods, Figure 1
Zero set for $\rho_a(\lambda, x)$ satisfying properties 1)–4)

Theorem 3 *Let f: $\boldsymbol{R}^p \to \boldsymbol{R}^p$ be a C^2 map, ρ: $\boldsymbol{R}^m\times[0, 1)\times \boldsymbol{R}^p \to \boldsymbol{R}^p$ a C^2 map, and $\rho_a(\lambda, x) = \rho(a, \lambda, x)$. Suppose that*
1) ρ is transversal to zero.
Suppose also that for each fixed $a \in \boldsymbol{R}^m$,
2) $\rho_a(0, x) = 0$ has a unique nonsingular solution x_0,
3) $\rho_a(1, x) = f(x)$ $(x \in \boldsymbol{R}^p)$.
Then, for almost all $a \in \boldsymbol{R}^m$, there exists a zero curve γ of ρ_a emanating from $(0, x_0)$, along which the Jacobian matrix $D\rho_a$ has full rank.

If, in addition,
4) $\rho_a^{-1}(0)$ is bounded,
then γ reaches a point $(1, \bar{x})$ such that $f(\bar{x}) = 0$). Furthermore, if $Df(\bar{x})$ is invertible, then γ has finite arc length.

Any algorithm for tracking γ from $(0, x_0)$ to $(1, \bar{x})$, based on a homotopy map satisfying the hypotheses of this theorem, is called a *globally convergent probability-one homotopy algorithm*. Of course, the practical numerical details of tracking γ are nontrivial, and have been the subject of twenty years of research in numerical analysis. Production quality software called HOMPACK90 [6] exists for tracking γ. The distinctions between continuation, homotopy methods, and probability-one homotopy methods are subtle but worth noting. Only the latter are provably globally convergent and (by construction) expressly avoid dealing with singularities numerically, unlike continuation and homotopy methods which must explicitly handle singularities numerically.

Assumptions 2) and 3) in Theorem 3 are usually achieved by the construction of ρ (such as (2)), and are straightforward to verify. Although assumption 1) is trivial to verify for some maps, if λ and a are involved nonlinearly in ρ the verification is nontrivial. Assumption 4) is typically very hard to verify, and often is a deep result, since 1)–4) holding implies the *existence* of a solution to $f(x) = 0$.

Note that 1)–4) are sufficient, but not necessary, for the existence of a solution to $f(x) = 0$, which is why homotopy maps not satisfying the hypotheses of the theorem can still be very successful on practical problems. If 1)–3) hold and a solution does *not* exist, then 4) must fail, and nonexistence is manifested by γ going off to infinity. Properties 1)–3) are important because they guarantee good numerical properties along the zero curve γ, which, if bounded, results in a *globally convergent* algorithm. If γ is unbounded, then either the homotopy approach (with this particular ρ) has failed or $f(x) = 0$ has no solution.

A few remarks about the applicability and limitations of probability-one homotopy methods are in order. They are designed to solve a *single* nonlinear system of equations, *not* to track the solutions of a parameterized family of nonlinear systems as that parameter is varied. Thus drastic changes in the solution behavior with respect to that (natural problem) parameter have no effect on the efficacy of the homotopy algorithm, which is solving the problem for a *fixed* value of the natural parameter. In fact, it is precisely for this case of rapidly varying solutions that the probability-one homotopy approach is superior to classical continuation (which would be trying to track the rapidly varying solutions with respect to the problem parameter). Since the homotopy methods described here are not for general solution curve tracking, they are not (directly) applicable to bifurcation problems.

Homotopy methods also require the nonlinear system to be C^2 (twice continuously differentiable), and this limitation cannot be relaxed. However, requiring a finite-dimensional discretization to be smooth does not mean the solution to the infinite-dimensional problem must also be smooth. For example, a Galerkin formulation may produce a smooth nonlinear system in the basis function coefficients even though the basis functions themselves are discontinuous. Homotopy methods for optimization problems may converge to a local minimum or stationary point, and in this regard are no better or worse than other optimization algorithms. In special cases homotopy methods can find all the solutions if there is more than one, but in general the homotopy algorithms are only guaranteed to find one solution.

Optimization Homotopies

A few typical convergence theorems for optimization are given next (see the survey in [5] for more examples and references). Consider first the *unconstrained optimization* problem

$$\min_x f(x). \tag{3}$$

Theorem 4 *Let f: $\boldsymbol{R}^n \to \boldsymbol{R}$ be a C^3 convex map with a minimum at $\widetilde{x}$, $\|\widetilde{x}\|_2 \leq M$. Then for almost all a, $\|a\|_2 < M$, there exists a zero curve γ of the homotopy map*

$$\rho_a(\lambda, x) = \lambda \nabla f(x) + (1 - \lambda)(x - a),$$

along which the Jacobian matrix $D\rho_a(\lambda, x)$ has full rank, emanating from $(0, a)$ and reaching a point $(1, \widetilde{x})$, where $\widetilde{x}$ solves (3).

A function is called *uniformly convex* if it is convex and its Hessian's smallest eigenvalue is bounded away from zero. Consider next the constrained optimization problem

$$\min_{x \geq 0} f(x). \tag{4}$$

This is more general than it might appear because the general convex *quadratic program* reduces to a problem of the form (4).

Theorem 5 *Let f : $\boldsymbol{R}^n \to \boldsymbol{R}$ be a C^3 uniformly convex map. Then there exists $\delta > 0$ such that for almost all $a \geq 0$*

with $\|a\|_2 < \delta$ there exists a zero curve γ of the homotopy map

$$\rho_a(\lambda, x) = \lambda K(x) + (1 - \lambda)(x - a),$$

where

$$K_i(x) = -\left|\frac{\partial f(x)}{\partial x_i} - x_i\right|^3 + \left(\frac{\partial f(x)}{\partial x_i}\right)^3 + x_i^3,$$

along which the Jacobian matrix $D\rho_a(\lambda, x)$ has full rank, connecting (0, a) to a point $(1, \overline{x})$, where $\overline{x}$ solves the constrained optimization problem (4).

Given $F : \mathbf{R}^n \to \mathbf{R}^n$, the *nonlinear complementarity problem* is to find a vector $x \in \mathbf{R}^n$ such that

$$x \geq 0, \quad F(x) \geq 0, \quad x^\top F(x) = 0. \tag{5}$$

It is interesting that homotopy methods can be adapted to deal with nonlinear inequality constraints and combinatorial conditions as in (5). Define $G : \mathbf{R}^n \to \mathbf{R}^n$ by

$$G_i(z) = -|F_i(z) - z_i|^3 + (F_i(z))^3 + z_i^3, \quad i = 1, \ldots, n,$$

and let

$$\rho_a(\lambda, z) = \lambda G(z) + (1 - \lambda)(z - a).$$

Theorem 6 *Let $F : \boldsymbol{R}^n \to \boldsymbol{R}^n$ be a C^2 map, and let the Jacobian matrix DG(z) be nonsingular at every zero of G(z). Suppose there exists r > 0 such that z > 0 and $z_k = \|z\|_\infty \geq r$ imply $F_k(z) > 0$. Then for almost all a > 0 there exists a zero curve γ of $\rho_a(\lambda, z)$, along which the Jacobian matrix $D\rho_a(\lambda, z)$ has full rank, having finite arc length and connecting (0, a) to $(1, \overline{z})$, where $\overline{z}$ solves (5).*

Theorem 7 *Let $F : \boldsymbol{R}^n \to \boldsymbol{R}^n$ be a C^2 map, and let the Jacobian matrix DG(z) be nonsingular at every zero of G(z). Suppose there exists r > 0 such that $z \geq 0$ and $\|z\|_\infty \geq r$ imply $z_k F_k(z) > 0$ for some index k. Then there exists $\delta > 0$ such that for almost all $a \geq 0$ with $\|a\|_\infty < \delta$ there exists a zero curve γ of $\rho_a(\lambda, z)$, along which the Jacobian matrix $D\rho_a(\lambda, z)$ has full rank, having finite arc length and connecting (0, a) to $(1, \overline{z})$, where $\overline{z}$ solves (5).*

Homotopy algorithms for convex unconstrained optimization are generally not computationally competitive with other approaches. For constrained optimization the homotopy approach offers some advantages, and, especially for the nonlinear complementarity problem, is competitive with and often superior to other algorithms. Consider next the general nonlinear programming problem

$$\begin{cases} \min & \theta(x) \\ \text{s.t.} & g(x) \leq 0, \\ & h(x) = 0, \end{cases} \tag{6}$$

where $x \in \mathbf{R}^n$, θ is real valued, g is an m-dimensional vector, and h is a p-dimensional vector. Assume that θ, g, and h are C^2. The *Kuhn–Tucker necessary optimality conditions* for (6) are (cf. also ► Equality-constrained nonlinear programming: KKT necessary optimality conditions):

$$\begin{cases} \nabla\theta(x) + \beta^\top \nabla h(x) + \mu^\top \nabla g(x) = 0, \\ h(x) = 0, \\ g(x) \leq 0, \\ \mu \geq 0, \\ \mu^\top g(x) = 0, \end{cases} \tag{7}$$

where $\beta \in \mathbf{R}^p$ and $\mu \in \mathbf{R}^m$. The complementarity conditions $\mu \geq 0$, $g(x) \leq 0$, $\mu^\top g(x) = 0$ are replaced by the equivalent nonlinear system of equations

$$W(x, \mu) = 0, \tag{8}$$

where

$$W_i(x, \mu) = -|\mu_i + g_i(x)|^3 + \mu_i^3 - (g_i(x))^3, \quad i = 1, \ldots, m. \tag{9}$$

Thus the optimality conditions (7) take the form

$$F(x, \beta, \mu) = \begin{pmatrix} [\nabla\theta(x) + \beta^\top \nabla h(x) + \mu^\top \nabla g(x)]^\top \\ h(x) \\ W(x, \mu) \end{pmatrix} = 0. \tag{10}$$

With $z = (x, \beta, \mu)$, the proposed homotopy map is

$$\rho_a(\lambda, z) = \lambda F(z) + (1 - \lambda)(z - a), \tag{11}$$

where $a \in \mathbf{R}^{n+p+m}$. Simple conditions on θ, g, and h guaranteeing that the above homotopy map $\rho_a(\lambda, z)$ will work are unknown, although this map has worked very well on some difficult realistic engineering problems.

Globally Convergent Homotopy Methods, Table 1
Taxonomy of homotopy subroutines

$x = f(x)$		$F(x) = 0$		$\rho(a, \lambda, x) = 0$		algorithm
dense	sparse	dense	sparse	dense	sparse	
FIXPDF	FIXPDS	FIXPDF	FIXPDS	FIXPDF	FIXPDS	ordinary differential equation
FIXPNF	FIXPNS	FIXPNF	FIXPNS	FIXPNF	FIXPNS	normal flow
FIXPQF	FIXPQS	FIXPQF	FIXPQS	FIXPQF	FIXPQS	augmented Jacobian matrix

Frequently in practice the functions θ, g, and h involve a parameter vector c, and a solution to (6) is known for some $c = c^{(0)}$. Suppose that the problem under consideration has parameter vector $c = c^{(1)}$. Then

$$c = (1 - \lambda)c^{(0)} + \lambda c^{(1)} \tag{12}$$

parametrizes c by λ and $\theta = \theta(x;c) = \theta(x;c(\lambda))$, $g = g(x;c(\lambda))$, $h = h(x;c(\lambda))$. The optimality conditions in (10) become functions of λ as well, $F(\lambda, x, \beta, \mu) = 0$, and

$$\rho_a(\lambda, z) = \lambda F(\lambda, z) + (1 - \lambda)(z - a) \tag{13}$$

is a highly implicit nonlinear function of λ. If $F(0, z^{(0)}) = 0$, a good choice for a in practice has been found to be $a = z^{(0)}$. A natural choice for a homotopy would be simply

$$F(\lambda, z) = 0, \tag{14}$$

since the solution $z^{(0)}$ to $F(0, z) = 0$ (the problem corresponding to $c = c^{(0)}$) is known. However, for various technical reasons, (13) is much better than (14).

Software

There are several software packages implementing both continuous and simplicial homotopy methods; see [1] and [6] for a discussion of some of these packages. A production quality software package written in Fortran 90 is described here. *HOMPACK90* [6] is a Fortran 90 collection of codes for finding zeros or fixed points of nonlinear systems using globally convergent probability-one homotopy algorithms. Three qualitatively different algorithms (ordinary differential equation based, normal flow, quasi-Newton augmented Jacobian matrix) are provided for tracking homotopy zero curves, as well as separate routines for dense and sparse Jacobian matrices. A high level driver for the special case of polynomial systems is also provided. HOMPACK90 features elegant interfaces, use of modules, support for several sparse matrix data structures, and modern iterative algorithms for large sparse Jacobian matrices.

HOMPACK90 is logically organized in two different ways: by algorithm/problem type and by subroutine level. There are three levels of subroutines. The top level consists of drivers, one for each problem type and algorithm type. The second subroutine level implements the major components of the algorithms such as stepping along the homotopy zero curve, computing tangents, and the end game for the solution at $\lambda = 1$. The third subroutine level handles high level numerical linear algebra such as QR factorization, and includes some LAPACK and BLAS routines. The organization of HOMPACK90 by algorithm/problem type is shown in Table 1, which lists the driver name for each algorithm and problem type.

The naming convention is

$$FIXP \begin{Bmatrix} D \\ N \\ Q \end{Bmatrix} \begin{Bmatrix} F \\ S \end{Bmatrix},$$

where $D \approx$ ordinary differential equation algorithm, $N \approx$ normal flow algorithm, $Q \approx$ quasi-Newton augmented Jacobian matrix algorithm, $F \approx$ dense Jacobian matrix, and $S \approx$ sparse Jacobian matrix. Depending on the problem type and the driver chosen, the user must write exactly two subroutines, whose interfaces are specified in the module HOMOTOPY, defining the problem (f or ρ). The module REAL_PRECISION specifies the real numeric model with

SELECTED_REAL_KIND(13),

which will result in 64-bit real arithmetic on a Cray, DEC VAX, and IEEE 754 Standard compliant hardware.

The special purpose polynomial system solver POLSYS1H can find all solutions in complex projective space of a *polynomial system of equations*. Since a polynomial programming problem (where the objective function, inequality constraints, and equality constraints are all in terms of polynomials) can be formulated as a polynomial system of equations, POLSYS1H can effectively find the *global optimum* of a polynomial program. However, polynomial systems can have a huge number of solutions, so this approach is only practical for small polynomial programs (e. g., surface intersection problems that arise in CAD/CAM modeling).

The organization of the Fortran 90 code into modules gives an object oriented flavor to the package. For instance, all of the drivers are encapsulated in a single MODULE HOMPACK90. The user's calling program would then simply contain a statement like

USE HOMPACK90, ONLY : FIXPNF

Many scientific programmers prefer the reverse call paradigm, whereby a subroutine returns to the calling program whenever the subroutine needs certain information (e. g., a function value) or a certain operation performed (e. g., a matrix-vector multiply). Two reverse call subroutines (STEPNX, ROOTNX) are provided for 'expert' users. STEPNX is an expert reverse call stepping routine for tracking a homotopy zero curve γ that returns to the caller for all linear algebra, all function and derivative values, and can deal gracefully with situations such as the function being undefined at the requested steplength.

ROOTNX provides an expert reverse call end game routine that finds a point on the zero curve where $g(\lambda, x) = 0$, as opposed to just the point where $\lambda = 1$. Thus ROOTNX can find turning points, bifurcation points, and other 'special' points along the zero curve. The combination of STEPNX and ROOTNX provide considerable flexibility for an expert user.

See also

- ► Parametric Optimization: Embeddings, Path Following and Singularities
- ► Topology of Global Optimization

References

1. Allgower EL, Georg K (1990) Numerical continuation methods. Springer, Berlin
2. Chow SN, Mallet-Paret J, Yorke JA (1978) Finding zeros of maps: homotopy methods that are constructive with probability one. Math Comput 32:887–899
3. Forster W (1980) Numerical solution of highly nonlinear problems. North-Holland, Amsterdam
4. Watson LT (1986) Numerical linear algebra aspects of globally convergent homotopy methods. SIAM Rev 28:529–545
5. Watson LT (1990) Globally convergent homotopy algorithms for nonlinear systems of equations. Nonlinear Dynamics 1:143–191
6. Watson LT, Haftka RT (1989) Modern homotopy methods in optimization. Comput Methods Appl Mechanics Engrg 74:289–305
7. Watson LT, Sosonkina M, Melville RC, Morgan AP, Walker HF (1997) Algorithm 777: HOMPACK90: A suite of Fortran 90 codes for globally convergent homotopy algorithms. ACM Trans Math Softw 23:514–549

Global Optimization Algorithms for Financial Planning Problems

PANOS PARPAS, BERÇ RUSTEM
Department of Computing, Imperial College,
London, UK

MSC2000: 90B50, 78M50, 91B28

Article Outline

Abstract

It is becoming apparent that convex financial planning models are at times a poor approximation of the real world. More realistic, and more relevant, models need to dispense with normality assumptions and concavity of the utility functions to be optimized. Moreover, the problems are large scale but structured; consequently specialized algorithms have been proposed for their solution. The aim of this article is to discuss a non-

convex portfolio-selection problem and describe algorithms that can be used for its solution.

Background

Modern portfolio theory started in the 1950s with H. Markowitz's work [16,17]. Since then a lot of research has been done in improving the basic models and dispensing with the limiting assumptions of the field. The aim of this article is to introduce the problem of optimization of higher-order moments of a portfolio. This model is an extension of the celebrated mean-variance model of Markowitz [16,17]. The inclusion of higher-order moments has been proposed as one possible augmentation to the model in order to make it more applicable. The applicability of the model can be broadened by relaxing one of its major assumptions, i. e. that the rate of returns are normal. In order to solve the portfolio-selection problem, we first need to address the problem of scenario generation, i. e. the description of the uncertainties used in the portfolio-selection problem. Both problems are non-convex, large-scale, and highly relevant in financial optimization.

We focus on a single-period model where the decision maker (DM) provides as input preferences with respect to mean, variance, skewness and possibly kurtosis of the portfolio. Using these four parameters we then formulate the multicriterion optimization problem as a standard non-linear programming problem. This version of the decision model is a non-convex linearly constrained problem.

Before we can solve the portfolio-selection problem we need to describe the uncertainties regarding the returns of the risky assets. In particular we need to specify: (1) the possible states of the world and (2) the probability of each state. A common approach to this modelling problem is the method of matching moments (see e. g. [5,9,20]). The first step in this approach is to use the historical data to estimate the moments (in this paper we consider the first four central moments, i. e. mean, variance, skewness and kurtosis). The second step is to compute a discrete distribution with the same statistical properties as those calculated in the previous step. Given that our interest is on real-world applications, we recognize that there may not always be a distribution that matches the calculated statistical properties. For this reason we formulate the problem as a least-squares problem [5,9]. The rationale behind this formulation is that we try to calculate a description of the uncertainty that matches our beliefs as well as possible. The scenario-generation problem also has a non-convex objective function and is linearly constrained.

For the two problems described above we apply a new stochastic global optimization algorithm that has been developed specifically for this class of problems. The algorithm is described in [19]. It is an extension of the constrained case of the so-called diffusion algorithm [1,4,6,7]. The method follows the trajectory of an appropriately defined stochastic differential equation (SDE). Feasibility of the trajectory is achieved by projecting its dynamics onto the set defined by the linear equality constraints. A barrier term is used for the purpose of forcing the trajectory to stay within any bound constraints (e. g. positivity of the probabilities, or bounds on how much of each asset to own).

A review of applications of global optimization to portfolio selection problems appeared in [13]. A deterministic global optimization algorithm for a multiperiod model appeared in [15]. This article complements the work mentioned above in the sense that we describe a complete framework for the solution of a realistic financial model. The type of models we consider, due to the large number of variables, cannot be solved by deterministic algorithms. Consequently, practitioners are left with two options: solve a simpler, but less relevant, model or use a heuristic algorithm (e. g. tabu-search or evolutionary algorithms). The approach proposed in this paper lies somewhere in the middle. The proposed algorithm belongs to the simulated-annealing family of algorithms, and it has been shown in [19] that it converges to the global optimum (in a probabilistic sense). Moreover, the computational experience reported in [19] seems to indicate that the method is robust (in terms of finding the global optimum) and reliable. We believe that such an approach will be useful in many practical applications.

Models

Scenario Generation

From its inception stochastic programming (SP) has found several diverse applications as an effective paradigm for modelling decisions under uncertainty. The focus of initial research was on developing effective algorithms for models of realistic size. An area that has only recently received attention is on methods to represent the uncertainties of the decision problem.

A review of available methods to generate meaningful descriptions of the uncertainties from data can be found in [5]. We will use a least-squares formulation (see e. g. [5,9]). It is motivated by the practical concern that the moments, given as input, may be inconsistent. Consequently, the best one can do is to find a distribution that fits the available data as well as possible. It is further assumed that the distribution is discrete. Under these assumptions the problem can be written as

$$\begin{aligned}&\min_{\omega,p} \sum_{i=1}^{n} \Big(\sum_{j=1}^{k} p_j m_i(\omega_j) - \mu_i \Big)^2 \\ &\text{s.t} \sum_{j=1}^{k} p_j = 1 p_j \geq 0 j = 1,\ldots,k,\end{aligned}$$

where μ_i represents the statistical properties of interest and $m_i(\cdot)$ is the associated 'moment' function. For example, if μ_i is the target mean for the ith asset, then $m_i(\omega_j) = \omega_j^i$ i. e. the jth realization of the ith asset. Numerical experiments using this approach for a multistage model were reported in [9] (without arbitrage considerations). Other methods such as maximum entropy [18] and semidefinite programming [2] enjoy strong theoretical properties but cannot be used when the data of the problem are inconsistent. A disadvantage of the least-squares model is that it is highly nonconvex, which makes it very difficult to handle numerically. These considerations lead to the development of the algorithm described in Sect. "A Stochastic Optimization Algorithm" (see also [19]) that can efficiently compute global optima for problems in this class.

When using scenario trees for financial planning problems it becomes necessary to address the issue of arbitrage opportunities [9,12]. An arbitrage opportunity is a self-financing trading strategy that generates a strictly positive cash flow in at least one state and whose payoffs are non-negative in all other states. In other words, it is possible to get something for nothing. In our implementation we eliminate arbitrage opportunities by computing a sufficient set of states so that the resulting scenario tree has the arbitrage-free property. This is achieved by a simple two-step process. In the first step we generate random rates of returns; these are sampled by a uniform distribution. We then test for arbitrage by solving the system

$$\begin{aligned}&x_0^i = e^{-r} \sum_{j=1}^{m} x_j^i \pi_j, \quad \sum_{j=1}^{m} \pi_j = 1, \pi_j \geq 0, \\ &\qquad j = 1,\ldots,m \quad i = 1,\ldots,n\,,\end{aligned} \tag{1}$$

where x_0^i represents the current (known) state of the world for the ith asset and x_j^i represents the jth realization of the ith asset in the next time period (these are generated by the simulations mentioned above). r is the riskless rate of return. The π_j are called the risk-neutral probabilities. According to a fundamental result of Harisson and Kerps [10], the existence of the risk-neutral probabilities is enough to guarantee that the scenario tree has the desired property. In the second step, we solve the least-squares problem with some of the states fixed to the states calculated in the first step. In other words, we solve the following problem:

$$\begin{aligned}&\min_{\omega,p} \sum_{i=1}^{n} \Big(\sum_{j=1}^{k} p_j m_i(\omega_j) + \sum_{l=1}^{m} p_l m_i(\hat{\omega}_l) - \mu_i \Big)^2 \\ &\text{s.t} \sum_{j=1}^{k+m} \quad p_j = 1 p_j \geq 0 \quad j = 1,\ldots,k+m\,.\end{aligned} \tag{2}$$

In the problem above, $\hat{\omega}$ are fixed. Solving the preceding problem guarantees a scenario tree that is arbitrage free.

Portfolio Selection

In this section we describe the portfolio-selection problem when higher-order terms are taken into account. The classical mean–variance approach to portfolio analysis seeks to balance risk (measured by variance) and reward (measured by expected value). There are many ways to specify the single-period problem. We will be using the following basic model:

$$\begin{aligned}&\min_{w} -\alpha \mathbb{E}[w] + \beta \mathbb{V}[w] \\ &\text{s.t} \sum_{i=1}^{n} w_i = 1 \quad l_i \leq w_i \leq u_i \quad i = 1,\ldots,n\,,\end{aligned} \tag{3}$$

where $\mathbb{E}[\cdot]$ and $\mathbb{V}[\cdot]$ represent the mean rate of return and its variance respectively. The single constraint is known as the *budget constraint* and it specifies the initial wealth (without loss of generality we have assumed that this is one). The α and β are positive scalars and are chosen so that $\alpha + \beta = 1$. They specify the DM's preferences, i. e. $\alpha = 1$ means that the DM is risk seeking, while $\beta = 1$ implies that the DM is risk averse. Any other selection of the parameters will produce a point on the efficient frontier. The decision variable (w) represents the commitment of the DM to a particular asset. Note that this problem is a convex quadratic programming problem for which very efficient algorithms exist. The interested reader is referred to the review in [23] for more information regarding the Markowitz model.

We propose an extension of the mean–variance model using higher-order moments. The vector-optimization problem can be formulated as a standard non-convex optimization problem using two additional scalars to act as weights. These weights are used to enforce the DM's preferences. The problem is then formulated as follows:

$$\begin{aligned} &\min_w -\alpha\mathbb{E}[w] + \beta\mathbb{V}[w] - \gamma\mathbb{S}[w] + \delta\mathbb{K}[w] \\ &\text{s.t} \sum_{i=1}^{n} w_i = 1 \quad l_i \le w_i \le u_i \quad i = 1, \ldots, n\,, \end{aligned} \tag{4}$$

where $\mathbb{S}[\cdot]$ and $\mathbb{K}[\cdot]$ represent the skewness and kurtosis of the rate of return respectively. γ and δ are positive scalars. The four scalar parameters are chosen so that they sum to one. Positive skewness is desirable (since it corresponds to higher returns, albeit with low probability), while kurtosis is undesirable since it implies that the DM is exposed to more risk. The model in (4) can be extended to multiple periods while maintaining the same structure (non-convex objective and linear constraints). The numerical solution of (2) and (4) will be discussed in the next section.

Methods

A Stochastic Optimization Algorithm

The models described in the previous section can be written as:

$$\begin{aligned} &\min_x f(x) \\ &\quad \text{s.t } Ax = b \\ &\qquad x \ge 0\,. \end{aligned}$$

A well-known method for obtaining a solution to an unconstrained optimization problem is to consider the following ordinary differential equation (ODE):

$$\mathrm{d}X(t) = -\nabla f(X(t))\,\mathrm{d}t\,. \tag{5}$$

By studying the behaviour of $X(t)$ for large t, it can be shown that $X(t)$ will eventually converge to a stationary point of the unconstrained problem. A review of so-called continuous-path methods can be found in [25]. A deficiency of using (5) to solve optimization problems is that it will get trapped in local minima. To allow the trajectory to escape from local minima, it has been proposed by various authors (e. g. [1,4,6,7]) to add a stochastic term that would allow the trajectory to 'climb' hills. One possible augmentation to (5) that would enable us to escape from local minima is to add noise. One then considers the *diffusion process*:

$$\mathrm{d}X(t) = -\nabla f(X(t))\,\mathrm{d}t + \sqrt{2T(t)}\,\mathrm{d}B(t)\,, \tag{6}$$

where $B(t)$ is the standard Brownian motion in $\mathbb{R}^n$. It has been shown in [4,6,7], under appropriate conditions on f and $T(t)$, that as $t \to \infty$, the transition probability of $X(t)$ converges to a probability measure Π. The latter has its support on the set of global minimizers.

For the sake of argument, suppose we did not have any linear constraints but only positivity constraints. We could then consider enforcing the feasibility of the iterates by using a barrier function. According to the algorithmic framework sketched out above, we could obtain a solution to our (simplified) problem by following the trajectory of the following SDE:

$$\mathrm{d}X(t) = -\nabla f(X(t))\,\mathrm{d}t + \mu X(t)^{-1}\,\mathrm{d}t + \sqrt{2T(t)}\,\mathrm{d}B(t)\,, \tag{7}$$

where $\mu > 0$ is the barrier parameter. By $X^{\text{-}1}$ we will denote an n-dimensional vector whose ith component is given by $1/X_i$. Having used a barrier function to deal with the positivity constraints, we can now introduce the linear constraints into our SDE@. This process has been carried out in [19] using the projected SDE:

$$\mathrm{d}X(t) = P[-\nabla f(X(t)) + \mu X(t)^{-1}]\,\mathrm{d}t + \sqrt{2T(t)}P\,\mathrm{d}B(t)\,, \tag{8}$$

where $P = I - A^T(AA^T)^{-1}A$. The proposed algorithm works in a similar manner to gradient-projection algorithms. The key difference is the addition of a barrier parameter for the positivity of the iterates and a stochastic term that helps the algorithm escape from local minima.

The global optimization problem can be solved by fixing μ and following the trajectory of (8) for a suitably defined function $T(t)$. After sufficient time passes, we reduce μ and repeat the process. The proof that following the trajectory of (8) will eventually lead us to the global minimum appears in [19]. Note that the projection matrix for the type of constraints we need to impose for our models is particularly simple. For a constraint of the type $\sum_{i=1}^{n} x_i = 1$ the projection matrix is given by

$$P_{ij} = \begin{cases} -\frac{1}{n} & \text{if } i \neq j, \\ \frac{n-1}{n} & \text{otherwise.} \end{cases}$$

Other Methods

In this article we have focused on the numerical solution of a financial planning problem using a stochastic algorithm. We end this article by briefly discussing other possible approaches. Only stochastic methods will be discussed; for deterministic methods we refer the interested reader to [13].

Two-phase methods: Methods belonging to this class, as the name suggests, have two phases: a local and global phase. In the global phase, the feasible region is uniformly sampled. From each feasible point a local optimization algorithm is started. The later process is the local phase. This basic algorithmic framework has been modified to improve its performance by various authors. Improving this type of method requires careful selection of the sample points from which to start the local optimizations. Inevitably there is some compromise between computational efficiency and theoretical convergence. For a review of two-phase methods we refer the reader to [21] and references therein.

Simulated annealing (SA): This family of algorithms was inspired by the physical behaviour of atoms in a liquid. The method was independently proposed by Cerny[3] and Kirkpatrick et al. [11]. The method is inspired by a fundamental question of statistical mechanics concerning the behaviour of the system in low temperatures. For example, will the atoms remain fluid or will they solidify? If they solidify, do they form a crystalline solid or a glass? It turns out [11] that if the temperature is decreased slowly, then they form a pure crystal; this state corresponds to the minimum energy of the system. If the temperature is decreased too quickly, then they form a crystal with many defects. SA algorithms generate a point from some distribution. Whether to accept the new point or not is decided by an acceptance function. The latter function is 'temperature' dependent. At high temperatures the function is likely to accept the new point, while at low temperatures only points close to the global optimum value are supposed to be accepted. As can be anticipated, the performance of the algorithm depends on the annealing schedule, i. e. how fast the temperature is reduced. Performance also depends on how points are sampled, the acceptance function and, of course, the stopping conditions. An excellent review article for SA is [14].

Stochastic adaptive search methods: These types of algorithms have strong theoretical properties but present challenging implementation issues. A typical algorithm from this class is the pure adaptive search method. This method works like a pure random search method but with the additional assumption of the ability to sample from a distribution that gives realizations that are strictly better than the incumbent. There exist many variants and combinations of this type of method, and an excellent review of them is given in [24].

Genetic algorithms: This class of algorithms has been inspired by concepts from evolutionary biology and from aspects of natural selection. There are two phases in these algorithms: generation of the population and updating. During the generation phase, candidate points (offsprings) are generated by sampling a p.d.f. This p.d.f. is usually specified from the original or the previous generation (the parents). In the second phase the population is updated. This update is performed by applying a selection mechanism and performing mutation operations on the population. There are very few theoretical results concerning the convergence properties of genetic algorithms. However, if their success in applications is anything to go by, then more attention needs to devoted to convergence aspects of the method. An excellent review of genetic algorithms is given in [22].

Tabu search: This is another heuristic algorithm that has been successfully used for global optimization (especially combinatorial problems) but lacks theoretical backing. This class of algorithms was proposed by Glover, and a review of the method appeared in [8]. The algorithm has three phases: preliminary search, intensification, and diversification. In the first phase, the algorithm takes the current configuration, examines neighbouring solutions, and selects the one with the best objective function value. This process is continued until no improving state can be identified. At this stage the possibility of returning to this point is ruled out by placing it into a list. This list is called the tabu list. In the second phase (intensification), the tabu list is cleared and the algorithm returns to the first phase. In the final stage (diversification), the most frequent moves that were placed into the tabu list during the first phase are placed from the start into the list. The algorithm then starts from a random initial point. In this phase the algorithm is not allowed to make any moves that are in the tabu list.

References

1. Aluffi-Pentini F, Parisi V, Zirilli F (1985) Global optimization and stochastic differential equations. J Optim Theory Appl 47(1):1–16
2. Bertsimas D, Sethuraman J (2000) Moment problems and semidefinite optimization. In: Handbook of semidefinite programming, vol 27. Int Ser Oper Res Manage Sci. Kluwer, Boston, pp 469–509
3. Černý V (1985) Thermodynamical approach to the traveling salesman problem: an efficient simulation algorithm. J Optim Theory Appl 45(1):41–51
4. Chiang TS, Hwang CR, Sheu SJ (1987) Diffusion for global optimization in R^n. SIAM J Control Optim 25(3):737–753
5. Dupacova J, Consigli G, Wallace SW (2000) Scenarios for multistage stochastic programs. Ann Oper Res 100:25–53 (2001)
6. Geman S, Hwang CR (1986) Diffusions for global optimization. SIAM J Control Optim 24(5):1031–1043
7. Gidas B (1986) The Langevin equation as a global minimization algorithm. In: Disordered systems and biological organization (Les Houches, 1985), vol 20. NATO Adv Sci Inst Ser F Comput Syst Sci. Springer, Berlin, pp 321–326
8. Glover F, Laguna M (1998) Tabu search. In: Handbook of combinatorial optimization, vol. 3. Kluwer, Boston, pp 621–757
9. Gülpınar N, Rustem B, Settergren R (2004) Simulation and optimization approaches to scenario tree generation. J Econ Dyn Control 28(7):1291–1315
10. Harrison JM, Kreps DM (1979) Martingales and arbitrage in multiperiod securities markets. J Econom Theory 20:381–408
11. Kirkpatrick S, Gelatt CD Jr, Vecchi MP (1983) Optimization by simulated annealing. Science 220(4598):671–680
12. Klaassen P (1997) Discretized reality and spurious profits in stochastic programming models for asset/liability management. Eur J Oper Res 101(2):374–392
13. Konno H (2005) Applications of global optimization to portfolio analysis. In: Audet C, Hansen P, Savard G (eds) Essays and Surveys in Global Optimization. Springer, Berlin, pp 195–210
14. Locatelli M (2002) Simulated annealing algorithms for continuous global optimization. In: Handbook of global optimization, vol 2, vol 62. Nonconvex Optim Appl. Kluwer, Dordrecht, pp 179–229
15. Maranas CD, Androulakis IP, Floudas CA, Berger AJ, Mulvey JM (1997) Solving long-term financial planning problems via global optimization. J Econ. Dynam Control 21(8–9):1405–1425 Computational financial modeling.
16. Markowitz HM (1952) Portfolio selection. J Finance 7:77–91
17. Markowitz HM (1952) The utility of wealth. J Polit Econ (60):151–158
18. Parpas P (2006) Algorithms in Stochastic Optimization. PhD Thesis, Imperial College London, May 2006
19. Parpas P, Rustem B, Pistikopoulos EN (2006) Linearly constrained global optimization and stochastic differential equations. J Global Optim 36(2):191–217
20. Prékopa A (1995) Stochastic programming, vol 324. Math Appl. Kluwer, Dordrecht
21. Schoen F (2002) Two-phase methods for global optimization. In: Handbook of global optimization, vol 2, vol 62. Nonconvex Optim Appl. Kluwer, Dordrecht, pp 151–177
22. Smith JE (2002) Genetic algorithms. In: Handbook of global optimization, vol 2, vol 62. Nonconvex Optim Appl. Kluwer, Dordrecht, pp 275–362
23. Steinbach MC (2001) Markowitz revisited: mean-variance models in financial portfolio analysis. SIAM Rev 43(1):31–85
24. Wood GR, Zabinsky ZB (2002) Stochastic adaptive search. In: Handbook of global optimization, vol 2, vol 62. Nonconvex Optim Appl. Kluwer, Dordrecht, pp 231–249
25. Zirilli F (1982) The use of ordinary differential equations in the solution of nonlinear systems of equations. In: Nonlinear optimization, 1981 (Cambridge, 1981), NATO Conf Ser II: Syst Sci. Academic, London, pp 39–46

Global Optimization in the Analysis and Management of Environmental Systems

JÁNOS D. PINTÉR
Pintér Consulting Services, Inc.,
and Dalhousie University, Halifax, Canada

MSC2000: 90C05

Article Outline

Keywords

Nonlinear decision models; Multi-extremality; Continuous global optimization; Applications in environmental systems modeling and management

Environmental Systems Analysis and Optimization

The harmonized consideration of technical, economic and environmental objectives in strategic planning and operational decision making is of paramount importance, on a worldwide scale. Environmental quality issues are of serious concern even in the most developed countries, although direct pollution control expenditures are typically in the 2–3 percent range of their gross domestic product. The 'optimized' or at least 'acceptable' solution of environmental quality problems requires the combination of knowledge from a multitude of areas, and requires an interdisciplinary effort.

In the past decades, mathematical programming (MP) models have been applied also to the analysis and management of environmental systems. The annotated bibliography [9] reviews over 350 works, including some thirty books. Note further that the engineering, economic and environmental science literature contains a very large amount of work that can serve as a basis and therefore is closely related to such modeling efforts. For instance, the classic textbook [28] reviews the basic quantitative models applied in describing physical, chemical and biological phenomena of relevance. A more recent exposition (with a somewhat broader scope) is presented in, for instance, [11]. The chapters in the latter edited volume discuss the following issues:

- environmental crisis, as a multidisciplinary challenge;
- soil pollution;
- air pollution;
- water pollution;
- water resources management;
- pesticides;
- gene technology;
- landscape planning;
- environmental economics;
- ecological aspects;
- environmental impact assessment;
- environmental management models.

Environmental management models are discussed – in the broader context of governmental planning and operations – already in [8]. In addition to items listed above, the (relevant) topics covered include also

- solid waste management;
- urban development;
- policy analysis.

Numerous further books can be mentioned; with varying emphasis on environmental science, engineering, economics or systems analysis. Consult, e. g., [1,2,3,4,6,10,13,15,16,17,18,19,23,24,25,29,31,32,33]. Most of these works also provide extensive lists of additional references.

In the framework of this short article there is no room to go into any detailed discussion of environmental models. Therefore we shall only emphasize one important methodological aspect reflected by the title: namely, the relevance of global optimization in this context.

The predominant majority of MP models presented, e. g., in the books listed or in [9] belong to (continuous or possibly mixed integer) linear programming, or to convex nonlinear programming, with additional – usually rather simplified – considerations regarding system stochasticity. At the same time, more detailed or more realistic models of natural systems and their governing processes often possess high (explicit or hidden) high nonlinearity. For instance, one may think of power laws, periodic or chaotic processes, and (semi)random fluctuations, reflected by many natural objects on various scales: mountains, waters, plants, animals, and so on. For related far-reaching discussions, consult, for example, [5,7,20,21], or [30]. Since many natural objects and processes are inherently nonlinear, management models that optimize the behavior of environmental systems frequently lead to multi-extremal decision problems. Continuous global optimization (GO) is aimed at finding the 'absolutely best' solution of such models, in the possible presence of many other (locally optimal) solutions of various quality. See ► Continuous global optimization: Models, algorithms and software and ► Continuous global optimization: Applications for a number of textbooks and WWW sites related to the subject of GO. Therefore, here we mention only the handbook [14] and the WWW site [22].

We shall illustrate the relevance of GO by two very general examples, adapted from [26]. The latter book presents also a number of other case studies related to environmental modeling and management, with numerous additional references pertinent to this subject.

Model Calibration

The incomplete or poor understanding of environmental – as well as many other complex – systems calls for descriptive model development as an essential tool of the related research. The following main phases of quantitative systems modeling can be distinguished:

- identification: formulation of principal modeling objectives, determination (selection) of suitable model structure;
- calibration: (inverse) model fitting to available data and background information;
- validation and application in analysis, forecasting, control, management.

Consequently, the 'adequate' or 'best' parameterization of descriptive models is an important stage in the process of understanding environmental systems. Interesting, practically motivated discussions of the model calibration problem are presented also in [1,3,12,32].

A fairly simple and commonly applied instance of the model calibration problem can be stated as follows. Given

- a descriptive system model (e. g. of a lake, river, groundwater or atmospheric system) that depends on certain unknown (physical, chemical) parameters; their vector is denoted by x;
- the set of a priori feasible parameterizations D;
- the model output values $y_t^{(m)} = y_t^{(m)}(x)$ at time moments $t = 1, \ldots, T$;
- a set of corresponding observations y_t at $t = 1, \ldots, T$;
- a discrepancy measure denoted by f which expresses the distance between $y_t^{(m)}$ and y_t.

Then the optimized model calibration problem can be formulated as

$$\begin{cases} \min & f(x) := f\{y_t^{(m)}(x), y_t\} \\ \text{s.t.} & x \in D. \end{cases} \tag{1}$$

Frequently, D is a finite n-interval (a 'box'); furthermore, f is a continuous or somewhat more special (smooth, Lipschitz, etc.) function. Additional structural assumptions regarding f may be difficult to postulate, due to the following reason. For each fixed parameter vector x, the model output sequence $\{y_t^{(m)}(x)\}$ may be produced by some implicit formulas, or by a computationally demanding numerical procedure (such as e. g., the solution of a system of partial differential equations). Consequently, although model (1) most typically belongs to the general class of continuous GO problems, a more specific classification may be difficult to provide. Therefore one needs to apply a GO procedure that enables the solution of the calibration problem under the very general conditions outlined above.

To conclude the brief discussion of this example, note that in [26] several variants of the calibration problem statement are studied in detail. Namely, the model development and solver system LGO is applied to solve model calibration problems related to water quality analysis in rivers and lakes, river flow hydraulics, and aquifer modeling. (More recent implementations of LGO are described elsewhere: consult, e. g., [27].)

'Black Box' Optimization (in Environmental Systems)

As outlined above, the more realistic – as opposed to strongly simplified – analysis of environmental processes frequently requires the development of sophisticated systems of (sub)models: these are then connected to a suitable optimization modeling framework. For examples of various complexity, consult [1,2,10,19,32]. We shall illustrate this point by briefly discussing a modeling framework for river water quality management: for additional details, see [26] and references therein.

Assume that the ambient water quality in a river at time t is characterized by a certain vector $s(t)$. The components in $s(t)$ can include, for instance the following: suspended solids concentration, dissolved oxygen concentration, biological oxygen demand, chemical oxygen demand, concentrations of micro-pollutants and heavy metals, and so on. Naturally, the resulting water quality is influenced by a number of factors. These include the often stochastically fluctuating (discharge or nonpoint source) pollution load, as well as the regional hydro-meteorological conditions (streamflow rate, water temperature, etc). Some of these factors can be directly observed, while some others may not be completely known. In a typical model development process, submodels are constructed to describe all physical, chemical, biological, and ecological processes of relevance. (As for an example, one can refer to the classical Streeter–Phelps differential equations that approximate the longitudinal evolution of biological oxygen demand in a river; consult [25,28].)

In order to combine such system description with management models, one has to be able to evaluate all decision considered. Each given decision x can be related, inter alia, to the location and sizing of industrial and municipal wastewater treatment plants, the control of nonpoint source (agricultural) pollution, the design of a wastewater sewage collection network, the daily operation of these facilities, and so on. The analysis frequently involves the computationally intensive evaluation of environmental quality – e. g., by solving a system of (partial) differential equations – for each decision option considered. The quite (possibly) more realistic stochastic extensions of such models may also require the execution of Monte-Carlo simulation cycles. Under such or similar circumstances, environmental management models can be (very) complex consisting of a number of 'black box' submodels. Consequently, the following general conceptual modeling framework may, and often will, lead to multi-extremal model instances requiring the application of suitable GO techniques:

$$\begin{aligned}&\min\{\mathrm{TCEM}(x)\},\\ &\mathrm{EQ}_{\min} \le \mathrm{EQ}(x) \le \mathrm{EQ}_{\max},\\ &\mathrm{TF}_{\min} \le \mathrm{TF}(x) \le \mathrm{TF}_{\max},\end{aligned} \tag{2}$$

in which

- $\mathrm{TCEM}(x)$ is total (discounted, expected) costs of environmental management;
- $\mathrm{EQ}(x)$ is resulting environmental quality (vector);
- $\mathrm{EQ}_{\min}$ and $\mathrm{EQ}_{\max}$ are vector bounds on 'acceptable' environmental quality indicators;
- $\mathrm{TF}(x)$ are resulting technical system characteristics (vector);
- $\mathrm{TF}_{\min}$ and $\mathrm{TF}_{\max}$ are vector bounds on 'acceptable' technical characteristics.

Numerous other examples could be cited: similarly to the case considered above, they may involve the solution of systems of (algebraic, ordinary or partial differential) equations, and/or the statistical analysis of the environmental (model) system studied. For further examples – including data analysis, combination of expert opinions, environmental model calibration, industrial wastewater management, regional pollution management in rivers and lakes, risk assessment and control of accidental pollution – in the context of global optimization consult, e. g., [26], and references therein.

See also

- ▶ Continuous Global Optimization: Applications
- ▶ Continuous Global Optimization: Models, Algorithms and Software
- ▶ Interval Global Optimization
- ▶ Mixed Integer Nonlinear Programming
- ▶ Optimization in Water Resources

References

1. Beck MB (1985) Water quality management: A review of the development and application of mathematical models. Springer, Berlin
2. Beck MB (ed) (1987) Systems analysis in water quality management. Pergamon, Oxford
3. Beck MB, van Straten G (eds) (1983) Uncertainty and forecasting in water quality. Springer, Berlin
4. Bower BT (ed) (1977) Regional residuals environmental quality management. Johns Hopkins Univ. Press, Baltimore, MD
5. Casti JL (1990) Searching for certainty. Morrow, New York
6. Dorfman R, Jacoby HD, Thomas HA (eds) (1974) Models for managing regional water quality. Harvard Univ. Press, Cambridge, MA
7. Eigen M, Winkler R (1975) Das Spiel. Piper, Munich
8. Gass SI, Sisson RI (eds) (1974) A guide to models in governmental planning and operations. Environmental Protection Agency, Washington, DC
9. Greenberg HJ (1995) Mathematical programming models for environmental quality control. Oper Res 43:578–622
10. Haith DA (1982) Environmental systems optimization. Wiley, New York

11. Hansen PE, Jørgensen SE (eds) (1991) Introduction to environmental management. Elsevier, Amsterdam
12. Hendrix EMT (1998) Global optimization at work. PhD Thesis, LU Wageningen
13. Holling CS (ed) (1978) Adaptive environmental assessment and management. IIASA & Wiley, New York
14. Horst R, Pardalos PM (eds) (1995) Handbook of global optimization. Kluwer, Dordrecht
15. Jørgensen SE (ed) (1983) Applications of ecological modelling in environmental management. Elsevier, Amsterdam
16. Kleindorfer PR, Kunreuther HC (eds) (1987) Insuring and managing hazardous risks: From Seveso to Bhopal. Springer, Berlin
17. Kneese AV, Ayres RU, d'Arge RC (1970) Economics and the environment: A materials balance approach. Johns Hopkins Univ. Press, Baltimore, MD
18. Kneese AV, Bower BT (1968) Managing water quality: Economics, technology, institutions. Johns Hopkins Univ. Press, Baltimore, MD
19. Loucks DP, Stedinger JR, Haith DA (1981) Water resources systems planning and analysis. Prentice-Hall, Englewood Cliffs, NJ
20. Mandelbrot BB (1983) The fractal geometry of nature. Freeman, New York
21. Murray JD (1983) Mathematical biology. Springer, Berlin
22. Neumaier A (1999) Global optimization. http://solon.cma.univie.ac.at/~neum/glopt.html
23. Nijkamp P (1980) Environmental policy analysis: Operational methods and models. Wiley, New York
24. Novotny W, Chesters G (1982) Handbook of nonpoint pollution. v. Nostrand, Princeton, NJ
25. Orlob GT (1983) Mathematical modeling of water quality: Streams, lakes and reservoirs. Wiley, New York
26. Pintér JD (1996) Global optimization in action. Kluwer, Dordrecht
27. Pintér JD (1998) A model development system for global optimization. In: De Leone R, Murli A, Pardalos PM, Toraldo G (eds) High Performance Software for Nonlinear Optimization: Status and Perspectives. Kluwer, Dordrecht, pp 301–314
28. Rich LG (1972) Environmental systems engineering. McGraw-Hill, New York
29. Richardson ML (ed) (1988) Risk assessment of chemicals in the environment. The Royal Soc. Chemistry London, London
30. Schroeder M (1991) Fractals, chaos, power laws. Freeman, New York
31. Seneca JJ, Taussig MK (1974) Environmental economics. Prentice-Hall, Englewood Cliffs, NJ
32. Somlyódy L, van Straten G (eds) (1983) Modeling and managing shallow lake eutrophication. Springer, Berlin
33. United States Environmental Protection Agency (1988) Waste minimization opportunity assessment manual. Techn. Report EPA Cincinnati

Global Optimization: Application to Phase Equilibrium Problems

MARK A. STADTHERR
Department Chemical Engineering,
University Notre Dame, Notre Dame, USA

MSC2000: 80A10, 80A22, 90C90, 65H20

Article Outline

Keywords

Interval analysis; Global optimization; Phase equilibrium; Phase stability; Interval Newton

The reliable calculation of phase equilibrium for multicomponent mixtures is a critical aspect in the simulation, optimization and design of a wide variety of industrial processes, especially those involving separation operations such as distillation and extraction. It is also important in the simulation of enhanced oil recovery processes such as miscible or immiscible gas flooding. Unfortunately, however, even when accurate models of the necessary thermodynamic properties are available, it is often very difficult to actually solve the phase equilibrium problem reliably.

Background

The computation of phase equilibrium is often considered in two stages, as outlined by M.L. Michelsen [12, 13]. The first involves the *phase stability* problem, that is, to determine whether or not a given mixture will split into multiple phases. The second involves the *phase split* problem, that is to determine the amounts and compositions of the phases assumed to be present. After a phase split problem is solved it may be necessary to do phase stability analysis on the results to determine whether the postulated number of phases was correct, and if not repeat the phase split problem. Both the phase stability and phase split problems can be formulated as minimization problems, or as equivalent nonlinear equation solving problems.

For determining phase equilibrium at constant temperature and pressure, the most commonly considered case, a model of the Gibbs free energy of the system is required. This is usually based on an excess Gibbs energy model (activity coefficient model) or an equation of state model. At equilibrium the total Gibbs energy of the system is minimized. Phase stability analysis may be interpreted as a global optimality test that determines whether the phase being tested corresponds to a global optimum in the total Gibbs energy of the system. If it is determined that a phase will split, then a phase split problem is solved, which can be interpreted as finding a *local* minimum in the total Gibbs energy of the system. This local minimum can then be tested for global optimality using phase stability analysis. If necessary the phase split calculation must then be repeated, perhaps changing the number of phases assumed to be present, until a solution is found that meets the global optimality test. Clearly the correct solution of the phase stability problem, itself a global optimization problem, is the key in this two-stage global optimization procedure for phase equilibrium. As emphasized in [10], while it is possible to apply rigorous global optimization techniques directly to the phase equilibrium problem, it is computationally more efficient to use a two-stage approach such as outlined above, since the dimensionality of the global optimization problem that must be solved (phase stability problem) is less than that of the full phase equilibrium problem.

In solving the phase stability problem, the conventional solution methods are initialization dependent, and may fail by converging to trivial or nonphysical solutions or to a point that is a local but not a global minimum. Thus there is no guarantee that the phase equilibrium problem has been correctly solved. Because of the difficulties that may arise in solving phase equilibrium problems by standard methods (e. g., [12,13]), there has been significant interest in the development of more reliable methods. For example, the methods of A.C. Sun and W.D. Seider [16], who use a homotopy continuation approach, and of S.K. Wasylkiewicz, L.N. Sridhar, M.F. Malone and M.F. Doherty [18], who use an approach based on topological considerations, can offer significant improvements in reliability. C.M. McDonald and C.A. Floudas [7,8,9,10] show that, for certain activity coefficient models, the phase stability and equilibrium problems can be made amenable to solution by powerful global optimization techniques, which provide a mathematical guarantee of reliability.

An alternative approach for solving the phase stability problem, based on interval analysis, that provides both mathematical and computational guarantees of global optimality, was originally suggested by M.A. Stadtherr, C.A. Schnepper and J.F. Brennecke [15], who applied it in connection with activity coefficient models, as later done also in [11]. This technique, in particular the use of an *interval Newton* and *generalized bisection* algorithm, is initialization independent and can solve the phase stability problem with mathematical certainty, and, since it deals automatically with rounding error, with computational certainty as well. J.Z. Hua, Brennecke and Stadtherr [3,4,5,6] extended this method to problems modeled with cubic *equation of state* models, in particular the Van der Waals, Peng–Robinson, and Soave–Redlich–Kwong models. Though interval analysis provides a *general purpose* and *model independent* approach for guaranteed solution of the phase stability problem, the discussion below will focus on the use of cubic equation of state models.

Phase Stability Analysis

The determination of phase stability is often done using tangent plane analysis [1,12]. A phase at specified temperature T, pressure P, and feed mole fraction vector $\mathbf{z}$ is unstable and can split (in this context, 'unstable' refers to both the thermodynamically metastable and classically unstable cases), if the molar Gibbs energy of mixing surface $m(\mathbf{x}, v)$ ever falls below a plane tangent to the surface at $\mathbf{z}$. That is, if the tangent plane distance

$$D(\mathbf{x}, v) = m(\mathbf{x}, v) - m_0 - \sum_{i=1}^{n} \left(\frac{\partial m}{\partial x_i} \right)_0 (x_i - z_i)$$

is negative for any composition (mole fraction) vector $\mathbf{x}$, the phase is unstable. The subscript zero indicates evaluation at $\mathbf{x} = \mathbf{z}$, n is the number of components, and v is the molar volume of the mixture. A common approach for determining if D is ever negative is to minimize D subject to the mole fractions summing to one

$$1-\sum_{i=1}^{n} x_i = 0 \qquad (1)$$

and subject to the equation of state relating $\mathbf{x}$ and v:

$$P-\frac{RT}{v-b}+\frac{a}{v^2+ubv+wb^2}=0. \qquad (2)$$

Here a and b are functions of $\mathbf{x}$ determined by specified mixing rules. The 'standard' mixing rules are $b = \sum_{i=1}^{n} x_i b_i$ and $a = \sum_{i=1}^{n} \sum_{j=1}^{n} x_i x_j a_{ij}$, with $a_{ij} = (1-k_{ij})\sqrt{a_i a_j}$. The $a_i(T)$ and b_i are pure component properties determined from the system temperature T, the critical temperatures T_{ci}, the critical pressures P_{ci} and acentric factors ω_i. The binary interaction parameter k_{ij} is generally determined experimentally by fitting binary vapor-liquid equilibrium data. Equation (2) is a generalized cubic equation of state model. With the appropriate choice of u and w, common models such as Peng–Robinson ($u = 2$, $w = -1$), Soave–Redlich–Kwong ($u = 1$, $w = 0$), and Van der Waals ($u = 0$, $w = 0$) may be obtained. It is readily shown that the stationary points in this optimization problem must satisfy

$$s_i(\mathbf{x}, v) - s_i(\mathbf{z}, v_0) = 0, \quad i = 1, \dots, n-1, \qquad (3)$$

where

$$s_i = \left(\frac{\partial m}{\partial x_i}\right) - \left(\frac{\partial m}{\partial x_n}\right).$$

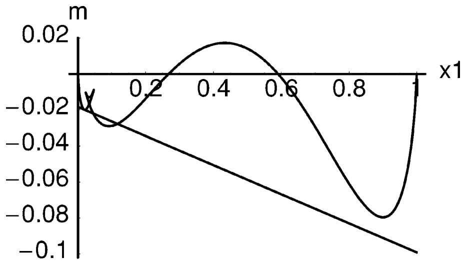
The $(n+1) \times (n+1)$ system given by equations (1), (2) and (3) above can be used to solve for the stationary points in the optimization problem.

The equation system for the stationary points has a trivial root at $(\mathbf{x}, v) = (\mathbf{z}, v_0)$ and frequently has multiple nontrivial roots as well. Thus conventional equation solving techniques may fail by converging to the trivial root or give an incorrect answer to the phase stability problem by converging to a stationary point that is not the global minimum of D. This is aptly demonstrated by the experiments of K.A. Green, S. Zhou and K.D. Luks [2], who show that the pattern of convergence from different initial guesses demonstrates a complex fractal-like behavior for even very simple models like Van der Waals. The problem is further complicated by the fact that the cubic equation of state (2) may have multiple real volume roots v.

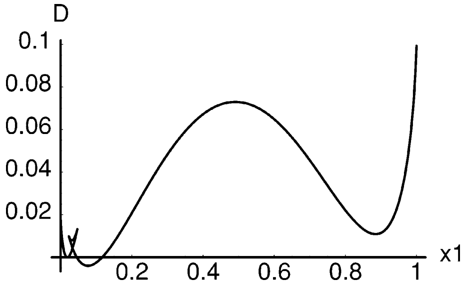
As an example of a system that causes numerical difficulties, consider the binary mixture of hydrogen sulfide (component 1) and methane (component 2) at a temperature of 190 K and pressure of 40.53 bar (40 atm) modeled using the Soave–Redlich–Kwong equation of state, and with an overall feed composition of $z_1 = 0.0187$. Figure 1 shows a plot of the reduced Gibbs energy of mixing m vs. x_1 for this system (in the reduced composition space where $x_2 = 1 - x_1$), and also shows the tangent at the feed composition.

The corresponding tangent plane distance function is shown in Fig. 2 and Fig. 3.

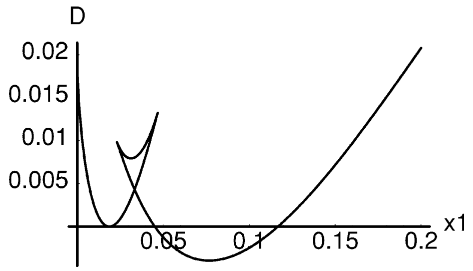
Note that this system has a region, around x_1 of 0.03 to 0.05, where multiple real volume roots occur and thus multiple values of m and D exist; only the lowest values are physically significant. This system has five stationary points, four minima and one maximum. Conventional locally convergent methods are typically used with multiple initial guesses, generally at or near



Global Optimization: Application to Phase Equilibrium Problems, Figure 1
Reduced Gibbs energy of mixing m versus x_1 for the system hydrogen sulfide and methane, showing tangent at a feed composition of 0.0187



Global Optimization: Application to Phase Equilibrium Problems, Figure 2
Tangent plane distance D versus x_1 for the example system of Fig. 1. See Fig. 3 for enlargement of area near the origin



Global Optimization: Application to Phase Equilibrium Problems, Figure 3
Enlargement of part of Fig. 2, showing area near the origin

the pure components ($x_1 = 0$ and $x_1 = 1$). When this is done convergence will likely be to the local minimum at the feed composition (0.0187) and to the local minimum around 0.88. The global minimum with $D < 0$ is missed, leading to the incorrect conclusion that the mixture is stable.

Interval Analysis

Interval analysis makes possible the mathematically and computationally guaranteed solution of the phase stability problem. Since the mole fraction variables x_i are known to lie between zero and one, and it is easy to put physical upper and lower bounds on the molar volume v as well, a feasible interval for all variables is readily identified. By applying an interval Newton/generalized bisection approach to the entire feasible interval, enclosures of *all* the stationary points of the tangent plane distance D can be found by solving the nonlinear equation system (1)–(3), and the *global* minimum of D thus identified. This approach requires no initial guess, and is applicable to any model for the Gibbs energy, not just those derived from equations of state. For the binary system used as an example above, all five stationary points are easily found, and the global minimum at $x_1 = 0.0767$, $v = 64.06$ cm^3/mol, and $D = -\,0.004$ thus identified [3,6].

The efficiency of the interval approach can depend significantly on how tightly one can compute *interval extensions* for the functions involved. The interval extension of a function over a given interval is an enclosure for the range of the function over that interval. When the *natural* interval extension, that is the function range computed using interval arithmetic, is used, it may tightly bound the actual function range. However, it is not uncommon for the natural interval extension to provide a significant overestimation of the true function range, especially for functions of the complexity encountered in the phase stability and equilibrium problems.

Some tightening of bounds can be achieved by taking advantage of information about function monotonicity. Another simple and effective way to alleviate this difficulty in this context is to focus on tightening the enclosure when computing interval extensions of mole fraction weighted averages, such as $\overline{r} = \sum_{i=1}^{n} x_i r_i$, where the r_i are constants. Due to the mixing rules for determining a and b, such expressions occur frequently, both in the equation of state (2) itself, as well in the derived model $m(\mathbf{x}, v)$ for the Gibbs energy of mixing and thus in equation (3). The natural interval extension of $\overline{r}$ will yield the true range (within roundout) of the expression in the space in which all the mole fraction variables x_i are independent. However, the range can be tightened by considering the constraint that the mole fractions must sum to one. One approach for doing this is simply to eliminate one of the mole fraction variables, say x_n. Then an enclosure for the range of $\overline{r}$ in the constrained space can be determined by computing the natural interval extension of $r_n + \sum_{i=1}^{n-1}(r_i - r_n)x_i$. However, this may not yield the sharpest possible bounds on $\overline{r}$ in the constrained space.

For constructing the *exact* (within roundout) bounds on $\overline{r}$ in the constrained space, S.R. Tessier [17] and Hua, Brennecke and Stadtherr [5] have presented a very simple method, based on the observation that at the extrema of $\overline{r}$ in the constrained space, at least $n - 1$ of the mole fraction variables must be at their upper or lower bound. This observation can be derived by viewing the problem of bounding the range of $\overline{r}$ in the constrained space as a linear programming problem. As shown in [5], when the constrained space interval extensions for mole fraction weighted averages are used, together with information about function monotonicity, significant improvements in computational efficiency, nearly an order of magnitude even for small (binary and ternary) problems, can be achieved in using the interval approach for solving the phase stability problem.

For small problems, it is usually efficient to globally minimize D by finding all of its stationary points, since this does not require repeated evaluation of the range of D. However, in general, for making a determination of phase stability or instability, finding *all* the stationary points is not really necessary, nor for larger problems, desirable. For example, if an interval is encountered over which the interval extension of D has a negative upper bound, this guarantees that there is a point at which $D < 0$, and so one can immediately conclude that the mixture is unstable without determining all the stationary points. It is also possible to easily make use of the underlying global minimization problem. Since the objective function D has a known value of zero at the mixture feed composition (tangent point), any interval over which the interval extension of D has a lower bound greater than zero cannot contain the global minimum and can be discarded, even though it may contain a stationary point (at which D will be positive and thus not of interest). Thus, one can essentially combine the interval-Newton technique with an interval branch and bound procedure in which lower bounds are generated using interval techniques.

Also, it should be noted that the global interval approach described here can easily be combined with existing local methods for determining phase stability and equilibrium. First, some (fast) local method is used. If it indicates instability then this is the correct answer as it means a point at which $D < 0$ has been found. If the local method indicates stability, however, this may not be the correct answer since the local method may have missed the global minimum in D. Applying interval analysis as described here can then be used to confirm that the mixture is stable if that is the case, or to correctly determine that it is really unstable if that is the case.

Conclusion

As demonstrated in [3,4,5,6,11,15], interval analysis can be used to solve phase stability and equilibrium problems efficiently and with complete reliability, providing a method that can guarantee with mathematical and computational certainty that the correct result is found, and thus eliminating computational problems that are encountered with conventional techniques. The method is initialization independent; it is also model independent, straightforward to use, and can be applied in connection with any equation of state or activity coefficient model for the Gibbs free energy of a mixture. There are many other problems in the analysis of phase behavior, and in chemical process analysis in general [14], that likewise are amenable to solution using this powerful approach.

See also

- ▶ Automatic Differentiation: Point and Interval
- ▶ Automatic Differentiation: Point and Interval Taylor Operators
- ▶ Bounding Derivative Ranges
- ▶ Global Optimization in Phase and Chemical Reaction Equilibrium
- ▶ Interval Analysis: Application to Chemical Engineering Design Problems
- ▶ Interval Analysis: Differential Equations
- ▶ Interval Analysis: Eigenvalue Bounds of Interval Matrices
- ▶ Interval Analysis: Intermediate Terms
- ▶ Interval Analysis: Nondifferentiable Problems
- ▶ Interval Analysis: Parallel Methods for Global Optimization
- ▶ Interval Analysis: Subdivision Directions in Interval Branch and Bound Methods
- ▶ Interval Analysis: Systems of Nonlinear Equations
- ▶ Interval Analysis: Unconstrained and Constrained Optimization
- ▶ Interval Analysis: Verifying Feasibility
- ▶ Interval Constraints
- ▶ Interval Fixed Point Theory
- ▶ Interval Global Optimization
- ▶ Interval Linear Systems
- ▶ Interval Newton Methods
- ▶ Optimality Criteria for Multiphase Chemical Equilibrium

References

1. Baker LE, Pierce AC, Luks KD (1982) Gibbs energy analysis of phase equilibria. Soc Petrol Eng J 22:731–742
2. Green KA, Zhou S, Luks KD (1993) The fractal response of robust solution techniques to the stationary point problem. Fluid Phase Equilib 84:49–78
3. Hua JZ, Brennecke JF, Stadtherr MA (1996) Reliable phase stability analysis for cubic equation of state models. Comput Chem Eng 20:S395–S400
4. Hua JZ, Brennecke JF, Stadtherr MA (1996) Reliable prediction of phase stability using an interval-Newton method. Fluid Phase Equilib 116:52–59
5. Hua JZ, Brennecke JF, Stadtherr MA (1998) Enhanced interval analysis for phase stability: Cubic equation of state models. Industr Eng Chem Res 37:1519–1527

6. Hua JZ, Brennecke JF, Stadtherr MA (1998) Reliable computation of phase stability using interval analysis: Cubic equation of state models. Comput Chem Eng 22:1207–1214
7. McDonald CM, Floudas CA (1995) Global optimization and analysis for the Gibbs free energy function using the UNIFAC, Wilson, and ASOG equations. Industr Eng Chem Res 34:1674–1687
8. McDonald CM, Floudas CA (1995) Global optimization for the phase and chemical equilibrium problem: Application to the NRTL equation. Comput Chem Eng 19:1111–1139
9. McDonald CM, Floudas CA (1995) Global optimization for the phase stability problem. AIChE J 41:1798–1814
10. McDonald CM, Floudas CA (1997) GLOPEQ: A new computational tool for the phase and chemical equilibrium problem. Comput Chem Eng 21:1–23
11. McKinnon KIM, Millar CG, Mongeau M (1996) Global optimization for the chemical and phase equilibrium problem using interval analysis. In: Floudas CA, Pardalos PM (eds) State of the Art in Global Optimization: Computational Methods and Applications. Kluwer, Dordrecht, pp 365–382
12. Michelsen ML (1982) The isothermal flash problem. Part I: Stability. Fluid Phase Equilib 9:1–19
13. Michelsen ML (1982) The isothermal flash problem. Part II: Phase-split calculation. Fluid Phase Equilib 9:21–40
14. Schnepper CA, Stadtherr MA (1996) Robust process simulation using interval methods. Comput Chem Eng 20:187–199
15. Stadtherr MA, Schnepper CA, Brennecke JF (1995) Robust phase stability analysis using interval methods. AIChE Symp Ser 91(304):356–359
16. Sun AC, Seider WD (1995) Homotopy-continuation method for stability analysis in the global minimization of the Gibbs free energy. Fluid Phase Equilib 103:213–249
17. Tessier SR (1997) Enhanced interval analysis for phase stability: Excess Gibbs energy models. MSc Thesis Dept Chemical Engin Univ Notre Dame
18. Wasylkiewicz SK, Sridhar LN, Malone MF, Doherty MF (1996) Global stability analysis and calculation of liquid-liquid equilibrium in multicomponent mixtures. Industr Eng Chem Res 35:1395–1408

Global Optimization Based on Statistical Models

ANTANAS ŽILINSKAS
Institute Math. and Informatics,
Vytautas Magnus University, Vilnius, Lithuania

MSC2000: 90C30

Article Outline

Keywords

Global optimization; Statistical models; Multimodal functions; Rational choice

Many practically significant problems require to optimize in a 'black box' situation, when the objective function is given by a code, but its structure is not known. In some algorithms, developed for such a case, different heuristic ideas are implemented. A disadvantage of the heuristic algorithms is dependence of the results on many parameters which choice is difficult because of rather vague meaning of these parameters. To develop a theory of global optimization the 'black box' should be replaced by a 'grey box' corresponding to some model of predictability/uncertainty of values of an objective function.

A model of an objective function is an important counterpart of any optimization theory (e. g., quadratic models are widely used to construct algorithms for local nonlinear optimization). The uncertainty on values of multimodal functions at the arbitrary points of the feasible region is more essential than uncertainty on the value of the objective function which will be calculated at the current iteration of the local descent. Therefore, the global optimization models that describe the objective function with respect to information obtained during the previous iterations are different from polynomial models used in local optimization. Different models may be used; e. g., a deterministic model, defining the guaranteed intervals for unknown function values, or a statistical model, modeling the uncertainty on function value by means of a random variable. The choice of a model is crucial because it defines the methodology of constructing the corresponding algorithms. A Lipschitzian type model enables the construction of global optimization algorithms with guaranteed (worst case) accuracy. However, the number of function evaluations in the worst case grows drastically with the dimensionality of the problem and the prescribed accuracy. In spite of this pessimistic theoretical result many practical rather complicated problems have been solved heuristically. Because a *heuristics* is a human experience based methodology, oriented towards average (typical, normal) conditions, it seems reasonable to develop a theory formalizing the principle of rational behavior with respect to average conditions in global optimization. The average rationality is well justified for playing a 'game against nature' which models optimization conditions better than an antagonistic game where the principle of minimax (guaranteed result) is well justified. The method ology of average rationality was applied to develop the general theory of rational choice under statistically interpreted uncertainty [4]. This general theory was further specified to develop the theory of global optimization based on statistical models of multimodal functions [11].

To construct a statistical model of multimodal function $f(x)$, $x \in A \subset \mathbf{R}^n$, the *axiomatic approach* is applied: the rationality of comparisons of likelihood of different values of $f(\cdot)$ is postulated by simple, intuitively acceptable axioms, and it is proved that the interpretation of an unknown value $f(x)$ as a Gaussian random variable ξ_x is compatible with the axioms. The parameters of ξ_x (mean value $m(x|(x_i, y_i))$ and variance $\sigma^2(x|(x_i, y_i))$, where $y_i = f(x_i)$ are known function values obtained during the search) are introduced by axiomatic theory of extrapolation under uncertainty. In the one-dimensional case both functions are very simple: $m(x|(x_i, y_i))$ is piecewise linear (connecting the neighboring trial points) and σ^2 $(x|(x_i, y_i))$ is piecewise quadratic.

By means of further (more restrictive) assumptions, the statistical models, corresponding to the stochastic functions, may be specified. The one-dimensional model corresponding to the Wiener process was introduced in [3]. However, the specification of a model as a stochastic function is not very reasonable: this normally involves additional very serious implementation difficulties and does not help to choose the model according to the a priori information on the problem. Using a statistical model the algorithm is constructed maximizing the probability to find better points than those found during the previous search. Such a strategy is justified also by the natural axioms of rationality of search. In the one-dimensional case the algorithm is easy to implement. In the multidimensional case, an auxiliary optimization problem must be solved [8].

Although the algorithm is based on the statistical model it is described without of use of randomization. Therefore it may be investigated by usual deterministic methods, e. g. the convergence of the algorithm in the is proved under weak assumptions on the underlying statistical model (continuity of $m(x|\cdot)$, $\sigma^2(x|\cdot)$ and weak dependence of both characteristics at point x on (x_i, y_i) for relatively remote points x_i [8]).

The models and algorithms of this approach are well grounded theoretically because they are derived from natural assumptions on rational behavior of an optimizer. As a topic for further research, the theory of average complexity seems very prospective. It would be important to evaluate the complexity of practically efficient algorithms constructed by the approach as well as to obtain general bounds and compare them with those obtained for Lipschitzian algorithms. The first results in this direction are interesting even for the one-dimensional case: the limit distribution of error of passive random search in case of the Wiener model exists or does not exist depending on a subtle interpretation of the model [2]. Other important theoretical topics are: developing dual (global-local) models for the multidimensional case, and justification of multidimensional statistical models oriented towards algorithms of the branch and bound type (cf. also ► Integer programming: Branch and bound methods), whose auxiliary computations would be essentially less time consuming than maximization of the probability over the whole feasible region at each iteration.

Many algorithms were constructed using different statistical models and more or less theoretically justified ideas. For example, a Bayesian algorithm (cf. also ► Bayesian global optimization) is defined by minimizing the average error with respect to the stochastic function chosen for a model [5]. By interpolation, the next calculation of a value of the objective function is performed at minimum point of $m(\cdot|(x_i, y_i))$ [1,6]. For the information-statistical method, an *ad hoc* one-dimensional model is constructed [1,7]. The algorithms may be generalized for the case with 'noisy' functions, see for example the algorithm in [8,10].

The known results from the theory of stochastic functions as well as axiomatic construction of statistical models do not give numerically tractable models which are completely adequate to describe local and global properties of a typical global optimization problem [1]. But in the framework of statistical models the adequacy, e. g., to local prop erties of the objective function, might be tested as a statistical hypothesis. If the statistical model is locally inadequate in a subset of the feasible region, then the objective function is assumed unimodal in this subset and a local minimum of $f(x)$ may be found by a local technique. An example of the combination of global and local search with a stopping rule corresponding to a high probability of finding the global minimum is presented in [9].

In the case of one-dimensional global optimization there are many competing algorithms including algorithms based on statistical models [8]. The algorithms representing different approaches may be compared with sufficient reliability by means of experimental testing. Since the codes in one-dimensional case are very precise realizations of theoretical algorithms then influence of implementation specifics is insignificant (at least with respect to multidimensional cases) and the comparison results may be generalized from codes to corresponding approaches. The results in [8] show that the algorithm from [9] and its modification [8] outperforms algorithms based on Lipschitzian type models even if a good estimate of the Lipschitz constant is available. The comparison of multidimensional algorithms is methodologically more difficult, partly because of very different stopping conditions. But generally speaking, the algorithms based on statistical models are efficient with respect to the number of evaluations of the objective function for the multimodal functions up to 10–15 variables [8]. The auxiliary computations require much computing time and computer memory. Therefore, such algorithms are rational to use for the problems, whose objective unction is expensive to evaluate. If an objective function is cheap to evaluate, the gain obtained from a low number of function evaluations may be less than the loss caused by the auxiliary computations.

A detailed review of the subject is presented in [8]; further references may be found in [1].

See also

- ▶ Adaptive Global Search
- ▶ Adaptive Simulated Annealing and its Application to Protein Folding
- ▶ αBB Algorithm
- ▶ Bayesian Global Optimization
- ▶ Continuous Global Optimization: Applications
- ▶ Continuous Global Optimization: Models, Algorithms and Software
- ▶ Differential Equations and Global Optimization
- ▶ DIRECT Global Optimization Algorithm
- ▶ Genetic Algorithms for Protein Structure Prediction
- ▶ Global Optimization in Binary Star Astronomy
- ▶ Global Optimization Methods for Systems of Nonlinear Equations
- ▶ Global Optimization Using Space Filling
- ▶ Monte-Carlo Simulated Annealing in Protein Folding
- ▶ Packet Annealing
- ▶ Random Search Methods
- ▶ Simulated Annealing
- ▶ Simulated Annealing Methods in Protein Folding
- ▶ Stochastic Global Optimization: Stopping Rules
- ▶ Stochastic Global Optimization: Two-phase Methods
- ▶ Topology of Global Optimization

References

1. Boender G, Romeijn E (1995) Stochastic methods. In: Horst R, Pardalos PM (eds) Handbook Global Optim. Kluwer, Dordrecht, pp 829–869
2. Calvin J, Glynn P (1997) Average case behavior of random search for the maximum. J Appl Probab 34:631–642
3. Kushner H (1962) A versatile stochastic model of a function of unknown and time-varying form. J Math Anal Appl 5:150–167
4. Luce D, Suppes P (1965) Preference, utility and subjective probability. In: Luce D, Bush R, Galanter E (eds) Handbook Math. Psychology. Wiley, New York, pp 249–410
5. Mockus J (1989) Bayesian approach to global optimization. Kluwer, Dordrecht
6. Shagen I (1980) Stochastic interpolation applied to the optimization of expensive objective functions. In: COMPSTAT 1980. Physica Verlag, Heidelberg, pp 302–307
7. Strongin R (1978) Numerical methods in multiextremal optimization. Nauka, Moscow
8. Törn A, Žilinskas A (1989) Global optimization. Springer, Berlin
9. Žilinskas A (1978) Optimization of one-dimensional multimodal functions, Algorithm AS 133. Applied Statist, 23:367–385
10. Žilinskas A (1980) MIMUN-optimization of one-dimensional multimodal functions in the presence of noise, Algoritmus 44. Aplikace Mat 25:392–402
11. Žilinskas A (1985) Axiomatic characterisation of a global optimization algorithm and investigation of it's search strategy. Oper Res Lett 4:35–39

Global Optimization in Batch Design Under Uncertainty

S. T. Harding, Christodoulos A. Floudas
Department Chemical Engineering,
Princeton University, Princeton, USA

MSC2000: 90C26

Article Outline

Keywords

Batch plant design; Multiproduct; Multipurpose; Uncertainty

Batch processes are a popular method for manufacturing products in low volume or that require several complicated steps in the synthesis procedure. The growth in the market for specialty chemicals has contributed to the demand for efficient batch plants. Batch processes are especially attractive due to their inherent *flexibility*. They can accommodate a wide range of production requirements. Batch equipment can be reconfigured to produce more than one product. Finally, certain pieces of equipment in batch processes can be used for more than one task.

An important area of concern in the design of batch processes is their ability to accommodate changes in production requirements and processing parameters. The key issue is: given some degree of uncertainty in a) the future demand for the products and b) the parameters that describe the chemical and physical steps involved in the process, what is the appropriate amount of flexibility the process should possess so as to maintain feasible operation while maximizing profits?

Many methods have been proposed for the design of batch plants under known market conditions and nominal operating conditions. Two major classes of batch plant designs are *multiproduct* plants and *multipurpose* plants. In the multiproduct plant, all products follow the same sequence of processing steps. Typically, one product is produced at a time in what is termed a *single-product campaign* (SPC). Multipurpose batch plants allow products to be processed using different sequences of equipment, and in some cases products can be produced simultaneously.

While significant progress has been made in the design and scheduling of batch plants, until recently the issues of flexibility and design under uncertainty have received little attention. Among the first to address the problem of batch plant design under uncertainty in a novel way were [10], and [8]. They divided the variables in the design problem into five categories: structural, design, state, operating, and uncertain. Structural variables describe the interconnections of the equipment in the plant. Design variables describe the size of the process equipment and are fixed once the plant is constructed. State variables are dependent variables and are determined once the design and operating variables are specified. Operating variables are those whose values can be changed in response to variations in the uncertain variables. Finally, the uncertain parameters are the quantities that can have random values which can be described by a *probability distribution*. Usually the uncertain parameters have normal distributions and are considered to be independent of each other. [8] also introduced the distinction between variations which have short-term effects and those with long-term effects. [18] extended this idea, suggesting a distinction between ‘hard’ and ‘soft’ constraints in which the former must be satisfied for feasible plant operation, but the latter may be violated, subject to a penalty in the objective function. They considered the time required to produce a product as uncertain and developed a problem formulation.

In [12], and [13] the authors addressed the problem of multiproduct batch plant design with uncertainties in both demand for the products and in technical parameters such as processing times and size factors. They restricted their designs to one piece of equipment per stage. [3] presented several variations on the problem of design with uncertain demands. They used interval methods to develop different solution procedures, including a two-stage approach and a penalty function approach. Another type of batch plant is the multipurpose plant. [14] proposed a *scenario*-based approach for the design of multipurpose batch plants with uncertain production requirements. The multipurpose approach resulted in a large scale MILP model for which efficient techniques for obtaining good upper and lower bounds were proposed. [15] developed a model for the multiproduct batch design problem which takes into account uncertainties in the product demands and in equipment availability. They considered the problem of design feasibility separately from the maximization of profits and presented an approach for achieving both criteria. [16] addressed the problem of uncertain demands, and used a scenario-based approach with discrete probability distributions for the demands. In addition, they considered the *scheduling* problem as a second stage, following the design problem. [6], and [7] considered the multiproduct batch plant design problem based on a stochastic programming formulation. They developed a relaxation of the production feasibility requirement and added a penalty term to the objective function to account for partial feasibility. Through this analysis, the problem can be reformulated as a single large scale *nonconvex optimization problem*. [2] extended this work to the design of multipurpose batch plants and implemented an efficient Gaussian quadrature technique to improve the estimation of the expected profit. [5] identified special structures in the nonconvex constraints for multiproduct and multipurpose batch design formulations. These properties can be exploited to obtain tight bounds on the global solution. This allows very large scale design problems to be solved in reasonable CPU time using the αBB method of [1].

Conceptual Framework

Most batch design problems are variations on the same basic model of a batch plant. The plant consists of M processing stages where each stage j contains N_j identical pieces of equipment. The volume of each unit, V_j, is a design variable, and the number of units per stage, N_j, may be a variable or a fixed parameter.

In the batch plant, NP products are to be made, and the amount of each produced is Q_i. Each product is produced in a number of batches of identical size, B_i. Using these definitions, a number of constraints on the design of the plant can be imposed. These constraints are:

1) an upper limit on the batch size,
2) a lower limit on the amount of time between batches,
3) an upper limit on the total processing time allowed, and
4) a constraint on the production related to the demand for each product. The basic form of these constraints is shown below, for a multiproduct batch plant with single-product campaigns.

Constraints on Batch Size

The batch size for each product i cannot be larger than the size of the pieces of equipment in each stage j. This can be written

$$B_i \leq \frac{V_j}{S_{ij}}, \qquad i = 1, \dots, NP, \quad j = 1, \dots, M.$$

The size factor, S_{ij}, is the capacity required in stage j to process one unit of product i.

Minimum Cycle Time

In order to make sure that each batch is processed separately in a given stage, one batch cannot begin processing until the previous batch has been processed for a certain amount of time. This is called the *cycle time*

$$T_{Li} \geq \frac{t_{ij}}{N_j}, \qquad i = 1, \dots, NP, \quad j = 1, \dots, M.$$

The time factor, t_{ij} is the amount of time to process one batch of product i in stage j.

Constraints on Production Time

The amount of time needed to produce all of the batches must be less than the total time available, H,

$$\sum_{i=1}^{NP} \frac{Q_i}{B_i} T_{Li} \leq H .$$

Demand Constraints

The production for each product must meet the demand.

$$Q_i = D_i .$$

Economic Objective Function

The objective is to maximize profits. The *profit* is calculated by subtracting the annualized capital costs from the revenues:

$$\text{Profit} = \sum_{i=1}^{NP} Q_i \cdot p_i - \sum_{j=1}^{M} \alpha_j N_j V_j^{\beta_j} ,$$

where p_i is the price of product i. The annualization factor for the cost of the units in stage j is α_j.

In the case where the number of units per stage, N_j is variable and/or the unit sizes, V_j, take only discrete values, this problem is a *mixed integer nonlinear optimization* problem (MINLP). If N_j is fixed and the unit sizes are continuous, the problem is a *nonlinear program* (NLP). In either case, the problem is nonconvex, therefore conventional mixed integer and nonlinear solvers cannot be used robustly. Instead, global optimization techniques must be employed to guarantee that the optimal solution is located.

Sources of Uncertainty

Within the mathematical framework for a multiproduct batch plant there are a number of possible sources of uncertainty. The most commonly studied are uncertainty in the process parameters, like the size factors, S_{ij}, and the time factors, t_{ij}, and uncertainty in the product demand, D_i. In addition to these, [3] considered uncertainty in the time horizon, H, and in the product prices, p_i.

Uncertainty in the process parameters is model inherent uncertainty, as classified by [11]. That is, uncertainty in the process parameters affects the feasible operation of the batch plant. Conversely, uncertainty in the product demand is an external source of uncertainty, therefore it only affects the objective function, and not the feasibility of the plant design.

Uncertainty in Process Parameters

The size factors and processing times affect the feasible design and operation of the batch plant. The goal is to design a plant that can operate feasibly, even if there is some uncertainty in the values of these parameters. The approach that is commonly followed is to consider a number of different scenarios, where each scenario corresponds to a set of parameter realizations. For example, if the size factors, S_{ij}, have some nominal value, $\overline{S}_{ij}$, then one scenario is that all of the size factors are at their nominal value. Similarly, if we have some knowledge about the amount of uncertainty in the size factors, we can construct a lower extreme scenario, where each size factor is at its lower bound, S_{ij}^L, and an upper extreme scenario, S_{ij}^U. The new set of size factors, reflecting the different scenarios is represented by the parameter S_{ij}^p. The scenarios can be weighted using the factor, w^p.

The set of constraints for the batch design problem must be modified so that the design is feasible over the whole set of scenarios, P:

$$B_i \leq \frac{V_j}{S_{ij}^p} , \quad T_{Li}^p \geq \frac{t_{ij}^p}{N_j} ,$$

$$\sum_{i=1}^{NP} \frac{Q_i^p}{B_i} T_{Li}^p \leq H .$$

Uncertainty in Product Demand

Uncertainty in the demand for the products affects the profitability of the plant. In this case, the product demand is given by a probability distribution function $J(\theta_i)$ where θ_i represents the uncertain demand for product i. The calculation of the expected revenues requires the integration over an optimization problem:

$$\mathsf{E}_\theta \left[\max_{Q_i} \sum_{i=1}^{NP} p_i Q_i \right] = \int_{\theta \in R(V_j, N_j)} \max_{Q_i} \left\{ \sum_{i=1}^{NP} p_i Q_i \right\} J(\theta) \, d\theta \, . \quad (1)$$

The integration should be performed over the feasible region of the plant, which is unknown at the design stage. See [6] for a *Gaussian quadrature* approach to discretize the integration. The range of uncertain demands is covered by a grid, where each point on the grid represents a set of demand realizations, and is assigned a weight corresponding to its probability, $\omega^q J^q$. The set of quadrature points is represented by Q. The expected revenues are now calculated as a multiple summation:

$$\begin{aligned}&\mathsf{E}_\theta\left[\max_{Q_i}\sum_{i=1}^{NP} p_i Q_i\right]\\ &=\sum_{p=1}^{P}\frac{1}{w^p}\sum_{q=1}^{Q}\omega^q J^q\sum_{i=1}^{NP} p_i Q_i^{qp}.\end{aligned}$$

In addition, the time horizon constraint must be modified:

$$\sum_{i=1}^{NP}\frac{Q_i^{qp}}{B_i}T_{Li}^p \le H, \quad \forall p \in P,\ \forall q \in Q.$$

Global Optimization Approaches

The set of constraints for the design of a multiproduct batch plant under uncertainty form a nonconvex optimization problem. Global optimization techniques must be used in order to ensure that the true optimal design is located.

Following the analysis of [9], an exponential transformation can be applied, reducing the number of nonlinear terms in the model.

$$\begin{aligned}V_j &= \exp(v_j), \quad \forall j \in M,\\ B_i &= \exp(b_i), \quad \forall i \in NP,\\ T_{Li}^p &= \exp(t_{Li}^p), \quad \forall i \in NP.\end{aligned}$$

In [5] and [6] global optimization methods were developed to solve this problem, where the number of units in each stage, N_j, is fixed. In this case, the cycle time becomes a parameter, determined by,

$$\begin{aligned}&t_{Li}^p = \max_j\left\{\ln\left(\frac{t_{ij}^p}{N_j}\right)\right\},\\ &\forall i \in NP,\ \forall p \in P.\end{aligned}$$

The nonlinear optimization problem to be solved is written as a minimization:

$$\left\{\begin{array}{ll}\displaystyle\min_{b_i,v_j,Q_i^{qp}} & \displaystyle\delta\sum_{j=1}^{M}\alpha_j N_j\exp\left(\beta_j v_j\right)\\ & \displaystyle-\sum_{p=1}^{P}\frac{1}{w^p}\sum_{q=1}^{Q}\omega^q J^q\sum_{i=1}^{NP}p_i Q_i^{qp}\\ & \displaystyle+\gamma\sum_{p=1}^{P}\frac{1}{w^p}\sum_{q=1}^{Q}\omega^q J^q\sum_{i=1}^{NP}p_i\left(\theta_i^q - Q_i^{qp}\right)\\ \text{s.t.} & v_j \ge \ln(S_{ij}^p) + b_i\\ & \displaystyle\sum_{i=1}^{NP}Q_i^{qp}\cdot\exp(t_{Li}^p - b_i) \le H\\ & \theta_i^L \le Q_i^{qp} \le \theta_i^q\\ & \ln(V_j^L) \le v_j \le \ln(V_j^U)\\ & \displaystyle\min_{j,p}\ln\left(\frac{V_j^L}{S_{ij}^p}\right) \le b_i \le \min_{j,p}\ln\left(\frac{V_j^U}{S_{ij}^p}\right).\end{array}\right. \tag{2}$$

Note that the time horizon constraint is the only nonconvex constraint remaining in the problem formulation. A *penalty* term is added to the objective function to account for unsatisfied demand, the penalty parameter is γ.

The GOP Approach

In [7] and [2] the GOP algorithm of [4,17] has been applied to solve design formulations for both multipurpose and multiproduct batch plants. GOP converges to the global optimum solution by solving a primal problem and a number of relaxed dual problems in each iteration. In [7] it is observed that if the variables in the batch design problem are partitioned so that $y = \{v_j, b_i\}$ and $x = \{Q_i^{qp}\}$, then the problem is convex in y for every fixed x, and linear in x for every fixed y. This satisfies Condition A) of the GOP algorithm.

A property was developed in [7] that allows the number of relaxed duals per iteration to be reduced from $2^{NP\cdot Q}$ to 2^{NP}, making the problem computationally tractable.

αBB Approach

The αBB approach of [1] was applied in [5] to solve both multiproduct and multipurpose design formulations. αBB is a *branch and bound* approach that converges to the global solution by solving a sequence of upper and lower bounding problems. The lower bounding problem is formulated by subtracting a quadratic term, multiplied by the constant α, from each of the nonconvex terms, thus convexifying the problem. Often, the size of the α term must be estimated, resulting in poor lower bounds in the first few levels of the branch and bound tree. However, the nonconvex terms in the batch plant design formulation allow the *exact* value of α to be calculated, resulting in a tight lower bound on the global solution. This technique has been used to find the optimal design for a multiproduct batch plant with 5 products in 6 stages. This corresponds to a nonconvex NLP with 15,636 variables, 3155 constraints, and 15,625 nonconvex terms.

Other Types of Batch Plants

In addition to the multiproduct batch plant with single-product campaign illustrated in the preceding sections, there are many other batch plant design formulations that can be adapted to consider the issue of uncertainty in design.

Mixed-Product Campaign

This is another example of a multiproduct batch plant. In this case, storage of the intermediate products is allowed between processing steps. In addition, batches of different products can be alternated. This allows a reduction in the total production time. Rather than being limited by the largest cycle time for all stages, this method calculates the total production time for each stage:

$$T_j^{qp,\mathrm{tot}} \geq \sum_{i=1}^{NP} \left(\frac{Q_i^{qp}}{B_i} \right) t_{ij}^p .$$

The total time for each stage must be less than the total time allowed:

$$H \geq T_j^{qp,\mathrm{tot}} \geq \sum_{i=1}^{NP} \left(\frac{Q_i^{qp}}{B_i} \right) t_{ij}^p .$$

This can be written

$$\sum_{i=1}^{NP} \left(\frac{Q_i^{qp}}{B_i} \right) t_{ij}^p \leq H .$$

Note that this constraint has the same form as the time horizon constraint for the single-product campaign formulation.

Multipurpose Batch Plant-Single Equipment Sequence

In a multipurpose batch plant, the equipment can be used for more than one function, therefore each product may have a different route through the plant. In the single equipment sequence case, there is one distinct route for each product. Production is carried out in a sequence of campaigns L, and there may be more than one product produced simultaneously in a campaign, h. The time needed for each campaign, C_h, is based on the maximum cycle time for all products in the campaign,

$$\sum_{h=1}^{L} \alpha_{hi} C_h^{qp} \geq \left(\frac{Q_i^{qp}}{B_i} \right) T_{Li}^p ,$$

where

$$\alpha_{hi} = \begin{cases} 1 & \text{if product } i \text{ is allowed} \\ & \quad \text{in campaign } h, \\ 0 & \text{else.} \end{cases}$$

Finally, the sum of all campaign times must be less than the total time available:

$$\sum_{h=1}^{L} C_h^{qp} \leq H .$$

Multipurpose Batch Plant-Multiple Equipment Sequence

In this case, there are multiple routes through the plant for each product i, PR_i. The total amount of product i produced is the sum over the production of i in each route:

$$Q_i^{qp} = \sum_{r \in PR_i} q_r^{qp} .$$

The time for campaign C_h is based on the maximum cycle time for each route in the campaign,

$$\sum_{h=1}^{L} \alpha_{hr} C_h^{qp} \geq \left(\frac{q_r^{qp}}{B_r} \right) t_{Lr}^p .$$

The sum of all campaign times must be less than the total time available,

$$\sum_{h=1}^{L} C_h^{qp} \leq H.$$

Note that in both of the multipurpose batch design formulations shown above, the constraints that are added are either linear, or have the exact same form of nonconvexities as shown for the multiproduct batch design formulation. Therefore, the global optimization techniques discussed in Section 'Global Optimization Approaches' are applicable to these problems.

See also

- ► αBB Algorithm
- ► Continuous Global Optimization: Models, Algorithms and Software
- ► Global Optimization in Generalized Geometric Programming
- ► Global Optimization Methods for Systems of Nonlinear Equations
- ► Global Optimization in Phase and Chemical Reaction Equilibrium
- ► Interval Global Optimization
- ► MINLP: Branch and Bound Global Optimization Algorithm
- ► MINLP: Global Optimization with αBB
- ► Smooth Nonlinear Nonconvex Optimization

References

1. Androulakis IP, Maranas CD, Floudas CA (1995) αBB: A global optimization method for general constrained nonconvex problems. J Global Optim 7:337–363
2. Epperly TGW, Ierapetritou MG, Pistikopoulos EN (1997) On the global and efficient solution of stochastic batch plant design problems. Comput Chem Eng 21:1411–1431
3. Fichtner G, Reinhart H-J, Rippin DWT (1990) The design of flexible chemical plants by the application of interval mathematics. Comput Chem Eng 14:1311–1316
4. Floudas CA, Visweswaran V (1990) A global optimization algorithm (GOP) for certain classes of nonconvex NLPs: I. Theory. Comput Chem Eng 14:1397–1417
5. Harding ST, Floudas CA (1997) Global optimization in multiproduct and multipurpose batch design under uncertainty. Industr Eng Chem Res 36:1644–1664
6. Ierapetritou MG, Pistikopoulos EN (1995) Design of multiproduct batch plants with uncertain demands. Comput Chem Engin 19:S627–S632
7. Ierapetritou MG, Pistikopoulos EN (1996) Batch plant design and operations under uncertainty. Industr Eng Chem Res 35:772–787
8. Johns WR, Marketos G, Rippin DWT (1978) The optimal design of chemical plant to meet time-varying demands in the presence of technological and commercial uncertainty. Trans Inst Chem Eng 56:249–257
9. Kocis GR, Grossmann IE (1988) Global optimization of nonconvex mixed-integer nonlinear programming (MINLP) problems in process synthesis. Industr Eng Chem Res 27:1407
10. Marketos G (1975) The optimal design of chemical plant considering uncertainty and changing circumstances. PhD Thesis, ETH Zurich
11. Pistikopoulos EN (1995) Uncertainty in process design and operations. Comput Chem Eng 19:S553–S563
12. Reinhart HJ, Rippin DWT (1986) The design of flexible batch chemical plants. In: 1986 AIChE Annual Meeting
13. Reinhart HJ, Rippin DWT (1987) Design of flexible multiproduct plants: A new procedure for optimal equipment sizing under uncertainty. In: 1987 AIChE Annual Meeting
14. Shah N, Pantelides CC (1992) Design of multipurpose batch plants with uncertain production requirements. Industr Eng Chem Res 31:1325–1337
15. Straub DA, Grossmann IE (1992) Evaluation and optimization of stochastic flexibility in multiproduct batch plants. Comput Chem Eng 16:69–87
16. Subrahmanyam S, Pekny JF, Reklaitis GV (1994) Design of batch chemical plants under market uncertainty. Industr Eng Chem Res 33:2688–2701
17. Visweswaran V, Floudas CA (1993) New properties and computational improvement of the GOP algorithm for problems with quadratic objective function and constraints. J Global Optim 3(3):439–462
18. Wellons HS, Reklaitis GV (1989) The design of multiproduct batch plants under uncertainty with staged expansion. Comput Chem Eng 13:115–126

Global Optimization in Binary Star Astronomy
GO4BSA

DIMITRI POURBAIX
Royal Observatory of Belgium, Brussels, Belgium

MSC2000: 90C26, 90C90

Article Outline

Keywords

Astronomy; Binary; Star; Orbit; Mass

The global optimization techniques are still quite unpopular in the astronomical community, in particular, among the double stars astronomers. Among the reasons of their reticence one finds a long practice of manual and graphical methods, 'least squares' adjustments of a linearized objective function, differential correction, etc.

This article does not present, unfortunately, the state of the art in *orbits determination*, even if a few astronomers, mostly young ones, tries to convince the others that a global minimization step is useful. This article presents a possible way to obtain the orbital parameters of double-lined spectroscopic visual *binaries*.

Astronomical Problem

The generic terms 'binary star' designate two stars that are gravitationally linked together. Since J. Kepler, one knows that such an interaction leads to an elliptic orbital motion of one star around each the other (Kepler's first law). The Kepler third law tells us that there is a simple relation between the orbital period (P), the semimajor axis of the relative orbit (a) and the *mass* sum of the 2 stars (M_A (the mass of the brighter star) and M_B (the mass of the fainter component)):

$$\frac{a^3}{P^2} = M_A + M_B ,$$

where a is expressed in astronomical unit (1 A.U. is equal to the average distance of the Earth from the Sun), P is expressed in years and the masses in solar masses ($M_\odot$). This relation is still, almost 400 years after Kepler, the only direct and hypothesis-free method to estimate stellar masses.

A visual binary corresponds to a situation where the 2 stars are visually resolved and the orbital motion, projected on the plane orthogonal to the sight direction, can be perceived. From the relative positions of B with respect to A along time (t, x and y), one can extract the 7 parameters characterizing the visual orbit. Among these parameters, there are P and the angular value of a (expressed in seconds of arc). The latter cannot be converted into its linear value in A.U. unless the distance to the binary system is known (or, equivalently, the parallax of the system, ϖ, is known).

A binary star is spectroscopic if the motion of its spectral lines is observable. This motion is due to the *Doppler effect*: all lines issued from one star are shifted toward the blue (red) side of the spectrum when that star is moving toward (away from) the observer. The wavelength shift between the laboratory wavelength, λ_L, and the observed one, λ_O, is connected to the radial velocity V through:

$$\frac{\lambda_O - \lambda_L}{\lambda_L} = \frac{V}{c}$$

where c stands for the speed of the light in the vacuum. In a double-lined spectroscopic binary, lines from the two components are seen in the spectrum.

The radial velocity curve $((t, V_A), (t, V_B))$ of each component along time shows a periodic variation. Lets K_A designates the amplitude of the radial velocity curve of component A and K_B the amplitude of component B. There is a relation between the mass ratio and the K. values:

$$\frac{K_A}{K_B} = \frac{M_B}{M_A}$$

The amplitudes are usually expressed in km/s.

Hence, if a binary star is simultaneously visual and double-lined spectroscopic, one can extract the individual masses and the distance to the system with no extra hypothesis.

Objective Function

To describe the observations of a double-lined *spectroscopic visual binary* requires at least 10 parameters. By observations, one means the relative positions of the fainter component with respect to the brighter star and the radial velocities of both components. Why more than 10 parameters could be necessary is beyond the scope of this paper. Among the different possible sets of 10 parameters, we select:

- $a^{('')}$: the angular semimajor axis of the relative orbit of the fainter component around the brighter star;
- i: the inclination of the orbital plane with respect to the plane orthogonal to the direction sight;

- ω: the argument of the periastron;
- Ω: the longitude of the ascending node;
- e: the eccentricity;
- P: the period;
- T: the periastron epoch (one of them);
- V_0: the radial velocity of the system's center of mass;
- ϖ: the *parallax* of the system;
- κ: the ratio of the semimajor axis (relative to the brighter component) to the sum of the two semimajor axes.

The most natural way to combine visual and spectroscopic observations is to use a least squares approach and to seek the minimum of an expression like:

$$\begin{aligned} &D(a, i, \dots, \varpi, \kappa) \\ &= \sum_{j=1}^{N_v}\left[\left(\frac{\overset{o}{x}_j - \widehat{x}_j}{\sigma_{x_j}}\right)^2 + \left(\frac{\overset{o}{y}_j - \widehat{y}_j}{\sigma_{y_j}}\right)^2\right] \\ &+ \sum_{k=1}^{N_{sA}}\left(\frac{\overset{o}{V}_{A_k} - \widehat{V}_{A_k}}{\sigma_{V_{A_k}}}\right)^2 + \sum_{l=1}^{N_{sB}}\left(\frac{\overset{o}{V}_{B_l} - \widehat{V}_{B_l}}{\sigma_{V_{B_l}}}\right)^2 \end{aligned} \tag{1}$$

where the hat (super) stands for the adjusted (observed) quantity and $\sigma.$ are the a priori known (or estimated) standard deviations of the observations.

In fact, yet this idea of combining the two aspects of the orbit is unusual. Most of the time, astronomers keep the separation when computing the orbital parameters. Visual observers compute their own orbit and spectroscopists theirs: one group simply fixes some parameters (w, e, P and T) to the values obtained by the other group (e. g., [5]). A few papers only presents a *simultaneous adjustment* of the ten parameters (e. g., [12,18]).

The reader could be puzzled by the fact that the expression of D seems to be too kind to have numerous local minima and to require a global optimization method to be minimized. A description of how x, y, V_A and V_B are computed is going to justify our approach.

The visual orbit requires

$$\begin{aligned} x &= AX + FY, \\ y &= BX + GY, \\ X &= \cos E - e, \\ Y &= \sqrt{1-e^2}\sin E, \end{aligned}$$

where X and Y (x and y) are the angular rectangular coordinates, in the orbital (tangential) plane, of the fainter component with respect to the brighter one; A, B, F and G are the Thiele–Innes constants, expressed in terms of $a^{('')}$, i, ω and Ω as

$$\begin{aligned} A &= a^{('')}(\cos\omega\cos\Omega - \sin\omega\sin\Omega\cos i), \\ B &= a^{('')}(\cos\omega\sin\Omega + \sin\omega\cos\Omega\cos i), \\ F &= a^{('')}(-\sin\omega\cos\Omega - \cos\omega\sin\Omega\cos i), \\ G &= a^{('')}(-\sin\omega\sin\Omega + \cos\omega\cos\Omega\cos i). \end{aligned}$$

E is the eccentric anomaly at time t, determined unambiguously by Kepler's equation

$$E - e\sin E = \frac{2\pi}{P}(t-T).$$

For a spectroscopic orbit j ($j = A$ or $j = B$), one needs

$$\begin{aligned} V_A &= V_0 - K_A(\cos(\omega+\nu) + e\cos\omega), \\ V_B &= V_0 + K_B(\cos(\omega+\nu) + e\cos\omega), \\ K_j &= \frac{2\pi a_j^{(\mathrm{km})}\sin i}{86400\cdot 365.242198781 P\sqrt{1-e^2}}, \\ \tan\frac{\nu}{2} &= \sqrt{\frac{1+e}{1-e}}\tan\frac{E}{2}. \end{aligned}$$

The angular separation in arcseconds is converted into its linear value using

$$\begin{aligned} a^{(\mathrm{km})} &= \frac{a^{('')}}{\varpi}\cdot 1.49598\cdot 10^8, \\ a_A^{(\mathrm{km})} &= \kappa a^{(\mathrm{km})}, \\ a_B^{(\mathrm{km})} &= (1-\kappa)a^{(\mathrm{km})}. \end{aligned}$$

Global Search

In front of a low-dimension but highly nonlinear problem, what can be used to find the minimum of an expression such as D (equation (1))? *Simulated annealing* ([8,11]) has already been successfully applied to the determination of the orbital parameters of *visual binaries* [14]. In that case, only 7 parameters are required, but the nature of the problem seems close enough to the current one to be tempted to use the same approach.

The implementation of SA used for the visual problem gives satisfaction ([1,15]). Nevertheless, the increase of the working space dimension is, by itself, enough to justify the search for an improved algorithm for the combined spectroscopic-visual problem.

Among the few SA implementations for continuous functions, the one in 'Numerical Recipes' [16] was selected. Although the published code behaves very well, some improvements (at least for our purpose) are possible. We are going to focus on modifications of the basic algorithm, mainly some improvements of the guess generator. A rough pseudocode of the algorithm in [16] is given below:

```
DO
        use a simplex to get a new solution;
        decrease the temperature;
WHILE   (temperature > $T_{\min}$);
```

Suggested pseudocode after [16]

Let's first remind that the guess generator proposed in [16] is based on a thermally disturbed *simplex* [13]. When the temperature approaches 0, the generator reduces to the *Nelder–Mead algorithm* and a local convergence can be expected. W.H. Press et al. announce a local convergence whereas V. Torczon [17] showed such a convergence cannot be guaranteed with the Nelder–Mead algorithm.

The major drawback of this algorithm is that the simplex can degenerate (a vertex becomes a linear combination of strictly less than the other n ones). If that happens, only a subspace of the complete working space can be visited and the risk of missing the minimum raises.

To decide whether or not to reinitialize the simplex can be based on the mean of the values at the n+1 vertices. The mean is compared with the mean at the previous temperature. If the relative change is not important enough or the generator stops at a local minimum, a new simplex is generated. The best point ever met is chosen as one of the vertices.

A natural way to initialize a simplex is to choose the n remaining vertices such that each edge issued from the (n+1)th point is parallel to a different axis of coordinates. A refined version of that approach is adopted. Instead of randomly choosing the value of the component in the interval of accepted values for that component, some *'taboo' restrictions* are added.

The overall working space is divided in regions. When a new simplex is generated, each cells containing a vertex are marked as taboo. The random selection of the value of a component is repeated until the resulting cell (C) does not lie in a taboo region (TL). Even if the best point ever met does not change between two successive re-initializations, this procedure guarantees that the two simplices are different. That raises the probability of visiting the overall space. Practically, the taboo cells are kept in a circular linked list and discarded when space for a new cell is required. The resulting pseudocode is given below:

```
DO
    use a simplex to get a new solution;
    IF      initialization required
    THEN    adopt the best solution as the ($n$+1)th
            vertex;
            for the first $n$ vertices ($V_i$)
            DO
                $V_i = V_{n+1}$;
                DO
                change the $i$th component of $V_i$;
                identify C;
                WHILE (C in TL);
                add C to TL;
            OD;
    FI;
    decrease the temperature;
WHILE (temperature > $T_{\min}$);
```

Adopted pseudocode

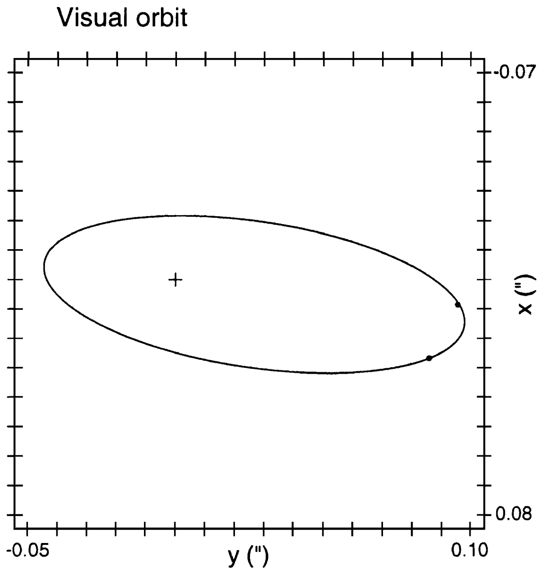
Ingber's algorithm ([6,7]) is used for the annealing schedule. The initial temperature is set to $10^{\langle log_{10}(D)\rangle}$ where $\langle \log_{10}(D)\rangle$ stands for the mean of the logarithm of the objective function over the first generated simplex.

Element	Value	Std. dev.
$a('')$	0.072	0.0010
$i(°)$	68	1.3
$\omega(°)$	352	2.2
$\Omega(°)$	262.0	0.53
e	0.38	0.016
P(yr)	1.7255	0.00098
T (Besselian yr)	1979.332	0.0099
V_0(km/s)	−9.78	0.13
$\varpi('')$	0.038	0.0012
κ	0.349	0.0096
mass $A(M_\odot)$	1.5	0.18
mass $B(M_\odot)$	0.8	0.12

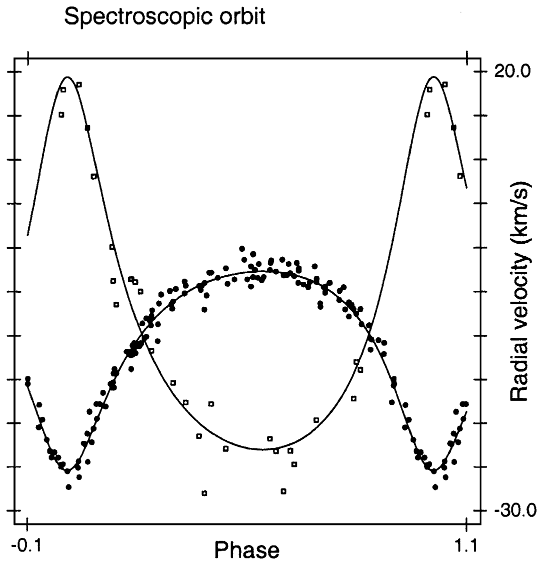
Orbital parameters of HIP111170 and their standard derivations

Example 1 (HIP111170)

The *double star* HIP111170 (= HR8851 = HD213429) is a good example to illustrate how appropriate a *simultaneous adjustment* is whereas a disjoint one would failed. The visual observations ([9,10]) are too few to allow a visual orbit determination: 3.5 observations (2 quantities) are necessary to adjust 7 parameters. Fortunately, the spectroscopic data are more numerous and the two radial velocity curves are well covered. From a mathematical point of view, two visual observations is the minimum if the spectroscopic observations [3] are well spread over the two curves.



Global Optimization in Binary Star Astronomy, Figure 1
Adjusted visual orbit of HIP 111170. The cross represents component *A*



Global Optimization in Binary Star Astronomy, Figure 2
Adjusted spectroscopic orbits of HIP 111170

The table above gives the orbital parameters used for the figures. The obtained *parallax* is in quite good agreement with the 0.03918±0.00183″ after the *Hipparcos* mission [4].

Conclusion

Even when the observations seem very precise, the objective function describing the residual between the observed and computed data has many local minima. Astronomers should be aware of that fact as they should be aware of techniques to efficiently tackle such situations.

See also

- ► *α*BB Algorithm
- ► Continuous Global Optimization: Applications
- ► Continuous Global Optimization: Models, Algorithms and Software
- ► Differential Equations and Global Optimization
- ► DIRECT Global Optimization Algorithm
- ► Global Optimization Based on Statistical Models
- ► Global Optimization Methods for Systems of Nonlinear Equations
- ► Global Optimization Using Space Filling
- ► Topology of Global Optimization

References

1. Carette E, de Greve JP, van Rensebergen W, Lampens P (1995) γ Circini: A young visual binary with pre-main-sequence component(s)? Astronomy and Astrophysics 296:139
2. Docobo JA, Elipe A, McAlister H (eds) (1997) Visual Double Stars: Formation, Dynamics and Evolutionary Tracks. Kluwer, Dordrecht
3. Duquennoy A, Mayor M, Griffin RF, Beavers WI, Eitter JJ (1988) Duplicity in the solar neighbourhood; V. Spectroscopic orbit of the nearby double-lined star HR 8581. Astronomy and Astrophysics Suppl Ser 75:167
4. ESA (1997) The Hipparcos and Tycho catalogues. ESA SP-1200
5. Hummel CA, Armstrong JT, Buscher DF, Mozurkewich D, Quirrenbach A, Vivekanand M (1995) Orbits of small angu-

lar scale binaries resolved with the Mark III interferometer. Astronomical J 110:376
6. Ingber L (1993) Adaptative simulated annealing (asa). Techn Report Caltech 1
7. Ingber L (1993) Simulated annealing: Practice versus theory. Res Note Caltech 1
8. Kirkpatrick S, Gelatt CD Jr, Vecchi MP (1983) Optimization by simulated annealing. Science 220(4598):671
9. McAlister HA, Hartkopf WI, Franz OG (1990) ICCD speckle observations of binary stars. V Measurements during 1988-1989 from the Kitt Peak and the Cerro Tololo 4 m telescope. Astronomical J 99:965
10. McAlister HA, Hartkopf WI, Hutter DJ, Shara MM, Franz OG (1987) ICCD speckle observations of binary stars. I: A survey of duplicity among the bright stars. Astronomical J 93:183
11. Metropolis N, Rosenbluth AW, Rosenbluth MN, Teller AH, Teller E (1953) Equation of state calculations by fast computing machines. J Chem Phys 21(6):1087
12. Morbey CL (1975) A synthesis of the solutions of spectroscopic and visual binary orbits. Publ Astronomical Soc Pacific 87:689
13. Nelder JA, Mead R (1965) A simplex method for function minimization. Comput J 7:308
14. Pourbaix D (1994) A trial-and-error approach to the determination of the orbital parameters of visual binaries. Astronomy and Astrophysics 290:682
15. Pourbaix D, Lampens P (1997) A new method used to revisit the visual orbit of the spectroscopic triple system eta Orionis A. In: Docobo JA, Elipe A and McAlister H (eds) Visual Double Stars: Formation, Dynamics and Evolutionary Tracks. Kluwer, Dordrecht, p 383
16. Press WH, Teukolsky SA, Vetterling WT, Flannery BP (1992) Numerical recipes in C, 2nd edn. Cambridge Univ. Press, Cambridge
17. Torczon V (1991) On the convergence of the multidirectional search algorithm. SIAM J Optim 1(1):123
18. Torres G (1995) A visual-spectroscopic orbit for the binary Σ248. Publ Astronomical Soc Pacific 107:524

Global Optimization: Cutting Angle Method

ADIL BAGIROV[1], GLEB BELIAKOV[2]
[1] Centre for Informatics and Applied Optimization, School of Information Technology and Mathematical Sciences, University of Ballarat, Victoria, Australia
[2] School of Engineering and Information Technology, Deakin University, Victoria, Australia

MSC2000: 90C26, 65K05, 90C56, 65K10

Article Outline

Introduction

The cutting angle method (CAM) is a deterministic method for solving different classes of global optimization problems. It is a version of the generalized cutting plane method, and it works by building a sequence of tight underestimates of the objective function. The sequence of global minima of the underestimates converges to the global minimum of the objective function. It can also be seen from the perspective of branch-and-bound type methods, which iterate the steps of branching (partitioning the domain), bounding the objective function on the elements of the partition, and also fathoming (eliminating those elements of the partition which cannot contain the global minimum).

The key element of CAM is the construction of tight underestimates of the objective function and their efficient minimization in a structured optimization problem. CAM is based on the theory of abstract convexity [23], which provides the necessary tools for building accurate underestimates of various classes of functions. Such underestimates arise from a generalization of the following classical result: each convex function is the upper envelop of its affine minorants [21]. In abstract convex analysis, the requirement of linearity of the minorants is dropped, and abstract convex functions are represented as the upper envelops of some simple minorants, or support functions, which are not necessarily affine. Depending on the choice of the support functions, one obtains different flavours of abstract convex analysis.

By using a subset of support functions, one obtains an approximation of an abstract convex function from below. Such one-sided approximation, or underestimate, is very useful in optimization, as the global minimum of the underestimate provides a lower bound on the global minimum of the objective function. One can find the global minimum of the objective function as the limiting point of the sequence of global minima of the underestimates. This is the principle of the cutting angle method of global optimization [1,2,23].

The cutting angle method was first introduced for global minimization of increasing positive homogeneous (IPH) functions over the unit simplex [1,2,23]. Then it was extended to a broader class of Lipschitz programming problems [9,25]. In this Chapter, after providing the necessary theoretical background, we will describe versions of CAM for global minimization of IPH and Lipschitz functions over a polytope (in particular the unit simplex), and provide details of its algorithmic implementation.

Definitions

Notation

- n is the dimension of the optimization problem;
- $I = \{1, \dots, n\}$;
- x_i is the ith coordinate of a vector $x \in \mathbb{R}^n$;
- $x^k \in \mathbb{R}^n$ denotes the k-th vector of some sequence $\{x^k\}_{k=1}^K$;
- $[l, x] = \sum_{i \in I} l_i x_i$ is the inner product of vectors l and x;
- if $x, y \in \mathbb{R}^n$ then $x \geq y \Leftrightarrow x_i \geq y_i$ for all $i \in I$;
- if $x, y \in \mathbb{R}^n$ then $x \gg y \Leftrightarrow x_i > y_i$ for all $i \in I$;
- $\mathbb{R}^n_+ := \{x = (x_1, \dots, x_n) \in \mathbb{R}^n : x_i \geq 0$ for all $i \in I\}$ (nonnegative orthant);
- $\mathbb{R}_{+\infty}$ denotes $(-\infty, +\infty]$;
- $e^m = (0, \dots, 0, 1, 0, \dots, 0)$ denotes the m-th unit orth of the space $\mathbb{R}^n$.
- $S = \{x \in \mathbb{R}^n_+ : \sum_{i \in I} x_i = 1\}$ (unit simplex).

Abstract Convex Functions

Let $X \subset \mathbb{R}^n$ be some set, and let H be a nonempty set of functions $h\colon X \to V \subset [-\infty, +\infty]$. We have the following definitions [23].

Definition 1 A function f is abstract convex with respect to the set of functions H (or H-convex) if there exists $U \subset H$ such that:

$$f(x) = \sup\{h(x)\colon h \in U\}, \quad \forall x \in X .$$

Definition 2 The set U of H-minorants of f is called the support set of f with respect to the set of functions H:

$$\mathrm{supp}(f, H) = \{h \in H, h(x) \leq f(x) \quad \forall x \in X\} .$$

Definition 3 H-subgradient of f at x is a function $h \in H$ such that:

$$f(y) \geq h(y) - (h(x) - f(x)), \quad \forall y \in X .$$

The set of all H-subgradients of f at x is called H-subdifferential

$$\partial_H f(x) = \{h \in H\colon f(y) \geq h(y) - (h(x) - f(x)), \\ \forall y \in X\} .$$

Definition 4 The set $\partial^*_H f(x)$ at x is defined as

$$\partial^*_H f(x) = \{h \in \mathrm{supp}(f, H)\colon h(x) = f(x)\} .$$

Proposition 1 [23], *p.10. If the set H is closed under vertical shifts, i. e., $(h \in H, c \in R)$ implies $h - c \in H$, then $\partial^*_H f(x) = \partial_H f(x)$.*

When the set of support functions H consists of all affine functions, then we obtain the classical convexity. Next we examine two other examples of sets of support functions H.

IPH Functions

Recall that a function f defined on $\mathbb{R}^n_+$ is increasing if $x \geq y$ implies $f(x) \geq f(y)$.

Definition 5 A function $f\colon \mathbb{R}^n_+ \to \mathbb{R}$ is called IPH (Increasing Positively Homogeneous functions of degree one) if

$$\forall x, y \in \mathbb{R}^n_+, \quad x \geq y \Rightarrow f(x) \geq f(y);$$
$$\forall x \in \mathbb{R}^n_+, \forall \lambda > 0\colon f(\lambda x) = \lambda f(x) .$$

Let the set H_1 be the set of min-type functions

$$H_1 = \{h\colon h(x) = \min_{i \in I} a_i x_i, a \in \mathbb{R}^n_+, x \in \mathbb{R}^n_+\} .$$

Proposition 2 [23] *A function $f\colon \mathbb{R}^n_+ \to \mathbb{R}_{+\infty}$ is abstract convex with respect to H_1 if and only if f is IPH.*

Example 1 The following functions are IPH:

1) $f(x) = \sum_{i \in I} a_i x_i$ with $a_i \geq 0$;
2) $p_k(x) = \left(\sum_{i \in I} x_i^k\right)^{\frac{1}{k}} (k > 0)$;
3) $f(x) = \sqrt{[Ax, x]}$ where A is a matrix with nonnegative entries;
4) $f(x) = \prod_{j \in J} x_j^{t_j}$ where $J \subset I,\ t_j > 0, \sum_{j \in J} t_j = 1$.

It is easy to check that

- the sum of two IPH functions is also an IPH function;
- if f is IPH, then the function γf is IPH for all $\gamma > 0$;
- let T be an arbitrary index set and $(f_t)_{t \in T}$ be a family of IPH functions. Then the function $f_{\inf}(x) = \inf_{t \in T} f_t(x)$ is IPH;
- let $(f_t)_{t \in T}$ be the same family and there exists a point $y \gg 0$ such that $\sup_{t \in T} f_t(y) < +\infty$ then the function $f_{\sup}(x) = \sup_{t \in T} f_t(x)$ is finite and IPH.

These properties allow us to give two more examples of IPH functions.

Example 2 The following maxmin functions are IPH:

1)

$$f(x) = \max_{k \in K} \min_{j \in J} \sum_{i \in I} a_i^{jk} x_i$$

where $a_i^{jk} \geq 0,\ k \in K, j \in J, i \in I$. Here J and K are finite sets of indices;

2)

$$f(x) = \max_{k \in K} \min_{j \in J_k} \sum_{i \in I} a_i^j x_i \tag{1}$$

where $a_i^j \geq 0, \quad j \in J_k, k \in K$. Here J_k and K are finite sets of indices.

Note that an arbitrary piecewise linear function f generated by a collection of linear functions $f^1, \dots, f^m$ can be represented in the form (1) (see [5]); hence an arbitrary piecewise linear function generated by nonnegative vectors is IPH.

Let $l \in \mathbb{R}^n_+, l \neq 0$ and $I(l) = \{i \in I \colon l_i > 0\}$. We consider the function $x \mapsto \langle l, x \rangle$ defined by the formula $l(x) = \langle l, x \rangle$ where the coupling function $\langle \cdot, \cdot \rangle$ is defined as

$$\langle l, x \rangle = \min_{i \in I(l)} l_i x_i\,. \tag{2}$$

Here $I(l) = \{i \in \{1, \dots, n\} \mid l_i > 0\}$. This function is called a min-type function generated by the vector l. We shall denote this function by the same symbol $l(x)$. Clearly a min-type function is IPH. It follows from Proposition 2 that:

- A finite function f defined on $\mathbb{R}^n_+$ is IPH if and only if
$$f(x) = \max\{\langle l, x \rangle \colon l \in H_1, l \leq f\}\,; \tag{3}$$
- Let $x^0 \in \mathbb{R}^n_+$ be a vector such that $f(x^0) > 0$ and $l = f(x^0)/x^0$. Then
$$\langle l, x \rangle \leq f(x)$$
for all $x \in \mathbb{R}^n_+$ and $\langle l, x^0 \rangle = f(x^0)$.

The vector $f(x^0)/x^0$ is called the support vector of a function f at a point x^0.

Lipschitz Functions

Definition 6 A function $f \colon X \to \mathbb{R}$ is called Lipschitz-continuous in X, if there exists a number $M > 0$ such that

$$\forall x, y \in X \colon |f(x) - f(y)| \leq M||x - y||\,.$$

The smallest such number is called the Lipschitz constant of f in the norm $|| \cdot ||$[1].

Let the set H_2 be the set of functions of the form

$$H_2 = \{h \colon h(x) = a - C||x - b||, \quad x, b \in \mathbb{R}^n, a \in \mathbb{R}, C \in \mathbb{R}_+\}\,.$$

Proposition 3 [23] *A function $f \colon \mathbb{R}^n \to \mathbb{R}_{+\infty}$ is H_2-convex if and only if f is a lower semicontinuous function. The H_2-subdifferential of f is not empty if f is Lipschitz.*

There is an interesting relation between IPH functions and Lipschitz functions, which allows one to formulate the problem of minimization of Lipschitz function over the unit simplex as the problem of minimization of IPH functions restricted to the unit simplex.

Theorem 1 *(see [23,25]). Let $f \colon S \to \mathbb{R}$ be a Lipschitz function and let*

$$M = \sup_{x, y \in S, x \neq y} \frac{|f(x) - f(y)|}{\|x - y\|_1} \tag{4}$$

[1] The norm $|| \cdot ||$ can be replaced by any metric, or, more generally, any distance function based on Minkowski gauge. For example, a polyhedral distance $d_P(x, y) = \max\{[(x - y), h_i] \mid 1 \leq i \leq m\}$, where $h_i \in \mathbb{R}^n, i = 1, \dots, m$ is the set of vectors that define a finite polyhedron $P = \bigcap_{i=1}^m \{x \mid [x, h_i] \leq 1\}$.

be the least Lipschitz constant of f in $\|\cdot\|_1$-norm, where $\|x\|_1 = \sum_{i\in I}|x_i|$. Assume that

$$\min_{x\in S} f(x) \geq 2M\,.$$

Then there exists an IPH function $g\colon \mathbb{R}^n_+ \to \mathbb{R}$ such that $g(x) = f(x)$ for all $x \in S$.

Methods

We consider the problem of global minimization of an H-convex function f on a compact convex set $D \subset X$,

$$\text{minimize } f(x) \quad \text{subject to } x \in D\,. \tag{5}$$

We will deal with the two mentioned cases of f being H_1-convex (IPH) and H_2-convex (Lipschitz).

Generalized Cutting Plane Method

A consequence of Propositions 2 and 3 is that we can approximate H-convex functions from below using a finite subset of functions from $\operatorname{supp}(f,H)$. Suppose we know a number of values of the function f at the points x^k, $k = 1,\dots,K$. Then the pointwise maximum of the support functions $h^k \in \partial^*_H f(x^K)$,

$$H^K(x) = \max_{k=1,\dots,K} h^k(x) \tag{6}$$

is a lower approximation, or underestimate of f. We have the following generalization of the classical cutting plane method by Kelley [16].

$K_{\max}$ is the limit on the number of iterations of the algorithm. The problem at Step 2.1 is called the auxiliary, or relaxed, problem. Its efficient solution is the key to numerical performance of the algorithm. For convex objective functions, H^K is piecewise affine, and the solution to the relaxed problem is done by linear programming. However, when we consider other abstract convex functions, like IPH or Lipschitz, the relaxed problem is not linear, but it also has a special structure that leads to its efficient solution.

```
Step 0. (Initialisation)
0.1   Set $K = 1$.
0.2   Choose an arbitrary initial point $x^1 \in D$.
Step 1. (Calculate H-subdifferential)
1.1   Calculate $h^K \in \partial^*_H f(x^K)$.
1.2   Define $H^K(x) := \max_{k=1,\dots,K} h^k(x)$, for all $x \in D$.
Step 2. (Minimize $H^K$)
2.1   Solve the Problem
        Minimize   $H^K(x)$   subject to   $x \in D$.
      Let $x^*$ be its solution.
2.2   Set $K := K + 1$, $x^K := x^*$.
Step 3. (Stopping criterion)
3.1   If $K < K_{max}$ and $f_{best} - H^K(x^*) > \epsilon$ go to
      Step 1.
```

Global Optimization: Cutting Angle Method, Algorithm 1 Generalized Cutting Plane Algorithm

Global Minimization of IPH Functions over Unit Simplex

In this section we present an algorithm for the search for a global minimizer of an IPH function f over the unit simplex S, that is we shall study the following optimization problem:

$$\text{minimize } f(x) \quad \text{subject to } x \in S \tag{7}$$

where f is an IPH function defined on $\mathbb{R}^n_+$. Note that an IPH function is nonnegative on $\mathbb{R}^n_+$, since $f(x) \geq f(0) = 0$. We assume that $f(x) > 0$ for all $x \in S$. It follows from positiveness of f that $I(l) = I(x)$ for all $x \in S$ and $l(x) = f(x)/x$.

Since $I(e^m) = \{m\}$, then the vector $l = f(e^m)/e^m$ can be represented in the form $l = f(e^m)e^m$ and

$$\langle f(e^m)e^m, x\rangle = f(e^m)x_m\,.$$

Remark 1 Note that $H^K(x) := \max_{k=1,\dots,K} \min_{i\in I(l^k)} l^k_i x_i \equiv \max\left\{H^{K-1}(x),\ \min_{i\in I(l^K)} l^K_i x_i\right\}$, which simplifies solution to the auxiliary problem at Step 2.1.

This Algorithm reduces the problem of global minimization (7) to the sequence of auxiliary problems. It provides lower and upper estimates of the global minimum f_* for the problem (7). Indeed, let $\lambda_K = \min_{x\in S} H^K(x)$ be the value of the auxiliary problem. It follows from (3) that

$$\langle l^k, x\rangle \equiv \min_{i\in I(l^k)} l_i^k x_i \le f(x) \text{ for all } x \in S,$$
$$k = 1,\dots,K\,.$$

Hence $H^K(x) \le f(x)$ for all $x \in S$ and $\lambda_K \equiv \min_{x\in S} H^K(x) \le \min_{x\in S} f(x)$. Thus λ_K is a lower estimate of the global minimum $f_\star$. Consider the number $\mu_K = \min_{k=1,\dots,K} f(x^k) =: f_{best}$. Clearly $\mu_K \ge f_*$, so μ_K is an upper estimate of $f_\star$. It is shown in [23] that λ_K is an increasing sequence and $\mu_K - \lambda_K \to 0$ as $K \to +\infty$. Thus we have a stopping criterion, which enables us to obtain an approximate solution with an arbitrary given tolerance.

Global Minimization of Lipschitz Functions

Method Based on IPH Functions By using Theorem 1, global minimization of Lipschitz function over the simplex S can be reduced to the global minimization of a certain IPH function over S.

Let $f\colon S \to \mathbb{R}$ be a Lipschitz function and let

$$c \ge 2M - \min_{x\in S} f(x)\,, \tag{8}$$

where M is defined by (4). Let $f_1(x) = f(x) + c$. It follows from Theorem 1 that the function f_1 can be extended to an IPH function g. The problem

$$\text{minimize } g(x) \quad \text{subject to} \quad x \in S \tag{9}$$

is clearly equivalent to the problem

$$\text{minimize } f_1(x) \quad \text{subject to} \quad x \in S\,. \tag{10}$$

Thus we apply the cutting angle method to solve problem (10). Clearly functions f and f_1 have the same minimizers on the simplex S. If the constant c in (8) is known, CAM is applied for the minimization of a Lipschitz function f over S with no modification. If c is unknown, we can assume that c is a sufficiently large number, however numerical experiments show that CAM is rather sensitive to the choice of c, in particular, when c is very large, the method converges very slowly. In order to estimate c we need to know an upper bound on the least Lipschitz constant M and a lower estimate of the global minimum of f.

If the feasible domain is not the unit simplex S but a polytope, it can be embedded into S with a simple change of variables. Solution to the constrained auxiliary problem in Step 2.1 of the algorithm was investigated in [8].

Direct Method Consider H_2-convex functions, which, by Proposition 3 include all Lipschitz functions. Let d_P be a polyhedral distance function. As a consequence of H_2-convexity, we can approximate Lipschitz functions from below using underestimates of the form

$$\begin{aligned} H^K(x) &= \max_{k=1,\dots,K} h^k(x) \\ &= \max_{k=1,\dots,K} (f(x^k) - C d_P(x, x^k))\,, \end{aligned} \tag{11}$$

where $C \ge M$, and M is the Lipschitz constant of f with respect to the distance d_P. Then we apply the Algorithm 1 to function f in the feasible domain D. The auxiliary problem as Step 2.1 becomes

$$\text{minimize} \max_{k=1,\dots,K} (f(x^k) - C d_P(x, x^k))$$
$$\text{subject to } x \in D\,.$$

```
Step 0. (Initialisation)
0.1  Take points $x^m = e^m$, $m = 1,\dots,n$. Set $K = n$.
0.2  Calculate $l^k = f(x^k)/x^k$, $k = 1,\dots,K$.
Step 1. (Calculate H-subdifferential)
1.1  Define $H^K(x) := \max_{k=1,\dots,K} \min_{i\in I(l^k)} l_i^k x_i$, for all
     $x \in S$.
Step 2. (Minimize $H^K$)
2.1  Solve the Problem
       Minimize $H^K(x)$ subject to $x \in S$.
     Let $x^*$ be its solution.
2.2  Set $K := K + 1$, $x^K := x^*$.
2.3  Compute $l^K = f(x^K)/x^K$
Step 3. (Stopping criterion)
3.1  If $K < K_{max}$ and $f_{best} - H^K(x^*) > \epsilon$ go to
     Step 1.
```

Global Optimization: Cutting Angle Method, Algorithm 2
Cutting Angle Algorithm for IPH functions

The same considerations about the convergence of the algorithm as those for Algorithm 2 are applied. Note that in the univariate case the underestimate H^K in (11) is exactly the same as the saw-tooth underestimate in Piyavski-Shubert method [20,26] if d_P is symmetric.

For minimization of Lipschitz functions, an estimate of the Lipschitz constant is required in both cases, when transforming f to an IPH function, or using Algorithm 1 directly. The crucial part in both methods is the efficient solution to the auxiliary problem in Step 2.1. The next section presents a very fast combinatorial algorithm for enumeration of all local minimizers of functions H^K.

The Auxiliary Problem

The *Step 2.1* (find the global minimum of $H^K(x)$) is the most difficult part of the cutting angle method. This problem is stated in the following form:

$$\text{minimize } H^K(x) \quad \text{subject to } x \in S \tag{12}$$

where

$$H^K(x) = \max_{k \le K} \min_{i \in I(l^k)} l_i^k x_i = \max_{k \le K} h^k(x)\,, \tag{13}$$

$K \ge n$, $l^k = f(x^k)/x^k$ are given vectors, $k = 1, \dots, K$. Note that $x^k = e^k,\ k = 1, \dots, n$.

Proposition 4 [2,3] *Let $K > n$, $l^k = l_k^k e^k$, $k = 1, \dots, n$, $l^k > 0$, $|I(l^k)| \ge 2$, $k = n+1, \dots, K$. Then each local minimizer of the function $H^K(x)$ defined by (13) over the simplex S is a strictly positive vector.*

Corollary 1 *Let $\{x^k\}$ be a sequence generated by Algorithm 2. Then $x^k \gg 0$ for all $k > n$.*

Let $\mathrm{ri}(S) = \{x \in S\colon x_i > 0 \text{ for all } i \in I\}$ be the relative interior of the simplex S. It follows from Proposition 4 and Corollary 1 that we can solve the problem (12) by sorting the local minima of the function H^K over the set ri (S). We now describe some properties of local minima of H^K on ri (S), which will allow us to identify these minima explicitly.

It is well known that functions h^k and H^K are directionally differentiable. Let $f'(x,u)$ denote directional derivative of the function f at the point x in the direction u. Also let

$$\begin{aligned} R(x) &= \{k\colon h^k(x) = H^K(x)\}\,, \\ Q_k(x) &= \{i \in I(l^k)\colon l_i^k x_i = h^k(x)\}\,. \end{aligned} \tag{14}$$

Proposition 5 *(see, for example, [13]). Let $x \gg 0$. Then*

$$(h^k)'(x,u) = \min_{i \in Q_k(x)} l_i^k u_i\,;$$

$$(H^K)'(x,u) = \max_{k \in R(x)} (h^k)'(x,u) = \max_{k \in R(x)} \min_{i \in Q_k(x)} l_i^k u_i\,.$$

Let $x \in S$. The cone

$$\begin{aligned} K(x,S) = \{&u \in \mathbb{R}^n\colon\ \exists \alpha_0 > 0 \\ &\text{such that } x + \alpha u \in S \ \ \forall \alpha \in (0,\alpha_0)\} \end{aligned}$$

is called the tangent cone at the point x with respect to the simplex S. The following necessary conditions for a local minimum hold (see, for example, [13]). Suppose $x \in \mathrm{ri}(S)$. Then $K(x,S) = \{u\colon \sum_{i \in I} u_i = 0\}$.

Proposition 6 *Let $x \in S$ be a local minimizer of the function H^K over the set S. Then $(H^K)'(x,u) \ge 0$ for all $u \in K(x,S)$.*

Applying Propositions 5 and 6 we obtain the following result.

Proposition 7 [2,3] *Let $x \gg 0$ be a local minimizer of the function H^K over the set ri (S), such that $H^K(x) > 0$. Then there exists an ordered subset $\{l^{k_1}, l^{k_2}, \dots, l^{k_n}\}$ of the set $\{l^1, \dots, l^K\}$ such that*

1)

$$x = \left(\frac{d}{l_1^{k_1}}, \dots, \frac{d}{l_n^{k_n}}\right) \text{ where } d = \frac{1}{\sum_{i \in I} \frac{1}{l_i^{k_i}}}\,; \tag{15}$$

2)

$$\max_{k \le K} \min_{i \in I(l^k)} \frac{l_i^k}{l_i^{k_i}} = 1; \tag{16}$$

3) Either $k_i = \{i\}$ for all $i \in I$ or there exists $m \in I$ such that $k_m \ge n+1$; if $k_m \le n$ then $k_m = m$;

4) if $k_m \ge n+1$ and $l_i^{k_m} \ne 0$ then $l_i^{k_m} > l_i^{k_i}$ for all $i \in I, i \ne m$.

Solution of the Auxiliary Problem

It follows from Propositions 4 and 7 that we can find a global minimizer of the function H^K defined by (13) over the unit simplex using the following procedure:

- sort all subsets $\{l^{k_1}, \ldots, l^{k_n}\}$ of the given set $l^1, \ldots, l^K$ vectors, such that (16) holds and $l_i^{k_m} > l_i^{k_i}$, $i \neq m$ if $k_m \geq n+1$, $i \in I(l^{k_m})$ and $k_m = m$ if $k_m \leq n$;
- for each such subset, find the vector x defined by (15);
- choose the vector with the least value of the function H^K among all the vectors described above.

Thus, the search for a global minimizer is reduced to sorting some subsets, containing n elements of the given set $\{l^1, \ldots l^K\}$ with $K > n$. Fortunately, Proposition 7 allows one to substantially diminish the number of sorted subsets.

The subsets $L = \{l^{k_1}, \ldots, l^{k_n}\}$ can be visualized with the help of an $n \times n$ matrix whose rows are given by the participating support vectors

$$L = \begin{pmatrix} l_1^{k_1} & l_2^{k_1} & \ldots & l_n^{k_1} \\ l_1^{k_2} & l_2^{k_2} & \ldots & l_n^{k_2} \\ \vdots & \vdots & \ddots & \vdots \\ l_1^{k_n} & l_2^{k_n} & \ldots & l_n^{k_n} \end{pmatrix} . \tag{17}$$

The conditions 2) and 4) of Proposition 7 are then easily interpreted as follows. Condition 4) implies that the diagonal elements of matrix L are smaller than elements in their respective columns, and condition 2) implies that the diagonal of L is not dominated by any other support vector $l^k \notin L$ (zero entries of matrix L are excluded from compaisons). Thus we obtain a combinatorial problem of enumerating all combinations L that satisfy conditions 2) and 4).

However it is impractical to enumerate all such combinations directly for large K. Fortunately there is no need to do so. It was shown in [6,7,8] that the required combinations can be put into a tree structure. The leaves of the tree correspond to the local minimizers of H^K, whereas the intermediate nodes correspond to the minimizers of $H^n, H^{n+1}, \ldots, H^{K-1}$.The incremental algorithm based on the tree structure makes computations very efficient numerically (as processing of queries using trees requires logarithmic time of the number of nodes). It is possible to enumerate several billions of local minimizers of H^K (e. g., when $n = 5$ and $K = 100,000$) in a matter of seconds on a standard Pentium IV based workstation.

The direct method of minimization of Lipschitz functions involves solution to a different auxiliary problem, that of minimizing H^K given in (11), with d_P being a simplicial distance function. It turns out that a very similar method of enumeration of local minimizers of H^K, by putting them in a tree structure, also works [9]. There is a counterpart of Proposition 7, with the difference that the support vectors are defined by

$$l_i^k = \frac{f(x^k)}{C} - x_i^k \,, \tag{18}$$

and the local minima and minimizers of H^K are identified through

$$\begin{aligned} d = H^K(x^*) &= \frac{C(Trace(L)+1)}{n} , \\ x_i^* &= \frac{d}{C} - l_i^{k_i}, i = 1, \ldots, n, \end{aligned} \tag{19}$$

where constant C is chosen greater or equal to the Lipschitz constant M of f in the simplicial distance d_P. Thus both versions of CAM, for IPH and for Lipschitz functions, share the same algorithm, but with different definitions of support vectors.

The actual algorithms for enumeration of local minima of H^K and maintaining the tree structure, as well as treatment of linear constraints, are presented in [7,8,9]. The algorithms involve a crucial fathoming step, and can be seen as branch-and-bound type algorithms [9,12,23].

Conclusions

Cutting angle methods are versions of the generalized cutting plane method for IPH, Lipschitz and other classes of abstract convex functions. The main idea of this deterministic method is to replace the original problem of minimizing f with a sequence of relaxed problems with special structure. The objective functions in the relaxed problems provides tight lower estimates of f, and the sequence of their solutions converge to the global minimum of f. Efficient solution to the relaxed problem makes CAM very fast on a class of global optimization problems.

Optimization is not the only field such underestimates are applied. Versions of CAM are also used for non-uniform random variate generation [10] and multivariate data interpolation [11].

Both versions of CAM described here have been successfully applied to a number of real life problems, including very difficult molecular geometry prediction and protein folding problems [12,17]. A software library GANSO for global and non-smooth optimization, which includes the cutting angle method, is available from http://www.ganso.com.au.

References

1. Andramonov MY, Rubinov AM, Glover BM (1999) Cutting angle method in global optimization. Appl Math Lett 12:95–100
2. Bagirov AM, Rubinov AM (2000) Global minimization of increasing positively homogeneous functions over the unit simplex. Ann Oper Res 98:171–187
3. Bagirov AM, Rubinov AM (2001) Modified versions of the cutting angle method. In: Hadjisavvas N, Pardalos PM (eds) Advances in Convex Analysis and Global Optimization. Kluwer, Dordrecht, pp 245–268
4. Bagirov AM, Rubinov AM (2003) The cutting angle method and a local search. J Global Optim 27:193–213
5. Bartels SG, Kuntz L, Sholtes S (1995) Continuous selections of linear functions and nonsmooth critical point theory. Nonlinear Anal TMA 24:385–407
6. Batten LM, Beliakov G (2002) Fast algorithm for the cutting angle method of global optimization. J Global Optim 24:149–161
7. Beliakov G (2003) Geometry and combinatorics of the cutting angle method. Optimization 52:379–394
8. Beliakov G (2004) The Cutting Angle Method – a tool for constrained global optimization. Optim Methods Softw 19:137–151
9. Beliakov G (2005) A review of applications of the Cutting Angle methods. In: Rubinov A, Jeyakumar V (eds) Continuous Optimization. Springer, New York, pp 209–248
10. Beliakov G (2005) Universal nonuniform random vector generator based on acceptance-rejection. ACM Trans Modelling Comp Simulation 15:205–232
11. Beliakov G (2006) Interpolation of Lipschitz functions. J Comp Appl Math 196:20–44
12. Beliakov G, Lim KF (2007) Challenges of continuous global optimization in molecular structure prediciton. Eur J Oper Res 181(3):1198–1213
13. Demyanov VF, Rubinov AM (1995) Constructive Nonsmooth Analysis. Peter Lang, Frankfurt am Main
14. Horst R, Pardalos PM, Thoai NV (1995) Introduction to Global Optimization. Kluwer, Dordrecht
15. Horst R, Tuy H (1996) Global Optimization: Deterministic Approaches, 3rd edn. Springer, Berlin
16. Kelley JE (1960) The cutting-plane method for solving convex programs. J SIAM 8:703–712
17. Lim KF, Beliakov G, Batten LM (2003) Predicting molecular structures: Application of the cutting angle method. Phys Chem Chem Phys 5:3884–3890
18. Pallaschke D, Rolewicz S (1997) Foundations of Mathematical Optimization (Convex Analysis without Linearity). Kluwer, Dordrecht
19. Pinter JD (1996) Global Optimization in Action. Continuous and Lipschitz Optimization: Algorithms, Implementation and Applications. Kluwer, Dordrecht
20. Piyavskii SA (1972) An algorithm for finding the absolute extremum of a function. USSR Comp Math Math Phys 12:57–67
21. Rockafellar RT (1970) Convex Analysis. Princeton University Press, Princeton
22. Rolewicz S (1999) Convex analysis without linearity. Control Cybernetics 23:247–256
23. Rubinov AM (2000) Abstract Convexity and Global Optimization. Kluwer, Dordrecht
24. Rubinov AM, Andramonov MY (1999) Minimizing increasing star-shaped functions based on abstract convexity. J Global Optim 15:19–39
25. Rubinov AM, Andramonov MY (1999) Lipschitz programming via increasing convex-along-rays fuinctions. Optim Methods Softw 10:763–781
26. Shubert BO (1972) A sequential method seeking the global maximum of a function. SIAM J Numerical Anal 9:379–388
27. Singer I (1997) Abstract Convex Analysis. Wiley-Interscience Publication, New York

Global Optimization: Envelope Representation

A. M. RUBINOV
School Inform. Techn. and Math. Sci.,
University Ballarat, Ballarat, Australia

MSC2000: 90C26

Article Outline

Keywords

Abstract convexity; Envelope; -subdifferential; Support set; Supremal generator; Min-type function; Cutting angle method; Global optimization; Lipschitz programming

Some classical methods of finite-dimensional convex minimization can be extended for quite broad classes of multi-extremal optimization problems. One successful generalization is based on the so-called *envelope representation* of the objective function.

We begin with the simplest case of a convex differentiable function f in order to introduce this approach. For such a function the *tangent hyperplane* $T = \{x\nabla f(y)(x - y)+ f(y) = 0\}$ is simultaneously a *support hyperplane*. That is, the inequality $f(x) \geq f(y)+ \nabla f(y)(x - y)$ holds for each x. This inequality can be expressed also in the following form: the affine function

$$h_y(x) = \nabla f(y)(x - y) + f(y) \qquad (1)$$

is a support function for the function f. Thus the function f can be represented as the pointwise maximum of the functions of the form h_y:

$$f(x) = \max_y h_y(x).$$

One of the main results of convex analysis asserts that an arbitrary lower semicontinuous convex function f (perhaps admitting the value $+\infty$) is the *upper envelope* (UE) of the set of all its affine minorants:

$$f(x) = \sup \left\{ h(x) \colon \begin{array}{c} h \text{ is an affine function,} \\ h \leq f \end{array} \right\} .$$

(The inequality $h \leq f$ stands for $h(x) \leq f(x)$ for all x.) The supremum above is attained if and only if the *subdifferential* of f at the point x is nonempty. Since affine functions are defined by means of linear functions, one can say that convexity is‘linearity + envelope representation’.

As it turns out the contribution of‘envelope representation’ to the convexity is fairly large. This observation stimulated the development of the rich theory of‘convexity without linearity’. (See [12,14,19] and references therein.) In particular, functions which can be represented as UE of subsets of a set of sufficiently simple functions are studied in this theory.

We need the following definition. Let H be a set of functions. A function f is called *abstract convex* (AC) with respect to H (or H-*convex*) if f is the UE of a subset from H, that is

$$f(x) = \sup \{h(x) \colon \; h \in H, h \leq f\} . \qquad (2)$$

The set H is called the *set of elementary functions*. For applications we need sufficiently simple elementary functions.

Many results from convex analysis related to various kinds of convex duality can be extended to *abstract convex analysis* Abstract convexity sheds some new lights to the classical *Fenchel–Moreau duality* and the so-called *level sets conjugation* (see [19]). The set $s(f, H) = \{h \in H{:}h \leq f\}$, presented in (2), is called the *support set* of f. The mapping $f \longmapsto s(f, H)$ is called the *Minkowski duality* ([9]). The support set accumulates a global information of a function f in terms of the set of elementary functions H and it can be useful in the study of global optimization problems involving the function f.

One of the main notions of convex analysis, which plays the key role for applications to optimization, is the subdifferential. There are two equivalent definitions of the subdifferential of a convex function. The first of them is based on the global behavior of the function. A linear function l is called a *subgradient* (i. e. a member of the subdifferential) of the function f at a point y if the affine function $h(x) = l(x)- (l(y)- f(y))$ is a *support function* with respect to f, that is $h(x) \leq f(x)$ for all x. The second definition has a local nature and is connected with local approximation of the function: the subdifferential is a closed convex set of linear functions such that the *directional derivative* $u \longmapsto f'_x(u)$ at the point x is presented as the UE of this set. For a differentiable convex function these two definitions reflect respectively support and tangent sides of the gradient.

The various generalizations of the second definition have led to development of the rich theory of *nonsmooth analysis*. The natural field for generalizations of the first definition is AC.

A function $h \in H$ is called the *subgradient* (or H-*subgradient*) of an H-convex function f at a point y if $f(x) \geq h(x)- (h(y)- f(y))$ for all x. The set $\partial_H f(y)$ of all subgradients of f at y is referred to as the *subdifferential* of the function f at the point y.

Let H' be the closure of the set H under vertical shifts, that is

$$H' = \left\{ h' \colon \begin{array}{c} h'(x) = h(x) - c , \\ h \in H, \; c \in \mathbf{R} \end{array} \right\} .$$

Clearly $h \in \partial_{H'} f(y)$ if and only if $f(y) = \max\{h'(y){:}h' \leq f, h' \in H'\}$. Thus if H is already closed under shifts then

$$\partial_H f(y) = \{h \in s(f, H) \colon \; h(y) = f(y)\} . \qquad (3)$$

Thus the subdifferential is not empty if and only if the supremum in (2) is attained.

Sometimes (3) is used for the definition of the subdifferential for an arbitrary set of elementary functions H (not necessary closed under shifts).

Many methods of convex minimization are based on the local properties of the convex subdifferential (more precisely, on the directional derivative). However there are some methods which exploit only the support property of the subdifferential. The conceptual schemes of these methods can be easily extended for AC functions. One of these methods is presented below.

Consider the following problem

$$f(x) \to \min, \quad x \in X, \tag{4}$$

where X is a compact set. Assume that f is AC with respect to a set of elementary functions H. We consider the following algorithm based on the *generalized cutting plane* idea, which is a nonlinear generalization of the classical cutting plane method.

```
0  Let $k := 0$. Choose an arbitrary initial point $x_0 \in X$;
1  Calculate a subgradient in the form (3) that is an
   element $h_k \in s(f, H)$ such that $h_k(x_k) = f(x_k)$;
2  Find a global optimum $y^*$ of the problem

       $\max_{0 \leq i \leq k} h_i(x) \to \min, \; x \in X.$    (5)

3  Let $x_{k+1} = y^*, \; k := k + 1$. Go to step 1.
```

Conceptual scheme (generalized cutting plane method)

Convergence of the sequence constructed by this procedure to a global minimizer has been proved under very mild assumptions by D. Pallaschke and S. Rolewicz [12]. Upper and lower estimates of the optimal value of the problem (4) can be computed, which lead to an efficient stopping criterion (compare with [2]).

There are two major difficulties in the numerical implementation of the Algorithm. The first is the calculation of a subgradient. In general it is very difficult to find it numerically, however it is possible in several important particular cases. The second difficulty is the solution of the auxiliary problem (5). This is a linear programming problem in the case of the set H of affine functions, but for sets of more complicated functions the problem (5) is essentially of a combinatorial nature or a problem of convex maximization.

The simplest example of this approach is *Lipschitz programming*. If f is a Lipschitz function we can, for example, take as H the set of functions h of the form $h(x) = -$ a $\|x - x_o\| - c$, where a is a positive and c is a real number, $x_o \in X$. In order to find an H-subgradient we should take $a > L$ where L is the Lipschitz constant of the function f; thus we need to know an upper estimate of this constant; this is a special piece of global information about this function. With such H the problem (4) can be reduced to a sequence of special problems of concave minimization. Some known algorithms of Lipschitz programming fall within the described approach [11,21].

For fairly large classes of functions defined on the cone $\mathbf{R}^n_+$ of all n-vectors with nonnegative coordinates it is possible to take as H a set of functions which includes as its main part a *min-type function* of the form

$$\begin{aligned} &l(x) = \min_{i \in \mathcal{T}(l)} l_i x_i, \quad x \in \mathbf{R}^n_+, \\ &\text{with } \mathcal{T}(l) = \{i \colon \; l_i > 0\}\,. \end{aligned} \tag{6}$$

We define the infimum over empty set to be zero. If l is a strictly positive vector and c a positive number then the set $\{x \colon \min_i l_i x_i \leq c\}$ is a complement to a 'right angle'. Exploiting min-type functions instead of linear functions allows us to separate a point from the (not necessary convex) set by the complements of 'right angles'.

Various classes of elementary functions arise, based on the set L of all functions of the form (6) with $l \in \mathbf{R}^n_+$. In particular, L itself and sets

$$\begin{aligned} H_1 &= \{h \colon \; h(x) = l(x) - c, \; l \in L, \; c \in \mathbf{R}\}\,, \\ H_2 &= \{h \colon \; h(x) = \min(l(x), \; c), \; l \in L, \; c \in \mathbf{R}\} \end{aligned}$$

are convenient for applications. The classes of AC with respect to H_1 and H_2 functions are quite large [14]. The first of them consists of all increasing (with respect to the usual order relation) functions f such that the function of a real variable $t \to f(tx)$, $t \in [0, +\infty)$, is convex for all $x \in \mathbf{R}^n_+$. This class contains all homogeneous functions of degree $\delta \geq 1$, their sums and UE of sets of such functions. In particular it contains all polynomials with nonnegative coefficients. The second class consists of all increasing functions f such that $f(tx) \geq tf(x)$ for all $x \in \mathbf{R}^n_+$ and $t \in [0, 1]$. Concave increasing functions f with $f(0) \geq 0$ and UE of sets of such functions belong to this class. Also, positively homogeneous functions of degree $\delta \leq 1$, their sums and UE of sets of such functions belong to it.

For minimizing AC functions with respect to H_i (i = 1, 2) we need again to calculate the H_i-subgradients in the form (5) and then to reduce the problem (4) to a sequence of auxiliary problems. A version of the generalized cutting plane method in such a case is called *'cutting angle method'* ([2,14]).

A.M. Rubinov et al. ([1,14,16,17]) have demonstrated that for AC functions generated by various classes of min-type functions it is possible to find subgradients very easily. In particular, only the number $f(x)$ (resp. $f'(x, x)$) is required for the calculation of an element of $\partial_{H_2} f(x)$ (resp. $\partial_{H_1} f(x)$), without any additional information about a global behavior of the function f. Thus the main problem with implementation of the cutting angle method is to solve the auxiliary subproblem, which is a problem of the mixed integer programming of a special kind in this case.

Let L be the set of all functions (6) with $l \in \mathbf{R}^n_+$. It can be shown ([14,16]) that a function f defined on $\mathbf{R}^n_+$ is L-convex if and only if f is IPH (increasing and positively homogeneous of degree one).*IPH functions* can serve for the miminization of a Lipschitz function over the unit simplex $S_n = \{x \in \mathbf{R}^n_+ : \sum_i x_i = 1\}$. First ([14,15]), for each Lipschitz function g defined on S_n there exists a constant c>0 such that the function $\widetilde{g}(x) = g(x) + c$ can be extended to an IPH function defined on $\mathbf{R}^n_+$. Second, the auxiliary problem (5) for problem (4) with an IPH function f and $X = S_n$, has a special structure and can be efficiently solved for fairly large n (see [14, Chap. 9] and references therein). Thus, the minimization of a Lipschitz function over the unit simplex can be efficiently accomplished by the cutting angle method.

Numerical experiments demonstrate that a combination of the cutting angle method with a local search is very efficient, since the cutting angle method allows one to leave a local minimizer fairly quickly.

Envelope representation is useful also in the study of some theoretical problems arising in optimization. Many interesting examples of such applications can be found in the books [12,14,19]. In particular, a general scheme of penalty and augmented Lagrangian based on the notion of the subdifferential is presented in [12]. I. Singer [19] demonstrated that Fenchel–Moreau duality leads to a unified theory of duality results for very general optimization problems. It can be shown [18] that AC forms the natural framework for the study of *solvability theorems* (generalizations of Farkas' lemma; cf. ► Farkas lemma; ► Farkas lemma: Generalizations). In contrast with numerical methods based on applications of subdifferentials, the study of solvability theorems is based on application of support sets. AC serves also for the study of some problems of quasiconvex minimization (see for example [10,13,20]).

A subset H of a set X of functions is called the *supremal generator* ([9]) of X if each function from X is AC with respect to H. There exist very small supremal generators of very large classes of functions. The following two examples of such supremal generators are useful for nonsmooth optimization.

1) Recall that a function f is called *positively homogeneous* (PH) of degree k if $p(\lambda x) = \lambda^k p(x)$ for $\lambda > 0$. It can be shown ([14]) that the set of all functions of the form

$$h(x) = -a \left(\sum_{i=1}^{n} x_i^2 \right)^{\frac{1}{2}} + \sum_{i=1}^{n} l_i x_i , \qquad (7)$$

where $a \leq 0$, $l_1, \ldots, l_n$ are real numbers is a supremal generator of the set PH_1 of all lower semicontinuous PH functions of degree one defined on n-dimensional space $\mathbf{R}^n$. Since each function (7) is concave it follows that the set of all concave PH functions of degree one is a supremal generator of PH_1.

2) It can be shown ([3,4,9,14]) that the set $\mathcal{H}$ of all quadratic functions h of the form

$$h(x) = -a \sum_{i=1}^{n} x_i^2 + \sum_{i=1}^{n} l_i x_i + c , \qquad (8)$$

where $a \geq 0$, $l_1, \ldots, l_n$, c are real numbers is a supremal generator of the set of all lower semicontinuous functions f:$\mathbf{R}^n \to \mathbf{R} \cup \{+\infty\}$ minored by $\mathcal{H}$ in the following sense: there exists $h \in \mathcal{H}$ such that $f \geq h$.

Supremal generators are a convenient tool in the study of nonsmooth optimization problems. A local approximation of the first (resp. second) order of a nonsmooth function is fulfilled very often by various kinds of generalized derivatives of the first (resp. second) order, which are PH functions of the first (resp. second) degree. Practical applications of these derivatives to optimization are based on their representation in terms of linear (resp. quadratic) functions.

Linearization of lower semicontinuous PH functions of the first degree can be accomplished by supremal generators of the space PH_1, consisting of concave functions. Each finite concave function $g \in \mathrm{PH}_1$ can be presented as $\min\left\{l(x)\colon\ l \in \overline{\partial} g(0)\right\}$ where $\overline{\partial} g(0)$ is the superdifferential (in the sense of convex analysis) of this function g at the origin. Hence each function $g \in \mathrm{PH}_1$ can be linearized by the operation sup min.

The second order approximation of a nonsmooth function f at a point x can be accomplished by the *subjet*, that is the set

$$\partial^{2,-} f(x) = \left\{ (\nabla g(x), \nabla^2 g(x))\colon \begin{array}{c} f - g \text{ has a} \\ \text{local minimum } x \\ \text{with } g \in C^2(\mathbf{R}^n) \end{array} \right\} .$$

(Here $\nabla\ g(x)$ (resp. $\nabla^2\ g(x)$) stands for the gradient (resp. Hessian) of a function g at a point x.) Let $\mathcal{H}$ be the set of all functions of the form (8). It can be shown (see [5,6]) that the subjet $\partial^{2,-} f(x)$ is nonempty if and only if the $\mathcal{H}$-subdifferential $\partial_{\mathcal{H}} f(x)$ is not empty. AC with respect to $\mathcal{H}$ can also serve for supremal representation of the second order generalized derivatives of nonsmooth functions in terms of quadratic functions (see[5]).

See also

- ▶ Dini and Hadamard Derivatives in Optimization
- ▶ Nondifferentiable Optimization
- ▶ Nondifferentiable Optimization: Cutting Plane Methods
- ▶ Nondifferentiable Optimization: Minimax Problems
- ▶ Nondifferentiable Optimization: Newton Method
- ▶ Nondifferentiable Optimization: Parametric Programming
- ▶ Nondifferentiable Optimization: Relaxation Methods
- ▶ Nondifferentiable Optimization: Subgradient Optimization Methods

References

1. Abasov TM, Rubinov AM (1994) On the class of H-convex functions. Russian Acad Sci Dokl Math 48:95–97
2. Andramonov MYu, Rubinov AM, Glover BM (1999) Cutting angle methods in global optimization. Applied Math Lett 12:95–100
3. Balder EJ (1977) An extension of duality-stability relations to nonconvex optimization problems. SIAM J Control Optim 15:329–343
4. Dolecki S, Kurcyusz S (1978) On Φ-convexity in extremal problems. SIAM J Control Optim 16:277–300
5. Eberhard A, Nyblom M (1998) Jets, generalized convexity, proximal normality and differences of functions. Nonlinear Anal (TMA) 34:319–360
6. Eberhard A, Nyblom N, Ralph D (1998) Applying generalized convexity notions to jets. In: Croizeix J-P, Martinez-Legaz J-E, Volle M (eds) Generalized Convexity, Generalized Monotonicity. Kluwer, Dordrecht, pp 111–158
7. Horst R, Pardalos PM (eds) (1996) Handbook of global optimization. Kluwer, Dordrecht
8. Kelley J (1960) The cutting plane method for solving convex programs. SIAM J 8:703–712
9. Kutateladze SS, Rubinov AM (1972) Minkowski duality and its applications. Russian Math Surveys 27:137–191
10. Martinez-Legaz J-E (1988) Quasiconvex duality theory by generalized conjugation methods. Optim 19:603–652
11. Mladineo RH (1986) An algorithm for finding the global maximum of a multimodal, multivariate function. Math Program 34:188–200
12. Pallaschke D, Rolewicz S (1997) Foundations of mathematical optimization (convex analysis without linearity). Kluwer, Dordrecht
13. Penot JP, Volle M (1990) On quasiconvex duality. Math Oper Res 15:597–625
14. Rubinov AM (2000) Abstract convexity and global optimization. Kluwer, Dordrecht
15. Rubinov AM, Andramonov MYu (1999) Lipschitz programming via increasing convex-along-rays functions. Optim Methods Softw 10:763–781
16. Rubinov AM, Andramonov MYu (1999) Minimizing increasing star-shaped functions based on abstract convexity. J Global Optim 15:19–39
17. Rubinov AM, Glover BM (1999) Increasing convex-along-rays functions with applications to global optimization. J Optim Th Appl 102(3)
18. Rubinov AM, Glover BM, Jeyakumar V (1995) A general approach to dual characterization of solvability of inequality systems with applications. J Convex Anal 2:309–344
19. Singer I (1997) Abstract convex analysis. Wiley/Interscience, New York
20. Volle M (1985) Conjugasion par tranches. Ann Mat Pura Appl 139:279–312
21. Wood GR (1992) The bisection method in higher dimensions. Math Program 55:319–337

Global Optimization: Filled Function Methods

Hong-Xuan Huang
Department of Industrial Engineering, Tsinghua University, Beijing, People's Republic of China

MSC2000: 90C26, 90C30, 90C59, 65K05

Article Outline

Keywords and Phrases

Basin; Filled function; Filled function method

Introduction

The *filled function methods* describe a class of global optimization methods for attacking the problem of finding a global minimizer of a function $f\colon X \to \Re$ over a certain subset $X \subset \Re^n$. Each variant of such methods replaces the objective function $f(x)$ by a specific auxiliary function that is associated with a local minimum and some parameters in every iteration, and is minimized through some local search strategies. The term "filled function" means that every auxiliary function can fill the region of attraction at a certain neighborhood of a local minimum of the objective function.

The definition of a filled function involves some basic concepts. The term "basin" was introduced first in [1]. A *basin* of a function $f(x)$ at an isolated minimizer x_1^* denotes a connected domain B_1^* which contains x_1^* and in which starting from any point the steepest descent trajectory of $f(x)$ converges to x_1^*, but outside of which the steepest descent trajectory of $f(x)$ does not converge to x_1^*. Accordingly, a *hill* of a function $f(x)$ at a maximizer x_1^* is a basin of $-f(x)$ at the point x_1^*.

In addition, the basin B_2^* at a minimizer x_2^* is *lower* (or *higher*) than the basin B_1^* at another minimizer x_1^* if the following inequality holds:

$$f(x_2^*) < f(x_1^*) \text{ (or } f(x_2^*) \geq f(x_1^*)) \,.$$

Definitions

The first kind of filled function method was proposed in [5] for the unconstrained optimization problem

$$\min_{x \in \Re^n} f(x) \,.$$

The corresponding filled function involved two parameters, and was defined by

$$P(x, x_1^*, r, \rho) = \frac{1}{r + f(x)} \exp\left(-\frac{\|x - x_1^*\|^2}{\rho^2}\right) \,, \quad (1)$$

where x_1^* is a minimizer of the objective function $f(x)$, and r and ρ are parameters such that $r + f(x_1^*) > 0$, $\rho > 0$. In order to demonstrate the principle of the filled function method, people usually assume that the function $f(x)$ is twice continuously differentiable and coercive, i. e., its Hessian is continuous and the following condition holds:

$$\lim_{\|x\| \to +\infty} f(x) = +\infty \,. \quad (2)$$

It is also assumed that the function $f(x)$ has only a finite number of minimizers in a closed domain $\Omega \subset \Re^n$ that contains all global minimizers of $f(x)$.

Under certain other conditions concerning the parameters r and ρ, the function $P(x, x_1^*, r, \rho)$ defined in (1) has three properties as follows:

(a) x_1^* is a maximizer of $P(x, x_1^*, r, \rho)$ and the whole basin B_1^* at x_1^* becomes a part of a hill of $P(x, x_1^*, r, \rho)$ at x_1^*.
(b) $P(x, x_1^*, r, \rho)$ has no minimizers or saddle points in any higher basin of $f(x)$ than B_1^* at x_1^*.
(c) $f(x)$ has a lower basin B than B_1^* at x_1^*, then there is a point x' in such a basin B that minimizes $P(x, x_1^*, r, \rho)$ on the line through x' and x_1^*.

A function satisfying the above three properties is said to be a *filled function* of $f(x)$ at the local minimizer x_1^*. Note that the above definition just lists the main properties required for a filled function, in which the number of parameters is not an important factor (see the discussion about categories of filled functions below).

Usually, when people develop a variant of the filled function method, property c in the above definition may be replaced by a similar one. For example, it was replaced in [21] by

(c$_1$) If $f(x)$ has a basin B_2^* at x_2^* that is lower than B_1^*, then there is a point $x' \in B_2^*$ that minimizes $P(x, x_1^*, r, \rho)$ on the line through x_1^* and x'', for every x'' in some neighborhoods of x_2^*.

Note that property c$_1$ is much stronger than that required in [5] since a minimizer is required for lines connecting the current minimizer with every point in some neighborhoods of a next better minimizer.

In addition, for the unconstrained global optimization problem, in [16] two classes of continuously differentiable filled functions with multiplicative and additive structures, respectively, were proposed which assumed the existence of a local minimizer in a lower basin but not just on lines.

Under such assumptions as the objective function $f\colon \Re^n \to \Re$ is coercive, continuously differentiable and has finite local minimizers, another stronger variant of the filled functions can be found in [18], where the concept of a basin at a local minimizer was extended to that of a *G-basin*. A subset $B^* \subset \Re^n$ is said to be a *G-basin* of $f(x)$ corresponding to a local minimizer x^* if it is a connected domain with the following properties:

(i) $f(x) \geq f(x^*)$ for any $x \in B^*$;
(ii) $\bar{x} \in B^*$ is a local minimizer of $f(x)$ if and only if $f(\bar{x}) = f(x^*)$.

The definition in [18] requires that a filled function $p(x)$ is differentiable and satisfies some modifications of conditions a and b as follows:

(a′) *x_1^* is a strictly local maximizer of $p(x)$.*
(b′) *For any $x \neq x^*$ satisfying $f(x) \geq f(x^*)$, x is not a stationary point of $p(x)$.*

Furthermore, any lower local minimizer $\bar{x}$ of $f(x)$ than a nonglobal minimizer x^* is also a local minimizer of the filled function and is lower than every point on the boundary of the box set Ω which contains all global minimizers of $f(x)$. For points higher than x^* in Ω, the farther they are from x^* implies a lower value of the filled function.

Recently, in order to take advantages of filled functions and reduce the difficulty in adjusting the value of parameters, the concept a locally filled function was introduced in [9,22], which was based on the concept of a local basin.

Given a bounded and closed convex set $\omega \subset \Omega$ and a basin B_1 of the objective function $f(x)$ at a local minimizer x_1^*, if the set

$$B_1(\omega) := \omega \cap B_1 \neq \emptyset ,$$

then $B_1(\omega)$ is called a *local basin* associated with x_1^* and ω. Furthermore, a continuously differential function $P(x)$ is said to be a *locally filled function* associated with ω at a local minimizer $x_1{}^*$ of $f(x)$ if the following conditions hold:

(a$_2$) x_1^* is an interior point of ω and a strict local maximizer of $P(x)$.
(b$_2$) If $B_1(\omega)$ is a local basin containing the point x_1^*, then $P(x)$ does not have any local minimizer or saddle point in $B_1(\omega)$.
(c$_2$) If there exist local basins lower than $B_1(\omega)$, then at least one of such local basins, e. g., $B_2(\omega)$, satisfies the following condition: There is a point $x_2 \in B_2(\omega)$ such that $P(x)$ decreases strictly along the segment connecting x_1^* and x_2, that is, $P((1-\alpha)x_1^* + \alpha x_2)$ is decreasing strictly with respect to $\alpha \in [0,1]$.

In [9,22], the difference between a filled function and a locally filled function was illustrated by such a function $y = f(x)$ defined on the interval [–0.5,0.5] as

$$f(x) = z_1(\sin(12\pi x) + 1.5) ,$$

where the variable z_1 was defined by

$$z_1 = \log(z_2 + 10^{-5}) + 10 ,$$

$$z_2 = \left(\left(x - \frac{1}{4}\right)^2 \left(x + \frac{1}{4}\right)^2 + 10^{-4}\right) x^2 .$$

Note that $x^* = 0.2366$ is one of its local minimizers. An auxiliary function

$$Q(x, x^*, A) = -[f(x) - f(x^*)] \exp\left(A\|x - x^*\|^2\right)$$

does not satisfy the definition of the filled function on [–0.5,0.5] for the parameter $A = 16$, but it satisfies all conditions associated with a locally filled function for the parameter $A = 16$ and the choice of the interval $\omega = [-0.1, 0.3]$.

Methods

If the objective function $f\colon \Re^n \to \Re$ is coercive, then its global minimizer can be found in a suitable large bounded closed set $\Omega \subset \Re^n$ which should be explored completely. In general, let us denote the feasible region for minimizing $f\colon X \subset \Re^n \to \Re$ by Ω, and assume that for any point $x \in \partial\Omega$, $f(x) > \min_{y\in\Omega} f(y)$.

The *basic outline of filled function methods* can be described as follows:

Step 1 Choose an initial point $x_1 \in \Omega$. Denote the maximum of the iteration number and the index of the iterative process by Iter_No and k, respectively. Set $k = 0$.

Step 2 Minimize the function $f(x)$ in Ω starting from the point $x_1 \in \Omega$ and obtain a local minimizer x_1^* of $f(x)$. Denote the basin of the objective function f at x_1^* by B_1^*.

Step 3 Choose two suitable parameters r and ρ, and construct a filled function $P(x, x_1^*, r, \rho)$ associated with x_1^* and f, for example, which is defined by (1).

Step 4 Minimize the filled function $P(x, x_1^*, r, \rho)$ and find another point x_2 in a lower basin B_2^* of f than B_1^* if such a point x_2 exists for a suitable choice of parameters r and ρ.

Step 4.1 If a lower basin B_2^* of f than B_1^* at x_1^* is found, then a new local minimizer x_2^* can be obtained by any local search strategy. Furthermore, perform the replacement of variables such as

$$x_2^* \to x_1^*, \quad B_2^* \to B_1^*, \quad k+1 \to k\,,$$

and go to step 3 (The method continues searching for a global minimum by minimizing another filled function corresponding to the local minimizer x_2^*).

Step 4.2 Otherwise, either the parameters should be adjusted again by an internal updating strategy, or no better local minimizer than x_1^* can be found in Ω.

Step 5 If the iterative index $k >$ Iter_No, or no better local minimizer of f can be found in Ω, the current best local minimizer will be regarded as a global minimizer of f in Ω.

In the above outline of filled function methods, how to choose parameters in a filled function is an important issue, and it may be implemented through an internal iterative process for minimizing $P(x, x_1^*, r, \rho)$ approximately in order to find a lower basin of f or an increasing direction $\bar{x} - x_1^*$ for $P(x, x_1^*, r, \rho)$ at a point $\bar{x}$. An algorithmic implementation and some practical considerations can be found in [5].

Until now people have proposed many kinds of filled functions, for which some are general, while many others are specific [3,5,6,7,8,10,11,12,13,14,15,17,20,21,23]. These filled functions can be classified into four categories.

Two-Parameter Filled Functions

A two-parameter filled function was presented in (1). Although the first filled function method was proposed to deal with unconstrained optimization problems, the two-parameter filled function method had been extended to find a constrained global minimizer [3].

The constrained optimization problem can be formulated as follows:

$$\begin{aligned} &\text{Minimize } f(x),\\ &\text{subject to } g_i(x) \ge 0, \quad i \in \mathcal{I}\,,\\ &\qquad\qquad h_j(x) = 0, \quad j \in \mathcal{E}\,, \end{aligned} \tag{3}$$

where $\mathcal{I}$ and $\mathcal{E}$ are indices sets corresponding to inequalities and equalities, respectively. The two-parameter filled function for problem (3) is defined by

$$P_F(x, x_1^*, r, \rho) = \frac{1}{r + F(x)} \exp\left(-\frac{\|x - x_1^*\|^2}{\rho^2}\right), \tag{4}$$

where

$$F(x) = f(x) + \sum_{i\in\mathcal{I}} \lambda_i \max\{0, -g_i(x)\} + \sum_{j\in\mathcal{E}} \mu_j |h_j(x)| \tag{5}$$

is an exact penalty function for the constrained minimization problem (3), and $\lambda \in \Re_{++}^{|\mathcal{I}|}$, $\mu \in \Re_{++}^{|\mathcal{E}|}$. Since the function defined by (4) is a nonsmooth filled function, the definitions such as basin and filled function should be modified accordingly, see [3].

Two-parameter filled functions have two disadvantages. One is that the changes of both the filled function and its gradient (if available) are affected by the term $\exp(-\|x - x_1^*\|^2/\rho^2)$. When $\|x - x_1^*\|^2$ is large, it is difficult to distinguish these changes, so some pseudo-minimizers, or saddle points or higher minimizers of the filled functions may be located. The other is that the coordination between r and ρ is very difficult; even a global minimizer x^* may be lost for an improper setting of parameters.

Several modified two-parameter filled functions were proposed in [7] as follows:

$$\tilde{P}(x, x_1^*, r, \rho) = \frac{1}{r + f(x)} \exp\left(-\frac{\|x - x_1^*\|}{\rho^2}\right),$$
$$G(x, x_1^*, r, \rho) = -\rho^2 \log[r + f(x)] - \|x - x_1^*\|^2,$$
$$\tilde{G}(x, x_1^*, r, \rho) = -\rho^2 \log[r + f(x)] - \|x - x_1^*\|.$$

A more general form of filled functions with two parameters can be found in [20]:

$$P(x, r, A) = \psi(r + f(x))\exp(-Aw(\|x - x_k^*\|^\beta)), \quad (6)$$

where $\beta \geq 1$, $A > 0$, the parameter r is chosen such that $r + f(x) > 0$ for all $x \in \Omega$, and the functions $\psi(t)$, $w(t)$ have the following properties:

(i) $\psi(t)$ and $w(t)$ are continuously differentiable for $t \in (0, +\infty)$.
(ii) For $t \in (0, +\infty)$, $\psi(t) > 0$, $\psi'(t) < 0$ and $\psi'(t)/\psi(t)$ is monotonically increasing.
(iii) $w(0) = 0$ and for any $t \in (0, +\infty)$, $w(t) > 0$, $w'(t) \geq c > 0$.

Note that choices for the functions $\psi(t)$ and $w(t)$ can be $1/t^a (a > 0)$, $\operatorname{csch}(t)$, $\exp(1/t) - 1, \ldots$ and $t, \sinh(t), e^t - 1, \ldots$, respectively. The general form of filled functions in (6) includes the class of generalized filled functions considered in [24], which are special two-parameter filled functions.

Since the above filled functions may tend to zero or $-\infty$ as $\|x\| \to +\infty$ for some objective functions $f(x)$ or $F(x)$, they do not approximate a coercive objective function properly. In such a case, a coercive filled function may be preferred. In [8] the concept of a globally convexized filled function for a twice continuously differentiable function $f\colon \Omega \to \Re$ was introduced.

A continuous function $U(x)$ is a *globally convexized filled function* if it has three properties:

(a) $U(x)$ has no stationary point in the region

$$S_1 = \{x \mid f(x) \geq f(x_1^*), x \in \Omega\},$$

except a prefixed point $x_0 \in S_1$ that is a minimizer of $U(x)$.

(b) $U(x)$ has a minimizer in the region (if it exists)

$$S_2 = \{x \mid f(x) < f(x_1^*), x \in \Omega\}.$$

(c) $\lim_{\|x\| \to +\infty} U(x) = +\infty$.

Two successful globally convexized filled functions can be found in [8] as follows:

$$U_1(x, x_1^*, x_0, A, h) = \|x - x_0\| \times \arctan\{A[f(x) - f(x_1^*) + h]\},$$
$$U_2(x, x_1^*, x_0, A, h) = \|x - x_0\| \times \tanh\{A[f(x) - f(x_1^*) + h]\}.$$

In general, such globally convexized filled functions may be expressed by

$$U(x, x_1^*, x_0, A, h) = \eta(\|x - x_0\|)\phi(A[f(x) - f(x_1^*) + h])$$

for a large enough $A > 0$ and a suitable parameter h such that

$$0 < h < f(x_1^*) - f(x^*),$$

where x^* is a global minimizer of $f(x)$, x_1^* is not a global but is a local minimizer of $f(x)$, and $\eta(t)$ and $\varphi(t)$ are continuously differentiable univariate functions satisfying the following conditions [8]:

(i) $\eta(0) = 0, \eta'(t) \geq \alpha > 0, \forall t \geq 0$.
(ii) $\phi(0) = 0$, $\phi(t)$ is monotonically increasing for all $t \in \Re$ (or for $t \in (-t_1, +\infty)$, where $t_1 > 0$).
(iii) $\phi'(t) > 0, \forall t \in \Re$ (or $\phi'(t) > 0, \forall t \in (-t_1, +\infty)$, where $t_1 > 0$).
(iv) When $t \to +\infty$, $\phi'(t)$ is monotonically decreasing to 0 at least as fast as $1/t$.

Note that choices for these two functions can be t, $\tan(t)$, $e^t - 1, \ldots$ for $\eta(t)$ and $\arctan t, \tanh(t), 1 - e^{-t}, \ldots$ for $\phi(t)$.

Single-Parameter Filled Functions

In order to reduce the difficulty in coordination between r and ρ in a two-parameter filled function, several single-parameter filled functions were proposed in [7]:

$$Q(x, x_1^*, A) = -[f(x) - f(x_1^*)] \exp\left(A\|x - x_1^*\|^2\right),$$
$$\tilde{Q}(x, x_1^*, A) = -[f(x) - f(x_1^*)] \exp\left(A\|x - x_1^*\|\right),$$
$$\nabla E(x, x_1^*, A) = -\nabla f(x) - 2A[f(x) - f(x_1^*)](x - x_1^*),$$
$$\nabla \tilde{E}(x, x_1^*, A) = -\nabla f(x) - A[f(x) - f(x_1^*)]\frac{x - x_1^*}{\|x - x_1^*\|}.$$

More and more single-parameter filled functions appeared afterwards. For example,

$$H(x, x_1^*, a) = \frac{1}{\ln[1 + f(x) - f(x_1^*)]} - a\|x - x_1^*\|^2$$

was proposed in [11], which is defined only for the region where $f(x) \geq f(x_1^*) - 1$. The L function

$$L(x, x_1^*, a) = -\rho\|x - x_1^*\|^2 - [f(x) - f(x_1^*)]^{1/m}$$

and the mitigated L_2 function

$$\mathrm{ML}_2(x, x_1^*, a) = \rho\phi\left(\frac{1}{\|x - x_1^*\|^p}\right) - [f(x) - f(x_1^*)]^{1/m}$$

were proposed in [12] and [13], respectively, where $m > 1$ is a prefixed natural number, ρ is a positive parameter, and φ is a mitigator. A function $y\colon \Re \to \Re$ is said to be a *mitigator* if it is a twice continuously differentiable function in its domain and has the following properties:

(i) $y(0) = 0$, $y'(t) > 0$, and $y''(t) < 0$ for all $t > 0$.

(ii) $\lim_{t \to +\infty} y(t)$ exists.

Note that the ML_2 function can reduce the negative definite effect in the Hessian of a single-parameter filled function such as the L function significantly. The numerical results and generalizations can be found in [12,13,14,15].

A more general form for the single-parameter filled functions can be expressed by

$$Q(x, A) = -\phi(f(x) - f(x_k^*))\exp(Aw(\|x - x_k^*\|^\beta)) ,$$

where $\beta \geq 1$, $A > 0$, and the functions $\varphi(t)$ and $w(t)$ have the following properties [20]:

(i) $\varphi(t)$ is continuously differentiable for $t \geq 0$.

(ii) $\phi(0) = 0, \phi'(t) > 0,\ \forall t \geq 0$.

(iii) $\phi'(t)/\phi(t)$ is monotonically decreasing for $t \in (0, +\infty)$.

(iv) $w(0) = 0$ and for any $t \in (0, +\infty)$, $w(t) > 0$, $w'(t) \geq c > 0$.

Note that the choices for these functions can be t, $a^t - 1 (a > 1)$, $\sinh(t)$, ... for $\varphi(t)$ and $t, \sinh(t), e^t - 1$, ... for $w(t)$.

In order to avoid the influence of the exponential term, a general single-parameter filled function can be set by

$$U(x, A) = -\eta(f(x) - f(x_k^*)) - Aw(\|x - x_k^*\|^\beta) ,$$

where the function $\eta(t)$ is continuous on $[0, +\infty)$ and is differentiable in $(0, +\infty)$. Furthermore, the functions $\eta(t)$ and $w(t)$ have the following properties [20]:

(i) $\eta(0) = 0$;

(ii) $\eta'(t) > 0$ is monotonically decreasing for $t \in (0, +\infty)$ and $\lim_{t \to 0^+} \eta'(t) = +\infty$;

(iii) $w(0) = 0$ and for any $t \in (0, +\infty)$, $w(t) > 0$, $w'(t) \geq c > 0$.

Nonsmooth Filled Functions

It is well known that the constrained optimization problem can be formulated as a nonsmooth optimization problem by using the exact penalty function; see [3] or (3)–(5). With use of the methods of nonsmooth analysis, a nonsmooth unconstrained optimization problem was studied in [10], which involved a modified filled function as follows

$$P_F(x, x_1^*, r, \rho) = \ln\left(1 + \frac{1}{r + F(x)}\right)\exp\left(-\frac{\|x - x_1^*\|^2}{\rho^2}\right) , \quad (7)$$

where $F(x)$ is a weak semismooth objective function and x_1^* is a local minimizer of $F(x)$.

For a composite function $F(x)$ in the form

$$F(x) = f(x) + h(c(x)) ,$$

where $f(x)$ and $c(x) = (c_1(x), \ldots, c_m(x))^T$ are smooth functions and $h\colon R^m \to R$ is convex but nonsmooth [2], a kind of two-parameter filled function

$$P(x, r, A) = \psi(r + f(x))\exp(-A\|x - x_k^*\|^2)$$

was considered in [20], where the function $\psi(t)$ has properties such as:

(i) $\psi(t) > 0$ for $t \geq 0$.

(ii) $\psi(t)$ is monotonically decreasing for $t \geq 0$.

(iii) $\psi(t_1) - \psi(t_2) \leq c_2(t_2 - t_1)$ for $t_2 > t_1 \geq 0$, where $c_2 > 0$ is a constant.

In addition, for the single-parameter filled functions, we can also consider some general forms as follows:

$$U(x, A) = -\phi(f(x) - f(x_k^*))\exp(A\|x - x_k^*\|^2) ,$$

or

$$\tilde{U}(x, A) = -\phi(f(x) - f(x_k^*)) - A\|x - x_k^*\|^2 ,$$

where $A > 0$ is a parameter, and the function $\varphi(t)$ is required to satisfy certain conditions [20]:

(i) $\phi(0) = 0$, $\phi(t)$ is monotonically increasing for $t \geq 0$;

(ii) $c_1(t_2 - t_1) \leq \phi(t_2) - \phi(t_1) \leq c_2(t_2 - t_1)$ for $t_2 > t_1 \geq 0$, where $0 < c_1 \leq c_2$ are constants.

Note that even for a continuously differential unconstrained optimization problem, there may exist a nonsmooth filled function. For example, a two-parameter nonsmooth filled function

$$\begin{aligned} P(x, x_1^*, \rho, \mu) = f(x_1^*) &- \min[f(x_1^*), f(x)] \\ &- \rho\|x - x_1^*\|^2 \\ &+ \mu\{\max[0, f(x) - f(x_1^*)]\}^2 \end{aligned} \tag{8}$$

was introduced in [21], where $f(x)$ is coercive and Lipschitz continuous with a constant L in $\Re^n$.

Discrete Filled Functions

After the concept of the filled functions was introduced for continuous global optimization by Ge [5], some people tried to transform discrete global optimization problems into continuous ones and then to solve them by the continuous filled function methods [6,17,23].

For the discrete case, since the third property of a continuous filled function usually does not hold, such an extension is not trivial. Difficulties may also occur when continuous optimization methods are applied to deal with discrete optimization problems where the gradient vectors are unavailable or expensive to compute.

Discrete filled functions are related to the concept of the discrete neighborhood. The *discrete neighborhood* for a point $x \in \mathbb{Z}^n$ is usually defined by

$$\mathcal{N}(x) = \{x, x \pm e_i \mid i = 1, 2, \ldots, n\} ,$$

where e_i is the ith unit vector (i. e., the n-dimensional vector with the ith component equal to 1 and all other components equal to 0). On the basis of the local search approach and the two-parameter filled function defined by (1), Zhu [23] proposed an approximate algorithm for a class of nonlinear integer programming problems

$$\min_{x \in \Omega \cap \mathbb{Z}^n} f(x) ,$$

where Ω is a bounded closed box with all vertices integral. The algorithm is a direct method, which tries to improve a current discrete local minimal solution by minimizing an associated filled function. In [23], the author used two examples to illustrate the numerical performance of the algorithm proposed there.

In addition, based on the concept of 1/5-*neighborhood* of an integer point x such as

$$\mathcal{N}(x) = \left\{ y \in \Re^n \mid \|y - x\|_\infty \leq \frac{1}{5} \right\} ,$$

Ge and Huang [6] investigated unconstrained nonlinear integer programming, constrained nonlinear integer programming, and mixed nonlinear integer programming problems. For such cases, the authors tried to use a penalty function to transform a nonlinear integer programming problem into a global optimization problem, which can be solved by the filled function method if the objective function is twice continuously differentiable in $\Re^n$, and its gradient and Hessian matrix are bounded. In particular, when the constraints are equalities, all constrained functions are assumed to be twice continuously differentiable.

The unconstrained nonlinear integer programming model in [6] has the form:

$$\begin{aligned} &\text{Minimize } f(x) , \\ &\text{subject to } |x_i| \leq b_i, i = 1, 2, \ldots, n \\ &\qquad\qquad x \in \mathbb{Z}^n , \end{aligned} \tag{9}$$

where each b_i is an integer. Under certain conditions, if x^* is a global minimizer of a penalty function

$$\phi_1(x, k) = f(x) - k \sum_{i=1}^{n} \cos 2\pi x_i$$

in the box $\{x \mid |x_i| \leq b_i, i = 1, 2, \ldots, n\}$ and x^* is in a 1/5-neighborhood of an integer point $\bar{x}$, then $\bar{x}$ is a solution of problem (9).

For some integer $m < n$, if the second constraint in (9), $x \in \mathbb{Z}^n$, is replaced by $x_i \in \mathbb{Z}(i = m, m + 1, \ldots, n)$, then the corresponding problem is called the *mixed nonlinear integer programming problem*, for which a similar function

$$\phi_2(x, k) = f(x) - k \sum_{i=m}^{n} \cos 2\pi x_i$$

can be used as a penalty function.

Similarly, for a constrained nonlinear integer programming problem

$$\begin{aligned}&\text{Minimize } f(x)\,,\\&\text{subject to } c_i(x)=0,\, i=1,2,\,\dots\,,p\,,\\&\qquad\min\{0,c_i(x)\}=0,\, i=p+1,\,\dots\,,q\,,\\&\qquad|x_i|\le b_i,\, i=1,2,\,\dots\,,n\\&\qquad x\in\mathbb{Z}^n\,,\end{aligned}\tag{10}$$

some results can be derived by using the following penalty function:

$$\begin{aligned}\phi_3(x,r,k)=f(x)+r\sum_{i=1}^{p}c_i^2(x)\\+r\sum_{i=p+1}^{q}\left[\min\{0,c_i(x)\}\right]^2\\-k\sum_{i=1}^{n}\cos 2\pi x_i\,.\end{aligned}$$

The minimization of $\phi_3(x,r,k)$ can be dealt with by the filled function method proposed for constrained optimization problems [3].

For the discrete optimization problem

$$\min_{x\in X\subset\mathbb{Z}^n} f(x)\,,$$

where f is a Lipschitz function, X is a bounded and (strictly) pathwise connected domain, Ng et al. [17] modified the definition of continuous filled functions in order to allow them to be applied to discrete cases. Now we give a definition of a discrete filled function as follows:

> Given a discrete local minimizer x^* of a function $f\colon X\subset\mathbb{Z}^n\to R$, let B^* be the discrete basin of f at x^* over X. A function $F\colon X\to R$ is said to be a *discrete filled function* of f at x^* if it satisfies the following conditions:
>
> (a) x^* is a strict local maximizer of F over X;
>
> (b) F has no discrete local minimizers in B^* or in any discrete basin of f higher than B^*;
>
> (c) If f has a discrete basin B^{**} at x^{**} that is lower than B^*, then there is a discrete point $x'\in B^{**}$ that minimizes F on a discrete path $\{x^*,\,\dots\,,x',\,\dots\,,x^{**}\}$ in X.

On the basis of the two-parameter nonsmooth filled function defined by (8) at a local minimizer x_1^*, a two-phase algorithm was proposed to solve a discrete global optimization problem in [17]. In phase 1, a discrete steepest descent method was applied to find a local minimizer x_1^* of f over X, which was called the *local search*. Phase 2 searched for a minimum of the discrete filled function defined by (8) on a discrete path in X via some special search directions, which was called *global search*. The global search would identify a point x' in a discrete basin lower than the discrete basin B_1^* of f at x_1^*. The algorithm stopped when minimizing a discrete filled function did not yield a better solution than the current best local minimizer.

Summary

Many existing filled function methods require the assumption that the objective function has only a finite number of local minimizers. In addition, they also require that these local minima have different objective values. The assignment of single/two parameters in a filled function is a very important issue for ensuring the existence of a specific point for the filled function, by which a better local minimum of the original objective function can be found in a lower basin if it exists. Note that even for a local minimizer existing in a lower basin, how to find it is still a reduced optimization problem.

Furthermore, it is hard to find a general stopping criterion for the filled function methods, i. e., to check whether a feasible point obtained by any of the filled function methods is a global minimizer or not. All these drawbacks indicate that research on the filled function methods will be fascinating in the future. People may consider extensive approaches to solve global optimization problems, for example, by using modified functions which include some nonfilled functions [19], or by using locally filled functions which are integrated with techniques in cluster analysis [9,22].

References

1. Dixon LCW, Gomulka J, Hersom SE (1976) Reflections on global optimization. In: Dixon LCW Optimization in Action. Academic Press, New York, pp 398–435
2. Fletcher R (1981) Practical Methods of Optimization, vol 2. Wiley, New York

3. Ge RP (1987) The theory of filled function methods for finding global minimizers of nonlinearly constrained minimization problems. Presented at SIAM Conference on Numerical Optimization, Boulder, Colorado, 1984. See also J Comput Math 5(1):1–9
4. Ge RP (1989) A parallel global optimization algorithm for rational separable-fractorable functions. Appl Math Comput 32(1):61–72
5. Ge RP (1990) A filled function method for finding a global minimizer of a function of several variables. Presented at the Dundee Conference on Numerical Analysis, Dunded, Scotland, 1983. See also Math Programm 46:191–204
6. Ge RP, Huang CB (1989) A continuous approach to nonlinear integer programming. Appl Math Comput 34(1):39–60
7. Ge RP, Qin YF (1987) A class of filled functions for finding global minimizers of a function of several variables. J Optim Theory Appl 54(2):241–252
8. Ge RP, Qin YF (1990) The globally convexized filled functions for globally optimization. Appl Math Comput 35:131–158
9. Huang HX, Zhao Y (2007) A hybrid global optimization algorithm based on locally filled functions and cluster analysis. Int J Comput Sci Eng 3:194–202
10. Kong M, Zhuang JN (1996) A modified filled function method for finding a global minimizer of a non-smooth function of several variables (In Chinese). Num Math J Chinese Univ 18(2):165–174
11. Liu X (2001) Finding global minima with a computable filled function. J Global Optim 19:151–161
12. Liu X (2002) A computable filled function used for global minimization. Appl Math Comput 126(2–3):271–278
13. Liu X (2002) Several filled functions with mitigators. Appl Math Comput 133(2–3):375–387
14. Liu X (2004) Two new classes of filled functions. Appl Math Comput 149(2):577–588
15. Liu X (2004) The impelling function method applied to global optimization. Appl Math Comput 151(3):745–754
16. Lucidi S, Piccialli V (2002) New classes of global convexized filled functions for global optimization. J Global Optim 24:219–236
17. Ng CK, Zhang LS, Li D, Tian WW (2005) Discrete filled function method for discrete global optimization. Comput Optim Appl 31:87–115
18. Wu ZY, Li HWJ, Zhang LS, Yang XM (2006) A novel filled function method and quasi-filled function method for global optimization. Comput Optim Appl 34(2):249–272
19. Wu ZY, Zhang LS, Teo KL, Bai FS (2005) New modified function method for global optimization. J Optim Theory Appl 125(1):181–203
20. Xu Z, Huang HX, Pardalos PM, Xu CX (2001) Filled functions for unconstrained global optimization. J Global Optim 20:49–65
21. Zhang LS, Ng CK, Li D, Tian WW (2004) A new filled function method for global optimization. J Global Optim 28:17–43
22. Zhao Y (2006) Study of hybrid optimization methods based on locally filled functions. Master Thesis (In Chinese), Tsinghua University
23. Zhu WX (1998) An approximate algorithm for nonlinear integer programming. Appl Math Comput 93(2/3):183–193
24. Zhuang JN (1994) A generalized filled function method for finding the global minimizer of a function of several variables (In Chinese). Num Math J Chinese Univ 16(3):279–287

Global Optimization: Functional Forms

CHRYSANTHOS E. GOUNARIS,
CHRISTODOULOS A. FLOUDAS
Department of Chemical Engineering,
Princeton University, Princeton, USA

Article Outline

Keywords and Phrases

Global optimization; Convex underestimators; αBB; Functional forms

Introduction

Given the wide variety of different global optimization techniques, every time we have a new optimization problem we must select the best technique for solving this problem. This selection problem is made more complex by the fact that most techniques for solving global optimization problems have parameters that need to be adjusted to the problem or to the class of problems. For example, in gradient methods, one can select different step sizes.

When we have a single or few parameters to choose, it is possible to empirically try many values and come up with an (almost) optimal value. Thus, in such situations, we can identify an optimal version of the corresponding technique. In other approaches, such as methods like convex underestimators (described in detail in the next section), instead of selecting the value of single *number*-valued parameter, we have to select the auxiliary *function*. It is not practically possible to test all possible functions, so it is not easy to identify an optimal version of the corresponding technique [9].

This entry presents the work of Floudas and Kreinovich [9,10] on the functional forms of convex underestimators for twice continuously differentiable functions. They consider the problem of selecting the best auxiliary function within a given global optimization technique. Specifically, they showed that in many such selection situations, natural symmetry requirements enables one to either analytically solve the problem of finding the optimal auxiliary function, or at least reduce this problem to the easier-to-solve problem of finding a few parameters.

In particular, they showed that we can thus explain both the αBB method [1,2,6,16] and the generalized αBB recently proposed in [4,5]. A recent review article on these deterministic global optimization approaches can be found in [8].

Selecting Convex Underestimators: The αBB Method

It is well known that convex functions are computationally easier to minimize than non-convex ones (see [7]). This relative easiness is not only an empirical fact, it also has a theoretical justification (see [13,19]).

Because of this relative easiness, one of the approaches for minimization of a non-convex function $f(x) = f(x_1, \dots, x_n)$ (under certain constraints) over a box $[x^L, , x^U] = [x_1^L, x_1^U] \times \dots \times [x_n^L, x_n^U]$ is to first minimize its convex "underestimator", i. e., a convex function $L(x) \le f(x)$. Since $L(x)$ is an underestimator, the minimum of $L(x)$ is a lower bound for the minimum of $f(x)$. By selecting $L(x)$ as close to $f(x)$ as possible, we can get estimates for $\min f(x)$ which are as close to the actual minimum as possible.

The quality of approximation improves when the boxes become smaller. To get more accurate bounds on $\min f(x)$, we can bisect the box $[x^L, x^U]$ into sub-boxes within a regular branch-and-bound framework, and use the above technique to estimate $\min f(x)$ after considering the result of each node and utilizing fathoming of branches where appropriate.

A known efficient approach to designing a convex underestimator is the αBB global optimization algorithm [1,2,6,16], in which we select an underestimator $L(x) = f(x) + \Phi(x)$, where

$$\Phi(x) = -\sum_{i=1}^{n} \alpha_i \cdot (x_i - x_i^L) \cdot (x_i^U - x_i) \,. \tag{1}$$

Here, the parameters α_i are selected in such a way that the resulting function $L(x)$ is convex and still not too far away from the original objective function $f(x)$. For a thorough presentation of ways to select these parameters, see [1,2,3,11].

In many optimization problems, the αBB techniques are very efficient, but in some non-convex optimization problems, it is desirable to improve their performance. One way to do that is to provide a more general class of methods, with more parameters to tune. In the αBB techniques, for each coordinate x_i, we have a single parameter α_i affecting this coordinate. Changing α_i is equivalent to a linear re-scaling of x_i. Indeed, if we change the unit for measuring x_i to a new unit which is λ_i times smaller, then all the numerical values become λ_i times larger: $x_i \to y_i = g_i(x_i)$, where $g_i(x_i) = \lambda_i \cdot x_i$. In principle, we can have two different re-scalings:

- $x_i \to y_i = g_i(x_i) = \lambda_i \cdot x_i$ on the interval $[x^L{}_i, x_i]$, and
- $x_i \to z_i = h_i(x_i) = \mu_i \cdot x_i$ on the interval $[x_i, x^U{}_i]$.

If we substitute the new values $y_i = g_i(x_i)$ and $z_i = h_i(x_i)$ into the formula (1), then we get the following expression

$$\Phi(x) = -\sum_{i=1}^{n} \alpha_i \cdot \big(g_i(x_i) - g_i(x_i^L)\big) \cdot \big(h_i(x_i^U) - h_i(x_i)\big) \,. \tag{2}$$

For the above linear re-scalings, we get

$$\widetilde{\Phi}(x) = -\sum_{i=1}^{n} \widetilde{\alpha}_i \cdot (x_i - x_i^L) \cdot (x_i^U - x_i) \,,$$

where $\widetilde{\alpha}_i = \alpha_i \cdot \lambda_i \cdot \mu_i$.

From this viewpoint, a natural generalization is to replace *linear* re-scalings $g_i(x_i)$ and $h_i(x_i)$ with *non-linear* ones, that is, to consider convex underestimators of the type $L(x) = f(x) + \Phi(x)$, where $\Phi(x)$ is described by the formula (2) with non-linear functions $g_i(x_i)$ and $h_i(x_i)$. Now, instead of selecting a number α_i for each coordinate i, we have an additional freedom of choosing arbitrary non-linear functions $g_i(x_i)$ and $h_i(x_i)$. The question of which are the best choices is naturally posed. In [4,5], several different non-linear functions were tried, and it turned out that among the tested functions, the best results were achieved for the exponential functions $g_i(x_i) = \exp(\gamma_i \cdot x_i)$ and $h_i(x_i) = -\exp(-\gamma_i \cdot x_i)$. For these functions, the expression (2) can be somewhat simplified: indeed,

$$\begin{aligned}&\alpha_i \cdot \left(g_i(x_i) - g_i(x_i^L)\right) \cdot \left(h_i(x_i^U) - h_i(x_i)\right)\\ &\quad = \alpha_i \cdot (e^{\gamma_i \cdot x_i} - e^{\gamma_i \cdot x_i^L}) \cdot (-e^{-\gamma_i \cdot x_i^U} + e^{-\gamma_i \cdot x_i})\\ &\quad = \widetilde{\alpha}_i \cdot (1 - e^{\gamma_i \cdot (x_i - x_i^L)}) \cdot (1 - e^{\gamma_i \cdot (x_i^U - x_i)}) ,\end{aligned}$$

where $\widetilde{\alpha}_i \overset{\text{def}}{=} \alpha_i \cdot e^{\gamma_i \cdot (x_i^U - x_i^L)}$.

Two related questions naturally arise and are addressed in the work of Floudas and Kreinovich [9,10]:

- first, a *practical* question: an empirical choice is made by using only finitely many functions; is this choice indeed the best – or there are other, even better functions $g_i(x_i)$ and $h_i(x_i)$, which we did not discover because we did not try them?
- second, a *theoretical* question: how can we explain the above empirical fact?

Shift Invariance

The starting point for measuring each coordinate x_i is often a matter of arbitrary choice. If a selection of the functions $g_i(x_i)$ and $h_i(x_i)$ is "optimal" (in some intuitive sense), then the results of using these optimal functions should not change if we simply change the starting point for measuring x_i, that is, replace each value x_i with a new value $x_i + s$, where s is the shift in the starting point. Indeed, otherwise, if the "quality" of the resulting convex underestimators changes with shift, we could apply a shift and get better functions $g_i(x_i)$ and $h_i(x_i)$ – which contradicts the assumption that the selection of $g_i(x_i)$ and $h_i(x_i)$ is already optimal.

The "optimal" choices $g_i(x_i)$ and $g_i(x_i)$ can be determined from the requirement that each component $\alpha_i \cdot (g_i(x_i) - g_i(x_i^L)) \cdot (h_i(x_i^U) - h_i(x_i))$ in the sum (2) be invariant under the corresponding shift, that is, that they satisfy the following definition.

Definition 1 A pair of smooth functions $(g(x), h(x))$ from real numbers to real numbers is *shift-invariant* if for every s and α, there exists $\widetilde{\alpha}(\alpha, s)$ such that for every x^L, x, and x^U, we have

$$\begin{aligned}&\alpha \cdot \left(g(x) - g(x^L)\right) \cdot \left(h(x^U) - h(x)\right)\\ &\quad = \widetilde{\alpha}(\alpha, s) \cdot \left(g(x + s) - g(x^L + s)\right)\\ &\qquad \cdot \left(h(x^U + s) - h(x + s)\right) .\end{aligned} \tag{3}$$

At first glance, shift invariance is a reasonable but weak property. It turns out, however, that this seemingly weak property actually almost uniquely determines the optimal selection of exponential functions. Proposition 1 applies.

Proposition 1 *If a pair of functions $(g(x), h(x))$ is shift-invariant, then this pair is either exponential or linear, i. e., each of the functions $g(x)$ and $h(x)$ has the form $g(x) = A + C \cdot \exp(\gamma \cdot x)$ or $g(x) = A + k \cdot x$.*

For a proof, see [9] or [10].

Sign Invariance

In addition to shift, another natural symmetry is changing the sign. If we require that the expression (2) remain invariant under a replacement of x by $-x$, then we get the relation between $g(x)$ and $h(x)$: $h(x) = -g(-x)$. So, if a pair $(g(x), h(x))$ is shift-invariant and sign-invariant, then:

- either $g(x) = \exp(\gamma \cdot x)$ and $h(x) = -\exp(-\gamma \cdot x)$,
- or $g(x) = h(x) = x$.

In other words, the optimal generalized αBB scheme is either the original αBB [1,2,6,16], or the scheme with exponential functions described in [4,5].

Scale Invariance

Sign-invariance can be perceived as a special case of scale-invariance. Scale-invariance addresses a change in the unit for measuring x, that is, transformations $x \to \lambda \cdot x$.

We have already shown that there are only two shift-invariant solutions: exponential and linear functions. Out of these two solutions, only the linear solution – corresponding to the original αBB – is scale-invariant. Thus, if we also require scale-invariance, we restrict ourselves only to original techniques and miss the (often better) exponential generalizations.

Although imposing both shift- and scale-invariance leads to restrictions, one could still choose to employ only the latter, formally expressed as follows:

Definition 2 A pair of smooth functions $(g(x), h(x))$ from real numbers to real numbers is *scale-invariant* if for every λ and α, there exists $\widetilde{\alpha}(\alpha, \lambda)$ such that for every x^L, x, and x^U, we have

$$\begin{aligned}\alpha \cdot \big(g(x) - g(x^L)\big) \cdot \big(h(x^U) - h(x)\big) \\ = \widetilde{\alpha}(\alpha, \lambda) \cdot \big(g(\lambda \cdot x) - g(\lambda \cdot x^L)\big) \cdot \big(h(\lambda \cdot x^U) \\ - h(\lambda \cdot x)\big)\end{aligned} \quad (4)$$

The following proposition applies. For a proof, see [9].

Proposition 2 *If a pair of functions $(g(x), h(x))$ is scale-invariant, then this pair is either exponential or linear, i. e., each of the functions $g(x)$ and $h(x)$ has the form $g(x) = A \cdot x^\gamma$ or $g(x) = A + k \cdot \ln(x)$.*

From the theoretical viewpoint, these functions may look as good as the exponential functions coming from shift invariance, but in practice, they do not work so well. The problem with these solutions is that they do not preserve smoothness. Both linear and exponential functions which come from shift-invariance are infinitely differentiable for all x and hence, adding the corresponding term $\Phi(x)$ will not decrease the smoothness level of the objective function. In contrast, the functions $g(x) = x^\gamma$ which come from scale invariance are not infinitely differentiable at $x = 0$ or when γ is not integer. So, if we use scale invariance to select a convex underestimator, we end up with a new parameter γ which only attains integer-valued values and is, thus, less flexible than the continuous-valued parameters coming from scale-invariance.

Generalization of Shift Invariance

Instead of the expression (2), we can consider an even more general expression

$$\Phi(x) = -\sum_{i=1}^{n} \alpha_i \cdot a_i(a,\, x^L) \cdot b_i(x_i,\, x_i^U)\,. \quad (5)$$

What can be concluded from shift-invariance in this more general case?

Definition 3 A pair of smooth functions $(a(x, x^L), b(x, x^U))$ from real numbers to real numbers is *shift-invariant* if for every s and α, there exists $\widetilde{\alpha}(\alpha, s)$ such that for every x^L, x, and x^U, we have

$$\begin{aligned}&\alpha \cdot a(x,\, x^L) \cdot b(x,\, x^U) \\ &\quad = \widetilde{\alpha}(\alpha,\, s) \cdot a(x+s,\, x^L+s) \cdot b(x+s,\, x^U+s)\,.\end{aligned} \quad (6)$$

Regarding such functions, Floudas and Kreinovich [9] proved the following proposition:

Proposition 3 *If a pair of functions $(a(x, x^L), b(x, x^U))$ is shift-invariant, then*

$$a(x,\, x^L) \cdot b(x,\, x^U) = A(x - x^L) \cdot B(x^U - x) \cdot e^{\gamma \cdot x^L}$$

for some functions $A(x)$ and $B(x)$ and for some real number γ.

Comment. If we additionally require that the expression $a(x,\, x^L) \cdot b(x,\, x^U)$ be invariant under $x \to -x$, then we conclude that $B(x) = A(x)$.

Another shift-invariance result comes from the following observation. Both the αBB expression

$$-(x - x^L) \cdot (x^U - x)$$

and the generalized expression

$$-(1 - e^{\gamma \cdot (x - x^L)}) \cdot (1 - e^{\gamma \cdot (x^U - x)})$$

have the form $a(x - x^L) \cdot a(x^U - x)$ with $a(0) = 0$. The differences $x - x^L$ and $x^U - x$ come from the fact that we want these expressions to be shift-invariant. The product form makes sense, since we want the product to be 0 on each border $x = x^L$ and $x = x^U$ of the corresponding interval $[x^L, x^U]$.

On the other hand, it is well known that optimizing a product is more difficult than optimizing a sum; since we will be minimizing the expression $f(x) + \Phi(x)$, it is therefore desirable to be able to reformulate it in terms of the easier-to-minimize sum, e. g., as $b(x - x^L) + b(x^U - x) + c(x^U - x^L)$ for some functions b and c (for minimization purposes, c does not depend on x and is thus a constant). It is worth mentioning that both the αBB expression and its exponential generalization allow such representation. Note that:

$$\begin{aligned}&-(x-x^L)\cdot(x^U-x)\\&\quad=\frac{1}{2}\cdot(x-x^L)^2+\frac{1}{2}\cdot(x^U-x)^2-\frac{1}{2}\cdot(x^U-x^L)^2;\end{aligned}$$

and

$$\begin{aligned}&-(1-e^{\gamma\cdot(x-x^L)})\cdot(1-e^{\gamma\cdot(x^U-x)})\\&\quad=-1+e^{\gamma\cdot(x-x^L)}+e^{\gamma\cdot(x^U-x)}-e^{\gamma\cdot(x^U-x^L)}.\end{aligned}$$

Interestingly, the above two expressions are the only ones which have this easiness-to-compute property:

Definition 4 We say that a smooth function $a(x)$ from real numbers to real numbers describes an *easy-to-compute* underestimator if $a(0) = 0$, $a'(0) \neq 0$, and there exist smooth functions $b(x)$ and $c(x)$ such that for every x, x^L, and x^U, we have

$$a(x-x^L)\cdot a(x^U-x)=b(x-x^L)+b(x^U-x)+c(x^U-x^L). \tag{7}$$

The condition $a'(0) \neq 0$ comes from the fact that otherwise, for small $\Delta x \overset{\text{def}}{=} x - x^L$ and $x^U - x$, each value $a(x-x^L)$ will be quadratic in $x-x^L$, the resulting product will be fourth order, and we will not be able to compensate for quadratic non-convex terms in the original objective function $f(x)$ – which defeats the purpose of using $f(x) + \Phi(x)$ as a *convex* underestimator.

Proposition 4 *The only functions which describe easy-to-compute underestimators are $a(x) = k\cdot x$ and $a(x) = k\cdot(1-e^{\gamma\cdot x})$.*

This is another shift-invariance related result that is also proven in [9]. It selects linear and exponential functions as "the best" in some reasonable sense. Floudas and Kreinovich [9] proved that any "natural" shift-invariant optimality criterion on the set of all possible underestimator methods selects either a linear or an exponential function.

Final Remarks

The work of Floudas and Kreinovich [9,10] has a much further-reaching effect than on the case of αBB-based convex underestimation mainly discussed here. A symmetry-based approach leads to optimal techniques also in the cases of optimal bisection (for selecting box-splitting strategies) and optimal selection of penalty and barrier functions. Other empirically optimal techniques can also be explained by symmetry-based arguments. These include the "epsilon-inflation" technique [15,18], results in simulated annealing and genetic algorithms [17], as well as optimal selection of probabilities in swarm optimization [12,14].

References

1. Adjiman CS, Androulakis IP, Floudas CA (1998) A Global Optimization Method, αBB, for General Twice-Differentiable Constrained NLPs II. Implementation and Computational Results. Comput Chem Eng 22:1159–1179
2. Adjiman CS, Dallwig S, Floudas CA, and Neumaier A (1998) A Global Optimization Method, αBB, for General Twice-Differentiable Constrained NLPs I. Theoretical Advances. Comput Chem Eng 22:1137–1158
3. Adjiman CS, Floudas CA (1996) Rigorous Convex Underestimators for General Twice-Differentiable Problems. J Global Optim 9:23–40
4. Akrotirianakis IG, Floudas CA (2004) A New Class of Improved Convex Underestimators for Twice Continuously Differentiable Constrained NLPs. J Global Optim 30:367–390
5. Akrotirianakis IG, Floudas CA (2004) Computational Experience with a New Class of Convex Underestimators : Box-Constrained NLP Problems. J Global Optim 29:249–264
6. Androulakis IP, Maranas CD, Floudas CA (1995) αBB: A Global Optimization Method for General Constrained Nonconvex Problems. J Global Optim 7:337–363
7. Floudas CA (2000) Deterministic Global Optimization: Theory, Algorithms and Applications. Kluwer, Dordrecht
8. Floudas CA, Akrotirianakis IG, Caratzoulas S, Meyer CA, Kallrath J (2005) Global Optimization in the 21st Century: Advances and Challenges. Comput Chem Eng 29:1185–1202
9. Floudas CA, Kreinovich V (2006) Towards Optimal Techniques for Solving Global Optimization Problems: Symmetry-Based Approach. In: Torn A, Zilinskas J (eds) Models and Algorithms for Global Optimization. Springer, Berlin, pp 21–42
10. Floudas CA, Kreinovich V (2007) On the Functional Form of Convex Understimators for Twice Continuously Differentiable Functions. Optim Lett 1:187–192
11. Hertz D, Adjiman CS, Floudas CA (1999) Two results on bounding the roots of interval polynomials. Comput Chem Eng 23:1333–1339
12. Iourinski D, Starks SA, Kreinovich V, Smith SF (2002) Swarm Intelligence: Theoretical Proof that Empirical Techniques are Optimal. In: Proceedings of the 2002 World Automation Congress WAC 2002, Orlando, FL, pp 107–112
13. Kearfott RB, Kreinovich V (2005) Beyond Convex? Global Optimization is Feasible only for Convex Objective Functions: A Theorem. J Global Optim 33:617–624

14. Kennedy J, Eberhart R, Shi Y (2001) Swarm Intelligence. Morgan Kaufmann, New York
15. Kreinovich V, Starks SA, Meyer G (1997) On a Theoretical Justification of the Choice of Epsilon-Inflation in PASCAL-XSC. Reliable Comput 3:437–452
16. Maranas CD, Floudas CA (1994) Global Minimum Potential Energy Conformations of Small Molecules. J Global Optim 4:135–170
17. Nguyen HT, Kreinovich V (1997) Applications of Continuous Mathematics to Computer Science. Kluwer, Dordrecht
18. Rump SM (1998) A Note on Epsilon-Inflation. Reliable Comput 4:371–375
19. Vavasis SA (1991) Nonlinear Optimization: Complexity Issues. Oxford University Press, Oxford

Global Optimization: g-αBB Approach

CHRYSANTHOS E. GOUNARIS,
CHRISTODOULOS A. FLOUDAS
Department of Chemical Engineering,
Princeton University, Princeton, USA

MSC2000: 49M37, 65K10, 90C26, 90C30, 46N10, 47N10

Article Outline

Keywords and Phrases

Convex underestimators; αBB; Global optimization

Introduction

Various deterministic global optimization algorithms that utilize a branch and bound framework make use of convex underestimators of the functions under consideration. For a recent review of such approaches, see [7]. For arbitrarily nonconvex C^2-continuous functions $f(x)$, defined in domain $X = [x^L, x^U]$, the αBB underestimator [1,2,3,6,10] is typically used. This is constructed by adding to the original function the following separable relaxation term, $\phi(x;\alpha)$:

$$\phi(x;\alpha) = -\sum_{i=1}^{n} \alpha_i (x_i - x_i^L)(x_i^U - x_i)\,, \tag{1}$$

where $\alpha_i \geq 0$, $i = 1, 2, \ldots, n$. The resulting underestimator of $f(x)$ would thus be

$$L_{\alpha BB}(x;\alpha) = f(x) + \phi(x;\alpha)\,. \tag{2}$$

Since the relaxation term is separable, the following relationship exists among the Hessian matrices of $L_{\alpha BB}(x;\phi)$, $f(x)$ and $\phi(x;\alpha)$:

$$\nabla^2 L_{\alpha BB}(x;\alpha) = \nabla^2 f(x) + 2A\,, \tag{3}$$

where $A = \nabla^2 \phi(x;\alpha) = \operatorname{diag}\{\alpha_1, \alpha_2, \ldots, \alpha_n\}$. The addition of the relaxation term corresponds to a diagonal shift of the Hessian matrix. Therefore, if we select large enough values for the α_i parameters, the nonconvexities in the original function can be overpowered and the resulting underestimator $L_{\alpha BB}(x;\alpha)$ becomes convex.

A number of rigorous methods have been devised in order to select appropriate values for these parameters [1,2,3,8]. Extensive computational testing of the algorithm [3] showed that the most efficient of those methods is the one based on the *scaled Gerschgorin* theorem. According to this method, it suffices to select

$$\alpha_i = \max\left[0, -\frac{1}{2}\left(\underline{h_{ii}} - \sum_{\substack{j=1\\ j\neq i}}^{n} \max\left\{|\underline{h_{ij}}|, |\overline{h_{ij}}|\right\} \frac{(x_j^U - x_j^L)}{(x_i^U - x_i^L)}\right)\right]\,, \tag{4}$$

where $\underline{h_{vu}}$ and $\overline{h_{vu}}$ are lower and upper bounds of $\partial^2 f/\partial x_v x_u$ that can be calculated by interval analysis.

The g-αBB approach was developed in [4,5] and offers an alternative convex underestimation functional form than the one originally proposed in the αBB theory. The new relaxation scheme suggests subtraction of a similar separable term that is of exponential, rather than quadratic, nature.

The New Relaxation Term

In this section, we present the new relaxation function. It shares most of the characteristics of the relaxation function, $\phi(x;\alpha)$, used in the original αBB underestimator described above. However, it possesses novel additional properties that enable it to derive convex underestimators that are tighter to the original function. Thus, the new underestimators can help expedite the branch and bound process of the overall global optimization framework.

The new relaxation function is defined as follows:

$$\Phi(x;\gamma) = -\sum_{i=1}^{n}(1-e^{\gamma_i(x_i-x_i^L)})(1-e^{\gamma_i(x_i^U-x_i)}) , \quad (5)$$

where $\gamma = (\gamma_1, \gamma_2, \ldots, \gamma_n)^{\mathrm{T}}$ is a vector of nonnegative parameters. As will be explained later, these parameters play a similar role as the α_i's in the original αBB method.

The gradient of $\Phi(x;\gamma)$ is

$$\nabla\Phi(x;\gamma) = -\begin{pmatrix} -\gamma_1 e^{\gamma_1(x_1-x_1^L)} + \gamma_1 e^{\gamma_1(x_1^U-x_1)} \\ -\gamma_2 e^{\gamma_2(x_2-x_2^L)} + \gamma_2 e^{\gamma_2(x_2^U-x_2)} \\ \vdots \\ -\gamma_n e^{\gamma_n(x_n-x_n^L)} + \gamma_n e^{\gamma_n(x_n^U-x_n)} \end{pmatrix}$$

and its Hessian is defined by the diagonal matrix

$$\nabla^2\Phi(x;\gamma) = \mathrm{diag}\Big\{\gamma_i^2 e^{\gamma_i(x_i-x_i^L)} + \gamma_i^2 e^{\gamma_i(x_i^U-x_i)} : i = 1, 2, \ldots, n\Big\} .$$

Note that $\nabla^2\Phi(x;\gamma)$ is a function of x as opposed to the Hessian matrix of $\phi(x;\alpha)$, used in αBB, which is constant throughout the domain X.

The new relaxation function $\Phi(x;\gamma)$ has the following important properties:

P1: $\Phi(x;\gamma) \leq 0$, for all $x \in [x^L, x^U]$.
P2: $\Phi(x;\gamma) = 0$ at the corner points of the interval $[x^L, x^U]$.
P3: $\Phi(x;\gamma)$ is a convex function.
P4: $\Phi(x;\gamma)$ achieves its minimum at the middle point, x^{mid}, of X and its maximum at the corner points.
P5: The diagonal element of $\nabla^2\Phi(x;\gamma)$ is a convex function and achieves its minimum at the middle point and its maximum at the endpoints of $[x_i^L, x_i^U]$.

The New Underestimating Function

The new underestimating function, $L_1(x;\gamma)$, is formed by adding $\Phi(x;\gamma)$ to the nonconvex function $f(x)$, that is,

$$L_1(x;\gamma) = f(x) + \Phi(x;\gamma) . \quad (6)$$

The Hessian of L_1 is

$$\nabla^2 L_1(x;\gamma) = \nabla^2 f(x) + \nabla^2\Phi(x;\gamma) .$$

The underestimator $L_1(x;\gamma)$ has the following important properties:

U1: $L_1(x;\gamma)$ is an underestimator of $f(x)$.
U2: $L_1(x;\gamma)$ matches $f(x)$ at all corner points of X.
U3: The maximum separation distance between the nonconvex function $f(x)$ and its underestimator $L_1(x;\gamma)$ is bounded.
U4: The underestimators constructed over supersets of the current set are always less tight than the underestimator constructed over the current box constraints.

Since the function $\Phi(x;\gamma)$ is convex for every $x \in X$ and $\gamma \geq 0$, all nonconvexities in the original function $f(x)$ can be eliminated, provided that the parameters γ_i have the appropriate values. The selection of these values is presented in the next section.

Selection of Appropriate Parameter Values

The initial values for the γ_i parameters are selected by solving the following system of nonlinear equations:

$$\ell_i + \gamma_i^2 + \gamma_i^2 e^{\gamma(x_i^U-x_i^L)} = 0, \quad i = 1, 2 \ldots, n , \quad (7)$$

where $\ell_i \leq 0$, $i = 1, 2, \ldots, n$. The parameters ℓ_i convey second-order characteristics of the original nonconvex function into the construction process of the underestimator. Candidate values for these parameters can be selected as follows:

$$\ell_i = -2\alpha_i , \quad i = 1, 2 \ldots, n , \quad (8)$$

where $\alpha_i \geq 0$, $i = 1, 2, \ldots, n$ are the parameters that correspond to the original αBB method, as given by (4). Akrotirianakis and Floudas [4] proved that such a selection for the γ_i parameters always results in an underestimator that is tighter than the one resulting from the original method, i. e., (2). However, this new underestimator is not necessarily convex. Furthermore, they proved that there always exists some selection of γ_i parameters that results in a convex underestimator.

Therefore, they developed a systematic procedure that determines values for all parameters γ_i that not only guarantee the convexity of the underestimating function $L_1(x;\gamma)$ but also ensure that $L_1(x;\gamma)$ is at least as tight as the underestimating function $L_{\alpha BB}(x;\alpha)$. This procedure is an iterative scheme that is based on interval analysis and consecutive partitions of the domain X. Before we present the scheme, let us present two additional results from [4] that are relevant:

Theorem 1 *Let $\underline{\gamma} = (\underline{\gamma}_1, \underline{\gamma}_2, \dots, \underline{\gamma}_n)^T$ be the solution of system (7), with ℓ_i defined by (8). Then, the two underestimators $L_1(x;\underline{\gamma})$ and $L_{\alpha BB}(x;\underline{\alpha})$, where*

$$\underline{\alpha} = \left(\frac{4(1 - e^{0.5\underline{\gamma}_1(x_1^U - x_1^L)})^2}{(x_1^U - x_1^L)^2}, \dots, \frac{4(1 - e^{0.5\underline{\gamma}_n(x_n^U - x_n^L)})^2}{(x_n^U - x_n^L)^2} \right)^T , \quad (9)$$

have the same maximum separation distance from $f(x)$.

Theorem 2 *Let $\overline{\alpha} = (\overline{\alpha}_1, \overline{\alpha}_2, \dots, \overline{\alpha}_n)^T$ be the values of the α parameters as computed by (4). Then, the two underestimators $L_1(x;\overline{\gamma})$ and $L_{\alpha BB}(x;\overline{\alpha})$, where*

$$\overline{\gamma} = \left(\frac{2\log(1 + \sqrt{\overline{\alpha}_1}(x_1^U - x_1^L)/2)}{x_1^U - x_1^L}, \dots, \frac{2\log(1 + \sqrt{\overline{\alpha}_n}(x_n^U - x_n^L)/2)}{x_n^U - x_n^L} \right)^T , \quad (10)$$

have the same maximum separation distance from $f(x)$.

The main result of the above two theorems is that for any $\gamma \in [\underline{\gamma}, \overline{\gamma}]$ there exists an $\alpha \in [\underline{\alpha}, \overline{\alpha}]$, such that the underestimators $L_1(x;\gamma)$ and $L_{\alpha BB}(x;\alpha)$ have the same maximum separation distance from the nonconvex function $f(x)$. From all these pairs of underestimators, the only one that is known to be convex a priori is $L_{\alpha BB}(x;\overline{\alpha})$, since this is the one resulting from the classical αBB method. However, as will be apparent from the examples presented later, the underestimators $L_{\alpha BB}(x;\alpha)$ and $L_1(x;\gamma)$ are convex within a large portion of the intervals $[\underline{\alpha}, \overline{\alpha}]$ and $[\underline{\gamma}, \overline{\gamma}]$, respectively. On the basis of the above observations, it is natural to search for a vector γ in the interval $[\underline{\gamma}, \overline{\gamma}]$ or for a vector α in the interval $[\underline{\alpha}, \overline{\alpha}]$, so that at least one of the underestimators $L_1(x;\gamma)$ and $L_{\alpha BB}(x;\alpha)$ is convex.

The algorithm described below was developed in [4] for the appropriate selection of values for the γ parameters, so that the corresponding underestimator is both a convex function and at least as tight as the underestimator used by the classical αBB method. It searches for a vector $\gamma \in [\underline{\gamma}, \overline{\gamma}]$ so that the corresponding $\alpha \in [\underline{\alpha}, \overline{\alpha}]$ produces an underestimating function $L_{\alpha BB}(x;\alpha)$ that is convex. The search starts by setting $\gamma = \underline{\gamma}$ and $\alpha = \underline{\alpha}$ and then checking whether $L_{\alpha BB}(x;\underline{\alpha})$ is convex. This is done by using the scaled Gerschgorin method to determine lower bounds on the eigenvalues of the Hessian matrix $\nabla^2 L_{\alpha BB}(x;\underline{\alpha})$. For those lower bounds that are negative, the intervals of the corresponding variables are bisected, thereby generating a number of subdomains that are stored in a list, denoted by Λ_1. Then, the algorithm checks whether $\nabla^2 L_{\alpha BB}(x;\underline{\alpha})$ is positive semidefinite in each of those subdomains using again the scaled Gerschgorin method. If the size of the list, Λ_1, exceeds a certain number of nodes, then $\nabla^2 L_{\alpha BB}(x;\underline{\alpha})$ is most likely *not* positive semidefinite. The values of all γ_i's have to then be increased by a prespecified positive quantity, $\eta > 0$, and the corresponding values of the new α_i's are calculated. The algorithm now tries to verify whether $\nabla^2 L_{\alpha BB}(x;\alpha)$, with the new increased α parameters, is positive semidefinite. It continues in this manner until the list Λ_1 becomes empty. In that case, the corresponding α values make the Hessian matrix, $\nabla^2 L_{\alpha BB}(x;\alpha)$, positive semidefinite for all $x \in X$ and consequently $L_{\alpha BB}(x;\alpha)$ is a convex underestimator. The main reason for using the underestimator $L_{\alpha BB}(x;\alpha)$ instead of the underestimator $L_1(x;\gamma)$ is that it is easier to verify the positive definiteness of the matrix $\nabla^2 L_{\alpha BB}(x;\alpha)$ than that of the matrix $\nabla^2 L_1(x;\gamma)$. For more details see Alg. 1

Termination of the above algorithm is guaranteed by the fact that $L_{\alpha BB}(x;\overline{\alpha})$ is known, a priori, to be convex underestimator.

Computational Results

Because an iterative procedure is needed to determine appropriate values for the γ_i parameters, the construction of the new underestimators requires more computational effort than that required for the classical αBB method. However, within a global optimization framework, actual computational savings may be realized since the tighter underestimators produced by the

Algorithm:

```
Step 1 (initialization): Set $K = 1, J = 1, J_{\max} = 2^n + 1\ \eta = 1.1\ X_J = X\ \Lambda_1 = \{X_J\}$ and $\gamma_{i,K} = \underline{\gamma}_i$.
Step 2: For all$i = 1, 2 \cdots, n$, use (9) to calculate the $\alpha_{i,K}$ that correspond to $\gamma_{i,K}$, and form the underestimator
    $L_{\alpha BB}(x; \alpha_K)$.
Step 3: If the maximum separation distance of $L_{\alpha BB}(x; \alpha_K)$ from $f(x)$ is less than the maximum separation
    distance of $L_{\alpha BB}(x; \overline{\alpha})$ from $f(x)$ then go to step 4.
    Otherwise, adopt as an underestimator the classical $\alpha BB$ underestimator, $L_{\alpha BB}(x; \overline{\alpha})$, and stop.
Step 4: Check whether $L_{\alpha BB}(x; \alpha_K)$ is convex:
    Repeat
        Step 4.1: Remove the last element from the list $\Lambda_1$ of unexplored subdomains. Let us name that subdomain
            $X_{last}$.
        Step 4.2: Form the interval Hessian $[\nabla^2 L_{\alpha BB}(x; \alpha_K)]$ with $x \in X_{last}$.
        Step 4.3: Use (4) and (8) to find lower bounds on each eigenvalue of the interval Hessian
            $[\nabla^2 L_{\alpha BB}(x; \alpha_K)]$ in $X_{last}$.
        Step 4.4: Form the set $I_- = \{i : \ell_i < 0\}$.
        Step 4.5: If $I_- \neq \emptyset$, bisect all intervals $[x^L_{i,last}, x^U_{i,last}]$ with $i \in I_-$, and add them at the end of the list $\Lambda_1$.
        Step 4.6: Set $J = J + 2^{|I-|} - 1$, where $|I_-|$ represents the cardinality of the set $I_-$ (i. e., a total of $2^{|I-|}$ new
            subdomains have been generated and added to the list and one node has been removed).
    Until ($\Lambda_1 = \emptyset$ or $J = J_{\max}$).
Step 5: If $\Lambda_1 = \emptyset$ then stop. The Hessian $\nabla^2 L_{\alpha BB}(x; \alpha_K)$ is positive semidefinite for all $x \in X$ and $L_{\alpha BB}(x; \alpha_K)$
    is a convex underestimator. Also the underestimator $L_{\alpha BB}(x; \alpha_K)$ is tighter than the underestimator
    $L_{\alpha BB}(x; \overline{\alpha})$ obtained by the classical $\alpha BB$ method.
    Otherwise, increase the values of all $\gamma_{i,K}$, $i = 1, 2, \ldots, n$ by setting $\gamma_{i,K+1} = \eta\gamma_{i,K}$. Set $K = K + 1$ and go to
    step 2.
```

Global Optimization: g-αBB Approach, Algorithm 1

new method could expedite the branch and bound process through faster fathoming and visits to fewer tree nodes.

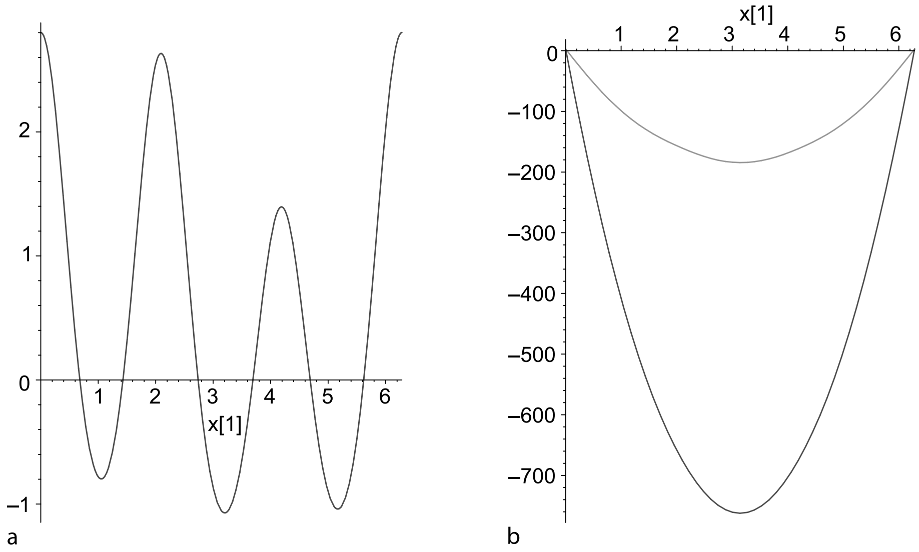
A detailed computational comparison between the new underestimators and the ones used by the classical αBB method was performed by Akrotirianakis and Floudas [5]. They concluded that the new underestimators usually perform better than the classical αBB method, in terms of both the overall CPU time and the number of nodes generated by the enumeration tree. It was also observed that the new underestimators perform better when the problem involves many arbitrarily nonconvex terms in the objective or constraints.

In the same study, Akrotirianakis and Floudas [5] also presented a hybrid optimization framework where underestimators $L_1(x; \underline{\gamma})$ were used to construct the relaxation in every node of the branch and bound tree. A stochastic random-linkage algorithm [9] was then employed to solve these relaxations and the method exhibited improved computational efficiency. Interestingly enough, the method located the actual global optimum in all case studies, despite the lack of theoretical guarantees owing to the fact that the underestimators $L_1(x; \underline{\gamma})$ are not necessarily convex.

As an illustration, we present here two examples from [5]:

Example 1

This example involves a nonconvex function that describes the molecular conformation of pseudoethane. It is taken from [11], where the global minimum potential energy conformation of small molecules is studied. The Lennard-Jones potential is expressed in terms of a sim-



Global Optimization: g-αBB Approach, Figure 1
Function $f_1(x)$ and comparison of underestimators $L_{\alpha BB}(x;\overline{\alpha})$ and $L_{\alpha BB}(x;\alpha)$

ple dihedral angle. The potential energy of the molecule is given by the following function:

$$\begin{aligned} f_1(x) = {} & \frac{588600}{(3r_0^2{-}4cos(\theta)r_0^2{-}2(sin^2(\theta)cos(x{-}\frac{2\pi}{3}){-}cos^2(\theta))r_0^2)^6} \\ & - \frac{1079.1}{3r_0^2{-}4cos(\theta)r_0^2{-}2(sin^2(\theta)cos(x{-}\frac{2\pi}{3}){-}cos^2(\theta))r_0^2)^3} \\ & + \frac{600800}{(3r_0^2{-}4cos(\theta)r_0^2{-}2(sin^2(\theta)cos(x){-}cos^2(\theta))r_0^2)^6} \\ & - \frac{1071.5}{(3r_0^2{-}4cos(\theta)r_0^2{-}2(sin^2(\theta)cos(x){-}cos^2(\theta))r_0^2)^3} \\ & + \frac{481300}{(3r_0^2{-}4cos(\theta)r_0^2{-}2(sin^2(\theta{+}\frac{2\pi}{3})cos(x){-}cos^2(\theta))r_0^2)^6} \\ & - \frac{1064.6}{(3r_0^2{-}4cos(\theta)r_0^2{-}2(sin^2(\theta{+}\frac{2\pi}{3})cos(x){-}cos^2(\theta))r_0^2)^3} , \end{aligned}$$

where r_0 is the covalent bond length ($r_0 = 1.54A$), θ is the covalent bond angle ($\theta = 109.5^o$) and x is the dihedral angle ($x \in X = [0, 2\pi]$). Figure 1 depicts the graph of $f_1(x)$.

The value of the α parameter computed by the classical αBB method using (4) is $\overline{\alpha} = 77.124$ and the corresponding value for the γ parameter, obtained by (10), is $\overline{\gamma} = 1.0673$. Also, by solving (7) for γ we obtain $\underline{\gamma} = 0.8521$ and the corresponding value for the α parameter, obtained by (9), is $\underline{\alpha} = 18.579$. The iterative algorithm checks whether there exist values of $\gamma \in [\underline{\gamma}, \overline{\gamma}]$ and $\alpha \in [\underline{\alpha}, \overline{\alpha}]$ such that the underestimator $L_{\alpha BB}(x;\alpha)$ is convex. After 16 iterations it concludes that if $\alpha = \underline{\alpha}$, then $L_{\alpha BB}(x;\alpha)$ is a convex underestimator of $f_1(x)$. Furthermore, if $\gamma = \underline{\gamma}$, then $L_1(x;\gamma)$ is also a convex underestimator of $f_1(x)$. Note that the values of γ and α did not have to increase at all.

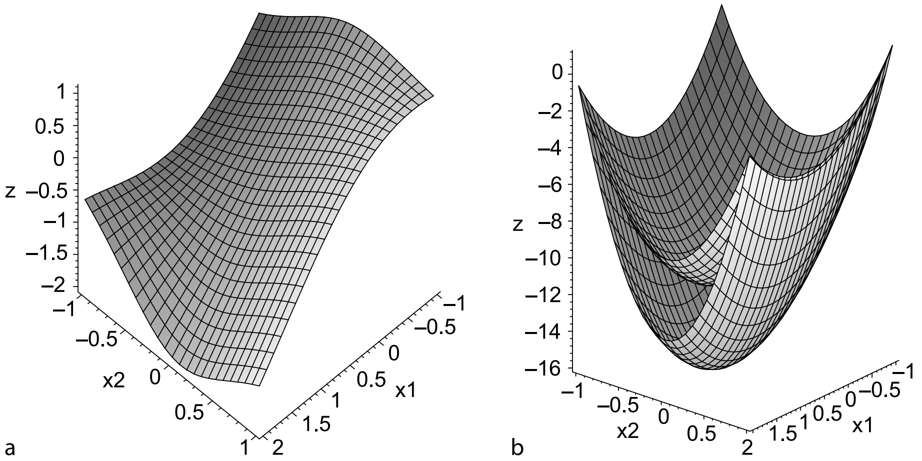
The resulting minima of the two underestimators $L_{\alpha BB}(x;\overline{\alpha})$ and $L_{\alpha BB}(x;\alpha)$ are -762.2377 and -184.4244, respectively. Figure 1 depicts these two underestimators and reveals the improvement in tightness.

Example 2 This example is taken from [2] and examines the following two-dimensional nonconvex function:

$$f_2(x) = \cos(x_1)\sin(x_2) - \frac{x_1}{x_2^2 + 1} ,$$

where $x_1 \in [-1, 2]$ and $x_2 \in [1, 1]$. The above function possesses three minima and its graph is depicted in Fig. 2. The values of the $\overline{\alpha}$ parameters computed by the classical αBB method using (4) are $\overline{\alpha}_1 = 1.921$ and $\overline{\alpha}_2 = 10.921$. Using (10), we can determine the corresponding value for the $\overline{\gamma}$ parameters; these are $\overline{\gamma}_1 = 0.75$ and $\overline{\gamma}_2 = 1.46$. Also, by solving (7) for γ_i, $i = 1, 2$, we obtain $\underline{\gamma}_1 = 0.672$ and $\underline{\gamma}_2 = 1.267$. Using (9), we can determine the corresponding values for the $\underline{\alpha}$ parameters; these are $\underline{\alpha}_1 = 1.3456$ and $\underline{\alpha}_2 = 6.5$.

The iterative algorithm checks whether there exist values of $\gamma_i \in [\underline{\gamma}_i, \overline{\gamma}_i]$, $i = 1, 2$ and $\alpha_i \in [\underline{\alpha}_i, \overline{\alpha}_i]$, $i = 1, 2$, such that the underestimator $L_{\alpha BB}(x;\alpha)$ is convex. After eight iterations it concludes



Global Optimization: g-αBB Approach, Figure 2
Function $f_2(x)$ and comparison of underestimators $L_{\alpha BB}(x;\overline{\alpha})$ and $L_{\alpha BB}(x;\alpha)$

that if $\alpha = (1.8325, \underline{\alpha}_2)$, then $L_{\alpha BB}(x;\alpha)$ is a convex underestimator of $f_2(x)$. Also, if $\gamma = (0.74, \underline{\gamma}_2)$, then $L_1(x;\gamma)$ is also a convex underestimator of $f_2(x)$. Note that only the value of γ_1 had to be increased from its original value, $\underline{\gamma}_1$, and the increase was only by 10%.

The resulting minima of the two underestimators $L_{\alpha BB}(x;\overline{\alpha})$ and $L_{\alpha BB}(x;\alpha)$ are -15.88469 and -10.22767, respectively. Figure 2 depicts these two underestimators and reveals the improvement in tightness.

References

1. Adjiman CS, Floudas CA (1996) Rigorous convex underestimators for general twice-differentiable problems. J Glob Optim 9:23–40
2. Adjiman CS, Dallwig S, Floudas CA, Neumaier A (1998) A global optimization method, *αBB*, for general twice-differentiable constrained NLPS I. theoretical advances. Comput Chem Eng 22:1137–1158
3. Adjiman CS, Androulakis IP, Floudas CA (1998) A global optimization method, αBB, for general twice-differentiable constrained NLPs II. Implementation and Computational Results. Comput Chem Eng 22:1159–1179
4. Akrotirianakis IG, Floudas CA (2004) A new class of improved convex underestimators for twice continuously differentiable constrained NLPs. J Glob Optim 30:367–390
5. Akrotirianakis IG, Floudas CA (2004) Computational experience with a new class of convex underestimators: box-constrained NLP problems. J Glob Optim 29:249–264
6. Androulakis IP, Maranas CD, Floudas CA (1995) *αBB*: A global optimization method for general constrained nonconvex problems. J Glob Optim 7:337–363
7. Floudas CA, Akrotirianakis IG, Caratzoulas S, Meyer CA, Kallrath J (2005) Global optimization in the 21st century: advances and challenges. Comput Chem Eng 29: 1185–1202
8. Hertz D, Adjiman CS, Floudas CA (1999) Two results on bounding the roots of interval polynomials. Comput Chem Eng 23:1333–1339
9. Locatelli M, Schoen F (1999) Random linkage: a family of acceptance/rejection algorithms for global optimization. Math Program 85:379–396
10. Maranas CD, Floudas CA (1994) Global minimum potential energy conformations of small molecules. J Glob Optim 4:135–170
11. Maranas CD, Floudas CA (1994) A deterministic global optimization approach for molecular structure determination. J Chem Phys 100:1247–1261

Global Optimization in Generalized Geometric Programming

COSTAS D. MARANAS
Pennsylvania State University, University Park, USA

MSC2000: 90C26, 90C90

Article Outline

Keywords

Signomials; Generalized geometric programming; Global optimization; Robust stability analysis

Generalized geometric GGP or *signomial programming* (GGP) problems are characterized by an objective function and constraints which are the difference of two *posynomials*. A posynomial $G(\mathbf{x})$ is simply the sum of a number of *posynomial terms* or *monomials* $g_k(\mathbf{x})$, $k = 1, \ldots, K$, multiplied by some positive real constants c_k, $k = 1, \ldots, K$. Each monomial $g_k(\mathbf{x})$ is in turn the product of a number of positive variables each of them raised to some real power,

$$g_k(\mathbf{x}) = x_1^{d_{1,k}} \cdots x_n^{d_{N,k}}, \quad k = 1, \ldots, K,$$

where $d_{1,k}, \ldots, d_{N,k} \in \mathbf{R}$ and are not necessarily integers. The term 'geometric programming' was adopted because of the key role that the well-known arithmetic-geometric inequality played in the initial developments. Generalized geometric problems were first introduced and studied by U. Passy and D.J. Wilde [28] and G.J. Blau and Wilde [8] when existing (posynomial) geometric programming (GP) formulations failed to account for the presence of negatively signed monomials in models for important engineering applications. These applications are extensively reviewed in [31] and [16]. Chemical engineering applications include heat exchanger network design [14], chemical reactor design [8,9], optimal condenser design [4], oxygen production [21], membrane separation process design [12], optimal design of cooling towers [16], chemical equilibrium problems [29], optimal control [23], batch plant modeling [20,33], optimal location of hydrogen supply centers [3] and many more.

By grouping together monomials with identical sign, the generalized geometric problem can be formulated as the following nonlinear optimization problem:

$$GGP \begin{cases} \min\limits_{\mathbf{t}} & G_0(\mathbf{t}) = G_0^+(\mathbf{t}) - G_0^-(\mathbf{t}) \\ \text{s.t.} & G_j(\mathbf{t}) = G_j^+(\mathbf{t}) - G_j^-(\mathbf{t}) \leq 0, \\ & j = 1, \ldots, M, \\ & t_i \geq 0, \quad i = 1, \ldots, N, \end{cases}$$

where

$$G_j^+(\mathbf{t}) = \sum_{k \in K_j^+} c_{jk} \prod_{i=1}^N t_i^{\alpha_{ijk}}, \quad j = 0, \ldots, M,$$

$$G_j^-(\mathbf{t}) = \sum_{k \in K_j^-} c_{jk} \prod_{i=1}^N t_i^{\alpha_{ijk}}, \quad j = 0, \ldots, M,$$

where $\mathbf{t} = (t_1, \ldots, t_N)$ is the positive variable vector; G_j^+, G_j^-, $j = 0, \ldots, M$, are positive posynomial functions in $\mathbf{t}$; α_{ijk} are arbitrary real constant exponents; and c_{jk} are positive coefficients. Also, the sets K_j^+, K_j^- count how many positively/negatively signed monomials form posynomials G_j^+, G_j^- respectively. In general, formulation GGP corresponds to a nonlinear optimization problem with nonconvex objective function and/or constraint set. Note that if we set $K_j^- = 0$ for all $j = 0, \ldots, M$ then the mathematical model for GGP reduces to the (posynomial) geometric programming (GP) formulation which laid the foundation for the theory of generalized geometric problems.

Unlike (posynomial) problems (GP), the problems GGP remain nonconvex in both their primal and dual representation and no known transformation can convexify them. They may involve multiple local minima and/or nonconvex feasible regions and therefore are much more difficult problems to solve. Local optimization approaches for solving GGP problems include bounding procedures based on *posynomial condensation* [2,5,13,15,23]; iterative solution of KKT conditions [9,25,32]; and adaptations of general purpose nonlinear programming methods [1,7,10,19,24,26,31]. A computational comparison of available codes for signomial programming is given in [12,32]. While local optimization methods for solving GGP problems are ubiquitous, application of specialized global optimization algorithms on GGP problems is scarce. J.E. Falk [17] proposed such a global optimization algorithm based on the exponential variable transformation of GGP and the convex relaxation and branch and bounding on the space of exponents of negative monomials ($j = 1, \ldots, M$ and $k \in K_j^-$). Based on these ideas, C.D. Maranas and C.A. Floudas [27] proposed an alternative partitioning in the typically smaller space of variables $i =$

1, …, N. The proposed branch and bound type algorithm attains finite ϵ-convergence to the global minimum through the successive refinement of a convex relaxation of the feasible region and/or of the objective function and the subsequent solution of a series of nonlinear convex optimization problems. The efficiency of the proposed approach is enhanced by eliminating variables through monotonicity analysis, by maintaining tightly bound variables through rescaling, by further improving the supplied variable bounds through convex minimization. The proposed approach was applied to a large number of test examples, in particular robust stability analysis problems.

Robust Stability Analysis

Robust stability analysis of nonlinear systems involves the identification of the largest possible region in the un- certain model parameter space for which the controller manages to attenuate any disturbances in the system. The stability of a feedback structure is determined by the roots of the closed loop characteristic equation:

$$\det(I + P(s, \mathbf{q})C(s, \mathbf{q})) = 0,$$

where $\mathbf{q}$ is the vector of the uncertain model parameters, and $P(s)$, $C(s)$ the transfer functions of the plant and controller, respectively. After expanding the determinant we have:

$$\begin{aligned} P(s, \mathbf{q}) &= a_n(\mathbf{q})s^n \\ &+ a_{n-1}(\mathbf{q})s^{n-1} + \cdots + a_0(\mathbf{q}) = 0, \end{aligned}$$

where the coefficients $a_i(\mathbf{q})$, $i = 0, \ldots, n$, are typically multivariable polynomial functions. The 'zero exclusion condition' (ZEC) implies that a system with characteristic equation $P(\mathbf{q}, s) = 0$ is stable only if it does not have any roots on the imaginary axis for any realization of the $\mathbf{q}$s in the uncertain model parameter space $\mathcal{Q}$:

$$0 \notin P(j\omega, \mathbf{q}), \quad \forall \mathbf{q} \in \mathcal{Q}, \text{ and } \forall \omega \in [0, \infty].$$

A stability margin k_m can then be defined as follows:

$$k_m(j\omega) = \inf\{k\colon\ P(j\omega, \mathbf{q}(k)) = 0,\ \forall \mathbf{q} \in \mathcal{Q}\}\,.$$

Robust stability for this model is then guaranteed if and only if

$$k_m \geq 1.$$

Geometrically, k_m expands the initial uncertain parameter region $\mathcal{Q}$ as much as possible without loosing stability. Note that, typically real parameter uncertainty is expressed as bounds on the real parameters of the model.

Checking the stability of a particular system with characteristic equation $P(j\omega, \mathbf{q})$ involves the solution of the following nonconvex optimization problem.

$$(S)\begin{cases} \displaystyle\min_{q_i, k \geq 0, \omega \geq 0} & k \\ \text{s.t.} & \operatorname{Re}[P(j\omega, \mathbf{q})] = 0 \\ & \operatorname{Im}[P(j\omega, \mathbf{q})] = 0 \\ & q_i^N - \Delta q_i^- k \leq q_i \\ & \quad \leq q_i^N + \Delta q_i^+ k, \\ & i = 1, \ldots, n, \end{cases}$$

where $\mathbf{q}^N$ is a stable nominal point for the uncertain parameters and $\Delta\mathbf{q}^+$, $\Delta\mathbf{q}^-$ are estimated bounds. Note that it is important to be able to always locate the global minimum of (S), otherwise the stability margin might be overestimated. This overestimation can sometimes lead to the erroneous conclusion that a system is stable when it is not. Because for most problems without time delays $a_i(\mathbf{q})$, $i = 0, \ldots, n$, are multivariable polynomial functions, formulation (S) corresponds to a generalized geometric problem. Next, an illustrative robust stability example is highlighted.

This example was studied in [18] and [30]. The plant has three uncertain parameters and the characteristic equation is:

$$\begin{aligned} P(s, q_1, q_2, q_3) = s^4 &+ (10 + q_2 + q_3)s^3 \\ &+ (q_2q_3 + 10q_2 + 10q_3)s^2 \\ &+ (1 - q_2q_3 + q_1)s + 2q_1\,. \end{aligned}$$

The nominal values of the parameters of the system are

$$q_1^N = 800, \quad q_2^N = 4, \quad q_3^N = 6,$$

and the bounded perturbations are:

$$\begin{aligned} \Delta q_1^+ &= \Delta q_1^- = 800, \\ \Delta q_2^+ &= \Delta q_2^- = 2, \\ \Delta q_3^+ &= \Delta q_3^- = 3. \end{aligned}$$

After eliminating ω the zero exclusion formulation becomes:

$$\begin{cases} \min & k \\ \text{s.t.} & 10q2^2q_3^3 + 10q_2^3q_3^2 + 200q_2^2q_3^2 \\ & \quad +100q_2^3q_3 + 100q_2q_3^3 + q_1q_2q_3^2 \\ & \quad +q_1q_2^2q_3 + 1000q_2q_3^2 + 8q_1q_3^2 \\ & \quad +1000q_2^2q_3 + 8q_1q_2^2 + 6q_1q_2q_3 \\ & \quad +60q_1q_3 + 60q_1q_2 \\ & \quad -q_1^2 - 200q_1 \leq 0 \\ & 800 - 800k \leq q_1 \leq 800 + 800k \\ & 4 - 2k \leq q_2 \leq 4 + 2k \\ & 6 - 3k \leq q_3 \leq 6 + 3k . \end{cases}$$

The stability margin is found to be $k_m = 0.3417$, which implies that the system is unstable. Furthermore, the first instability occurs at:

$$\begin{aligned} q_1^* &= 1073.4 , \\ q_2^* &= 3.318 , \\ q_3^* &= 4.975 . \end{aligned}$$

See also

- ► αBB Algorithm
- ► Continuous Global Optimization: Models, Algorithms and Software
- ► Convex Envelopes in Optimization Problems
- ► Global Optimization in Batch Design Under Uncertainty
- ► Global Optimization Methods for Systems of Nonlinear Equations
- ► Global Optimization in Phase and Chemical Reaction Equilibrium
- ► Interval Global Optimization
- ► MINLP: Branch and Bound Global Optimization Algorithm
- ► MINLP: Global Optimization with αBB
- ► Smooth Nonlinear Nonconvex Optimization

References

1. Abrams RA, Wu CT (1978) Projection and restriction methods in geometric and related problems. JOTA 26:59
2. Avriel M, Dembo R, Passy U (1975) Solution of generalized geometric programs. SIAM Internat J Numer Methods Eng 26:291
3. Avriel M, Gurovich V (1967) Optimal condenser design by geometric programming. I–EC Proc Des Developm 6: 256
4. Avriel M, Wilde DJ (1967) Optimal condenser design by geometric programming. I–EC Proc Des Developm 6:256
5. Avriel M, Williams AC (1970) Complementary geometric programming. SIAM J Appl Math 19:125
6. Avriel M, Williams AC (1971) An extension of geometric programming with applications in engineering optimization. J Eng Math 5(3):187
7. Beck PA, Ecker JG (1975) A modified concave simplex algorithm for geometric programming. JOTA 15:189
8. Blau GE, Wilde DJ (1969) Generalized polynomial programming. Canad J Chem Eng 47:317
9. Blau GE, Wilde DJ (1971) A Lagrangian algorithm for equality constrained generalized polynomial optimization. AIChE 17:235
10. Bradley J (1973) An algorithm for the numerical solution of prototype geometric programs. Inst Indust Res and Standards (Dublin, Ireland)
11. Dembo RS (1976) A set of geometric programming test problems and their solutions. Math Program 10:192
12. Dembo RS (1978) Optimal design of a membrane separation process using signomial programming. Math Program 15:12
13. Duffin RJ (1970) Linearizing geometric problems. SIAM Rev 12:211
14. Duffin RJ, Peterson EL (1966) Duality theory for geometric programming. SIAM J Appl Math 14:1307
15. Duffin RJ, Peterson EL (1972) Reversed geometric programming treated by harmonic means. Indiana Univ Math J 22:531
16. Ecker JG, Wiebking RD (1978) Optimal design of a dry–type natural–draft cooling tower by geometric programming. JOTA 26:305
17. Falk JE (1973) Global solutions of signomial programs. Report The George Washington Univ Program in Logistics
18. Gaston RRE, Safonov MG (1988) Exact calculation of the multi-loop stability margin. IEEE Trans Autom Control 33:68
19. Haarhoff PC, Buys JD (1970) A new method for the optimization of a nonlinear function subject to nonlinear constraints. Comput J 13:178
20. Hellinckx LJ, Rijckaert MJ (1971) Minimization of capital investment for batch processes. I–EC Proc Des Developm 10:422
21. Hellinckx LJ, Rijckaert MJ (1972) Optimal capacities of production facilities: An application of geometric programming. Canad J Chem Eng 50:148
22. Horst R, Tuy H (1990) Global optimization, deterministic approaches. Springer, Berlin
23. Jefferson TR, Scott CH (1978) Generalized geometric programming applied to problems of optimal control: I. Theory. JOTA 26:117

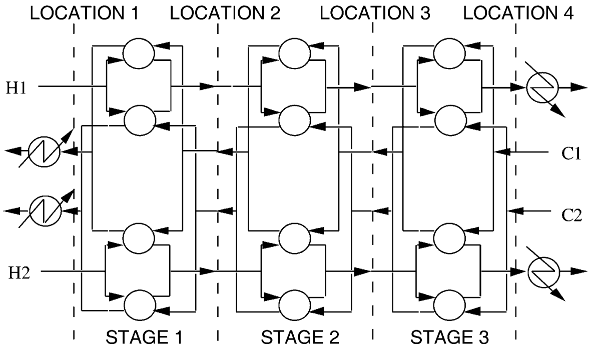
24. Kochenberger A, Woolsey RED, McCarl BA (1973) On the solution of geometric programming via separable programming. Oper Res Quart 24:285
25. Kuester JL, Mize JH (1973) Optimization techniques with Fortran. McGraw-Hill, New York
26. Ratner M, Lasdon LS, Jain A (1978) Solving geometric problems using GRG: Results and comparisons. JOTA 26:253
27. Maranas CD, Floudas CA (1997) Global optimization in generalized geometric programming. Comput Chem Eng 21:251
28. Passy U, Wilde DJ (1967) Generalized polynomial optimizations. SIAM J Appl Math 15:1344
29. Passy U, Wilde DJ (1967) A geometric programming algorithm for solving chemical equilibrium problems. SIAM Rev 11:89
30. Psarris P, Floudas CA (1995) Robust stability analysis of linear and nonlinear systems with real parameter uncertainty. Comput Chem Eng 5:699
31. Rijckaert MJ, Martens XM (1978) Bibliographical note on geometric programming. JOTA 2:325
32. Rijckaert MJ, Martens XM (1978) Comparison of generalized geometric programming algorithms. JOTA 26:205
33. Salomone HE, Iribarren OA (1992) Posynomial modeling of batch plants: A procedure to include process decision variables. Comput Chem Eng 16:173

Global Optimization of Heat Exchanger Networks

JUAN M. ZAMORA[1], IGNACIO E. GROSSMANN[2]
[1] University Autónoma Metropolitana-Iztapalapa, Mexico City, Mexico
[2] Carnegie Mellon University, Pittsburgh, USA

MSC2000: 90C26, 90C90

Article Outline



Global Optimization of Heat Exchanger Networks, Figure 1
Head exchanger network superstructure

Keywords

Global optimization; Heat exchanger networks; Convex relaxations; Global optimal design; Bilinear terms; Linear fractional terms

The cost of energy represents an important part of the total operating cost of many processing plants. Therefore, the recovery of energy through heat exchanger networks (HENs) has played an important role in industry, and has been a major concern of design engineers for the last two decades (for reviews, see [5,10,11]). Design approaches based on mathematical programming techniques and models have been developed and applied in the synthesis and the optimization of HENs (see for instance [3,12,18]). The synthesis of HENs with a mathematical modeling framework involves the optimization of a superstructure like the one in Fig. 1 [18], and represents a difficult global optimization problem from a deterministic point of view [20].

Nonconvexities are introduced into mathematical models for HENs by the fractional powers of linear fractional terms that appear in heat transfer area cost terms,

$$\text{Area Cost} = C\left(\frac{q}{U\Delta T}\right)^{\beta} .$$

Here the variables are the heat transfer rate, q, and the logarithmic mean temperature difference driving force, ΔT or LMTD, U is the heat transfer coefficient, and C and β are cost coefficient and exponent, respectively. Other sources of nonconvexities in mathematical programming models for heat exchanger networks arise due to the logarithmic mean temperature difference driving force, which can be given rigorously

$$\mathrm{LMTD} = \frac{\left[dt_{\mathrm{h}} - dt_{\mathrm{c}}\right]}{\log\left[\frac{dt_{\mathrm{h}}}{dt_{\mathrm{c}}}\right]},$$

or by an approximation like the ones due to W.R. Paterson [13]

$$\mathrm{LMTD} = \frac{1}{3}\left[\frac{1}{2}(dt_{\mathrm{h}} + dt_{\mathrm{c}})\right] + \frac{2}{3}\sqrt{dt_{\mathrm{h}} dt_{\mathrm{c}}}$$

or J.J.J. Chen [2]

$$\mathrm{LMTD} = \left[\frac{(dt_{\mathrm{h}})(dt_{\mathrm{c}})(dt_{\mathrm{h}} + dt_{\mathrm{c}})}{2}\right]^{\frac{1}{3}}.$$

Here, dt_{h} and dt_{c} are the temperature differences at the hot and cold extremes in the heat exchanger. Nonconvexities in mathematical models of HENs also may appear in the form of bilinear terms that are used to model the nonisothermal mixing of process streams. For instance, the energy balance for modeling the nonisothermal mixing of process streams 1 and 2 to produce stream 3 would require the inclusion of the following bilinear equation in the mathematical model:

$$f_1 t_1 + f_2 t_2 = f_3 t_3,$$

in which f stands for heat capacity flowrate, and t for stream temperature.

The issue of determining a global optimum solution for problems involving heat exchanger networks was first considered in [17]. Since then, representative global optimization problems in heat exchanger networks have been posed, see for instance [4]. Nevertheless, deterministic global optimization algorithms, and their application to the optimization of certain classes of NLP and MINLP models in heat exchanger networks appeared only until the 1990s in [1,6,9,14,15,16,20,21,22].

Most of the applications of deterministic global optimization algorithms for the solution of nonconvex problems involving HENs are based on a *branch and bound* framework [7,8]. Within the branch and bound approach for global optimization, lower bounds of the global minimum value of the objective function are computed by solving a convex relaxation of the original nonconvex problem over subsections of the search region. For the development of the convex relaxations for nonconvex problems in HENs, the following properties are exploited.

Property 1 ([19,20,21,22]) Let θ and ΔT be continuous positive variables with $\Delta T > 0$. Also, let U, C, α and β be positive constants, with $\beta > 0$, and $\alpha = (\beta + 1)/\beta$. Then, the function

$$C\left(\frac{\theta^{\alpha}}{U \Delta T}\right)^{\beta}$$

is convex. Furthermore, if q is a positive variable, and S is a convex subset in $\mathbf{R}^2_+$, the convex optimization problem in (2) can be used to compute a rigorous lower bound for the solution of the problem in (1), i. e., the problem in (2) is a valid convex relaxation of the problem in (1):

$$\begin{cases} \text{GloMin} & C\left(\frac{q}{U \Delta T}\right)^{\beta} \\ \text{s.t.} & (q, \Delta T) \subseteq S \\ & 0 \le q^L \le q \le q^U, \end{cases} \tag{1}$$

$$\begin{cases} \min & C\left(\frac{\theta^{\alpha}}{U \Delta T}\right)^{\beta} \\ \text{s.t.} & \theta \ge (q^L)^{\frac{1}{\alpha}} \\ & \quad + \frac{(q^U)^{\frac{1}{\alpha}} - (q^L)^{\frac{1}{\alpha}}}{q^U - q^L}(q - q^L) \\ & (q, \Delta T) \subseteq S, \\ & 0 \le q^L \le q \le q^U, \quad \theta \ge 0. \end{cases} \tag{2}$$

Property 2 ([19]) Let dt_{h}, dt_{c} and ΔT, be continuous positive variables. Also, let T_1 and T_2, be positive constants such that $T_1 - T_2 > 0$. Then the following inequalities are convex:

$$\Delta T \le \frac{\left[dt_{\mathrm{h}} - dt_{\mathrm{c}}\right]}{\log\left[\frac{dt_{\mathrm{h}}}{dt_{\mathrm{c}}}\right]},$$

$$\Delta T \le \frac{\left[dt_{\mathrm{h}} - (T_1 - T_2)\right]}{\log\left[\frac{dt_{\mathrm{h}}}{(T_1 - T_2)}\right]},$$

$$\Delta T \le \frac{\left[(T_1 - T_2) - dt_{\mathrm{c}}\right]}{\log\left[\frac{(T_1 - T_2)}{dt_{\mathrm{c}}}\right]}$$

Property 3 ([19]) Let dt_{h}, dt_{c} and ΔT, be continuous positive variables. Also, let T_1 and T_2, be positive constants such that $T_1 - T_2 > 0$. Then the following inequalities, which are based on the Paterson approximation [13] for the LMTD, are convex:

$$\Delta T \leq \frac{1}{3}\left[\frac{(dt_{\mathrm{h}} + dt_{\mathrm{c}})}{2}\right] + \frac{2}{3}\sqrt{dt_{\mathrm{h}} dt_{\mathrm{c}}},$$

$$\Delta T \leq \frac{1}{3}\left[\frac{(dt_{\mathrm{h}} + T_1 - T_2)}{2}\right] + \frac{2}{3}\sqrt{dt_{\mathrm{h}}(T_1 - T_2)},$$

$$\Delta T \leq \frac{1}{3}\left[\frac{(T_1 - T_2 + dt_{\mathrm{c}})}{2}\right] + \frac{2}{3}\sqrt{(T_1 - T_2) dt_{\mathrm{c}}}.$$

Property 4 ([19]) Let dt_{h}, dt_{c} and ΔT, be continuous positive variables. Also, let T_1 and T_2, be positive constants such that $T_1 - T_2 > 0$. Then the following inequalities, which are based on the Chen approximation [2] for the LMTD, are convex:

$$\Delta T \leq \left[\frac{(dt_{\mathrm{h}})(dt_{\mathrm{c}})(dt_{\mathrm{h}} + dt_{\mathrm{c}})}{2}\right]^{\frac{1}{3}},$$

$$\Delta T \leq \left[\frac{(dt_{\mathrm{h}})(T_1 - T_2)(dt_{\mathrm{h}} + T_1 - T_2)}{2}\right]^{\frac{1}{3}},$$

$$\Delta T \leq \left[\frac{(T_1 - T_2)(dt_{\mathrm{c}})(T_1 - T_2 + dt_{\mathrm{c}})}{2}\right]^{\frac{1}{3}}.$$

Property 5 ([19]) Let dt_{h}, dt_{c} be continuous positive variables, and let ΔT be the logarithmic mean temperature difference, $\Delta T = [dt_{\mathrm{h}} - dt_{\mathrm{c}}/\log[dt_{\mathrm{h}}/dt_{\mathrm{c}}]$. Also, assume that r is a constant determined by the ratio of two particular values of dt_{h} and dt_{c}. Then, the following bounding inequality is valid, and holds as an equality along the line determined by the ratio $r = dt_{\mathrm{h}}/dt_{\mathrm{c}}$:

$$\Delta T \leq P(r) dt_{\mathrm{h}} + Q(r) dt_{\mathrm{c}},$$

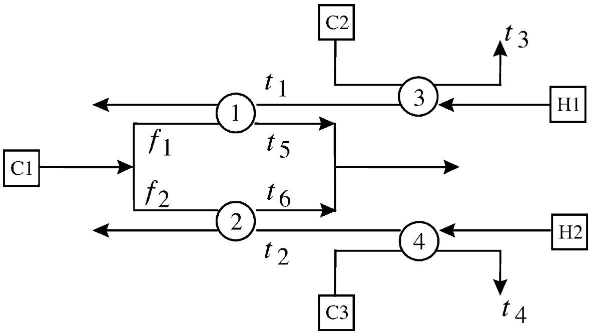
where

$$P(r) = \begin{cases} 0.5 & \text{if } r = 1, \\ \frac{1/r - 1 + \log(r)}{[\log(r)]^2} & \text{if } r \neq 1, \end{cases}$$

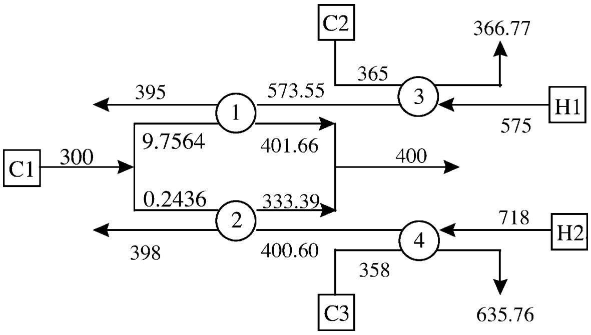
$$Q(r) = \begin{cases} 0.5 & \text{if } r = 1, \\ \frac{r - 1 - \log(r)}{[\log(r)]^2} & \text{if } r \neq 1. \end{cases}$$

Several other useful properties and their application in the development of convex relaxations for HENs problems can be found in [1,6,14,19], and [20,21,22]

As an illustrative example of the use of the above properties, and the application of global optimization techniques in heat exchanger networks, consider the determination of the global optimal design of the HEN shown in Fig. 2 [14]; stream data and cost information are included in Table 1. This problem was originally solved in [14] and [21] using the arithmetic mean temperature difference driving force (AMTD), and assuming isothermal mixing of process streams ($t_5 = t_6$).



Global Optimization of Heat Exchanger Networks, Figure 2
Heat exchanger network for the illustrative problem



Global Optimization of Heat Exchanger Networks, Figure 3
Global optimum HEN design of the illustrative problem

Figure 3 shows the global optimum solution of the nonconvex model (P) associated with the illustrative problem. A design with a total network cost of \$36,199 is determined. Note that model (P) does not assume isothermal mixing, utilizes the approximation by Chen [2], and enforces a minimum approach temperature of 5 degrees. The global optimization of model (P) was performed with the *branch and contract algorithm* proposed in [21,23]; the convex model (R) was used in the computation of rigorous lower bounds of the total network cost. The solution process required 7 branch and bound nodes, and approximately 37 cpu seconds of a Pentium I processor running at 133Mhz. Alternative suboptimal solutions for the illustrative problem based

Global Optimization of Heat Exchanger Networks, Table 1
Problem data for illustrative example

Stream	Tin (K)	Tout (K)	F (kW K^{-1})
H1	575	395	5.555
H2	718	398	3.125
C1	300	400	10
C2	365	–	4.545
C3	358	–	3.571
Cost of Heat Exchanger 1 ($yr^{-1}) = 270[$A_1$(m^2)]			
Cost of Heat Exchanger 2 ($yr^{-1}) = 720[$A_2$(m^2)]			
Cost of Heat Exchanger 3 ($yr^{-1}) = 240[$A_3$(m^2)]			
Cost of Heat Exchanger 4 ($yr^{-1}) = 900[$A_4$(m^2)]			
U_1 =U_1 = 0.1 kW m^{-2} K^{-1}			
U_3 =U_4 = 1.0 kW m^{-2}K^{-1}			

on the rigorous LMTD include network designs with total costs of $38,513, $39,809, $41,836, and $47,681.

Nonconvex Model (P)

Indices

1, 2, 3, 4	=	index for heat exchangers
1*h*, 2*h*, 3*h*, 4*h*	=	hot side of heat exchangers
1*c*, 2*c*, 3*c*, 4*c*	=	cold side of heat exchangers

Parameters

U_1, U_2, U_3, U_4	=	overall heat transfer coefficients

Positive Variables

t	=	stream temperature
dt	=	temperature difference at end of heat exchanger
ΔT	=	approximation of the logarithmic mean temperature difference
q	=	heat transfer rate
f	=	heat capacity flowrate

Objective Function

$$\min 270\frac{q_1}{U_1\Delta T_1} + 720\frac{q_2}{U_2\Delta T_2} + 240\frac{q_3}{U_3\Delta T_3} + 900\frac{q_4}{U_4\Delta T_4}.$$

Model Constraints

$$q_1 = 5.555(t_1 - 395),$$

$$q_1 = f_1(t_5 - 300),$$

$$q_2 = 3.125(t_2 - 398),$$

$$q_2 = f_2(t_6 - 300),$$

$$q_3 = 4.545(t_3 - 365),$$

$$q_3 = 5.555(575 - t_1),$$

$$q_4 = 3.571(t_4 - 358),$$

$$q_4 = 3.125(718 - t_2),$$

$$q_1 + q_2 = 1000,$$

$$q_1 + q_3 = 999.9,$$

$$q_2 + q_4 = 1000,$$

$$f_1 + f_2 = 10,$$

$$dt_{1\mathrm{h}} = t_1 - t_5,$$

$$dt_{1\mathrm{c}} = 95,$$

$$dt_{2\mathrm{h}} = t_2 - t_6,$$

$$dt_{2\mathrm{c}} = 98,$$

$$dt_{3\mathrm{h}} = 575 - t_3,$$

$$dt_{3\mathrm{c}} = t_1 - 365,$$

$$dt_{4\mathrm{h}} = 718 - t_4,$$

$$dt_{4\mathrm{c}} = t_2 - 358,$$

$$\Delta T_1 = \left[\frac{(dt_{1\mathrm{h}})(dt_{1\mathrm{c}})(dt_{1\mathrm{h}} + dt_{1\mathrm{c}})}{2}\right]^{\frac{1}{3}},$$

$$\Delta T_2 = \left[\frac{(dt_{2\mathrm{h}})(dt_{2\mathrm{c}})(dt_{2\mathrm{h}} + dt_{2\mathrm{c}})}{2}\right]^{\frac{1}{3}},$$

$$\Delta T_3 = \left[\frac{(dt_{3\mathrm{h}})(dt_{3\mathrm{c}})(dt_{3\mathrm{h}} + dt_{3\mathrm{c}})}{2}\right]^{\frac{1}{3}},$$

$$\Delta T_4 = \left[\frac{(dt_{4\mathrm{h}})(dt_{4\mathrm{c}})(dt_{4\mathrm{h}} + dt_{4\mathrm{c}})}{2}\right]^{\frac{1}{3}},$$

$$f_1 t_5 + f_2 t_6 = 4000,$$

$$0 \le q_i^L \le q_i \le q_i^U, \quad i = 1, 2, 3, 4,$$

$$0 \le t_j^L \le t_j \le t_j^U, \quad j = 1, 2, 3, 4, 5, 6,$$

$$dt_k \ge 5, \quad k = \mathrm{1h, 1c, 2h, 2c, 3h, 3c, 4h, 4c}$$

$$0 \le f_1^L \le f_1 \le f_1^U, \quad 0 \le f_2^L \le f_2 \le f_2^U.$$

Convex Model (R)

Objective Function

$$\min 270\frac{[\theta_1]^2}{U_1\Delta T_1}+720\frac{[\theta_2]^2}{U_2\Delta T_2}+240\frac{[\theta_3]^2}{U_3\Delta T_3}+900\frac{[\theta_4]^2}{U_4\Delta T_4}.$$

Model Constraints

$$\theta_i \geq (q_i^L)^{\frac{1}{2}}+\frac{(q_i^U)^{\frac{1}{2}}-(q_i^L)^{\frac{1}{2}}}{q_i^U-q_i^L}(q_i-q_i^L),$$

$$i=1,2,3,4,$$

$$q_1=5.555(t_1-395),$$

$$q_1=y_{15}-300f_1,$$

$$q_2=3.125(t_2-398),$$

$$q_2=y_{26}-300f_2,$$

$$q_3=4.545(t_3-365),$$

$$q_3=5.555(575-t_1),$$

$$q_4=3.571(t_4-358),$$

$$q_4=3.125(718-t_2),$$

$$q_1+q_2=1000,$$

$$q_1+q_3=999.9,$$

$$q_2+q_4=1000,$$

$$f_1+f_2=10,$$

$$dt_{1\mathrm{h}}=t_1-t_5,$$

$$dt_{1\mathrm{c}}=95,$$

$$dt_{2\mathrm{h}}=t_2-t_6,$$

$$dt_{2\mathrm{c}}=98,$$

$$dt_{3\mathrm{h}}=575-t_3,$$

$$dt_{3\mathrm{c}}=t_1-365,$$

$$dt_{4\mathrm{h}}=718-t_4,$$

$$dt_{4\mathrm{c}}=t_2-358,$$

$$\Delta T_1 \leq \left[\frac{(dt_{1\mathrm{h}})(dt_{1\mathrm{c}})(dt_{1\mathrm{h}}+dt_{1\mathrm{c}})}{2}\right]^{\frac{1}{3}},$$

$$\Delta T_2 \leq \left[\frac{(dt_{2\mathrm{h}})(dt_{2\mathrm{c}})(dt_{2\mathrm{h}}+dt_{2\mathrm{c}})}{2}\right]^{\frac{1}{3}},$$

$$\Delta T_3 \leq \left[\frac{(dt_{3\mathrm{h}})(dt_{3\mathrm{c}})(dt_{3\mathrm{h}}+dt_{3\mathrm{c}})}{2}\right]^{\frac{1}{3}},$$

$$\Delta T_4 \leq \left[\frac{(dt_{4\mathrm{h}})(dt_{4\mathrm{c}})(dt_{4\mathrm{h}}+dt_{4\mathrm{c}})}{2}\right]^{\frac{1}{3}},$$

$$z_{11}=t_5-300,$$

$$z_{22}=t_6-300, \quad y_{15}+y_{26}=4000,$$

$$y_{15}\geq t_5^L f_1+f_1^L t_5-f_1^L t_5^L,$$

$$y_{15}\geq t_5^U f_1+f_1^U t_5-f_1^U t_5^U,$$

$$y_{15}\leq t_5^L f_1+f_1^U t_5-f_1^U t_5^L,$$

$$y_{15}\leq t_5^U f_1+f_1^L t_5-f_1^L t_5^U,$$

$$y_{26}\geq t_6^L f_2+f_2^L t_6-f_2^L t_6^L,$$

$$y_{26}\geq t_6^U f_2+f_2^U t_6-f_2^U t_6^U,$$

$$y_{26}\leq t_6^L f_2+f_2^U t_6-f_2^U t_6^L,$$

$$y_{26}\leq t_6^U f_2+f_2^L t_6-f_2^L t_6^U,$$

$$z_{11}\geq \frac{1}{f_1}\left(\frac{q_1+\sqrt{q_1^L q_1^U}}{\sqrt{q_1^L}+\sqrt{q_1^U}}\right)^2,$$

$$z_{22}\geq \frac{1}{f_2}\left(\frac{q_2+\sqrt{q_2^L q_2^U}}{\sqrt{q_2^L}+\sqrt{q_2^U}}\right)^2,$$

$$z_{11}\geq \frac{q_1}{f_1^L}+q_1^U\left(\frac{1}{f_1}-\frac{1}{f_1^L}\right),$$

$$z_{11}\geq \frac{q_1}{f_1^U}+q_1^L\left(\frac{1}{f_1}-\frac{1}{f_1^U}\right),$$

$$z_{22}\geq \frac{q_2}{f_2^L}+q_2^U\left(\frac{1}{f_2}-\frac{1}{f_2^L}\right),$$

$$z_{22}\geq \frac{q_2}{f_2^U}+q_2^L\left(\frac{1}{f_2}-\frac{1}{f_2^U}\right),$$

$$z_{11}\leq \frac{1}{f_1^L f_1^U}\left(f_1^U q_1-q_1^L f_1+q_1^L f_1^L\right),$$

$$z_{11}\leq \frac{1}{f_1^L f_1^U}\left(f_1^L q_1-q_1^U f_1+q_1^U f_1^U\right),$$

$$z_{22}\leq \frac{1}{f_2^L f_2^U}(f_2^U q_2-q_2^L f_2+q_2^L f_2^L),$$

$$z_{22}\leq \frac{1}{f_2^L f_2^U}\left(f_2^L q_2-q_2^U f_2+q_2^U f_2^U\right),$$

$$0\leq q_i^L\leq q_i\leq q_i^U, \quad i=1,2,3,4,$$

$$0\leq t_j^L\leq t_j\leq t_j^U, \quad j=1,2,3,4,5,6,$$

$$dt_k\geq 5, \quad k=1\mathrm{h},1\mathrm{c},2\mathrm{h},2\mathrm{c},3\mathrm{h},3\mathrm{c},4\mathrm{h},4\mathrm{c},$$

$$0\leq f_1^L\leq f_1\leq f_1^U, \quad 0\leq f_2^L\leq f_2\leq f_2^U,$$

$$y_{15},y_{26},z_{11},z_{22}\geq 0.$$

See also

- ► MINLP: Global Optimization with αBB
- ► MINLP: Heat Exchanger Network Synthesis
- ► MINLP: Mass and Heat Exchanger Networks
- ► Mixed Integer Linear Programming: Heat Exchanger Network Synthesis
- ► Mixed Integer Linear Programming: Mass and Heat Exchanger Networks

References

1. Adjiman CS, Androulakis IP, Floudas CA (1997) Global optimization of MINLP problems in process synthesis and design. Comput Chem Eng 21:S445–S450
2. Chen JJJ (1987) Letter to the Editors: Comments on improvement on a replacement for the logarithmic mean. Chem Eng Sci 42:2488–2489
3. Ciric AR, Floudas CA (1991) Heat exchanger network synthesis without decomposition. Comput Chem Eng 15:385–396
4. Floudas CA, Pardalos PM (1990) A collection of test problems for constrained global optimization algorithms. no. 455 of Lecture Notes Computer Sci Springer, Berlin
5. Gundersen T, Naess L (1988) The synthesis of cost optimal heat exchanger network synthesis, An industrial review of the state of the art. Comput Chem Eng 12:503–530
6. Hashemi-Ahmady A, Zamora JM, Gundersen T (1999) A sequential framework for optimal synthesis of industrial size heat exchanger networks. In: Proc. 2nd Conf. Process Integration, Modeling and Optimization for Energy Saving and Pollution Reduction (PRESS'99), Hungarian Chemical Soc.
7. Horst R, Pardalos PM (eds) (1995) Handbook of global optimization. Kluwer, Dordrecht
8. Horst R, Tuy H (1993) Global optimization: Deterministic approaches, 2nd edn. Springer, Berlin
9. Iyer RR, Grossmann IE (1996) Global optimization of heat exchanger networks with fixed configuration for multiperiod design. In: Grossmann IE (ed) Global Optimization in Engineering Design. Kluwer, Dordrecht
10. Jezowski J (1994) Heat exchanger network grassroot and retrofit design: The review of the state of the art - Part I: Heat exchanger network targeting and insight based methods of synthesis. Hungarian J Industr Chem 22:279–294
11. Jezowski J (1994) Heat exchanger network grassroot and retrofit design: The review of the state of the art - Part II: Heat exchanger network synthesis by mathematical methods and approaches for retrofit design. Hungarian J Industr Chem 22:295–308
12. Papoulias SA, Grossmann IE (1983) A structural optimization approach in process synthesis II. Heat recovery networks. Comput Chem Eng 7:707–721
13. Paterson WR (1984) A replacement for the logarithmic mean. Chem Eng Sci 39:1635–1636
14. Quesada I, Grossmann IE (1993) Global optimization algorithm for heat exchanger networks. Industr Eng Chem Res 32:487–499
15. Ryoo HS, Sahinidis NV (1995) Global optimization of nonconvex NLPs and MINLPs with applications in process design. Comput Chem Eng 19:551–566
16. Visweswaran V, Floudas CA (1996) Computational results for an efficient implementation of the GOP algorithm and its variants. In: Grossmann IE (ed) Global Optimization in Engineering Design. Kluwer, Dordrecht
17. Westerberg AW, Shah JV (1978) Assuring a global optimum by the use of an upper bound on the lower (dual) bound. Comput Chem Eng 2:83–92
18. Yee TF, Grossmann IE (1990) Simultaneous optimization models for heat integration-II. Heat exchanger network synthesis. Comput Chem Eng 14:1165–1184
19. Zamora JM (1997) Global optimization of nonconvex NLP and MINLP models. PhD Thesis, Dept. Chemical Engin. Carnegie-Mellon Univ.
20. Zamora JM, Grossmann IE (1997) A comprehensive global optimization approach for the synthesis of heat exchanger networks with no stream splits. Comput Chem Eng 21:S65–S70
21. Zamora JM, Grossmann IE (1998) Continuous global optimization of structured process systems models. Comput Chem Eng 22:1749–1770
22. Zamora JM, Grossmann IE (1998) A global MINLP optimization algorithm for the synthesis of heat exchanger networks with no stream splits. Comput Chem Eng 22:367–384
23. Zamora JM, Grossmann IE (1999) A branch and contract algorithm for problems with concave univariate, bilinear and linear fractional terms. J Global Optim 14:217–249

Global Optimization: Hit and Run Methods

ZELDA B. ZABINSKY
Industrial Engineering University Washington,
Seattle, USA

MSC2000: 90C26, 90C90

Article Outline

Keywords

Global optimization; Stochastic methods; Random search algorithms; Adaptive search; Simulated annealing; Improving hit and run; Hit and run methods; Mixed discrete-continuous global optimization; Pure random search; Pure adaptive search

The *hit and run algorithms* fall into the category of sequential random search methods (cf. also ► Random search methods), or stochastic methods. These methods can be applied to a broad class of global optimization problems. They seem especially useful for problems with black-box functions which have no known structure. These problems often involve a very large number of variables, and may include both continuous and discrete variables.

The concept of hit and run is to iteratively generate a sequence of points by taking steps of random length in randomly generated directions. R.L. Smith, in 1984 [12], showed that this method can be used to generate points within a set S that are asymptotically uniformly distributed. The hit and run method was originally applied to identifying nonredundant constraints in linear programs [1,3], and in stochastic programming [2].

Hit and run was first applied to optimization in [16], and the name *improving hit and run* (IHR) was adopted. The term 'improving' was intended to indicate that the sequence of points were improving with regard to their objective function values. The IHR algorithm couples the idea of *pure adaptive search* [8,15] with the hit and run generator to produce an easily implemented sequential random search algorithm. Pure adaptive search (PAS, see also ► Random search methods) predicts that points uniformly generated in improving level sets has, on the average, a linear number of iterations in terms of dimension. One way to approximate PAS, would be to use hit and run to generate approximately uniform points, and then select those that land in improving level sets. This is the idea behind improving hit and run.

In addition to IHR, a family of methods have been developed that are based on hit and run. Other variations include: adding an acceptance probability with a cooling schedule, varying the choice of direction, varying the length of step, and modifying the sampling method to include a mixture of continuous and discrete variables.

Hit and Run Based Algorithms

The underlying concept of hit and run based algorithms is that, if hit and run could generate a uniformly distributed point in an improving level set, then PAS predicts that we need only a linear number of such points. The point generated by just one iteration of hit and run is far from uniform and may not be in the improving set, so the number of function evaluations is not expected to be linear in dimension, but in [16] it was shown that the expected number of function evaluations for IHR on the class of elliptical programs (e. g. positive definite quadratic programs) is polynomial in dimension, $O(n^{5/2})$. The number of function evaluations includes those points that are rejected because they do not fall into the improving level set. This theoretical performance result motivates the use of hit and run for optimization. Numerical experience indicates that IHR has been especially useful in high-dimensional global optimization problems when there are many local minima embedded within a broad convex structure.

The general framework for a hit and run based optimization algorithm for solving a global optimization problem,

$$\begin{cases} \min & f(x) \\ \text{s.t.} & x \in S, \end{cases}$$

where f is a real-valued function on S, is stated below.

```
PROCEDURE hit and run optimization method()
    InputInstance();
    Generate an initial solution $X_0$;
    Set $Y_0 = f(X_0)$;
    Set $k = 0$;
    DO until stopping criterion is met;
        Generate a random direction $D_k$;
        Generate a random steplength $\lambda_k$;
        Evaluate candidate point $W_k = X_k + \lambda_k D_k$;
        Update the new point,
            $X_{k+1} = \begin{cases} W_k & \text{if candidate point accepted} \\ X_k & \text{if rejected} \end{cases}$
        Set $Y_{k+1} = \min(Y_k, f(X_{k+1}))$;
    OD;
    RETURN(Best solution found, $Y_{k+1}$);
END hit and run optimization method;
```

Pseudocode for a hit and run based optimisation algorithm

Improving hit and run uses the most basic hit and run generator, which is to generate a direction vector D_k that is uniformly distributed on a hypersphere, and then generate a steplength λ_k which is generated uniformly on the intersection of D_k with the feasible set S. In many applications, S may be an n-dimensional polytope described by linear constraints, in which case the intersection of a direction with S is easily computed using a slight modification of a minimum ratio test (see [16] for details). This is the most basic hit and run generator, but several variations have been developed.

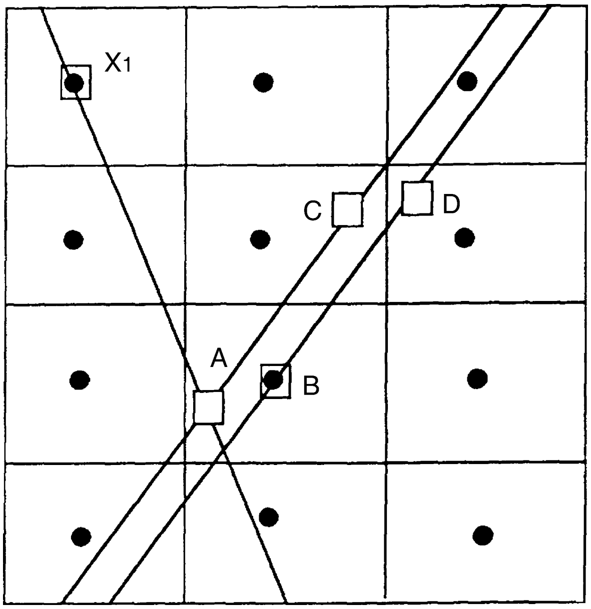
One variation is to add an acceptance probability with a cooling schedule to the *hit and run generator*, as in simulated annealing (cf. ► Simulated annealing). This was developed in [10] and called the *hide-and-seek algorithm*. Just as IHR was motivated by pure adaptive search, hide-and-seek was motivated by adaptive search [9] (see also ► Random search methods). Adaptive search generates a series of points according to a sequence of Boltzman distributions, with parameter T changing on each iteration. The theory predicts that *adaptive search* with decreasing temperature parameter T will converge with probability one to the global optimum, and the number of improving points have the same linear bound as PAS. Hide-and-seek uses the basic hit and run generator, but accepts the candidate point with the Metropolis criterion and parameter T. It is interesting to consider the two extremes of the acceptance probability: if the temperature is fixed at infinity, then all candidate points are accepted, and the hit and run generator approximates *pure random search* with a uniform distribution; at the other extreme if the temperature is fixed to zero, then only improving points are accepted, and we have *improving hit and run*. H.E. Romeijn and Smith derived a cooling schedule which essentially starts with hit and run, and approaches IHR. They proved that hide-and-seek will eventually converge to the global optimum, even though it may experience deteriorations in objective function values. They also present computational results on several test functions, which compare favorably with other algorithms in the literature.

A second variation to the basic hit and run generator is to modify the direction distribution. Thus far, we have only described choosing a direction according to a uniform distribution on an n-dimensional hypersphere, which has also been termed *hyperspherical direction* (HD) choice. In [16] and [10], the direction distribution is defined more generally; the direction may be generated from a multivariate normal distribution with mean 0 and covariance matrix H. If the H matrix is the identity matrix, then the direction distribution is essentially the uniform distribution on a hypersphere. In [4] a nonuniform direction distribution is derived that optimizes the rate of convergence of the algorithm. Although exact implementation of the optimal direction distribution may be very difficult, it motivates an adaptive direction choice rule called *artificial centering hit and run*.

Another choice for direction distribution is the *coordinate direction* (CD) method, in which the direction is chosen uniformly from the n coordinate vectors (spanning $\mathbf{R}^n$). Both HD and CD versions of direction choice were presented and applied to identifying nonredundant linear constraints in [1]. They were also tested in the context of global optimization in [14]. Computationally, CD can outperform HD on specific problems where the optimum is properly aligned, however HD is guaranteed to converge with probability one, while it is easy to construct problems where CD will never converge to the global optimum. A simple example is given in [5] where local minima are lined up on the coordinate directions, and it is impossible for the CD algorithm to leave the local minimum unless it accepts a nonimproving point. For such an example, in [5] it is shown that the CD algorithm coupled with a nonzero acceptance probability for nonimproving points will converge with probability one. Experimental results were also reported.

A third variation to the basic hit and run generator modifies it to be applicable to discrete domains [7,11]. Hit and run as described so far has been defined on a continuous domain. An extension to a discrete domain was accomplished by superimposing the discrete domain onto a continuous real number system. It was motivated by design variables such as fiber angles in a composite laminate, or diameters in a 10-bar truss, where the discrete variables have a natural continuous analog. Two slightly different modifications have been introduced.

In [11] the candidate points were generated using Hit and run on the expanded continuous domain, where the objective function of a nondiscrete point is equal to the objective function evaluated at its nearest



Global Optimization: Hit and Run Methods, 1
Two schemes to modify hit and run to disrete domains

discrete value. In this way, the modified algorithm operates on a continuous domain where the objective function is a multidimensional step function, with plateaus surrounding the discrete points. This modification still converges with probability 1 to the global optimum, as proven in [11].

The diagram in Fig. 1 illustrates this method. Starting from point X_1, hit and run on the continuous domain generates a candidate point such as A. The objective function at A is set equal to that of its nearest discrete point B, forcing $f(A) = f(B)$. If the candidate point is accepted, then $X_2 = A$, and another candidate point (shown as C) is generated.

A second scheme to modify hit and run to operate on discrete domains is to similarly generate a point on a continuous domain, and then round the generated point to its nearest discrete point in the domain on each iteration [6,7,13]. Again starting from point X_1 in Fig. 1, suppose A is generated. In this version, the candidate point is taken as the nearest discrete neighbor, in this example B. The objective function is evaluated at B, $f(B)$, and if the point is accepted, then $X_2 = B$. The difference in this variation is illustrated by noting that the next candidate point is generated from B instead of from A, see point D in Fig. 1. Also note that only discrete points are maintained. In [6,7] it is shown that this second scheme dominates the first scheme in terms of average performance for the special class of spherical programs, and numerical results have been promising.

Another modification to the basic hit and run generator is in the way the steplength is generated. Instead of generating the point uniformly on the whole line segment, the line segment can be restricted to a fixed length, or adaptively modified. S. Neogi [6] refers to this as full-line length, restricted line length, or adaptive stepsize. In [6] the adaptive stepsize is coupled with an acceptance probability to maintain a fixed probability of generating an improving point. See [6] for a more detailed discussion of this variation of a *simulated annealing* algorithm based on the *hit and run generator*.

The many variations of hit and run have been numerically tested on many test functions and applied to real applications. All of the papers referenced in this article include numerical results, but the details are left to the individual papers. Overall, the theoretical motivations and numerical experience leads us to believe that hit and run is a promising approach to global optimization.

See also

- ► Random Search Methods
- ► Stochastic Global Optimization: Stopping Rules
- ► Stochastic Global Optimization: Two-phase Methods

References

1. Berbee HCP, Boender CGE, Rinnooy Kan AHG, Scheffer CL, Smith RL, Telgen J (1987) Hit-and-run algorithms for the identification of nonredundant linear inequalities. Math Program 37:184–207
2. Birge JR, Smith RL (1984) Random procedures for nonredundant constraint identification in stochastic linear programs. Amer J Math Management Sci 4:41–70
3. Boneh A, Golan A (1979) Constraints' redundancy and feasible region boundedness by random feasible point generator. Third European Congress Oper. Res., EURO III, Amsterdam, 9-11 April 1979
4. Kaufman DE, Smith RL (1998) Direction choice for accelerated convergence in hit-and-run sampling. Oper Res 46(1):84–95
5. Kristinsdottir BP (1997) Analysis and development of random search algorithms. PhD Thesis Univ. Washington
6. Neogi S (1997) Design of large composite structures using global optimization and finite element analysis. PhD Thesis Univ. Washington

7. Neogi S, Zabinsky ZB, Tuttle ME (1994) Optimal design of composites using mixed discrete and continuous variables. Proc. ASME Winter Annual Meeting, Symp. Processing, Design and Performance of Composite Materials, vol 52. Dekker, New York, pp 91–107
8. Patel NR, Smith RL, Zabinsky ZB (1988) Pure adaptive search in Monte Carlo optimization. Math Program 4:317–328
9. Romeijn HE, Smith RL (1994) Simulated annealing and adaptive search in global optimization. Probab Eng Inform Sci 8:571–590
10. Romeijn HE, Smith RL (1994) Simulated annealing for constrained global optimization. J Global Optim 5:101–126
11. Romeijn HE, Zabinsky ZB, Graesser DL, Neogi S (1999) Simulated annealing for mixed integer/continuous global optimization. J Optim Th Appl 101(1)
12. Smith RL (1984) Efficient Monte Carlo procedures for generating points uniformly distributed over bounded regions. Oper Res 32:1296–1308
13. Zabinsky ZB (1998) Stochastic methods for practical global optimization. J Global Optim 13:433–444
14. Zabinsky ZB, Graesser DL, Tuttle ME, Kim GI (1992) Global optimization of composite laminate using improving hit and run. In: Floudas CA, Pardalos PM (eds) Recent Advances in Global Optimization. Princeton Univ. Press, Princeton, 343–365
15. Zabinsky ZB, Smith RL (1992) Pure adaptive search in global optimization. Math Program 53:323–338
16. Zabinsky ZB, Smith RL, McDonald JF, Romeijn HE, Kaufman DE (1993) Improving hit and run for global optimization. J Global Optim 3:171–192

Global Optimization: Interval Analysis and Balanced Interval Arithmetic

Julius Žilinskas[1], Ian David Lockhart Bogle[2]

[1] Institute of Mathematics and Informatics,
Vilnius, Lithuania

[2] Centre for Process Systems Engineering,
Department of Chemical Engineering,
University College London, London, UK

MSC2000: 65K05, 90C30, 90C57, 65G30, 65G40

Article Outline

Keywords and Phrases

Global optimization; Interval arithmetic; Interval computations

Introduction

Mathematically the global optimization problem is formulated as

$$f^* = \min_{X \in D} f(X) ,$$

where a nonlinear function $f(X)$, $f : \mathbb{R}^n \to \mathbb{R}$, of continuous *variables* X, is an *objective function*; $D \in \mathbb{R}^n$ is a *feasible region*; n is a *number of variables*. A *global minimum* f^* and one or all *global minimizers* X^*:

$$f(X^*) = f^*$$

should be found. No assumptions on unimodality are included in the formulation of the problem. Most often an objective function is defined by an analytical formula or an algorithm, which evaluates the value of the objective function using the values of variables and arithmetic operations. ► Continuous global optimization: models, algorithms and software.

One of the classes of methods for global optimization are methods based on *interval arithmetic*. Interval arithmetic [10] provides bounds for the function values over hyper-rectangular regions defined by intervals of variables. The bounds may be used in global optimization to detect the sub-regions of the feasible region which cannot contain a global minimizer. Such sub-regions may be discarded from the subsequent search for a minimum.

Interval arithmetic provides guaranteed bounds but sometimes they are too pessimistic. Interval arithmetic is used in global optimization to provide guaranteed solutions, but there are problems for which the time for optimization is too long. A disadvantage of interval arithmetic is the dependency problem [5]: when a given variable occurs more than once in interval computation, it is treated as a different variable in each occurrence. This causes widening of computed intervals and overestimation of the range of function values.

Analysis of both overestimating and underestimating intervals is useful to estimate how much interval bounds overestimate the range of function values. Moreover inner interval arithmetic operations may be used instead of standard interval arithmetic operations in some cases when dependency of operands is known or operands are known to be monotonic. Although monotonicity cannot easily be determined in advance, inner and standard interval arithmetic operations may be chosen randomly building random interval arithmetic, estimating the range of real function values from a sample of random intervals.

Methods / Applications

Interval Analysis in Global Optimization

Interval arithmetic is proposed in [10]. Interval arithmetic operates with real intervals $\underline{\overline{x}} = \left[\underline{x}, \overline{x}\right] = \{x \in \mathbb{R} | \underline{x} \leq x \leq \overline{x}\}$, defined by two real numbers $\underline{x} \in \mathbb{R}$ and $\overline{x} \in \mathbb{R}$, $\underline{x} \leq \overline{x}$. For any real arithmetic operation $x \circ y$ the corresponding interval arithmetic operation $\underline{\overline{x}} \circ \underline{\overline{y}}$ is defined as an operation whose result is an interval containing every possible number produced by the real operation with the real numbers from each interval. The interval arithmetic operations are defined as:

$$\underline{\overline{x}} + \underline{\overline{y}} = \left[\underline{x} + \underline{y}, \overline{x} + \overline{y}\right],$$

$$\underline{\overline{x}} - \underline{\overline{y}} = \left[\underline{x} - \overline{y}, \overline{x} - \underline{y}\right],$$

$$\underline{\overline{x}} \times \underline{\overline{y}} = \begin{cases} \left[\underline{x}\,\underline{y}, \overline{x}\overline{y}\right], & \underline{\overline{x}} > 0\,, \underline{\overline{y}} > 0\,, \\ \left[\overline{x}\,\underline{y}, \overline{x}\overline{y}\right], & \underline{\overline{x}} > 0\,, 0 \in \underline{\overline{y}}\,, \\ \left[\overline{x}\underline{y}, \underline{x}\,\overline{y}\right], & \underline{\overline{x}} > 0\,, \underline{\overline{y}} < 0\,, \\ \left[\underline{x}\,\overline{y}, \overline{x}\overline{y}\right], & 0 \in \underline{\overline{x}}\,, \underline{\overline{y}} > 0\,, \\ \left[\min(\underline{x}\overline{y}\,, \overline{x}\underline{y})\,,\right. & \\ \quad \left.\max(\underline{x}\underline{y}, \overline{x}\overline{y})\right], & 0 \in \underline{\overline{x}}\,, 0 \in \underline{\overline{y}}\,, \\ \left[\overline{x}\underline{y}, \underline{x}\underline{y}\right], & 0 \in \underline{\overline{x}}\,, \underline{\overline{y}} < 0\,, \\ \left[\underline{x}\overline{y}, \overline{x}\underline{y}\right], & \underline{\overline{x}} < 0\,, \underline{\overline{y}} > 0\,, \\ \left[\underline{x}\overline{y}, \underline{x}\underline{y}\right], & \underline{\overline{x}} < 0\,, 0 \in \underline{\overline{y}}\,, \\ \left[\overline{x}\overline{y}, \underline{x}\underline{y}\right], & \underline{\overline{x}} < 0\,, \underline{\overline{y}} < 0\,, \end{cases}$$

$$\underline{\overline{x}} / \underline{\overline{y}} = \begin{cases} \left[\underline{x}/\overline{y}, \overline{x}/\underline{y}\right], & \underline{\overline{x}} > 0\,, \underline{\overline{y}} > 0\,, \\ \left[\overline{x}/\overline{y}, \underline{x}/\underline{y}\right], & \underline{\overline{x}} > 0\,, \underline{\overline{y}} < 0\,, \\ \left[\underline{x}/\underline{y}, \overline{x}/\underline{y}\right], & 0 \in \underline{\overline{x}}\,, \underline{\overline{y}} > 0\,, \\ \left[\overline{x}/\overline{y}, \underline{x}/\overline{y}\right], & 0 \in \underline{\overline{x}}\,, \underline{\overline{y}} < 0\,, \\ \left[\underline{x}/\underline{y}, \overline{x}/\overline{y}\right], & \underline{\overline{x}} < 0\,, \underline{\overline{y}} > 0\,, \\ \left[\overline{x}/\underline{y}, \underline{x}/\overline{y}\right], & \underline{\overline{x}} < 0\,, \underline{\overline{y}} < 0\,. \end{cases}$$

An interval function can be constructed replacing the usual arithmetic operations by interval arithmetic operations in the formula or the algorithm for calculating values of the function. An interval value of the function can be evaluated using the interval function with interval arguments. The resulting interval always encloses the range of real function values in the hyper-rectangular region defined by the vector of interval arguments:

$$\left\{f\,(X)\,|X \in \underline{\overline{X}}, \underline{X} \in \mathbb{R}^n\,, \overline{X} \in \mathbb{R}^n\right\} \subseteq \underline{\overline{f}}\left(\underline{\overline{X}}\right)\,,$$

where $f : \mathbb{R}^n \to \mathbb{R}$, $\underline{\overline{f}} : [\mathbb{R}, \mathbb{R}]^n \to [\mathbb{R}, \mathbb{R}]$. Because of this property the interval value of the function can be used as the lower and upper bounds for the function in the region which may be used in global optimization.

The first version of interval global optimization algorithm was oriented to minimization of a rational function by bisection of sub-domains [12]. Interval methods for global optimization were further developed in [3,4,11], where the interval Newton method and the test of strict monotonicity were introduced. A thorough description including theoretical as well as practical aspects can be found in [5] where a very efficient interval global optimization method involving monotonicity and non-convexity tests and the special interval Newton method is described. ▶ Interval global optimization.

A *branch and bound* technique is usually used to construct interval global optimization algorithms. An iteration of a classical branch and bound algorithm processes a yet unexplored sub-region of the feasible region. Iterations have three main components: selection of the sub-region from a candidate list to process, bound calculation, and branching. In interval global optimization algorithms bounds are calculated using interval arithmetic. All interval global optimization branch and bound algorithms use the hyperrectangular partitions and branching is usually performed bisecting the hyper-rectangle into two. Variants of interval branch-and-bound algorithms for global optimization where the bisection was substituted by the subdivision of subregions into many subregions in a single iteration step have been investigated in [2]. The convergence properties have been investigated in detail. An extensive numerical study is presented in [8]. ▶ Bisection global optimization methods; ▶ Interval analysis: Subdivision directions in interval branch and bound techniques.

The tightness of bounds is a very important factor for efficiency of branch and bound based global optimization algorithms. An experimental model of interval arithmetic with controllable tightness of bounds to investigate the impact of bound tightening in interval global optimization was proposed in [14]. The experimental results on efficiency of tightening bounds were presented for several test and practical problems. Experiments have shown that the relative tightness of bounds strongly influences efficiency of global optimization algorithms based on the branch and bound approach combined with interval arithmetic.

Underestimating Interval Arithmetic

Kaucher arithmetic [6,7] defining underestimates is useful to estimate how much interval bounds overestimate the range of function values. Kaucher arithmetic operations ($\circ_{\mathrm{u}}$) are defined as:

$$\underline{\overline{x}} +_{\mathrm{u}} \underline{\overline{y}} = \left[\underline{x} + \overline{y} \vee \overline{x} + \underline{y}\right],$$

$$\underline{\overline{x}} -_{\mathrm{u}} \underline{\overline{y}} = \left[\underline{x} - \underline{y} \vee \overline{x} - \overline{y}\right],$$

$$\underline{\overline{x}} \times_{\mathrm{u}} \underline{\overline{y}} = \begin{cases} \left[\underline{x}\,\overline{y} \vee \overline{x}\underline{y}\right], & \underline{\overline{x}} > 0\,, \underline{\overline{y}} > 0 \\ & \text{or } \ \underline{\overline{x}} < 0\,, \underline{\overline{y}} < 0\,, \\ \left[\underline{x}\,\underline{y}, \underline{x}\,\overline{y}\right], & \underline{\overline{x}} > 0\,, 0 \in \underline{\overline{y}}\,, \\ \left[\overline{xy} \vee \underline{x}\,\underline{y}\right], & \underline{\overline{x}} > 0\,, \underline{\overline{y}} < 0 \\ & \text{or } \ \underline{\overline{x}} < 0\,, \underline{\overline{y}} > 0\,, \\ \left[\underline{x}\,\underline{y}, \overline{x}\underline{y}\right], & 0 \in \underline{\overline{x}}\,, \underline{\overline{y}} > 0\,, \\ [0, 0]\,, & 0 \in \underline{\overline{x}}\,, 0 \in \underline{\overline{y}}\,, \\ \left[\overline{xy}, \underline{x}\,\overline{y}\right], & 0 \in \underline{\overline{x}}\,, \underline{\overline{y}} < 0\,, \\ \left[\overline{xy}, \overline{x}\underline{y}\right], & \underline{\overline{x}} < 0\,, 0 \in \underline{\overline{y}}\,, \end{cases}$$

$$\underline{\overline{x}} /_{\mathrm{u}} \underline{\overline{y}} = \begin{cases} \left[\underline{x}/\underline{y} \vee \overline{x}/\overline{y}\right], & \underline{\overline{x}} > 0\,, \underline{\overline{y}} > 0 \\ & \text{or } \ \underline{\overline{x}} < 0\,, \underline{\overline{y}} < 0\,, \\ \left[\overline{x}/\underline{y} \vee \underline{x}/\overline{y}\right], & \underline{\overline{x}} > 0\,, \underline{\overline{y}} < 0 \\ & \text{or } \ \underline{\overline{x}} < 0\,, \underline{\overline{y}} > 0\,, \\ \left[\underline{x}/\overline{y}, \overline{x}/\overline{y}\right], & 0 \in \underline{\overline{x}}\,, \underline{\overline{y}} > 0\,, \\ \left[\overline{x}/\underline{y}, \underline{x}/\underline{y}\right], & 0 \in \underline{\overline{x}}\,, \underline{\overline{y}} < 0\,, \end{cases}$$

where $[a \vee b] = [\min(a, b), \max(a, b)]$. Underestimating interval arithmetic guarantees to underestimate:

$$\underline{\overline{f}_{\mathrm{u}}}\left(\underline{\overline{X}}\right) \subseteq \left\{f(X) \,|\, X \in \underline{\overline{X}},\right\} \subseteq \underline{\overline{f}}\left(\underline{\overline{X}}\right).$$

An interval defined by Kaucher arithmetic is a worst case estimate and can be the degenerate interval $[0, 0]$. A regularized version of Kaucher arithmetic proposed in [13] assumes regularity of the dependency between variables. In the underestimation assuming regularity of the dependency between variables, multiplication operation ($\times_{\mathrm{ur}}$) is defined differently from Kaucher arithmetic:

$$\underline{\overline{x}} +_{\mathrm{ur}} \underline{\overline{y}} = \underline{\overline{x}} +_{\mathrm{u}} \underline{\overline{y}}\,,$$

$$\underline{\overline{x}} -_{\mathrm{ur}} \underline{\overline{y}} = \underline{\overline{x}} -_{\mathrm{u}} \underline{\overline{y}}\,,$$

$$\underline{\overline{x}} \times_{\mathrm{ur}} \underline{\overline{y}} = \begin{cases} \left[\min(\underline{x}\,\overline{y}, \overline{x}\underline{y}), \mu(\underline{x}, \overline{x}, \underline{y}, \overline{y})\right], \\ \qquad \underline{\overline{x}} > 0\,, \underline{\overline{y}} > 0 \ \text{ or } \ \underline{\overline{x}} < 0\,, \underline{\overline{y}} < 0\,, \\ \left[\mu(\underline{x}, \overline{x}, \overline{y}, \underline{y}), \max(\underline{x}\,\underline{y}, \overline{xy})\right], \\ \qquad \underline{\overline{x}} > 0\,, \underline{\overline{y}} < 0 \ \text{ or } \ \underline{\overline{x}} < 0\,, \underline{\overline{y}} > 0\,, \\ \left[\mu(\underline{x}, \overline{x}, \overline{y}, \underline{y}), \mu(\underline{x}, \overline{x}, \underline{y}, \overline{y})\right], \\ \qquad \text{otherwise}\,, \end{cases}$$

$$\underline{\overline{x}} /_{\mathrm{ur}} \underline{\overline{y}} = \underline{\overline{x}} /_{\mathrm{u}} \underline{\overline{y}}\,,$$

where

$$\mu(x_1, x_2, y_1, y_2) = \begin{cases} x_2 y_1\,, & \frac{(x_2-x_1)y_2-x_1(y_2-y_1)}{2(x_2-x_1)(y_2-y_1)} > 1\,, \\ x_1 y_2\,, & \frac{(x_2-x_1)y_2-x_1(y_2-y_1)}{2(x_2-x_1)(y_2-y_1)} < 0\,, \\ \frac{(x_2 y_2-x_1 y_1)^2}{4(x_2-x_1)(y_2-y_1)}\,, & \text{otherwise}\,. \end{cases}$$

In [1,9] *inner interval arithmetic* is defined. If the operands in the interval operations to calculate the function values are known to be monotonic then standard interval arithmetic operations may be combined with inner interval operations to tighten resulting intervals without losing the guarantee of enclosure [1]. If it is known that operands in subtraction or division are dependent or are monotonic and have the same monotonicity (either both are monotonically increasing or both monotonically decreasing) then inner interval operations may be used instead of standard interval operations. If it is known that operands in summation or multiplication are monotonic and do not have the same monotonicity (one is monotonically increasing and another is monotonically decreasing) then inner interval operations may be used instead of standard interval operations.

The difference between inner interval operations ($\circ_{\mathrm{in}}$) and underestimating interval operations ($\circ_{\mathrm{u}}$) concerns the result of multiplication:

$$
\begin{aligned}
&\overline{\underline{x}} +_{\mathrm{in}} \overline{\underline{y}} = \overline{\underline{x}} +_{\mathrm{u}} \overline{\underline{y}}\,,\\
&\overline{\underline{x}} -_{\mathrm{in}} \overline{\underline{y}} = \overline{\underline{x}} -_{\mathrm{u}} \overline{\underline{y}}\,,\\
&\overline{\underline{x}} \times_{\mathrm{in}} \overline{\underline{y}} = \begin{cases} \left[\max(\underline{x}\,\overline{y}, \overline{x}\underline{y}), \min(\underline{x}\,\underline{y}, \overline{x y})\right]\,, & \\ & 0 \in \overline{\underline{x}}, 0 \in \overline{\underline{y}},\\ \overline{\underline{x}} \times_{\mathrm{u}} \overline{\underline{y}}, & \text{otherwise}\,, \end{cases}\\
&\overline{\underline{x}} /_{\mathrm{in}} \overline{\underline{y}} = \overline{\underline{x}} /_{\mathrm{u}} \overline{\underline{y}}\,.
\end{aligned}
$$

Random Interval Arithmetic

It is difficult computationally to find which operands are dependent, to be certain they are monotonic, and to determine their monotonicity (intervals of the derivatives of all operands need to be found). *Random interval arithmetic* proposed in [1] is obtained by choosing standard or inner interval operations *randomly with the same probability* at each step of the computation. The range of function values is estimated using a number of sample intervals evaluated using random interval arithmetic. The estimation is based on the assumptions that the distribution of the centres of the evaluated intervals is normal with a very small relative standard deviation and the distribution of the radii is normal but taking only positive values. The mean value of the centres μ_{centres}, the mean value of the radii μ_{radii} and the standard deviations of the radii σ_{radii} of the random intervals are used to evaluate an approximate range of the function

$$[\mu_{\mathrm{centres}} \pm (\mu_{\mathrm{radii}} + \alpha\sigma_{\mathrm{radii}})]\ , \tag{1}$$

where α is between 1 and 3 depending on the number of samples and the desired probability that the exact range is included in the estimated range. It is suggested in [1] that a compromise between efficiency and robustness can be obtained using $\alpha = 1.5$ and 30 samples. Experimental results presented in [1] for some functions over small intervals show that random interval arithmetic provides tight estimates of the ranges of the considered function values with probability close to 1. However, in the experiments, the intervals of variables of the function considered were small. In the case of large intervals of variables, and particularly for multi-variable functions, the obtained estimates for a range of function values frequently do not fully enclose the range of function values.

For the application of random interval arithmetic to global optimization it is important to extend these ideas to the case of functions defined over large multidimensional regions. *Balanced random interval arithmetic* proposed in [16] extending the ideas of [1], is obtained by choosing standard and inner interval operations at each step of the computation *randomly with predefined probabilities* for the standard and inner operations. A number of sample intervals are evaluated. It is assumed that the distribution of centres of the evaluated balanced random intervals is normal and that the distribution of radii is folded normal, also known as absolute normal, because the radii cannot be negative. The range of values of the function in the defined region is estimated using the mean values (μ) and the standard deviations (σ) of centres and radii of the evaluated balanced random intervals:

$$[\mu_{\mathrm{centres}} \pm (3.0\sigma_{\mathrm{centres}} + \mu_{\mathrm{radii}} + 3.0\sigma_{\mathrm{radii}})]\ . \tag{2}$$

The ranges of values of the objective function estimated using balanced random interval arithmetic can be used in the general branch and bound framework building a stochastic global optimization algorithm. The performance of such an algorithm has been evaluated experimentally on market model estimation [17] and on chemical engineering problems. When speed of optimization is more important than guaranteed reliability, such an algorithm is a good alternative to the algorithm with standard interval arithmetic because it is several times faster.

Balanced Interval Arithmetic

The exact range of function values lies between the results of overestimating and underestimating interval arithmetic. Estimates of the ranges of function values estimated from the results of standard interval arithmetic and inner interval arithmetic were investigated in [15]. There, *balanced interval arithmetic* is defined as the weighted mean of the overestimating and underestimating intervals of the function:

$$pc \times \overline{\underline{f}}\left(\overline{\underline{X}}\right) + (1 - pc) \times \overline{\underline{f}}_{\mathrm{u}}\left(\overline{\underline{X}}\right) , \qquad (3)$$

where the predefined coefficient $0 \leq pc \leq 1$ defines the balance between overestimating and underestimating intervals.

The ranges of the values of several functions estimated using balanced interval arithmetic and using balanced random interval arithmetic have been experimentally compared [15]. The results of the experiments have shown that ranges estimated using balanced interval arithmetic compete with ranges estimated using balanced random interval arithmetic. However balanced interval arithmetic is not based on the assumptions of normal distributions and does not require several samples.

The ranges of values of the objective function estimated using balanced interval arithmetic can be used in the general branch and bound framework building a deterministic global optimization algorithm. When the predefined coefficient pc is less than 1, the algorithm may be faster than the algorithm with standard interval arithmetic.

For each interval function, there exists α, $0 \leq \alpha \leq 1$, for which

$$\left\{f(X)\,|X \in \overline{\underline{X}}, \right\} \subseteq \alpha \times \overline{\underline{f}}\left(\overline{\underline{X}}\right) + (1 - \alpha) \times \overline{\underline{f}}_{\mathrm{u}}\left(\overline{\underline{X}}\right)$$

for all possible sub-regions of the feasible region, $\overline{\underline{X}} \subseteq D$. The algorithm guarantees the exact solution if $pc \geq \alpha$.

See also

- ► Bisection Global Optimization Methods
- ► Continuous Global Optimization: Models, Algorithms and Software
- ► Interval Analysis: Parallel Methods for Global Optimization
- ► Interval Analysis: Subdivision Directions in Interval Branch and Bound Methods
- ► Interval Analysis: Unconstrained and Constrained Optimization
- ► Interval Global Optimization
- ► Interval Linear Systems

References

1. Alt R, Lamotte JL (2001) Experiments on the evaluation of functional ranges using random interval arithmetic. Math Comput Simul 56:17–34
2. Csallner AE, Csendes T, Markót MC (2000) Multisection in interval branch-and-bound methods for global optimization I. Theoretical results. J Glob Optim 16:371–392
3. Hansen E (1978) A globally convergent interval method for computing and bounding real roots. BIT 18:415–424
4. Hansen E (1978) Global optimization using interval analysis – the multidimensional case. Numer Math 34:247–270
5. Hansen E, Walster G (2003) Global Optimization Using Interval Analysis, 2nd edn. Marcel Dekker, New York
6. Kaucher E (1977) Über Eigenschaften und Anwendungsmöglichkeiten der erweiterten Intervallrechnung und des hyperbolischen Fastkörpers über **R**$'$. Comput Suppl 1:81–94
7. Kreinovich V, Nesterov VM, Zheludeva NA (1996) Interval methods that are guaranteed to underestimate (and the resulting new justification of Kaucher arithmetic). Reliab Comput 2:119–124
8. Markót MC, Csendes T, Csallner AE (2000) Multisection in interval branch-and-bound methods for global optimization II. Numerical tests. J Global Optim 16:219–228
9. Markov S (1995) On directed interval arithmetic and its applications. J Univers Comput Sci 1:514–526
10. Moore RE (1966) Interval Analysis. Prentice-Hall, Englewood Cliffs
11. Moore RE (1977) A test for existence of solutions to nonlinear systems. SIAM J Numer Anal 14:611–615
12. Skelboe S (1974) Computation of rational interval functions. BIT 14:87–95
13. Žilinskas A, Žilinskas J (2005) On underestimating in interval computations. BIT Numer Math 45:415–427
14. Žilinskas A, Žilinskas J (2006) On efficiency of tightening bounds in interval global optimization. Lect Note Comput Sci 3732:197–205
15. Žilinskas J (2006) Estimation of functional ranges using standard and inner interval arithmetic. Inform 17:125–136
16. Žilinskas J, Bogle IDL (2004) Balanced random interval arithmetic. Comput Chem Eng 28:839–851
17. Žilinskas J, Bogle IDL (2006) Balanced random interval arithmetic in market model estimation. Eur J Oper Res 175:1367–1378

Global Optimization in Lennard–Jones and Morse Clusters

COSTAS D. MARANAS
Pennsylvania State University, University Park, USA

MSC2000: 90C26, 90C90

Article Outline

Keywords

Lennard–Jones microcluster; Morse microcluster; Minimum potential energy; Global optimization

Microclusters are [11] aggregates of atoms, ions, or molecules, sufficiently small that a significant proportion of these units is present on their surfaces. They correspond to systems that are neither single entities nor continua composed by an infinite number of units, but lie somewhere in between bridging the gap between single atoms or molecules and bulk matter. Typically, microclusters consist of two to several hundred atoms. A key word pertaining to the novel features of microclusters is *size effects* [26]. The microscopic size of microclusters gives rise to unique properties in two ways. First, a large percentage of a cluster's atoms are on or close to the surface, and surface atoms do not arrange themselves in the same way as do atoms in bulk matter, but instead they tend to avoid being exposed on the surface. Assuming a spherical shape, the fraction of the number of surface atoms is $4/n^{1/3}$. For $n = 10^2$ this number is 86%, for $n = 10^3$ is 40% and for $n = 10^4$ is still 20%. For example, in a cluster of 55 argon atoms at least 42 atoms are on the surface in some sense. This effect completely overwhelmsthe tendency of atoms to arrange themselves in a regular crystalline array as they normally do in bulk matter. For instance, the ordering of silicon atoms in the Si_{10} cluster is completely different from the ordering in the silicon crystalline structure. It appears that clusters consisting of specific numbers of atoms are extremely stable, as they show up more prominently in the mass spectrum than neighboring cluster sizes. These numbers of particles that enhance stability are called *magic numbers* and they are substance specific [2]. For instance [3], xenium clusters consisting of $N = 13, 19, 23, 25, \ldots$ are particularly stable, although for sodium clusters the magic numbers are $N = 8, 20, 40, 58, 92, \ldots$.

The study of the topography of the potential energy function of a microcluster in the internal configurational space was and still remains a central problem in this area of research [11,13]. This problemcan be succinctly stated as follows: Given N particles interacting with two-body central forces, find their configuration(s) in the three-dimensional Euclidean space involving the global minimum total potential energy.

This can be expressed mathematically as follows:

$$V = \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} v(r_{ij}),$$

where

$$r_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2},$$
$$x_1 = y_1 = z_1 = y_2 = z_2 = z_3 = 0.$$

Here, V is the total potential energy of the microcluster as the summation of all two-body interaction terms, $v(r_{ij})$ is the potential energy term corresponding to the interaction of particle i with particle j, and r_{ij} is the Euclidean distance between i and j. Note that in the double summation, j spans from $i + 1$ to N so that we avoid double counting pair interactions and the interaction of a particle with itself. Furthermore, by specifying $x_1 = y_1 = z_1 = 0$, we fix the first particle at (0, 0, 0) eliminating all three translational degrees of freedom of the microcluster. By further imposing $y_2 = z_2 = z_3 = 0$ we eliminate the rotational degrees of freedom as well. Pair potentials that have been used in cluster studies include the following [11]:

1) $v(r) = (n - m)^{-1} [nr^{-m} - mr^{-n}]$ (Mie);
2) $v(r) = 4\,\epsilon\{(\sigma/r)^{12} - (\sigma/r)^6\}$ (Lennard–Jones);
3) $v(r) = [1 - e^{a(1-r)}]^2 - 1$ (Morse);
4) $v(r) = Ae^{-ar^2} - Be^{-br^2}$ (Gaussian);
5) $v(r) = z^{\alpha} z^{\beta}/r + Ae^{-r/\rho}$ (Born–Meyer);

Lennard–Jones and Morse potential models are the most popular selections to describe the force field.

Even under simplifying assumptions about the interaction energy, the minimization of the total potential energy is very difficult to solve because it corresponds to a nonconvex optimization problem involving numerous local minima. Hoare [11] claimed that the number of local minima of an n—atom microcluster grows as $\exp(n^2)$. In fact, L.T. Wille [34] has shown that the complexity of determining the global minimum energy of a cluster of particles interacting via two-body forces belongs to the class *NP*. In other words, there is no known algorithm that can solve this problem in nonexponential time [22]. A geometrical, possibly topological proof that a local minimum is both unique and global is not likely to be found because there still exist unsolved problems in the theory of sphere packings where difficulties are without any doubt less acute [4,5,6,10], than those in the minimization problem at hand.

Existing methods use physical intuition, approximation procedures, mimicking of physical phenomena, random searches, lattice optimization/relaxation, or local/global optimization approaches. M.R. Hoare, in a series of papers [12,13,14,15,16], proposed a method of finding minima of the total potential function of an $5 \leq N \leq 66$ particle Lennard–Jones cluster based on a *growth scheme* involving the following steps: First, a particular compact *seed structure* involving a small number of atoms is selected which is likely to appear in the N-particle structure. At each iteration an extra particle is placed at all packing vertices and the resulting structures are tested for geometrical uniqueness. The distinct structures are then relaxed and a local optimization procedure locates and records the local minima involved. Each of the minima then serve as a new seed structure in repetition of the procedure. Finally, all of the generated distinct local minima are tabulated in decreasing order of binding energies. A number of 'growth rules' are incorporated in the procedure that alleviates the computational effort. Using this method, Hoare generated a large number of local minima for structures from 5 to 66 particles. However, no claim for complete enumeration of all local minima, and thus detection of the global minimum, can be made. In fact, it has been reported [32] that solutions of low-symmetry are not likely to be found with this method.

Piela's method [25] is based on the simple idea of smoothly deforming the potential energy hypersurface [29], in such a way as to make shallow potential wells disappear gradually, while the deeper ones grow at their expense. As the potential wells evolve they change their position and size. One then eventually ends up with a single potential well that has absorbed all the others which hopefully corresponds to the global minimum. A local optimization procedure then can easily find the single local minimum corresponding to the global one as well. The hypersurface is deformed using the diffusion equation, with the original shape of the hypersurface representing the initial concentration distribution. The main advantage of this method is that you do not have to explore the myriads of local optima, nor do you have to know their position beforehand. However, the approach depends on the conjecture that shallow potential wells disappear faster than deeper ones. In fact, it has been observed that when the global minimum lies on a narrow potential well of large depth, it might disappear faster than a wider, originally shallower, potential well.

Simulated annealing [18] variations has been widely used either alone, or in conjunction with some other method(s). A large number of researchers have been using this method for finding the global minimum of the potential energy function. Wille [32,33] solved the potential minimization problem for up to 25 particles, interacting under two-body Lennard–Jones forces and he found two new minima for $N = 24$ that were better than the one reported in [11]. P. Ballone and P. Milani [1] using a semi-empirical many-body potential, solved for the ground-geometries of carbon clusters in the range $50 \leq M \leq 72$ and found that all the structures of low energies are hollow spheres with nearly graphitic atomic arrangement. D. Hohl and R.O. Jones [17] applied the same methodology also to phosphorus clusters P_2 to P_8, arriving to arather counterintuitive most stable structure for P_8. In [23] a combined simulated annealing and a quasi-Newton-like conjugate-gradient method is used for determining the structure of mixed argon-xenon clusters interactingwith two-body Lennard–Jones forces. In [30,31] the binding energy of Nickel Lennard–Jones clusters is studied using the simulated annealing method in a canonical ensemble Monte-Carlo technique. The simulated annealing method can be viewed as a method for stochastically tracing the annealing process by Monte-Carlo simulation. D. Shalloway [27,28] presented a deterministic method for annealing the objective function by tracing the evolution of a multiple-Gaussian-packet approximation and using notions from renormalization group theory. This method has been applied to microcluster conformation problems and it appears that in most of the test problems was able to identify the global minimum.

Lattice optimization techniques have been very efficient in generating structures involving the lowest known potential energy. In [7] it is proposed that the most energetically favored microclusters in the range $20 \leq N \leq 50$ are the onesthat involve interpenetrating icosahedra (polyicosahedra) or (PIC). For $N \leq 55$

a double icosahedral (DIC) growth scheme was introduced [8] and for $55 \leq N \leq 147$ [9] a third layer icosahedral structure using two different surface arrangements was presented. Using these notions, J.A. Northby [24] derived optimal configurations for Lennard–Jones microclusters in the range $13 \leq N \leq 147$ based on a lattice optimization/relaxation algorithm. First a heuristic procedure is employed for finding a set of lattice local minimizers assuming icosahedral- (IC) or face-centered (FC) arrangements. Then, the currently best lattice minimizers are relaxed by using a local optimization algorithm. G.L. Xue [35] improved on Northby's method [24] by reducing the time complexity of the algorithm. Furthermore, by relaxing every lattice local minimizer a number of better optimal configurations were found in the range $13 \leq N \leq 147$. However, it appeared that the best local lattice does not always relax to the structure involving thelowest total Lennard–Jones potential energy. A parallel implementation [19] allowed results on minimum energies for clusters of up to $N = 1{,}000$ atoms. Also by employing a parallel version of a two-level simulated annealing algorithm [36,37,38] solutions for clustersizes as large as $N = 100{,}000$ have been reported.

C.D. Maranas and C.A. Floudas [20,21] introduced deterministic global optimization to the microcluster minimum potential energy problem. It was shown that the problem is convex only if both the first and second derivatives of the pairwise potential energy model with respect to the Euclidean distance are positive. This left only a narrow convex envelope for both Lennard–Jones and Morse potential energy models. To widen this envelope, the sum of squares of all Cartesian coordinates multiplied by a positive parameter α were added to the original objective function. It was shown that there exists a value for α such that the augmented objective function is convex. An upper bound for this value was identified. Based on these developments a branch and bound algorithm was devised based on the convex lower bounding of the objective function through the use of the α parameter. The algorithm was implemented for finding the global minimum configuration of small Lennard–Jones and Morse microclusters. For larger ones lower and upper bounds were derived by using a relaxation procedure. Later, these ideas sparked the development of the αBB algorithm for general nonconvex optimization problems.

See also

- ▶ Adaptive Simulated Annealing and its Application to Protein Folding
- ▶ Genetic Algorithms
- ▶ Global Optimization in Protein Folding
- ▶ Molecular Structure Determination: Convex Global Underestimation
- ▶ Monte-Carlo Simulated Annealing in Protein Folding
- ▶ Multiple Minima Problem in Protein Folding: αBB Global Optimization Approach
- ▶ Packet Annealing
- ▶ Phase Problem in X-ray Crystallography: Shake and Bake Approach
- ▶ Protein Folding: Generalized-ensemble Algorithms
- ▶ Simulated Annealing
- ▶ Simulated Annealing Methods in Protein Folding

References

1. Ballone P, Milani P (1990) Phys Rev 42:3905
2. Beck TL, Jellinek J, Berry RS (1987) J Chem Phys 87:545
3. Berry RS (1990) Scientif Amer
4. Boerdijk AH (1953) Philips Res Report 7:303
5. Conway JH, Sloane NJA (1988) Sphere packings, lattices and groups. Springer, Berlin
6. Coxeter HSM (1961) Introduction to geometry. Wiley, New York
7. Farges J, de Feraudy MF, Raoult B, Torchet G (1983) J Chem Phys 78:5067
8. Farges J, de Feraudy MF, Raoult B, Torchet G (1985) Surf Sci 156:370
9. Farges J, de Feraudy MF, Raoult B, Torchet G (1986) J Chem Phys 84:3491
10. Fejes-Toth L (1954) Regular figures. MacMillan, New York
11. Hoare MR (1979) Adv Chem Phys 40:49
12. Hoare MR, McInnes J (1972) Faraday Discuss Chem Soc 61:12
13. Hoare MR, McInnes J (1983) Adv Phys 32:791
14. Hoare MR, Pal P (1971) Nat Phys Sci 230:5
15. Hoare MR, Pal P (1971) Adv Phys 20:161
16. Hoare MR, Pal P (1972) J Crystallogr Growth 17:77
17. Hohl D, Jones RO, Car R, Parrinello M (1988) J Chem Phys 89:6823
18. Kirkpatrick S, Gelatt CD, Vecchi MP (1983) Science 220:671
19. Maier RS, Rosen JB, Xue GL (1992) Army High Performance Computing Res Center Preprint Univ Minnesota, 92–031
20. Maranas CD, Floudas CA (1992) J Chem Phys 97:10
21. Maranas CD, Floudas CA (1993) Ann Oper Res 42:85
22. Garey MR, Johnson DS (1979) Computers and intractability: A guide to the theory of NP-completeness. Freeman, New York

23. Navon IM, Brown FB, Robertson H (1990) Comput Chem 14:305
24. Northby JA (1987) J Chem Phys 87:6166
25. Piela L, Kostrowicki J, Scheraga HA (1989) J Phys Chem 93:3339
26. Pool R (1990) Science 24:1186
27. Shalloway D (1991) Recent advances in globaloptimization. Princeton Univ. Press, Princeton
28. Shalloway D (1992) J Global Optim 3:281
29. Stillinger FH, Weber TA 1429
30. Vlachos DG, Schmidt LD, Aris R (1992) J Chem Phys 96:6880
31. Vlachos DG, Schmidt LD, Aris R (1992) J Chem Phys 96:6891
32. Wille LT (1986) Nature 34:46
33. Wille LT (1987) Chem Phys Lett 133:405
34. Wille LT, Vennik J (1985) J Phys A 18:L419
35. Xue GL (1992) Army High Performance Computing Res Center Preprint Univ Minnesota
36. Xue GL (1992) Army High Performance Computing Res Center Preprint Univ Minnesota, 92-047
37. Xue GL (1997) J Global Optim 11:83
38. Xue GL, Maier RS, Rosen JB (1992) Internat. Conf. Supercomputing 6th ACM Internat. Conf. Supercomputing, Washington, DC, 19–23 July, p 409

Global Optimization in Location Problems

HOANG TUY
Institute of Mathematics, VAST, Hanoi, Vietnam

MSC2000: 90C26, 65D18, 90B85

Article Outline

A general mathematical problem encountered in various applications is to find the configuration of r unknown points in $\mathbb{R}^n$ (quite often $n \leq 3$) satisfying a number of constraints on their mutual distances and their distances to m fixed points, while minimizing a given function of these distances. Often the unknown points represent the locations of facilities to be constructed to serve the users located at the fixed points, so as to minimize a cost function (travel time, transport cost for customers, etc.) or to maximize the global attraction (utility, number of customers, etc.). Also the unknown points may represent the cluster centers while the fixed points are the objects to be classified into groups (clusters). The biggest challenge occurs when the unknown points represent the objects (atoms, particles) whose interactions depend upon their mutual distances: the objective function in these problems is then interpreted as a "potential energy function" that should attain a global minimum at the unknown configuration.

For many years, combinatorial geometric reasoning and nonlinear programming methods have been the basic tools in the study of these problems. However, since most nonconvex problems are characterized by the existence of many local nonglobal minimizers, other more suitable methods have to be used to efficiently cope with this difficulty.

Global optimization methods began to be introduced in these fields more than two decades ago [9,15]. Subsequently, dc optimization techniques were used to tackle facility location with nonconvex objective functions and nonconvex constraints [5,6,12,13,19,20,21].

Single Facility Location

The first location problem, introduced by Weber (1909), was to find the location of a facility so as to minimize the sum of its weighted distances to a given set of users located in a plane. Over the years this unconstrained convex minimization problem has been further and further generalized, leading to more and more complex models of location.

Minisum and Maxisum

Suppose a new facility is designed to serve m users located at $a^1, \ldots, > a^m \in \mathbb{R}^2_+$. Certain users, henceforth called the "attraction points," are interested in having the facility located as close to them as possible. Others, called the "repulsion points," would like the facility to be located as far away from them as possible. Let J_1, J_2 denote the index sets of attraction and repulsion points, respectively. For each user $j = 1, \dots, m$ a function $q_j(t)$ is known that measures the cost of traveling a distance t away from a^j; also, $h_j(x)$ is a function of the distance from user j to point $x \in \mathbb{R}^2$. It is assumed that the function $q_j(t)$ is *concave increasing* with $q_j(t) \to +\infty$ as $t \to \infty$, while $h_j(x)$ is a convex function such that $h_j(x) \to +\infty$ as $\|x\| \to +\infty$. So if x is the unknown location of the facility, then to take account of the interest of attraction points, one should try to minimize the sum $\sum_{j \in J_1} q_j(h_j(x))$, whereas from the point of view of repulsion points one should try to maximize the sum $\sum_{j \in J_2} q_j(h_j(x))$. Under these conditions, a reasonable objective of the decision maker may be to locate the facility so as to minimize the quantity

$$\sum_{j \in J_1} q_j(h_j(x)) - \sum_{j \in J_2} q_j(h_j(x))$$

over $\mathbb{R}^n_+$. Denoting the right derivative of $q_j(t)$ at 0 by $q_j^+(0)$ and assuming $q_j^+(0) < +\infty \forall j$, it can easily be seen that each function $g_j(x) := K_j h_j(x) + q_j[h_j(x)]$ is convex for $K_j \geq q_j^+(0)$, and so we come up with the dc optimization problem

$$\min\{G(x) - H(x) | x \in \mathbb{R}^n_+\} , \qquad (1)$$

where $G(x)$, $H(x)$ are convex functions defined by

$$G(x) = \sum_{j \in J_2} g_j(x) + \sum_{j \in J_1} K_j h_j(x) ,$$

$$H(x) = \sum_{j \in J_1} g_j(x) + \sum_{j \in J_2} K_j h_j(x) .$$

Problems with the above objective function are called *minisum* problems.

In other circumstances, instead of minimizing the cost, one may seek to maximize the total attraction

$$\sum_{j \in J_1} q_j[h_j(x)] - \sum_{j \in J_2} q_j[h_j(x)] ,$$

where each q_j is a *convex decreasing* function. Assuming $q_j^+(0) > -\infty$, the problem is then

$$\max\{\tilde{G}(x) - \tilde{H}(x) | x \in \mathbb{R}^n_+\} , \qquad (2)$$

where $\tilde{G}(x)$, $\tilde{H}(x)$ are now the convex functions

$$\tilde{G}(x) = \sum_{j \in J_1} g_j(x) + \sum_{j \in J_2} K_j h_j(x) ,$$

$$\tilde{H}(x) = \sum_{j \in J_2} g_j(x) + \sum_{j \in J_1} K_j h_j(x) .$$

Obviously, any maxisum problem can be converted into a minisum one and vice versa. Most problems studied in the literature are minisum, under much more restricted assumptions than in the above setting (see [16] and references therein). Weber's classical formulation corresponds to the case $J_2 = \varnothing$ (no repulsion points) and $h_j(x) = \|x - a^j\|, q_j(t) = w_j t, w_j \geq 0, \forall j.$ The cases $J_2 \neq \emptyset$ with $q_j(t)$ nonlinear have begun to be investigated only recently, motivated by growing concerns about the environment.

Maximin and Minimax

When siting emergency services, like a fire station, one does not want to maximize the overall attraction but rather to guarantee for every user a minimal attraction as large as possible. The problem, often referred to as the *p-center problem*, can be formulated as

$$\max\left\{\min_{j=1,\dots,m} q_j[h_j(x)] | x \in \mathbb{R}^n_+\right\} , \qquad (3)$$

where $q_j(t)$ are *convex decreasing* functions (minimax problem). Assuming $|q_j^+(0)| < \infty \forall j$ as previously, we have the dc representation $q_j[h_j(x)] = g_j(x) - K_j h_j(x)$, hence

$$\min_{j=1,\dots,m} q_j[h_j(x)] = \sum_{j=1}^{n} g_j(x) - \max_{j=1,\dots,n}\left[K_j h_j(x) + \sum_{i \neq j} g_i(x)\right] ,$$

and so (3) is again a dc optimization problem.

By contrast, when siting an obnoxious facility, one wants to minimize the maximal attraction to an user, so the optimization problem to be solved is

$$\max\left\{\min_{j=1,\dots,k} q_j[h_j(x)] | x \in \mathbb{R}^n_+\right\} , \qquad (4)$$

where $q_j(t)$ are *concave increasing* functions (minimax problem). Again, assuming $|q_j^+(0)| < \infty \forall j$, we have the dc representation $q_j[h_j(x)] = K_j(x) - g_j(x)$, and so

$$\max_{j=1,\dots,m} q_j[h_j(x)] = \max_{j=1,\dots,n}\left[K_j h_j(x) + \sum_{i \neq j} g_j(x)\right] - \sum_{j=1}^{m} g_j(x) ,$$

i. e., the minmax location problem (4) is again a dc optimization problem.

A special maximin location problem worth mentioning is the *design centering* problem encountered in engineering design. Given a compact convex set $B \subset \mathbb{R}^n$ containing 0 in its interior and m compact convex sets D_j, $j = 1, \dots, m$ contained in a compact convex set $C \subset \mathbb{R}^n$, find $x \in C$ so as to maximize

$$r(x) = \min_{j=0,1,\dots,m} r_j(x)\,,$$

where $r_j(x) = \min\{p(y - x)\colon y \in D_j\}$, $p\colon \mathbb{R}^n \to \mathbb{R}_+$ is the gauge of B and $D_0 = \mathbb{R}^n \setminus C$. It can be shown [17] that the function $r_0(x)$ is concave while $r_1(x), \dots, r_m(x)$ are convex, so this can be viewed as a maximin problem in which each D_j is a user and $r_j(x)$ is the distance from point x to user j.

Constrained Location

In the real world many factors may set restrictions on the facility sites. Therefore, practical location problems are often constrained.

Location on Union of Convex Sets

The most simple type of restriction is that the facility can be located only in one of several given convex regions $C_1, \dots, C_k$ [8]. If $C_i = \{x\colon c_i(x) \le 0\}$, with $c_i(x)$ being convex functions, then the constraint $x \in \cup_{i=1}^{k} C_i$ can be expressed as

$$\min_{i=1,\dots,k} c_i(x) \le 0\,,$$

which is a dc constraint.

Location on Area with Forbidden Regions

In other circumstances, the facility can be located only outside some forbidden regions that are, for instance, open convex sets $C_i^o = \{x\colon c_i(x) < 0\}$, with $c_i(x)$ being convex functions (see, e. g., [2]). Since the constraint $x \notin \cup_{i=1}^{k} C_i^o$ is equivalent to $\min_{i=1,\dots,k} c_i(x) \ge 0$, this is again a dc constraint.

General Constrained Location Problem

The most general situation occurs when the constraint set is a compact, not necessarily convex, set. However, a striking result of dc analysis shows that even in this general case the constraint can be expressed as a dc inequality [12,22].

Of course the corresponding dc optimization problem is very hard. Although a method (the relief indicator method [18]) exists for dealing with general nonconvex constraints, so far it only works in low dimension.

Multiple Source

When more than one facility is to be located, the objective function depends upon whether these facilities provide the same service or different services to the users.

If there are $r \ge 2$ facilities providing the same service, these facilities are called *sources*. Each user is then served by the closest source. So if x^i is the unknown location of the ith facility and $X = (x^1, \dots, x^r) \in (\mathbb{R}^2)^r$, then the overall attraction is

$$\sum_{jinJ_1} q_j[\tilde{h}_j(X)] - \sum_{j\in J_2} q_j[\tilde{h}_j(X)]\,, \tag{5}$$

where $\tilde{h}_j(X) = \min\{h_j(x^i)\colon i = 1, \dots, r\}$ and q_j, h_j are as previously. Since $\tilde{h}_j(X) = \sum_{i=1}^{r} h_j(x^i) - \max_{i=1,\dots,r} \sum_{i\neq l} h_j(x^i)$, the first term in (5) is the dc function

$$\sum_{j\in J_1} g_j(X) - \sum_{j\in J_1} K_j\left[\sum_{i=1}^{r} h_j(x^i) + \max_{l=1,\dots,r} \sum_{i\neq l} h_j(x^i)\right],$$

where $K_j \ge |q_j^+(0)|$ and

$$g_j(X) = q_j[\tilde{h}_j(X)] + K_j\left[\sum_{i=1}^{r} h_j(x^i) + \max_{l=1,\dots,r} \sum_{i\neq l} h_j(x^i)\right]$$

is a convex function. Similarly for the second term in (5). Hence the objective function in the r source problem is a dc function on $(\mathbb{R}^2)^r$.

The multisource problem is usually referred to as the *generalized Weber problem*, or also the *r-median problem* when $J_2 = \emptyset$. Traditionally it is often viewed as a location-allocation problem and formulated as a mixed 0-1 integer programming problem (see, e. g., [16]).

Clustering

In many practical situations we have a set of objects of a certain kind that we want to classify into $r \ge 2$ groups (clusters), each including elements close to each other in some well-defined sense. In the simplest case, this gives rise to the following problem: for a given finite set of points $a^1, \dots, a^m \in \mathbb{R}^n$, find r cluster centers $x^i \in \mathbb{R}^n, i = 1, \dots, r$, such that the sum of the minima over $i \in \{1, \dots, r\}$ of the "distance" between each point a^j and the cluster centers x^i, $i = 1, \dots, r$, is minimized. If $d(a, x)$ denotes the distance from a to x, then the problem is

$$\min\left\{ \sum_{j=1}^{m} \min_{i=1,\dots,r} d(a^j, x^i) \colon x^i \in [0, b] \right\} . \qquad (6)$$

Formally, this is nothing but the r-median problem, i. e., the generalized Weber problem with $J_2 = \emptyset$.

If $d(a, x) = \sum_{i=1}^{n} |a_i - x_i|$, then, using the equality $|a_i - x_i| = \min\{y_i \colon -y_i \le a_i - x_i \le y_i\}$, problem (6) can be written as

$$\begin{aligned} &\sum_{j=1}^{m} \min_{l=1,\dots,r} \left(\sum_{i=1}^{n} y_i^{jl} \right) \\ &-y^{jl} \le a^j - x^l \le y^{jl} \\ &j = 1, \dots, m, l = 1, \dots, r , \end{aligned}$$

which is a concave minimization problem under linear constraints. One way to cope with the large dimension of this problem is to replace it with the equivalent bilinear program

$$\begin{aligned} &\min \sum_{j=1}^{m} \sum_{l=1}^{r} t_{jl} y^{jl} \\ &\text{s.t.} - y^{jl} \le a^j - x^l \le y^{jl} \quad j = 1, \dots, m, l = 1, \dots, r \\ &\quad \sum_{l=1}^{r} t_{jl} y^{jl}, \; t_{jl} \ge 0, \sum_{l=1}^{r} t_{jl} = 1 . \end{aligned}$$

and to solve the latter approximately to a local optimum by alternately fixing t and y.

When $d(a, x) = \sqrt{\sum_{i=1}^{n} (a_i - x_i)^2}$, the problem is no longer a concave minimization but can be reduced to a dc program by easy manipulations. In [1] results of solving the generalized Weber problem with $m = 10{,}000$, $p = 2$, and $m = 1{,}000$, $p = 3$, by dc methods are reported. Alternatively, (6) can also be transformed into a monotonic optimization and solved by recently developed monotonic optimization methods [23,24]. For this observe that $d(a, x) = (d(a, x) + \sum_{i=1}^{n} x_i) - \sum_{i=1}^{n} x_i$, and since $u(a, x) = d(a, x) + \sum_{i=1}^{n} x_i$ and $\sum_{i=1}^{n} x_i$ are both increasing functions, it follows that $d(a,x)$ is a dm (*d*ifference of *m*onotonic) function, and, hence, (6) is a monotonic optimization problem.

Multiple Facility

When the $r \ge 2$ facilities to be located provide different services, aside from the costs due to interactions between facilities and users, one should also consider the costs due to pairwise interactions between facilities. The latter costs can be expressed by functions of the form $\rho_{il}[h_{il}(x^i, x^l)]$, where again $h_{il}(x^i, x^l)$ are convex nonnegative valued functions and $\rho_{il}(t)$ are concave increasing functions on $[0, +\infty)$ with finite right derivatives at 0. The total cost one would like to minimize is then

$$\sum_{i=1}^{r} F_i(x^i) + \sum_{i<l} \rho_{il}[h_{il}(x^i, x^l)] , \qquad (7)$$

where $F_i(x^i) = \sum_{j \in J_1} q_{ji}[h_j(x^i)] - \sum_{j \in J_2} q_{ji}[h_j(x^i)]$ and q_{ji}, h_j are as in minisum single facility problems.

As we saw above, each function $F_i(x^i)$ is dc, hence each function $\rho_{il}[h_{il}(x^i, x^l)]$ is dc, too, and (7) is again a dc function on $(\mathbb{R}^2)^r$. In the special case when there are no repulsion points (every $F_i(x^i)$ is convex) and the pairwise interactions between facilities $\rho_{il}(t)$ are convex, this is simply a convex function. Also, in the absence of interactions between facilities ($\rho_{ij}(.) = 0 \; \forall ij$), the minimization of function (7) splits into r independent single facility minisum problems.

Molecular Conformation

A variant of the multifacility problem that has risen to attract much research in recent years is the so-called *molecular conformation* problem encountered in computational biology, computational chemistry, and protein folding. This is the problem of determining ground states or stable states of certain classes of molecular clusters and proteins and can be stated as follows [14]. Given a cluster of N atoms (in three-dimensional space), we wish to locate their centers $x^1, \dots, x^N$ so as to minimize the potential energy function

$$V_N(x^1, \dots, x^N) = \sum_{1 \le i < j \le N} v(\|x^i - x^j\|) ,$$

where $\|.\|$ is the euclidean norm and $v(r)$ the interatomic pair potential. This can be viewed as a multifacility problem in which there is no user but many facilities (the number N may be rather large; see, e. g., [14]). In models used for computation, the pair potentials of interest include the following:

$$v(r) = r^{-12} - 2r^{-6} \text{ (Lennard-Jones)} ,$$
$$v(r) = \left[1 - e^{\alpha(1-r)}\right]^2 - 1 \text{ (Morse)} ,$$
$$v(r) = \frac{z^\alpha z^\beta}{r} + Ae^{\frac{-r}{\rho}} \text{ (Born-Meyer)} .$$

Using representation theorems in dc optimization, it can be seen that these functions are dc (at least for $r \geq \varepsilon$, where ε is an arbitrary small positive number).

Distance Geometry

A related problem that also has applications in molecular conformation, and other questions such as surveying and satellite ranging, data visualization, and pattern recognition, etc., is the *multidimensional scaling problem* or *distance geometry problem*. It consists in finding r objects $x^1, \ldots, x^r$ in $\mathbb{R}^n$ such that the quantity

$$V_r(x^1, \ldots, x^r) = \sum_{i<j} w_{ij} \left(\delta_{ij}^2 - \|x^i - x^j\|^2\right)^2 \quad (8)$$

is smallest, where $\Delta = (\delta_{ij})$, $W = (w_{ij})$ are symmetric matrices of order r such that

$$\delta_{ij} = \delta_{ji} \geq 0, \quad w_{ij} = w_{ji} \geq 0 \quad (i < j);$$
$$\delta_{ii} = w_{ii} = 0 \quad (i = 1, \ldots, r) .$$

By writing this problem as

$$\begin{aligned} \min \quad & \sum_{i<j} w_{ij} \|x^i - x^j\|^2 - 2 \sum_{i<j} w_{ij}\delta_{ij} \|x^i - x^j\| \\ \text{s.t.} \quad & x^i \in \mathbb{R}^n (i = 1, \ldots, r) \end{aligned} \quad (9)$$

or, alternatively, as

$$\min \sum_{i,j} w_{ij} t_{ij}^2 \left| \begin{array}{l} -t_{ij} \leq \delta_{ij}^2 - \|x^i - x^j\|^2 \leq t_{ij} \ (\forall i < j) \\ x^i \in \mathbb{R}^n, \quad i = 1, \ldots, r \end{array} \right. \quad (10)$$

we again obtain a dc optimization problem that is also a monotonic optimization problem.

References

1. Al-Khayyal FA, Tuy H, Zhou F (2002) Large-Scale Single Facility Continuous Location by D.C. Optimization. Optimization 51:271–292
2. Aneja YP, Parlar M (1994) Algorithms for Weber facility location in the presence of forbidden regions and/or barriers to travel. Transp Sci 28:70–216
3. Chen R (1983) Solution of minisum and minimax location-allocation problems with euclidean distances. Nav Res Logist Q 30:449–459
4. Chen R (1988) Conditional minisum and minimax location-allocation problems in Euclidean space. Transp Sci 22:157–160
5. Chen P, Hansen P, Jaumard B, Tuy H (1992) Weber's problem with attraction and repulsion. J Reg Sci 32:467–409
6. Chen P, Hansen P, Jaumard B, Tuy H (1998) Solution of the multifacility Weber and conditional Weber problems by D.C. Programming. Oper Res 46:548–562
7. Dresner Z (ed) (1995) Facility Location: A Survey of Applications and Methods. Springer, Berlin
8. Hansen P et al (1982) An Algorithm for a Constrained Weber Problem. Manage Sci 28:1285–1295
9. Hansen P et al (1985) The Minisum and Minimax Location-Probems Revisited. Oper Res 33:1251–1265
10. Horst R, Tuy H (1996) Global Optimization, 3rd edn. Springer, Berlin
11. Idrissi H, Loridan P, Michelot C (1988) Approximation of Solutions for Location Problems. J Optim Theory Appl 56:127–143
12. Konno H, Thach PT, Tuy H (1997) Optimization on Low Rank Nonconvex Structures. Kluwer, Dordrecht
13. Maranas CD, Floudas CA (1993) A global Optimization method for Weber's problem with attraction and repulsion. In: Hager WW, Heran DW, Pardalos PM (eds) Large Scale Optimization: State of the Art. Kluwer, Dordrecht, pp 1–12
14. Maranas CD, Floudas CA (1994) Global minimum potential energy conformations of small molecules. J Global Optim 4:135–171
15. Plastria F (1992) The generalized big square small square method for planar single facility location. Eur J Oper Res 62:163–174
16. Plastria F (1995) Continuous location problems. In: Dresner Z (ed) Facility Location: A Survey of Applications and Methods. Springer, Berlin, pp 225–262
17. Thach PT (1988) The design centering problem as a dc programming problem. Math Programm 41:229–248
18. Thach PT, Tuy H (1990) The relief indicator method for constrained global optimization. Naval Res Logist 37:473–497
19. Tuy H, Al-Khayyal FA (1992) Global Optimization of a Nonconvex Single Facility Problem by Sequential Unconstrained Convex Minimization. J Global Optim 2:61–71

20. Tuy H (1996) A General D.C. Approach to Location Problems. In: Floudas CA, Pardalos PM (eds) State of the Art in Global Optimization. Kluwer, Dordrecht, pp 413–432
21. Tuy H, Al-Khayyal FA, Zhou F (1995) D.C. optimization method for single facility location problem. J Global Optim 7:209–227
22. Tuy H (1998) Convex Analysis and Global Optimization. Kluwer, Dordrecht
23. Tuy H (2000) Monotonic Optimization: Problems and Solution Approaches. SIAM J Optim 11:464–494
24. Tuy H, Minoux M, Hoai Phuong NT (2006) Discrete Monotonic Optimization with Application to A Discrete Location Problem. SIAM J Optim 17:78–97

Global Optimization Methods for Harmonic Retrieval

William Edmonson[1], Wen Lee[2]
[1] Hampton University, Virginia, USA
[2] University Florida, Gainesville, USA

MSC2000: 90C26, 65T40, 90C30, 90C90

Article Outline

Keywords

Global optimization; Harmonic retrieval; Expectation maximization; Interval methods

The *harmonic retrieval* (HR) problem is an ubiquitous problem that arises in various applications, such as signal modeling and direction-of-arrival. It consists of estimating the parameters of multiple sinusoids from noisy data. The data is modeled as

$$y(t) = \sum_{k=1}^{K} a_k^* \sin(2\pi f_k^* t) + n(t), \quad t = 1, \dots, N, \qquad (1)$$

where a_k^*, f_k^*, and ϕ_k^* are the amplitude, frequency, and phase of the kth sinusoid, respectively. It is assumed that the number of sinusoids, K, is known and all frequencies satisfy $0 < f_k^* < 0.5$, $k = 1, \dots, K$, and $f_k^* \neq f_j^*$ for $k \neq j$. In addition, the noise, $n(t)$, is assumed to be zero-mean, white Gaussian noise (WGN) with variance σ_n^2. Given the data, $y(t)$ for $t = 1, \dots, N$, the goal is to estimate the sinusoid parameters, $\theta^* = [a_1^*, \dots, a_K^*, f_1^*, \dots, f_K^*]$.

The conventional FFT or periodogram-based methods [4, Chapt. 1]are only able to solve the HR problem when frequencies are spaced more than $1/N$ cycles/sample apart, where N is the number of available data points. To tackle the problem where the difference between any two frequencies is smaller than the threshold $1/N$, high resolution techniques must be used [4, Chapt. 5]. The *sinusoidal parameter estimation problem* is based on solving the *least squares* (LS) problem (P):

$$(P) \quad \widehat{\theta}_{\mathrm{LS}} \triangleq \arg\min_{\theta} J(\theta), \qquad (2)$$

where

$$J(\theta) = \sum_{t=1}^{N} \left\{ y(t) - \sum_{k=1}^{K} a_k \sin(2\pi f_k t + \phi_k) \right\}^2, \qquad (3)$$

and $\theta = [a_1, \dots, a_K, f_1, \dots, f_K, \phi_1, \dots, \phi_K]$. We can see from (3) that the objective function is nonconvex, which suggests that a global optimization method represents the most appropriate procedure for determining $\widehat{\theta}_{\mathrm{LS}}$.

Two methods that have been proposed for solving (P) are the one proposed in [8], for which we will refer to as *Stoica's method* and the *Iterative Quadratic Maximum Likelihood method* (briefly: IQML method) [1]. Both methods can not guarantee convergence unless the initial conditions are sufficiently close to the global minimum. Stoica's method first generates initial estimates using the *overdetermined Yule–Walker method*. Then, it improves on these estimates by using a periodogram-based procedure and a simplified Gauss–Newton algorithm to iteratively maximize the likelihood function. In [8], it was shown experimentally that Stoica's method requires extremely large data records. The well known IQML method is an iterative quadratic maximization algorithm that attempts to determine the *maximum likelihood* (ML) estimates in terms of a prediction polynomial. This algorithm, as our experiments show, produces poor estimates for short data records and/or low signal-to-noise ratio (SNR). The IQML algorithm is also noted to sometimes fail to converge and the estimated frequencies are almost always inconsistent [9].

Taking a different approach, we will apply the *global optimization* algorithm of *interval methods* (IM) to the HR problem (2). Interval method type algorithms [3,6,7] have proven to be an excellent and reliable procedure for solving global optimization problems involving nonconvex objective functions. One of the reasons one chooses interval methods is because they are applicable to most optimization problems regardless of convexity and differentiability of the objective function, or knowledge of its Lipschitz constant. Additionally, for *continuous* objective functions, its convergence to a global optimum interval has been proven [3]. In using the IM method for solving the LS estimates of (2), convergence is very slow.

One way to overcome the problem of slow convergence is to decompose the problem whereby optimization occurs over smaller dimensions and in parallel. This can be accomplished through combining the *expectation-maximization algorithm* (briefly: *EM algorithm*) [2] with the interval method. This proposed combination of the EM algorithm with the interval method is defined as the *expectation-maximization interval* method (EMIM) algorithm. The EM algorithm represents a computationally efficient method for solving estimation problems. For the HR problem, the EM algorithm decomposes the HR problem into K subproblems, where K is the number of sinusoids. The K subproblems, which are nonconvex optimization problems, are then solved using an IM global optimization method. This results in an algorithm that is able to converge to the global minimum interval with significantly reduced computational complexity, in comparison with using the IM algorithm alone for solving (P).

Interval Arithmetic

Interval methods are a class of global optimization algorithms that utilize *interval arithmetic*. An interval which contains the global minimum is found by partitioning the search space into regions, where at each iteration, regions are selected for further search by additional partitioning. Those partitions that cannot contain the global minimum are discarded. A major advantage of interval methods is their ability to find the global minimum of nonconvex differentiable or nondifferentiable objective functions.

Interval arithmetic [6] was developed to automatically estimate and control numerical errors caused by finite precision of computer arithmetic. The *INTLIB* library [5] is used to implement interval arithmetic as used in the IM algorithm. A real interval number $X = [a, b]$ consists of the set set$\{x\colon a \leq x \leq b\}$ of real numbers. Additional notations used here are: the upper bound (ub) of $X = b$, the lower bound (lb) of $X = a$, the mid-point of X is $m(X) = (a + b)/2$, and the width of X is $w(X) = b - a$. Furthermore, $w(\mathbf{X}) = \max\{w(X_i)\}_{i=1}^{i=n}$ where $\mathbf{X} = [X_1, \ldots, X_n]^{\mathsf{T}}$. The general interval arithmetic operational rules is defined as $X \Box Y = \{x \Box y \colon x \in X, y \in Y\}$, where X and Y are real interval numbers and $\Box$ represents the arithmetic operations of plus, minus, multiplication, and division. For additional information on interval arithmetic see [6,7].

The *unconstrained global optimization* problem can be described as

$$\min_{x \in D} g(x), \tag{4}$$

where $g(x)\colon \mathbf{R}^n \leftarrow \mathbf{R}$, $x \in \mathbf{R}^n$ and $D \in \mathbf{R}^n$ represents the feasible region. The main tool for solving the problem in (4) is the concept of *inclusion function*. A function $G(X)\colon \mathfrak{I}^n \rightarrow \mathfrak{I}$ is an inclusion function of the objective function $g(x)$, if $x \in Y$ implies that $g(x) \in G(Y)$ and that the *isotonicity property* is met (i. e. $X \subseteq Y$ implies that $F(X) \subseteq G(Y)$). The inclusion function with isotonicity property provides the theory for the use of interval methods as a global optimization procedure. In short, inclusion functions represent the range of function values of f over the interval X.

The optimization procedure for the interval method involves continually bisecting a box X_i from an initial box, X_0, until $G(X^i)$, the inclusion function, contains the global minimum given that $w(G(X^i)) \leq \epsilon$. What differentiates this method from the method of exhaustive search is that regions of the objective are discarded from evaluation if the lb $G(X^i)$ in the list, $\mathcal{L}$, is greater than the minimum between the past or present value of ub $G(X^j)$ given that $i \neq j$. The algorithm of E.R.

Global Optimization Methods for Harmonic Retrieval, Table 1
A pseudocode for interval methods

```
PROCEDURE interval method
    Set $Y := X$;
    Calculate $G(Y)$, $y :=$lb$G(Y)$, $\tilde{g} :=$ub $G(m)$
        where $m =$mid $Y$
        Initialize list $\mathcal{L} := \{(Y, y)\}$.
    REPEAT until convergence
        Choose a coordinate direction $k$, parallel to $Y_i$,
        and of max length.
        Bisect $Y$ to obtain boxes $V_1$, $V_2$, where
        $Y = V_1 \cup V_2$.
        Calculate $G(V_1)$ and $G(V_2)$ and $v_i :=$lb $G(V_i)$
        for $i = 1,2$.
        Place $(G(V_i), v_i)$ at end of list.
        Choose pair $(\tilde{Y}, \tilde{y})$ from $\mathcal{L}$ such that $\tilde{y} \leq z$,
        $\forall (Z, z)$.
        Discard pairs from list, $(Z, z)$, if $z > \tilde{g}$.
        Terminate if $\omega(Z) < \epsilon$, $\forall Z$, in $\mathcal{L}$.
        Denote first pair of list by $(Y, y)$.
        Compute $m :=$ mid $Y$ and
        $\tilde{g} = \min(\tilde{g}$, ub $G(m))$.
    RETURN
END interval method
```

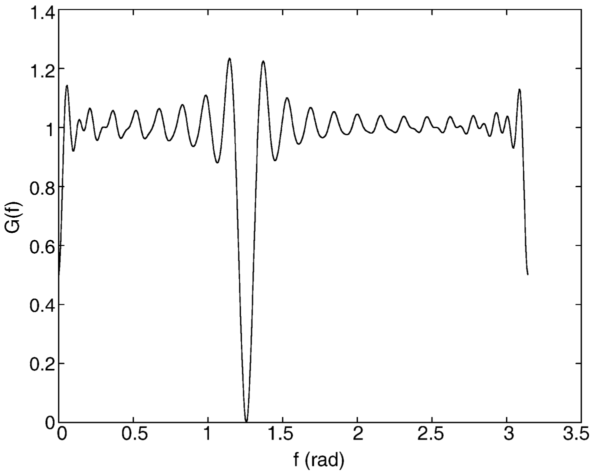
Hansen [3,7] is the particular interval method that will be used for locating the LS estimates of the HR problem and is outlined in Table 1. In [7], it was proven that convergence to the global minimum was achieved if $w(G(X)) \to 0$ as $w(X) \to 0$.

Interval Method for Solving HR

To apply the IM to solving the HR problem, the objective function (3) must be placed in its inclusion form:

$$\mathbf{J}(\Theta) = \sum_{t=1}^{N} \left[y(t) - \sum_{k=1}^{K} A_k \sin(2\pi F_k t + \Phi_k) \right]^2 , \qquad (5)$$

where $\Theta = [A_1, \ldots, A_K, F_1, \ldots, F_K, \Phi_1, \ldots, \Phi_K]$ and A_k, F_k, and Φ_k are the interval counterparts of a_k, f_k, and ϕ_k, respectively. Throughout this paper capital letters represent interval variables that correspond to its real variable equivalent. The initial interval, Θ_0, is chosen such that it encompasses the global minimum. This is accomplished by choosing an interval that is determined from a priori information or from other high resolution HR methods [4]. The IM of Hansen's, described in previous section, is used to determine the global minimum, Θ^*, of (5). The objection function (5) for a single frequency, phase and amplitude held constant, is plotted in Fig. 1). It can easily see that this represents a very difficult but practical problem for global optimization.



Global Optimization Methods for Harmonic Retrieval, Figure 1
Objective function of a single sinusoid

Simulations

In this section, a numerical experiment will be demonstrated to show the performance of the IM for solving the HR problem (P). The experiments consist of estimating the sinusoid parameters for the following data,

$$\begin{aligned} &y(t) = 1.0 \sin(2\pi(0.2)t + 0.0) + n(t), \\ &t = 1, \ldots, 35, \end{aligned} \qquad (6)$$

where $n(t)$ is white Gaussian noise. We choose the initial box for the IM algorithm to be $\Theta = [A, F, \Phi]^{\mathsf{T}} = [[0.71.2], [0.10.3], [00.4]]^{\mathsf{T}}$. The signal-to-noise-ratio (SNR) is defined as

$$10 \log \left[\sum_{k=1}^{K} 0.5 \frac{(a_k^*)^2}{\sigma_n^2} \right] ,$$

where σ_n^2 is the variance of the noise. The results of this simulation, shown in Table 2, is described in terms of sample mean and standard deviation based on 50

Global Optimization Methods for Harmonic Retrieval, Table 2 IM estimates

IM: $N = 35$ (50 MC runs)			
SNR	10	5	0
$a^* = 1.0$	1.0155 ± 0.0518	1.0447 ± 0.0913	1.0633 ± 0.1365
$f^* = .20$	0.1995 $\pm 6.591 \cdot 10^{-4}$	0.1993 ± 0.0012	0.1989 ± 0.0017
$\phi^* = 0$	0.0654 ± 0.0564	0.0839 ± 0.0975	0.1193 ± 0.1319

Global Optimization Methods for Harmonic Retrieval, Table 3 IQML Estimates

IQML: $N = 35$ (50 MC runs)			
SNR	10	5	0
$a^* = 1.0$	1.0080 ± 0.0533	0.9862 ± 0.1479	0.8919 ± 0.3230
$f^* = .20$	0.1998 $\pm 7.623 \cdot 10^{-4}$	0.1970 ± 0.0202	0.1728 ± 0.0839
$\phi^* = 0$	0.0141 ± 0.0949	-0.0126 ± 0.2852	0.5288 ± 1.1429

Monte-Carlo (MC) runs. This results are based on the midpoints of Θ. Note that the final estimates, $\widehat{\theta}$, are very close to the true value of θ^* with a small standard deviation. In comparison with IQML, see Table 3, the IM fares considerably better in both mean and standard deviation. This is particularly notable when comparing the frequency component, which represents the most important feature of harmonic retrieval.

The convergence rate of the IM is sensitive to the order, K, of the HR problem. In fact, the dimensionality of the parameter space, $\mathcal{J}^n$, increases at a rate of $3K$. Thus, as n increases, the convergence rate becomes prohibitively slow. The *curse of dimensionality* can be mitigated through decomposition and parallelizing the problem by utilizing the EM algorithm as described in the next section.

EMIM

The detailed development of the EM algorithm [2] is well-known, and will be outlined here as part of the development of the EMIM algorithm for solving the HR problem. To determine the LS estimates of the sinusoidal parameters, the EM algorithm first decomposes the observed data $y(t)$ into its signal components (E step) and then estimates the parameters of each signal component separately (M step). The algorithm iterates back and forth between the E step and M step, using the current estimate to decompose the observed data better and thus improve the next parameter estimate.

For the HR problem, the incomplete data is the observed data, $y(1), \ldots, y(N)$. The complete data is modeled as the following K data records:

$$y_k(t) = a_k^* \sin(2\pi f_k^* t + \phi_k^*) + n_k(t),$$
$$k = 1, \ldots, K,$$

where $n_k(t) = \beta_k[y(t) - \sum_{k=1}^K a_k^* \sin(2\pi f_k^* t + \phi_k^*)]$. The β_k's are arbitrary real-valued scalars satisfying $\sum_{k=1}^K \beta_k = 1$ and $\beta_k \geq 0$. Thus $\sum_{k=1}^K n_k(t) = n(t)$, for $t = 1, \ldots, N$. The EM algorithm, beginning with $n = 0$, is represented by the following two steps:

E) For $k = 1, \ldots, K$, compute

$$\begin{aligned}\widehat{y}_k^{(n)}(t) &= \widehat{a}_k^{(n)} \sin(2\pi \widehat{f}_k^{(n)} t + \widehat{\phi}_k^{(n)}) \\ &+ \beta_k \left[y(t) - \sum_{l=1}^{K} \widehat{a}_l^{(n)} \sin(2\pi \widehat{f}_l^{(n)} t + \widehat{\phi}_l^{(n)}) \right].\end{aligned} \tag{7}$$

M) For $k = 1, \ldots, K$,

$$\widehat{\theta}_k^{(n+1)} = \arg \min_{a_k, f_k, \phi_k} J_k^{(n)}, \tag{8}$$

where

$$J_k^{(n)} = \sum_{t=1}^{N} (\widehat{y}_k^{(n)}(t) - a_k \sin(2\pi f_k t + \phi_k))^2. \tag{9}$$

The parameter vector $\widehat{\theta}_k^{(n)} \triangleq [\widehat{a}_k^{(n)}, \widehat{f}_k^{(n)}, \widehat{\phi}_k^{(n)}]^\top$ is the estimate for $\theta_k^* \triangleq [a_k^*, f_k^*, \phi_k^*]^\top$ after n iterations. In the original HR problem, we have to search the $(3 \times K)$-dimensional parameter space to find the minimum value of the least squares objective function. But after the EM algorithm decomposes the HR problem into K smaller subproblems, we only have to solve K subproblems each of which requires the search of a 3-dimensional parameter space to find the global optimal point(s). This results in a significant reduction in computational complexity.

To solve the minimization problem in M step, we resort to using the IM for finding the final interval that contains the point minimizing the objective function. Since IM has been proven to converge to the global optimum for *continuous* objective functions [3], this algorithm will not be trapped in the local extremum. Needed in the IM algorithm is the inclusion function of the objective function (6), which is constructed by forming the natural interval extension [3,7] of J_k:

$$\mathcal{J}_k^{(n)} = \sum_{t=1}^{N} \left(\widehat{\gamma}_k^{(n)}(t) - A_k \sin(2\pi F_k t + \Phi_k) \right)^2 , \quad (10)$$

where A_k, F_k, and Φ_k are the interval counterparts of a_k, f_k, and ϕ_k, respectively. The initial value $\widehat{\theta}_k^{(0)} = [\widehat{a}_k^{(0)}, \widehat{f}_k^{(0)}, \widehat{\phi}_k^{(0)}]^\top$ are arbitrarily guessed or can come from other high-resolution estimation methods. The initial interval $\Theta_{k,0} = [A_{k,0}, F_{k,0}, \Phi_{k,0}]^\top$ for the M) step is the region over which the minimization is carried out. This initial interval $\Theta_{k,0}$ is used at the beginning of each M) step of the EMIM algorithm. At the $(n+1)$st iteration of EMIM, the IM partitions $\Theta_{k,0}$ iteratively to find the final interval estimate $\widehat{\Theta}_k^{(n+1)}$. The $m(\widehat{\Theta}_k^{(n+1)}) = \widehat{\theta}_k^{(n+1)}$ will be used as the parameter estimate to compute $\widehat{\gamma}_k^{(n+1)}(t)$ for the next iteration of the EMIM algorithm. The process is repeated until $\sum_{k=1}^{K} \left\| \widehat{\theta}_k^{(n+1)} - \widehat{\theta}_k^{(n)} \right\| \leq \rho$, where ρ is chosen by the user.

Consider the case where $\widehat{\theta}_i^{(0)} = \widehat{\theta}_j^{(0)}$ and $\beta_i = \beta_j$. It is straightforward to see that $\widehat{\gamma}_i^{(n)}(t) = \widehat{\gamma}_j^{(n)}(t)$ and $\mathcal{J}_i = \mathcal{J}_j$ in the E)-step and M)-step, respectively. Thus, $\widehat{\Theta}_i^{(n+1)} = \widehat{\Theta}_j^{(n+1)}$ for all n which means that the final estimates for θ_i and θ_j will be the same. In order to avoid this problem, β_i must not equal β_j or $\widehat{\theta}_i^{(0)}$ must not equal $\widehat{\theta}_j^{(0)}$ in order to fully exploit the capability of the EMIM algorithm.

Simulations

Our experiments consist of estimating the sinusoidal parameters for the following data,

$$\begin{aligned} y(t) &= 1.0 \sin(2\pi(0.2)t + 0.0) \\ &\quad + 1.0 \sin(2\pi(0.22)t + 0.0) + n(t), \\ t &= 1, \dots, 35, \end{aligned}$$

where $n(t)$ is white Gaussian noise. Since $|0.2 - 0.22| < 1/35 = 0.02857$, the periodogram cannot be used to determine the frequencies. We choose the initial box for the EMIM algorithm to be:

$$\begin{aligned} &[\Theta_{1,0}, \Theta_{2,0}]^\top \\ &\quad = [A_{1,0}, F_{1,0}, \Phi_{1,0}, A_{2,0}, F_{2,0}, \Phi_{2,0}]^\top \\ &\quad = [[0.7\ 1.2], [0.1\ 0.3], [0\ 0.4], \\ &[0.7\ 1.2], [0.1\ 0.3], [0\ 0.4]]^\top \end{aligned}$$

and $\beta_1 = 0.09$, $\beta_2 = 0.91$. The signal-to-noise-ratio (SNR) is defined as

$$10 \log \left[\sum_{k=1}^{K} 0.5 \frac{(a_k^*)^2}{\sigma_n^2} \right] ,$$

where σ_n^2 is the variance of the noise. If no a priori information about the possible values of the sinusoid parameters is available, the full range of possible values for the frequency, the phase, and the amplitude must be used as the initial intervals. Utilizing the full range will impose no difficulty when very fast computing engines are used. However, other high resolution techniques can be used to yield a smaller and more cogent initial interval.

Using 50 MC runs, we computed the sample means and standard deviations for the EMIM and the IQML algorithms. (See Table 4 and Table 5, respectively). As for the EMIM, the mid-points of the final interval estimates are considered as the final estimates, thus the

Global Optimization Methods for Harmonic Retrieval, Table 4
EMIM estimates

EMIM: $N = 35$, $\rho = 10^{-6}$ (50MC runs)			
SNR	10	5	0
$a_1^* = 1.0$	1.0305 ±0.0992	1.0235 ±0.1389	1.0263 ±0.1622
$f_1^* = .20$	0.1993 $\pm 2.209 \cdot 10^{-4}$	0.1993 $\pm 4.119 \cdot 10^{-4}$	0.1969 ±0.0110
$\phi_1^* = 0$	0.0631 ±0.0764	0.0851 ±0.1152	0.1369 ±0.1609
$a_2^* = 1.0$	1.0284 ±0.0744	1.0501 ±0.1036	1.0995 ±0.1054
$f_2^* = .22$	0.2192 ±0.0012	0.2194 ±0.0016	0.2182 ±0.0051
$\phi_2^* = 0$	0.0746 ±0.1177	0.0757 ±0.1224	0.1314 ±0.1662

Global Optimization Methods for Harmonic Retrieval, Table 5
IQML estimates

IQML: $N = 35$ (50 MC runs)			
SNR	10	5	0
$a_1^* = 1.0$	0.9549 ±0.3283	0.6615 ±0.2908	0.7404 ±0.2778
$f_1^* = .20$	0.1963 ±0.0137	0.1707 ±0.0476	0.1404 ±0.0836
$\phi_1^* = 0$	0.3332 ±0.6117	0.9323 ±1.0843	0.5472 ±1.0659
$a_2^* = 1.0$	0.9013 ±0.3732	0.7567 ±0.2683	0.8582 ±0.2788
$f_2^* = .22$	0.2428 ±0.0685	0.2559 ±0.0867	0.2721 ±0.0985
$\phi_2^* = 0$	−0.0886 ±0.4852	−0.0079 ±0.7185	0.3123 ±0.8358

sample mean and variance can be calculated accordingly. Note that the EMIM generates estimates which have mean values very close to the true parameter values and relatively very small variances. As for the IQML, its variance for each value of SNR is significantly larger than the corresponding EMIM. Clearly, EMIM outperforms IQML by providing estimates that are less biased with smaller variances.

Conclusion

In comparison between the two types of IM algorithms with the IQML method, it was shown that both the IM and EMIM algorithms represent a powerful tool for solving the HR problem. Furthermore, it has been noted that by decomposing the problem by the EMIM algorithm does not degrade the performance of using the IM.

We have shown *experimentally* that the IM and EMIM algorithms are robust for very short data records and low SNR. Nevertheless, if the dimensionality is low or convergence to the ML estimates is desired, then the IM algorithm can be used. For either EMIM or IM, convergence time can be improved by generating initial interval of smaller widths by using other high resolution HR methods. Furthermore, using a multi-processor computer to implement the decomposed sub-problems in parallel can also reduce the execution time.

See also

► Signal Processing with Higher Order Statistics

References

1. Bresler Y, Macovski A (1986) Exact maximum likelihood parameter estimation of superimposed exponential signals in noise. IEEE Trans Acoustics, Speech and Signal Processing 34:1081–89
2. Feder M, Weinstein E (1988) Parameter estimation of superimposed signals using the EM algorithm. IEEE Trans Acoustics, Speech and Signal Processing 36:477–489
3. Hansen E (1992) Global optimization using interval analysis. M. Dekker, New York
4. Kay SM (1988) Modern spectral estimation: Theory and application. Prentice-Hall, Englewood Cliffs, NJ
5. Kearfott R, Dawande M, Du K, Hu C (1992) INTLIB: A portable FORTRAN 77 elementary function library. Interval Comput 3:96–105
6. Moore RE (1979) Methods and applications of interval analysis. SIAM, Philadelphia
7. Ratschek H, Rokne J (1988) New computer methods for global optimization. Horwood, Westergate
8. Stoica P et al (1989) Maximum likelihood estimation of the parameters of multiple sinusoids from noisy measurements. IEEE Trans Acoustics, Speech and Signal Processing 37:378–392
9. Stoica P, Li J, Söderström T (1998) On the inconsistency of IQML. IEEE Trans Signal Processing 37:378–392

Global Optimization Methods for Systems of Nonlinear Equations

GO for SNE

Nguyen V. Thoai
University Trier, Trier, Germany

MSC2000: 65H10, 90C26, 90C30

Article Outline

Keywords

Systems of nonlinear equations; Global optimization

The problem of finding a solution of a *system of equations* and/or *system of inequalities* is one of the main research subjects in numerical analysis and optimization. The source of systems of equations and/or inequalities contains many 'real-world' problems ([2,7]), the nonlinear complementarity problem (cf. also ▶ Generalized nonlinear complementarity problem), the variational inequality problem (cf. also ▶ Variational inequalities) over a convex set, Karush-Kuhn-Tucker systems, the feasibility problem, the problem of computing a Brouwer's fixed point ([10,15]).

In general, a system of nonlinear equations and/or inequalities is given by

$$(SNE)\begin{cases} h_i(x) = 0, & i \in I, \\ g_j(x) \leq 0, & j \in J, \\ & x \in X, \end{cases}$$

where I, J are finite index sets, $X \subseteq \mathbf{R}^n$ is a convex set, and h_i $(i \in I)$, g_j $(j \in J)$ are nonlinear functions defined on a suitable set containing X.

Solution methods for (SNE), which are based on convex and nonsmooth optimization techniques, and fixed point algorithms can be found in [2,3,4,5,14,15], and references given therein.

In order to apply global optimization methods for solving (SNE), one defines a vector function h: $\mathbf{R}^n \to \mathbf{R}^{|I|}$ having components $h_i(x) (i \in I)$, a function

$$f(x) = \max\{\|h(x)\|, \{g_j(x)\colon \ j \in J\}\},$$

where $\|\cdot\|$ is any vector norm on $\mathbf{R}^{|I|}$, and considers the following *global optimization problem*

$$(GOP)\ f^* = \min\{f(x)\colon \ x \in X\}.$$

In particular, the function f in (GOP) can be defined by

$$f(x) = \max\left\{\{|h_i(x)|\colon \ i \in I\}, \{g_j(x)\colon \ j \in J\}\right\}.$$

In general, a vector $x^* \in \mathbf{R}^n$ is a solution of (SNE) if and only if it is a *global optimal solution* of (GOP) and $f^* = f(x^*) = 0$. Thus, finding a solution of (SNE) can be replaced by computing a global optimal solution of (GOP). In the case that $I = \emptyset$, i. e., (SNE) is a system of inequalities, global optimization algorithms to (GOP) will terminate whenever a feasible point $x \in X$ is found satisfying $f(x) \leq 0$. While applying a global optimization algorithm to (GOP), if it is pointed out that $f^* > 0$ (e. g., a lower bound μ of f^* can be computed such that $\mu > 0$), then obviously (SNE) has no solution.

There are three main classes of (SNE), which can be solved by implementable methods in global optimization:

i) The functions h_i $(i \in I)$ and g_j $(j \in J)$ are all d.c. (a function is called *d.c.* if it can be expressed as the difference of two convex functions, see ▶ D.C. programming).

ii) The functions h_i $(i \in I)$ and g_j $(j \in J)$ are all Lipschitzian with Lipschitz constants L_i $(i \in I)$ and M_j $(j \in J)$, respectively.

iii) The corresponding problem (GOP) can be replaced by a *convex relaxation problem*.

For class i), the function f in (GOP) is d.c., and one can find an explicit form of f as the difference of two convex functions, so that d.c. programming techniques can be applied ([9,11,12,18,19]).

For class ii), if in the definition of f, ℓ_p-norms are used, i. e.

$$\|h(x)\|_p = \begin{cases} \left(\sum_{i\in I} |h_i(x)|^p\right)^{\frac{1}{p}}, & 1 \leq p < \infty \\ \max_{i\in I} |h_i(x)|, & p = \infty, \end{cases}$$

then f is Lipschitzian with Lipschitz constant $L = \max\ \{\sum_{i\in I} L_i,\ \{M_j\colon\ j \in J\}\}$. Algorithms for solving Lipschitz optimization problems can be found in [6,7,8,9,10,12,16,17].

Techniques for the construction of convex relaxation problems for some special cases of class iii) are given in [13].

See also

▶ αBB Algorithm
▶ Continuous Global Optimization: Applications
▶ Continuous Global Optimization: Models, Algorithms and Software
▶ Contraction-mapping
▶ Convex Envelopes in Optimization Problems
▶ Differential Equations and Global Optimization
▶ DIRECT Global Optimization Algorithm
▶ Global Optimization Based on Statistical Models
▶ Global Optimization in Batch Design Under Uncertainty

- ▶ Global Optimization in Binary Star Astronomy
- ▶ Global Optimization in Generalized Geometric Programming
- ▶ Global Optimization of Heat Exchanger Networks
- ▶ Global Optimization in Phase and Chemical Reaction Equilibrium
- ▶ Global Optimization Using Space Filling
- ▶ Interval Analysis: Systems of Nonlinear Equations
- ▶ Interval Global Optimization
- ▶ MINLP: Branch and Bound Global Optimization Algorithm
- ▶ MINLP: Global Optimization with αBB
- ▶ MINLP: Heat Exchanger Network Synthesis
- ▶ MINLP: Mass and Heat Exchanger Networks
- ▶ Mixed Integer Linear Programming: Heat Exchanger Network Synthesis
- ▶ Mixed Integer Linear Programming: Mass and Heat Exchanger Networks
- ▶ Nonlinear Least Squares: Newton-type Methods
- ▶ Nonlinear Systems of Equations: Application to the Enclosure of All Azeotropes
- ▶ Smooth Nonlinear Nonconvex Optimization
- ▶ Topology of Global Optimization

References

1. Allgower EL, Georg K (1980) Simplicial and continuation methods for approximating fixed points and solutions to system of equations. SIAM Rev 22:28–85
2. Dennis JE, Schnabel RB (1983) Numerical methods for nonlinear equations and unconstrained optimization. Prentice-Hall, Englewood Cliffs, NJ
3. Forster W (1992) Some computational methods for systems of nonlinear equations and systems of polynomial equations. J Global Optim 2:317–356
4. Gabriel A, Pang J-S (1994) A trust region method for constrained nonsmooth equations. In: Hage WW, Hearn DW, Pardalos PM (eds) Large Scale Optimization: State of the Art. Kluwer, Dordrecht
5. Goffin J-L, Luo Z-Q, Ye Y (1994) On the complexity of a column generation algorithm for convex or quasiconvex feasibility problems. In: Hage WW, Hearn DW, Pardalos PM (eds) Large Scale Optimization: State of the Art. Kluwer, Dordrecht
6. Hansen P, Jaumard B (1995) Lipschitz optimization. In: Horst R, Pardalos PM (eds) Handbook of Global Optimization. Kluwer, Dordrecht
7. Hendrix EMT, Pinter J (1991) An application of Lipschitzian global optimization to product design. J Global Optim 1:389–401
8. Horst R, Nast M, Thoai NV (1997) New LP-bound in multivariate Lipschitz optimization: Theory and applications. J Optim Th Appl 86:369–388
9. Horst R, Pardalos PM, Thoai NV (1995) Introduction to global optimization. Kluwer, Dordrecht
10. Horst R, Thoai NV (1989) Branch and bound methods for solving systems of Lipschitzian equations and inequalities. J Optim Th Appl 58:139–145
11. Horst R, Thoai NV (1999) DC programming: Overview. J Optim Th Appl 103:1–43
12. Horst R, Tuy H (1993) Global optimization: Deterministic approaches, 2nd edn. Springer, Berlin
13. Maranas CD, Floudas CA (1995) Finding all solutions of nonlinearly constrained systems of equations. J Global Optim 7:143–182
14. Nesterov Y, Nemirovskii A (1994) Interior-point polynomial algorithms in convex programming. SIAM, Philadelphia
15. Pang J-S, Qi L (1993) Nonsmooth equations: Motivation and algorithms. SIAM J Optim 3:443–465
16. Pinter J (1991) Solving nonlinear equation systems via global partition and search. Computing 43:309–323
17. Strongin RG (1992) Algorithms for multi-extremal mathematical programming problems employing the set of joint space-filling curves. J Global Optim 2:357–378
18. Thoai NV (1994) Employment of conical algorithm and outer approximation method in D.C. programming. J Math 22:71–85
19. Tuy H (1995) D.C. optimization: Theory, methods and algorithms. In: Horst R, Pardalos PM (eds) Handbook Global Optim. Kluwer, Dordrecht, pp 149–216

Global Optimization in Multiplicative Programming

TAKAHITO KUNO
University Tsukuba, Ibaraki, Japan

MSC2000: 90C26

Article Outline

Keywords

Global optimization; Nonconvex minimization; Low-rank nonconvexity; Parametric methods

Multiplicative functions, products of real-valued functions f_i, $i = 1, \ldots, p$, are generally nonconvex functions even though each f_i is convex. As a result, most multiplicative programming problems containing $\prod_{i=1}^{p} f_i(x)$ in the objective and/or constraints are nonconvex minimization; and hence we need global optimization to look for a global minimum in stacks of local minima. Fortunately, however, the number p of f_is in multiplicative functions encountered in practical applications is rather small in comparison with the number n of variables; e. g. two or three in geometrical optimization [10] and at most five in multiple objective optimization [1]. As will be seen later, this enables us to embed the troublesome nonconvexity into a small subspace of dimension p. Exploiting such a property, called *low-rank nonconvexity* [6], a number of researchers have developed efficient algorithms since the late 1980s years to solve various subclasses of multiplicative programming problems, including the *linear multiplicative program*

$$\begin{cases} \min & (\mathbf{c}_1^\top \mathbf{x} + c_{10})(\mathbf{c}_2^\top \mathbf{x} + c_{20}) \\ \text{s.t.} & \mathbf{x} \in D, \end{cases} \tag{1}$$

where $D \subset \mathbf{R}^n$ is a polytope and $\mathbf{c}_i^\top \mathbf{x} + c_i0 > 0$ for any $\mathbf{x} \in D$; the *convex multiplicative program*

$$\begin{cases} \min & \prod_{i=1}^{p} f_i(\mathbf{x}) \\ \text{s.t.} & \mathbf{x} \in D, \end{cases} \tag{2}$$

where D is a compact convex set and the f_is are convex functions positive-valued on D; the *generalized convex multiplicative program*

$$\begin{cases} \min & \sum_{i=1}^{p} f_{2i-1}(\mathbf{x}) f_{2i}(\mathbf{x}) + g(\mathbf{x}) \\ \text{s.t.} & \mathbf{x} \in D, \end{cases} \tag{3}$$

where D and the f_is are the same as in (2) and g is a convex function; and the convex program with an additional *convex multiplicative constraint*

$$\begin{cases} \min & g(\mathbf{x}) \\ \text{s.t.} & \mathbf{x} \in D \\ & \prod_{i=1}^{p} f_i(\mathbf{x}) \leq 1, \end{cases} \tag{4}$$

where D, the f_is and g are the same as in (3). As long as p is a small number, all of these nonconvex programs can be solved in a practical amount of time even if n exceeds a few hundreds.

Linear Multiplicative Program

Problem (1), though simple looking, is *NP*-hard (cf. also ▶ Complexity theory; ▶ Complexity classes in optimization) as shown in [11]. There are two major methods, each of which is based on a variant of parametric simplex algorithms for linear programming [12].

The first method introduces a parameter $\xi \geq 0$ and transforms (1) into an equivalent problem:

$$\begin{cases} \min & \xi f_1(\mathbf{x}) \\ \text{s.t.} & \mathbf{x} \in D \\ & f_2(\mathbf{x}) \leq \xi, \quad \xi \geq 0, \end{cases} \tag{5}$$

where $f_i(\mathbf{x}) = \mathbf{c}_i^\top \mathbf{x} + c_i0$, $i = 1, 2$. To solve (5), we need only to solve

$$\min \{ f_1(\mathbf{x}) \colon \ \mathbf{x} \in D, \ f_2(\mathbf{x}) \leq \xi \} \tag{6}$$

for all $\xi \geq \xi_{\min} = \min\{f_2(\mathbf{x}) \colon \mathbf{x} \in D\}$, using the *parametric right-hand side simplex algorithm* (cf. also ▶ Parametric linear programming: Cost simplex algorithm). We then have a set of optimal solutions $\mathbf{x}(\xi)$ to (6) and the analytical expression of

$$\phi(\xi) = \xi f_1(\mathbf{x}(\xi)),$$

which is a piecewise quadratic function over $\xi \geq \xi_{\min}$. Let

$$\xi^* \in \arg\min \{ \phi(\xi) \colon \ \xi \geq \xi_{\min} \} .$$

Then $\mathbf{x}(\xi^*)$ is an optimal solution to (1).

This parametric method, proposed by K. Swarup [13] in the middle 1960s, was originally used for finding a locally optimal solution to (1). Strangely, it had not been appreciated as an efficient global optimization tool until the second parametric method was developed by H. Konno and T. Kuno [4] more than twenty years later.

The second method also introduces a parameter $\xi \geq 0$, but in a deferent way:

$$\begin{cases} \min & F(\mathbf{x}, \xi) \equiv \xi f_1(\mathbf{x}) + \frac{f_2(\mathbf{x})}{\xi} \\ \text{s.t.} & \mathbf{x} \in D, \quad \xi \geq 0. \end{cases} \tag{7}$$

For any $\mathbf{x}$ we have

$$\min \{F(\mathbf{x}, \xi) \colon \ \xi \geq 0\} = 2\sqrt{f_1(\mathbf{x}) f_2(\mathbf{x})}.$$

Therefore, (7) is equivalent to (1); moreover, (7) is equivalent to finding a minimum point ξ^* of a function

$$\psi(\xi) = \min \{F(\mathbf{x}, \xi) \colon \ \mathbf{x} \in D\} \tag{8}$$

over $\xi > 0$. Since the right-hand side of (8) is a linear program, we can locate ξ^* using the *parametric objective simplex algorithm*. In fact, noting that $\lambda = \xi/(\xi + 1/\xi)$ maps $\Xi = \{\xi \colon \xi > 0\}$ to a unit interval $\{\lambda \colon 0 < \lambda < 1\}$, we solve

$$\min \left\{\lambda \mathbf{c}_1^\top \mathbf{x} + (1 - \lambda)\mathbf{c}_2^\top \mathbf{x} \colon \ \mathbf{x} \in D\right\} \tag{9}$$

parametrically over $\lambda \in (0, 1)$. Let $\mathbf{x}(\lambda)$ denote an optimal solution to (9). Then

$$\lambda^* \in \arg\min \{f_1(\mathbf{x}(\lambda)) f_2(\mathbf{x}(\lambda)) \colon \ \lambda \in (0, 1)\}$$

gives $\xi^* = \sqrt{\frac{\lambda^*}{(1-\lambda^*)}}$; and $\mathbf{x}(\lambda^*)$ is an optimal solution to (1).

Under some probabilistic assumptions, the average number of simplex pivots needed to solve a linear program with a single parameter is known to be polynomial in the problem input length [12]. Hence, (1) can also be solved in polynomial time on the average, which contrasts sharply with the result of the worst-case analysis.

Convex Multiplicative Program

The above parametric methods for (1) can be extended to more general classes of multiplicative programming problems. For example, (7) is directly applicable to the special case of (2) where $p = 2$; but it is difficult to design an algorithm for solving (7) parametrically when the f_is are nonlinear functions. One effective approach in this case is branch and bound on the set of parameter values $\Xi = \{\xi \colon \xi > 0\}$ [7] (cf. also ▶ Integer programming: Branch and bound methods).

Let $\mathcal{F}$ denote the family of functions of the form:

$$\alpha\xi + \frac{\beta}{\xi},$$

where $\alpha, \beta \in \mathbf{R}$. The function ψ defined by (8) is a pointwise minimum of some functions in $\mathcal{F}$ such that $\alpha = f_1(\mathbf{x})$ and $\beta = f_2(\mathbf{x})$ for $\mathbf{x} \in D$. The family $\mathcal{F}$ possesses the following properties:

i) Any two points $(\xi_s, \psi_s), (\xi_t, \psi_t) \in \mathbf{R}^2$, with $0 < \xi_s < \xi_t$, uniquely determine

$$\frac{\psi_s \xi_s - \psi_t \xi_t}{\xi_s^2 - \xi_t^2}\xi + \frac{\psi_s/\xi_s - \psi_t/\xi_t}{1/\xi_s^2 - 1/\xi_t^2}/\xi \in \mathcal{F};$$

ii) Any function in $\mathcal{F}$ is Lipschitz continuous over $\xi \geq \xi'$ for any $\xi' > 0$;

iii) Two distinct functions in $\mathcal{F}$ have at most one intersection point over $\xi > 0$.

Suppose $[\xi_s, \xi_t] \subset \Xi$ is an interval containing ξ^*. Since f_1 and f_2 are convex, $F(\cdot, \xi)$ is also a convex function for any $\xi > 0$; and hence $\psi(\xi_s)$ and $\psi(\xi_t)$ can be computed by convex programming. For $(\xi_s, \psi(\xi_s))$ and $(\xi_t, \psi(\xi_t))$, let us construct a function in $\mathcal{F}$ according to i):

$$u(\xi; \xi_s, \xi_t) = \frac{\psi(\xi_s)\xi_s - \psi(\xi_t)\xi_t}{\xi_s^2 - \xi_t^2}\xi + \frac{\psi(\xi_s)/\xi_s - \psi(\xi_t)/\xi_t}{1/\xi_s^2 - 1/\xi_t^2}/\xi.$$

From iii) we have

$$u(\xi; \xi_s, \xi_t) \leq \psi(\xi), \quad \forall \xi \in [\xi_s, \xi_t].$$

Let $\xi_m \in \arg\min \{u(\xi; \xi_s, \xi_t) \colon \xi \in [\xi_s, \xi_t]\}$ and

$$u_2(\xi) = \begin{cases} u(\xi; \xi_s, \xi_m) & \text{if } 0 < \xi \leq \xi_m, \\ u(\xi; \xi_m, \xi_t) & \text{if } \xi \geq \xi_m. \end{cases}$$

Then u_2 underestimates ψ over $[\xi_s, \xi_t]$ and is better than $u_1 = u(\cdot; \xi_s, \xi_t)$ in the sense:

$$u_1(\xi) \leq u_2(\xi) \leq \psi(\xi), \quad \forall \xi \in [\xi_s, \xi_t].$$

In this way, as improving the *underestimator* of ψ successively, we can generate the sequence of minimum points of u_ks convergent to ξ^*.

The parametrization (7) can further be extended to (2) with $p \geq 2$ [8] as follows:

$$\begin{cases} \min & F(\mathbf{x}, \boldsymbol{\xi}) \equiv \sum_{i=1}^{p} \xi_i f_i(\mathbf{x}) \\ \text{s.t.} & \mathbf{x} \in D, \\ & \prod_{i=1}^{p} \xi_i \geq 1, \quad \boldsymbol{\xi} \geq 0. \end{cases} \tag{10}$$

Karush–Kuhn–Tucker conditions with respect to $\boldsymbol{\xi}$ imply the equivalence between (2) and (10). Let

$$\psi(\boldsymbol{\xi}) = \min\{F(\mathbf{x}, \boldsymbol{\xi}) : \mathbf{x} \in D\}.$$

Then (10) reduces to a problem with p variables:

$$\begin{cases} \min & \psi(\boldsymbol{\xi}) \\ \text{s.t.} & \prod_{i=1}^{p} \xi_i \geq 1, \quad \boldsymbol{\xi} \geq 0. \end{cases} \tag{11}$$

The objective function ψ is concave and coordinatewise nondecreasing; and its value at any $\boldsymbol{\xi} \geq 0$ can be computed by convex programming.

An alternative approach to (2) with $p \geq 2$ [14] is a generalization of (5):

$$\begin{cases} \min & \prod_{i=1}^{p} \xi_i \\ \text{s.t.} & \mathbf{x} \in D, \\ & f_i(\mathbf{x}) \leq \xi_i, \quad i = 1, \ldots, p, \\ & \boldsymbol{\xi} \geq 0. \end{cases} \tag{12}$$

Let $W \in \mathbf{R}^n \times \mathbf{R}^p$ denote the feasible region of (12) and

$$\Omega = \left\{\boldsymbol{\xi} \in \mathbf{R}^p : \exists \mathbf{x},\ (\mathbf{x}, \boldsymbol{\xi}) \in W\right\}.$$

Then (12) also reduces to a problem with p variables:

$$\begin{cases} \min & \sum_{i=1}^{p} \log \xi_i \\ \text{s.t.} & \boldsymbol{\xi} \in \Omega. \end{cases} \tag{13}$$

The objective function is concave; the feasible region Ω is a projection of the convex set W and hence a convex set.

Both (11) and (13) are concave minimization problems (cf. also ► Concave programming); however, even general-purpose algorithms such as branch and bound and outer approximation (cf. also ► Generalized outer approximation) [3] can handle them very efficiently when p is less than five.

Other Multiplicative Programs

In a way similar to (11), problem (3) can reduce to a concave minimization problem with $2p$ variables [5] through a parametrization:

$$\begin{cases} \min & \sum_{i=1}^{p} \frac{\xi_{2i-1}(f_{2i-1}(\mathbf{x}))^2 + \xi_{2i}(f_{2i}(\mathbf{x}))^2}{2} \\ & \quad +g(\mathbf{x}) \\ \text{s.t.} & \mathbf{x} \in D \\ & \xi_{2i-1}\xi_{2i} \geq 1,\ i = 1, \ldots, p, \\ & \boldsymbol{\xi} \geq 0. \end{cases} \tag{14}$$

Let $\psi(\boldsymbol{\xi})$ denote the optimal value of (14) with fixed $\boldsymbol{\xi}$. Then (14) reduces to

$$\begin{cases} \min & \psi(\boldsymbol{\xi}) \\ \text{s.t.} & \xi_{2i-1}\xi_{2i} \geq 1,\ i = 1, \ldots, p, \\ & \boldsymbol{\xi} \geq 0. \end{cases} \tag{15}$$

The objective function ψ is concave and coordinatewise nondecreasing. For problem (4), we can use the following parametrization [9]:

$$\begin{cases} \min & g(\mathbf{x}) \\ \text{s.t.} & \mathbf{x} \in D \\ & f_i(\mathbf{x}) \leq \xi_i, i = 1, \ldots, p, \\ & \prod_{i=1}^{p} \xi_i \leq 1, \quad \boldsymbol{\xi} \geq 0. \end{cases} \tag{16}$$

Let $\psi(\boldsymbol{\xi})$ denote the optimal value of (16) with fixed $\boldsymbol{\xi}$. Then (16) is equivalent to

$$\begin{cases} \min & \psi(\boldsymbol{\xi}) \\ \text{s.t.} & \prod_{i=1}^{p} \xi_i \leq 1, \quad \boldsymbol{\xi} \geq 0. \end{cases} \tag{17}$$

The objective function ψ is convex; but the feasible region is a *d.c. set* (difference of two convex sets). Thus, we can solve (3) and (4) by solving smaller-size problems (15) and (17), respectively. For a more complete survey of the algorithms, see the article by Konno and Kuno in [2].

We have seen that the parametric approach offers an efficient tool to handle multiplicative programming problems. This approach is not specific to the multiplicative structure but can be extended to a much wider class of nonconvex minimization problems, including minimum concave-cost flow problems, facility location, multilevel programming and so forth. The textbook [6] shows how the parametric approach can be generalized to a broad class of problems.

See also

- ► Linear Programming
- ► Multiparametric Linear Programming
- ► Multiplicative Programming
- ► Parametric Linear Programming: Cost Simplex Algorithm

References

1. Geoffrion M (1967) Solving bicriterion mathematical programs. Oper Res 15:39–54
2. Horst R, Pardalos PM (1995) Handbook of global optimization. Kluwer, Dordrecht
3. Horst R, Tuy H (1993) Global optimization: deterministic approaches, 2nd edn. Springer, Berlin
4. Konno H, Kuno T (1992) Linear multiplicative programming. Math Program 56:51–64
5. Konno H, Kuno T, Yajima Y (1994) Global minimization of a generalized convex multiplicative function. J Global Optim 4:47–62
6. Konno H, Thach PT, Tuy H (1997) Optimization on low rank nonconvex structures. Kluwer, Dordrecht
7. Kuno T, Konno H (1991) A parametric successive underestimation method for convex multiplicative programming problems. J Global Optim 1:267–285
8. Kuno T, Yajima Y, Konno H (1993) An outer approximation method for minimizing the product of several convex functions on a convex set. J Global Optim 3:325–335
9. Kuno T, Yajima Y, Yamamoto Y, Konno H (1994) Convex program with an additional constraint on the product of several convex functions. Europ J Oper Res 77:314–324
10. Maling K, Mueller SH, Heller WR (1982) On finding most optimal rectangular package plans. In: Proc. 19th Design Automation Conf, pp 663–670
11. Matsui T (1996) NP-hardness of linear multiplicative programming and related problems. J Global Optim 9:113–119
12. Schrijver A (1986) Theory of linear and integer programming. Wiley, New York
13. Swarup K (1966) Programming with indefinite quadratic function with linear constraints. Cahiers CERO 8:132–136
14. Thoai NV (1991) A global optimization approach for solving the convex multiplicative programming problem. J Global Optim 1:341–357

Global Optimization: p-αBB Approach

CHRYSANTHOS E. GOUNARIS,
CHRISTODOULOS A. FLOUDAS
Department of Chemical Engineering,
Princeton University, Princeton, USA

MSC2000: 49M37, 65K10, 90C26, 90C30, 46N10, 47N10

Article Outline

Keywords and Phrases

Convex underestimators; αBB; Global optimization

Introduction

Various deterministic global optimization algorithms that utilize a branch and bound framework make use of convex underestimators of the functions under consideration. This entry presents the work of Meyer and Floudas [11] on the convex underestimation of C^2-continuous functions. The work extends and refines the convex underestimation approach used in the αBB global optimization algorithm [1,2,3,4,10]. A recent review of deterministic global optimization approaches can be found in [6].

Let $f : \mathcal{R}^n \to \mathcal{R}$ be a smooth nonconvex C^2-continuous function. Its convex underestimator $\phi : \mathcal{R}^n \in x \to \mathcal{R}$ is defined as:

$$\phi(x) := f(x) - q(x) \tag{1}$$

where $q : \mathcal{R}^n \to \mathcal{R}$ is some perturbation function.

In the classical αBB approach, a series of simplifications are made to yield an efficient convexification procedure. The first of these simpifications is the imposition of a quadratic structure on the perturbation function:

$$q(x) := \sum_{i=1}^{n} \alpha_i \left(\overline{x}_i - x_i\right)\left(x_i - \underline{x}_i\right) \,. \tag{2}$$

To ensure that $q(x)$ is nonnegative, α is assumed to be nonnegative. Observe that $q(x)$, a quadratic function with a diagonal Hessian matrix $\nabla^2 q(x) := 2\,\mathrm{diag}(\alpha)$ has an eigenvalue–eigenvector structure that is uniform over the entire domain $\mathbf{x}$ with eigenvectors that are aligned with the coordinate axes. In the work of Adjiman et al. [2], a second simplification is introduced in which the interval extension, $\mathbf{H}^{\mathbf{x}}$, is used instead of $\nabla^2 f(x)$ itself. The interval extension of the matrix $\nabla^2 f(x) \in \mathcal{R}^{n\times n}$ is a matrix of intervals of $\mathcal{R}$. Each element $\mathbf{H}^{\mathbf{x}}{}_{i,j}$ of the matrix $\mathbf{H}^{\mathbf{x}}$ is defined in such a way that

$$\left.\frac{\partial^2 f}{\partial x_i \partial x_j}\right|_x \in \mathbf{H}^{\mathbf{x}}_{ij} \qquad \text{for all } x \in \mathbf{x}.$$

Computing the tightest possible interval extension is in itself a global optimization problem. In practice, an interval extension can be calculated using interval arithmetic [12,14,16]. The overestimation made in the interval calculations may result in a significant loss of accuracy. Adjiman et al. [2] applied the work of [5,7,8,9,13,15,17,18], and devised various methods to compute α vectors that guarantee the convexity of the underestimators. The tightness of the underestimators is dependent on the particular α calculation method used. Extensive computational testing [1] showed that the method based on the scaled Gerschgorin theorem performs better in practice.

In the work of Meyer and Floudas [11], the form of the αBB perturbation function and the way in which it is calculated are reexamined, a novel spline based method for convex underestimation is proposed and an efficient means of computing these tighter underestimators is elucidated.

An α Spline Underestimator

The size of the domain $\mathbf{x}$ affects the result of every step in the α calculation and strongly influences the tightness of the resulting convex underestimator. In particular, reducing $\mathbf{x}$ reduces the mismatch between the assumed quadratic functional form and the ideal form; it reduces the overestimation in the interval extension of the Hessian matrix; and the maximum separation distance has been shown to be a quadratic function of interval length [10]. It is therefore useful to construct a convex underestimator using a number of different α vectors, each applying to a subregion of the full domain $\mathbf{x}$.

Let $f(x) : \mathcal{R}^n \to \mathcal{R}$ be a C^2-continuous function. For each variable $x_i \in \mathcal{R}$, let the interval $[\underline{x}_i, \overline{x}_i]$ be partitioned into N_i subintervals. The endpoints of these subintervals are denoted with $x_i^0, x_i^1, \cdots, x_i^{N_i}$, where $\underline{x}_i = x_i^0 < x_i^1 < \cdots < x_i^k < \cdots < x_i^{N_i} = \overline{x}_i$. In this notation, the kth interval is $[x_i^{k-1}, x_i^k]$. A smooth convex underestimator of $f(x)$ over $\mathbf{x}$ is defined by (1). The new perturbation function, $q(x)$, would be:

$$q(x) := \sum_{i=1}^{n} q_i^k(x_i) \qquad \text{for } x_i \in \left[x_i^{k-1}, x_i^k\right] , \tag{3}$$

$$q_i^k(x_i) := \alpha_i^k \left(x_i - x_i^{k-1}\right) \cdot \left(x_i^k - x_i\right) + \beta_i^k x_i + \gamma_i^k \,. \tag{4}$$

In each interval $[x_i^{k-1}, x_i^k]$, $\alpha_i^k \geq 0$ is chosen such that $\nabla^2\phi(x)$, the Hessian matrix of $\phi(x)$, is positive semi-definite for all members of the set $\left\{x \in \mathbf{x} : x_i \in [x_i^{k-1}, x_i^k]\right\}$. $q_i^k(x_i)$ is the quadratic function associated with variable i in interval k. The function $q(x)$ is a piecewise quadratic function contructed from the functions $q_i^k(x_i)$.

The continuity and smoothness properties of $q(x)$ are produced in a spline-like manner. For $q(x)$ to be smooth the q_i^k functions and their gradients must match at the endpoints x_i^k. In addition, we require that $q(x) = 0$ at the vertices of the hyperrectangle $\mathbf{x}$. To satisfy these requirements, the following conditions are imposed for all $i = 1, \ldots, n$:

$$\begin{aligned}
q_i^1\left(x_i^0\right) &= 0 \\
q_i^k\left(x_i^k\right) &= q_i^{k+1}\left(x_i^k\right) \qquad \forall k = 1, \ldots, N_i - 1 \\
q_i^{N_i}\left(x_i^{N_i}\right) &= 0 \\
\left.\frac{\mathrm{d}q_i^k}{\mathrm{d}x_i}\right|_{x_i^k} &= \left.\frac{\mathrm{d}q_i^{k+1}}{\mathrm{d}x_i}\right|_{x_i^k} \qquad \forall k = 1, \ldots, N_i - 1
\end{aligned} \tag{5}$$

Expanding these equations for each $i = 1, \ldots, n$, one obtains the following system of equations:

$$\begin{aligned}
\beta_i^1 x_i^0 + \gamma_i^1 &= 0 \\
\beta_i^k x_i^k + \gamma_i^k &= \beta_i^{k+1} x_i^k + \gamma_i^{k+1} \\
&\qquad \forall k = 1, \ldots, N_i - 1 \\
\beta_i^{N_i} x_i^{N_i} + \gamma_i^{N_i} &= 0 \\
-\alpha_i^k \left(x_i^k - x_i^{k-1} \right) + \beta_i^k &= \alpha_i^{k+1} \left(x_i^{k+1} - x_i^k \right) + \beta_i^{k+1} \\
&\qquad \forall k = 1, \ldots, N_i - 1
\end{aligned} \tag{6}$$

which can be represented as:

$$\begin{bmatrix}
-x_i^0 & & & & & -1 & & & & \\
x_i^1 & -x_i^1 & & & & 1 & -1 & & & \\
& \ddots & \ddots & & & & \ddots & \ddots & & \\
& & x_i^k & -x_i^k & & & & 1 & -1 & \\
& & & \ddots & \ddots & & & & \ddots & \ddots \\
& & & & x_i^{N_i} & & & & & 1 \\
-1 & 1 & & & & & & & & \\
& -1 & 1 & & & & & & & \\
& & \ddots & \ddots & & & & & & \\
& & & -1 & 1 & & & & &
\end{bmatrix}
\begin{bmatrix}
\beta_i^1 \\ \beta_i^2 \\ \vdots \\ \beta_i^k \\ \vdots \\ \beta_i^{N_i} \\ \gamma_i^1 \\ \gamma_i^2 \\ \vdots \\ \gamma_i^{N_i}
\end{bmatrix}
= \begin{bmatrix}
0 \\ 0 \\ \vdots \\ 0 \\ \vdots \\ 0 \\ s_1 \\ s_2 \\ \vdots \\ s_i^{N_i-1}
\end{bmatrix} \tag{7}$$

where $s_i^k = -\alpha_i^k (x_i^k - x_i^{k-1}) - \alpha_i^{k+1} (x_i^{k+1} - x_i^k)$.

The solution of the above linear system of equations is:

$$\beta_i^1 = \frac{\sum_{k=1}^{N_i-1} s_i^k \left(x_i^k - x_i^{N_i} \right)}{x_i^{N_i} - x_i^0}$$

$$\beta_i^k = \beta_i^1 + \sum_{j=1}^{k-1} s_i^j \qquad \forall k = 2, \ldots, N_i \tag{8}$$

$$\gamma_i^k = -\beta_i^1 x_i^0 - \sum_{j=1}^{k-1} s_i^j x_i^j \qquad \forall k = 1, \ldots, N_i$$

For a rigorous proof of the continuity, smoothness, convexity and underestimation properties of underestimator $\phi(x)$, see [11].

Nonconcave Perturbation

Consider a function $f(x)$ which is convex in one subdomain and concave in another. In the α spline approach, $\phi(x)$ can be convex even if the α values are negative in the regions in which $f(x)$ is strictly convex. In the classical αBB underestimator, the underestimation property is guaranteed by the concavity of $q(x)$, as given in (2). The concavity of $q(x)$ is, in turn, a result of the non-negativity of the α values. In this section, we discuss how the underestimation property of $\phi(x)$ can be maintained when some α values are negative.

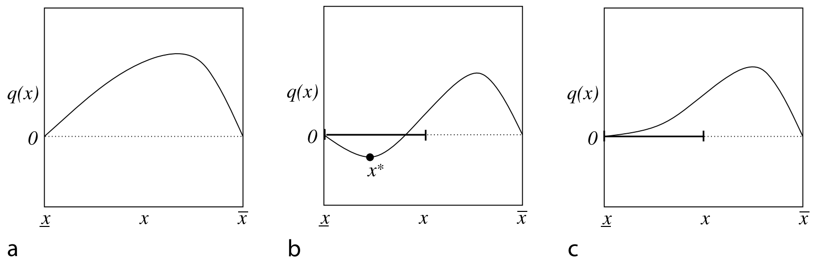
The underestimation property, $\phi(x) \leq f(x)$ for all $x \in \mathbf{x}$, is ensured by the following condition:

$$\min_{x \in \mathbf{x}} q(x) \geq 0 \tag{9}$$

Instead of solving minimization problems, the key idea is to adjust the α's to prevent the creation of local minima at any nonvertex point in $\mathbf{x}$ by prohibiting the occurrence of stationary points on convex regions of the perturbation function. This is illustrated in Fig. 1. In Fig. 1a, a concave perturbation function is depicted. The non-negativity of this function follows from its concavity. In Fig. 1b, a perturbation function is shown which is convex over the domain marked with a bold line.

The point x^* is a stationary point of q in this convex region and we note that $q(x^*)$ is negative. In Fig. 1c, the perturbation function is again convex over the marked region but there is no stationary point in this region. This function is non-negative over the entire domain $[\underline{x}, \overline{x}]$.

Using this idea, Meyer and Floudas [11] derived a tight convex underestimator by starting with $q(x)$, with non-negative α values as defined in Sect. "An α Spline Underestimator", and making the zero α's negative, one at a time, while maintaining the convexity



Global Optimization: p-αBB Approach, Figure 1
(a) Concave, (b) nonconcave, and (c) nonnegative nonconcave perturbation functions

of $\phi(x)$ and avoiding the generation of stationary points on the convex portions of $q(x)$. For the rest of this section we will assume that $f : \mathbf{x} \to \mathcal{R}$ is a univariate function, $\mathbf{x} = [\underline{x}, \overline{x}] \subset \mathcal{R}$. The separable structure of the α spline function allows the techniques developed here to be applied to the multivariate case.

Note that the β and γ parameters defining $q(x)$ are functions of the α's and the endpoints, $x^0, \ldots, x^N$. The following formula, derived from (8), is an expression for β^k in terms of $\alpha^1, \ldots, \alpha^N$.

$$\begin{aligned}\beta^k = \quad & \frac{1}{x^N - x^0} \sum_{j=1}^{k-1} \Big(-\alpha^j \left(x^j - x^{j-1}\right) \left(x^j - x^0\right) \\ & \qquad - \alpha^{j+1} \left(x^{j+1} - x^j\right) \left(x^j - x^0\right) \Big) \\ & + \frac{1}{x^N - x^0} \sum_{j=k}^{N-1} \Big(-\alpha^j \left(x^j - x^{j-1}\right) \left(x^j - x^N\right) \\ & \qquad - \alpha^{j+1} \left(x^{j+1} - x^j\right) \left(x^j - x^N\right) \Big)\end{aligned} \tag{10}$$

Suppose that having calculated $\beta \in \mathcal{R}^N$ for some given $\alpha \in \mathcal{R}^N$, we wish to modify some element α^j. Meyer and Floudas [11] derived formulae that may be used to update the β's following such an α update. Under the substitution $\alpha^j \to \tilde{\alpha}^j$, the elements $\tilde{\beta}^1, \ldots, \tilde{\beta}^N$ that satisfy (8) may be expressed in terms of $\beta^1, \ldots, \beta^N$, α^j and $\tilde{\alpha}^j$ using the following update formulae:

$$\begin{aligned}\tilde{\beta}^k - \beta^k &= \frac{1}{x^N - x^0} \left(\alpha^j - \tilde{\alpha}^j\right) \left(x^j - x^{j-1}\right) \left(x^{j-1} - x^0\right) \\ &\quad + \frac{1}{x^N - x^0} \left(\alpha^j - \tilde{\alpha}^j\right) \left(x^j - x^{j-1}\right) \left(x^j - x^0\right) \\ &= \frac{1}{x^N - x^0} \left(\alpha^j - \tilde{\alpha}^j\right) \left(x^j - x^{j-1}\right) \\ &\qquad \times \left(x^{j-1} + x^j - 2x^0\right) \qquad \text{if } j < k\end{aligned} \tag{11}$$

$$\begin{aligned}\tilde{\beta}^k - \beta^k &= \frac{1}{x^N - x^0} \left(\alpha^j - \tilde{\alpha}^j\right) \left(x^j - x^{j-1}\right) \left(x^{j-1} - x^0\right) \\ &\quad + \frac{1}{x^N - x^0} \left(\alpha^j - \tilde{\alpha}^j\right) \left(x^j - x^{j-1}\right) \left(x^j - x^N\right) \\ &= \frac{1}{x^N - x^0} \left(\alpha^j - \tilde{\alpha}^j\right) \left(x^j - x^{j-1}\right) \\ &\qquad \times \left(x^{j-1} + x^j - x^0 - x^N\right) \qquad \text{if } j = k\end{aligned} \tag{12}$$

$$\begin{aligned}\tilde{\beta}^k - \beta^k &= \frac{1}{x^N - x^0} \left(\alpha^j - \tilde{\alpha}^j\right) \left(x^j - x^{j-1}\right) \\ &\quad \cdot \left(x^{j-1} - x^N\right) \\ &\quad + \frac{1}{x^N - x^0} \left(\alpha^j - \tilde{\alpha}^j\right) \left(x^j - x^{j-1}\right) \\ &\quad \cdot \left(x^j - x^N\right) \\ &= \frac{1}{x^N - x^0} \left(\alpha^j - \tilde{\alpha}^j\right) \left(x^j - x^{j-1}\right) \\ &\qquad \times \left(x^{j-1} + x^j - 2x^N\right) \qquad \text{if } j > k\end{aligned} \tag{13}$$

A stationary point x^* of the function $q : \mathcal{R} \to \mathcal{R}$ is one that satisfies:

$$\left.\frac{\mathrm{d}q}{\mathrm{d}x}\right|_{x^*} = 0 \Leftrightarrow \alpha^k \left(x^k + x^{k-1} - 2x^*\right) + \beta^k = 0$$

in some interval $x^* \in [x^{k-1}, x^k]$. It follows that an interval k contains no stationary point if either $\frac{1}{2}(x^k + x^{k-1} + \beta^k/\alpha^k) > x^k$ or $\frac{1}{2}(x^k + x^{k-1} + \beta^k/\alpha^k) < x^{k-1}$.

Meyer and Floudas [11] derived conditions on α^j that guarantee the absence of such stationary points. Their results are summarized in the following three Lemmas, which correspond to cases $j < k$, $j = k$ and $j > k$, respectively.

Lemma 1 *Consider two intervals* $[x^{j-1}, x^j]$ *and* $[x^{k-1}, x^k]$ *where* $j < k$. *Let the sequence of* α *values* *defining* $q^k(x)$ *be*

$$\{\alpha^1, \ldots, \alpha^j, \ldots, \alpha^k, \ldots, \alpha^{N-1}\} ,$$

where $\alpha^k < 0$. *Let* $\tilde{q}^k(x)$ *be the function defined by the sequence of* α *values*

$$\{\alpha^1, \ldots, \tilde{\alpha}^j, \ldots, \alpha^k, \ldots, \alpha^{N-1}\} ,$$

where $\tilde{\alpha}^j < 0$. *There exists no stationary point of* $\tilde{q}^k(x)$ *on the interval* $[x^{k-1}, x^k]$ *if either of the following bounds on* $\tilde{\alpha}^j$ *hold:*

$$\tilde{\alpha}^j > \frac{\left(x^N - x^0\right)\left(-\alpha^k\left(x^k - x^{k-1}\right) + \beta^k\right)}{\left(x^j - x^{j-1}\right)\left(x^j + x^{j-1} - 2x^0\right)} + \frac{\alpha^j\left(x^j - x^{j-1}\right)\left(x^{j-1} + x^j - 2x^0\right)}{\left(x^j - x^{j-1}\right)\left(x^j + x^{j-1} - 2x^0\right)} ,$$

or

$$\tilde{\alpha}^j < \frac{\left(x^N - x^0\right)\left(\alpha^k\left(x^k - x^{k-1}\right) + \beta^k\right)}{\left(x^j - x^{j-1}\right)\left(x^j + x^{j-1} - 2x^0\right)} + \frac{\alpha^j\left(x^j - x^{j-1}\right)\left(x^{j-1} + x^j - 2x^0\right)}{\left(x^j - x^{j-1}\right)\left(x^j + x^{j-1} - 2x^0\right)} .$$

Lemma 2 *Consider an interval* $[x^{k-1}, x^k]$. *Let* $\{\alpha^1, \alpha^2, \ldots, \alpha^{N-1}\}$ *be the sequence of* α *values determining* $q^k(x)$. *Let* $\tilde{q}^k(x)$ *be the function defined by the sequence of* α *values*

$$\{\alpha^1, \ldots, \alpha^{k-1}, \tilde{\alpha}^k, \alpha^{k+1}, \ldots, \alpha^{N-1}\}$$

where $\tilde{\alpha}^k < 0$. *A stationary point of* $\tilde{q}(x)$ *does not exist on the interval* $[x^{k-1}, x^k]$ *if either of the following conditions hold:*

$$\begin{aligned} \tilde{\alpha}^k &> \frac{\zeta}{\left(x^k - x^{k-1}\right)\left(x^k + x^{k-1} - 2x^0\right)} \quad \text{if } \zeta \leq 0 \\ \tilde{\alpha}^k &> \frac{\zeta}{\left(x^k - x^{k-1}\right)\left(x^k + x^{k-1} - 2x^N\right)} \quad \text{if } \zeta > 0 \end{aligned} \qquad (14)$$

where

$$\zeta = \beta^k\left(x^N - x^0\right) + \alpha^k\left(x^k - x^{k-1}\right)\left(x^{k-1} + x^k - x^0 - x^N\right) .$$

Lemma 3 *Consider two intervals* $[x^{j-1}, x^j]$ *and* $[x^k, x^{k-1}]$ *where* $j > k$. *Let* $\alpha^k < 0$, *and* $\{\alpha^1, \ldots, \alpha^k, \ldots, \alpha^j, \ldots, \alpha^{N-1}\}$ *be the sequence of* α *values determining* $q^k(x)$. *Let* $\tilde{q}^k(x)$ *be the function defined by the sequence of* α *values* $\{\alpha^1, \ldots, \alpha^k, \ldots, \tilde{\alpha}^j, \ldots, \alpha^{N-1}\}$ *where* $\tilde{\alpha}^j < 0$. *A stationary point of* $\tilde{q}^k(x)$ *does not exist on the interval* $[x^{k-1}, x^k]$ *if either of the following bounds on* $\tilde{\alpha}^j$ *hold:*

$$\tilde{\alpha}^j > \frac{\left(x^N - x^0\right)\left(\alpha^k\left(x^k - x^{k-1}\right) + \beta^k\right)}{\left(x^j - x^{j-1}\right)\left(x^j + x^{j-1} - 2x^N\right)} + \frac{\alpha^j\left(x^j - x^{j-1}\right)\left(x^{j-1} + x^j - 2x^N\right)}{\left(x^j - x^{j-1}\right)\left(x^j + x^{j-1} - 2x^N\right)} ,$$

$$\tilde{\alpha}^j < -\frac{\left(x^N - x^0\right)\left(\alpha^k\left(x^k - x^{k-1}\right) - \beta^k\right)}{\left(x^j - x^{j-1}\right)\left(x^j + x^{j-1} - 2x^N\right)} + \frac{\alpha^j\left(x^j - x^{j-1}\right)\left(x^{j-1} + x^j - 2x^N\right)}{\left(x^j - x^{j-1}\right)\left(x^j + x^{j-1} - 2x^N\right)} .$$

When $q(x)$ is concave on a set of intervals and is guaranteed to have no stationary point on the remainder of the intervals, $q(x)$ is monotonically nondecreasing between x^0 and a global maximum x^* and monotonically nonincreasing between x^* and x^N. Under the aforementioned conditions, the perturbation function $q(x)$ is always non-negative and, thus, $\phi(x)$ is a valid underestimator of $f(x)$ [11].

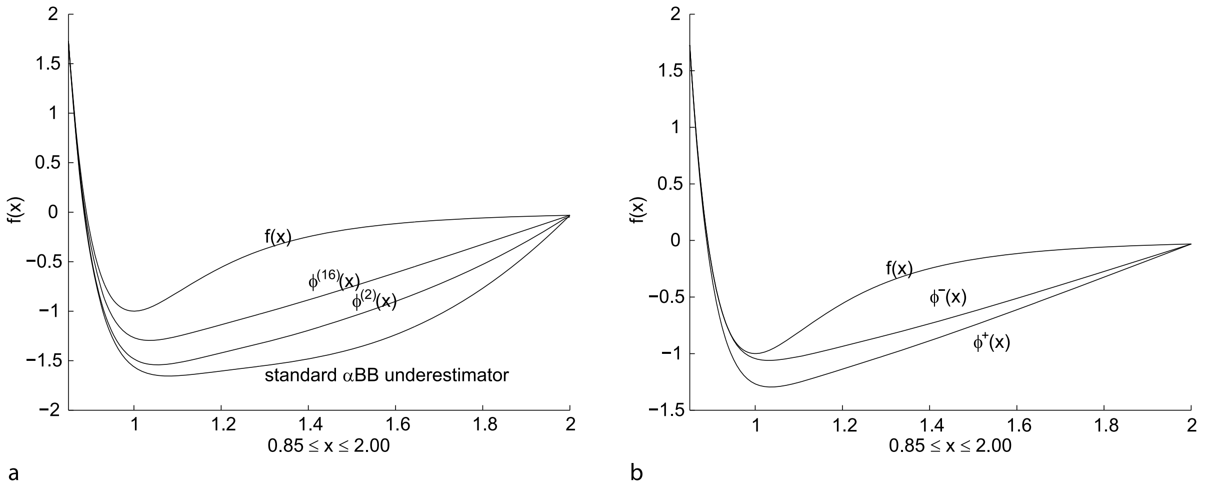
Illustrative Example

As an illustration, we present here an example from Meyer and Floudas [11]. It involves the well-known Lennard–Jones potential energy function:

$$f(x) = \frac{1}{x^{12}} - \frac{2}{x^6}$$

in the interval $[\underline{x}, \overline{x}] = [0.85, 2.00]$. The first term of this function is a convex function and dominates when x is small, while the second term is a concave function which dominates when x is large. The minimum eigenvalue of this function in an interval $[\underline{x}, \overline{x}]$ can be calculated explicitly as follows:

$$\min f'' = \begin{cases} \dfrac{156}{\overline{x}^{14}} - \dfrac{84}{\overline{x}^8} & \text{if } \ \overline{x} \leq 1.21707 \\ -7.47810 & \text{if } \ [\underline{x}, \overline{x}] \ni 1.21707 \\ \dfrac{156}{\underline{x}^{14}} - \dfrac{84}{\underline{x}^8} & \text{if } \ \underline{x} \geq 1.21707 . \end{cases}$$



Global Optimization: p-αBB Approach, Figure 2
Lennard–Jones convex underestimators with (a) concave and (b) nonconcave perturbations

Global Optimization: p-αBB Approach, Table 1
Parameters for 2 subinterval perturbation for Lennard–Jones function

k	x^k	$\min f''$	α^k	β^k	γ^k
0	0.850				
1	1.425	−7.47810	3.73905	1.62764	−1.38349
2	2.000	−3.84462	1.92231	−1.62764	3.25528

Global Optimization: p-αBB Approach, Table 2
Parameters for 16 subinterval perturbation of Lennard–Jones function

k	x^k	$\min f''$	α^k	β^k	γ^k
0	0.850000				
1	0.921875	326.18127	0.00000	1.78326	−1.51577
2	0.993750	81.99112	0.00000	1.78326	−1.51577
3	1.065625	13.55346	0.00000	1.78326	−1.51577
4	1.137500	−4.27629	2.13815	1.62958	−1.35200
5	1.209375	−7.46047	3.73024	1.20779	−0.87222
6	1.281250	−7.47810	3.73905	0.67093	−0.22296
7	1.353125	−6.71098	3.35549	0.16101	0.43038
8	1.425000	−5.21291	2.60645	−0.26750	1.01021
9	1.496875	−3.84462	1.92231	−0.59301	1.47405
10	1.568750	−2.78248	1.39124	−0.83117	1.83055
11	1.640625	−2.00473	1.00236	−1.00321	2.10044
12	1.712500	−1.44791	0.72395	−1.12729	2.30401
13	1.784375	−1.05201	0.52600	−1.21713	2.45786
14	1.856250	−0.77029	0.38515	−1.28262	2.57472
15	1.928125	−0.56887	0.28443	−1.33074	2.66405
16	2.000000	−0.42385	0.21192	−1.36642	2.73284

Global Optimization: p-αBB Approach, Table 3
Parameters defining nonconcave perturbations for the Lennard–Jones potential

k	x^k	$\min f''$	α^k	β^k	γ^k
0	0.850000				
1	0.921875	326.18127	0.00000	0.00000	0.00000
2	0.993750	81.99112	−7.37920	0.53038	−0.48894
3	1.065625	13.55346	−6.77673	1.54784	−1.50004
4	1.137500	−4.27629	2.13815	1.88124	−1.85532
5	1.209375	−7.46047	3.73024	1.45945	−1.37553
6	1.281250	−7.47810	3.73905	0.92259	−0.72627
7	1.353125	−6.71098	3.35549	0.41267	−0.07294
8	1.425000	−5.21291	2.60645	−0.01584	0.50689
9	1.496875	−3.84462	1.92230	−0.34135	0.97074
10	1.568750	−2.78248	1.39124	−0.57951	1.32724
11	1.640625	−2.00473	1.00236	−0.75155	1.59713
12	1.712500	−1.44791	0.72395	−0.87563	1.80069
13	1.784375	−1.05201	0.52600	−0.96547	1.95454
14	1.856250	−0.77029	0.38515	−1.03096	2.07140
15	1.928125	−0.56887	0.28443	−1.07909	2.16074
16	2.000000	−0.42385	0.21192	−1.11476	2.22952

The classical αBB underestimator for this function and interval is $f(x) - \frac{7.47810}{2}\left(\overline{x} - x\right)\left(x - \underline{x}\right)$. Bisecting the domain and applying (8), we obtain a convex underestimator defined by the parameters in Table 1.

Partitioning the domain into 16 equal sized subintervals and applying (8), we obtain the convex underestimator $\phi(x)$ with the parameters defining $q(x)$ of Table 2.

The potential energy function, the classical αBB underestimator, and the $\phi(x)$ underestimators are shown in Fig. 2a. In this figure, the α spline underestimator based on 2 subregions is denoted as $\phi^{(2)}$, while that based on 16 subregions is denoted as $\phi^{(16)}$.

Figure 2b depicts the strengthening of an underestimation function through the use of nonconcave perturbations. A negative α value has been assigned to two of the three regions in which the second derivative is strictly positive, as shown in Table 3. The resulting underestimator is depicted as $\phi^{-}(x)$, while the notation $\phi^{+}(x)$ is used to depict the underestimator with no negative α's (same as $\phi^{(16)}$ in Fig. 2a).

References

1. Adjiman CS, Androulakis IP, Floudas CA (1998) A Global Optimization Method, αBB, for General Twice-Differentiable Constrained NLPs II. Implementation and Computational Results. Comput Chem Eng 22:1159–1179
2. Adjiman CS, Dallwig S, Floudas CA, Neumaier A (1998) A Global Optimization Method, αBB, for General Twice-Differentiable Constrained NLPs I. Theoretical Advances. Comput Chem Eng 22:1137–1158
3. Adjiman CS, Floudas CA (1996) Rigorous Convex Underestimators for General Twice-Differentiable Problems. J Glob Optim 9:23–40
4. Androulakis IP, Maranas CD, Floudas CA (1995) αBB: A Global Optimization Method for General Constrained Nonconvex Problems. J Glob Optim 7:337–363
5. Deif AS (1991) The Interval Eigenvalue Problem. Z Angew Math Mech 71:61–64
6. Floudas CA, Akrotirianakis IG, Caratzoulas S, Meyer CA, Kallrath J (2005) Global Optimization in the 21st Century: Advances and Challenges. Comput Chem Eng 29:1185–1202
7. Gerschgorin S (1931) Über die Abgrenzung der Eigenwerte einer Matrix. Izv Akad Nauk SSSR Ser Mat 6:749–754
8. Hertz D (1992) The Extreme Eigenvalues and Stability of Real Symmetric Interval Matrices. IEEE Trans Autom Control 37:532–535
9. Kharitonov VL (1979) Asymptotic Stability of an Equilibrium Position of a Family of Systems of Linear Differential Equations. Differ Equations 78:1483–1485
10. Maranas CD, Floudas CA (1994) Global Minimum Potential Energy Conformations of Small Molecules. J Glob Optim 4:135–170
11. Meyer CA, Floudas CA (2005) Convex Underestimation of Twice Continuously Differentiable Functions by Piecewise Quadratic Perturbation : Spline αBB Underestimators. J Glob Optim 32:221–258
12. Moore RE (1966) Interval Analysis. Prentice Hall, Englewood Cliffs
13. Mori T, Kokame H (1994) Eigenvalue Bounds for a Certain Class of Interval Matrices. IEICE Trans Fundam 10: 1707–1709
14. Neumaier A (1990) Interval Methods for Systems of Equations. Cambridge University Press, Cambridge
15. Neumaier A (1992) An Optimality Criterion for Global Quadratic Optimization. J Glob Optim 2:201–208
16. Ratschek H, Rokne J (1984) Computer Methods for the Range of Functions. Ellis Horwood Limited, Chichester
17. Rohn J (1994) Bounds on Eigenvalues of Interval Matrices. In: Institute of Computer Science. Technical Report no.688. Academy of Sciences, Prague
18. Stephens C, Bomze IM, Csendes T, Horst R, Pardalos PM (1997) Interval and Bounding Hessians. In: Bomze IM (ed) Developments in Global Optimization. Kluwer, Dordrecht, pp 109–199

Global Optimization in Phase and Chemical Reaction Equilibrium

CONOR M. MCDONALD
E.I. DuPont de Nemours & Co., Wilmington, USA

MSC2000: 90C26, 90C90

Article Outline

See also
References

Keywords

Global optimization; Phase equilibrium; Phase stability problem

The prediction of the behavior of fluid mixtures is a fundamental aspect of chemical process engineering. The physico-chemical problem of computing solutions to the phase and chemical equilibrium problem is central to the design, control and operation of many important processes. These include distillation (standard and azeotropic), extraction trains, petroleum reservoirs and applications involving gases at high pressure. The ubiquity of the flash calculation in chemical engineering is just one example of its prevalence. Because the properties of many fluids vary in a complex fashion, the thermodynamic models that have arisen to describe their behavior create some difficulties in their application. These challenges will be explored in this article.

Problem Statement

The equilibrium condition is characterized by an extremum of some thermodynamic condition. Most commonly, the focus is on systems that attain equilibrium states under conditions of constant pressure (P) and temperature (T) where the global minimum value of the Gibbs free energy describes the true equilibrium state. The problem may be stated as follows:

> Given C components participating in up to P potential phases under isothermal and isobaric conditions find the number of phases and the distribution of components in those phases that yield the global minimum of the Gibbs free energy.

The requisite material balance constraints must also be satisfied. In what follows, all quantities associated with the Gibbs free energy are treated as dimensionless by dividing by RT, where R is the universal gas constant. The total Gibbs free energy is given by the summation of the molar Gibbs free energies for each phase:

$$G = \sum_{k \in P} n^k g^k = \sum_{k \in P} G^k ,$$

where n^k is the total number of mols present in phase k; g^k and G^k are respectively the *molar* and *total* Gibbs free energy of phase k. The composition variables can be defined intensively in terms of mol fractions ($\mathbf{x} \equiv \{x_i^k\}$), or extensively, as the number of mols of component i in phase k ($\mathbf{n} \equiv \{n_i^k\}$). It is easy to move from one form to the other via the relation $n_i^k = n^k x_i^k$. g^k is naturally expressed with intensive variables while extensive variables are appropriate for G^k. The equilibrium solution must also satisfy the linear material balance constraints.

Thermodynamic Models

Turning to the available thermodynamic models available to predict fluid phase behavior, these typically lead to expressions for the molar Gibbs functions that are mathematically complex, nonlinear and nonconvex. In this section, the analysis is presented for the molar Gibbs function.

Liquid Phases

Many liquid phases are only partially miscible (referred to as phase splitting). Nonideality is often expressed through the employment of *excess functions* which attempt to correlate the deviation of the system from ideality. The excess Gibbs free energy is simply the amount by which the Gibbs free energy is above that of an ideal solution:

$$g^{\mathrm{E}}(\mathbf{x}) = g(\mathbf{x}) - g^{\mathrm{I}}(\mathbf{x})$$

with

$$g^{\mathrm{I}}(\mathbf{x}) = \sum_{i \in C} x_i \mu_i^{\circ} + \sum_{i \in C} x_i \ln x_i ,$$

where μ_i° is the chemical potential of pure component i referred to the standard state. $g^{\mathrm{I}}(\mathbf{x})$ is convex. A number of different expressions of increasing complexity are now summarized for the excess Gibbs functions The only variables are the mol fractions x_i and all other quantities are parameters particular to the thermodynamic model. References to these equations and their parameters can be found in [21].

The Wilson Equation

Because the molar Gibbs free energy is convex in this case, this equation is the only model described here that cannot be used to predict phase splitting.

$$g^{\mathrm{E}}(\mathbf{x}) = -\sum_{i \in C} x_i \ln \sum_{j \in C} \Lambda_{ij} x_j .$$

Regular Solutions

This equation is bilinear:

$$g^{\mathrm{E}}(\mathbf{x}) = \sum_{i\in C}\sum_{j\in C} A_{ij}x_i x_j .$$

The NRTL Equation

This widely used model consists of a summation of bilinear fractional terms:

$$g^{\mathrm{E}}(\mathbf{x}) = \sum_{i\in C} x_i \frac{\sum_j \tau_{ji} G_{ji} x_j}{\sum_j G_{ji} x_j} .$$

The next three models are nonconvex in form. They are grouped together because it has been shown in [16] how they can be transformed into the difference of two convex functions (d.c. form), allowing the application of standard branch and bound global optimization algorithms.

The UNIQUAC Equation

The excess Gibbs function is composed of a residual part and a combinatorial part, denoted $g_{\mathrm{C}}^{\mathrm{E}}(\mathbf{x})$, defined as:

$$\begin{aligned} g_{\mathrm{C}}^{\mathrm{E}}(\mathbf{x}) = & \sum_{i\in C} x_i \left[1 - \frac{z}{2} q_i\right] \ln \frac{r_i x_i}{\sum_j r_j x_j} \\ & + \sum_{i\in C} \frac{z}{2} q_i x_i \ln \frac{q_i x_i}{\sum_j q_j x_j} . \end{aligned}$$

The excess Gibbs function is then given as:

$$g^{\mathrm{E}}(\mathbf{x}) = g_{\mathrm{C}}^{\mathrm{E}}(\mathbf{x}) + \sum_{i\in C} q_i' x_i \ln \frac{\sum_j q_j' x_j}{\sum_j q_j' \tau_{ji} x_j} .$$

The next two models represent the behavior of molecules in mixtures by aggregating the properties of constituent functional groups (represented by the index set $G = \{g\} = \{l\}$).

The UNIFAC Equation

The combinatorial part is the same as for the UNIQUAC equation:

$$\begin{aligned} g^{\mathrm{E}}(\mathbf{x}) = & \, g_{\mathrm{C}}^{\mathrm{E}}(\mathbf{x}) + \sum_{i\in C} x_i \sum_{g\in G} \nu_{gi} \\ & \times \left\{ Q_g \ln \frac{\sum_j x_j q_j}{\sum_j x_j \sum_l Q_l \nu_{lj} \Psi_{lg}} - \ln \Gamma_g^{(i)} \right\} . \end{aligned}$$

The ASOG Equation

$$\begin{aligned} g^{\mathrm{E}}(\mathbf{x}) = & \sum_{i\in C} x_i \ln \frac{s_i}{\sum_j x_j s_j} + \sum_{i\in C} x_i \sum_{g\in G} \nu_{gi} \\ & \times \left\{ \ln \frac{\sum_j x_j \sum_l \nu_{lj}}{\sum_j x_j \sum_l \nu_{lj} a_{gl}} - \ln \Gamma_g^{(i)} \right\} . \end{aligned}$$

Of all the above methods, the NRTL, UNIQUAC and UNIFAC are currently the most commonly used. Notice that some of the correlations are of high mathematical complexity. While this is necessary in order to predict multiple liquid phases, it can lead to problems where extraneous and erroneous additional phases are predicted. An example is given in [19] where the NRTL equation mathematically predicts three liquid phases when the physical mixture has only two phases.

Vapor Phases

Deviation from ideality in vapor phases is often expressed through the use of fugacity coefficients:

$$g(\mathbf{x}, z) - g^{I}(\mathbf{x}) = \ln \phi(\mathbf{x}, z),$$

where $\phi(\mathbf{x}, z)$ is the fugacity coefficient of the mixture. The standard state is usually assumed to be an ideal gas at T and unit fugacity. The compressibility $z = pv/RT$ measures deviation from the ideal gas law, and an expression for it is required to calculate $\phi(\mathbf{x}, z)$. For an ideal gas, $z = 1$; otherwise, z is often obtained from an equation of state (EOS) which correlates the temperature, pressure, volume and the composition of nonideal mixtures. This equation of state then becomes an additional constraint (typically nonlinear and nonconvex) that must be obeyed over all compositions. One possible generalized equation of state can be written in its standard form as:

$$z - \alpha B - \frac{z - \alpha B}{z - B} + \frac{A}{z + \beta B} = 0, \qquad (1)$$

$$A = \sum_{i\in C}\sum_{j\in C} A_{ij} x_i x_j , \quad B = \sum_{i\in C} B_i x_i , \qquad (2)$$

where α and β are constants that depend on the equation of state employed. The more important equations of state include the van der Waals ($\alpha = \beta = 0$), Soave–Redlich–Kwong ($\alpha = 0$, $\beta = 1$), and Peng–Robinson ($\alpha = \sqrt{2} - 1$, $\beta = \sqrt{2} + 1$). See [26] for a thorough review. Note that (1) is composed of the sum of a linear fractional and a bilinear fractional function, and that (2) defines A as a bilinear function. This means that when an equation of state is used, an additional level of complexity is added to the problem in the form of nonconvex and nonlinear constraints.

As is demonstrated in several standard texts [26], the overall mixture fugacity coefficient can be obtained using (1) as:

$$\ln\phi(\mathbf{x}, z) = (z-1) - \ln(z-B) + \frac{1}{(\alpha+\beta)}\frac{A}{B}\ln\frac{z-\alpha B}{z+\beta B}\,.$$

This function is highly nonlinear and nonconvex, consisting of a bilinear fractional function (A/B) multiplying the logarithm of a linear fractional function.

Obtaining Equilibrium Solutions

Here the global minimum of the total Gibbs function is sought subject to the material balance constraints. Because the total Gibbs function is used, extensive variables are appropriate. Following [23], assume there are π phase classes characterized by a separate thermodynamic model. π_{EOS} represents the phase class where an EOS is used. Before solving the problem, P_π, the number of phases consistent with phase class π, must be selected. $P = \cup_\pi P_\pi$. The solution will then yield $P_\pi^{eq} \leq P_\pi$ where P_π^{eq} is the number of phases of class π present in nonzero amounts at equilibrium. Consider a potential LLV mixture: if the NRTL is used to model two liquid phases, and the Peng–Robinson equation for a single vapor phase, then π_1 = NRTL, P_{π_1} = 2; π_2 = PR, P_{π_2} = 1. If the actual physical mixture at equilibrium is calculated as LV, then $P_{\pi_1}^{eq} = P_{\pi_2}^{eq} = 1$. The phase rule [26] gives an upper bound on the number of possible phases. The optimization formulation can now be written as:

$$(\mathrm{G})\begin{cases}\min\limits_{n\in N} & G = \sum\limits_{p\in\pi}\sum\limits_{k\in P_\pi} G^k\\ \text{s.t.} & \text{EOS}^k = 0, \quad \forall k \in P_{\pi_{\text{EOS}}},\end{cases}$$

where

$$N = \left\{\mathbf{n}\colon\ \sum_k n_i^k = n_i^T,\ \forall i, \quad n_i^k \geq 0,\ \forall i, k\right\}.$$

Here, n_i^T is the total number of moles of component i in the mixture. Note that the equation of state in (G) comprising (1) and (2) is assumed to be written in extensive form.

Equation Based Approaches

Even though (G) is naturally expressed as an optimization problem, equation based approaches are by far the most prevalent due to their use in commercial chemical process simulators. The first order necessary optimality conditions of (G) reduce to a set of nonlinear equations, corresponding to the condition of equality of chemical potentials (μ_i^k):

$$\mu_i^k = \mu_i^{k'}, \quad \forall i \in C,\ \forall k, k' \in P. \tag{3}$$

All chemical engineering undergraduates encounter the direct iteration K-value method for solving (3), known as the single stage flash calculation. A general description is supplied by [12]. The inside-out algorithm of [2] is of especial prominence due to its superior performance to other methods. Because these equations are nonconvex, there may be several solutions which satisfy them, and these methods are prone to failure, especially at conditions close to the critical point (which is called the plait point for liquid phases).

Local Optimization

Given the problems associated with the equation based approaches, various attempts to solve (G) using local optimization have been attempted. A steepest descent method was used in [27] and is known as the RAND method. Various methods were compared to an implementation of Wolfe's quadratic programming algorithm in [5]. A variable projection method was used in [3]. Several other variants of Newton based methods have been employed (see [17] for a brief summary). None of these methods—typically Newton or quasi-Newton algorithms—removes the possibility of converging to a local optimum, or a trivial solution (saddle point where the mol fractions in two phases of the same class π are the same), and are highly dependent on starting point. A major problem is that P_π is unknown and must be guessed, and therefore, the incorrect number of phases P_π^{eq} is easily obtained with these methods. Another key problem in these approaches is the development of numerical singularities when phases coalesce or split as the algorithm progresses [22].

Global Optimization

The above facts motivate the employment of global optimization techniques because if an approach can be guaranteed to obtain the global optimum solution of (G), then a sufficiency condition for phase equilibrium is automatically supplied. The first use of global optimization to solve (G) was undertaken in [17], where it was shown how the GOP algorithm [4] could be used in cases where the NRTL equation was used to model liquid phase behavior. New variables were introduced so that the formulation of (G) would consist of a biconvex objective function and a bilinear constraint set, satisfying the requirements of the GOP to guarantee global optimality. For the UNIQUAC equation, a branch and bound global optimization algorithm described in [10] was implemented to determine the global minimum of (G) [15]. A key aspect of the work in [17] was the mathematical transformation of the nonconvex expressions for the Gibbs free energies into forms with special structure, namely the difference of two convex function (d.c. form). Similar transformations and this same algorithm can be also applied to the UNIFAC, ASOG and modified Wilson equations, as shown in [16]. These were the first approaches to guarantee convergence to the global solution of (G), regardless of the supplied initial point.

Verifying Equilibrium Solutions

The tangent plane criterion provides an alternative sufficiency condition for a candidate equilibrium solution to correspond to a global minimum of the Gibbs free energy [7]. A candidate solution must satisfy the necessary condition for equilibrium—that is, satisfy (3). Stability requires that the tangent hyperplane constructed using the chemical potential values of the candidate solution (denoted $\mu_i^{\dagger}$) at no point lies above the molar Gibbs surface for all phase classes used to model the mixture. Stated in optimization terms, if the global minimum of the tangent plane distance function, D_{π}, for each phase class π used to represent the behavior of the mixture, is nonnegative $\forall \pi$, then the candidate solution corresponds to a global minimum of the Gibbs free energy [1]. The phase stability problem is defined for a phase class π as:

$$(S)\begin{cases} \min\limits_{x \in X} & D^{\pi} = g^{\pi} - \sum\limits_{i \in C} x_i \mu_i^{\dagger} \\ \text{s.t.} & \mathrm{EOS}(\mathbf{x}, z) = 0 \quad \text{if } \pi_{\mathrm{EOS}} \subset \pi, \end{cases}$$

where $X = \{\mathbf{x}\colon \sum_i x_i = 1,\ x_i \geq 0,\ \forall i\}$. Clearly, (1) and (2) are required for (S) when $\pi_{\mathrm{EOS}} \subset \pi . g^{\pi}$ is obtained from the appropriate thermodynamic models described earlier. Therefore, it is seen that the approach involves verifying that a candidate solution is the equilibrium one.

Equation Based Approaches

As with (G), the first order necessary optimality conditions of (S) reduce to a set of nonlinear equations:

$$\mu_i^{\pi} - \mu_i^{\dagger} = K, \quad \text{s.t. } \mathbf{x} \in X, \tag{4}$$

where K is a constant. The EOS must be satisfied if $\pi_{\mathrm{EOS}} \subset \pi$. If a nonnegative solution to this set of equations is obtained, then the postulated solution is assumed to be stable. Standard direct iteration methods have been used [20] as well as homotopy continuation methods [24] to solve (4). However, no guarantee of obtaining all stationary points can be provided with the typical equation based approach. However, an interval Newton method has been used in [11] to ϵ-enclose all stationary points. This work can be considered a 'global' method for equation solving. It should be noted that a branch and bound global optimization algorithm [13] has been used to obtain all homogeneous azeotropes in mixtures [9]; because the condition of azeotropy adds a single linear constraint (equality of mol fractions in all phases) to (3), this approach can in principle be used to guarantee obtaining all ϵ-global solutions to both (3) and (4).

Global Optimization

The advantage of a global optimization approach is that if a nonnegative solution is found, then it can be definitively asserted that the candidate solution is the globally stable equilibrium one, unlike available local algorithms. It is shown in [18] how global optimization can be used to solve (S), using the GOP algorithm for the NRTL equation, and a branch and bound algorithm for the UNIQUAC equation. For the modified Wilson, ASOG and UNIFAC equations, it was shown in [16] how this same branch and bound algorithm could be used after transforming the expressions for $g(\mathbf{x})$ into d.c. form. It has been shown how the formulations for (G) and (S) involving equations of state can be transformed into biconvex form allowing the application of a number of global optimization algorithms [14], although no implementation was undertaken. An important recent extension of global optimization to the case of equations of state is supplied in [8] where the nonlinear terms are validly underestimated within the framework of a branch and bound algorithm.

Combining Approaches

From the above development, it is apparent that:

1) To obtain a candidate equilibrium solution, either solve (G) or (3); and
2) To verify a candidate as the equilibrium solution, either solve (S) or (4).

Approach 1) is problematic because the a priori selection of P_π represents a formidable challenge. If too few phases are allowed, then convergence to constrained minima can occur; if too many are assumed, then numerical problems may arise, or convergence to trivial or local extrema may occur. Therefore, the concept of using the tangent plane criterion to provide initial guesses for (G) or (3) has been shown to greatly increase reliability with a tolerable increase in computational effort. In addition, when solving (G) or (3), the number of composition variables is $N_V = |C||P|$, while for (S) or (4), $N_V = |C|$. The performance of the RAND method was found to considerably improve when combined with a phase-splitting algorithm [6]. The seminal work of M.L. Michelsen [20] proposed an iterative approach whereby the solution from the tangent plane criterion is used to initialize the search for the equilibrium solution. This is implemented using a direct substitution method (K-value approach) as well as an optimization method. The calculations are computationally efficient and reported to be quite reliable, although there is the danger of predicting a stable phase distribution, when, in fact, this is not the case. In a comparative study for liquid-liquid phase splitting [25], Michelsen's method was found to be the most reliable. A similar iterative approach using homotopy continuation methods to solve (4) have also been used in [24]. However, there are a number of difficulties associated with these approaches. First, no guarantee of obtaining all stationary points can be provided. Second, since the solutions obtained from the stability problem are then used to initiate the search for a solution with a lower Gibbs free energy, these guesses may lead to local optima, or even infeasible equilibrium solutions. Therefore, no guarantees can be made of having obtained the equilibrium solution, even though overall reliability is significantly increased.

Global Optimization

When solving (G) using global optimization, the maximum allowable number of phases $P_\pi, \forall\pi$, must be considered for rigorous determination of phase and chemical equilibrium. This leads to high computational effort when often the global solution is generated early in the global optimization search [19]. For these reasons, an algorithm known as GLOPEQ (global optimization for the phase and chemical equilibrium problem) was implemented in [19]. An iterative approach was proposed based on the fact that solving (S) to global optimality to verify a candidate solution is vastly preferable to solving (G). GLOPEQ therefore leads to significant computational savings over other global optimization approaches. It should be noted that the approach described in [8] can be incorporated into GLOPEQ, extending its applicability and giving the first global optimization method for both nonideal liquid and vapor phases. The key difference between GLOPEQ and the other local iterative approaches is that global optimization is used at each step of the algorithm, allowing a guarantee to be made of obtaining the true equilibrium solution no matter the starting point.

Reaction Equilibria

If reaction occurs in the mixture, then the permissible regions N and X must be adjusted. See [23] for an elegant analysis of the tangent plane criterion for reacting mixtures. Note that N and X remain linear and they do not affect the global optimization approach for solving (G) or (S).

General Comments on Efficiency

Clearly local approaches, while less reliable, are more efficient than global optimization approaches. Because of the relatively heavy computational burden of global optimization, these approaches are more justified for off-line analysis as they could not be practically used in a chemical process simulator. Having said that, computational times of seconds for highly nonideal mixtures of up to eight components [8] provide a great deal of promise for improving the robustness of the equilibrium calculation without resulting in excessive solution times.

See also

- ▶ αBB Algorithm
- ▶ Continuous Global Optimization: Models, Algorithms and Software
- ▶ Generalized Primal-relaxed Dual Approach
- ▶ Global Optimization: Application to Phase Equilibrium Problems
- ▶ Global Optimization in Batch Design Under Uncertainty
- ▶ Global Optimization in Generalized Geometric Programming
- ▶ Global Optimization Methods for Systems of Nonlinear Equations
- ▶ Interval Global Optimization
- ▶ MINLP: Branch and Bound Global Optimization Algorithm
- ▶ MINLP: Global Optimization with αBB
- ▶ Optimality Criteria for Multiphase Chemical Equilibrium
- ▶ Smooth Nonlinear Nonconvex Optimization

References

1. Baker LE, Pierce AC, Luks KD (1982) Gibbs energy analysis of phase equilibria. Soc Petrol Eng J 731
2. Boston JF, Britt HI (1978) A radically different formulation and solution of the single stage flash problem. Comput Chem Eng 2:109–122
3. Castillo J, Grossmann IE (1981) Computation of phase and chemical equilibria. Comput Chem Eng 5:99
4. Floudas CA, Visweswaran V (1993) A primal-relaxed dual global optimization approach. JOTA 78(2):187
5. Gautam R, Seider WD (1979) Computation of phase and chemical equilibrium, Part I: Local and constrained minima in Gibbs free energy. AIChE J 25(6):991
6. Gautam R, Seider WD (1979) Computation of phase and chemical equilibrium, Part II: Phase-splitting. AIChE J 25(6):999
7. Gibbs JW (1873) A method of geometrical representation of the thermodynamic properties of substances by means of surfaces. Trans Connecticut Acad 2:382–404
8. Harding ST, Floudas CA (2000) Phase stability with cubic equations of state: A global optimization approach. AIChE J
9. Harding ST, Maranas CD, McDonald CM, Floudas CA (1997) Locating all homogeneous azeotropes in multicomponent mixtures. I-EC Res 36:160–178
10. Horst R, Tuy H (1992) Global optimization, 2nd edn. Springer, Berlin
11. Hua JZ, Brennecke JF, Stadtherr MA (1998) Reliable computation of phase stability using interval analysis: Cubic equation of state models. Comput Chem Eng 22(9):1207–1214
12. King CJ (1980) Separation processes, 2nd edn. McGraw-Hill, New York
13. Maranas CD, Floudas CA (1995) Finding all solutions of nonlinearly constrained systems of equations. JOGO 7:143–182
14. McDonald CM (1999) A novel reformulation of the phase equilibrium problem when using equations of state and subsequent application of global optimization. Presented at AIChE Annual Meeting
15. McDonald CM, Floudas CA (1994) Decomposition based and branch and bound global optimization approaches for the phase equilibrium problem. J Global Optim 5:205–251
16. McDonald CM, Floudas CA (1995) Global optimization and analysis for the Gibbs free energy function using the UNIFAC, Wilson and ASOG equations. I-EC Res 34:1674
17. McDonald CM, Floudas CA (1995) Global optimization for the phase and chemical equilibrium problem: Application to the NRTL equation. Comput Chem Eng 19(11):1111
18. McDonald CM, Floudas CA (1995) Global optimization for the phase stability problem. AIChE J 41(7):1798
19. McDonald CM, Floudas CA (1997) GLOPEQ: A new computational tool for the phase and chemical equilibrium problem. Comput Chem Eng 21(1):1–23
20. Michelsen ML (1982) The isothermal flash problem. Part I. Stability. Part II. Phase–split calculation. Fluid Phase Equilib 9:1–40
21. Reid RC, Prausnitz JM, Poling BE (1987) The properties of gases and liquids, 4th edn. McGraw-Hill, New York
22. Seider WD, Brengel DD, Widagdo S (1991) Nonlinear analysis in process design. AIChE J 37(1):1
23. Smith JV, Missen RW, Smith WR (1993) General optimality criteria for multiphase multireaction chemical equilibrium. AIChE J 39(4):707
24. Sun AC, Seider WD (1995) Homotopy-continuation method for stability analysis in the global minimization of the Gibbs free energy. Fluid Phase Equilib 103:213
25. Swank DJ, Mullins JC (1986) Evaluation of methods for calculating liquid-liquid phase-splitting. Fluid Phase Equilib 30:101
26. Walas SM (1985) Phase equilibria in chemical engineering. Butterworths, London
27. White WB, Johnson SM, Dantzig GB (1958) Chemical equilibrium in complex mixtures. J Chem Phys 28(5):751

Global Optimization of Planar Multilayered Dielectric Structures

CLAIRE S. ADJIMAN[1], R.F. OULTON[2]
[1] Centre for Process Systems Engineering,
Dept. of Chemical Engineering,
Imperial College London, London, UK
[2] NSF Center for Scalable
and Integrated Nanomanufacturing,
University of California at Berkeley, Berkeley, USA

MSC2000: 65K99

Article Outline

Introduction

Multi-layered dielectric structures are relevant in many applications that seek to influence electromagnetic radiation across the infrared, optical and X-ray spectra. Anti-reflection coatings, components for integrated optics and semiconductor lasers are based on multilayered dielectric designs; they are generally modeled using the transfer matrix method that has been in widespread use for the past thirty years [5,26]. In many cases optical designs can be devised by deductive reasoning, but, as design objectives have become more elaborate, robust numerical optimization techniques have become increasingly relevant. Baumeister reported the first refinement technique for multilayer dielectric in 1958 [3].

The synthesis of multi-layered dielectric structure designs requires a robust global optimization approach. The mathematical model that describes the optical properties of these structures is highly non-linear and presents any solver the task of sifting through countless local minima. Early approaches relied on stochastic global methods. The lack of deterministic methods in the literature highlights the challenging mathematical task of identifying minimizing convex approximations. As far as the authors know, the only deterministic approach proposed to date is limited in scope due to approximations that are made to derive model equations such that the problem has a unique solution.

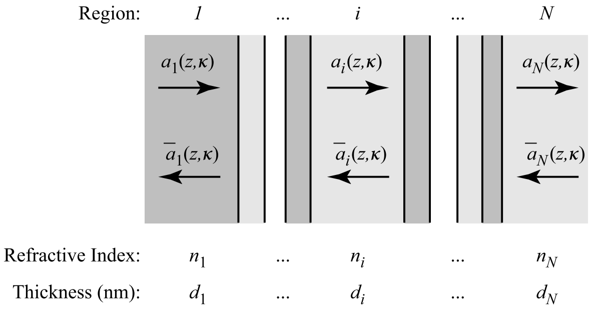
This encyclopedia entry examines the problem of multilayer dielectric design, which has been treated with a range of algorithms over the past 20 years. Stochastic approaches are reviewed including Simulated Annealing, Genetic Algorithms and a Multi-Level approach. A deterministic minimization approach is also discussed. The study may be considered a review and critical comparison of techniques for electromagnetic filter design.

Formulation

Statement of Physical Problem

Multilayered dielectric structures have two modes of operation: in passive mode, a structure reflects or transmits light from an external source as a function of the input wavelength and direction; in active mode, a structure creates light internally and distributes the emission both spectrally and spatially. Figure 1 illustrates these two modes of operation. Here, $a_i^{(\mu)}(\kappa, z_i)$ and $\bar{a}_i^{(\mu)}(\kappa, z_i)$ are the forward and backward propagating amplitudes in Region i respectively. The superscript in brackets ($\mu = \{s, p\}$) indicates the polarization, which is described as either Transverse Electric (s) or Transverse Magnetic (p).

In the passive geometry, amplitudes are equated with real measurable quantities: $|a_1(\kappa, z_1)|^2 = 1$, $|\bar{a}_1(\kappa, z_1)|^2 = R(\kappa)$, $|a_N(\kappa, z_N)|^2 = T(\kappa)$ and $|\bar{a}_N(\kappa, z_N)|^2 = 0$, where $R(\kappa)$ and $T(\kappa)$ are the reflectivity and transmissivity of the structure respectively. In active mode $|a_1(\kappa, z)|^2 = 0$, $|\bar{a}_1(\kappa, z_1)|^2 = P_b(\kappa)$, $|a_N(\kappa, z_N)|^2 = P_f(\kappa)$, $|\bar{a}_N(\kappa, z_N)|^2 = 0$ where P_f and P_b are the forward and backwards emission powers. The source amplitudes, $a_i(\kappa, z_i) = A(\kappa)$ and $\bar{a}_2(\kappa, z_i) = \bar{A}(\kappa)$ are dependent on the type of emission source and will not be elaborated upon here. For more information on these issues, the reader should refer to [10] and [4]. In both operation modes, the structure interacts with



Global Optimization of Planar Multilayered Dielectric Structures, Figure 1
Schematic of a multilayer dielectric structure highlighting the nomenclature …

light as a function of its wavelength, λ, and the optical angle parameter, $\kappa = n_i \sin\theta_i$, which is related to the propagation angle, θ_i, in region i. Note that κ is invariant throughout the structure unlike θ_i, which varies with the refractive index, n_i, due to Snell's law ($n_i \sin\theta_i = \text{constant}$).

The following analysis considers only the passive mode of operation, although the approach is readily adaptable to describe problems involving the active mode. Therefore, $R(\kappa)$ and $T(\kappa)$ are usually part of some expression to be minimized. The design variables to be optimized are the refractive indices, n_i, and layer thicknesses, d_i, throughout the structure. The optimization problem is posed as follows: a single valued objective function $\mathcal{F}\{R(\kappa;\mathbf{n},\mathbf{d}), T(\kappa;\mathbf{n},\mathbf{d})\}$ involving the reflectivity and transmissivity must be minimized subject to unknown variables $\mathbf{n} = \{n_i\}$ and $\mathbf{d} = \{d_i\}$ where $i \in \{1,\dots,N\}$. The problem is typically bounded, defining a variable space of finite extent: for example, the unknown variables here are constrained to upper, $\mathbf{n}^U$, $\mathbf{d}^U$ and lower, $\mathbf{n}^L$, $\mathbf{d}^L$ bounds. This is summarized as,

$$\begin{aligned} &\min_{\mathbf{n},\mathbf{d}} \mathcal{F}\{R(\kappa;\mathbf{n},\mathbf{d}),\ T(\kappa;\mathbf{n},\mathbf{d})\} \\ s.t.\quad &\mathbf{n} - \mathbf{n}^U \le 0 \\ &\mathbf{n}^L - \mathbf{n} \le 0 \\ &\mathbf{d} - \mathbf{d}^U \le 0 \\ &\mathbf{d}^L - \mathbf{d} \le 0\,. \end{aligned} \tag{1}$$

The use of the transfer matrix method to describe the propagation of light through multi-layer planar dielectric materials is well-established [5,26]. Oulton and Adjiman [28] present an alternative and more compact representation highlighting the mathematical details of the model and symmetries that are useful for writing efficient code and deriving compact analytical gradients for local optimization.

Consider the schematic for a general multilayered structure in Fig. 1. The transfer matrix, $\mathbf{T}^{(\mu)}_{ji}(\kappa)$, relates the electromagnetic field amplitudes in regions i and j at z_i as follows,

$$\begin{aligned} &\begin{pmatrix} a^{(\mu)}_j(\kappa,\, z_i) \\ \bar{a}^{(\mu)}_j(\kappa,\, z_i) \end{pmatrix} \\ &= \begin{pmatrix} X^{(\mu)+}_{j,\,i}(\kappa) & X^{(\mu)-}_{j,\,i}(\kappa) \\ X^{(\mu)-}_{j,\,i}(\kappa) & X^{(\mu)+}_{j,\,i}(\kappa) \end{pmatrix} \begin{pmatrix} a^{(\mu)}_i(\kappa,\, z_i) \\ \bar{a}^{(\mu)}_i(\kappa,\, z_i) \end{pmatrix} \\ &= \mathbf{T}^{(\mu)}_{ji}(\kappa) \begin{pmatrix} a^{(\mu)}_i(\kappa,\, z_i) \\ \bar{a}^{(\mu)}_i(\kappa,\, z_i) \end{pmatrix} \end{aligned} \tag{2}$$

$$X^{(\mu)\pm}_{j,\,i} = \frac{1}{2}\left(C^{(\mu)}_{i,\,j} \pm \frac{1}{C^{(\mu)}_{i,\,j}} \right)\,. \tag{3}$$

The coupling coefficients, $C^{(\mu)}_{i,\,j}$ are

$$\begin{aligned} C^{(s)}_{i,\,j} &= \sqrt{\frac{\beta_j}{\beta_i}} \\ C^{(p)}_{i,\,j} &= \frac{n_j}{n_i}\sqrt{\frac{\beta_i}{\beta_j}}\,. \end{aligned} \tag{4}$$

Here, $\beta_i = k_0\sqrt{n_i^2 - \kappa^2}$ is related to κ and λ through $k_0 = 2\pi/\lambda$, the wavenumber of the incident light. β_i is the component of the wavenumber normal to the planar layers and is sometimes referred to as the propagation constant. Note that $\mathbf{T}^{(\mu)}_{ji}(\kappa)$ is symmetric with only two independent elements.

To describe propagation across region j, of thickness $d_j = z_j - z_i$, the amplitudes, $a_j(\kappa,\, z_j)$ and $\bar{a}_j(\kappa,\, z_j)$ at z_j are related to the amplitudes, $a_j(\kappa,\, z_i)$ and $\bar{a}_j(\kappa,\, z_i)$ at z_i by the transfer matrix, $\mathbf{P}_j(\kappa)$, which is independent of polarization for isotropic materials.

$$\begin{aligned} &\begin{pmatrix} a^{(\mu)}_j(\kappa,\, z_j) \\ \bar{a}^{(\mu)}_j(\kappa,\, z_j) \end{pmatrix} \\ &= \begin{pmatrix} e^{\mathrm{i}\beta_j d_j} & 0 \\ 0 & e^{-\mathrm{i}\beta_j d_j} \end{pmatrix} \begin{pmatrix} a^{(\mu)}_j(\kappa,\, z_i) \\ \bar{a}^{(\mu)}_j(\kappa,\, z_i) \end{pmatrix} \\ &= \mathbf{P}_j(\kappa) \begin{pmatrix} a^{(\mu)}_j(\kappa,\, z_i) \\ \bar{a}^{(\mu)}_j(\kappa,\, z_i) \end{pmatrix}\,. \end{aligned} \tag{5}$$

In order to relate the fields at the interfaces between regions $i+2$ and $i+1$ at z_{i+1} and regions $i+1$ and i at z_i, interface and propagation matrices are multiplied together such that,

$$\begin{pmatrix} a_{i+1}(\kappa,\, z_{i+1}) \\ \bar{a}_{i+1}(\kappa,\, z_{i+1}) \end{pmatrix} = \mathbf{M}^{(\mu)}_{i+1,\,i}(\kappa) \begin{pmatrix} a_i(\kappa,\, z_i) \\ \bar{a}_i(\kappa,\, z_i) \end{pmatrix} \quad (6)$$

where $\mathbf{M}^{(\mu)}_{i+1,\,i}(\kappa) = \mathbf{P}_{i+1}(\kappa)\mathbf{T}^{(\mu)}_{i+1,\,i}(\kappa)$. In the general formulation, the amplitudes can be expressed as a vector of plane wave modes corresponding to the various angles of propagation within the planar dielectric medium. When considering N values of κ (angles of incidence), the transfer matrix, $\mathbf{M}^{(\mu)}_{i+1,\,i}$ will be a $2N \times 2N$ matrix. Notice, however, that due to Snell's law and the law of reflection $\mathbf{M}^{(\mu)}_{i+1,\,i}$ is sparse with only $4N$ components along the diagonals of each quadrant of $\mathbf{M}^{(\mu)}_{i+1,\,i}$. Therefore there are only $2N$ independent components. From here on, the parameter κ will be dropped from the mathematical expressions for brevity.

Analytical Gradients for Effective Optimization

The efficiency and accuracy of local optimization can be enhanced by using analytically determined gradients. Methods for determining the gradients of transfer matrices have been presented in the literature [29], but here the new formalism allows a compact analytical evaluation which leads to simplified coding and efficient strategies for calculating large numbers of gradients for one structure.

Consider the derivative of the standard mode matching matrix equation with respect to the variable, ζ_i for an optical structure with N layers:

$$\frac{\partial}{\partial\zeta_i}\begin{pmatrix} a^{(\mu)}_N(z_N) \\ \bar{a}^{(\mu)}_N(z_N) \end{pmatrix} = \frac{\partial\mathbf{M}^{(\mu)}_{N,1}}{\partial\zeta_i}\begin{pmatrix} a^{(\mu)}_1(z_1) \\ \bar{a}^{(\mu)}_1(z_1) \end{pmatrix} + \mathbf{M}^{(\mu)}_{N,1}\frac{\partial}{\partial\zeta_i}\begin{pmatrix} a^{(\mu)}_1(z_1) \\ \bar{a}^{(\mu)}_1(z_1) \end{pmatrix}. \quad (7)$$

Given any two constant boundary condition amplitudes the matrix equation can be solved for the derivatives of the unknown amplitudes. Consider now the derivative of the matrix with respect to the variables of a given layer.

The derivative with respect to d_i is the easiest to evaluate as only one phase matrix, $\mathbf{P}_i$, needs to be differentiated. The matrix derivative is:

$$\begin{aligned}\frac{\partial\mathbf{M}^{(\mu)}_{N,1}}{\partial d_i} &= \mathbf{M}^{(\mu)}_{N,\,i}\mathbf{dM}^{(\mu)}_{di}\mathbf{M}^{(\mu)}_{i,\,1} \\ &= \mathrm{i}k_{z\,i}\mathbf{M}^{(\mu)}_{N,\,i}\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}\mathbf{M}^{(\mu)}_{i,\,1}\,.\end{aligned} \quad (8)$$

Differentiation with respect to the refractive index, n_i, is much more complicated as it involves the product of three matrices and must be evaluated using the Leibniz rule. In the current representation, transfer matrix symmetries can be exploited to give a concise form as in Eq. (8), which can be written for the two polarizations as follows:

$$\frac{\partial\mathbf{M}^{(\mu)}_{N,1}}{\partial n_i} = \mathbf{M}^{(\mu)}_{N,\,i}\mathbf{dM}^{(\mu)}_{n_i}\mathbf{M}^{(\mu)}_{i,\,1} \quad (9)$$

where,

$$\begin{aligned}\mathbf{dM}^{(s)}_{n_i} &= \frac{n_i k_0^2}{\beta_i^2}\begin{pmatrix} \mathrm{i}\beta_i d_i & \frac{1}{2}\left\{e^{2\mathrm{i}\beta_i d_i}-1\right\} \\ \frac{1}{2}\left\{e^{-2\mathrm{i}\beta_i d_i}-1\right\} & -\mathrm{i}\beta_i d_i \end{pmatrix} \\ \mathbf{dM}^{(p)}_{n_i} &= \frac{n_i k_0^2}{\beta_i^2} \\ &\begin{pmatrix} \mathrm{i}\beta_i d_i & \frac{1}{2}\frac{\beta_i^2-k_x^2}{k_0^2 n_i^2}\left\{e^{2\mathrm{i}\beta_i d_i}-1\right\} \\ \frac{1}{2}\frac{\beta_i^2-k_x^2}{k_0^2 n_i^2}\left\{e^{-2\mathrm{i}\beta_i d_i}-1\right\} & -\mathrm{i}\beta_i d_i \end{pmatrix}.\end{aligned} \quad (10)$$

These are extremely concise forms for the matrix derivatives of fairly complicated expressions where each gradient only requires the evaluation of a supplementary transfer matrix, $\mathbf{dM}^{(\mu)}_{\zeta_i}$ and one additional matrix evaluation.

The reflectivity, R and transmissivity, T, involve the absolute square of the field amplitudes. Given the derivative of the amplitude, $a_i(z_i)$, the derivative of its absolute square is given by,

$$\begin{aligned}\frac{dR}{d\zeta_i} &= \frac{d(a_i(z_i)a_i(z_i)^*)}{d\zeta_i} \\ &= 2\Re\{a_i(z_i)\}\Re\left\{\frac{da_i(z_i)}{d\zeta_i}\right\} \\ &\quad + 2\Im\{a_i(z_i)\}\Im\left\{\frac{da_i(z_i)}{d\zeta_i}\right\}.\end{aligned} \quad (11)$$

Methods and Applications

By the early 1990s, a range of optimization methods were being used to generate optical multi-layer filter designs. Dobrowolski and co-workers reviewed ten methods for computational speed and effectiveness at reaching an optimum solution to determine which would be best suited to these highly non-linear problems [11]. Amongst these were both global and local methods but, at this time of limited computer power, no particular approach was deemed superior over the fixed calculation time of 2 hours. The authors rated a local gradient-based modified Gauss-Newton method highly for its consistency over the problems investigated. The observation that the global optimization methods performed on a par with local methods, despite the clear limitation in fixed calculation time, was also noted.

Computing power today is not as great an issue and the use of global optimization techniques for multi-layered optical design has attracted a great deal of attention (see references throughout this Section). Global algorithms can be broadly divided into two categories: deterministic and stochastic. Deterministic methods guarantee a global solution, usually at the expense of calculation time, whereas stochastic methods converge rapidly to solutions with only a probabilistic guarantee of global optimality in finite time. Liberti and Kucherenko investigated these contrasting philosophies by comparing the deterministic spatial Branch and Bound (sBB) and Stochastic Multi Level Single Linkage (MLSL) methods for a range of test functions [24]. The authors concluded that the stochastic method, in the cases studied, converged faster to a global optimum with a high degree of probability, but the deterministic method could perform better in cases where specific theoretical assumptions about a problem's analytical structure could be taken into account. In general, deterministic methods require preparation for a particular problem, whereas stochastic methods can be more readily adapted for *black-box* scenarios. Nevertheless, stochastic approaches cannot guarantee global optimality in finite time.

In this section, a range of global optimization approaches are reviewed. It is most useful that in the study of these methods, some authors have examined the same numerical synthesis problem: the design of an anti-reflection coating to operate in the far infrared [1,6,11,25,28,36,44]. The objective is to minimize a Germanium (Ge: refractive index $n_{Ge} = 4.2$) and Zinc Oxide (ZnS: refractive index $n_{\mathrm{ZnS}} = 2.2$) multi-layered structure to achieve a normal incidence reflectivity, $R(\kappa = 0) \mapsto 0$ for $N_\lambda = 47$ equidistant wavelengths in the spectral band $7.7 \leq \lambda \leq 12.3$ μm. The incident medium is air and the substrate which the structure is built on has refractive index $n_{\mathrm{Sub}} = 4$.

The objective function, $\mathcal{F}(\mathbf{d}, \lambda_i, R_i)$, was chosen to be the same as that used by authors in the literature to allow comparative studies.

$$\mathcal{F}(\mathbf{d}, \lambda_i, R_i) = \left[\frac{1}{N_\lambda} \sum_{i=1}^{N_\lambda} R_i(\lambda_i)^2 \right]^{-1/2} . \qquad (12)$$

In the following studies, the optimum layer thicknesses for reproducing the best designs are omitted for brevity, so the reader should consider the relevant references for this information.

Multi-Level Approaches

In Multi-Level (ML) approaches (e. g., [20,21,23,24,28]), different starting points are generated by a higher-level algorithm, and the problem is solved from each starting point by a lower-level local optimization algorithm. This approach is very general because it requires no tuning. It has been applied by Oulton and Adjiman [28] to the design of multi-layered dielectric device design by using a deterministic sampling of the parameter space and local nonlinear programming (NLP) solver. The approach is able to rank many local solutions for post-optimization analysis. It is also non-adaptive at the global level in that the algorithm depends only on the current state and not on previously calculated states. This brings two advantages: firstly, non-adaptive methods are deemed superior to adaptive ones in multi-processor applications, which is certainly a benefit for computationally intensive global optimization problems. Secondly, non-adaptive algorithms have freedom over the specification of convergence criteria. Since the optimization algorithm in [28] essentially operates by batch local optimization, it can be halted according to criteria such as the number of global iterations or after a set time limit. Rigorous criteria are also applicable to the ML strategy [20,21,23]: as the algorithm progresses the probability that the current best local solution is the global one increases, primarily due to the global search coverage guaranteed through the appropriate choice of the sampling approach; for instance, Oulton and Adjiman used a Sobol' sequence [33], a deterministic Low Discrepancy Sequence (LDS) which provides a *good* coverage of the variable space. The Sobol' LDS was selected because its construction is based on i) homogeneity as the number of sample points, $n \mapsto \infty$, ii) good distribution for small n and iii) fast computational algorithm. All these features, but particularly ii), make this LDS most applicable to the current problem. There are a variety of LDSs that are constructed on differing requirements such as Holton, Faure, Niederreiter and Sobol' amongst others [7,18,27,33].

Simulated Annealing

Simulated annealing (SA) has been applied to a variety of electromagnetic multilayer design problems in the infra-red, ultra-violet and X-ray spectra [6,8,9,15,22,41,42]. SA [22] operates by randomly changing an initial design in small steps and accepting the changes based on an evaluation of the new design performance according to criteria that become increasingly stringent as the algorithm progresses. Changes are always accepted if they result in a better design. On the other hand, a worse design is accepted with a probability based on a Boltzmann distribution. The probability of acceptance is tuned by changing the Boltzmann temperature according to a user-specified schedule. Wider exploration of the variable space at the start of the optimization is achieved by setting a high temperature, which essentially allows the algorithm to accept worse designs and thereby move between local regions of attraction. A cooling schedule restricts the algorithm's ability to investigate adjacent local regions and forces convergence to a local optimum. In this case, it is clear that convergence to the global optimum will be dependent on the initial design and especially on the cooling schedule.

The first reports on SA applied to multilayer design highlighted mainly the technique's ability to avoid local minima [41], although adaptations to avoid deep local minima were also reported [8]. These reports were for structure in the visible to near infrared spectra. The method has recently seen use in the design of reflectors for UV [15] and X-ray [9] spectra. These have applications that include neutron optics, X-ray astrophysics and synchrotron radiation. In this region of the spectrum, matter interacts with electromagnetic radiation differently requiring a modified transfer matrix description that accounts for surface roughness and interdiffusion (See [9] and references therein). Wu et al. have applied simulated annealing to a different optics problem involving diffraction gratings [42]. This important design problem concerns the efficient coupling of light into and out of waveguides and optical interconnects.

Boudet et al. [6] use SA to synthesize multilayer designs for the problem given in the introduction to this section. Results for $N_L = 17$ (triangle) and $N_L = 20$ (filled triangle) are shown in Fig. 2b along with results generated using other approaches. The merits of the method are discussed in Sect. "A Comparison of Methods for an Infrared Filter Design".

Genetic and Memetic Algorithms

Evolutionary or Genetic Algorithms (GA) are the preferred method in the optics community [2,14,16,17,19,25,39,43,44,45,46,47]. GAs operate on the principle that the evolution of a random population of parameterizations, subject to iterative rules of reproduction and mutation, will converge to a region of attraction containing the global optimum [17]. Members of the population with high performance are given a greater likelihood of reproducing thereby generating a better population than the one before. Mutation prevents the algorithm converging too quickly and provides the mechanism by which the variable space can be explored more fully. Usually, local optimization is required to refine the final solution. In the case of the Memetic algorithm, local optimization is performed on each new member of the population. This approach could therefore be considered as a multi level approach (see earlier Section).

Eisenhammer et al. [14] optimized the performance of heat mirrors for solar cells: these are high pass filters that transmit optical solar radiation but reflect thermal radiation, which would otherwise be lost by the solar cell. Their designs differ slightly from typical dielectric multilayers since they incorporate metals, which help to reflect thermal frequencies. Bagnoud and Salin [2] and Yakovelev and Tempea [43] have applied GAs to the design of chirpped mirrors, which are used to make ultra-fast lasers with fempto-second pulses. These sophisticated mirrors are designed to have a reflectance over a broad range of frequencies and, in addition, must also be compensated to ensure stability of the reflection phases. Yakovelev and Tempea [43] used a memetic algorithm and report fast convergence compared with the standard GA. Hoorfar et al. [19] developed the GA approach by considering the choice of dielectric materials from a list of candidates for the multilayer design. The authors thereby treat the mixed parameter approach (i. e. discrete and continuous optimization parameters) to which the GA approach appears amenable. Other authors have developed the technique still further by considering multiple objective and constraint functions [39]. The standard infrared filter design problem introduced by Aguilera et al. [1] has been treated by Martin et al. [25,44,46]: the results of these studies are shown in Fig. 2b along with those of other global optimization approaches.

Needle Optimization

Most optimization strategies for multilayer design problems consider the variation of layer thickness d_i and layer refractive index n_i. Variations of the standard techniques including multiple objective functions and mixed parameter optimization have also been discussed in this article. However, few methods consider the variation in the number of layers of a multilayer design problem. The needle optimization approach tackles the problem exactly from this perspective [32,34,35,37,38,40]. Firstly, the optimum position for introducing a needle like layer perturbation to a structure is determined: this usually corresponds to an insertion point that gives optimum convergence of the objective function. Tikhonov Jr et al. [35] provide an algorithm for locating this optimum position before needle insertion. For some objective functions, this is analytically determined, but, for flexibility, numerical approaches are available also [34,40]. Following insertion of a needle, the new design is used as the starting point for a local optimization. The needle insertion and local optimization procedure is repeated until no more refinement is possible within the constraints of the problem at hand.

An alternative approach to this problem, which has not yet been explored, would be to formulate the problem as Mixed-Integer Program, in which the existence or otherwise of each putative layer would be represented by a binary variable. This problem could be solved locally using standard techniques, and many of the global methods discussed here could be applied.

Deterministic Methods

Deterministic algorithms generally require the nonlinear set of model equations to be analyzed to obtain a convex problem which underestimates the minimum of the original design problem. Using one of various search approaches, such as Branch and Bound, it is possible to converge to a feasible global minimum by successively solving such problems, which produce tighter and tighter bounds on the solution along an infeasible design path. These methods are reviewed extensively elsewhere in the Encyclopedia.

Due to the complexity of the highly coupled transfer matrix equations, it is difficult to find appropriate convex estimators. However, Tikhonravov and Dobrowolsky [36] treat the above problem using an approximate infeasible path approach, reducing the problem to a quadratic programming problem with linear inequality constraints with one global optimum solution. Feasible solutions are obtained by local optimization of the resulting design. In their method, the reflection calculation is approximated for $\kappa = 0$ by i) assuming continuous variation of the refractive index profile, and ii) assuming only a small reflectivity, $R(0)$. In the scope of general multilayer optimization problems, these conditions are fairly restrictive, but they are applicable to the filter design problem posed by Aguilera et al. [1]. Strictly, this is not a deterministic global optimization method because it is based on solving an approximate problem to global optimality, and there is no guarantee that this corresponds to the global solution of the original problem. Tikhonravov and Dobrowolsky [36] perform the local optimization of a discretized structure to find a feasible solution. One interesting aspect of their approach is the proof of an optimal relationship between the minimum objective function and the optical thickness of a filter for a given set of material parameters. Although solutions along this line may not exist, the condition marks the limit of global optimality. The limiting condition of optimality is plotted in Fig. 2 and marks a theoretical boundary above which all solutions must lie. This is useful as a benchmark for the development of deterministic global optimization algo-

Global Optimization of Planar Multilayered Dielectric Structures, Table 1
Details of solvers and their implementations in this study. (a) [25]; (b) [6]; (c) [44]; (d) [28]. * Value estimated based on 1600 generations and 100 members per population. ** Value estimated based on computation time and CPU type, taking into account details from [25]. † Number of function evaluations depends on number of layers design and fixed computation time of 5 hrs

	Function Evals.	CPU Time	Language	CPU
(a)	160,000*	6–10 Hrs	C++	HP Apollo Series
(b)	∼100,000**	5 Hrs	C++	HP Apollo 715/75
(c)	150,000	*Unknown*	C++	*Unknown*
(d)	150,000–250,000†	5 Hrs	MatLab/C++	Intel PIV 2 GHz

Global Optimization of Planar Multilayered Dielectric Structures, Table 2
Comparison of optimum solutions found using ML [28] and GA [44]. The ML algorithm performed between approximately 150, 000 and 250, 000 function evaluations, depending on the number of layers, while the GA algorithm used between 150, 000 and 650, 000 function evaluations (specific number is unknown)

	Number of Layers	15	17	23	26	27	36
GA	Merit Function (%)	0.855	0.697	0.577	0.523	0.553	0.494
	Optical Thickness (μm)	20.34	27.04	40.17	50.99	44.98	71.15
ML	Merit Function (%)	0.675	0.638	0.556	0.531	0.535	0.507
	Optical Thickness (μm)	31.26	38.19	45.19	56.28	52.10	64.47

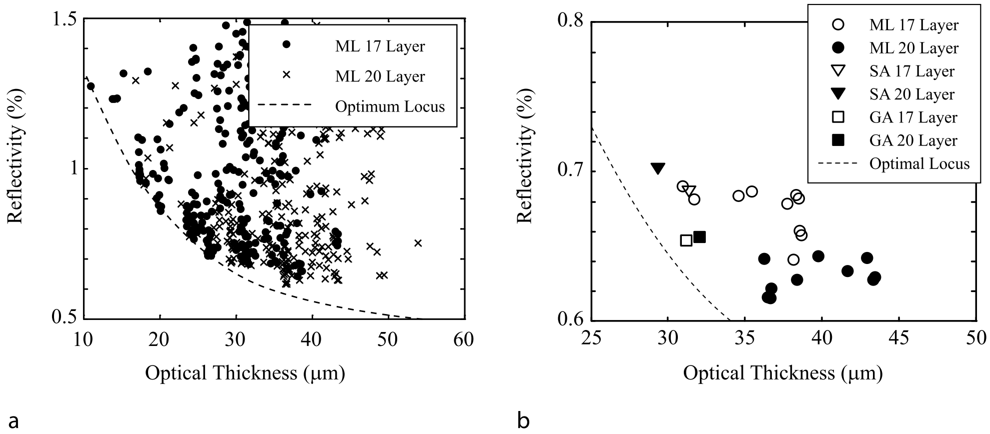
rithms and for assessing the performance of stochastic methods.

A Comparison of Methods for an Infrared Filter Design

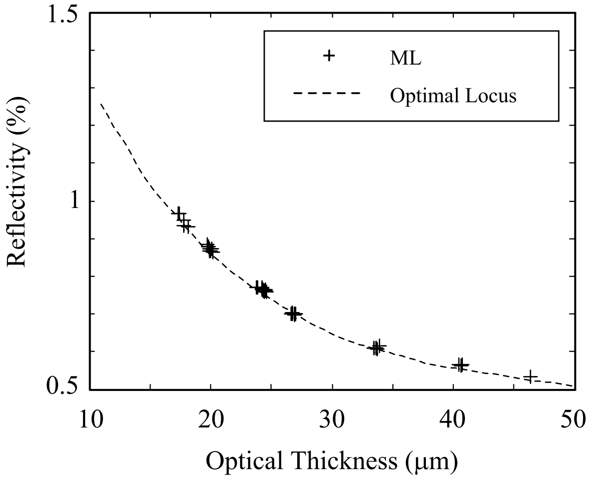
It is very difficult to compare the general performance of optimization approaches. In the following study, methods are compared through their performance in solving the infrared filter design problem that was described in the introduction to this Section. In each case, the same problem with precisely the same objective function is considered. In addition, past authors have terminated their solvers after a set number of iterations to allow a fair comparison with other methods. However, this can be a confusing measure of convergence as, in the case of GAs, a global optimum may not have been reached and in the case of SA, the cooling schedule may limit the effectiveness of the method. Consequently, the reader will note that there is no consensus over the global optimum between any of the optimization methods. Nevertheless, it is important to place each solver on an equal footing, and the number of function evaluations will be used as a measure of this. Table 1 shows information specific to each solver used in the study. It should be noted here, that only past studies of this problem using the methods discussed are compared in this study. Other studies that treat this problem can be found in [1,11,12,30,31] amongst others.

Yang and Kao [44] have provided an extensive study on this problem analyzing designs with varying layer number. The same approach was taken to generate data for the ML approach following the strategy in [28]. A direct comparison of optima found by GA [44] and ML methods is shown in Table 2 as a function of the number of design layers. Here, GA was allowed 150,000 function evaluations before stopping, whereas ML was allowed 5 hours, which, depending on the number layers, allows between 150,000 and 250,000 function evaluations. Both methods operate equally well, but, ML tends to locate slightly better solutions at the expense of optical thickness (this is equivalent the sum of the layer thicknesses multiplied by the respective refractive indices). This is to be expected due to the slightly larger number of function evalualtions allowed for structures with lower numbers of layers.

The trade-off between the optical thickness of a filter and its reflectivity has been examined by Dombrowolsky et al. [13]. Based on a quasi-deterministic quadratic approach [36] to the anti-reflection coating design, they



Global Optimization of Planar Multilayered Dielectric Structures, Figure 2
(a) Positions of all local solutions found using ML approach for 17 and 20 layers in the 5 hour calculation time. (b) Comparison of optimum solutions of GA [25], SA [6], ML [28] and optimum locus for this problem [13]. Note that for the GA and SA algorithms, the maximum optical thickness of the filter is 32 μm, whereas, the ML algorithm has no upper limit. The top 10 solutions for the ML approach are shown



Global Optimization of Planar Multilayered Dielectric Structures, Figure 3
New results of the ML approach generated by varying the number of layers from 15 to 30. Through post optimization analysis, the top 50 results nearest the optical locus [13] were identified

specify a locus merit function against optical thickness. This represents a theoretical limit on optimality for a given anti-reflection bandwidth. This can be tested in this instance: Fig. 2a shows the locus of solutions for merit function, $\mathcal{F}$, against optical thickness using the ML approach [28]. Here, dots represent the solutions for the 17 layer structure and crosses, solutions of the 20 layer structure for the ML approach. It is clear that all solutions appear on or above the optimal locus represented by the broken line. Note however, that the optimal locus does not guarantee that solutions should be found on or near it. Figure 2b shows these results alongside optical designs using GA [25] and SA [6]. Here, it is important to note that the GA and SA approaches limit the total optical thickness to 32 μm, whereas the ML algorithm is free to locate solutions over a larger range. Despite this, all methods appear comparable, with perhaps the GA appearing superior over SA. The effectiveness of the ML approach in identifying solutions near the optimal locus can actually be assessed after optimization.

An advantage of the ML approach is the ability to perform post optimization analysis on local minima making the optimization problem highly adaptive. This is appealing because supplementary design criteria can be taken into account without having to alter the objective function directly; the optimization is usually extremely sensitive to the form of the objective function. This can be quite effective since ML generates between 100 and 200 local solutions in the 5 hour calculation time, depending on the number of layers in a design. For example, further analysis of the local optima in the current example allows solutions near the optimal locus to be identified. New results were generated using the same approach as in [28] for designs ranging in layer number from 15–30. The top 50 solutions nearest the optimal locus were then filtered out from the complete set and are plotted in Fig. 3. Clearly, solutions very close to the optimal locus highlight the effectiveness of ML. However, the reduced number of function evaluations for structures with larger numbers of periods limits the effectiveness of the search. It is also interesting to note, that this analysis identifies gaps along the optimal locus. This suggests that, in some cases, extra layers are redundant when seeking to optimize both layer thickness and merit function.

Conclusions

The design of multi-layered dielectric optical structures can be formulated as a highly nonlinear optimization problem in which the thickness and refractive index of each layer is to be optimized, based on an appropriate objective function. This problem is known to have a large number of local minima, and several global optimization algorithm have been proposed to tackle it. These algorithms are mostly stochastic search algorithms (Simulated Annealing, Genetic Algorithms and Memetic Algorithms) or deterministic algorithms with a probabilistic guarantee of convergence (Multi-Level Algorithm). A deterministic approach with guaranteed global optimality has also been proposed based on an approximation of the design problem. The performance of several of these algorithms has been compared for a specific problem.

Future work on this design problem must continue to address the challenges posed by the large number of local optima which exist. The design formulation can also be extended to include the number of layers as one of the design variables. An early and encouraging effort in this direction is the needle optimization algorithm.

References

1. Aguilera JA, Aguilera J, Baumeister P, Bloom A, Coursen D, Dobrowolski JA, Goldstein FT, Gustafson DE, Kemp RA (1988) Antireflection coatings for germanium IR optics: a comparison of numerical design methods. Appl Opt 27:2832–2840
2. Bagnoud V, Salin F (1998) Global optimization of pulse compression in chirped pulse amplification. IEEE J Sel Top Quant Electron 4:445–448
3. Baumeister P (1958) Design of multilayer filters by successive approximations. J Opt Soc Am 48:955–958
4. Benisty H, Stanley R, Mayer M (1998) Method of source terms for dipole emission modification in modes of arbitrary planar structures. J Opt Soc Am A 15:1192–1201
5. Born M, Wolf E (1999) Principles of Optics, 6th edn. Cambridge University Press, Cambridge, UK
6. Boudet T, Chaton P, Herault L, Gonon G, Jouanet L, Keller P (1996) Thin-film designs by simulated annealing. Appl Opt 35:6219–6222
7. Bratley P, Fox BL, and Neiderreiter H (1992) Implementation and tests of low discrepancy sequences. ACM Trans Mode Comput Simul 2:195–213
8. Chang CP, Lee YH (1990) Optimization of a thin-film multilayer design by use of the generalized simulated-annealing method. Opt Lett 15:595–597
9. Cheng X, Wang Z, Zhang Z, Wang F, Chen L (2006) Design of X-ray super-mirrors using simulated annealing algorithm. Opt Commun 265:197–206
10. Crawford OH (1988) Radiation from dipoles embedded in layered system. J Chem Phys 89:6017–6027
11. Dobrowolski JA, Kemp RA (1990) Refinement of optical multilayer systems with different optimization procedures. Appl Opt 29:2876–2892
12. Dobrowolski JA, Kemp RA (1992) Interface design method for two-material optical multilayer coatings. Appl Opt 31:6747–6756
13. Dobrowolski JA, Tikhonaravov AV, Trubetskov MK, Sullivan BT, Verly PG (1996) Optimal single-band normal-incidence antireflection coatings. Appl Opt 35:644–658
14. Eisenhammer T, Lazarov M, Leutbecher M, Schiffel U, Sizmann R (1993) Optimization of interference filters with genetic algorithms applied to silver-based heat mirrors. Appl Opt 32:6310–6315
15. Graf T, Michette A, Powell A (2003) Design of multilayer mirrors for XUV applications using simulated annealing. J Phys IV France 104:243
16. Hagemana J, Wehrens R, van Sprang H, Buydens L (2003) Hybrid genetic algorithm-tabu search approach for optimizing multilayer optical coatings. Anal Chim Acta 490:211–222
17. Holland JH (1975) Adaptation in Natural and Artificial Systtems. University of Michigan Press, Ann Harbor
18. Holton JH (1960) On the efficiency of certain quasi-random sequences of points in evaluating multi-dimensional integrals. Numer Math 2:84–90
19. Hoorfar A, Zhu J, Nelatury S (2003) Electromagnetic optimization using a mixed-parameter self adaptve evolutionary algorithm. Microw Opt Technol Lett 39:267–271
20. Kan AHGR, Timmer GT (1987) Stochastic global optimization methods, part I, Clustering methods. Math Prog 39:27–56
21. Kan AHGR, Timmer GT (1987) Stochastic global optimization methods, part II, Multilevel methods. Math Prog 39:57–78

22. Kirkpatrick S, Gelatt CD, Vecchi MP (1983) Optimization by simulated annealing. Science 220:671–680
23. Kucherenko S, Sytsko Y (2005) Application of deterministic low-discrepancy sequences in global optimization. Comput Optim Appl 30:297–318
24. Liberti L, Kucherenko S (2005) Comparison of deterministic and stochastic approaches to global optimization. Int Trans Oper Res 12:263–281
25. Martin S, Rivory J, Schoenauer M (1995) Synthesis of optical multilayer systems using genetic algorithms. Appl Opt 34:2247–2254
26. McLeod HA (2001) Thin Film Optical Filters, 3rd edn. Adam Hilger L&D, Bristol
27. Niederreiter H (1987) Point sets and sequences with small discrepancy. Monatsh Math 104:273–337
28. Oulton RF, Adjiman CS (2006) Global optimization and modelling techniques for planar multilayered dielectric structures. Appl Optim 45:5910–5922
29. Peng K-O, de la Fonteijne M (1985) Derivatives of transmittance and reflectance for an absorbing multilayer stack. Appl Optim 24:501–503
30. Premoli A, Rastello ML (1992) Minimax refining of optical multilayer systems. Appl Opt 31:1597–1605
31. Rastello ML, Premoli A (1992) Continuation method for synthesizing antireflection coatings. Appl Opt 31:6741–6746
32. Schulz U, Schallenberg UB, Kaiser N (2002) Antireflection coating design for plastic optics. Appl Opt 41:3107–3110
33. Sobol IM (1967) On the distribution of points in a cube and the approximate evaluation of integrals. Comput Math Phys 7:86–112
34. Sullivan BT, Dobrowolski JA (1996) Implementation of a numerical needle method for thin-film design. Appl Opt 35:5484–5492
35. Tikhonov Jr AN, Tikhonravov AV, Trubetskov MK (1993) Second order optimization methods in the synthesis of multilayer coatings. J Comput Math Math Phys 33:1339–1352
36. Tikhonravov AV, Dobrowolski JA (1993) Quasi-optimal synthesis for antireflection coatings: a new method. Appl Opt 32:4265–4275
37. Tikhonravov AV, Trubetskov MK, DeBell GW (1996) Application of the needle optimization technique to the design of optical coatings. Appl Opt 35:5493–5508
38. Tikhonravov AV, Trubetskov MK, DeBell GW (2007) Optical coating design approaches based on the needle optimization technique. Appl Opt 46:704–710
39. Venkatarayalu NV, Ray T, Gan Y-B (2005) Multilayer dielectric filter design using a multi-objective evolutionary algorithm. IEEE Trans Antennas Propag 53:3625–3632
40. Verly PG (2001) Modified needle method with simultaneous thickness and refractive-index refinement for the synthesis of inhomogeneous and multilayer optical thin films. Appl Opt 40:5718–5725
41. Wild WJ, Buhay H (1986) Thin-film multilayer design optimization using a monte carlo approach. Opt Lett 11:745–747
42. Wu S-D, Gaylord TK, Maikisch JS, Glytsis EN (2006) Optimization of anisotropically etched silicon surface-relief gratings for substrate-mode optical interconnects. Appl Opt 45:15–21
43. Yakovelev V, Tempea G (2002) Optimization of chirped mirrors. Appl Opt 41:6514–6520
44. Yang J-M, Kao C-Y (1998) An Evolutionary Algorithm for Synthesizing Optical Thin-Film Dessigns. Lecture Notes in Computer Science. Springer, Heidelberg, p 1498
45. Yang J-M, Kao C-Y (2001) An evolutionary algorithm for the synthesis of multilayer coatings at oblique light incidence. IEEE J Lightwave Technol 19:559–570
46. Yang J-M, Horng J-T, Lin C-J, Kao C-Y (2001) Optical coating designs using the family competition evolutionary algorithm. Evol Comput 9:421–443
47. Zhou G, Chen Y, Wang Z, Song H (1999) Genetic local search algorithm for optimization design of diffractive optical elements. Appl Opt 38:4281–4290

Global Optimization in Protein Folding

DANIEL R. RIPOLL[1], ADAM LIWO[2],
HAROLD A. SCHERAGA[2]

[1] Computational Biology Service Unit,
Cornell University, Ithaca, USA

[2] Baker Laboratory of Chemistry,
Cornell University, Ithaca, USA

MSC2000: 60J15, 60J60, 60J70, 60K35, 65C05, 65C10, 65C20, 68U20, 70-08, 82B21, 82B31, 82B41, 82B80, 92C40, 92E10

Article Outline

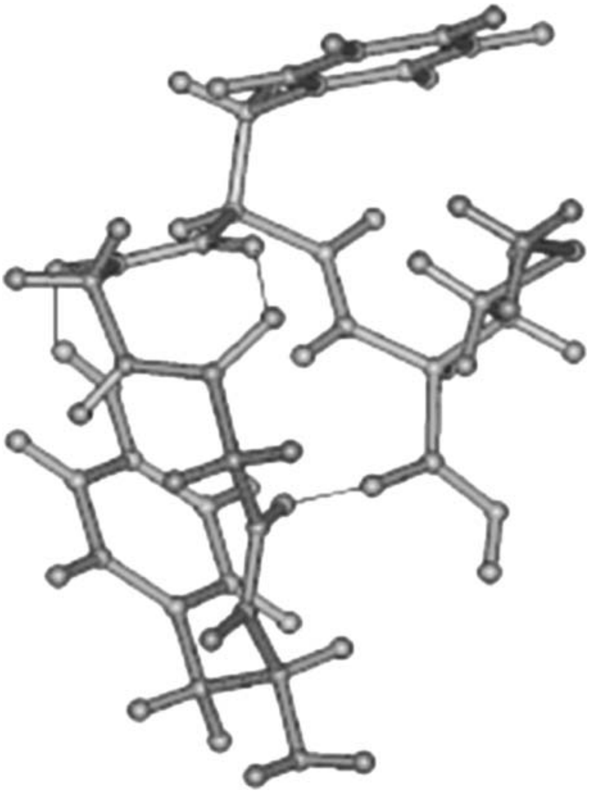
Keywords and Phrases

Global optimization; Protein folding; Multiple-minima problem; Markov process; Molecular dynamics

Introduction

The *multiple-minima problem*, i. e., the large number of minima associated with the potential functions used to represent the conformational energy of a polypeptide chain, is one of the greatest obstacles to overcome in order to compute the three-dimensional structure of a protein. Despite much effort and a large number of interesting ideas and approaches, progress toward the solution of this problem has been very slow. An exhaustive search of the conformational hypersurface of a large polypeptide is not computationally feasible even with today's supercomputers. Originally, the challenge was to locate the *global energy minimum* of small oligopeptides such as the pentapeptide Metenkephalin [1,17,20,26,45,46,48,55,57,58,59,72,73,79,91].

Since the global minimum of a potential function for a specific sequence is not known a priori, the only possibility of locating the global minimum of the potential energy is to carry out a large number of independent tests and determine if there is convergence to a unique conformation. This approach has been used in the test studies of Met-enkephalin in which hundreds of independent runs using different techniques have led to a *unique* lowest energy conformation, shown in Fig. 1, for the Empirical Conformational Energy Program for Peptides (ECEPP/2 [44,50,89], and ECEPP/3 [49]) potential energy functions. Similar results have been achieved for other test cases corresponding to larger sequences [1,23,39,56,62,77,78,80,81,94]. More recently, we have focused our efforts on the development of searching techniques that combine molecular dynamics with a coarse-grained representation of the protein structure. This approach to the protein folding problem is more rigorous since it accounts for entropic contributions and, on the other hand, is computationally more advantageous due to the simplified treatment of the complexities of the amino acid geometry. Our laboratory has made considerable progress in this area of research during the past few years, and we present a description of some of the successful methods that we have developed.



Global Optimization in Protein Folding, Figure 1
Lowest-energy conformation of Metenkephalin using the ECEPP/2 force field [27]

The Build-up Procedure

While systematic and exhaustive enumeration of all possible conformations is not practically feasible for polypeptides and proteins, attempts have been made to develop algorithms that lead to a truncated *systematic search* of the conformational space of these molecules. One of these methods, developed in our laboratory, the build-up procedure, [9,71,85,86,88,91] assumes that short-range interactions play a dominant role in determining the conformation of a polypeptide chain. Thus, the method starts by locating the low-energy conformations of small fragments of the chain by an exhaustive energy minimization procedure. Then, a selection of the minima is carried out, keeping those that lie within an appropriate chosen upper bound (the cutoff energy) of the lowest-energy fragment. Subsequently, the limited set of minima for one fragment is combined with the set of another fragment to form larger peptides which are also subjected to energy-minimization. This process is repeated until the whole chain is eventually built up from its constituent parts. At successive stages of the algorithm, more and more long-range interactions come into play.

Outline of the Procedure

1. The smallest fragment that the build-up procedure uses to construct a polypeptide conformation is the single amino acid. The ECEPP/2 minimum-energy conformations of terminally blocked single residues were reported by M. Vásquez, G. Némethy and H.A. Scheraga [90]. The conformations were ordered by increasing energy using a cutoff energy of 5 kcal/mol and were classified according to the code defined by S.S. Zimmerman, M.S. Pottle, G. Némethy and H.A. Scheraga [98]. The ECEPP/3 force field produces the same energy minima for all blocked amino acids with the exception of the proline and hydroxyproline residues.
2. All possible dipeptides for a given molecule are generated from single-residue data (for a peptide with *n* residues there are n−1 dipeptides). After energy-minimization, the dipeptides are sorted and are used to construct tripeptides.
3. Subsequent steps to form larger fragments of the polypeptide chain involve joining two fragments with one or more residues in common, e. g. after generating conformations for the tripeptides, these can be used to construct tetrapeptides from two tripeptides having two residues in common. This process is continued until the whole polypeptide chain is built.

Drawbacks of the Procedure

One of the major difficulties of the build-up procedure is that the number of conformations of fragments that must be energy-minimized and stored at each step increases exponentially. A partial solution, aside from using an energy cut-off, is to retain only those minima whose backbone conformations differ significantly: e. g. when several local minima have almost identical backbone but different side-chain conformations, only the lowest-energy minimum is kept while the degenerate ones are discarded. This approach drastically reduces the number of conformations to be stored at each stage of the procedure; however, it may lead to problems at later stages because the side-chain rotamers that are most favorable energetically in smaller fragments are not necessarily favored in the whole polypeptide chain. Another difficulty associated with the procedure is that atomic overlaps can occur when two fragments are joined in an arbitrary manner. These overlaps lead to conformations with extremely high energy for which minimization is usually not computationally feasible. A set of algorithms designed to surmount these problems was presented by K.D. Gibson and H.A. Scheraga [9].

Applications

The build-up procedure has been used extensively in a number of studies of different molecules, among them Metenkephalin [91], Gramicidin S [6,51], Melittin [71], bovine pancreatic trypsin inhibitor [92,93] and collagen [41,42,43]. The method appears to work well for small oligopeptides and fibrous proteins but, except in a few cases, its application to larger molecules becomes unmanageable for polypeptide chains containing 10 or more amino acid residues.

The Self Consistent Electrostatic Field Method

Among all the interactions that lead to protein folding, *electrostatic interactions* are the only ones characterized as long-range. Therefore, they undoubtedly must play an important role in folding. The dominant effects of electrostatic interactions in proteins are well recognized [60]. Among these effects, it is worth mentioning:

- The orientation of the CO and NH dipoles in α-helices are very favorable electrostatically [95] leading to a large dipole moment associated with this type of secondary structure.
- The electric field produced by an α-helix constitutes a very important stabilizing factor of the native conformations of proteins containing this type of secondary structure [11].
- The relative orientations of α-helices and β-sheets in proteins are favorable electrostatically [4,12,25].

Based on this evidence, L. Piela and H.A. Scheraga [62], hypothesized that that the native conformation of a protein arises when the electrostatic interactions are near optimal, or equivalently, that the native conformation must have approximately optimal orientations of its group dipoles in the electric field generated by the whole molecule and its surrounding solvent. Based on this assumption (which was later confirmed through rigorous calculations on an extensive set of proteins [82]), a conformational search method, named the Self-Consistent Electric Field (SCEF) method, was developed. The SCEF procedure was implemented as follows:

1. Given an arbitrary starting conformation of the molecule, minimize the total (e. g. ECEPP/3) conformational energy to reach the nearest local minimum.
2. For this minimized conformation, the *electric field* due to the whole molecule is calculated at each CO and NH group of the peptide units, and also in the middle of the C′-N peptide bond.
3. The direction of the electric field with respect to the CO and NH bond dipole moments provides information as to which peptide units are badly oriented. This electrostatic analysis of the alignment between the permanent dipoles and the electric field, is used to generate a *diagnostic rotation*. The diagnostic rotation is the variation that must be applied to a given torsional angle to obtain the best alignment of the worst oriented peptide-unit dipoles with respect to the electric field, e. g., if the electrostatic analysis indicates that the *dipole moment* of the peptide bond between residues *i* and *i+1* is the worst oriented, the diagnostic rotation will describe a change of the corresponding backbone dihedral angles ψ_i and ϕ_{i+1} required to align the dipole moment of the unit.
4. Carry out the diagnostic rotation.
5. Use the new conformation of the molecule as the starting point in step 1:
   - if a *new* local minimum is reached, then *repeat* the procedure from step 2 for the new local minimum;
   - if the *same* local minimum is found, then *step 3 must be repeated*, but using the diagnostic rotation for the next worst-oriented dipole.
6. Steps 1–5 are repeated until self-consistency is achieved, i. e., until further application of the procedure does not change the conformation of the molecule.

Computation of the Electric Field and Dipole Moments

If **r** represents the position vector assigned to the dipole moment *i* of a group of atoms, then the electric field, $\mathcal{E}(\boldsymbol{r})$, is computed as:

$$\mathcal{E}(\boldsymbol{r}) = (1/\epsilon)\sum_k{}' q_k(\boldsymbol{r}-\boldsymbol{r}_k)\big/\left|\boldsymbol{r}-\boldsymbol{r}_k\right|^3 \tag{1}$$

where ϵ is the dielectric constant, q_k indicates the charge on atom k with position vector $\boldsymbol{r}_k$ and the prime in the summation sign indicates that the atoms which contribute to the ith dipole moment as well as those other atoms covalently bonded to them should be excluded from the computation.

The electric field is computed at three points, $\boldsymbol{r}_{i,\mathrm{CO}}$, $\boldsymbol{r}_{i,\mathrm{NH}}$, and $\boldsymbol{r}_i$. These are reference points with respect to which the dipole moments of the CO bond, $\boldsymbol{\mu}_i^{\mathrm{CO}}$, the NH bond, $\boldsymbol{\mu}_i^{\mathrm{NH}}$, and the whole ith peptide unit, $\boldsymbol{\mu}_i$, respectively, are calculated. These dipole moments are computed according to the following relations:

$$\boldsymbol{\mu}_i^{\mathrm{CO}} = q_{\mathrm{C}}(\boldsymbol{r}_{i,\mathrm{C}} - \boldsymbol{r}_{i,\mathrm{CO}}) + q_{\mathrm{O}}(\boldsymbol{r}_{i,\mathrm{O}} - \boldsymbol{r}_{i,\mathrm{CO}}) \tag{2}$$

$$\boldsymbol{\mu}_i^{\mathrm{NH}} = q_{\mathrm{N}}(\boldsymbol{r}_{i,\mathrm{N}} - \boldsymbol{r}_{i,\mathrm{NH}}) + q_{\mathrm{H}}(\boldsymbol{r}_{i,\mathrm{H}} - \boldsymbol{r}_{i,\mathrm{NH}}) \tag{3}$$

$$\begin{aligned}\boldsymbol{\mu}_i = {} & q_{\mathrm{C}}(\boldsymbol{r}_{i,\mathrm{C}} - \boldsymbol{r}_i) + q_{\mathrm{O}}(\boldsymbol{r}_{i,\mathrm{O}} - \boldsymbol{r}_i) \\ & + q_{\mathrm{N}}(\boldsymbol{r}_{i,\mathrm{N}} - \boldsymbol{r}_i) + q_{\mathrm{H}}(\boldsymbol{r}_{i,\mathrm{H}} - \boldsymbol{r}_i)\end{aligned} \tag{4}$$

$\boldsymbol{r}_{i,\mathrm{CO}}$, $\boldsymbol{r}_{i,\mathrm{NH}}$, and $\boldsymbol{r}_i$ are chosen so that the bond quadrupole moments of the CO and NH bonds, $\boldsymbol{Q}_{\mathrm{CO}}$, $\boldsymbol{Q}_{\mathrm{NH}}$, respectively, vanish, i. e., the three points satisfy the following relations:

$$\boldsymbol{Q}_{\mathrm{CO}} = q_C\left|\boldsymbol{r}_{i,C} - \boldsymbol{r}_{i,\mathrm{CO}}\right|^2 + q_O\left|\boldsymbol{r}_{i,O} - \boldsymbol{r}_{i,\mathrm{CO}}\right|^2 = 0 \tag{5}$$

$$\mathbf{Q}_{\mathrm{NH}} = q_N|\boldsymbol{r}_{i,N} - \boldsymbol{r}_{i,\mathrm{NH}}|^2 + q_H|\boldsymbol{r}_{i,H} - \boldsymbol{r}_{i,\mathrm{NH}}|^2 = 0 \quad (6)$$

and,

$$\boldsymbol{r}_i = (\boldsymbol{r}_{i,C} - \boldsymbol{r}_{i,N})/2\,. \quad (7)$$

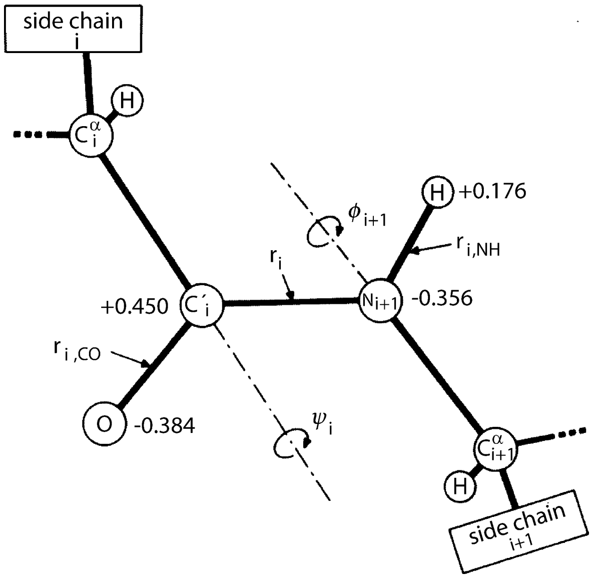
Degree of Alignment of a Dipole Moment with the Electric Field

The process of aligning a particular dipole moment, $\boldsymbol{\mu}_i^X$ (with X being CO or NH), with the electric field can be accomplished by rotations of the backbone dihedral angles ψ_i, and ϕ_{i+1} (see Fig. 2). When such a rotation is carried out, only the electric field components perpendicular to the rotation axis will change:

$$\mathcal{E}_{\perp k}(\boldsymbol{r}_{i,\mathrm{CO}}) = \mathcal{E}(\boldsymbol{r}_{i,\mathrm{CO}}) - [\mathcal{E}(\boldsymbol{r}_{i,\mathrm{CO}}) \cdot \boldsymbol{e}_{i,k}]\boldsymbol{e}_{i,k} \quad (8)$$

$$\mathcal{E}_{\perp k}(\boldsymbol{r}_{i,\mathrm{NH}}) = \mathcal{E}(\boldsymbol{r}_{i,\mathrm{NH}}) - [\mathcal{E}(\boldsymbol{r}_{i,\mathrm{NH}}) \cdot \boldsymbol{e}_{i,k}]\boldsymbol{e}_{i,k} \quad (9)$$

where $\boldsymbol{e}_{i,k}$ for $k = 1, 2$ denotes the unit vector along the axes of rotation, ψ_i, and ϕ_{i+1}, respectively. Furthermore, in writing these equations it was assumed that the points $\boldsymbol{r}_{i,\mathrm{CO}}$ and $\boldsymbol{r}_{i,\mathrm{NH}}$ are sufficiently close to the rotation axis.



Global Optimization in Protein Folding, Figure 2
SCEF peptide unit *i* with the atomic charges used in the ECEPP force field

The energy, E, of a dipole in an electric field is a given by:

$$E = -\boldsymbol{\mu} \cdot \mathcal{E}\,. \quad (10)$$

Assuming that the electric field in the neighborhood of the ith peptide group is relatively uniform, a lower bound for the energy gain due to a rotation is represented by:

$$\Delta E_i = \Delta E_i^{\mathrm{CO}} + \Delta E_i^{\mathrm{NH}} \quad (11)$$

where the individual energy gains, $\Delta E_i^{\mathrm{CO}} (< 0)$ and $\Delta E_i^{\mathrm{NH}} (< 0)$, to align the dipole and the field vectors are,

$$\Delta E_i^X = -|\boldsymbol{\mu}_{i,\perp k}^X|\,|\mathcal{E}_{\perp k}(\boldsymbol{r}_{i,X})| + \left[\boldsymbol{\mu}_{i,\perp k}^X \cdot \mathcal{E}_{\perp k}(\boldsymbol{r}_{i,X})\right] \quad (12)$$

with

$$\boldsymbol{\mu}_{i,\perp k}^X = \boldsymbol{\mu}_i^X - (\boldsymbol{\mu}_i^X \cdot \boldsymbol{e}_{i,k})\boldsymbol{e}_{i,k}\,. \quad (13)$$

The value of ΔE_i given by Eq. (11) is used as a measure of the deviation from perfect alignment in the electric field of the ith peptide unit.

Best-possible Alignment of a Dipole Moment with the Electric Field

From an analysis of the ΔE_is, it is possible to detect which peptide unit is the most unfavorably oriented in the electric field. The SCEF method provides a mechanism to compute the rotation that should lead to an improved orientation of this peptide unit with respect to the electric field. To accomplish this, the electric field $\mathcal{E}(\boldsymbol{r}_i)$ at the ith peptide unit can be viewed as the sum of two contributions generated by the portions of the polypeptide chain on both sides of the ith unit: (a) $\mathcal{E}_N(\boldsymbol{r}_i)$ generated by the part of the molecule containing the N-terminus; and (b) $\mathcal{E}_C(\boldsymbol{r}_i)$ generated by the part of the molecule containing the C-terminus,

$$\mathcal{E}(\boldsymbol{r}_i) = \mathcal{E}_N(\boldsymbol{r}_i) + \mathcal{E}_C(\boldsymbol{r}_i)\,. \quad (14)$$

The components of $\boldsymbol{\mu}_i$, $\mathcal{E}_N(\boldsymbol{r}_i)$ and $\mathcal{E}_C(\boldsymbol{r}_i)$ parallel to an axis of rotation do not change with rotations about this axis. On the other hand, the perpendicular components of these vectors with respect to a given axis, say $\boldsymbol{e}_{i,k}$, do change with rotations about the axis and they are given by:

$$\boldsymbol{\mu}_{i,\perp k} = \boldsymbol{\mu}_i - \boldsymbol{e}_{i,k}(\boldsymbol{\mu}_i \cdot \boldsymbol{e}_{i,k}) \tag{15}$$

$$\mathcal{E}_{N,\perp k}(\boldsymbol{r}_i) = \mathcal{E}_N(\boldsymbol{r}_i) - \boldsymbol{e}_{i,k}\left[\mathcal{E}_N(\boldsymbol{r}_i) \cdot \boldsymbol{e}_{i,k}\right] \tag{16}$$

$$\mathcal{E}_{C,\perp k}(\boldsymbol{r}_i) = \mathcal{E}_C(\boldsymbol{r}_i) - \boldsymbol{e}_{i,k}\left[\mathcal{E}_C(\boldsymbol{r}_i) \cdot \boldsymbol{e}_{i,k}\right] . \tag{17}$$

If $\boldsymbol{\mu}_{i,\perp k}$ does not lie along $\mathcal{E}_{\perp k} = \mathcal{E}_{N,\perp k} + \mathcal{E}_{C,\perp k}$, perfect alignment between the vectors can be obtained by a *single* rotation about $\boldsymbol{e}_{i,k}$. For $k = 1$, a rotation about the ψ_i axis produces a change of $\mathcal{E}_{N,\perp 1}$ to $\mathcal{E}'_{N,\perp 1}$. Similarly for $k = 2$, a rotation about the ϕ_{i+1} axis leads to a change of $\mathcal{E}_{C,\perp 2}$ to $\mathcal{E}'_{C,\perp 2}$. Therefore, alignment is achieved if either *one* of the following equations is satisfied: for $k = 1$ (ψ_i axis),

$$\boldsymbol{\mu}_{i,\perp 1} \cdot (\mathcal{E}'_{N,\perp 1} + \mathcal{E}_{C,\perp 1}) = |\boldsymbol{\mu}_{i,\perp 1}|\,|\mathcal{E}'_{N,\perp 1} + \mathcal{E}_{C,\perp 1}| \tag{18}$$

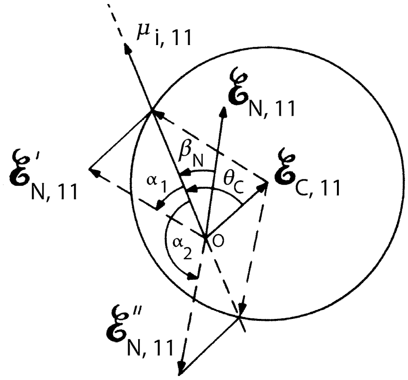
for $k = 2$ (ϕ_{i+1} axis),

$$\boldsymbol{\mu}_{i,\perp 2} \cdot (\mathcal{E}_{N,\perp 2} + \mathcal{E}'_{C,\perp 2}) = |\boldsymbol{\mu}_{i,\perp 2}|\,|\mathcal{E}_{N,\perp 2} + \mathcal{E}'_{C,\perp 2}| . \tag{19}$$

From geometrical considerations (see Fig. 3), the solution of Eq. (18) (similarly for Eq. (19)) is found to satisfy the relation:

$$|\alpha| = \arccos(c/b) \tag{20}$$

where $b = |\mathcal{E}_{N,\perp 1}|$, $c = d^{1/2}$ with $d = b^2 - a^2 \sin^2 \theta_C$, $a = |\mathcal{E}_{C,\perp 1}|$, and θ_C is the angle between $\mathcal{E}_{C,\perp 1}$ and $\boldsymbol{\mu}_{i,\perp 1}$. Equation (20) has various numbers of solutions. If they exist, these solutions correspond to rotations of the dipole moment $\boldsymbol{\mu}_{i,\perp 1}$ with different energies. The value leading to the lowest energy represents the solution to the alignment Eq. (18). This rotation of ψ_i leads to an energy gain given by,



Global Optimization in Protein Folding, Figure 3
SCEF: solution of alignment Eq. (18)

$$\Delta E_{i,N} = -\boldsymbol{\mu}_{i,\perp 1} \cdot (\mathcal{E}'_{N,\perp 1} - \mathcal{E}_{N,\perp 1}) . \tag{21}$$

Expressions similar to Eq. (20) and (21) have been derived for the rotation around the ϕ_{i+1} axis ($k = 2$) and for the corresponding energy gain, $\Delta E_{i,C}$.

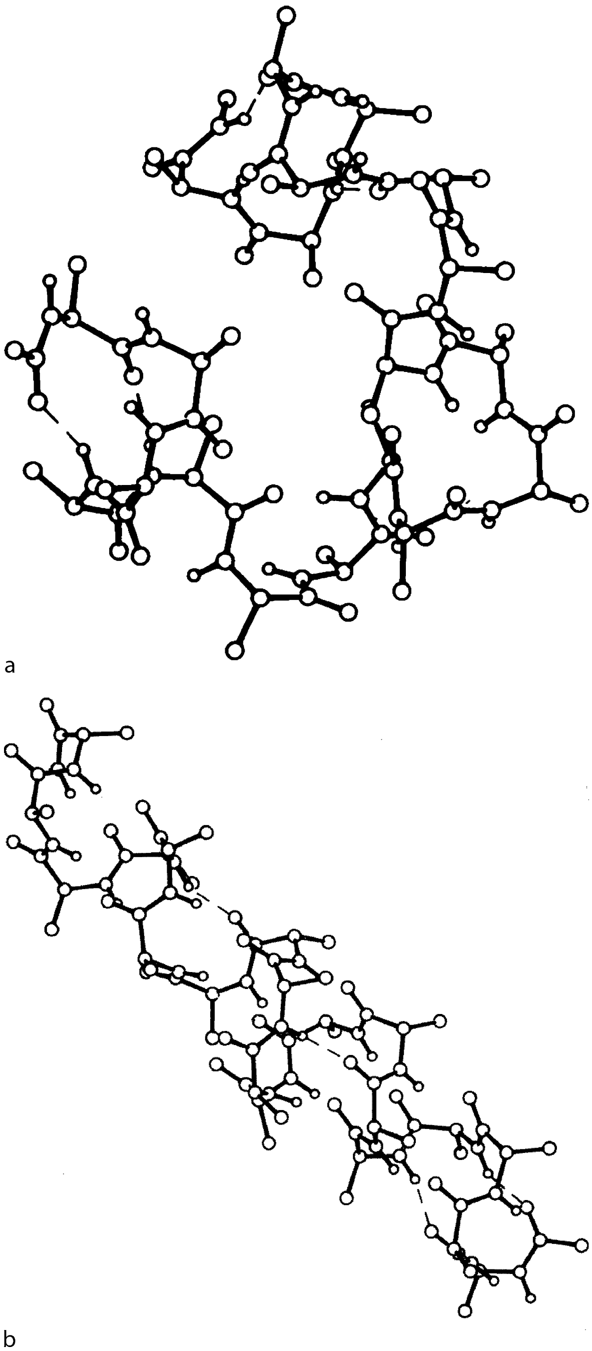
It should be mentioned that, in reality, the solution given by Eq. (20) produces an approximate alignment of $\boldsymbol{\mu}_{i,\perp 1}$ with the corresponding electric field component. The reason is that the derivation of these equations was based on the assumptions that (a) the center of the peptide unit is on the ψ_i axis of rotation, and (b) the electric field is homogeneous. While, in reality, these conditions are not satisfied, the results obtained from these expressions are reasonably accurate [62].

Finally, after both rotations about the ψ_i and ϕ_{i+1} axis have been computed, the SCEF method has to decide which rotation should be implemented. The method selects the rotation associated with the more negative energy gain ($\Delta E_{i,N}$ or $\Delta E_{i,C}$). In those cases where no solution is found for ψ_i and ϕ_{i+1}, another unfavorable peptide unit is chosen.

Applications

The procedure was tested on a 19-residue poly(L-alanine) chain [62] with acetyl-and N-methyl amide terminal blocking groups. The starting conformations were a series of partially α-helical conformations representing different degrees of distortion from the canonical right-handed *α-helix*. The right-handed α-helical conformation corresponds to the global energy minimum of the ECEPP/2 (and ECEPP/3) potential function. In the four cases reported, the procedure was able to achieve the conformation corresponding to the global energy minimum in a very short computation time.

Figure 4a shows the starting conformation of one of the tests. The conformation contains only 1.5 α-helical turns at each terminus and 70.6% of the native hydrogen bonds are broken. In subsequent iterations of the SCEF procedure, the right-handed α-helix shown at the bottom of Fig. 4b, was completely recovered.



Global Optimization in Protein Folding, Figure 4
SCEF method: application to poly(L-alanine)

The SCEF procedure was also used [76] in a restrictive search of the conformational space of the 58-residue protein bovine pancreatic trypsin inhibitor (BPTI). In this application, the algorithm led to a series of conformations with up to 50 kcal/mol lower than the starting conformation.

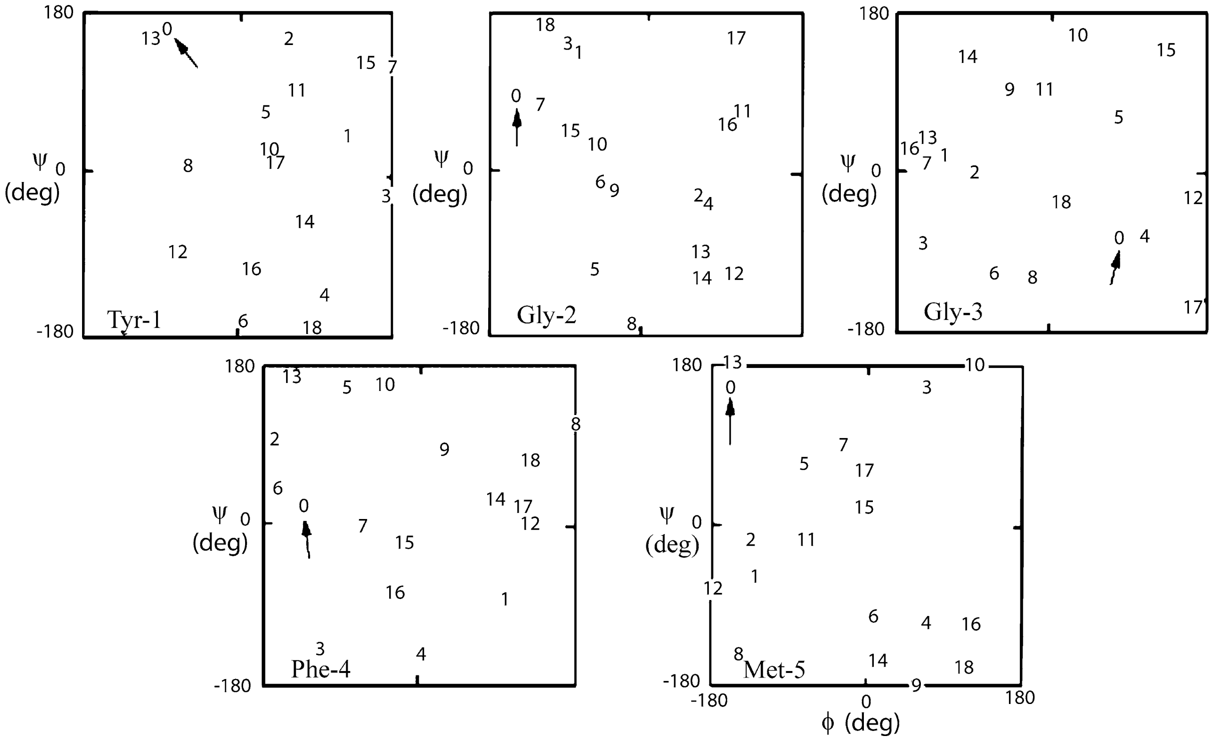
The Monte Carlo-Minimization Method

The Monte Carlo-Minimization (MCM) [26], [27] method developed by Z. Li and H.A. Scheraga was motivated by experimental studies indicating that proteins are not static structures but instead undergo fluctuations. For a protein to be stable, its *native conformation* must be stable not only to small perturbations but also against larger-scale *thermal fluctuations*. Based on these considerations, Li and Scheraga developed a *stochastic approach* for *global optimization* of polypeptides and proteins that combines the power of the Metropolis Monte Carlo method [40] in global *combinatorial optimization* and that of conventional *energy minimization* to find local minima. The underlying working hypothesis of the method is that protein folding can be considered as a *Markov process*, with (a) Boltzmann transition probabilities, and (b) this Markov process should lead to a unique absorbing state [3] that corresponds to the native state for a natural biologically active protein. For this absorbing state, equilibrium is reached after a sufficiently long time and the stationary probability of occurrence approaches unity.

The *Metropolis Monte Carlo method* can simulate the thermal processes, by taking into account both random fluctuations and energetic considerations. However, straightforward applications of the Metropolis Monte Carlo method to polypeptides has proven to be quite inefficient [10,57,74] mainly because (a) a high-dimensional conformational space has to be sampled by making small increments of the variables in each step, and (b) The large energy barriers in the conformational space tend to confine the sampling to a very restrictive region of the space. To overcome these difficulties, the MCM method includes conventional energy minimization as a second important feature. Thus, the MCM method generates a Markov walk on the hyperlattice of all discrete energy minima, with Boltzmann transition probabilities.

The procedure implemented in the MCM algorithm is as follows:

- Given an energy-minimized conformation, $C^{\min}_{\text{curr}}$, with total energy $E^{\min}_{\text{curr}}$, a Monte Carlo sampling strategy is used to generate a perturbed conforma-



Global Optimization in Protein Folding, Figure 5
$\phi - \psi$ maps for the five residues of Metenkephalin showing the backbone dihedral angles of 18 random starting conformations (indicated by the numbers 1 to 18) for the MCM method. The backbone dihedral angles of the global minimum achieved by the MCM method in all the runs (see Fig. 1) are indicated by 0

tion C_{pert}. The sampling strategy consists of random changes, involving k dihedral angles of the total number N_{dieh} used to described the molecule. The number of changes are generated with probabilities 2^{-k} ($k = 1, 2, \ldots, N_{\text{dieh}}$). This probability selection implies that fluctuations involving more degrees of freedom are sampled with successively lower probabilities. This sampling strategy satisfies the ergodicity requirements, i. e., any local minimum is accessible from any other one after a finite number of random sampling steps. Furthermore, in order to improve the average acceptance ratio, random changes involving backbone dihedral angles are sampled more frequently than those of side chains. This type of sampling strategy led to an average acceptance ratio of approximately 20% at 0°C for Metenkephalin.

- The randomly generated conformation, C_{pert}, is then subjected to conventional energy minimization until it reaches the nearest local minimum of the *potential energy function* (ECEPP/2 or ECEPP/3). Minimization of the energy is carried out with the Secant Unconstrained Minimization Solver (SUMSL) algorithm [8]. The resulting conformation, $C^{\text{min}}_{\text{pert}}$, has a total energy $E^{\text{min}}_{\text{pert}}$ and is usually free of atomic overlaps.
- The energies of the conformations $C^{\text{min}}_{\text{pert}}$ and $C^{\text{min}}_{\text{curr}}$ are compared, and the Metropolis criterion is used to decide which conformation is to be kept, i. e., if the energy difference $\Delta E = E^{\text{min}}_{\text{pert}} - E^{\text{min}}_{\text{curr}} < 0$, or (when $\Delta E > 0$) if $e^{-\Delta E/RT}$ is greater than a randomly generated number between 0 and 1, the new conformation, $C^{\text{min}}_{\text{pert}}$ replaces the current $C^{\text{min}}_{\text{curr}}$; otherwise, $C^{\text{min}}_{\text{pert}}$ is discarded.

Applications

The MCM procedure was successfully applied to study the conformational preference of the pentapeptide Metenkephalin [26,27]. In its initial application [26], 13 of 18 random starting conformations of this oligopeptide converged to the global minimum, shown in Fig. 1, within the time of the simulations. Using a different sampling strategy [27], the 5 remaining runs also converged to the same lowest energy structure. Figure 5 shows the values of the backbone dihedral angles, ϕ and ψ for the 18 starting conformations.

As a further development, we extended the concept of MCM to include biasing the perturbations to electrostatic interaction, giving the Electrostatically Driven Monte Carlo (EDMC) method, which is described in the next section. More recently, we took advantage of grouping the conformations obtained in the search into families which are updated on the fly and, using the properties of the families in the subsequent steps of the search, this resulted in the conformation-family Monte Carlo (CFMC) method [65]. The CFMC method was used to search the conformational space of the B-domain of staphylococcal protein A in the united-residue representation [65] and for crystal structure prediction of small molecules [63].

The Electrostatically Driven Monte Carlo Method

The Electrostatically Driven Monte Carlo (EDMC) method, introduced by D.R. Ripoll and H.A. Scheraga, is a procedure for iteratively searching the conformational hypersurface of relatively small polypeptide molecules. The EDMC method incorporates the best features of the SCEF and MCM methods and combines them with a set of new techniques to produce a more efficient search of the conformational space.

The search for the the global energy minimum of a molecule proceeds as a "quasi-random walk" along a conformational pathway. As with the MCM method, this pathway is defined, in principle, by an infinite sequence of energy-minimized conformations encountered over an unbounded number of iterative steps of the algorithm. In practice, however, a finite number of iterations is specified for a given run. The underlying assumption behind the EDMC method is that (a) the electrostatic interactions should lead to conformations representing an improvement of the charge distribution, i. e. the new conformations are expected to have lower electrostatic and total energies; and (b) thermal fluctuations, on the other hand, are expected to introduce disorder within the molecule. These thermal effects could force the molecule to adopt conformations that are higher in energy, but may allow it to escape from stable local minima of relatively high energy.

The implementation of these ideas is accomplished as follows: Thermal effects are associated with random changes in the molecular conformation, i. e. a small set of randomly-chosen variables was altered randomly. On the other hand, the reordering effect of the electrostatic interactions was viewed as a tendency of all permanent dipole moments associated with the peptide units of the polypeptide, to attain their best possible alignment in the local electric field produced by the rest of the molecule. Additionally, a series of new features [77], included in the latest implementation of the EDMC method, has helped to accelerate the search and to optimize the process of generation of new conformations.

The Procedure

The first accepted conformation on the conformational pathway followed by the EDMC method is usually an unfolded state of the polypeptide chain (i. e. the initial values of the variables describing the molecular conformation are assigned randomly); its energy is minimized to relieve possible atomic overlaps. The subsequent accepted conformations are obtained by a variety of techniques described below. An *iteration* of the procedure is defined as a set of manipulations of the currently accepted conformation that leads to its *replacement* by a newly generated conformation.

The strategy used to produce new conformations within an iteration of the method is based upon a combination of movements associated with the electrostatic interactions and thermal motion.

(a) An important technique that the EDMC method uses to generate new conformations is based on an electrostatic analysis similar to that produced by the SCEF method [62], but extended to consider the permanent dipole moments of polar sidechains. As a first step of an iteration, this electrostatic analysis of the currently accepted conformation (the initial energy–minimized conformation or the accepted conformation from the previous iteration) is carried out to determine the alignment of the permanent dipoles with the local electric field produced by the whole molecule. As a result, *diagnostic rotations* that could improve the local dipole alignments with the electric field are produced for all permanent dipole moments. These diagnostic rotations are incorporated into a *prediction list* of possible conformational changes. The information contained in this list is used to generate new conformations in a subsequent search for states of lower energy.

(b) Since it may happen that none of these predictions leads to an acceptable conformation, a random and/or biased sampling technique is also used to generate additional conformations. The following procedure is followed:
   1. Specification of the mode in which the variable dihedral angles of the *selected residues* are to be altered:
      i) Select all variables at random;
      ii) Select the backbone variables randomly within specific regions of the $\phi - \psi$ map;
      iii) Select all variables from pre-computed low-energy conformations of the tri-peptides included in the sequence;
      iv) Select backbone variables compatible with regular structures β-sheets or α-helices).
   2. Random selection of i) the number of residues to be affected by the changes, and ii) their positions in the sequence.

The latest implementation of the algorithm [77] includes a technique to produce a *cluster analysis* of the accepted minima. The conformations are grouped into clusters using rms distance criteria and ranked on the basis of their total energies. Furthermore, every generated conformation, even if rejected, is associated with an existing cluster or family, but added to it only if its energy is lower than the one corresponding to the best member of that family.During an iteration, randomly generated conformations can also be produced by perturbing low-energy conformations included in any of the clusters (except the one containing the current accepted minima) using the protocol described in item (b) above.

A conformation generated by any of these two procedures (a or b) is subjected to minimization of the total energy where the backbone and side-chain dihedral angles of the molecule are considered as variables. The energy-minimization procedure is carried out with the SUMSL algorithm [8]. The value of the potential energy constitutes the basis for either the acceptance or rejection of the new minimum-energy conformation. A newly generated conformation must fulfill two criteria to be accepted:

1. If a generated conformation is found to correspond to an accepted minimum that has already been encountered more than a pre-defined number of times (usually 5–10), then it is automatically excluded from further consideration. This analysis of the long–term behavior of the search provides one of the criteria to ensure that the search does not become trapped in a set of local minima of the conformational space.
2. If a conformation satisfies the previous condition, its energy E_{new} is compared with the energy, E_{curr}, of the current accepted conformation, and the Metropolis criterion [40], as described for the MCM method, is applied.

When the energy of the new conformation passes both tests successfully, the conformation is accepted, replacing the current one, and a new iteration begins.

Backtrack

The number of conformations generated within a given iteration is limited (usually 100 to 200 conformations). It may happen that neither the set of electrostatic predictions, nor the set of randomly generated conformations produces an acceptable conformation. Under these circumstances, the algorithm then assumes that the current local minimum is quite stable and a new procedure named *backtrack* is triggered. The backtrack procedure attempts to displace the search to a different region of the conformational hypersurface by substantially altering the processes of generation and acceptance of conformations.

The backtrack procedure involves the following:

a) A new set of conformations is generated by changing a large number of variables simultaneously. In particular, the procedure tends to select the variables associated mainly with the backbone of the polypeptide chain; and,
b) the temperature parameter, T, used in the Metropolis acceptance criterion is *(i)* raised abruptly to a very high value, or *(ii)* steadily increased by means of a pre-defined heating scheme.

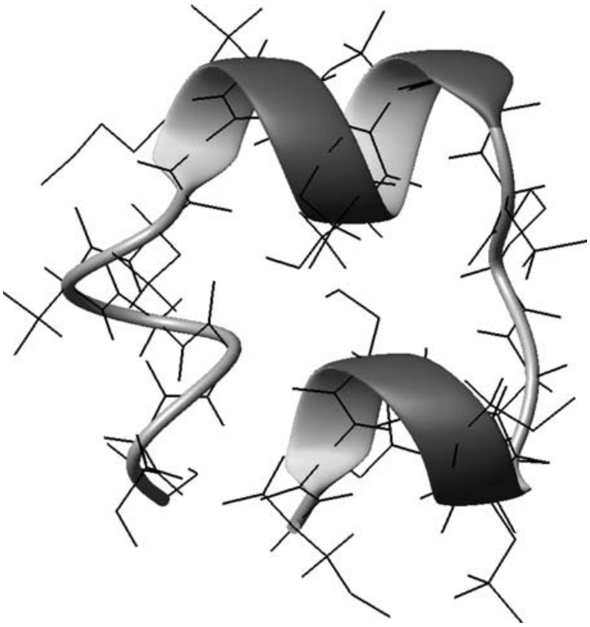
The backtrack procedure is applied until the acceptance test is satisfied, or until the number of generated conformations reaches a predetermined maximum value. In the rare event that the latter situation occurs, the run is terminated since it is assumed that it is practically impossible to escape from the current region of the conformational space. On the other hand, when a conformation from the backtrack procedure is accepted, the temperature parameter is reset to its original user-specified value, and the generation mechanism is switched back to the standard protocol described above.

The objective of the modified generation procedure during backtrack is to produce conformations substantially different from the current minima, while raising the temperature has the effect of increasing the probability of acceptance of conformations with energies much higher than the current local minimum. The backtrack mechanism has been shown to be an effective technique to help the search avoid being trapped in stable, high-energy regions of the conformational space.

The EDMC method has some similarities with *simulating annealing*, proposed by S. Kirkpatrick, C.D. Gelatt and M.P. Vecchi [15], since both make use of high temperatures to surmount large energy barriers. The difference is that the EDMC procedure concentrates the search in the low-energy regions of the conformational space using energy minimization and a low temperature value. High temperatures are used rarely during backtrack to escape from stable or already visited regions. Once this is accomplished, the temperature parameter is reset to its initial (low) value. A search using simulated annealing, on the other hand, starts with a high temperature value and this parameter is gradually reduced during the simulation. The expectation is that, given a *sufficiently high* initial temperature and a *good* annealing schedule, the search will overcome large energy barriers and will become localized in the low-energy region containing the global minimum.

Applications

The multiple-minima problem has been found to be computationally tractable by the EDMC method on existing computers for polypeptides sequences consisting of up to 20 amino acid residues.



Global Optimization in Protein Folding, Figure 6
Lowest-energy conformation of the membrane-bound portion of melittin for the ECEPP/3 force field determined by the Conformational Space Annealing [23] and the EDMC [77] methods

In applications to Metenkephalin [79], oxytocin [39], arginine-vasopressin [39], decaglycine [80], a 19-residue chain of poly(L-alanine) [78], and the 20-residue membrane-bound portion of melittin [77] (see Fig. 6), the EDMC algorithm converged to unique conformations presumed to be the global energy minima for those particular sequences.

In other applications, to a seven-residue peptide epitope [75], and a twelve-residue analogue of mastoparan and mastoparan X [7], the method identified very low-energy conformations, but it is not certain that the global energy minima were attained in these cases.

Lately, the EDMC method has been applied to the 36-residue villin headpiece subdomain [81], and the 45-residue fragment B-domain of staphylococcal protein A [94]. In both applications, unrestricted global searches that started from randomly generated conformations encountered in their paths low-energy basins that included native-like conformations. To our knowledge, the application to the B-domain of staphylococcal protein A was, at the time, the first *all-atom* simulation in which such a large protein was ever folded from random initial conformations without resort to knowledge-based information.

The EDMC method has also been used in restrictive searches of the conformational space of larger molecules. In an application to the 58-residue protein BPTI [76], the algorithm produced the lowest energy conformation known for BPTI using the ECEPP/2 or ECEPP/3 potential. In addition, the EDMC method has also been used to search the conformational properties of a non-oncogenic p21 protein [30] and a molecular switch designed as a biological logic gate [2].

The Diffusion Equation Method and Other Methods Based on the Deformation of the Potential-Energy Surface

The *diffusion equation method* (DEM) is a deterministic approach that attempts to solve the multiple-minima problem by deforming the potential energy hypersurface. The basic idea of the method, introduced by Piela et al.[61], is to deform the multivariable function that represents the potential energy in such a manner as to make the shallow wells disappear gradually, while other potential wells grow at their expense. Under the assumption that the shallower wells will disappear more easily than the deep wells, it is possible to envision an iterative procedure that, applied to the potential function, will change its shape, making most of the minima become shallower until they disappear, while leaving a single absorbing minimum related to the lowest minimum of the original function. At this point of the *deformation process*, a simple local minimization algorithm should be able to retrieve the position of the unique minimum from any starting point. However, since the deformation of the potential should likely have altered the location of all minima, the global minimum of the original function is not the same as the minimum of the deformed surface. Its location can, in principle, be attained by slowly reversing the deformation and using standard local minimization procedures. Piela et al. showed that the deformation of the hypersurface can be carried out with the aid of the diffusion equation. In this context, the original shape of the potential function has the meaning of an initial concentration (or temperature) distribution.

The diffusion equation method which must be solved to obtain a deformed potential-energy surface is given by Eq. (22).

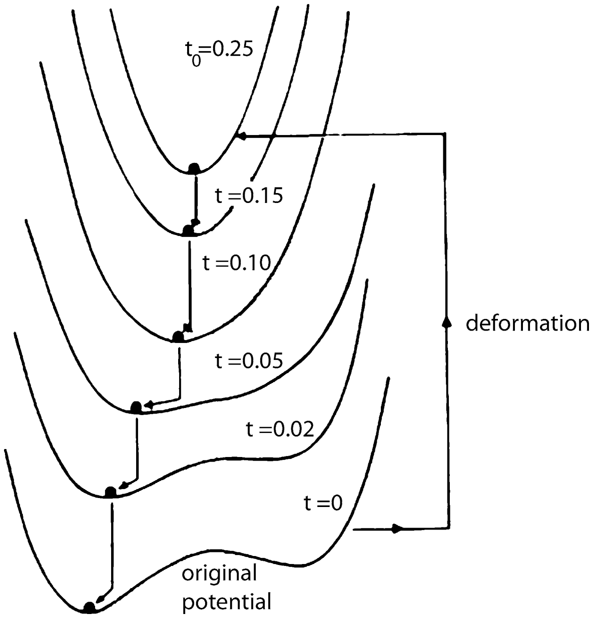
$$\nabla^2 F(x_1, x_2, \ldots, x_n; t) = \frac{\partial F(x_1, x_2, \ldots, x_n; t)}{\partial t} \tag{22}$$

where $x_1, x_2, \ldots, x_n$ are variables describing the conformation of a molecule, $\nabla^2 = \left(\partial^2/\partial x_1^2, \partial^2/\partial x_2^2, \ldots, \partial^2/\partial x_n^2\right)$ is the Laplacian operator, the variable t represents time and can be identified with the extent of deformation, and F is the deformed potential-energy function. Additionally, Eq. (22) is solved with the initial condition $F(x_1, x_2, \ldots, x_n; 0) = f(x_1, x_2, \ldots, x_n)$, where $f(x_1, x_2, \ldots, x_n)$ is the original (undeformed) potential-energy function. The function F usually represents a concentration or a temperature distribution. If the function $f(x_1, x_2, \ldots, x_n)$ is bounded, a solution of Eq. (22) exists for any positive value of t.

The procedure described above represents a spontaneous mass transport (or flow of heat) in a medium for an initial distribution of concentration (or temperature) given by the function $f(x_1, x_2, \ldots, x_n)$ (which in our case represents the conformational energy). Governed by the diffusion equation and independent of the initial conditions, the concentration (or temperature), will evolve with time in such a manner that it will become constant for $t = \infty$. However, it is expected that the concentration (or temperature) will exhibit a single minimum for certain (very large) values of t. This single minimum should represent the last trace of the potential well corresponding to the global minimum of the original hypersurface $f(x_1, x_2, \ldots, x_n)$. The deformation and its subsequent reversal to retrieve the position of the original minimum is illustrated in Fig. 7.

Application of the DEM consists of the following steps:

- Solve Eq. (22) using $F(x, 0) = f(x)$ as the initial condition or apply the operator $T(t)$ for a sufficiently large value of t (t_0); then, use a local minimization to locate the position $x^*_{t_0}$ of the unique minimum on the deformed surface. This is the starting point to be used in the reversing procedure.
- Apply the reversing procedure described above.
- For a reversing procedure involving m steps, the position x^*_0 obtained by minimizing $F(x^*_{t_0-(m-1)\Delta t}, t_0 = 0)$ should correspond, hopefully, to the position of the global minimum of the function f.



Global Optimization in Protein Folding, Figure 7
The DEM method: Illustration of the deformation of the original potential $f(x) = x^4 + 2x^3 + 0.9x^2$ by the operator $T(t) = exp(td^2/dx^2)$, and of the reversing procedure. The deformation applied by the operator $T(t_0 = 0.25)$ leads to a curve with a unique minimum that is achievable from any point of the space with a simple minimization. The reversing procedure is shown by the arrows directed downward. Each step of the reversing procedure is followed by minimization symbolized in the figure by a ball moving down hill from the minimum position of the upper curve and always reaching the position of the minimum in the lower curve. In the final step, the global minimum of the original function is found

Among other applications, the DEM has been applied to:

- A cluster of 55 *Lennard-Jones* atoms for which the global minimum was found [16].
- A single terminally blocked alanine [17].
- The pentapeptide Met-enkephalin [17] for which the method led to practically the same global-minimum backbone structure obtained by other methods. The test, however, was carried out under more restrictive conditions since only the backbone dihedral angles ϕ and ψ were considered as variables.
- Prediction of the crystal structures of hexasulfur and benzene molecules [96,97].

Although the DEM method is, in theory, a deterministic approach, we found [96,97] that it must be combined with a Monte Carlo search to work for more complex systems. When the potential-energy surface is deformed to contain just a single minimum, it is so flat that, to the numerical accuracy, it is effectively constant. Thus, deformation cannot be carried out to leave only one minimum. Moreover, the position of a minimum on a highly deformed surface is too far from that on the original energy surface. During the process of reversal, the single minimum splits into multiple minima and it is not clear which one of those should be chosen to continue the reversal. In our successful application to crystal-structure prediction [96,97] we, therefore, introduced the MCM search both on the deformed potential-energy surface and during reversal.

Taking advantage of the concept of the deformation of potential-energy surfaces, we developed several other methods for the search of the global minimum of the energy of polypeptide and proteins. The *distance scaling method* (DSM) [70] developed by J. Pillardy and L. Piela, (as well as its predecessor, the shift method (SM) [68]) attempts to solve the multiple-minima problem using transformations of the atom–atom distances that lead to *smoothing of the potential energy* hypersurface. These methods have subsequently evolved into the Self-Consistent Basin-to-Deformed-Basin Mapping (SCBDBM) method, in which the coupling between the basin containing the global energy minimum to the corresponding basin in the deformed potential-energy surface is established. The SCBDBM involves some Monte Carlo search on the deformed potential-energy surface and during the process of reversal. All three methods have been applied successfully to clusters of *argon atoms* and water molecules [67,68,69,70] and to the *prediction of crystal structures* [97]. The SCBDBM method was also applied [66] in searches for low-energy minima of poly-L-ananine chains of up to 100 amino-acid residues in length and the 10–55 fragment of the B-domain of staphylococcal protein A using a united-residue representation of the polypeptide chain. As opposed to DEM, the SM, DSM, and SCBDBM approaches, although not so elegant from the theoretical point of view, involve simple transformations of the potential-energy surface and are, therefore, much better for practical use than DEM, which requires solving a parabolic differential equation in multiple dimensions and with complicated boundary conditions, which is a highly non-trivial task.

Another approach related to deformation of the potential-energy surface has been developed by K.A. Olszewski, L. Piela and H.A. Scheraga [55] and termed Self-Consistent Mean Torsional Field (SCMTF) method. It is based on the idea that the ground-state solution of the *Schroedinger equation* contains information about the location of the global minimum. Their implementation uses a *mean field approximation* to solve a set of coupled Schroedinger equations in a dihedral-angle space. Each equation describes the changes of a single dihedral angle in the averaged field of the others. This approach was successful in finding the lowest-energy conformations of Met-enkephalin [55], and decaglycine and eikosaalanine chains [56].

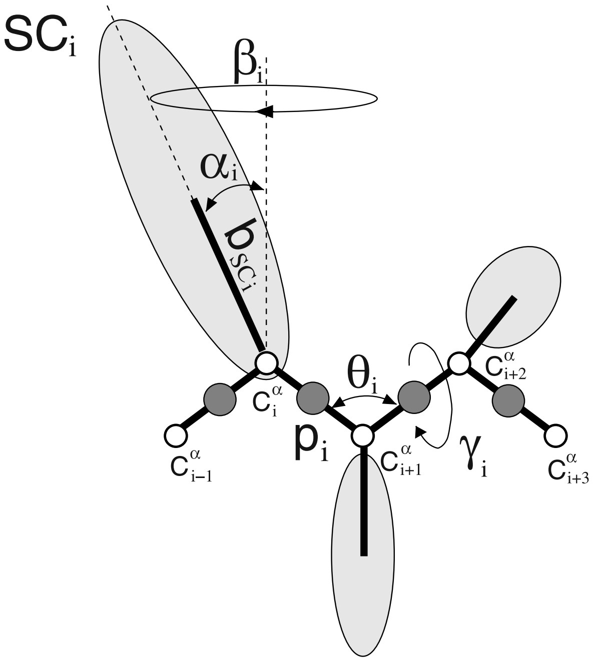
The Conformational Space Annealing Method

One of the most efficient methods to search the conformational space of polypeptide chains developed in our laboratory is the Conformational Space Annealing (CSA) method [19,21,22,24], which combines the ideas of genetic algorithms, the build-up procedure, random search, and local minimization. The CSA method begins with a randomly-generated population of conformations which are energy minimized to generate the *first bank* of conformations. The first bank is meant to represent a sparse sampling of the conformational space that captures short-range interactions. From the initial population, a number of conformations (called seeds) are selected as parents for the trial population. These "seed" conformations are altered in a nonrandom fashion to create new trial conformations. As in any genetic algorithm, the trial population is generated by the use of genetic operators: mutations and crossovers. Unlike traditional genetic algorithms, the mutation operator applied in CSA does not change the value of the selected variable randomly; instead it uses values of the corresponding variables in the initial population (the first bank) or in the current population of conformations as a pool of random numbers. A copy of the first bank is used as a source of "random" variables, which are not uniformly distributed but their distribution is determined by intramolecular interactions at this stage, mainly by steric overlap. The crossover operators copy a set of variables representing a continuous segment of the polypeptide chain of various size taken from a randomly selected conformation in the current population to a selected parent conformation (seed). This is described in detail in the next section. Attention is paid to assure that all trial conformations are significantly different from each other and from the parent conformations. After generation, all trial conformations are energy minimized. The next step of the CSA algorithm is the update of the current population (the bank) without increasing its size. Each trial conformation is compared to each existing conformation of the bank. If the trial conformation is similar to an existing conformation of the bank, only the lower-energy conformation out of these two is preserved. If the trial conformation is not similar to any existing conformation in the bank it represents a new distinct region of conformational space. Then it replaces the highest-energy conformation in the bank, if its energy is lower than the highest energy in the bank, otherwise it is discarded. The distance between conformations i and j is defined as the difference of their dihedral angles. If the distance, D_{ij}, is less than or equal to some predefined cutoff value, D_{cut}, conformations i and j are considered similar, otherwise they are considered different. CSA achieves its efficacy by beginning with a large D_{cut} value to essentially search all possible structures, and then gradually reduces ("anneals") D_{cut} by reducing the minimum distance between the conformations of the bank and focusing the search in low-energy regions of conformational space. After updating the current population, the seed conformations are selected from the set of conformations not selected as seeds previously; additionally attention is paid to cover the conformational space as broadly as possible by selecting conformations not similar to each other as seed conformations.

The CSA method was shown to be very efficient in finding the global minimum of the ECEPP/3 potential energy function for Metenkephalin [22] and melittin [24]; it was also implemented as a standard search technique with the coarse-grained UNRES force field developed in our laboratory (see next section).

Hierarchical Approach

Another approach developed in our laboratory [38,87] starts with a coarse-grained representation of a protein and provides atomistic details at the end. It can be summarized in the following three stages:



Global Optimization in Protein Folding, Figure 8
The UNRES model of polypeptide chains. The interaction sites are side-chain centroids of different sizes (SC) and the peptide-bond centers (*p*) indicated by shaded circles, whereas the α-carbon atoms (small empty circles) are introduced only to assist in defining the geometry. The virtual C^{α}–C^{α} bonds have a length of 3.8 Å, corresponding to a trans peptide group; θ and γ, denoting the virtual-bond angle and virtual-bond dihedral angle, respectively, are variable. Each side chain is attached to the corresponding α-carbon with a "bond length", b_{SC_i}, variable "bond angle", α_{SC_i}, formed by SC_i and the bisector of the angle defined by C^{α}_{i-1}, C^{α}_{i}, and C^{α}_{i+1}, and with a variable "dihedral angle" β_{SC_i} of counterclockwise rotation about the bisector, starting from the right side of the C^{α}_{i-1}, C^{α}_{i}, C^{α}_{i+1} frame

1 Extensive simulations with using the coarse-grained UNRES model [28,29,35,36,37,53,54] developed in our laboratory and subsequent selection of structures with the lowest free energy.
2 Conversion of selected coarse-grained structures to all-atom structures.
3 Exploration of the conformational space of all-atom structures in the neighborhood of geometries obtained in Stage 2.

In the UNRES model, a polypeptide chain is represented as a sequence of α-carbon atoms (C^{α}) with attached united side chains (SC) and united peptide groups (p), each of which is positioned in the middle between two consecutive C^{α} atoms, as shown in Fig. 8.

All three stages are executed using physics-based potentials; therefore, energy is the determinant of each of them. Stage 1 is the key point of the approach, because it provides the widest range of exploration of the conformational space. Consequently, we have put most of our effort in the development of the coarse-grained UNRES force field. To execute stage 2, we developed an approach in which the peptide groups are positioned first within an α-carbon trace to minimize their energy of local and electrostatic interactions [13] and, subsequently, the side-chain atoms are added to minimize the energy of the chain given a coarse-grained geometry [14]. The all-atom ECEPP/3 [49] force field is used in stage 3.

The effective energy function is a sum of different terms corresponding to interactions between the SC ($U_{\mathrm{SC}_i\mathrm{SC}_j}$), SC and p ($U_{\mathrm{SC}_i p_j}$), and p ($U_{p_i p_j}$) sites, as well as local terms corresponding to bending of virtual-bond angles θ (U_b), side-chain rotamers (U_{rot}), virtual-bond torsional (U_{tor}) and double-torsional (U_{tord}) terms, virtual-bond-stretching (U_{bond}) terms, correlation terms ($U^{(m)}_{\mathrm{corr}}$) pertaining to coupling between backbone-local and backbone-electrostatic interactions [29] (where m denotes the order of correlation), and a term accounting for the energetics of disulfide bonds (U_{SS}). Each of these terms is multiplied by an appropriate weight, w. The energy function is given by Eq. (23).

$$\begin{aligned}
U = {} & w_{\mathrm{SC}} \sum_{i<j} U_{\mathrm{SC}_i\mathrm{SC}_j} + w_{\mathrm{SC}p} \sum_{i \neq j} U_{\mathrm{SC}_i p_j} \\
& + w_{pp} \sum_{i<j-1} U^{el}_{p_i p_j} + w_{\mathrm{tor}} \sum_i U_{\mathrm{tor}}(\gamma_i) \\
& + w_{\mathrm{tord}} \sum_i U_{\mathrm{tord}}(\gamma_i, \gamma_{i+1}) + w_b \sum_i U_b(\theta_i) \\
& + w_{\mathrm{rot}} \sum_i U_{\mathrm{rot}}(\alpha_{\mathrm{SC}_i}, \beta_{\mathrm{SC}_i}) \\
& + \sum_{m=3}^{6} w^{(m)}_{\mathrm{corr}} U^{(m)}_{\mathrm{corr}} \\
& + w_{\mathrm{bond}} \sum_{i=1}^{n\mathrm{bond}} U_{\mathrm{bond}}(d_i) + w_{SS} \sum_i U_{SS;i} \, .
\end{aligned} \qquad (23)$$

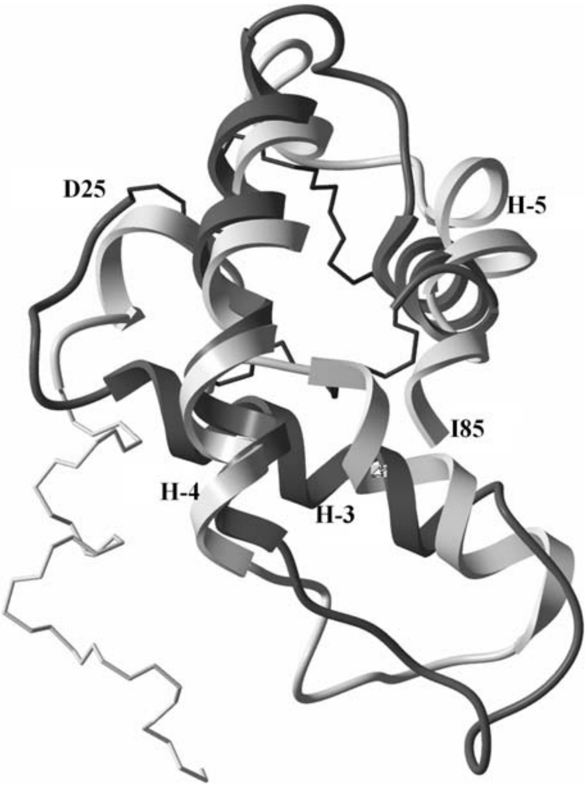
The expression for the effective energy in the UNRES model was derived based on the physics of interactions, as a cluster-cumulant [18] expansion of the effective free energy of a protein plus the surrounding solvent, in which the secondary degrees of freedom had been averaged out [29,31,35]. Most of the expressions were parameterized based on energy surfaces of models systems computed by ab initio molecular quantum mechanics [35,53]; some of them were parameterized based on the statistics from the PDB [36,37]. The energy-term weights (the w's in Eq. (23)) were determined [54] by using the method of hierarchical optimization of the potential-energy landscape developed in our laboratory [28], in which the energy of selected training proteins decreases with increasing native-likeness.

Using the Conformational Space Annealing (CSA) method [19,21,22] to search for the global energy minimum of the UNRES energy function, we achieved considerable success in the Community Wide Experiments of Techniques for Protein Structure Prediction (CASP). In CASP3, we made the best prediction for target T0061 (protein HDEA), predicting its 60-residue segment within 4.2 Å C^{α} RMSD from the experimental structure (PDB code: 1BG8) [34]. The experimental and predicted structures are superposed in Fig. 9.

At that time, our force field did not contain sufficient correlation terms and was unable to account for β-sheet formation. After introducing correlation terms [29], in the CASP4 – CASP6 experiments we were able to predict significant portions of the structures of $\alpha + \beta$ and β proteins [52,64]. In the CASP6 experiment [52], we predicted complete structures of five proteins and large portions of structure of other protein *without* ancillary information from protein structural databases. The largest α-helical protein, the whole of which except for a short C-terminal fragment was predicted in CASP6 was target T0198 (235 residues; we predicted the topology of its 208-residue α-helical part) and the largest $\alpha + \beta$ protein was T0230 (97 residues).

We extended our hierarchical approach to treat oligomeric proteins [83,84] and to proteins containing disulfide bonds [5]; the second extension includes the energy-based prediction of disulfide-bond topology.

Recently [33] we extended the implementation of the UNRES force field to mesoscopic dynamics. The corresponding simulations led us to the conclusion that conformational entropy makes a major contribution to the probability of occurrence of a family of conformations. A particular single conformation can have a very low *potential* energy but no chance to appear at room temperature if it belongs to a very narrow basin in the potential-energy surface. On the contrary, higher-energy conformations could form a very broad basin and, consequently, make an overwhelming contribution to the statistical ensemble at room temperature. Consequently, in our latest work [32] we have reformulated energy-based protein-structure prediction as a search of the basin with the lowest free energy at physiological temperatures, by using techniques based on molecular dynamics, such as replica-exchange molecular dynamics [47] to search conformational space.



Global Optimization in Protein Folding, Figure 9
Superposition of the crystal (dark grey) and predicted (light gray) structures of HDEA. The C^{α} atoms of the fragment included between residues D25 to I85 were superposed. The RMSD is 4.2 Å. Helices 3, 4 and 5 are indicated as H-3, H-4 and H-5, respectively

References

1. Androulakis IP, Maranas CD, Floudas CA (1997) Prediction of oligopeptide conformations via deterministic global optimization. J Glob Optim 11:1–34

2. Ashkenazi G, Ripoll DR, Lotan N, Scheraga HA (1997) A molecular switch for biological logic gates: conformational studies. Biosens Bioelectron 12:85–95
3. Bharucha-Reid AT (1960) Elements of the theory of Markov processes and their applications. McGraw-Hill, New York
4. Chou K-C, Némethy G, Scheraga HA (1983) Energetic approach to the packing of α-helices. 1. Equivalent helices. J Phys Chem 87:2869–2881
5. Czaplewski C, Ołdziej S, Liwo A, Scheraga HA (2004) Prediction of the structures of proteins with the UNRES force field, including dynamic formation and breaking of disulfide bonds. PEDS 17:29–36
6. Dygert M, Gō N, Scheraga HA (1975) Use of a symmetry condition to compute the conformation of gramicidin S. Macromolecules 8:750–761
7. Faerman CH, Ripoll DR (1992) Conformational analysis of a twelve-residue analogue of mastoparan and mastoparan X. Proteins Struc Func Gen 12:111–116
8. Gay DM (1983) Algorithm 611. Subroutines for unconstrained minimization using a model/trust-region approach. ACM Trans Math Softw 9:503–524
9. Gibson KD, Scheraga HA (1987) Revised algorithms for the build-up procedure for predicting protein conformations by energy minimization. J Comput Chem 8:826–834
10. Hagler AT, Stern PS, Sharon R, Becker JM, Naider F (1979) Computer simulation of the conformational properties of oligopeptides. Comparison of theoretical methods and analysis of experimental results. J Am Chem Soc 101:6842–6852
11. Hol WGJ (1985) The role of the α-helix dipole in protein function and structure. Prog Biophys Molec Biol 45: 149–195
12. Hol WGJ, Halie LM, Sander C (1981) Dipoles of the α-helix and β-sheet: their role in protein folding. Nature 294: 532–536
13. Kaźmierkiewicz R, Liwo A, Scheraga HA (2002) Energy-based reconstruction of a protein backbone from its α-carbon-trace by a Monte Carlo method. J Comput Chem 23:715–723
14. Kaźmierkiewicz R, Liwo A, Scheraga HA (2003) Addition of side chains to a known backbone with defined side-chain centroids. Biophys Chem 100:261–280, Erratum: Biophys Chem 106:91
15. Kirkpatrick S, Gelatt CD Jr, Vecchi MP (1983) Optimization by simulated annealing. Science 220:671–680
16. Kostrowicki J, Piela L, Cherayil BJ, Scheraga HA (1991) Performance of the diffusion equation method in searches for optimum structures of clusters of Lennard-Jones atoms. J Phys Chem 95:4113–4119
17. Kostrowicki J, Scheraga HA (1992) Application of the diffusion equation method for global optimization to oligopeptides. J Phys Chem 96:7442–7449
18. Kubo R (1962) Generalized cumulant expansion method. J Phys Soc Japan 17:1100–1120
19. Lee J, Liwo A, Scheraga HA (1999) Energy-based *den-ovo* protein folding by conformational space annealing and an off-lattice united-residue force field: application to the 10-55 fragment of staphylococcal protein A and to apo calbindin D9K. Proc Natl Acad Sci USA 96:2025–2030
20. Lee J, Scheraga HA, Rackovsky S (1997) New optimization method for conformational energy calculations on polypeptides: Conformational space annealing. J Comput Chem 18:1222–1232
21. Lee J, Scheraga HA (1999) Conformational space annealing by parallel computations: extensive conformational search of Met-enkephalin and of the 20-residue membrane-bound portion of melittin. Int J Quant Chem 75:255–265
22. Lee J, Scheraga HA, Rackovsky S (1997) New optimization method for conformational energy calculations on polypeptides: conformational space annealing. J Comput Chem 18:1222–1232
23. Lee J, Scheraga HA, Rackovsky S (1998) Conformational analysis of the 20-residue membrane-bound portion of melittin by conformational space annealing. Biopolymers 46:103–115
24. Lee J, Scheraga HA, Rackovsky S (1998) Conformational analysis of the 20-residue membrane-bound portion of melittin by conformational space annealing. Biopolymers 46:103–115
25. Levitt M, Chothia C (1976) Structural patterns in globular proteins. Nature 261:552–558
26. Li Z, Scheraga HA (1987) Monte Carlo-minimization approach to the multiple-minima problem in protein folding. Proc Natl Acad Sci USA 84:6611–6615
27. Li Z, Scheraga HA (1988) Structure and free energy of complex thermodynamic systems. J Molec Str (Theochem) 179:333–352
28. Liwo A, Arłukowicz P, Czaplewski C, Ołdziej S, Pillardy J, Scheraga HA (2002) A method for optimizing potential-energy functions by a hierarchical design of the potential-energy landscape: application to the UNRES force field. Proc Natl Acad Sci USA 99:1937–1942
29. Liwo A, Czaplewski C, Pillardy J, Scheraga HA (2001) Cumulant-based expressions for the multibody terms for the correlation between local and electrostatic interactions in the united-residue force field. J Chem Phys 115:2323–2347
30. Liwo A, Gibson KD, Scheraga HA, Brandt-Rauf PW, Monaco R, Pincus MR (1994) Comparison of the low energy conformations of an oncogenic and a non-oncogenic p21 protein, neither of which binds GTP or GDP. J Protein Chem 13:237–251
31. Liwo A, Kaźmierkiewicz R, Czaplewski C, Groth M, Ołdziej S, Wawak RJ, Rackovsky S, Pincus MR, Scheraga HA (1998) United-residue force field for off-lattice protein-structure simulations; III. Origin of backbone hydrogen-bonding cooperativity in united-residue potentials. J Comput Chem 19:259–276

32. Liwo A, Khalili M, Czaplewski C, Kalinowski S, Ołdziej S, Wachucik K, Scheraga HA (2007) Modification and optimization of the united-residue (UNRES) potential energy function for canonical simulations. I. Temperature dependence of the effective energy function and tests of the optimization method with single training proteins. J Phys Chem B 111:260–285
33. Liwo A, Khalili M, Scheraga HA (2005) Molecular dynamics with the united-residue (UNRES) model of polypeptide chains; test of the approach on model proteins. Proc Natl Acad Sci USA 102:2362–2367
34. Liwo A, Lee J, Ripoll DR, Pillardy J, Scheraga HA (1999) Protein structure prediction by global optimization of a potential energy function. Proc Natl Acad Sci USA 96:5482–5485
35. Liwo A, Ołdziej S, Czaplewski C, Kozłowska U, Scheraga HA (2004) Parameterization of backbone-electrostatic and multibody contributions to the UNRES force field for protein-structure prediction from ab initio energy surfaces of model systems. J Phys Chem B 108:9421–9438
36. Liwo A, Ołdziej S, Pincus MR, Wawak RJ, Rackovsky S, Scheraga HA (1997) A united-residue force field for off-lattice protein-structure simulations. I. Functional forms and parameters of long-range side-chain interaction potentials from protein crystal data. J Comput Chem 18:849–873
37. Liwo A, Pincus MR, Wawak RJ, Rackovsky S, Ołdziej S, Scheraga HA (1997) A united-residue force field for off-lattice protein-structure simulations. II: Parameterization of local interactions and determination of the weights of energy terms by Z-score optimization. J Comput Chem 18: 874–887
38. Liwo A, Pincus MR, Wawak RJ, Rackovsky S, Scheraga HA (1993) Prediction of protein conformation on the basis of a search for compact structures; test on avian pancreatic polypeptide. Protein Sci 2:1715–1731
39. Liwo A, Tempczyk A, Ołdziej S, Shenderovich MD, Hruby VJ, Talluri S, Ciarkowski J, Kasprzykowski F, Łankiewicz L, Grzonka Z (1996) Exploration of the conformational space of oxytocin and arginine-vasopressin using the electrostatically-driven Monte Carlo and molecular dynamics methods. Biopolymers 38:157–175
40. Metropolis N, Rosenbluth AW, Rosenbluth MN, Teller AH, Teller E (1953) Equation of state calculations by fast computing machines. J Chem Phys 21:1087–1092
41. Miller MH, Némethy G, Scheraga HA (1980) Calculation of the structures of collagen models. Role of interchain interactions in determining the triple-helical coiled-coil conformation. 2. Poly(glycyl-prolyl-hydroxyprolyl). Macromolecules 13:470–478
42. Miller MH, Némethy G, Scheraga HA (1980) Calculation of the structures of collagen models. Role of interchain interactions in determining the triple-helical coiled- coil conformation. 3. Poly(glycyl-prolyl-alanyl). Macromolecules 13:910–913
43. Miller MH, Scheraga HA (1976) Calculation of the structures of collagen models. Role of interchain interactions in determining the triple-helical coiled-coil conformation. I. Poly(glycyl-prolyl-prolyl). J Polym Sci Polym Symposia 54:171–200
44. Momany FA, McGuire RF, Burgess AW, Scheraga HA (1975) Energy parameters in polypeptides. VII. Geometric parameters, partial atomic charges, non-bonded interactions, hydrogen bond interactions and intrinsic torsional potential for the naturally occurring amino-acids. J Phys Chem 79:2361–2381
45. Morales LB, Garduño Juárez RG, Romero D (1991) Applications of simulated annealing to the multiple-minima problem in small peptides. J Biomol Struct Dyn 8:721–735
46. Morales LB, Garduño Juárez RG, Romero D (1992) The multiple-minima problem in small peptides revisited. The Threshold Accepting approach. J Biomol Struct Dyn 9: 951–957
47. Nanias M, Czaplewski C, Scheraga HA (2006) Replica exchange and multicanonical algorithms with the coarse-grained united-residue (UNRES) force field. J Chem Theor Comput 2:513–528
48. Nayeem A, Vila J, Scheraga HA (1991) A comparative study of simulated-annealing and Monte Carlo-with-minimization approaches to the minimum-energy structures of polypeptides: Metenkephalin. J Comp Chem 12:595–605
49. Némethy G, Gibson KD, Palmer KA, Yoon CN, Paterlini G, Zagari A, Rumsey S, Scheraga H (1992) Energy parameters in polypeptides. 10. Improved geometrical parameters and nonbonded interactions for use in the ECEPP/3 algorithm, with application to proline-containing peptides. J Phys Chem 96:6472–6484
50. Némethy G, Pottle MS, Scheraga HA (1983) Energy parameters in polypeptides. 9. Updating of geometrical parameters, nonbonded interactions, and hydrogen bond interactions for the naturally occurring amino acids. J Phys Chem 87:1883–1887
51. Némethy G, Scheraga HA (1984) Hydrogen bonding involving the ornithine side chain of gramicidin S. Biochem Biophys Res Commun 118:643–647
52. Ołdziej S, Czaplewski C, Liwo A, Chinchio M, Nanias M, Vila JA, Khalili M, Arnautova YA, Jagielska A, Makowski M, Schafroth HD, Kaźmierkiewicz R, Ripoll DR, Pillardy J, Saunders JA, Kang Y-K, Gibson KD, Scheraga HA (2005) Physics-based protein-structure prediction using a hierarchical protocol based on the UNRES force field – test with CASP5 and CASP6 targets. Proc Natl Acad Sci USA 102:7547–7552
53. Ołdziej S, Kozłowska U, Liwo A, Scheraga HA (2003) Determination of the potentials of mean force for rotation about $C^{\alpha} \cdots C^{\alpha}$ virtual bonds in polypeptides from the ab initio energy surfaces of terminally-blocked glycine, alanine, and proline. J Phys Chem A 107:8035–8046
54. Ołdziej S, Łagiewka J, Liwo A, Czaplewski C, Chinchio M, Nanias M, Scheraga HA (2004) Optimization of the UNRES

force field by hierarchical design of the potential-energy landscape: III. Use of many proteins in optimization. J Phys Chem B 108:16950–16959

55. Olszewski KA, Piela L, Scheraga HA (1992) Mean-field theory as a tool for intramolecular conformational optimization. 1. Tests on terminally-blocked alanine and Metenkephalin. J Phys Chem 96:4672–4676
56. Olszewski KA, Piela L, Scheraga HA (1993) Mean field theory as a tool for intramolecular conformational optimization. 2. Tests on the homopolypeptides decaglycine and icosalanine. J Phys Chem 97:260–266
57. Paine GH, Scheraga HA (1985) Prediction of the native conformation of a polypeptide by a statistical-mechanical procedure. I. Backbone structure of enkephalin. Biopolymers 24:1391–1436
58. Paine GH, Scheraga HA (1986) Prediction of the native conformation of a polypeptide by a statistical-mechanical procedure. II. Average backbone structure of enkephalin. Biopolymers 25:1547–1563
59. Paine GH, Scheraga HA (1987) Prediction of the native conformation of a polypeptide by a statistical-mechanical procedure. III. Probable and average conformations of enkephalin. Biopolymers 26:1125–1162
60. Perutz MF (1978) Electrostatic effects in proteins. Science 201:1187–1191
61. Piela L, Kostrowicki J, Scheraga HA (1989) The multiple-minima problem in the conformational analysis of molecules. Deformation of the potential energy hypersurface by the diffusion equation method. J Phys Chem 93:3339–3346
62. Piela L, Scheraga HA (1987) On the multiple-minima problem in the conformational analysis of polypeptides. I. Backbone degrees of freedom for a perturbed α-helix. Biopolymers 26:S33–S58
63. Pillardy J, Arnautova YA, Czaplewski C, Gibson KD, Scheraga HA (2001) Conformation-family Monte Carlo: a new method for crystal structure prediction. Proc Natl Acad Sci USA 98:12351–12356
64. Pillardy J, Czaplewski C, Liwo A, Lee J, Ripoll DR, Kaźmierkiewicz R, Ołdziej S, Wedemeyer WJ, Gibson KD, Arnautova YA, Saunders J, Ye Y-J, Scheraga HA (2001) Recent improvements in prediction of protein structure by global ptimization of a potential energy function. Proc Natl Acad Sci USA 98:2329–2333
65. Pillardy J, Czaplewski C, Wedemeyer WJ, Scheraga HA (2000) Conformation-family Monte Carlo (CFMC): an efficient computational tool for identifying the low-energy states of a macromolecule. Helv Chim Acta 83:2214–2230
66. Pillardy J, Liwo A, Groth M, Scheraga HA (1999) An efficient deformation-based global optimization method for off-lattice polymer chains; self-consistent basin-to-deformed-basin mapping (SCBDBM). Application to united-residue polypeptide chains. J Phys Chem B 103:7353–7366
67. Pillardy J, Liwo A, Scheraga HA (1999) An efficient deformation-based global optimization method (Self-Consistent Basin-to-Deformed-Basin Mapping (SCBDBM)). Application to Lennard-Jones atomic clusters. J Phys Chem A 103:9370–9377
68. Pillardy J, Olszewski KA, Piela L (1992) Performance of the shift method of global minimization in searches for optimum structures of clusters of Lennard-Jones atoms. J Phys Chem 96:4337–4341
69. Pillardy J, Olszewski KA, Piela L (1992) Theoretically predicted lowest-energy structures of water clusters. J Mol Struct (Theochem) 270:277–285
70. Pillardy J, Piela L (1997) Smoothing techniques of global optimization. The distance scaling method in searches for the most stable Lennard-Jones atomic clusters. J Comp Chem 18:2040–2049
71. Pincus MR, Klausner RD, Scheraga HA (1982) Calculation of the three-dimensional structure of the membrane-bound portion of melittin from its amino acid sequence. Proc Natl Acad Sci USA 79:5107–5110
72. Pincus MR, Murphy RB, Carty RP, Chen J, Shah D, Scheraga HA (1988) Conformational analysis of possible biologically active (receptor-bound) conformations of peptides derived from cholecystokinin, cerulein and little gastrin and the opiate peptide, Metenkephalin. Peptides 9(1):145–152
73. Purisima EO, Scheraga HA (1987) An approach to the multiple-minima problem in protein folding by relaxing dimensionality. Tests on enkephalin. J Mol Biol 196:697–709
74. Rapaport DC, Scheraga HA (1981) Evolution and stability of polypeptide chain conformation: a simulation study. Macromolecules 14:1238–1246
75. Ripoll DR (1992) Conformational study of a peptide epitope shows large preferences for β-turn conformations. Int J Pept Protein Res 40:575–581
76. Ripoll DR, Piela L, Vásquez M, Scheraga HA (1991) On the multiple-minima problem in the conformational analysis of polypeptides. V. Application of the self-consistent electrostatic field and the electrostatically driven Monte Carlo methods to bovine pancreatic trypsin inhibitor. Proteins Struc Func Gen 10:188–198
77. Ripoll DR, Liwo A, Scheraga HA (1998) New developments of the electrostatically driven Monte Carlo method – Test on the membrane bound portion of melittin. Biopolymers 46:117–126
78. Ripoll DR, Scheraga HA (1988) On the multiple-minima problem in the conformational analysis of polypeptides. II. An electrostatically driven Monte Carlo method-tests on poly(L-alanine). Biopolymers 27:1283–1303
79. Ripoll DR, Scheraga HA (1989) The multiple-minima problem in the conformational analysis of polypeptides. III. An electrostatically driven Monte Carlo method; tests on enkephalin. J Protein Chem 8:263–287
80. Ripoll DR, Vásquez MJ, Scheraga HA (1991) The electrostatically driven Monte Carlo method: Application to conformational analysis of decaglycine. Biopolymers 31:319–330

81. Ripoll DR, Vila JA, Scheraga HA (2004) Folding of the villin headpiece subdomain from random structures. Analysis of the charge distribution as a function of the pH. J Mol Biol 339:915–925
82. Ripoll DR, Vila JA, Scheraga HA (2005) On the orientation of the backbone dipoles in native folds. Proc Natl Acad Sci USA 102:7559–7564
83. Saunders JA, Scheraga HA (2003) Ab initio structure prediction of two α-helical oligomers with a multiple-chain united-residue force field and global search. Biopolymers 68:300–317
84. Saunders JA, Scheraga HA (2003) Challenges in structure prediction of oligomeric proteins at the united-residue level: searching the multiple-chain energy landscape with CSA and CFMC procedures. Biopolymers 68:318–332
85. Scheraga HA (1974) Prediction of protein conformation. In: Anfinsen CB, Schechter AN (eds) Current Topics in Biochemistry, 1973. Academic Press, New York, pp 1–42
86. Scheraga HA (1983) Recent progress in the theoretical treatment of protein folding. Biopolymers 22:1–14
87. Scheraga HA, Liwo A, Ołdziej S, Czaplewski C, Pillardy J, Ripoll DR, Vila JA, Kaźmierkiewicz R, Saunders JA, Arnautova YA, Jagielska A, Chinchio M, Nanias M (2004) The protein folding problem: global optimization of force fields. Front Biosci 9:3296–3323
88. Simon I, Némethy G, Scheraga HA (1978) Conformational energy calculations of the effects of sequence variations on the conformations of two tetrapeptides. Macromolecules 11:797–804
89. Sippl MJ, Némethy G, Scheraga HA (1984) Intermolecular potentials from crystal data. 6. Determination of empirical potentials for O-H$\cdots$O-C hydrogen bonds from packing configurations. J Phys Chem 88:6231–6233
90. Vásquez M, Némethy G, Scheraga HA (1983) Computed conformational states of the 20 naturally occurring amino acid residues and of the prototype residue a-aminobutyric acid. Macromolecules 16:1043–1049
91. Vásquez M, Scheraga HA (1985) Use of buildup and energy-minimization procedures to compute low-energy structures of the backbone of enkephalin. Biopolymers 24:1437–1447
92. Vásquez M, Scheraga HA (1988) Calculation of protein conformation by the build-up procedure. Application to bovine pancreatic trypsin inhibitor using limited simulated nuclear magnetic resonance data. J Biomol Struct Dyn 5:705–755
93. Vásquez M, Scheraga HA (1988) Variable-target-function and build-up procedures for the calculation of protein conformation. Application to bovine pancreatic trypsin inhibitor using limited simulated nuclear magnetic resonance data. J Biomol Struct Dyn 5:757–784
94. Vila JA, Ripoll DR, Scheraga HA (2003) Atomically detailed folding simulation of the B domain of staphylococcal protein A from random structures. Proc Natl Acad Sci USA 100:14812–14816
95. Wada A (1976) The α-helix as an electric macro-dipole. Adv Biophys 9:1–63
96. Wawak RJ, Gibson KD, Liwo A, Scheraga HA (1996) Theoretical prediction of a crystal structures. Proc Natl Acad Sci USA 93:1743–1746
97. Wawak RJ, Pillardy J, Liwo A, Gibson KD, Scheraga HA (1998) The diffusion equation and distance scaling methods of global optimization; applications to crystal structure prediction. J Phys Chem 102:2904–2918
98. Zimmerman SS, Pottle MS, Némethy G, Scheraga HA (1977) Conformational analysis of the twenty naturally occurring amino acid residues using ECEPP. Macromolecules 10:1–9

Global Optimization: Tight Convex Underestimators

CHRYSANTHOS E. GOUNARIS,
CHRISTODOULOS A. FLOUDAS
Department of Chemical Engineering,
Princeton University, Princeton, USA

MSC2000: 49M37, 65K10, 90C26, 90C30, 46N10, 47N10

Article Outline

Keywords and Phrases

Convex underestimators; αBB; Global optimization

Introduction

In their effort to locate the global solution, deterministic global optimization algorithms, like the αBB [1,2,6,14], employ a branch and bound framework. During this process, convex underestimation techniques are used to formulate relaxed convex problems that can be solved to optimality with the use of local solvers, thus providing valid lower bounds for the original problem. The tightness of the underestimators used is of fundamental importance for the computational performance of these algorithms, since a tighter relaxation can lead to faster fathoming and less nodes of the branch and bound tree to be visited [7]. A recent review article on deterministic global optimization approaches can be found in [8].

In the case of arbitrary nonconvex functions that do not exhibit an exploitable mathematical structure, the αBB general underestimator [3,6] can be used:

$$L(x) = f(x) - \sum_{v=1}^{V} \alpha_v (x_v - x_v^L)(x_v^U - x_v) \, . \tag{1}$$

Originally introduced in [14], this underestimator derives from the function by subtracting a positive quadratic ($\alpha_v \geq 0 \forall v$). Given sufficiently large values of the α_v parameters, all nonconvexities in the original function $f(x)$ can be overpowered, resulting into a convex underestimator $L(x)$ that is valid for the entire domain $[x^L, x^U]$. A number of rigorous methods have been devised in order to select appropriate values for these parameters [2,3,13]. Extensive computational testing of the algorithm [1] showed that the most efficient of those methods is the one based on the scaled Gherschgorin theorem. According to this method, it suffices to select:

$$\alpha_v = \max \left\{ 0, -\frac{1}{2} \left(\underline{h_{vv}} - \sum_{\substack{u=1 \\ u \neq v}}^{V} \max \left\{ |\underline{h_{vu}}|, |\overline{h_{vu}}| \right\} \frac{(x_u^U - x_u^L)}{(x_v^U - x_v^L)} \right) \right\} \tag{2}$$

where $\underline{h_{vu}}$ and $\overline{h_{vu}}$ are lower and upper bounds of $\partial^2 f / \partial x_v x_u$ that can be calculated by interval analysis.

One could use alternatively a new class of general purpose convex underestimators that has been developed by Akrotirianakis and Floudas [4,5]. These underestimators are derived in a similar fashion, by subtracting an exponential term from the original function, that is:

$$L_1(x) = f(x) - \sum_{v=1}^{V} \left(1 - e^{\gamma_v (x_v - x_v^L)} \right) \left(1 - e^{\gamma_v (x_v^U - x_v)} \right) \, . \tag{3}$$

An iterative systematic procedure is used to determine the values of the γ_v parameters so as the underestimating function to be convex. The procedure ensures also that the resulting underestimator $L_1(x)$ is tighter than $L(x)$, the one that results from the original method. Floudas and Kreinovich [9,10] have in fact shown that these two functional forms (original quadratic and exponential) are the only optimal ones, since they are the only ones to be shift-, sign- and scale-invariant.

Maranas and Floudas [14] showed that the maximum separation distance between the original function $f(x)$ and the underestimator $L(x)$ of (1) is a quadratic function of interval length. Because of this, as well as because of potentially less overestimation in the interval extension of the Hessian matrix elements h_{vu}, the underestimator would become tighter with shrinkage of the domain under consideration. This was firstly exploited in Meyer and Floudas [15], where a piecewise approach was utilized. The method proposed partitioning of the domain into many subdomains and construction of the corresponding αBB underestimator for each one of them. These underestimators, although not valid for the entire domain, are much tighter in their respective subdomains. A hyperplane is subsequently added to each one of these underestimators and is selected in such a way, so that the combination of all these convex *pieces* results into an overall convex underestimator that is continuous and smooth (C^1-continuity).

This entry describes the work of [11,12] on the development of tight convex underestimators. The construction of these underestimators is based on a piecewise application of the αBB underestimator, in a similar fashion with the p-αBB approach [15], but, instead of adding hyperplanes, we identify those supporting line segments that have to be combined with convex parts of the original underestimators so as to form a C^1-continuous convex underestimator that is valid for the overall domain under consideration. One can also consider only the lines defined by these linear segments, thus coming up with a piecewise linear underestimator that can easily be incorporated in the NLP relaxation as a set of linear constraints.

In their work, Gounaris and Floudas [12] also demonstrated how one can make use of the high quality results of the approach in the univariate case so as to extend its applicability to functions with a higher number of variables. This is achieved by proper projections of the multivariate αBB underestimators into select two-dimensional planes. Furthermore, since the method utilizes projections into lower-dimensional spaces, they explored ways to recover some of the information lost in this process. In particular, they apply the method after having transformed the original problem in an orthonormal fashion. This leads to the construction of even tighter underestimators, through the accumulation of additional valid linear cuts in the relaxation.

Theoretical Results for Univariate Functions

Let $f(x)$ be a univariate function that needs to be underestimated in $D = [x^L, x^U]$. We select an integer $N > 1$ and partition the complete domain in N segments of equal length. Thus, the *i-th* subdomain would be defined as $D_i = [x^{i-1}, x^i]$, where: $x^i = x^L + \frac{i}{N}(x^U - x^L), i = 0, 1, \ldots, N$.

For every subdomain D_i,$i = 1, 2, \ldots, N$, we construct the corresponding αBB underestimator:

$$P_i(x) = f(x) - \alpha^i (x - x^{i-1})(x^i - x)$$
$$\alpha^i = \max\left\{0, -\frac{1}{2}\underline{f''}_{(D_i)}\right\} \tag{4}$$

where $\underline{f''}_{(D_i)}$ is a lower bound of the second derivative that is valid for the entire subdomain D_i.

Note that although an underestimator $P_i\}(x)$ can be defined outside its respective subdomain, its convexity is only guaranteed for $x \in [x^{i-1}, x^i]$.

We define $P(x), x \in [x^L, x^U]$ to be the following branched function:

$$P(x) = P_i(x), \text{ if } \quad x^{i-1} \le x \le x^i \;. \tag{5}$$

Function $P(x)$ is a *piecewise* convex valid underestimator of $f(x)$. Since it is not convex, a convexification technique has to be employed. The proposed technique involves the identification of those supporting line segments that are required for an overall underestimator $U(x)$. The technique is based on two algorithms, called "*inner*" and "*outer*", which are described in detail in [11].

The underestimator $U(x)$ consists of the identified linear parts, as well as convex parts of the underestimators $P_i(x)$, therefore it is a C^1-continuous branched function. This might pose some computational complications if the lower bounding (relaxation) problem is to be solved by local optimization solvers that require C^2-continuity. In order to avoid this problem, one can take into account only the lines defined by the line segments. According to this alternative, we first identify the linear segments needed for the construction of underestimator $U(x)$, but we consider those as lines defined in $[x^L, x^U]$. Let there be K such lines denoted as $T_k(x), k = 1, 2, \ldots, K$ and arranged in order of ascending slope. If applicable, this set can be augmented with lines that are tangential to P_1 and P_N at the respective domain edges x^L and x^U.

Each of these lines T_k is a valid underestimator of function $f(x)$ across the whole domain. We define the function $V(x)$ to be the pointwise maximum of all these lines. $V(x)$ is convex, since it is the pointwise maximum of linear functions and it is obviously an underestimator, since it consists of pieces of other underestimators. At the expense of some tightness (in the regions where underestimator $U(x)$ consisted of convex parts), we now have a piecewise linear underestimator $V(x)$ that can be incorporated in the relaxation as a set of linear constraints. The whole lower bounding problem can now be formulated as a linear programming problem (LP).

Tightness of Univariate Underestimator

It is apparent that as the level of partitioning increases, the underestimator $P(x)$ comes closer to the function, and therefore convex underestimators $U(x)$ and $V(x)$ approach the convex envelope of $f(x)$. Gounaris and Floudas [11] proved the following two theorems that are relevant with the tightness of the resulting underestimators in the univariate case:

Theorem 1. *There is some finite partitioning level N, for which the convex underestimator U(x) is the convex envelope of function f(x).*

Theorem 2. *There is some finite partitioning level N, for which underestimator V(x) is ϵ-close to underestimator U(x), that is:*

$$\max_{x \in D}\{U(x) - V(x)\} < \epsilon \tag{6}$$

where: $\epsilon > 0$ is an arbitrarily small constant.

Since these univariate underestimators are very tight, the remaining question is whether we can exploit them so as to construct underestimators of functions in higher dimensions. Gounaris and Floudas [12] presented some extensions of the method for application on multivariate functions that involve dimension reduction of the problem through proper projections into lower-dimensional spaces. These extensions are described in Sect. "Extension to Multivariate Functions".

Extension to Multivariate Functions

Let $f(x)$ be a function of V variables that needs to be underestimated in a box domain $D = [x_1^L, x_1^U] \times \cdots \times [x_V^L, x_V^U]$. We choose integers $N_v > 1, v = 1, 2, \ldots, V$ and partition each range $[x_v^L, x_v^U]$ in N_v segments of equal length. Thus, the *j-th* segment of the v^{th} set would be defined as $[x_v^{j-1}, x_v^j]$, where: $x_v^j = x_v^L + \frac{j}{N_v}(x_v^U - x_v^L), j = 0, 1, \ldots, N_v$. The complete V-dimensional domain D has now been partitioned into $N = \prod_{v=1}^{V} N_v$ box subdomains of equal measures. Let D_i be such a V-dimensional subdomain. It is uniquely defined by a set of indices $i_v, 1 \le i_v \le N_v, \forall v = 1, 2, \ldots, V$. Thus, the i^{th} subdomain would be defined as $D_i = [x_1^{i_1-1}, x_1^{i_1}] \times \cdots \times [x_V^{i_V-1}, x_V^{i_V}]$.

For every subdomain D_i, $i = 1, 2, \ldots, N$, we construct the corresponding αBB underestimator [1,2,3,6,14]:

$$P_i(x) = f(x) - \sum_{v=1}^{V} \alpha_v^i (x_v - x_v^{i_v-1})(x_v^{i_v} - x_v)$$

$$\alpha_v^i = \max\left\{0, -\frac{1}{2}\left(\underline{h_{vv}^{(i)}} - \sum_{\substack{u=1\\u\neq v}}^{V} \max\left\{|\underline{h_{vu}^{(i)}}|, |\overline{h_{vu}^{(i)}}|\right\} \frac{(x_u^{i_u} - x_u^{i_u-1})}{(x_v^{i_v} - x_v^{i_v-1})}\right)\right\} \qquad (7)$$

where $\underline{h_{vu}^{(i)}}$ and $\overline{h_{vu}^{(i)}}$ are respectively lower and upper bounds of $\partial^2 f/\partial x_v x_u$ that are valid for the entire subdomain D_i.

Note that although an underestimator $P_i(x)$ can be defined outside its respective subdomain, its convexity is only guaranteed for $x \in D_i$.

We select variable $w, 1 \le w \le V$, which we designate to be the *active* variable, and enumerate all $M_w = N/N_w$ permutations of indices i_v, $v \neq w$. Every such permutation m, $1 \le m \le M_w$, corresponds to a subdomain $D_{wm} = [x_w^L, x_w^U] \times \prod_{\substack{v=1\\v\neq w}}^{V} [x_v^{i_v-1}, x_v^{i_v}]$, which can be further divided into N_w subdomains $D_{wmj} = [x_w^{j-1}, x_w^j] \times \prod_{\substack{v=1\\v\neq w}}^{V} [x_v^{i_v-1}, x_v^{i_v}]$, $j = 1, 2, \ldots, N_w$. These subdomains, belong to the set of the original subdomains D_i (for $i_w = j$) and therefore each one has an underestimator $P_{wmj}(x)$ associated with it, that is:

$$P_{wmj}(x) = f(x) - \alpha_w^i (x_w - x_w^{j-1})(x_w^j - x_w) - \sum_{\substack{v=1\\v\neq w}}^{V} \alpha_v^i (x_v - x_v^{i_v-1})(x_v^{i_v} - x_v) \qquad (8)$$

where index i satisfies $D_i = D_{wmj}$ and parameters $\alpha_v^i, v = 1, 2, \ldots, V$ are calculated according to (7).

For every such subdomain $D_{wmj}, j = 1, 2, \ldots, N_w$, we define the following univariate function:

$$G_{wmj}(x_w) = \min_{\substack{x_v\\ \forall v\neq w}} P_{wmj}(x), \quad x_w^{j-1} \le x_w \le x_w^j \,. \qquad (9)$$

Since they correspond to the minimum of a convex function over a subset of its variables, these functions are convex. Furthermore, each one is defined over a different segment of $[x_w^L, x_w^U]$. Therefore, each one can be considered as a convex *piece* of an overall *piecewise* convex underestimator. The latter is fully suitable for application of the convex underestimation method for univariate functions which was described in the previous sections.

Let $V_{wm}(x_w)$ be the piecewise linear underestimator obtained by the univariate method, and let it be the pointwise maximum of K_{wm} associated lines, that is:

$$V_{wm}(x_w) = \max\{T_{wmk}(x_w), \forall k = 1, 2, \ldots, K_{wm}\}, \quad x_w^L \le x_w \le x_w^U \qquad (10)$$

Without loss of generality, let us assume that the lines T_{wmk} are arranged in order of ascending slope, that is, $slope(T_{wm(k-1)}) < slope(T_{wmk}), k = 2, 3, \ldots, K_{wm}$, and that the set already includes the potential augmented tangents at the domain edges, designated earlier as T_0 and T_{K+1}.

Univariate underestimator $V_{wm}(x_w)$ could, in principle, be considered as a multivariate function that is dependent to only one variable, x_w, and defined over the whole multidimensional (dimension V) subdomain D_{wm}. That is:

$$V_{wm}(x_w) \to V_{wm}(x), \quad x \in D_{wm} \qquad (11)$$

Function $V_{wm}(x)$ is piecewise affine and consists of segments of V-dimensional hyperplanes. Since these hyperplanes depend only on the w^{th} variable, they are parallel to all standard basis vectors e_v with the exception of e_w (to which they are parallel only if the slope of the corresponding line T_{wmk} is zero). This function is a valid underestimator for the original function $f(x)$ across the whole subdomain D_{wm}.

Applying the aforementioned procedure for every permutation $m = 1, 2, \dots, M_w$, we come up with a collection of such underestimating segments, each of which is a valid underestimator for the function $f(x)$ across a subset of its original domain D. In order to develop a convex underestimator that would be valid for the whole domain, we have to *combine* all these segments. Let $m = 0$ denote the combination of all permutations $m = 1, 2, \dots, M_w$. This combination can be achieved back in the projection space, by computing the lower hull of the set of all underestimators $V_{wm}(x_w)$. In fact, one needs to consider only the vertex points of each underestimator $V_{wm}(x_k)$ (that is the points of intersection between two lines $T_{wm(k-1)}$ and T_{wmk}), as well as their end points $\left(x_w^L, T_{wm1}(x_w^L)\right)$ and $\left(x_w^U, T_{wm(K_{wm})}(x_w^U)\right)$. Any standard 2d convex hull algorithm (e. g., *Graham-Scan*) can be used for this purpose. The lower hull is a convex piecewise linear function $V_{w0}(x_w)$, and it is the pointwise maximum of K_{w0} lines, that is:

$$\begin{aligned} V_{w0}(x_w) = \max\{T_{w0k}(x_w), \forall k = 1, 2, \dots, K_{w0}\}\,, \\ x_w^L \le x_w \le x_w^U\,. \end{aligned} \tag{12}$$

By construction, this function is a convex underestimator of all pieces $G_{wmj}(x_w)$ for all permutations, that is:

$$\begin{aligned} V_{w0}(x_w) \le & G_{wmj}(x_w), x_w \in [x_w^{j-1}, x_w^j], \\ & \forall j = 1, 2, \dots, N_w, \forall m = 1, 2, \dots, M_w\,. \end{aligned} \tag{13}$$

Therefore, function $V_{w0}(x_w)$, if considered as $V_{w0}(x)$, is a valid underestimator for function $f(x)$ across its whole original domain D.

For any selection of the active variable w, the method will yield a convex (piecewise affine) underestimator which would be valid for the whole domain of interest, D. However, the method can be independently applied for every variable being active (one at a time), leading to a collection of valid underestimators. The pointwise maximum of all these is itself a valid convex underestimator, and is tighter (or equally tight) to the original function than any of its predecessors. Thus, the resulting underestimator is:

$$V(x) = \max\{V_{w0}(x), \forall w = 1, 2, \dots, V\}, x \in D\,. \tag{14}$$

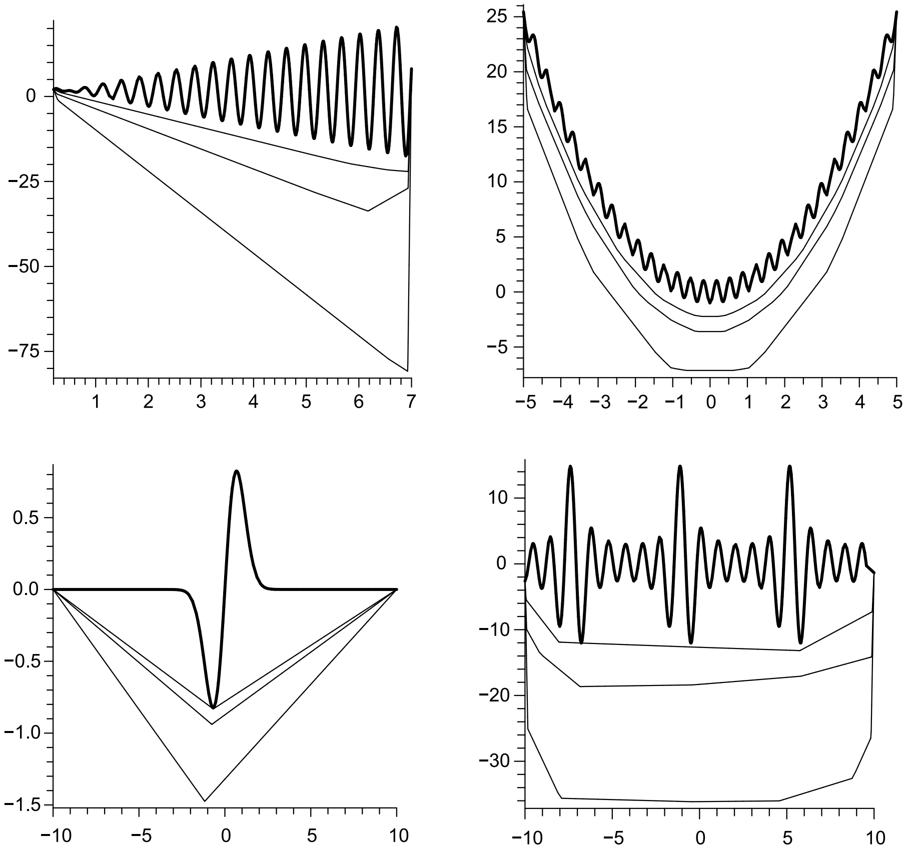
Note that the underestimator $V(x)$ is also piecewise hyperplanar, and can be represented in the problem relaxation as a set of linear constraints. Since we do not know explicitly which hyperplanes $T_{w0k}(x_w) \to T_{w0k}(x)$, $k = 1, 2, \dots, K_{w0}$, $w = 1, 2, \dots, V$ contribute some part of theirs to the overall underestimator $V(x)$, all of them should be included in this relaxation, despite the fact that some may end up being redundant.

Since our method produces piecewise affine underestimators $L \equiv V$, the resulting convex relaxation is just a linear programming problem (LP), which takes the form of (15).

$$\begin{aligned} & \min_{\mu,x} \mu \\ & s.t. \quad \mu \ge T_{w0k}^{(0)}(x_w) \begin{Bmatrix} \forall k = 1, 2, \dots, K_{w0}^{(0)} \\ \forall w = 1, 2, \dots, V \end{Bmatrix} \\ & T_{w0k}^{(q)}(x_w) \le 0 \begin{Bmatrix} \forall k = 1, 2, \dots, K_{w0}^{(q)} \\ \forall w = 1, 2, \dots, V \\ \forall q = 1, 2, \dots, Q \end{Bmatrix} \end{aligned} \tag{15}$$

Domain Rotation

The methodology presented in Section 4 involves the minimization of underestimators $P_{wmj}(x)$, over all their variables with the exception of one, variable x_w, which is designated as "active". Whenever such a projection into spaces of lower dimensionality is involved, there is the possibility that some useful information is lost. Some of this lost information will be recovered if we opt to apply the methodology for every variable being "active", one at a time, which basically calls for projecting into V different two-dimensional planes, each one being parallel to a different basis vector e_v, $v = 1, 2, \dots, V$. However, since there is a finite number of variables in our problem, there is a limited number of planes to which we can project. If we want to enhance further the col-



Global Optimization: Tight Convex Underestimators, Figure 1
Univariate functions f_{1-4} with underestimators $V(x)$ for three different partitioning levels ($N = 24$, 36 and 48)

lection of underestimators that we will eventually accumulate in the relaxation (thus improve our chances for better tightness/lower bound), we will have to project into additional planes, that do not correspond to some variable that is "natural" to the problem, rather than to some linear combination of theirs.

This can be achieved by applying an orthonormal transformation to the problem's variable space, that is:

$$x \to x' = R \cdot x \,. \tag{16}$$

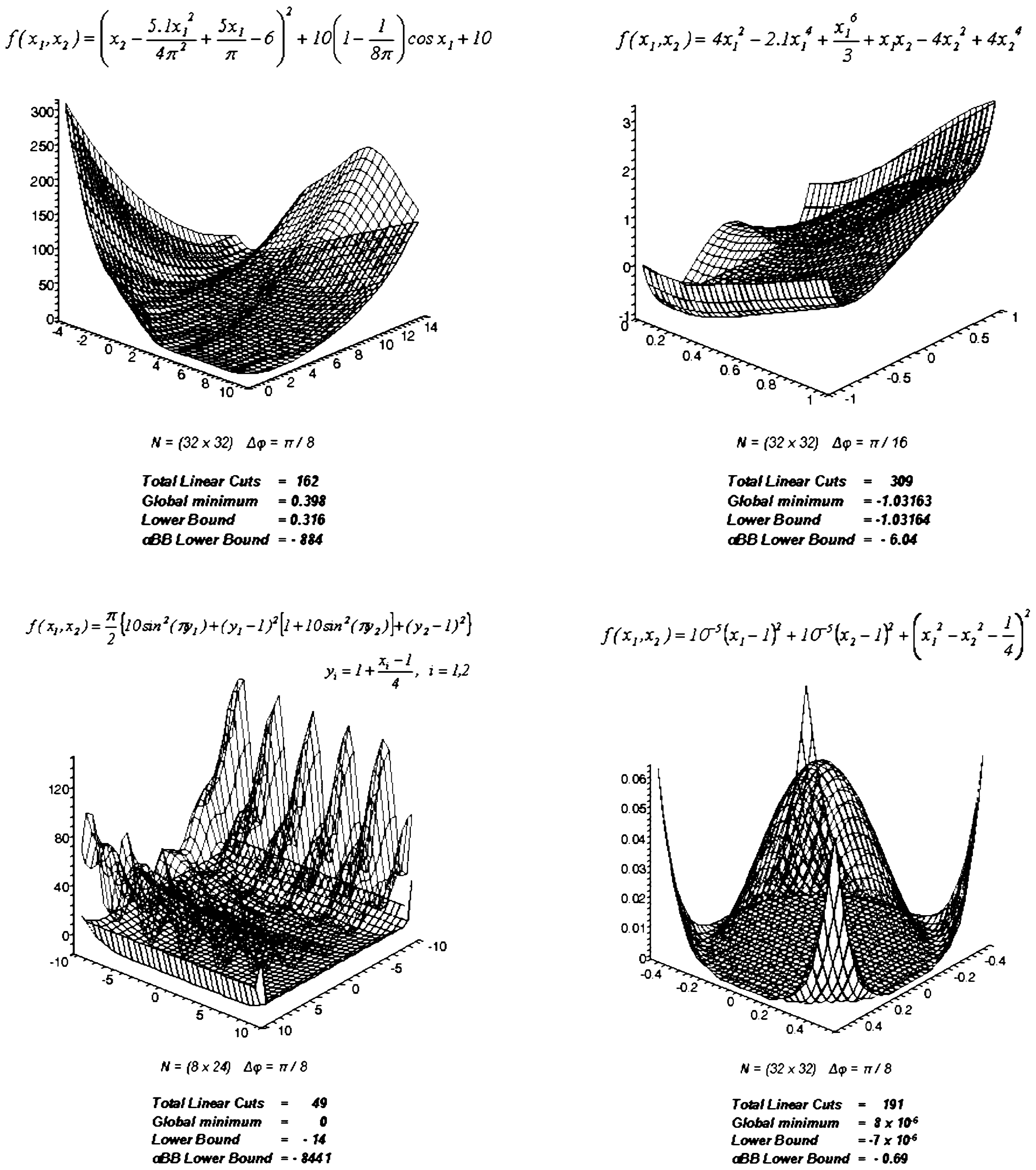
This transformation has to be orthogonal, which means that it should preserve the lengths of vectors and the angles between vectors. Furthermore, it should be an orientation-preserving transformation. A $V \times V$ matrix R that could provide such a transformation is called a *rotation matrix* and has to be a member of the special orthogonal group, that is:

$$R \in SO(V) \Leftrightarrow \begin{cases} R^{-1} = R^T \\ |R| = +1 \,. \end{cases} \tag{17}$$

In their work, Gounaris and Floudas [12] discuss the selection of a suitable such matrix. They rigorously address the issue of selecting a suitable "rotated" domain and some suitable level of partitioning, and they also present a method to calculate appropriate values for the α parameters in the transformed counterpart of the problem.

Examples

Figure 1 depicts the plots for four nonlinear univariate functions. In particular, for functions: $f_1(x) = (3x - 1.4)sin(18x) + 1.7$, $f_2(x) = x^2 - cos(18x)$, $f_3(x) = (x + sinx)e^{-x}$ and $f_4(x) = -\sum_{k=1}^{5} ksin[(k+1)x + k]$.



Global Optimization: Tight Convex Underestimators, Figure 2
Piecewise planar underestimators of bivariate functions

The underestimators presented correspond to partitioning in $N = 24$, 36 and 48 subdomains (increasing tightness).

Figure 2 depicts plots for four nonlinear bivariate functions. For each case, $N_1 \times N_2$ is the level of partitioning used and $\varDelta\varphi$ is the resolution of domain rotation. Some additional information regarding the improvement of lower bound, as well as the number of linear cuts that have to be accumulated in the relaxation, is also included.

References

1. Adjiman CS, Androulakis IP, Floudas CA (1998) A Global Optimization Method, αBB, for General Twice-Differentiable Constrained NLPs II Implementation and Computational Results. Comput Chem Eng 22:1159–1179
2. Adjiman CS, Dallwig S, Floudas CA, Neumaier A (1998) A Global Optimization Method, αBB, for General Twice-Differentiable Constrained NLPs I Theoretical Advances. Comput Chem Eng 22:1137–1158
3. Adjiman CS, Floudas CA (1996) Rigorous Convex Underestimators for General Twice-Differentiable Problems. J Global Optim 9:23–40
4. Akrotirianakis IG, Floudas CA (2004) A New Class of Improved Convex Underestimators for Twice Continuously Differentiable Constrained NLPs. J Global Optim 30:367–390
5. Akrotirianakis IG, Floudas CA (2004) Computational Experience with a New Class of Convex Underestimators : Box-Constrained NLP Problems. J Global Optim 29:249–264
6. Androulakis IP, Maranas CD, Floudas CA (1995) αBB: A Global Optimization Method for General Constrained Nonconvex Problems. J Global Optim 7:337–363
7. Floudas CA (2000) Deterministic Global Optimization: Theory, Algorithms and Applications. Kluwer
8. Floudas CA, Akrotirianakis IG, Caratzoulas S, Meyer CA, Kallrath J (2005) Global Optimization in the 21st Century: Advances and Challenges. Comput Chem Eng 29:1185–1202
9. Floudas CA, Kreinovich V (2007) Towards Optimal Techniques for Solving Global Optimization Problems: Symmetry-Based Approach. In: Torn A, Zilinskas J (eds) Models and Algorithms for Global Optimization. Springer, pp 21–42, ISBN 978-0-387-36720-0
10. Floudas CA, Kreinovich V (2007) On the Functional Form of Convex Underestimators for Twice Continuously Differentiable Functions. Optim Lett 1:187–192
11. Gounaris CE, Floudas CA (2008) Tight Convex Underestimators for C^2-Continuous Functions: I. Univariate Functions. J Global Optim, in press
12. Gounaris CE, Floudas CA (2008) Tight Convex Underestimators for C^2-Continuous Functions: II. Multivariate Functions. J Global Optim, in press
13. Hertz D, Adjiman CS, Floudas CA (1999) Two results on bounding the roots of interval polynomials. Comput Chemical Eng 23:1333–1339
14. Maranas CD, Floudas CA (1994) Global Minimum Potential Energy Conformations of Small Molecules. J Global Optim 4:135–170
15. Meyer CA, Floudas CA (2005) Convex Underestimation of Twice Continuously Differentiable Functions by Piecewise Quadratic Perturbation : Spline αBB Underestimators. J Global Optim 32:221–258

Global Optimization Using Space Filling

GOSF

ROMAN G. STRONGIN
Nizhni Novgorod State University,
Nizhni Novgorod, Russia

MSC2000: 90C26

Article Outline

Keywords

Constrained global optimization; Lipschitz optimization; Space filling curve; Peano curve; Index approach; Partial computability; E-reserved solution

A large number of decision problems in the world of applications may be formulated as searching for a *constrained global optimum* (minimum, for certainty)

$$\begin{aligned}\varphi^* &= \varphi(y^*)\\ &= \min\{\varphi(y)\colon\ y \in D,\ g_i(y) \le 0,\ 1 \le i \le m\}\,,\end{aligned}$$

where the *domain of search* (DS)

$$D = \left\{y \in \mathbf{R}^N\colon\ -2^{-1} \le y_j \le 2^{-1},\ 1 \le j \le N\right\},$$

$\mathbf{R}^N$ is the N-dimensional Euclidian space and the *objective function* $\varphi(y)$ (henceforth denoted $g_{m+1}(y)$) and the left-hand sides $g_i(y)$, $1 \le i \le m$, of the *constraints* are *Lipschitzian* (with respective constants L_i, $1 \le i \le m + 1$) and may be multi-extremal.

If DS is set defined by the hyperparallelepiped

$$S = \left\{w \in \mathbf{R}^N\colon\ a_j \le w_j \le b_j,\ 1 \le j \le N\right\},$$

then, by introducing the transformation

$$y_j = \frac{w_j - (a_j + b_j)/2}{\rho}, \qquad \rho = \max\left\{b_j - a_j\colon\ 1 \le j \le N\right\},$$

and the extra constraint

$$g_0(y) = \max\left\{|y_j| - \frac{b_j - a_j}{2\rho}\colon\ 1 \le j \le N\right\} \le 0,$$

it is possible to keep up the initial presentation D for DS (which is assumed to be the standard one) not altering the relations of Lipschitzian properties in dimensions.

The assumption of the divided functions g_i, $0 \le i \le m+1$, differences being bounded by the respective constants L_i (Lipschitzian property), which may be interpreted as a mathematical description of a limited power of change in real systems, provides a basis for estimating φ^* and y^*; by exploring DS with finite number of trials depending on the desired accuracy of search. This Lipschitzian approach ([2,5,9,20]) requires, in general, substantially less trials than the plain *uniform grid technique* owing to the thorough selection of each subsequent trial with the account of all the previously computed functions' values.

Such a selection turns into solving some auxiliary multidimensional optimization problem (MOP) of increasing multi-extremality (along with the accumulation of trial outcomes) at each step of the search process. But the case $N = 1$ is effectively solvable and, therefore, it is of interest to present MOP by its one-dimensional equivalent.

A possible way to do so ([1,7,11,12,14,15,18]) is to employ single-valued *Peano curves* $y(x)$ continuously mapping the unit interval [0, 1] on the x-axis onto the hypercube D and, thus, yielding the equality

$$\varphi^* = \varphi(x^*) = \min\left\{\varphi(y(x))\colon \begin{array}{c} x \in [0,1], \\ g_i(y(x)) \le 0, 0 \le i \le m \end{array}\right\}. \tag{1}$$

These curves, first introduced in [4,8], are 'filling' the cube, i. e. they pass through every point of D, and this gave rise to the term *space filling curves* (SFC); see survey [10].

The construction of SFC can be explained by following the scheme from [4]. Divide D into 2^N equal hypercubes of 'first-partition' by cutting D with the set of N mutually orthogonal hyperplanes (each plain is parallel to one of the coordinate ones and passes through the middle points of D edges orthogonal to this hyperplane). Then divide (in the above manner) each of the obtained first-partition cubes into 2^N second-partition cubes. Continuing this process, i. e. consequently cutting each cube of a current partition into 2^N cubes of the subsequent partition, yields hypercubes of any Mth partition with the edge-length equal 2^{-M}. The total number of cubes in the Mth partition is equal 2^{MN}.

Next, cut the interval [0, 1] into 2^N equal parts. Then, once again, cut each of these parts into 2^N smaller (equal) parts, etc. Designate $d(M, v)$ the subinterval of Mth partition, where v is the coordinate of the left endpoint of this interval. The length of $d(M, v)$ is equal 2^{-MN}. Assume that $v \in d(M, v)$, but the right endpoint of this subinterval (if it is not equal 1) does not belong to it.

Establish a mutually single-valued correspondence between all subintervals of any particular Mth partition and all subcubes of Mth partition. Henceforth, the notation $D(M, v)$ will stay for the subcube corresponding to the subinterval $d(M, v)$ and vice versa. Assume this correspondence to satisfy the following conditions:

- $D(M+1, v') \subset D(M, v'')$ if and only if $d(M+1, v') \subset d(M, v'')$.
- $d(M, v')$ and $d(M, v'')$ have a common endpoint (which is either v' or v'') if and only if $D(M, v')$ and $D(M, v'')$ have a common face (i. e. these subcubes are contiguous).

Now, a single-valued continuous map $y(x)$ is set by introducing the third requirement

- If $x \in d(M, v)$, then $y(x) \in D(M, v)$, for $M \ge 1$.

Note that for any integer $M \ge 1$ and any given $x \in [0, 1]$ there is just one subinterval meeting the condition $x \in d(M, v)$; the continuity is the consequence of the first two conditions.

Approximation of SFC

The center $y^c(x)$ of the subcube $D(M, v)$ containing $y(x)$ may be interpreted as an approximation to $y(x)$; the inequalities

$$\max\left\{\left|y_j^c(x) - y_j(x)\right|\colon\ 1 \le j \le N\right\} \le 2^{-(M+1)}, \qquad x \in [0,1],$$

reflect the accuracy attainable for any particular preset value of M.

A constructive way to establishing the above correspondence is described and substantiated in [3,12,15,19] and, in short, can be presented as follows. Introduce the auxiliary hypercube

$$\Delta = \left\{ y \in \mathbf{R}^N : \ -0.5 \le y_i \le 1.5,\ 1 \le i \le N \right\}$$

and designate $\Delta(s)$, $0 \le s \le 2^N - 1$, the subcubes of the first partition of Δ. The centers of $\Delta(s)$ (to be referred as $u(s)$) are N-dimensional binary vectors defined by the relations

$$\begin{aligned} &u_i(s) = (\beta_i + \beta_{i-1}) \mod 2, \\ &1 \le i < N, \quad u_N = \beta_{N-1}, \end{aligned} \tag{2}$$

where β_i, $0 \le i < N$, are digits in binary presentation of s:

$$s = \beta_{N-1} 2^{N-1} + \cdots + \beta_0 2^0. \tag{3}$$

Owing to this numeration, any two centers $u(s)$ and $u(s + 1)$, $0 \le s < 2^N - 1$, have just one different coordinate, which means that the corresponding subcubes $\Delta(s)$ and $\Delta(s + 1)$ are contiguous.

Next, let the binary form of v in $d(M, v)$ be

$$0 \le v = \sum_{i=1}^{MN} \alpha_i 2^{-i} < 1.$$

Then the identity $d(M, v) = d(z_1, \ldots, z_M)$, where

$$z_j = \sum_{i=1}^{N} \alpha_{(j-1)N+i} 2^i, \quad 1 \le j \le M, \tag{4}$$

provides the possibility to interpret $d(z_1, \ldots, z_M)$, as the z_Mth subinterval of the interval $d(z_1, \ldots, z_{M-1})$ divided into 2^N equal parts (the numeration streams from left to right along the x-axis). Note that the above identity implies $D(M, v) = D(z_1, \ldots, z_M)$.

Now, mapping Δ onto D by the linear transformation and assuming that $D(z_1) = D(s)$ if $D(s)$ is the image of $\Delta(s)$, we obtain the numeration (in the first partition of D) satisfying the above conditions. Then by mapping Δ onto each subcube $D(z_1)$ of the first partition, we get the desired numeration in the second partition of D, where $D(z_1, z_2) = D(z_1, s)$ if $D(z_1, s)$ is the image of $\Delta(s)$, and so on. To ensure that $D(z_1, 2^N - 1)$ and $D(z_1 + 1, 0)$ would also have a common face (and, in general, the last subcube in the first partition of $D(z_1, \ldots, z_M)$ and the first subcube in the first partition of $D(z_1, \ldots, z_M+1)$ would also be contiguous) we add some mechanism in the above numeration procedure to provide the necessary juxtapositioning.

Introduce the integer $l = l(z_1, \ldots, z_M)$ indicating the number of the only coordinate which has to be different for the center of the initial subcube $D(z_1, \ldots, z_M, 0)$ and the last subcube $D(z_1, \ldots, z_M, 2^N - 1)$ of the next partition of $D(z_1, \ldots, z_M)$ and the binary vector $w = w(z_1, \ldots, z_M)$ indicating the position of the center of the subcube $D(z_1, \ldots, z_M, 0)$. To do so we employ the integer function

$$l(s) = \begin{cases} 1 & \text{if } s = 0 \text{ or } s = 2^N - 1, \\ \min & \left\{ j\colon \ 2 \le j \le N,\ \beta_{j-1} = 1 \right\}, \\ & \text{otherwise}, \end{cases} \tag{5}$$

where β_{j-1} is from (3), and the binary vector-function

$$w_i(s+1) = w_i(s) = \begin{cases} \overline{u}_i(s), & i = 1, \\ u_i(s), & 2 \le i \le N, \end{cases} \tag{6}$$

where s is supposed to be the odd number, $\overline{u}_i$ stays for logical negation of u_i, and $w(0) = u(0)$. The amended procedure for successive numeration in subsequent partitions includes the operations:

- permutation of u_N and u_t in $u(z_j)$ from (2) and of w_N and w_t in $w(z_j)$ from (6) with $t = \mathrm{l}(z_{j-1})$, where z_{j-1} is from (4), $1 < j \le M$, and $l(z_{j-1})$ is from (5); $t = N$ if $j = 1$. New vectors are to be referred as $u^t(z_j)$ and $w^t(z_j)$;
- addition

  $$\begin{aligned} u_i^{tq}(z_j) &= (u_i^t(z_j) + q_i) \mod 2,\ 1 \le i \le N, \\ w_i^{tq}(z_j) &= (w_i^t(z_j) + q_i) \mod 2,\ 1 \le i \le N, \end{aligned}$$

  where $q = w(z_{j-1})$, $1 < j \le M$, and $q = (0, \ldots, 0) \in \mathbf{R}^N$ if $j = 1$;
- transformation

  $$l^t(z_j) = \begin{cases} N, & l(z_j) = t, \\ t, & l(z_j) = N, \\ l(z_j), & l(z_j) \ne N \text{ and } l(z_j) \ne t, \end{cases}$$

  where t is from the above permutation.

The successively computed values $u^{tq}(z_j)$, $w^{tq}(z_j)$, $l^t(z_j)$ are used instead of the initial values $u(z_j)$, $w(z_j)$, $l(z_j)$, $1 \leq j \leq M$, to obtain the approximation

$$y^c(x) = \sum_{j=1}^{M} (u^{tq}(z_j) - p)2^{-j}, \quad x \in d(M, \nu),$$

with $p = (2^{-1}, \ldots, 2^{-1}) \in \mathbf{R}^N$.

The important property of reducing dimensionality through SFC is that functions $g_i(y(x))$, $0 \leq i \leq m + 1$, from (1) corresponding to Lipschitzian functions from the initial MOP satisfy the *uniform Hölder conditions* ([7,11,15,19])

$$\left|g_i(y(x')) - g_i(y(x''))\right| \leq K_i(\left|x' - x''\right|)^{\frac{1}{N}}, \\ x', x'' \in [0, 1],$$

with respective coefficients $K_i = 4L_i\sqrt{N}$, $0 \leq i \leq m + 1$.

Problem (1) can further be reduced to an unconstrained case by employing the *index approach* (IA) ([7,13,17,18]) which makes no use of *penalties* and, thus, does not require any adjustments of penalty coefficients. Within IA functions $g_i(y(x))$ from (1) may not be defined throughout [0, 1]; they have to be computable only at the points $x \in [0, 1]$ meeting the conditions $g_k(y(x)) \leq 0$, $1 \leq k < i$ (this property is to be referred as *partial computability* of problem functionals). Therefore, within IA the *outcome* of each trial is given by a dyad

$$f(x) = g_\nu(y(x)), \quad \nu = \nu(x) = \nu(y(x)), \tag{7}$$

where ν is the number of the first constraint violated at the point x; this number is to be referred as the *index* of the corresponding point.

The unconstrained equivalent of (1) is

$$\psi(x^*) = \min\{\psi(x) \colon x \in [0, 1]\},$$

where

$$\psi(x) = \frac{g_\nu(y(x))}{K_\nu} - \begin{cases} 0, & \nu = \nu(x) \leq m, \\ \frac{\varphi^*}{K_\nu}, & \nu = \nu(x) = m + 1, \end{cases}$$

and x^* is a solution to (1). The algorithm presented below solves (1) by minimizing $\psi(x)$. It substitutes the unknown values φ^* and K_i, $0 \leq i \leq m{+}1$, by their running estimates; it also surmounts the discontinuity inherent to $\psi(x)$.

Algorithm

The first trial is to be executed at an arbitrary interior point $x^1 \in (0, 1)$. The choice of any subsequent point x^{k+1}, $k \geq 1$, is due to the rules:

1) Renumber the points $x^1, \ldots, x^k$ of the previous trials by subscripts in the increasing order of the coordinate, i. e.

$$0 = x_0 < \cdots < x_k < x_{k+1} = 1,$$

and associate them with the computed values $z_i = f(x_i)$, $1 \leq i \leq k$, from (7); values z_0 and z_{k+1} are undefined.

2) Collect in the sets

$$I_\nu = \{i \colon 1 \leq i \leq k,\ \nu = \nu(x_i)\}, \\ 0 \leq \nu \leq m + 1,$$

all subscripts corresponding to the points with equal indices; it is assumed that $\nu(x_0) = \nu(x_{k+1}) = -1$ and $I_{-1} = \{0, k + 1\}$.

3) Construct the unions

$$S_\nu = I_{-1} \cup \cdots \cup I_{\nu-1}, \quad 0 \leq \nu \leq m + 1,$$

and

$$T_\nu = I_{\nu+1} \cup \cdots \cup I_{m+1} \cup I_{m+2}, \\ 0 \leq \nu \leq m + 1,$$

of subscripts corresponding to the trial points with the indices less than ν and exceeding ν respectively; $I_{m+2} = \emptyset$ by the definition.

4) Compute the running lower bounds

$$\mu_\nu = \max \left\{ \frac{\left|z_j - z_i\right|}{(x_j - x_i)^{\frac{1}{N}}} \colon \begin{matrix} i, j \in I_\nu, \\ i < j \end{matrix} \right\} \tag{8}$$

for respective Hölder coefficients of the functions $g_\nu(y(x))$, $0 \leq \nu \leq m + 1$. If I_ν contains less than two elements or if μ_ν from (8) is equal zero, assume that $\mu_\nu = 1$.

5) Find the values

$$z_\nu^* = \begin{cases} -\varepsilon_\nu, & T_\nu \neq \emptyset, \\ \min\{z_i \colon i \in I_\nu\}, & T_\nu = \emptyset, \end{cases}$$

for all nonempty sets I_ν, $0 \leq \nu \leq \mathrm{m} + 1$; vector $\varepsilon = (\varepsilon_0, \ldots, \varepsilon_m)$ is the input of the algorithm.

6) Compute characteristics $R(i)$, $1 \leq i \leq k+1$, where

$$R(i) = \Delta_i + \frac{(z_i - z_{i-1})^2}{r^2 \mu_\nu^2 \Delta_i} - \frac{2(z_i + z_{i-1} - 2z_\nu^*)}{r\mu_\nu},$$
$$i - 1, \quad i \in I_\nu,$$
$$R(i) = 2\Delta_i - \frac{4(z_i - z_\nu^*)}{r\mu_\nu},$$
$$i \in I_\nu, \quad i - 1 \in S_\nu,$$
$$R(i) = 2\Delta_i - \frac{4(z_{i-1} - z_\nu^*)}{r\mu_\nu},$$
$$i - 1 \in I_\nu, \quad i \in S_\nu,$$
$$\Delta_i = (x_i - x_{i-1})^{\frac{1}{N}}.$$

Proper choice of the parameter $r > 1$ allows to use the product $r\mu_\nu$ as an upper bound for K_ν.

7) Select integer t from

$$R(t) = \max\{R(i)\colon\ 1 \leq i \leq k+1\}$$

and execute the subsequent trial at the point

$$x^{k+1} = \frac{x_t + x_{t-1}}{2} - \operatorname{sign}(z_t - z_{t-1}) \left[\frac{|z_t - z_{t-1}|}{\mu_\nu}\right]^N \cdot \frac{1}{2r}$$

if $\nu(x_t) = \nu(x_{t-1})$; otherwise, i. e. if $\nu(x_{t-1}) \neq \nu(x_t)$, the second term is omitted.

The concept of *ε-reserved solution $y\varepsilon$*, where

$$\varphi(y_\varepsilon) = \min\left\{\varphi(y)\colon \begin{array}{c} y \in D, \\ g_i(y) \leq -\varepsilon_i, \\ 0 \leq i \leq m \end{array}\right\}$$

and $\varepsilon_i > 0$, $0 \leq i \leq m$, provides interpretation for ε from Step 5). The sequence of points $\{x^k\}$ selected by the Steps 1)–7) in the interval [0, 1] generates the corresponding sequence $\{y^k\} = \{y(x^k)\}$ in D.

Convergence Conditions

([15,16,17] [18]). Assume that the following is true:

- the problem (1) has an ε-reserved solution;
- functions $g_i(y)$, $0 \leq i \leq m+1$, admit Lipschitzian continuations throughout D;
- from some Step onwards, the values μ_ν, $0 \leq \nu \leq m + 1$, from (8) satisfy the inequalities

$$r\mu_\nu > 16 L_\nu \sqrt{N}, \quad 0 \leq \nu \leq m+1.$$

Then any limit point $\overline{y}$ of the sequence $\{y^k\}$ generated by the above algorithm satisfies the conditions:

$$\varphi(\overline{y}) = \inf\left\{\varphi(y^k)\colon \begin{array}{c} k \in N_1, \\ g_i(y^k) \leq 0, \\ 0 \leq i \leq m \end{array}\right\} \leq \varphi(y_\varepsilon),$$

where N_1 is the set of positive integers.

As long as in applications SFC $y(x)$ is to be approximated by $y^c(x)$ corresponding to some Mth partition, it is important to notice that the substantiation of the above convergence conditions implies the relation

$$2^{-M} \ll \frac{1}{\sqrt{N}} \min_{0 \leq \nu \leq m} \left(\frac{\varepsilon_\nu}{L_\nu}\right),$$

which means that the existence of an ε-reserved solution may be interpreted as some kind of the *regularity conditions* (cf. [6]).

Dimensionality reduction through SFC causes some loss of the information on the closeness of trial points is the initial multidimensional space. Two close points in D may have substantially nonclose pre-images in [0, 1]. To overcome this obstacle, it is possible either to store all pre-images of each trial point (close points in D always have some close pre-images; see [12]) or to use some sets of shifted SFC to provide the better transfer of metric information (see [17]).

GOSF based on the reduction to one dimension by using SFC and on the reduction to unconstrained problems by employing IA admits effective parallelization (see [16,19]).

See also

- ▶ αBB Algorithm
- ▶ Continuous Global Optimization: Applications
- ▶ Continuous Global Optimization: Models, Algorithms and Software
- ▶ Differential Equations and Global Optimization
- ▶ DIRECT Global Optimization Algorithm
- ▶ Global Optimization Based on Statistical Models
- ▶ Global Optimization in Binary Star Astronomy
- ▶ Global Optimization Methods for Systems of Nonlinear Equations
- ▶ Topology of Global Optimization

References

1. Butz AR (1968) Space filling curves and mathematical programming. Inform Control 12(4):313–330
2. Evtushenko YuG (1985) Numerical optimization techniques. Transl Ser Math and Engin Optim. Software, New York
3. Gergel VP, Strongin LG, Strongin RG (1988) Neighbourhood method in recognition problems. Soviet J Comput Syst Sci 26(2):46–54
4. Hilbert D (1891) Über die steitige Abbildung einer Linie auf ein Flächenstück. Math Ann 38:459–460
5. Horst R, Pardalos PM (1995) Handbook of global optimization. Kluwer, Dordrecht
6. Kuhn HW, Tucker AW (1951) Nonlinear programming. Proc. Second Berkeley Symp. Math. Statistics and Probability, Univ. Calif. Press, Berkeley, pp 481–492
7. Markin DL, Strongin RG (1988) A method for solving multiextremal problems with non-convex constraints, that uses a priori information about estimates of the optimum. USSR J Comput Math Math Phys 27(1):33–39
8. Peano G (1890) Sur une courbe, qui remplit toute une aire plane. Math Ann 36:157–160
9. Pinter J (1996) Global optimization in action (continuous and Lipschitz optimization: Algorithms, implementations and applications). Kluwer, Dordrecht
10. Sagan H (1994) Space-filling curves. Springer, Berlin
11. Strongin RG (1973) On the convergence of an algorithm for finding a global extremum. Eng Cybernetics 11(4):549–555
12. Strongin RG Numerical methods in multi-extremal problems (Information-statistical algorithms), Nauka, Moscow. (In Russian)
13. Strongin RG (1985) Numerical methods for multiextremal nonlinear programming problems with nonconvex constraints. Lecture Notes Economics and Math Systems, 255:278–282
14. Strongin RG (1989) The information approach to multiextremal optimization problems. Stochastics and Stochastic Reports 27:65–82
15. Strongin RG (1990) Search for global optimum. Znanie, Moscow
16. Strongin RG (1991) Parallel multi-extremal optimization using a set of evolvents. USSR J Comput Math Math Phys 31(8):1173–1185. (In Russian)
17. Strongin RG (1992) Algorithms for multi-extremal mathematical programming problems employing the set of joint space-filling curves. J Global Optim 2:357–378
18. Strongin RG, Markin DL (1986) Minimization of multiextremal functions with nonconvex constraints. Cybernetics 22(4):486–493
19. Strongin RG, Sergeyev YAD (1992) Global multidimensional optimization on parallel computer. Parallel Comput 18:1259–1273
20. Sukharev AG (1989) Minimax algorithms in problems of numerical analysis. Nauka, Moscow

Global Optimization in Weber's Problem with Attraction and Repulsion

COSTAS D. MARANAS
Pennsylvania State University, University Park, USA

MSC2000: 90C26, 90C90

Article Outline

Keywords

Weber problem; Facility location; Global optimization

Weber's problem and all its variations with positive weights is clearly one of the most extensively studied problems in the area of continuous location theory. It frequently arises in planning situations where a single central facility must be located so as to minimize the total cost associated with serving a number of demand centers. In all these cases, the underlying assumption, that the associated service costs are directly proportional to the Euclidean distance of the demand center from the central facility, has been adopted.

Weber's problem with attraction and repulsion can be stated as follows: Given a number of 'attractive' or 'repulsive' points located on a $2D$-plane, find the position of a single facility inside an arbitrary region P such that the sum of the weighted distances of all points from the single facility is at its global minimum.

This problem can be formulated as the following nonlinear optimization problem:

$$\min_{(x,y)\in P} \sum_{i\in I^+} w_i \sqrt{(x-x_i)^2+(y-y_i)^2} - \sum_{i\in I^-} w_i \sqrt{(x-x_i)^2+(y-y_i)^2},$$

where I^+, I^- are the sets of attractive (users) and repulsive (residents) points, respectively; w_i, $i \in I^+$ the positive weight of the ith attractive point and $-w_i$, $i \in I^-$ the negative weight of the ith repulsive point; (x_i, y_i) are the coordinates of the ith attractive or repulsive point;

and P is the region where where the single facility must be situated.

The unconstrained version of this problem has been shown to involve a number of important properties. The first property provides a sufficient condition for having finite solutions or solutions at infinity. Z. Drezner and G.O. Wesolowsky [6] by using the well-known triangle inequality relation proved the following:

Property 1 For the unconstrained problem, if $W > 0$ then the global optimum location is finite; if $W < 0$ then the global optimum location is at infinity, where

$$W = \sum_{i \in I^+} w_i - \sum_{i \in I^-} w_i .$$

The second property deals with the localization of all local minimum solutions. Let R be the radius of the smallest circle enclosing all points. The square of this radius R can be obtained through the solution of the following nonlinear optimization problem:

$$\begin{cases} \min\limits_{x^c, y^c, R^2} & R^2 \\ \text{s.t.} & (x^c - x_i)^2 + (y^c - y_i)^2 \le R^2, \\ & \forall i \in I^+ \cup I^-, \end{cases}$$

which is convex in the combined space of the coordinates of the center of the circle (x^c, y^c) and the square of the radius of the circle R^2 enclosing all points. Drezner and Wesolowsky [6] proved the following localization property, which generalizes the *majority theorem* [24] for Weber's problem.

Property 2 For the unconstrained problem, all local minima and therefore the global minimum are inside a disc with a radius equal to

$$\rho = \frac{R}{\sqrt{1-\alpha^2}}$$

where

$$\alpha = \frac{W^-}{W^+},$$
$$W^+ = \sum_{i \in I^+} w_i , \quad W^- = \sum_{i \in I^-} w_i .$$

Note that the boundary of this disc is attainable. The case $\alpha = 1$ or, equivalently, $W = 0$ is accounted for by finding the optimal solution at infinity and comparing it with the best finite solution. Drezner and Wesolowsky [6] by using asymptotic analysis showed the following:

Property 3 For the unconstrained problem, if $W = 0$ the best solution at infinity is $-(A^2 + B^2)^{1/2}$ where:

$$A = \sum_{i \in I^+} w_i x_i - \sum_{i \in I^-} w_i x_i ,$$
$$B = \sum_{i \in I^+} w_i y_i - \sum_{i \in I^-} w_i y_i .$$

The following property examines whether a demand point corresponds to a local minimum [6].

Property 4 For the unconstrained problem, if there is a point i such that

$$w_i - (W_x + W_y)^{1/2} \begin{cases} > 0, & \text{then point } i \text{ is a local minimum,} \\ < 0, & \text{then point } i \text{ is not a local minimum,} \\ = 0, & \text{then both possibilities are open,} \end{cases}$$

where

$$W_x = \sum_{i,j \in I^+, i \neq j} \frac{w_i (x_i - x_j)}{\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}} - \sum_{i,j \in I^-, i \neq j} \frac{w_i (x_i - x_j)}{\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}},$$
$$W_y = \sum_{i,j \in I^+, i \neq j} \frac{w_i (y_i - y_j)}{\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}} - \sum_{i,j \in I^-, i \neq j} \frac{w_i (y_i - y_j)}{\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}}.$$

P.-C. Chen and others [5] and F. Plastria [14] derived independently the following sufficient condition for a demand point to be the global minimum solution.

Property 5 For the unconstrained problem, if there is a point $i^* \in I^+$ such that

$$w_{i^*} \ge \sum_{i \in I^+ \cup I^-, i \neq i^*} w_i ,$$

then (x_{i^*}, y_{i^*}) is the global optimum location.

It is quite straightforward to show that if all weights are positive then the expression for the weighted sum of the Euclidean distances is convex [13] and therefore the single local minimum corresponds to the global minimum. This means that the total expression for the sum of weighted Euclidean distances is a difference of two convex functions. As it has been noted earlier the presence of negative weights greatly complicates the location of the global minimum solution by introducing concave contributions in the objective function. This special class of difference of two convex functions (DC) optimization problems has recently (1990) received considerable attention [7]. The next theorem introduces a set of conditions for convexity of $F(x, y)$ at some point (x, y).

Property 6 $F(x, y)$ is convex at (x, y) if

$$\sum_{i \in I^+ \cup I^-} \frac{W_i}{r_i} \geq 0,$$

and

$$\sum_{i \in I^+ \cup I^-} \sum_{\substack{j \in I^+ \cup I^-, \\ j > i}} \frac{W_i W_j}{r_i^3 r_j^3} \times \left[(x - x_i)(y - y_j) + (x - x_j)(y - x_i)\right]^2 \geq 0,$$

where $r_i = \sqrt{(x - x_i)^2 + (y - y_i)^2}$, $i \in I^+ \cup I^-$ and

$$W_i = \begin{cases} w_i, & i \in I^+, \\ -w_i, & i \in I^-. \end{cases}$$

A proof of this property can be found in [11].

A special case of this problem, involving three points with weights equal to one, was first posed by P. Fermat in the seventeenth century and it was solved geometrically by E. Toricelli. E. Weiszfeld [23] first proposed a simple iterative algorithm but with no convergence proof. Later, H.W. Kuhn [8,9,10] proved that Weiszfeld's algorithm was convergent assuming no iterate coincided with any of the demand points. L.M. Ostresh [12] and E. Balas and others [1] proposed modifications of the Weiszfeld algorithm where by perturbing the current point, if it coincided with a demand point, was global convergence guaranteed. C.Y. Wang [22] proved that Weiszfeld's algorithm has linear rate of convergence under certain conditions and sublinear otherwise. More recently (1980s), P.H. Calamai and A.R. Conn [2,3,4] and M.L. Overton [13] introduced second order methods which involved local quadratic convergence and global convergence under conditions. G.L. Xue [25,26], and Xue and J.B. Rosen [15] proved unconditional global convergence and conditional local quadratic convergence for a second order algorithm and computational comparisons were carried out between Weiszfeld's algorithm and Newton's algorithm on a parallel machine.

Most papers address only positive weights reflecting the inherent assumption that all points 'attract' the central facility. However, in real world there exists an abundance of example problems where certain points 'repel' the central facility. For example nuclear plants, sewage treatment plants, or polluting industrial units may be desired to be as close as possible to their customers so that transportation costs are minimized but at the same time environmental considerations require that these facilities be as far as possible from residential areas and fragile ecological systems. This need to locate a facility away from certain points can be quantified through the use of negative weights as shown in [16,19]. A negative weight means that the value of the objective function is increased as the facility approaches the corresponding point. Therefore, the global optimum location of a facility is now the one that balances the repulsion and the attraction acting on the central facility. It is interesting to note that the introduction of negative weights greatly increases the complexity of the problem.

Weber's problem with some negative weights was first considered by L.-N. Tellier [17], who studied the case of two attractive and one repulsive point. Later, Tellier and D. Pollanski [18] analyzed exhaustively all different cases involving three demand points and derived statistical conclusions regarding the types of possible solutions. Drezner and Wesolowsky [6] proved a number of theoretical results and proposed a heuristic algorithm for locating the global minimum solution. However, it was Chen and others [5] who first presented an exact outer approximation algorithm for Weber's problem with attraction and repulsion by exploiting the d.c. structure of the problem. In addition, they [5] extend their procedure to exponentially decaying repulsion and facility location within a set of disjoint convex polygons. Later, Maranas and Floudas [11] proposed a branch and bound type global optimization algorithm for solving Weber's problems with attraction and repulsion. The approach was based on the iterative solution of a set of convex and concave lower bounding problems. Convergence to an ϵ-global minimum was proven and examples were solved with as many as 10,000 points. By analyzing the computational results they observed that for any given number of points N the difficulty of the problem increases as we introduce more repulsive points. This trend continues until about equal numbers of attractive and repulsive points are reached. Then, a sharp decrease in computational requirements is observed as more repulsive points are added. In fact, it is easier to solve problems involving more repulsive points than attractive ones. The standard deviation of the total number of required iterations and function evaluations is fairly small for all ratios of attractive to repulsive points with the sole exception of the $N^{+} = N^{-} = N/2$ case where the standard deviation is substantially increased. For a given ratio of attractive to repulsive points the CPU requirements increase almost linearly with N reflecting the fact that most of CPU time is spent on function evaluations.

A generalization of Weber's problem is the maximization of the sum of decreasing convex functions of arbitrary metrics. H. Tuy and F.A. Al-Khayyal [20] proposed the first algorithm for finding global solutions to the problem by reducing it to a sequence of unconstrained nondifferentiable convex minimization problems. Later, they [21] extended this work to account for repulsion as well and proposed a d.c. reformulation of the problem which enabled them to develop a global optimization procedure.

See also

- ► Combinatorial Optimization Algorithms in Resource Allocation Problems
- ► Competitive Facility Location
- ► Facility Location with Externalities
- ► Facility Location Problems with Spatial Interaction
- ► Facility Location with Staircase Costs
- ► MINLP: Application in Facility Location-allocation
- ► Multifacility and Restricted Location Problems
- ► Network Location: Covering Problems
- ► Optimizing Facility Location with Rectilinear Distances
- ► Production-distribution System Design Problem
- ► Resource Allocation for Epidemic Control
- ► Single Facility Location: Circle Covering Problem
- ► Single Facility Location: Multi-objective Euclidean Distance Location
- ► Single Facility Location: Multi-objective Rectilinear Distance Location
- ► Stochastic Transportation and Location Problems
- ► Voronoi Diagrams in Facility Location
- ► Warehouse Location Problem

References

1. Balas E, Yu CS (1982) A note on the Weiszfeld–Kuhn algorithm for the general Fermat problem. Managem Sci Res Report 484:1–6
2. Calamai PH, Conn AR (1980) A stable algorithm for solving the multifacility location problem involving Euclidean distances. SIAM J Sci Statist Comput 1:512–526
3. Calamai PH, Conn AR (1982) A second-order method for solving the continuous multifacility location problem. In: Numerical Analysis. Proc. 9th Biennial Conf. Dundee, Scotland, pp 1–25
4. Calamai PH, Conn AR (1987) A projected Newton method for lp norm location problems. Math Program 38:75–109
5. Chen P-C, Hansen P, Jaumard B, Tuy H (1992) Weber's problem with attraction and repulsion. J Reg Sci
6. Drezner Z, Wesolowsky GO (1991) The Weber problem on the plane with some negative weights. INFOR 29:87–99
7. Horst R, Tuy H (1990) Global optimization, deterministic approaches. Springer, Berlin
8. Kuhn HW (1967) On a pair of dual nonlinear programs. Nonlinear Programming. North-Holland, Amsterdam, pp 38–54
9. Kuhn HW (1973) A note on Fermat's problem. Math Program 4:94–107
10. Kuhn HW (1974) Steiner's problem revisited. In: Studies in Optimization. Math. Assoc. America, Washington, DC, pp 52–70
11. Maranas CD, Floudas CA (1993) A global optimization method for Weber's problem with attraction and repulsion. In: Proc. Large Scale Optimization: State of the Art Conf., Florida Univ., 15-17 Feb. 1993. Kluwer, Dordrecht, pp 259–293
12. Ostresh LM (1978) On the convergence of a class of iterative methods for solving the Weber location problem. Oper Res 26:597–609
13. Overton ML (1983) A quadratically convergent method for minimizing a sum of Euclidean norms. Math Program 27:34–63
14. Plastria F (1992) The effects of majority in Fermat–Weber problems with attraction and repulsion. YUGOR 1
15. Rosen JB, Xue G-L (1991) Computational comparison of two algorithms for the Euclidean single facility location problem. ORSA J Comput 3:207–212

16. Tellier L-N (1972) The Weber problem: Solution and interpretation. Geographical Anal 4:215–233
17. Tellier L-N (1985) Économie patiale: rationalitée économique de l'espace habité. Gaétan Morin, Chicoutimi, Québec
18. Tellier L-N (1989) The Weber problem: frequency of different solution types and extension to repulsive forces and dynamic processes. J Reg Sci 29:387–405
19. Tellier L-N, Ceccaldi X (1983) Phenomenes de polarization et de repulsion dans le context du probleme de Weber. Canad Regional Sci Assoc
20. Tuy H, Al-Khayyal FA (1992) Global optimization of a nonconvex single facility problem by sequential unconstrained convex minimization. J Global Optim 2:61–71
21. Tuy H, Al-Khayyal FA, Zhou F (1995) A D.C. optimization method for single facility location problems. J Global Optim 2:61–71
22. Wang CY (1975) On the convergence and rate of convergence of an iterative algorithm for the plant location problem. Qufu Shiyun Xuebao 2:14–25
23. Weiszfeld E (1937) Sur le point pour lequel la somme des distances de n points donnés est minimum. Tôhoku Math J 43:355–386
24. Witzgall C (1984) Optimal location of a single facility: Mathematical models and concepts. Report Nat Bureau Standards 8388
25. Xue G-L (1987) A fast convergent algorithm for $\min \sum_{i=1}^{m} \| x - a_i \|$ on a closed convex set. J Qufu Normal Univ 13(3):15–20
26. Xue G-L (1989) A globally and quadratically convergent algorithm for $\min \sum_{i=1}^{m} \| x - a_i \|$ type plant location problem. Acta Math Applic Sinica 12:65–72

Global Pairwise Protein Sequence Alignment via Mixed-Integer Linear Optimization

S. R. McAllister, R. Rajgaria,
Christodoulos A. Floudas
Department of Chemical Engineering,
Princeton University, Princeton, USA

MSC2000: 90C10, 92-08

Article Outline

Abstract

A well-studied problem in the area of computational biology is the sequence alignment problem. Three mixed-integer linear optimization models have been developed to address the global pairwise sequence alignment problem in a mathematically rigorous fashion. These formulations, in addition to their rigor, allow for (a) the natural introduction of functionally important conservation constraints, (b) the creation of a rank-ordered list of the highest scoring alignments and (c) the refinement of alignments by using pairwise interaction scores from simplified force fields. The third model, a path selection approach, employs some of the algorithmic advantages of dynamic programming methods, to outperform other optimization models.

Keywords and Phrases

Sequence alignment; Integer linear optimization; Global pairwise alignment; Rank-order list of alignments

Introduction

Sequence alignment methods aim to both identify related protein sequences and determine the best alignment between them. This approach provides a rough measure of evolutionary distance and may indicate possible relationships between the protein structure and function of similar sequences. Multiple scoring matrices have been developed based on the techniques of the percent of accepted mutations (PAM) [3] and protein blocks (BLOSUM) [5] to quantify this evolutionary distance between aligned residues.

The pairwise sequence alignment problem is most commonly addressed through either (i) global alignment or (ii) local alignment techniques. The goal of global alignment algorithms is to determine the highest scoring overall alignment spanning the length of both sequences. One widely used approach for this problem is a dynamic programming approach proposed by Needleman and Wunsch [10].

Proteins may share sequence similarity in some regions, but not in others. Local alignment algorithms are more suited to these problems and strive to align only the highest scoring subsequence match. Smith and Waterman extended the dynamic programming approach for global pairwise sequence alignment problems to address the local alignment problem [14]. Dynamic programming approaches are computationally inadequate for large scale database searches, so a number of heuristic algorithms for local pairwise sequence alignment have been proposed [1,2,11,12].

Several researchers have studied the effect of including information about near-optimal alignments. The investigation of the suboptimal paths and scores allows for an evaluation of the reliability of portions of a sequence alignment. A review of several approaches to this problem and their impact can be found elsewhere [17].

In some cases, an alignment between two sequences can be improved by constraining the problem to include biologically important information in the overall alignment. One example of this is the required conservation of certain residues that form a motif necessary for function. This problem has been addressed recently by dynamic programming algorithms [4,15].

Models

Several integer linear optimization (ILP) models have been developed to rigorously and completely address the problem of global pairwise sequence alignment in a general fashion. A comparison of the three approaches, a template-based model, a template-free model, and a path selection model are presented in the following sections. The formulation of the problem as an integer linear optimization problem provides a deterministic guarantee of identifying the global maximum alignment [6], allows for the introduction of integer cut constraints, provides a framework for the introduction of functionally-specific constraints, and shows promise for the optimal identification of pairwise interactions.

Template-Based Model

Consider two protein sequences $S1, S2$ of lengths M and N respectively, where $M > N$. Let the index i represent each position in Sequence S1 and the index j represent each position in S2, as shown in Eqs. 1–2.

$$i \in 1, 2 \ldots M \tag{1}$$

$$j \in 1, 2 \ldots N \tag{2}$$

The template-based optimization model assigns each amino acid of both sequences to a template to generate the optimal alignment. Equation 3 defines a template length K as the sum of the length of the larger sequence and the parameter N_GAPS_m, representing the maximum number of allowed gaps. This model requires the introduction of an index k, representing the position in the template, as defined by Eq. 4.

$$K = M + N\_GAPS_m \tag{3}$$

$$k \in 1, 2 \ldots K \tag{4}$$

The assignment of an amino acid to a template position requires the definition of the binary variables, y_{ik} and z_{jk}, as shown in Eqs. 5–6.

$$y_{ik} = \begin{cases} 1 & \text{if amino acid } i \text{ of S1 is assigned to} \\ & \text{template position} k \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

$$z_{jk} = \begin{cases} 1 & \text{if amino acid } j \text{ of S2 is assigned to} \\ & \text{template position } k \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

A position in the template may not have an amino acid assigned to it in the overall alignment. Therefore, Eqs. 7–8 introduce additional binary variables to represent these alignment gaps.

$$yg_k = \begin{cases} 1 & \text{if template position } k \text{ is a gap} \\ & \text{for Sequence S1} \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

$$zg_k = \begin{cases} 1 & \text{if template position } k \text{ is a gap} \\ & \text{for Sequence S2} \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

The objective function of this optimization model maximizes the alignment score, which is the sum of a scoring matrix value for each matching amino acid pair minus any associated penalties for gaps inserted in the sequence. The scoring matrix will assign a weight, w_{ij}, to any template position k that contains the amino acid in position i of Sequence S1 and also the amino acid in position j of S2.

For an affine gap penalty model with no penalties for gaps that begin or end a sequence, the objective function can then be posed as shown in Eq. 9. The contribution of the scoring matrix at positions i, j, w_{ij}, is considered only if position i of Sequence 1 is assigned to position k of the template, y_{ik}, and if position j of Sequence 2 is assigned to position k of the template, z_{jk}. The gap opening existence terms of Sequence 1, go_k^{S1}, and Sequence 2, go_k^{S2}, are weighted by the gap opening penalty value of wo to assess the penalty for the first residue of any gap in a sequence. The existence of a gap extension variable of either sequence, gl_k^{S1} and gl_k^{S2}, produces a penalty of wl for each occurrence. The active gl_k^{S1} and gl_k^{S2} variables that are contained within a beginning or an ending gap are counteracted by the product of gb_k^{S1}, gb_k^{S2}, ge_k^{S1}, or ge_k^{S2} with wl. Gap opening penalties at the beginning or end of a sequence are explicitly omitted through only summing over the reduced index, such that $2 \leq k \leq K-1$.

$$\max \sum_i \sum_j \sum_k w_{ij} \cdot y_{ik} \cdot z_{jk} - \sum_{k=2}^{K-1} (go_k^{S1} + go_k^{S2}) \cdot wo - \sum_{k=2}^{K-1} \left[(gl_k^{S1} - gb_k^{S1} - ge_k^{S1}) + (gl_k^{S2} - gb_k^{S2} - ge_k^{S2}) \right] \cdot wl \tag{9}$$

The objective function of Eq. 9 requires the linearization of the product of two binary variables and is subject to numerous constraints. The details of the model are available elsewhere [8].

Template-Free Model

Unlike the previously described mixed-integer linear programming formulation of the global pairwise sequence alignment problem in Sect. "Template-Based Model", the optimization model presented here does not assign the amino acids of each sequence to a template. However, information about the maximum number of allowable gaps is still included in this model, through the variable K in Eq. 3.

In the template-free model, a binary variable, z_{ij}, is defined in Eq. 10 to represent the alignment of position i in $S1$ to position j in $S2$. A method to handle gaps in the sequence still must be introduced into the model to account for the evolutionary changes that lead to residue insertions and deletions. Aligning a gap residue to another gap residue is not allowed. This observation leads to two possibilities of gap occurrences. A gap can either be in Sequence 1, across from a residue j in Sequence 2 or in Sequence 2, across from a residue i in Sequence 1. These possibilities are modeled with the binary variables zg_i and yg_j, defined by Eq. 11–12.

$$z_{ij} = \begin{cases} 1 & \text{if position } j \text{ in } S2 \text{ aligns with position } i \text{ in } S1 \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

$$zg_i = \begin{cases} 1 & \text{if no position } j \text{ in } S2 \text{ aligns to the residue in position } i \text{ of } S1 \\ 0 & \text{otherwise} \end{cases} \tag{11}$$

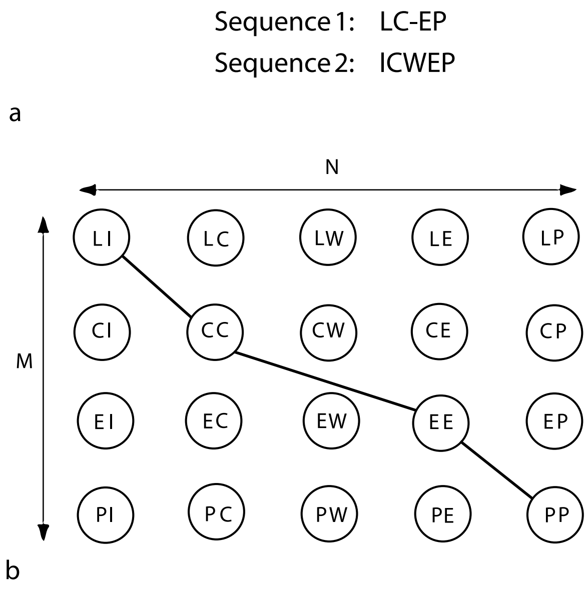
$$yg_j = \begin{cases} 1 & \text{if no position } i \text{ in } S1 \text{ aligns to the residue in position } j \text{ of } S2 \\ 0 & \text{otherwise} \end{cases} \tag{12}$$

The objective function in Eq. 13 maximizes the sum of the weights of the residue-residue alignments minus the sum of the gap penalties, plus the appropriate terms that remove the penalties from the gaps at the beginning and ends of the sequences. The scoring matrix values at any given pair of positions, w_{ij} are included when the binary variables indicating a sequence alignment that matches positions i and j, z_{ij}, are activated. For an affine gap penalty model, the variables representing the existence of a gap opening, go_j^{S1} and go_i^{S2}, and the existence of a gap extension, gl_j^{S1} and gl_i^{S2}, are multiplied by their respective weights, wo and wl. If a gap residue is present at the beginning or ending of a sequence, it will be accounted for in an active value for one of gb_j^{S1}, ge_j^{S1}, gb_i^{S2}, ge_i^{S2} to remove the penalty assigned by the previous terms.

$$\max \sum_{ij} w_{ij} z_{ij} - \sum_j (wo \cdot go_j^{S1} + wl \cdot gl_j^{S1})$$

$$
\begin{aligned}
&- \sum_{i} (wo \cdot go_i^{S2} + wl \cdot gl_i^{S2}) \\
&+ \sum_{j>1}^{N-1} wl \cdot (gb_j^{S1} + ge_j^{S1}) \\
&+ \sum_{i>1}^{M-1} wl \cdot (gb_i^{S2} + ge_i^{S2}) \\
&+ wo \cdot (ge_1^{S1} + ge_1^{S2}) \\
&+ wo \cdot (ge_M^{S1} + ge_N^{S2})
\end{aligned}
\tag{13}
$$

The objective function of Eq. 13 is subject to numerous constraints. The details of these constraints are available elsewhere [9].



Global Pairwise Protein Sequence Alignment via Mixed-Integer Linear Optimization, Figure 1
(a) Alignment of two hypothetical sequence fragments. (b) A node and path representation of the alignment problem as formulated by the mathematical model. Note the three active paths connecting the four selected node variables

Path Selection Model

Let us introduce a binary variable N_{ij} that represents the alignment of the residue at position i in Sequence 1 to the residue at position j in Sequence 2. This binary variables performs a similar role as z_{ij} in Sect. "Template-Free Model". The typical assignment of this match assesses a weight, w_{ij}, based on a scoring matrix developed through evolutionary analysis of protein sequences.

A successful sequence alignment will have many active N_{ij} variables, which we will designate as nodes. Let the binary variable $y_{ii'jj'}$ represent the existence of a connecting path between node N_{ij} and a neighboring node $N_{i'j'}$. Associated with this connecting path, is a weight parameter, $C_{ii'jj'}$, which can be calculated in advance from the scoring matrix w and any position dependent gap penalty form that is specified a priori. An example of the representation of the node and path variables is illustrated in Fig. 1.

Once these variables have been defined, the objective function of the optimal sequence alignment is merely the sum of the product of the variable for the existence of the path, $y_{ii'jj'}$, and the path weight, $C_{ii'jj'}$ as shown in Eq. 14.

$$
\max \sum_{i} \sum_{i'>i} \sum_{j} \sum_{j'>j} y_{ii'jj'} \cdot C_{ii'jj'} \tag{14}
$$

The variable $y_{ii'jj'}$ is defined only as the existence of a contact between two neighboring nodes, where each node $N_{i',j'}$ that has an incoming connecting path activated must also have an outgoing path. In effect, this constraint can be thought of as a "mass" balance around the node. This constraint is specified for all nodes except those that are allowed to begin or end an alignment by Eq. 15.

$$
\begin{aligned}
&\sum_{i<i'} \sum_{j<j'} y_{ii'jj'} - \sum_{i''>i'} \sum_{j''>j'} y_{i'i''j'j''} = 0 \\
&\forall\, 1 < i' < M,\ 1 < j' < N
\end{aligned}
\tag{15}
$$

Equation 16 requires an alignment that matches the first residue in one of the two sequences to a residue in the other sequence. This constraint invalidates any alignment that aligns the first residue in both sequences to a gap, a physically meaningless alignment and allows for the path weights, $C_{ii'jj'}$, to be precalculated.

$$
\begin{aligned}
&\sum_{i'>1} \sum_{j} \sum_{j'>j} y_{i=1,i'jj'} + \sum_{i} \sum_{i'>i} \sum_{j'>j} y_{ii',j=1,j'} \\
&\qquad - \sum_{i'>1} \sum_{j'>1} y_{i=1,i',j=1,j'} = 1
\end{aligned}
\tag{16}
$$

If one sequence ends in a gap, the terminal residues of the other sequence must be prevented from aligning to earlier residues in a physically unrealistic way. Equation 17 allows exactly one active node $N_{i,j}$ involving a terminal residue in either Sequence 1 or Sequence 2.

$$\sum_{i<M}\sum_{j}\sum_{j'>j} y_{i,i'=M,jj'} + \sum_{i}\sum_{i'>i}\sum_{j<N} y_{ii'j,j'=N} - \sum_{i<M}\sum_{j<N} y_{i,i'=M,j,j'=N} = 1 \qquad (17)$$

It is more efficient and more meaningful to restrict the search to within a maximum alignment length, K. Equations 18–19 require the sum of the sequence length and the number of gaps created by the alignment to be less than the maximum alignment length for Sequences 1 and 2 respectively.

$$\begin{aligned}&\sum_{i}\sum_{i'>i}\sum_{j}\sum_{j'>j+1}(j'-j)\cdot y_{ii'jj'} + \\ &\sum_{i}\sum_{i'>i}\sum_{j'>1}(j-1)\cdot y_{ii',j=1,j'} + \\ &\sum_{i}\sum_{i'>i}\sum_{j<N}(j'-N)\cdot y_{ii'jj'=N} + M \le K\end{aligned} \qquad (18)$$

$$\begin{aligned}&\sum_{i}\sum_{i'>i+1}\sum_{j}\sum_{j'>j}(i'-i)\cdot y_{ii'jj'} + \\ &\sum_{i'>1}\sum_{j}\sum_{j'>j}(j-1)\cdot y_{i=1,i'jj'} + \\ &\sum_{i<M}\sum_{j}\sum_{j'>j}(j'-N)\cdot y_{i,i'=M,jj'} + N \le K\end{aligned} \qquad (19)$$

Equations 14–19 form the general mathematical model for the path selection approach to the global pairwise sequence alignment problem. Any of the three models presented can be expanded to include functionally-specific constraints, integer cut constraints, and pairwise interactions. Only the constraints necessary to include these features in the path selection model will be presented here.

Functionally-Specific Constraints

For some sequence alignment problems, specific residues are related to the function of a protein and should be maintained in a meaningful sequence alignment. This idea can be enforced in a mathematically rigorous way. These constraints can only be defined if the node existence variables, N_{ij}, are connected to the path existence variables, $y_{ii'jj'}$. One way to accomplish this is by summing over a pair of indices within the path variables, as shown in Eqs. 20–21.

$$\sum_{i<i',j<j'} y_{ii'jj'} = N_{i'j'} \quad \forall i' > 1, j' > 1 \qquad (20)$$

$$\sum_{i'>i,j'>j} y_{ii'jj'} = N_{ij} \quad \forall i = 1 \text{ or } j = 1 \qquad (21)$$

Constraints enforcing residue identity can then be written in terms of the N_{ij} variables. If position i^* in Sequence 1 must be conserved to maintain function, then Eq. 22 enforces this requirement.

$$\sum_{j;AA_{i^*}=AA_j} N_{i^*j} = 1 \qquad (22)$$

Integer Cut Constraints

This alignment model can be further extended by introducing integer cut constraints. After each solve of the above model, the previous solution is excluded from the feasible solution space by Eq. 23. A is the set of active variables in the solution to be excluded, I is the set of inactive variables and card(A) is the cardinality of set A, or the number of members of set A.

$$\sum_{(ii'jj')\in A} y_{ii'jj'} - \sum_{(ii'jj')\in I} y_{ii'jj'} \le \mathrm{card}(A) - 1 \qquad (23)$$

Pairwise Interaction Scores

A score can also be assigned for the alignment of a pair of amino acids i, i' in one sequence to a specific pair of amino acids j, j' in the second sequence. One promising application of these pairwise interactions scores is the ability to better evaluate the fitness of an alignment between a protein of known structure and an unknown protein with remote sequence homology. A number of recently developed C^{α}-based distance dependent force fields [7,13,16] are a good source for these scores because they allow some flexibility between the backbones of these two structures.

A pairwise interaction score requires the definition of the variable $z_{ii'jj'}$, representing the successful alignment of both i, j (N_{ij}) and i', j' ($N_{i'j'}$). This variable is initially introduced in Eq. 24 as the product of two node existence binary variables.

$$z_{ii'jj'} = N_{ij}\cdot N_{i'j'} \quad \forall i, i', j, j' \qquad (24)$$

Equation 24 is nonlinear and must be linearized using standard optimization techniques by Eqs. 25–27, which replace Eq. 24.

$$\sum_{ij} z_{ii'jj'} \leq N_{i'j'} \qquad \forall i', j' \tag{25}$$

$$\sum_{i'j'} z_{ii'jj'} \leq N_{ij} \qquad \forall i, j \tag{26}$$

$$N_{ij} + N_{i'j'} - 1 \leq z_{ii'jj'} \qquad \forall i, i', j, j' \tag{27}$$

Let the score of a pairwise interaction be denoted as $P_{ii'jj'}$. The objective function of Eq. 14 is expanded to include an additional contribution as shown in Eq. 28.

$$\max \sum_{i} \sum_{i'>i} \sum_{j} \sum_{j'>j} y_{ii'jj'} \cdot C_{ii'jj'} + z_{ii'jj'} \cdot P_{ii'jj'} \tag{28}$$

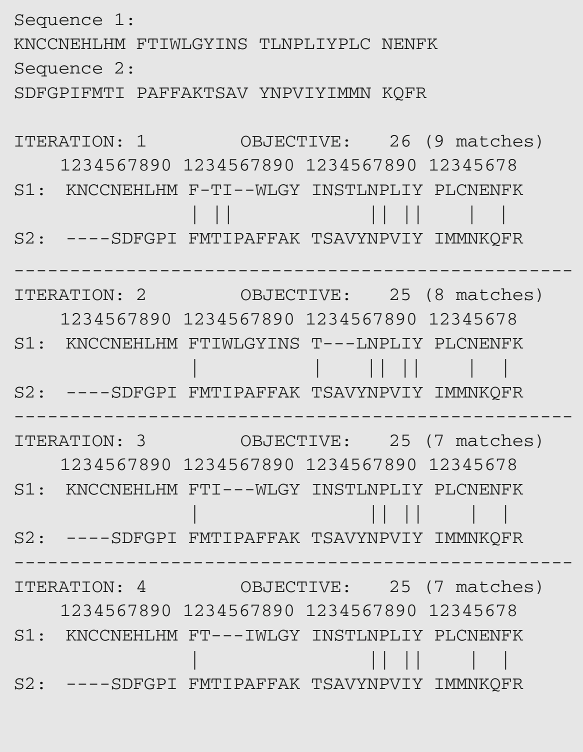
The ability of the sequence alignment models to easily allow for pairwise interaction scores illustrates their true power and flexibility. The model is guaranteed to converge to the optimal solution even for problems of this type. This guarantee suggests the effectiveness that could be achieved by incorporating such a model into a fold recognition framework.

Results and Discussion

The mixed-integer linear programming models of Sect. "Models" can address generic sequence alignment problems of a reasonable size. This method will be illustrated on an alignment of G-protein coupled receptors with the use of integer cut constraints and an alignment of pancreatic trypsin inhibitors demonstrating the use of functionally-relevant conservation constraints. All the alignments are calculated using the BLOSUM62 scoring matrix and an affine gap model with a gap opening penalty of 11 and a gap extension penalty of 1.

G-protein Coupled Receptors

G-protein coupled receptors are a type of membrane protein that regulate material and ion transport across a cell membrane, a reason they are a popular target for drug development. The alignment of the seventh transmembrane helix of bovine rhodopsin (34 amino acids) to the seventh transmembrane helix of H1R (35 amino acids), the first human histamine receptor, will be considered to illustrate alignment uncertainty [9]. Figure 2 shows the the regions of uncertainty in the sequence alignment using integer cut constraints. There is a strong conservation of alignment at the ends of the selected sequence, including the preservation of the highly conserved NPxxY motif. The central regions of the aligned sequences shows more variability. This observation could be a result of less structural conservation in the region, or less sequence similarity required for structural (and functional) conservation.



```
Sequence 1:
KNCCNEHLHM FTIWLGYINS TLNPLIYPLC NENFK
Sequence 2:
SDFGPIFMTI PAFFAKTSAV YNPVIYIMMN KQFR

ITERATION: 1         OBJECTIVE:   26 (9 matches)
     1234567890 1234567890 1234567890 12345678
S1:  KNCCNEHLHM F-TI--WLGY INSTLNPLIY PLCNENFK
                | ||            || ||    |  |
S2:  ----SDFGPI FMTIPAFFAK TSAVYNPVIY IMMNKQFR

-------------------------------------------------
ITERATION: 2         OBJECTIVE:   25 (8 matches)
     1234567890 1234567890 1234567890 12345678
S1:  KNCCNEHLHM FTIWLGYINS T---LNPLIY PLCNENFK
                |          |    || ||    |  |
S2:  ----SDFGPI FMTIPAFFAK TSAVYNPVIY IMMNKQFR
-------------------------------------------------
ITERATION: 3         OBJECTIVE:   25 (7 matches)
     1234567890 1234567890 1234567890 12345678
S1:  KNCCNEHLHM FTI---WLGY INSTLNPLIY PLCNENFK
                |               || ||    |  |
S2:  ----SDFGPI FMTIPAFFAK TSAVYNPVIY IMMNKQFR
-------------------------------------------------
ITERATION: 4         OBJECTIVE:   25 (7 matches)
     1234567890 1234567890 1234567890 12345678
S1:  KNCCNEHLHM FT---IWLGY INSTLNPLIY PLCNENFK
                |               || ||    |  |
S2:  ----SDFGPI FMTIPAFFAK TSAVYNPVIY IMMNKQFR
```

Global Pairwise Protein Sequence Alignment via Mixed-Integer Linear Optimization, Figure 2
A rank-ordered list of the top four optimal alignments of the helix 7 region of the human histamine receptor (Sequence 1) to the helix 7 region of the bovine rhodopsin (Sequence 2) for a template length of 50 residues

A comparison of the computational resources required for this problem is presented in Table 1. A larger template length results in a more complex optimization problem to be solved. The path selection model significantly outperforms the other formulations, especially for the larger template lengths.

```
Sequence 1:
MLKYTSISFL LIILLFSFTN ANPDCLLPIK TGPCKGSFPR YAYDSSEDKC
VEFIYGGCQA NANNFETIEE CEAACL

Sequence 2:
RPDFCLEPPY TGPCKARIIR YFYNAKAGLC QTFVYGGCRA KRNNFKSAED
CMRTCGGA

OBJECTIVE:     141 (26 exact matches)
     1234567890 1234567890 1234567890 1234567890 1234567890
S1:  MLKYTSISFL LIILLFSFTN ANPD-CLLPI KTGPCKGSFP RYAYDSSEDK
                             || || |   |||||     || |
S2:  ---------- ---------- -RPDFCLEPP YTGPCKARII RYFYNAKAGL

S1:  CVEFIYGGCQ ANANNFETIE ECEAACL--
     |  | ||||  |  |||   |  |   |
S2:  CQTFVYGGCR AKRNNFKSAE DCMRTCGGA
```

Global Pairwise Protein Sequence Alignment via Mixed-Integer Linear Optimization, Figure 3
Optimal alignment of bombyx mori kazal-type serine proteinase inhibitor 1 (Sequence 1) to bovine pancreatic trypsin inhibitor (Sequence 2), given the requirement of cysteine conservation and a template length of 100

Global Pairwise Protein Sequence Alignment via Mixed-Integer Linear Optimization, Table 1
Computational performance of the template-based (TB), template-free (TF) and path selection (PS) models for helix 7 of the G-protein coupled receptor proteins (run times in seconds on an Intel Pentium 3.2 GHz processor, using CPLEX 9.0)

K	TB	TF	PS	Objective
40	1000+	35.21	1.30	26,25,25,25
45	1000+	177.8	5.27	26,25,25,25
50	1000+	1000+	14.71	26,25,25,25

Global Pairwise Protein Sequence Alignment via Mixed-Integer Linear Optimization, Table 2
Computational performance of the template-based (TB), template-free (TF) and path selection (PS) models for the alignment of bombyx mori kazal-type serine proteinase inhibitor 1 to bovine pancreatic trypsin inhibitor (run times in seconds on an Intel Pentium 3.2 GHz processor, using CPLEX 9.0)

K	TB	TF	PS	Objective
80	886.4	0.36	0.64	141
90	1000+	80.3	1.74	141
100	1000+	912.4	2.77	141

Serine Protease Inhibitors

Serine protease inhibitors are responsible for regulating serine proteases, proteins necessary for hydrolyzing peptides. One well-studied protein within this class is the bovine pancreatic trypsin inhibitor (BPTI). Its native three-dimensional structure is stabilized by 3 disulfide bonds that are conserved across the class of serine protease inhibitors. An alignment of BPTI (58 amino acids) to the bombyx mori (domestic silkworm) kazal-type serine protease inhibitor (76 amino acids) has previously been investigated in the context of introducing constraints for the functionally important conservation of the disulfide bonds [8]. The results of such an alignment are presented in Fig. 3. The six conserved cysteine residues necessary for the formation of the three disulfide bridges that stabilize the functional protein are apparent from this alignment.

A comparison of the computational resources required for this problem is presented in Table 2. Even with the inclusion of the conservation constraints, the path selection model still solves this alignment example quite rapidly for large template lengths. Although the template-free approach slightly outperforms the path selection approach for short template length restrictions, it does not scale very well with increases in template length. Similar to the first example, the template-free approach solves the problem significantly faster than the template-based approach, but the path selection approach is superior to both of the mixed-integer linear programming techniques.

References

1. Altschul S, Gish W, Miller W, Myers EW, Lipman DJ (1990) Basic local alignment search tool. J Mol Biol 215:403–410
2. Altschul SF, Madden TL, Schaffer AA, Zhang J, Zhang Z, Miller W, Lipman DJ (1997) Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. Nucl Acids Res 25:3389–3402
3. Dayhoff M, Schwartz R, Orcutt B (1978) A model of evolutionary change in proteins. Atlas of Protein Sequences and Structures, vol 5. National Biomedical Research Foundation, Washington DC
4. He D, Arslan AN (2005) A space-efficient algorithm for the constrained pairwise sequence alignment problem. Genome Inform 16:237–246
5. Henikoff S, Henikoff JG (1992) Amino acid substitution matrices from protein blocks. Proc Natl Acad Sci USA 89:10915–10919
6. Floudas CA (1995) Nonlinear and Mixed-integer Optimization: Fundamentals and Applications. Oxford University Press, New York
7. Loose C, Klepeis JL, Floudas CA (2004) A new pairwise folding potential based on improved decoy generation and side-chain packing. Prot Struct Funct Bioinf 54:303–314
8. McAllister SR, Rajgaria R, Floudas CA (2007) A template-based mixed integer linear programming sequence alignment model. In: Torn A, Zilinskas J (eds) Models and Algorithms for Global Optimization, Springer Optimization and Its Applications. Springer, New York, pp 343–360
9. McAllister SR, Rajgaria R, Floudas CA (2007) Global pairwise sequence alignment through mixed integer linear programming: A template free approach. Optim Method Softw 22:127–144
10. Needleman SB, Wunsch CD (1970) A general method applicable to the search for similarities in the amino acid sequence of two proteins. J Mol Biol 48:443–453
11. Pearson WR (1990) Rapid and sensitive sequence comparison with fastp and fasta. Methods Enzymol 183:63–98
12. Pearson WR, Lipman DJ (1988) Improved tools for biological sequence comparison. Proc Natl Acad Sci USA 85:2444–2448
13. Rajgaria R, McAllister SR, Floudas CA (2006) A novel high resolution C-alpha C-alpha distance dependent force field based on a high quality decoy set. Prot Struct Funct Bioinf 65:726–741
14. Smith TF, Waterman MS (1981) Identification of common molecular subsequences. J Mol Biol 147:195–197
15. Tang CY, Lu CL, Chang MD, Tsai YT, Sun YJ, Chao KM, Chang JM, Chiou YH, Wu CM, Chang HT, Chou WI (2003) Constrained multiple sequence alignment tool development and its application to RNase family alignment. J Bioinform Comput Biol 1:267–287
16. Tobi D, Elber R (2000) Distance-dependent, pair potential for protein folding: Results from linear optimization. Prot Struct Funct Bioinf 41:40–46
17. Vingron M (1996) Near-optimal sequence alignment. Curr Opin Struct Biol 6:346–352

Global Supply Chain Models

BURAK EKSIOGLU
Industrial and Systems Engineering Department,
University Florida, Gainesville, USA

MSC2000: 90B05, 90B06

Article Outline

Keywords

Stochastic dynamic programming; Multiperiod stochastic program

A *supply chain* (SC) may be defined as an integrated process where several business entities such as suppliers, manufacturers, distributors, and retailers work together to plan, coordinate and control the flow of materials, parts, and finished goods from suppliers to customers. This chain is concerned with two distinct flows: a forward flow of materials and a backward flow of information. Similarly, a *global supply chain* (GSC) may be defined as a SC where one or more of these business entities operate in different countries. For many years, researchers and practitioners have concentrated on the individual processes and entities within the SC. Within the past few years, however, there has been an increasing effort in optimizing the entire SC. This article intends to highlight some of the early results from the 1960s to 1995 that have led to today's SC research and most of the recent results that address the design and management of GSC networks (as of 2000).

Within manufacturing and logistics research, the current stream of SC research is largely built on prior work in the area of multi-echelon inventory models. The early works [4,5] and [14] form the basis for most of the research done in this area. See [13] and [3] for extensive reviews of multi-echelon inventory models. For detailed and more recent discussions of multi-echelon models, see [12,19,20].

As companies began to realize the benefits of optimizing the SC as a single entity, researchers began utilizing operations research (OR) techniques to better model supply chains. See [2] for an extensive review of the literature in SC modeling. Typically, a SC model tries to determine:

- the transportation modes to be used;
- the suppliers to be selected;
- the amount of inventory to be held at various locations in the chain;
- the number of warehouses and plants to be used; and
- the location and capacities of these warehouses and plants.

However, as a result of the globalization of the economy, the models have become more complex. GSC models now often try to include factors such as exchange rates, international interest rates, trade barriers, taxes and duties, market prices, and duty drawbacks. All of these factors are generally difficult to include in mathematical models because of the uncertainty and nonlinearity they introduce.

See [21] and [7] for extensive reviews on GSC models. [21] concentrates on strategic production-distribution models whereas [7] focus on the integration of SC network optimization with real options pricing methods. This article complements these reviews by giving a chronological listing of the models in both areas.

In [15] an international *facility location* model is presented. This is one of the first mathematical programs that includes financial aspects in GSC modeling. The authors develop a *large scale nonlinear mixed integer programming problem* (MIP). The objective function takes into account the expected profit and the variance of the profit, where the variance of the profit is multiplied by a risk aversion factor. Plant capacities, market demands and financial constraints are included in the model. The formulation considers production and transportation costs, exchange rate fluctuations, international interest rates, market prices, import tariffs, and export taxes.

In [9] a deterministic model is proposed for maximizing the after tax profit of a large scale international distribution network. Transportation costs, fixed setup costs, variable production and purchasing costs, and fixed vendor costs are included in the model. The model enforces production capacity constraints, demand limits, material requirements at each plant, supplier capacity constraints, balance constraints at plants and distribution centers, feasible flow constraints, and offset trade requirements. The model is run sequentially over a fixed time horizon and computational results are presented for various problem sizes.

In [6] the differences are analyzed between an international SC model and a single-country model, and a dynamic, nonlinear MIP model is developed. The inclusion of features such as duties, tariffs, tax rates, and exchange rates produce models that are very difficult to solve optimally even for small size problems.

In [8] a normative model is presented for the operations of a global company. Plant location, capacity and product mix, and material and cash flow determination are the decisions included in the model. The model consists of a master problem and a set of subproblems. The master problem is a *multiperiod stochastic program* and the subproblems are single period stochastic programs. These problems are linked through a set of submodels such as a stochastic SC model, a financial flow model, a stochastic exchange rate model, and a price-demand model.

In [17] a *stochastic dynamic programming* (DP) model is developed that treats the SC as equivalent to owning a financial option instrument. The value of the option depends on the real exchange rate. The authors consider production switching between two manufacturing plants located in different countries depending on the real exchange rate. The model does not consider characteristics such as multiple products or different SC stages. The model becomes intractible for more than one exchange rate process.

In [1] a comprehensive, multiperiod, multicommodity MIP model is proposed which is used to optimize the SC of Digital Equipment Corporation (DEC). The objective of the model is to minimize a function of total production and distribution cost, savings from credit, and an additional term which contains production and transportation times. The total cost includes fixed and variable costs of production, transportation cost, material handling, inventory, and overhead costs. The savings from credit are due to reexporting products. The model enforces constraints on demand satisfaction, production and throughput capacities at each facility, and bounds on decision variables. In addition, international constraints such as duty drawback, duty relief, and offset trade are included. The authors describe how DEC used this model to manage their GSC.

In [18] a multiperiod stochastic DP is introduced that allows the firm to switch among several production modes to maximize profit. The production modes they consider are exporting from the home country, a joint venture with local partners, and establishing a wholly owned subsidiary for a foreign firm. It concludes by identifying cases in which one of these modes would be preferred to the others.

In [16] a stochastic DP formulation is developed for the valuation of global manufacturing strategy options. A hierarchical approach is proposed. First, the exchange rates are modeled by multinomial approximations. Then, options for alternative product and SC network designs are determined based on the firm's global manufacturing strategy. Finally, an MIP model for each exchange rate within every period is solved and the value of several manufacturing options is determined. The expected profit for each policy option is found by solving a stochastic DP using the values of the manufacturing policies.

In [11] the problem of operating a network of plants that are partially-owned subsidiaries of a multinational corporation is analyzed. Using real data, a model of three subsidiaries and four countries is developed for one industry and the effects of coordination under various macroeconomic conditions are discussed.

In [10] optimal policies for operating a network of plants located in different countries is studied. It is assumed that production costs are stochastic and are influenced by factors such as exchange rates, inflation, taxes, and tariffs. There is a one-time charge for switching (production volume changes between countries) and variable production costs are either concave or piecewise linear convex at each plant. It is also assumed that demand is deterministic and stationary. Under these assumptions a two-country, single market stochastic DP model is developed. The authors show that the optimal policy is always a barrier policy when switching costs are linear or step functions. (A *barrier policy* is a policy in which each plant operates either at a minimum or a maximum output level.)

The literature on GSC management is quite recent and the models developed usually do not consider most of the uncertainties that international corporations face. Each model addresses a limited number of the aspects of managing a GSC. There is an ongoing effort to develop more comprehensive and practical GSC design models that will accommodate the needs of the rapidly changing global economy.

See also

- ► Inventory Management in Supply Chains
- ► Nonconvex Network Flow Problems
- ► Operations Research Models for Supply Chain Management and Design
- ► Piecewise Linear Network Flow Problems

References

1. Arntzen BC, Brown GG, Harrison TP, Trafton LL (1995) Global supply chain management at Digital Equipment Corporation. Interfaces 25(1):69–93
2. Beamon BM (1998) Supply chain design and analysis: Models and methods. Internat J Production Economics 55:281–294
3. Bhatnagar R, Chandra P, Goyal SK (1993) Models for multiplant coordination. Europ J Oper Res 67:141–160
4. Clark AJ, Scarf H (1960) Optimal policies for a multi-echelon inventory problem. Managem Sci 6(4):475–490
5. Clark AJ, Scarf H (1962) Approximate solutions to a simple multi-echelon inventory problem. In: Arros KJ, Karlin S, Scarf H (eds) Stud. Appl. Probab. and Management Sci., Stanford Univ. Press, Palo Alto, CA, pp 88–110
6. Cohen MA, Fischer M, Jaikumar R (1989) Internat. manufacturing and distribution networks: A normative model framework. In: Ferdows K (ed) Managing Internat. Manufacturing. North-Holland, Amsterdam, pp 67–93
7. Cohen MA, Huchzermeier A (1998) Global supply chain management: A survey of research and applications. In: Tayur S, Ganeshan R, Magazine M (eds) Quantitative Models for Supply Chain Management. Kluwer, Dordrecht
8. Cohen MA, Kleindorfer PR (1993) Creating value through operations: The legacy of Elwood S. Buffa. In: Sarin RK (ed) Perspectives in Oper. Management (Essays in Honor of Elwood S. Buffa), Kluwer, Dordrecht, pp 3–21
9. Cohen MA, Lee HL (1989) Resource deployment analysis of global manufacturing and distribution networks. J Manufacturing Oper Management 2:81–104
10. Dasu S, Li L (1997) Optimal operating policies in the presence of exchange rate variability. Managem Sci 43(5):705–722
11. Dasu S, de la Torre J (1997) Optimizing an international network of partially owned plants under contiditions of trade liberalization. Managem Sci 43(3):313–333

12. Diks EB, de Kok AG, Lagodimos AG (1996) Multi-echelon systems: A service measure perspective. Europ J Oper Res 95:241–263
13. Federgruen A (1993) Centralized planning models for multi-echelon inventory systems under uncertainty. In: Graves S, Rinnooy Kan AHG, Zipkin P (eds) Logistics of Production and Inventory, North-Holland, Amsterdam, pp 133–173
14. Geffrion AM, Graves GW (1974) Multicommodity distribution system design by Benders Decomposition. Managem Sci 20(5):822–844
15. Hodder JE, Dincer MC (1986) A multifactor model for international plant location and financing under uncertainty. Comput Oper Res 13(5):601–609
16. Huchzermeier A, Cohen MA (1996) Valuing operational flexibility under exchange rate risk. Oper Res 44(1):100–113
17. Kogut B, Kulatilaka N (1994) Operating flexibility, global manufacturing, and the option value of a multinational network. Managem Sci 40(1):123–139
18. Kouvelis P, Sinha V (1995) Exchange rates and the choice of production strategies: Supplying Foreign Markets. Duke Univ., Durham, NC
19. Tayur S, Ganeshan R, Magazine M (1998) Quantitative models for supply chain management. Kluwer, Dordrecht
20. van Houtum GJ, Inderfurth K, Zijm WHM (1996) Materials coordination in stochastic multi-echelon systems. Europ J Oper Res 95:1–23
21. Vidal CJ, Goetschalckx M (1997) Strategic production-distribution models: A critical review with emphasis on global supply chain models. Europ J Oper Res 98:1–18

Global Terrain Methods

ANGELO LUCIA
Department of Chemical Engineering,
University of Rhode Island, Kingston, USA

Article Outline

Introduction

Global terrain methods [5,6,7] are a class of methods for solving nonlinear programming problems that are based on the simple concept of intelligently following valleys up and down on the terrain or landscape of three times continuously differentiable or C^3 objective function surfaces. They belong to the class of integral path or path following methods [1,2,3,4,8] and can also be used to solve systems of nonlinear equations formulated as nonlinear least-squares problems. The overall approach is based on the reliable and efficient computation of minima, saddle points, and singular points and a terrain-following algorithm to efficiently move from one stationary point to another or to a boundary of the feasible region. What makes global terrain methods superior to other path following methods is the Newton-based *predictor-corrector* method used to move uphill on the objective function landscape.

Formulation

The problem under consideration is that of finding a number of minima, saddle points, and singular points of a C^3 objective function, $\phi = \phi(z)$, defined on R^n subject to bounds on variables, $c(z)$, where z are the optimization variables. Let $F = F(z)$ denote the gradient of φ and $J(z)$ denote the $n \times n$ symmetric Jacobian matrix of F (or Hessian matrix of φ).

Problem Statement

The problem can be stated in the form

$$\text{Find } \{z_k^*\} : z_k^* \le c(z^*) \text{ such that } \nabla(F^T F) = 0\,, \quad (1)$$

where $\{z_k^*\}$ denotes a set of minima, saddle points, and/or singular points, and the constraints are given by

$$-z_i^* \le z_i^L \text{ and } z_i^* \le z_i^U\,, \tag{2}$$

where z_i^L and z_i^U are the lower and upper bounds on the variable z_i.

Note that $\nabla(F^T F) = J^T F = 0$ implies that either

$$F(z_k^*) = 0\,, \tag{3}$$

$$\det J = 0 \text{ with null space vector } F \neq 0\,. \tag{4}$$

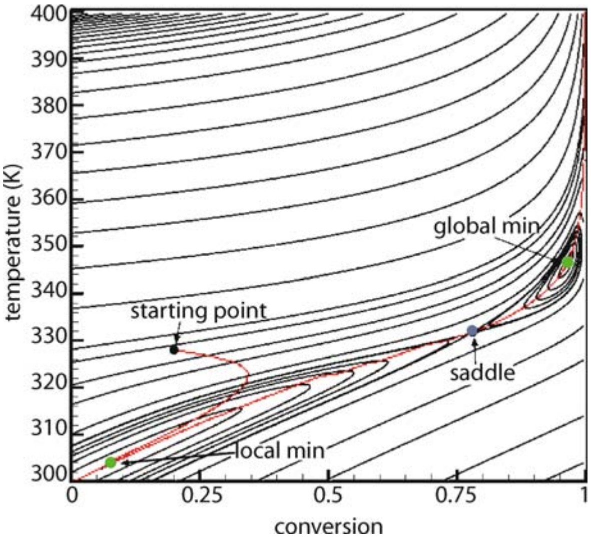
If $z_k^* *$ satisfies Eq. (3), it is either a minimum or a saddle point of φ whereas if z_k^* satisfies Eq. (4), it is a singular point of J. To distinguish between minima and saddle points, the Hessian matrix of $F^T F$ is required, which is

$$H = J^T J + \Sigma F_i G_i\,, \tag{5}$$

where F_i is the ith element function of F and where G_i is the corresponding element Hessian matrix of F_i. If all eigenvalues of H are positive, z_k^* is a minimum of φ. If at least one eigenvalue of H is negative, z_k^* is a saddle point of φ.

Geometrical Foundation

Figure 1 shows the contours of $F^T F$ along with the terrain path for a simple two dimensional reactor example. To understand the underlying geometric foundation on which global terrain methods are built, consider two neighboring contours or level curves along the curved valley shown in Fig. 1. Note that the distance, Δ, between any two neighboring level curves in the normalized gradient direction is largest exactly in the valley and that this distance decreases in magnitude as points move out of the valley along the same neighboring level curves (i. e., the contours become more tightly packed together). Therefore the norm of $J^T F$ must be smaller at any point in the valley than at any neighboring point on any given level curve since the same change in the least-squares function results from the largest change in distance. Thus the valley connecting the stationary points shown in Fig. 1 can be characterized as the collection of local minima in the norm of $J^T F$ over a set of level curves. This same constrained extremum in the gradient norm also characterizes ridges, ledges and other distinct features of the objective function landscape in any n-dimensional space.



Global Terrain Methods, Figure 1
Contours of a least squares surface

Valleys, ridges, ledge, etc. can be defined mathematically by a set of solutions, V, to a sequence of general nonlinearly, constrained optimization problems

$$V = \{\min g^T g \text{ such that } F^T F = L\,,\ \text{for all } L \in \Lambda\}\,, \tag{6}$$

where F and J are defined as before and where $g = 2J^T F$, L is any given value (or level) of the least-squares objective function, and Λ is some collection of contours. That is, for any given level curve, we find the point on L that corresponds to a local minimum in $g^T g$. The collection of minima for all levels gives all (or part) of a valley, ridge, or ledge. Equation (6) forms the geometrical backbone for global terrain methods and plays an important role in the development of predictor-corrector algorithms used to implement those ideas. Moreover, Λ is actually a computational by-product of the terrain-following approach.

It is useful to simultaneously monitor behavior on the landscape of $F^T F$ and the objective function landscape, noting that minima and saddle points on φ are minima on $F^T F$ while singular points on φ are saddle points on $F^T F$. Valleys on both surfaces closely align.

Methods

Terrain-following methods are comprised of a sequence of sub-problems that unfold dynamically during the course of solving a nonlinear programming problem. Since global terrain methods move up and down the landscape of $F^T F$, these sub-problems include

1) Reliable downhill equation solving.
2) Reliable and efficient computation of singular points.
3) Efficient uphill movement comprised of predictor-corrector calculations.
4) Reliable and efficient eigenvalue-eigenvector computations.
5) Effective bookkeeping.
6) A termination criterion to decide when the computations have finished.
7) Advanced techniques to deal with bifurcations and non-differentiable points.

1) Moving Downhill

Downhill computations use a trust region method and are capable of finding minima, saddle points, and sometimes singular points on an objective function surface. In finding the first point, say z_1^*, initiation of downhill computations is arbitrary. On subsequent downhill sub-problems, calculations always begin in the direction of the smallest negative eigenvalue of H.

The basic downhill iteration is defined as follows

$$\Delta = -\beta \Delta_N + (\beta - 1) g\,, \tag{7}$$

where $\Delta_N = J^{-1} F$ is the Newton direction and $\beta \in [0,1]$ is determined by the following simple rules. If $||\Delta_N|| \leq R$, then $\beta = 1$, where R is the trust region radius. If $||\Delta_N|| > R$ and $||F|| \geq R$, then $\beta = 0$. Otherwise, β is the unique value in Eq. (7) on [0,1] that satisfies $||\Delta|| = R$. The new iterate is accepted if it reduces $||F||$. Otherwise, the new iterate is rejected, the trust region radius is reduced and the calculations are repeated until a reduction in $||F||$ occurs. Downhill movement is terminated when either $||F|| \leq \varepsilon$, where ε is a convergence tolerance, or $||F|| \,/\, ||\Delta_N|| \leq \zeta$, where ζ is some small number (typically 10^{-6}). This latter condition implies that the Newton step is very large in comparison to the gradient and the computations are converging to a singular point. The algorithm then switches to quadratic acceleration.

2) Acceleration to Singular Points

During downhill movement, quadratic acceleration is used if $||F||/||\Delta_N|| \leq \zeta$. Quadratic acceleration is also used during uphill calculations to converge to singular points and is defined by

$$\Delta = -H^{-1} J^T F\,. \tag{8}$$

During acceleration, norm reduction in F is not enforced because H can have eigenvalues of mixed sign.

3) Moving Uphill

Uphill movement is initiated in the eigen-direction associated with the smallest positive eigenvalue of the Hessian matrix H and consists of two basic parts – Newton predictor steps and successive quadratic programming (SQP) corrector steps.

Uphill Predictor Steps Predictor steps follow a valley uphill but will 'drift' from the valley – as shown in the slight zigzag in the terrain path in Fig. 1, which shows this 'drift' (followed by corrector steps). Uphill Newton steps are defined by

$$\Delta_p = \alpha \Delta_N\,, \tag{9}$$

where $\Delta_N = J^{-1} F$ and the step size $\alpha \in (0,1]$.

Uphill Corrector Steps Corrector steps (again see Fig. 1) are used intermittently to force iterates back to a valley and are invoked when the condition

$$\theta = 57.295 \arccos\left[\left(\Delta_N^T c\right)/(||\Delta_N||\;||v||)\right] \geq \Theta\,, \tag{10}$$

is satisfied, where v is the current estimate of the eigenvector associated with the smallest positive eigenvalue of H and Θ is 5 degrees. Corrector steps are formulated as

$$\min g^T g \text{ such that } F^T F = L\,, \tag{11}$$

where L is the current value of $F^T F$. Corrector steps are iterative and are considered converged when the necessary conditions

$$F^T F - L = 0\,, \tag{12}$$

$$Hg - \lambda g = 0\,, \tag{13}$$

are satisfied. Corrector steps are computed using a successive quadratic programming (SQP) method; however, other methods can be used for this purpose. The SQP formulation for the problem defined by Eq. (11) is given by

$$\min g^T H\Delta_c + \frac{1}{2}\Delta_c^T M\Delta_c \text{ such that} \\ g^T\Delta_c = -(F^T F - L)\,, \tag{14}$$

where M is the Hessian matrix of the Lagrangian function. The Lagrangian function is defined by $L = g^T g - \lambda(F^T F - L)$, where λ is a Lagrange multiplier and where M is approximated by the rule $M = H^T H - \lambda H$.

4) Eigenvalue-Eigenvector Computations

It is not always necessary to find all eigenvalues and eigenvectors of H to decide whether to begin the next phase of the computations uphill or downhill – particularly for problems with large n. Often it is sufficient to compute a subset of eigenvalues and eigenvectors, which can be conveniently performed using the inverse power method. The inverse power method solves the inverse form of

$$Hv - \lambda v = 0\,, \tag{15}$$

by constructing the iteration

$$v_{k+1} = \lambda_k H^{-1} v_k\,, \tag{16}$$

$$\lambda_{k+1} = \frac{v_{k+1}^T v_{k+1}}{v_{k+1}^T H^{-1} v_{k+1}}\,, \tag{17}$$

where the calculations alternate between Eqs. (16) and (17) until $||v_{k+1} - \lambda_k H^{-1} v_k|| < \varepsilon$, where ε is some pre-specified tolerance. Note that an estimate of v is necessary to begin the inverse power method. Once the first eigenvalue, say λ_1, and its corresponding eigenvector, v_1, have been determined, the Hessian matrix is deflated using symmetric orthonormal projection to give an $(n-1) \times (n-1)$ symmetric matrix whose basis spans the space orthogonal to v_1. The inverse power method is used to find the next eigenvalue, λ_2, and its associated eigenvector, v_2, and then v_2 is lifted to R^n. This procedure of deflation by orthonormal projection to form an $(n-j) \times (n-j)$ symmetric matrix whose basis spans the space orthogonal to $\{v_1, v_2, \ldots, v_j\}$ followed by the inverse power method and the lifting of v_{j+1} to R^n is continued until as many eigenvalues and eigenvectors as desired are determined.

5) Effective Bookkeeping

Another important aspect of global terrain methods is that it is possible to avoid calculating the same z_k^* more than once by effective bookkeeping. This is accomplished by storing solution information that includes the set of solutions, the solution types (i. e., minimum, saddle point, or singular point), corresponding values of φ and $F^T F$, and the current set of eigenconnections (i. e., the smallest positive eigenvalue and associated eigenvector for minima and saddles, and the largest negative eigenvalue and associated eigenvector for singular points). Following the determination of the first stationary or singular point, z_1^*, uphill movement proceeds in the $+/-$ eigen-direction associated with the smallest positive eigenvalue of H. Assume that two new stationary or singular points, z_2^* and z_3^*, have been determined by these uphill calculations. The next move will be downhill from z_2^* in the eigen-directions, v_2, associated with the largest negative eigenvalue, λ_2, of H at z_2^*. However since z_2^* and z_3^* are connected by path to z_1^*, care must be exercised so as not follow the path back to z_1^*. To do this, nearest neighbors are determined by finding k such that

$$||z_2^* - z_k^*|| \text{ is minimum for all } k \neq 2\,. \tag{18}$$

Let j be the index for which Eq. (18) is satisfied. Following this, the direction $d_2 = z_2^* - z_j^*$ is defined. Correct downhill movement away from z_2^* is defined by whichever inequality

$$v_2^T d_2 < 0 \quad \text{or} \; - v_2^T d_2 < 0\,, \tag{19}$$

is satisfied. Note that the selection of the proper condition in Eq. (19) guarantees that initial movement from z_2^* will be in the direction away from the nearest solution z_j^*. Equations (18) and (19) can be easily generalized to give

$$||z_i^* - z_k^*|| \text{ is minimum for all } k \neq K\,, \tag{20}$$

$$v_i^T d_i < 0 \quad \text{or} \; - v_i^T d_i < 0\,, \tag{21}$$

where $d_2 = z_i^* - z_j^*$, j is the index that satisfies Eq. (20), and K is the current number of solutions.

6) Termination

Termination occurs when either the desired number of points $\{z_k^*\}$ have been calculated or a certain number of bounds are encountered. The first termination criterion is straightforward. In the normal case, termination occurs when two bounds have been encountered. When bifurcations have been detected, then the number of bounds that must be encountered for termination to occur is $n_\mathrm{b} + 2$, where n_b is the number of distinct bifurcations.

7) Advanced Techniques

For any global terrain method to be effective it must also to address issues such as parametric disconnectedness, integral path bifurcations, and non-differentiable points or manifolds.

Parametric Disconnectedness

Following solutions parametrically is the basis for many homotopy-continuation methods. However, when parametric solutions exist on disconnected branches of solution curves, continuation methods can have difficulties. Global terrain methods are completely unaffected by parametric disconnectedness since they operate in variable and not parameter space.

Non-differentiable Points and Manifolds

There are many engineering applications that exhibit non-differentiable points and/or manifolds as a consequence of inherent switching contained in the objective function. At the 'switch' points, non-differentiability can occur and there can be families of 'switch' points that form manifolds. Non-differentiable points or manifolds are easily detected because they often exhibit retrograde curvature as well as other qualitative changes in model behavior that can be readily monitored.

Figure 2 illustrates a case in which there is a non-differentiable manifold. In this figure, $z_1 = C_{10}, z_2 = C_{18}, z_3 = C_{21}, z_1+z_2+z_3 = 1$, which is why the feasible region is triangular shaped, and $0 \leq z_i \leq 1, i = 1, 2, 3$. This curved manifold of non-differentiable points denotes the boundary between qualitatively different types of behavior for the case where $\phi = \min[\phi_1, \phi_2]$ at each z and is usually not mentioned in discussions of optimization of physical models. However, it is important in computations. The global terrain methodology has no difficulties finding stationary and singular points on $F^T F$ in this case because it monitors all aspects of the φ thereby allowing switching take place on the fly and the correct stationary and singular points to be easily found.

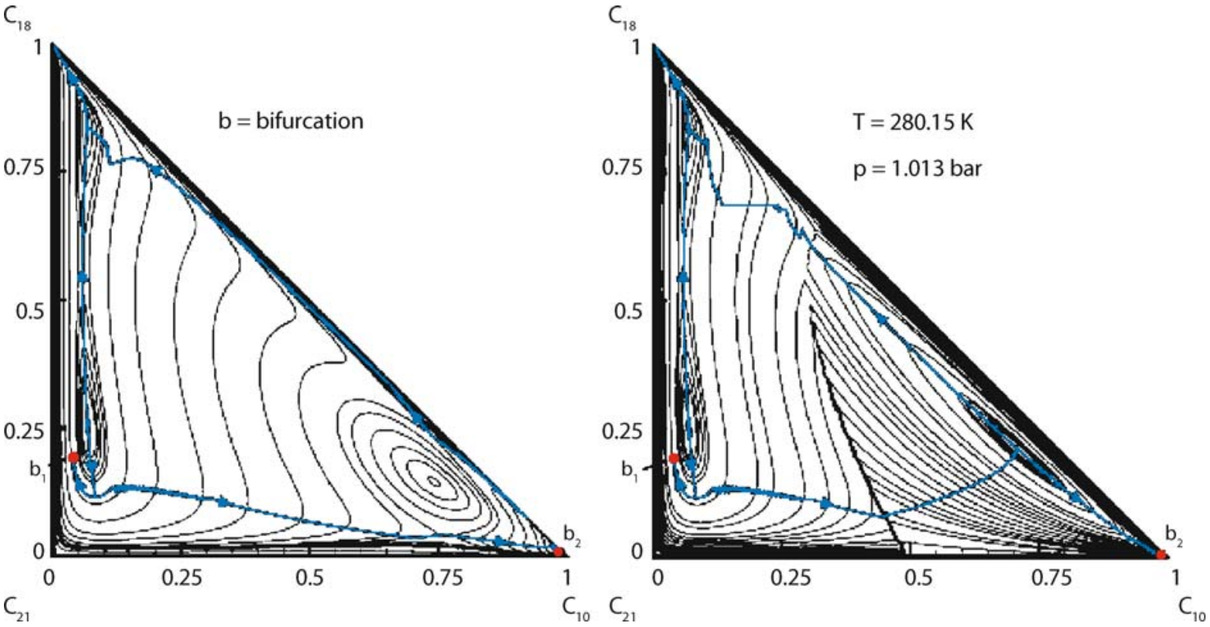
Integral Path Bifurcations

There are many applications in which integral paths either split into two or become tangent to a contour. These occurrences are called integral path bifurcations and can significantly impact the reliability of global terrain methods. Fortunately, Gauss curvature can provide a deterministic measure of the presence of bifurcation points.

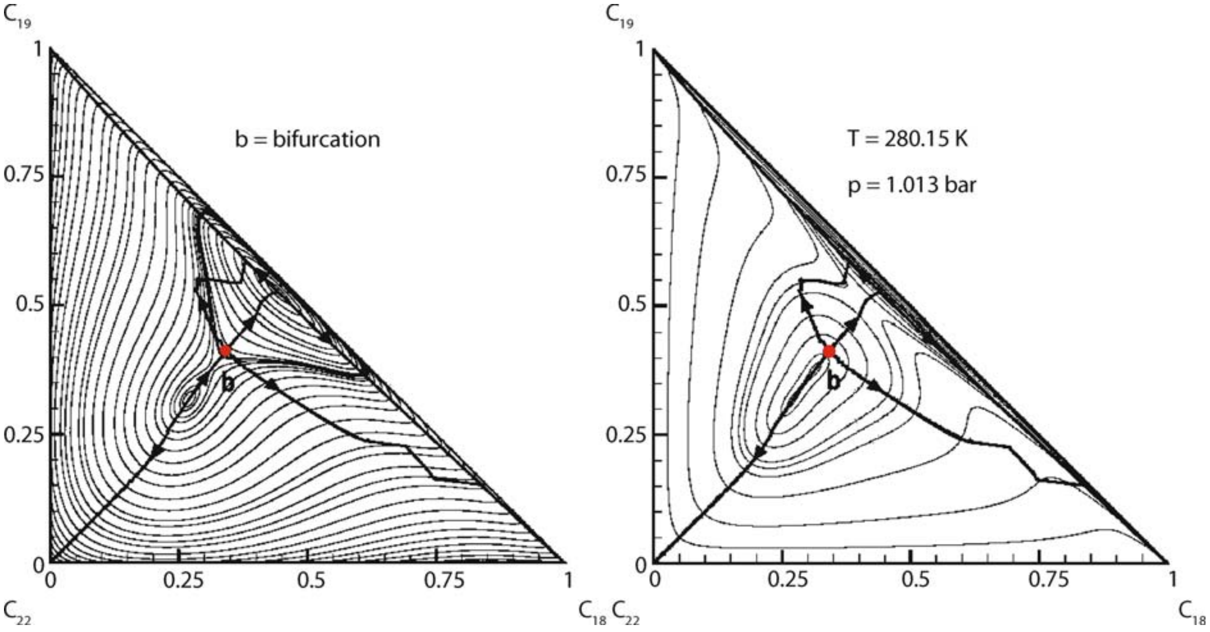
It is often easier to understand integral bifurcations from a geometrical perspective. Consider Fig. 3 where $z_1 = C_{18}, z_2 = C_{19}, z_3 = C_{22}, z_1 + z_2 + z_3 = 1$, and $0 \leq z_i \leq 1, i = 1, 2, 3$. Note that there is a pitchfork bifurcation at the point denoted by the point b on the integral path that runs from the two minima and the saddle point of $F^T F$ in the center of the triangle toward the saddle point and minimum very close to the hypotenuse of the triangular region. If the integral path bifurcation at b goes undetected, then the saddle point and minimum closest to the hypotenuse will not be found because corrector iterations will force iterates to turn toward the left or right hand branches of the pitchfork that end at the corners of the hypotenuse. Note, however, that the level curves begin to flatten in the neighborhood of the bifurcation point as the path moves toward the hypotenuse. This flattening, together with an eigenvector exchange from $J^T F$ to a vector in the tangent subspace of the level constraint, is a necessary condition for integral path *pitchfork* bifurcations, like the one that occurs at b. Moreover, flattening is relatively easy to measure by calculating (Gauss) curvature along a contour.

Gauss Curvature

To measure Gauss or Gauss–Kronecker curvature, it is necessary to calculate eigenvalues of the Hessian matrix, H, projected onto the tangent subspace of the level constraint, which is orthogonal to the gradient at any



Global Terrain Methods, Figure 2
An Objective Function & Gradient Surface with a Non-Differentiable Manifold (*left*) b ($F_1^T F_1$); (*right*) a Composite $F^T F$



Global Terrain Methods, Figure 3
Integral Path Bifurcation on Objective Function & Gradient Surfaces (*left*) Landscape of φ; (*right*) Landscape of $F^T F$

given point along the integral path. Gauss–Kronecker curvature corresponds to the determinant of this projected Hessian matrix. When the number of unknowns is two, this curvature is called Gauss curvature. Decreasing Gauss or Gauss–Kronecker curvature in a particular part of the feasible region indicates that the level curves are flattening and provides a strong reason to check for an exchange in the 'minimum' eigenvector of

H and, if warranted, to search for an integral path bifurcation.

Current implementation of these ideas measures flattening by calculating a few of the smallest eigenvalues (and eigenvectors) of the projection of H onto the tangent subspace *at each iteration* of the calculations. Without a theoretical basis that defines how often Gauss curvature should be measured, intermittent measurement seems ad hoc at best since very small regions of decreasing Gauss curvature could go undetected.

Finding Integral Path Tangent Bifurcations

This type of bifurcation point can be detected by measuring Gauss curvature and by comparing vectors along the flow of an integral path and vectors in the tangent subspace of the level sets for points on the path. When Gauss curvature decreases and the flow of the integral path becomes collinear to the tangent subspace, a tangent bifurcation point has occurred. Generally, this shows up as a 'jump' in the path to a point a considerable distance away on a neighboring level curve. Between these two points the value of the constrained minimum defining the path is degenerate and H has repeated eigenvalues.

Finding Integral Path Pitchfork Bifurcations

When flattening occurs but the flow of the integral path is not collinear to the tangent subspace of the level constraint, an eigenvalue exchange is sought. This exchange in the minimum eigenvalue of H from one associated with $J^T F$ to one associated with the tangent subspace of a level curve is easily determined by monitoring the eigenvalue associated with the terrain path and the smallest eigenvalue of the matrix H projected onto the tangent subspace. Once an eigenvalue exchange is detected, the algorithm searches for a possible bifurcation point by locating a maximum in the norm of $J^T F$ on the level curve, say L^*, where the eigenvalue exchange has been detected. This is because as contours flatten, the distance between these level curves becomes smaller and smaller, which is an indication that the nature of $||J^T F||$ on L^* has changed from a constrained minimum to a constrained maximum. See the discussion in [7]. Therefore, an approximate bifurcation point is calculated by solving the NLP problem

$$\max g^T g \text{ such that } F^T F = L^* . \qquad (22)$$

Note that Eq. (22) is very similar to Eq. (11). Thus the numerical methodology needed to solve Eq. (22) already exists in the form of the corrector algorithm. However, it is important to note that predictor iterates rarely land exactly on the contour corresponding to a pitchfork bifurcation because finite step sizes are used in the predictor-corrector calculations. They generally land close and thus the solution to Eq. (22) is usually a very good approximation of the bifurcation point – since all that is really needed to follow all branches of a pitchfork bifurcation is knowledge at a point following the eigenvector exchange. Moreover, because contours in the neighborhood of a pitchfork bifurcation point can be very flat, solving Eq. (22) can be challenging in some cases. Extreme flatness creates numerical problems because it implies that the Kuhn–Tucker conditions for Eq. (22) have a near singular coefficient matrix. Therefore, good step size control should be used when solving Eq. (22).

Finding All Branches Associated with a Bifurcation Point

Once a bifurcation point is located, all branches from the bifurcation must be followed in order to increase the probability of finding all relevant solution information. Locating these branches is reasonably straightforward. Tangent bifurcation points are characterized by collinearity and provide only a single branch for further exploration that, as noted, manifests itself by a 'jump' to a widely different point on a neighboring level curve. Pitchfork bifurcation points, on the other hand, provide three branches of further exploration defined by the gradient to the level curve L^*, and $+/-$ the 'minimum' eigenvector of H projected onto the tangent subspace at the bifurcation on L^*. Each of these vectors is easily computed. The gradient vector at a bifurcation, which corresponds to the middle part of the pitchfork, is a readily available byproduct of the calculations. The 'minimum' eigenvector of H on the tangent subspace at L^* is also easily determined. What is difficult is locating the valleys that correspond to the pair of minima of $g^T g$ on L^*. For this a careful initialization of our corrector algorithm is required to solve Eq. (11) with $L = L^*$.

Cases

There are several problem cases that are encompassed by the global terrain-following formulations and methods presented in earlier sections. These cases include

1) Nonlinear objective functions with simple bounds on variables.
2) Systems of nonlinear algebraic equations.
3) Nonlinear objective functions with simple bounds and linear constraints.

1) Nonlinear Objective Functions with Simple Bounds

This is the case on which the developments in the formulation and methods sections are based and no further discussion is necessary.

2) Systems of Nonlinear Algebraic Equations

For a system of algebraic equations, $F = 0$ is usually given and φ is irrelevant. The function of interest becomes the traditional nonlinear least squares function, $F^T F$, and the terrain methodology follows the strategies outlined in previous sections.

3) Nonlinear Objective Functions with Simple Bounds and Linear Constraints

Nonlinear programming problems that involve linear equality constraints are easily handled by global terrain methods by using the linear constraints to eliminate optimization variables. For m linear constraints, m optimization variables can be eliminated. However, it is important to understand that the gradient and Hessian matrix of φ must be adjusted to accommodate this variable elimination. This can be done by either using projection methods or by explicitly doing the elimination before formulating the optimization problem to be solved by the terrain methodology.

If projection is used then F is replaced by $P^T F$, where P is the $n \times m$ orthonormal projection matrix whose columns are orthogonal to all rows of the Jacobian matrix of the linear constraints. That is, if J_{LEQ} is the $m \times n$ Jacobian of the m linear equality constraints, then the projection matrix P satisfies $J_{LEQ} P = 0$. Additionally, the Hessian matrix of ϕ, J, must reflect implicit elimination and is easily computed to be $P^T J P$. These projections of F and J permit the use of the terrain methodology in R^{n-m} while still allowing any bounds on all variables to be enforced.

References

1. Baker J (1986) An algorithm for the location of transition states. J Comput Chem 7:385–395
2. Cerjan CJ, Miller WH (1981) On finding transition states. J Chem Phys 75:2800–2806
3. Diener I (1987) On the global convergence of path-following methods to determine all solutions to a system of nonlinear equations. Math Prog 39:181–188
4. Jongen HT, Stein O (2004) Constrained global optimization: adaptive gradient flows. In: Floudas CA, Pardalos P (eds) Frontiers in Global Optimization. Kluwer Acad, Boston
5. Lucia A, DiMaggio PA, Depa P (2004) A geometric methodology for global optimization. J Global Optim 29:297–314
6. Lucia A, Yang F (2002) Global terrain methods. Comput Chem Eng 26:529–546
7. Lucia A, Yang F (2003) Multivariable terrain methods. AIChE J 49:2553–2563
8. Page M, McIver JW (1988) On evaluating the reaction path Hamiltonian. J Chem Phys 88:922–935

Graph Coloring

GC

JUE XUE
Department Management Sci.,
City University Hong Kong, Kowloon, Hong Kong

MSC2000: 90C35

Article Outline

Keywords

Graph; Coloring; Optimization; Approximation; Algorithms

A graph $G = (V, E)$ consists of a *vertex set* V and an *edge set* $E \subseteq V \times V$. If $e = (i, j)$ ($\in E$) is an edge of G, then e is *incident* to i and j, and i and j are *adjacent*. Similarly, if two edges are incident to the same vertex, they are adjacent.

A vertex coloring of $G = (V, E)$ is an assignment of k colors to members of V (a *coloring*) so that adjacent vertices have different colors (G is *k-colorable*). The *graph coloring problem* (GC) is to find the minimum number k such that G is k-colorable.

When a positive integer weight w_i is associated with every $i \in V$ and a color assignment satisfies:

- every vertex i gets w_i different colors,
- $\forall (i, j) \in E$, i and j get $w_i + w_j$ different colors,

then this color assignment is a *weighted coloring*. The *weighted graph coloring problem* asks for the minimum number of colors needed for a weighted coloring of G.

An *edge coloring* and a *total coloring* of a given graph can be defined in a similar way:

- An edge coloring assigns colors to edges so that adjacent edges have different colors.
- A total coloring assigns colors to vertices and edges so that any pair of adjacent vertices, adjacent edges, and a vertex and any incident edge will have different colors.

The *edge coloring problem* or the *total coloring problem* asks for the minimum number of colors needed for an edge coloring or a total coloring, respectively [12,28,30]. Although the weighted graph coloring, edge coloring, and total coloring problems seem different from GC, they can be transformed into a GC [33,42]. Further generalizations of GC tend to change the structure of a coloring solution, and they move closer to other well-known combinatorial optimization problems [16,37].

GC is well-known in graph theory and combinatorial optimization. It starts with the famous four-coloring conjecture [24,38] which says four colors are enough to color any geographic map so that every country gets a color different from those used by its neighbors. Although the four-coloring conjecture is now considered a theorem [1,2], the process to prove or disprove it has inspired many interesting questions [32], and has helped the development of several branches of science, for example, the GC and the graph theory [27]. The interest in GC also comes from its vast number of applications in solving real world problems. For example, GC can be used to model problems in timetabling, scheduling, computer science, information systems, telecommunications, and other industrial applications [9,11,39]. Typically, a graph is constructed with its vertices representing items of interest and edges representing some undesirable binary relationship.

GC has several mathematical programming formulations. For example, one can use an integer variable $x_{ik} = 1$ to indicate when a vertex i is colored by k, and $x_{ik} = 0$ otherwise. One can also use an integer variable $y_k = 1$ to indicate color k is assigned to at least one vertex of G, and $y_k = 0$ otherwise. Then, the solution to the following mathematical programming problem provides an optimal (minimum) coloring of G:

$$\begin{cases} \min & \displaystyle\sum_{k=1}^{|V|} y_k \\ \text{s.t.} & \displaystyle\sum_{k=1}^{|V|} x_{ik} = 1, \quad \forall i \in V, \\ & x_{ik} + x_{jk} \le 1, \quad \forall (i, j) \in E, \\ & y_k \ge x_{ik}, \quad y_k, x_{ik} \in \{0, 1\}, \\ & \quad \forall i \in V,\ k = 1, \ldots, |V|, \end{cases}$$

where $|V|$ is the cardinality of the set V. In this problem, the objective function equals the number of colors used. The constraints ensure that every vertex is colored, that no adjacent vertices get the same color, and that the counting of used colors is correct.

For a feasible coloring, one can group the vertices into subsets based on their colors. Thus, vertices of each subset will be mutually nonadjacent. Such a subset of vertices is called a *stable set*, a *color class*, or an *independent set* [5,8,35,41]. Using the concept of a stable set, one can formulate GC as a set partitioning problem.

Let $S_1, \ldots, S_t$ be all the stable sets of G. Let A_S be a $0 - 1$ matrix whose rows are the characteristic vectors of the S_js. One can use a variable $s_j = 1$ to indicate that all members of S_j have the same color, and $s_j = 0$ otherwise. Then the solution to the following problem also provides an optimal (minimum) coloring of G:

$$\begin{cases} \min & \displaystyle\sum_{j=1}^{t} s_j \\ \text{s.t.} & sA_S = \vec{1}, \\ & s_j \in \{0, 1\}, \quad j = 1, \ldots, t, \end{cases}$$

where $s = (s_1, \ldots, s_t)$, and $\vec{1} = (1, \ldots, 1)$ is of dimension $|V|$.

Other mathematical formulations of GC based on quadratic programming, semidefinite programming etc. are also available. Different formulations have their own distinctive advantages in understanding the problem structure and in designing solution methods to solve the problem [22,33,34].

Checking whether G is k-colorable for an arbitrary integer k is an *NP*-complete problem [14,23]. It remains *NP*-complete even for fixed $k \geq 3$ [14,40]. Therefore, it is unlikely that the solution time of GC can be bounded by any polynomial function (*polynomial time*) [13]. However, GC can be solved in polynomial time for graphs of some special structures. For example, polynomial algorithms exist for perfect graphs, Meyniel graphs, and triangulated graphs [3,4,15,17,19,41].

Let us define the *performance guarantee* of an approximation method to be the worst ratio between the approximation solution value and the corresponding optimal solution value over all graphs of size $|V|$. Then, $O(|V| \log |V|)$ seems to be the first performance guarantee provided by a polynomial time GC heuristic [20]. This performance guarantee has being improved over the years. Let k be the optimal (minimum) number of colors needed to color a graph, and let Δ be the *maximum degree* (number of edges incident to a vertex) among all vertices. The two recent performance guarantees achieved by polynomial approximation algorithms for GC are $O(|V|(\log \log|V|)^2/(\log |V|)^3)$ and $\min\{O(\Delta^{1-2/k}), O(|V|^{1-3/(k+1)})\}$ [18,22]. On the other hand, it is known that unless $P = NP$, it is *NP*-hard to approximate an optimal graph coloring within a performance guarantee of $O(|V|^{\epsilon})$, $\epsilon > 0$ [14,15].

Available solution methods for GC can be divided into approximation algorithms and exact algorithms. These methods find a feasible graph coloring and an optimal graph coloring, respectively [29].

A popular way to find an approximation solution to GC is the *sequential greedy coloring heuristic* (SGCH). In a SGCH, the vertices are ordered in a sequence and are colored one at a time according to the sequence. Every vertex is colored by the smallest (first) feasible color. It is not hard to see that the initial vertex sequence decides the resulting graph coloring of a SGCH.

It is also known that there exists at least one sequence under which a SGCH will find an optimal coloring. However, finding an optimal vertex sequence is *NP*-hard. Extensive work aimed at finding 'good' vertex sequences can be found in the literature [10,32]. Once a feasible coloring is available, further improvement can be made using various methods, including: interchange, iterative improvement, and other searching techniques (such as simulated annealing and tabu search) [36].

To date, the most popular and efficient way to find an optimal solution to GC is through a *branch and bound* (BB), or *implicit enumeration*, algorithm. A BB algorithm typically consists of two parts: the *forward phases* and the *backtrack phases*. A forward phase starts from a partial coloring (e. g. Ø) and colors the remaining vertices to find a feasible graph coloring. For example, a SGCH can be used in place of a forward phase. A backtrack phase will decide the starting point of the next forward phase so that an alternative feasible graph coloring can be found.

Now let us consider how a simple BB algorithm [7] finds an optimal coloring of $G = (V, E)$. Let UB be the value of a current best coloring (initially set $UB = \infty$). Suppose the first forward phase applies a SGCH to vertex sequence $(v_1, \ldots, v_{|V|})$ and finds a feasible coloring of G. The number of colors used by the feasible coloring will be the new UB. Apparently, UB is an upper bound on the value of any feasible coloring that one needs to search for.

Since SGCH assigns the smallest feasible color to every vertex, a backtrack phase can be carried out by scanning the vertices in the reverse order of $(v_1, \ldots, v_{|V|})$. That is, finding the first vertex v_j that can be recolored by an alternative feasible color $< UB$, not used for v_j before. The new forward phase will start from the partial coloring of $\{v_1, \ldots, v_{j-1}\}$ and applies a SGCH to $(v_j, \ldots, v_{|V|})$, up to a v_i whose smallest feasible color is UB, or to $v_{|V|}$ that has a feasible color $< UB$. In the latter case, a better coloring is found. Then the BB algorithm will backtrack and repeat the above until it backtracks to vertex v_1 (the algorithm terminates).

Various improvement measurements are designed and tested for the above basic BB method. They include 'look ahead', 'dynamic reordering', choosing an appropriate feasible color (instead of the 'smallest') to color a vertex, using tighter lower and upper bounds, and a column generation approach [6,21,26,31,33]. These improvements have greatly reduced the search tree size and enhanced our ability to solve GC optimally. The state-of-the-art method for solving GC on randomly generated graphs seems to be limited to graphs of 100 vertices [21,31,42,43].

See also

- ▶ Adaptive Simulated Annealing and its Application to Protein Folding
- ▶ Branch and Price: Integer Programming with Column Generation
- ▶ Decomposition Techniques for MILP: Lagrangian Relaxation
- ▶ Feedback Set Problems
- ▶ Frequency Assignment Problem
- ▶ Generalized Assignment Problem
- ▶ Genetic Algorithms
- ▶ Global Optimization in Lennard–Jones and Morse Clusters
- ▶ Global Optimization in Protein Folding
- ▶ Graph Planarization
- ▶ Greedy Randomized Adaptive Search Procedures
- ▶ Heuristics for Maximum Clique and Independent Set
- ▶ Integer Linear Complementary Problem
- ▶ Integer Programming
- ▶ Integer Programming: Algebraic Methods
- ▶ Integer Programming: Branch and Bound Methods
- ▶ Integer Programming: Branch and Cut Algorithms
- ▶ Integer Programming: Cutting Plane Algorithms
- ▶ Integer Programming Duality
- ▶ Integer Programming: Lagrangian Relaxation
- ▶ LCP: Pardalos–Rosen Mixed Integer Formulation
- ▶ Maximum Constraint Satisfaction: Relaxations and Upper Bounds
- ▶ Mixed Integer Classification Problems
- ▶ Molecular Structure Determination: Convex Global Underestimation
- ▶ Monte-Carlo Simulated Annealing in Protein Folding
- ▶ Multi-objective Integer Linear Programming
- ▶ Multi-objective Mixed Integer Programming
- ▶ Multiparametric Mixed Integer Linear Programming
- ▶ Multiple Minima Problem in Protein Folding: αBB Global Optimization Approach
- ▶ Packet Annealing
- ▶ Parametric Mixed Integer Nonlinear Optimization
- ▶ Phase Problem in X-ray Crystallography: Shake and Bake Approach
- ▶ Protein Folding: Generalized-ensemble Algorithms
- ▶ Quadratic Assignment Problem
- ▶ Quadratic Semi-assignment Problem
- ▶ Set Covering, Packing and Partitioning Problems
- ▶ Simplicial Pivoting Algorithms for Integer Programming
- ▶ Simulated Annealing Methods in Protein Folding
- ▶ Stochastic Integer Programming: Continuity, Stability, Rates of Convergence
- ▶ Stochastic Integer Programs
- ▶ Time-dependent Traveling Salesman Problem

References

1. Appel K, Haken W (1977) Every planar map is four colorable. Part 1: Discharging. Illinois J Math 21:429–490
2. Appel K, Haken W (1977) Every planar map is four colorable. Part 2: Reducibility. Illinois J Math 21:491–567
3. Balas E (1986) A fast algorithm for finding an edge-maximal subgraph with a TR-formative coloring. Discrete Appl Math 15:123–134
4. Balas E, Xue J (1991) Minimum weighted coloring of triangulated graphs with application to maximum weight vertex packing and clique finding in arbitrary graphs. SIAM J Comput 20:209–221
5. Balas E, Xue J (1996) Weighted and unweighted maximum clique algorithms with upper bounds from fractional coloring. Algorithmica 15:397–412
6. Brelaz D (1979) New methods to color vertices of a graph. Comm ACM 22:251–256
7. Brown JR (1972) Chromatic scheduling and the chromatic number problem. Managem Sci 19:456–463
8. Carraghan R, Pardalos PM (1990) An exact algorithm for the maximum clique problem. Oper Res Lett 9:375–382
9. de Werra D (1985) An introduction to timetabling. Europ J Oper Res 19:151–162
10. de Werra D (1990) Heuristics for graph coloring. Computing 7:191–208
11. de Werra D, Gay Y (1994) Chromatic scheduling and frequency assignment. Discrete Appl Math 49:165–174
12. Fiorini S, Wilson RJ (1977) Edge-coloring of graphs. Res Notes Math, vol 16. Pitman, Boston, MA
13. Garey MR, Johnson DS (1979) Computers and intractability: A guide to the theory of NP-completeness. Freeman, New York
14. Garey MR, Johnson DS, Stockmeyer I (1976) Some simplified NP-complete graph problems. Theoret Comput Sci 1:237–267
15. Gavril F (1972) Algorithms for coloring maximum clique: minimum covering by cliques, and maximum independent set of a chordal graph. SIAM J Appl Math 1:181–187

16. Gionfriddo M (1979) A short survey of some generalized colorings of graphs. Ars Combin 21:295–322
17. Grötschel M, Lovász L, Schrijver A (1989) Polynomial algorithms for perfect graphs. Ann Discret Math 21:325–356
18. Halldórsson MM (1993) A still better performance guarantee for approximate graph coloring. Inform Process Lett 45:19–23
19. Hertz A (1990) A fast algorithm for coloring Meyniel graphs. J Combin Th B 50:231–240
20. Johnson DS (1974) Worst-case behavior of graph-coloring algorithms. In: Proc. 5th Southeastern Conf. Combinatorics: Graph Theory and Computing, Winnipeg, pp 513–528
21. Johnson DS, Trick M (eds) (1996) Cliques, coloring, and satisfiability: Second DIMACS implementation challenge. DIMACS. Amer. Math. Soc., Providence, RI
22. Karger D, Motwani R, Sudan M (1994) Approximate graph coloring by semidefinite programming. In: 35th Annual Symp. Foundations of Computer Sci., IEEE, New York, pp 2–13
23. Karp RM (1972) Reducibility among combinatorial problems. In: Miller RE, Thatcher JW (eds) Complexity of Computer Computations, Plenum, New York, pp 85–104
24. Kempe AB (1879) On the geographical problem of four colours. Amer J Math 2:193–200
25. Khanna S, Linial N, Safra S (1993) On the hardness of approximating the chromatic number. Proc. 2nd Israel Symp. Theory of Computing and Systems, IEEE, New York, pp 250–260
26. Korman SM (1979) The graph-colouring problem. In: Christofides N, Mingozzi A, Toth P, Sandi C (eds) Combinatorial Optimization. Wiley, New York, pp 211–235
27. Kubale M (1991) Graph coloring. In: Kent A, Williams JG (eds) Encycl. Microcomputers, vol 8. M. Dekker, New York, pp 47–69
28. Lee J, Leung J (1993) A comparison of two edge-coloring formulations. Oper Res Lett 13:215–223
29. Matula DW, Marble G, Isaacson D (1972) Graph coloring algorithms. In: Reed R (ed) Graph theory and computing. Academic Press, New York, pp 109–122
30. McDiarmid C, Sanchezarroyo A (1993) On total colorings of graphs. J Combin Th B 57:122–130
31. Mehrotra A, Trick MA (1995) A column generation approach for graph coloring. Techn Report GSIA Carnegie-Mellon Univ
32. Nelson R, Wilson RJ (eds) (1990) Graph colorings. Longman, Harlow
33. Pardalos PM, Mavridou T, Xue J (1998) The graph coloring problem: A bibliographic survey. In: Du D-Z and Pardalos PM (eds) Handbook Combinatorial Optim, vol 2. Kluwer, Dordrecht, pp 331–395
34. Pardalos PM, Wolkowicz H (1994) Quadratic assignment and related problems. DIMACS. Amer. Math. Soc., Providence, RI
35. Pardalos PM, Xue J (1994) The maximum clique problem. J Global Optim 3:463–482
36. Rayward-Smith VJ, Osman IH, Reeves CR, Smith GD (eds) (1996) Modern heuristic search methods. Wiley, New York
37. Roberts FS (1995) From garbage to rainbows: generalizations of graph coloring and their applications. In: Alavi Y, Schwenk AJ (eds) Graph Theory, Combinatorics, and Algorithms, vol 2. Wiley, New York, pp 1031–1052
38. Saaty TL, Keinen PC (1977) The four-color problem. McGraw-Hill, New York
39. Schmidt G, Ströhleim T (1980) Timetable construction - an annotated bibliography. Comput J 23:307–316
40. Stockmeyer L (1973) Planar 3-colorability is polynomial complete. ACM SIBACT News 5:19–25
41. Xue J (1994) Edge-maximal triangulated subgraphs and heuristics for the maximum clique problem. Networks 24:109–120
42. Xue J (1998) Solving the minimum weighted integer coloring problem. J Comput Optim Appl 11:53–64
43. Xue J, Liu J A network flow based lower bound for the minimum weighted integer coloring problem. Inform Process Lett (to appear).

Graph Planarization

MAURICIO G.C. RESENDE[1], CELSO C. RIBEIRO[2]
[1] Information Sci. Res., AT&T Labs Res.,
Florham Park, USA
[2] Department Computer Sci., Catholic University Rio de Janeiro, Rio de Janeiro, Brazil

MSC2000: 94C15, 90C10, 90C27

Article Outline

Keywords

Planarization; Graph; Planar graph; Heuristics; Exact algorithms

A graph is said to be *planar* if it can be drawn on the plane in such a way that no two of its edges cross. Given a graph $G = (V, E)$ with vertex set V and edge set E, the objective of *graph planarization* is to find a minimum cardinality subset of edges $F \subseteq E$ such that the graph $G' = (V, E \setminus F)$, resulting from the removal of the edges in F from G, is planar. This problem is also known as the *maximum planar subgraph* problem. A related and simpler problem is that of finding a *maximal planar subgraph*, which is a planar subgraph $G' = (V, E')$ of G such that the addition of any edge $e \in E \setminus E'$ to G' destroys its planarity.

Graph planarization is known to be *NP*-hard [21]. The proof of *NP*-completeness of its decision version is based on a transformation from the Hamiltonian path problem restricted to bipartite graphs. Although exact methods for solving the maximum planar subgraph problem have been recently proposed, most algorithms to date attempt to find good approximate solutions.

In this article, we survey graph planarization and related problems. In the next section, we describe variants and applications of the basic problem formulated above. Next, we describe the branch and cut algorithm of M. Jünger and P. Mutzel [16]. We then review work on heuristics based on planarity testing and those based on two- phase procedures. Finally, computational results are considered.

Variants and Applications

An application of graph planarization arises in the design of integrated circuits, in which a graph describing the circuit has to be decomposed into a minimum number of layers, each of which is a planar graph [19]. Other applications arise from variants of the basic graph planarization problem.

One such variant is the *maximum weighted planar graph* problem, in which positive weights are associated with the edges of the graph and one seeks a planar subgraph of maximum weight. Note that the basic graph planarization problem is a special case of the maximum weighted planar graph problem, in which all edge weights are equal to one. An application of this problem to *facility layout* is described in [13]. A graph is built in which the vertices represent the facilities and the edges define the relationships between them. The weight of each edge is the desirability that the two facilities that define the edge be adjacent in the design. A maximum weighted planar subgraph corresponds to a feasible layout with maximum benefit. In this paper, the authors also propose simulated annealing and tabu search heuristics for the approximate solution of the maximum weighted planar graph problem. Constructive heuristics based on maintaining a triangulated subgraph while making node and edge insertions are given in [8,11], and [20].

Another related variant is that of drawing a given graph such that the number of *edge crossings* is minimized. The *crossing number* problem has practical applications in *circuit design* and graph drawing, such as in CASE tools [27] and automated graphical display systems. One particular case is that of minimizing straight-line crossings in layered graphs. A GRASP and path relinking approach for the two-layer case is given in [17], where one can also find a survey of the literature. Algorithms for graph drawing are reviewed in [6].

In the *planar augmentation* problem, one wants to determine the minimum number of edges that need to be added to a planar graph such that the resulting graph is still planar and at least k-connected, where k is usually fixed to two or three. This variant has applications in automatic *graph drawing*, as well as in the design of *survivable networks* [24].

An Exact Algorithm

An exact branch and bound algorithm for the weighted graph planarization problem was introduced in [10], but was limited to small dense graphs. Only recently (1999) has there been a leap in the performance of exact methods for graph planarization with the *Jünger–Mutzel branch and cut algorithm* [16], which we describe next.

Given a graph $G = (V, E)$, their approach uses facet-defining inequalities for the planar subgraph polytope $\mathcal{PLS}(G)$. Let x_e be a 0–1 variable associated with each edge $e \in E$, such that $x_e = 1$ if and only if edge e appears in the maximum planar subgraph of G. Furthermore, let $x(F) = \sum_{e \in F} x_e$, for $F \subseteq E$.

Trivial inequalities $0 \leq x_e \leq 1$ are implicitly handled by the linear programming (LP) solver. The inequality $x(E) \leq 3|V| - 6$ is added to the initial linear program. Let x be the optimal solution of the LP relaxation associated with some node of the enumeration tree. For $0 \leq \epsilon \leq 1$, let $E_\epsilon = \{e \in E\} \ x_e \geq 1 - \epsilon\}$ and consider the graph $G_\epsilon = (V, E_\epsilon)$, to which the

Hopcroft–Tarjan planarity-testing algorithm [14] is applied. The algorithm stops if it finds an edge set F which induces a nonplanar graph in G. If the inequality $x(F) \leq |F| - 1$ is violated, it is added to the set of constraints of the current LP. The back edge of the path which proved the nonplanarity of the graph induced in G by F is removed and the planarity-testing algorithm proceeds, eventually identifying other forbidden subgraphs of the graph G_ϵ. Although these forbidden subgraphs do not necessarily define facets of $\mathcal{PLS}(G)$, they must contain facet-defining subgraphs. Facet-defining inequalities are identified as follows. Once a forbidden set F is found, where the inequality $x(F) \leq |F| - 1$ is violated, one successively deletes each edge $f \in F$ and applies the planarity-testing algorithm. If the graph induced by $F \setminus \{f\}$ is planar, then edge f is returned to F. In at most $|F|$ steps, F is reduced to a smaller edge set which induces a minimal planar subgraph, leading to the facet-defining inequality $x(F) \leq |F| - 1$ still violated by the current LP solution. Another simple heuristic searches for violated Euler facet-defining inequalities $x(F) \leq 3|V'| - 6$ or $x(F) \leq 2|V'| - 4$, where (V', F) is, respectively, a clique or a complete bipartite subgraph of G.

After an LP has been solved, its solution is exploited by the planarity-testing algorithm, to produce a feasible solution for the graph planarization problem. Such feasible solutions are used as lower bounds that are used not only for fathoming nodes in the branch and cut tree, but also for fixing variables using their reduced costs during a cutting plane phase. Other heuristics are implemented to enhance the practical performance of the algorithm.

Branching is done if no cutting plane has been found for the current infeasible solution. The variable chosen for branching is one with fractional value closest to 1/2, among those with maximum cost coefficient in the objective function.

Heuristics Based on Planarity Testing

The first linear time algorithm for planarity testing was proposed by J. Hopcroft and R.E. Tarjan [14]. T. Chiba, I. Nishioka and I. Shirakawa [4] used the basic ideas of this approach to devise an algorithm for finding a maximal planar subgraph of $G = (V, E)$ with time complexity $O(|V||E|)$. Later, J. Cai, X. Han and Tarjan [3] proposed another version of the above planarity testing algorithm. This new algorithm is based on processing edges instead of paths. It leads to another algorithm to find a maximal planar subgraph, with improved $O(|\mathrm{E}| \log |V|)$ time complexity.

A. Lempel, S. Even and I. Cederbaum [18] have proposed another approach to planarity testing. Although its original complexity was $O(|V|^2)$, K. Booth and G. Lueker [2] have shown that it can be implemented in linear time using *PQ*-trees. A few algorithms for finding a maximal planar subgraph based on this planarity testing approach have been proposed in the literature. However, Jünger, S. Leipert and Mutzel [15] show that attempts following this strategy are forced to fail.

Another approach for finding a maximal planar subgraph of a given graph works as follows. Start with an empty subgraph and successively add the edges of the original graph, whenever such addition maintains the planarity of the subgraph under construction. Using any of the planarity testing algorithms above described, such approach can be implemented in $O(|V||E)$ time complexity. An incremental planarity testing algorithm, based on an $O(\log|V)$ time-per-operation strategy for the problem of maintaining a planar graph under edge additions, was proposed by G. Di Battista and R. Tamassia [7]. Hence, their algorithm leads to a more efficient implementation of the incremental approach for finding a maximal planar subgraph with $O(|E| \log |V|)$ time complexity.

Two-Phase Heuristics

The heuristics described in this section are based on the separation of the computation into two phases. The first phase consists in devising a linear permutation of the nodes of the input graph, followed by placing them along a line. The second phase determines two sets of edges that may be represented without crossings above and below that line, respectively. Y. Takefuji and K.C. Lee [25] were the first to propose a heuristic using this idea. They use an arbitrary sequence of nodes in the first phase and apply a parallel heuristic using a neural network for the second phase. Takefuji, Lee, and Y.B. Cho [26] claimed superior performance of the two-phase approach of Takefuji and Lee [25] with respect to the heuristics described in the previous section.

Their approach was later extended and improved by O. Goldschmidt and A. Takvorian [12]. In the first phase, these authors attempt to use a linear permutation of the nodes associated with an Hamiltonian cycle of G. Two strategies are used:

i) a randomized algorithm [1] that almost certainly finds a Hamiltonian cycle if one exists; and

ii) a greedy deterministic algorithm that seeks a Hamiltonian cycle.

In the latter, the first node in the linear permutation is a minimum degree node in G. After the first k nodes of the permutation have been determined, say $v_1, \ldots, v_k$, the next node v_{k+1} is selected from the nodes adjacent to v_k in G having the least adjacencies in the subgraph G_k of G induced by $V \backslash \{v_1, \ldots, v_k\}$. If there is no node of G_k adjacent to v_k in G, then v_{k+1} is selected as a minimum degree node in G_k.

Let $H = (E, I)$ be a graph where each of its nodes corresponds to an edge of the input graph G. Nodes e_1 and e_2 of H are connected by an edge if the corresponding edges of G cross with respect to the linear permutation of the nodes established during the first phase. A graph is called an *overlap graph* if its nodes can be placed in one-to-one correspondence with a family of intervals on a line. Two intervals are said to *overlap* if they cross and none is contained in the other. Two nodes of the overlap graph are connected by an edge if and only if their corresponding intervals overlap. Hence, the graph H as constructed above is the overlap graph associated with the representation of G defined by the linear permutation of its nodes.

The second phase of the heuristic of Goldschmidt and Takvorian consists in two-coloring a maximum number of the nodes of the overlap graph H, such that each of the two color classes $\mathcal{B}$ (blue) and $\mathcal{R}$ (red) forms an independent set. Equivalently, the second phase seeks a *maximum bipartite subgraph* of the overlap graph H, i. e. a bipartite subgraph having the largest number of nodes. This problem is equivalent to drawing the edges of the input graph G above or below the line where its nodes have been placed, according to their linear permutation. A greedy algorithm is used to construct a maximal bipartite subgraph of the overlap graph. This algorithm finds a maximum independent set $\mathcal{B} \subseteq E$ of the overlap graph $H = (E, I)$, reduces the overlap graph by removing from it the nodes in $\mathcal{B}$ and all edges incident to nodes in $\mathcal{B}$, and then finds a maximum independent set $\mathcal{R} \subseteq E \backslash \mathcal{B}$ in the remaining overlap graph $H' = (E \backslash \mathcal{B}, I')$. The two independent sets so obtained induce a bipartite subgraph of the original overlap graph, not necessarily with a maximum number of nodes.

The linear permutation obtained in the first phase affects the size of the planar subgraph found in the second phase of the above heuristic. Moreover, it is not clear that the permutation produced by the greedy algorithm is the best. To produce possibly better permutations, *randomization* and *local search* have been introduced in the *greedy algorithm* by M.G.C. Resende and C.C. Ribeiro [22] in the form of a *greedy randomized adaptive search procedure* (GRASP).

A GRASP [9] is an iterative process, in which each iteration consists of two phases: construction and local search. The construction phase builds a feasible solution, whose neighborhood is explored by local search. The best solution over all GRASP iterations is returned as the result.

In the construction phase, a feasible solution is built, one element at a time. At each construction iteration, the next element to be added is determined by ordering all elements in a candidate list with respect to a greedy function that estimates the benefit of selecting each element. The adaptive component of the heuristic arises from the fact that the benefits associated with every element are updated at each iteration of the construction phase to reflect the changes brought on by the selection of the previous elements. The probabilistic component of a GRASP is characterized by randomly choosing one of the best candidates in the list, but usually not the top candidate. This way of making the choice allows for different solutions to be obtained at each iteration, but does not necessarily jeopardize the power of GRASP's adaptive greedy component.

The solutions generated by a GRASP construction are not guaranteed to be locally optimal, even with respect to simple neighborhood definitions. Hence, it is almost always beneficial to apply a local search to attempt to improve each constructed solution. A local search algorithm works in an iterative fashion by successively replacing the current solution by a better solution from its neighborhood.

Resende and Ribeiro [22] proposed an extension of the above described heuristic of Goldschmidt and Takvorian, in which a GRASP is used for finding a linear permutation of the nodes. In the construction phase of this GRASP, the greedy algorithm used in the first phase by Goldschmidt and Takvorian is randomized: instead of selecting the node of minimum degree among those yet unselected, the selection is made from a set of low degree nodes. The local search phase of this GRASP explores the neighborhood of the current permutation by swapping the positions of two nodes at a time, attempting to reduce the number of possible edge crossings.

Incorporating the second phase of the Goldschmidt–Takvorian heuristic to the above GRASP for finding a linear permutation of the nodes results in a GRASP for graph planarization.

Each iteration of this GRASP produces three edge sets: $\mathcal{B}$ (blue edges), $\mathcal{R}$ (red edges), and $\mathcal{P}$ (the remaining edges, which are referred to as the *pale edges*). By construction, $\mathcal{B}$, $\mathcal{R}$, and $\mathcal{P}$ are such that no red or pale edge can be colored blue. Likewise, pale edges cannot be colored red. However, if there exists a pale edge $p \in \mathcal{P}$ such that all blue edges that cross with p (let $\widehat{\mathcal{B}}_p \subseteq \mathcal{B}$ be the set of those blue edges) do not cross with any red edge $r \in \mathcal{R}$, then all blue edges $b \in \widehat{\mathcal{B}}_p$ can be colored red and p can be colored blue. In case this reassignment of colors is possible, then the size of the planar subgraph is increased by one edge. This post-optimization procedure is incorporated at the end of each GRASP iteration.

Computational Results

Detailed results on a set of 75 test problems described in the literature [5,12] are reported in [22]. The description of the code used can be found in [23]. Here, we summarize computational results illustrating the effectiveness of the two-phase heuristics described in the previous section, as well as that of the exact branch and cut algorithm. These results are based on a Fortran implementation of the GRASP heuristic of Resende and Ribeiro [22], on the original code of the branch and cut algorithm of Jünger and Mutzel [16], and on published results for the heuristics of Takefuji and Lee [25] and Goldschmidt and Takvorian [22] (using the greedy algorithm for building the linear permutation of the nodes).

We give, in the table below, results comparing the four approaches on a subset of the test problems described in [12]. For each instance, the table lists the number of nodes, the number of edges, and the size of the planar subgraphs produced by each algorithm. A time limit of 1000 seconds (on a SUN SPARCstation 10/41) was imposed on the runs of the branch and cut algorithm and the best solution found was returned as a heuristic solution when optimality was not attained in that time limit. This time limit was reached on instances G12–G19.

The results in this table show that the Goldschmidt–Takvorian algorithm is a substantial improvement over the neural network approach of Takefuji and Lee. The GRASP consistently outperforms both other two-phase heuristics, not only for the problems reported in this table, but also for all of the remaining instances considered in [22].

Problem	Nodes	Edges	T-L	G-T	R-R	J-M
G1	10	22	20	20	20	20
G2	45	85	80	80	82	82
G3	10	24	21	21	24	24
G4	10	25	22	21	24	24
G5	10	26	22	21	24	24
G6	10	27	22	21	24	24
G7	10	34	23	22	24	24
G8	25	69	58	60	69	69
G9	25	70	59	60	69	69
G10	25	71	58	59	69	69
G11	25	72	60	59	69	69
G12	25	90	61	62	67	68
G13	50	367	70	131	135	125
G14	50	491	100	136	143	133
G15	50	582	101	142	144	138
G16	100	451	92	180	196	187
G17	100	742	116	219	236	213
G18	100	922	115	237	246	223
G19	150	1064	127	297	311	290

A comparison of GRASP with the branch and cut algorithm depends heavily on the instances. The results reported in [22] can be separated into two groups. On 49 of the 55 instances in the first group, the GRASP either matched or produced better solutions than the branch and cut algorithm. On 30 of those 55 instances, the GRASP solution was strictly better than the branch and cut solution. Note that, on these instances, the branch and cut algorithm was forced to stop because of the 1000 second time limit. However, on all the remaining 20 instances, the branch and cut algorithm performs remarkably well and outperforms all other algorithms.

See also

- ► Feedback Set Problems
- ► Generalized Assignment Problem
- ► Graph Coloring
- ► Greedy Randomized Adaptive Search Procedures
- ► Optimization in Leveled Graphs
- ► Quadratic Assignment Problem
- ► Quadratic Semi-assignment Problem

References

1. Angluin D, Valiant LG (1979) Probabilistic algorithms for Hamiltonian circuits and matchings. J Comput Syst Sci 18:155–190
2. Booth K, Lueker G (1976) Testing for the consecutive ones property, interval graphs, and graph planarity using PQ-tree algorithms. J Comput Syst Sci 13:335–379
3. Cai J, Han X, Tarjan R (1993) An O(m log n)-time algorithm for the maximal planar subgraph problem. SIAM J Comput 22:1142–1162
4. Chiba T, Nishioka I, Shirakawa I (1979) An algorithm of maximal planarization of graphs. In: Proc. 1979 IEEE Symp. Circuits and Sys., pp 649–652
5. Cimikowski RJ (1995) An analysis of heuristics for the maximum planar subgraph problem. In: Proc. 6th ACM-SIAM Symp. Discrete Algorithms, pp 322–331
6. Di Battista G, Eades P, Tamassia R, Tollis IG (1994) Algorithms for drawing graphs: An annotated bibliography. Comput Geom Th Appl 1:235–282
7. Di Battista G, Tamassia R (1989) Incremental planarity testing. Proc. 30th IEEE Symp. FOCS, pp 436–441
8. Eades P, Foulds LR, Giffin JW (1982) An efficient heuristic for identifying a maximum weight planar subgraph. In: Lecture Notes Math, vol 952. Springer, Berlin, pp 239–251
9. Feo TA, Resende MGC (1995) Greedy randomized adaptive search procedures. J Global Optim 6:109–133
10. Foulds LR, Robinson RW (1976) A strategy for solving the plant layout problem. Oper Res Quart 27:845–855
11. Foulds LR, Robinson RW (1978) Graph theoretic heuristics for the plant layout problem. Internat J Production Res 16:27–37
12. Goldschmidt O, Takvorian A (1994) An efficient graph planarization two-phase heuristic. Networks 24:69–73
13. Hasan M, Osman IH (1995) Local search algorithms for the maximal planar layout problem. Internat Trans Oper Res 2:89–106
14. Hopcroft J, Tarjan RE (1974) Efficient planarity testing. J ACM 21:549–568
15. Jünger M, Leipert S, Mutzel P (1998) A note on computing a maximal planar subgraph using PQ-trees. Techn Report Inst Informatik Univ Köln 98.320
16. Jünger M, Mutzel P (1996) Maximum planar subgraphs and nice embeddings: Practical layout tools. Algorithmica 16:33–59
17. Laguna M, Marti R (1999) GRASP and path relinking for 2-layer straight line crossing minimization. INFORMS J Comput 11:44–52
18. Lempel A, Even S, Cedarbaum I (1966) An algorithm for planarity testing of graphs. Proc. Theory of Graphs Internat. Symp. Gordon and Breach, New York, pp 215–232
19. Lengauer T (1990) Combinatorial algorithms for integrated circuit layout. Wiley, New York
20. Leung J (1992) A new graph-theoretic heuristic for facility layout. Managem Sci 38:594–605
21. Liu PC, Geldmacher RC (1977) On the deletion of nonplanar edges of a graph. In: Proc. 10th SE Conf. Comb., Graph Theory, and Comput., pp 727–738
22. Resende MGC, Ribeiro CC (1997) A GRASP for graph planarization. Networks 29:173–189
23. Ribeiro CC, Resende MGC (1999) Algorithm 797: FORTAN subroutines for approximate solution of graph planarization problems using GRASP. ACM Trans Math Softw 25:341–352
24. Stoer M (1992) Design of survivable networks. Lecture Notes Math, vol 1531. Springer, Berlin
25. Takefuji Y, Lee KC (1989) A near-optimum parallel planarization algorithm. Science 245:1221–1223
26. Takefuji Y, Lee K-C, Cho YB (1991) Comments on an O(n2)algorithm for graph planarization. IEEE Trans Computer-Aided Design 10:1582–1583
27. Tamassia R, DiBattista G (1988) Automatic graph drawing and readability of diagrams. IEEE Trans Syst, Man Cybern 18:61–79

Graph Realization via Semidefinite Programming

ANTHONY MAN-CHO SO[1], YINYU YE[2]

[1] Department of Systems Engineering and Engineering Management, The Chinese University of Hong Kong, Hong Kong, China

[2] Department of Management Science and Engineering and, by courtesy, Electrical Engineering, Stanford University, Stanford, USA

MSC2000: 51K05, 52C25, 68Q25, 68U05, 90C22, 90C35

Article Outline

Introduction

Due to its fundamental nature and versatile modelling power, the *Graph Realization Problem* is one of the most well-studied problems in distance geometry and has received attention in many communities. In that problem, one is given a graph $G = (V, E)$ and a set of non-negative edge weights $\{d_{ij} : (i, j) \in E\}$, and the goal is to compute a *realization* of G in the Euclidean space $\mathbb{R}^k$ for a given dimension $k \geq 1$, i. e. to place the vertices of G in $\mathbb{R}^k$ such that the Euclidean distance between every pair of adjacent vertices v_i, v_j is equal to the prescribed weight d_{ij}. The Graph Realization Problem and its variants arise from applications in very diverse areas, the two most prominent of which being molecular conformation (see, e. g., [13,15,16,19,32]) and wireless sensor network localization (see, e. g., [2,8,14,22,24]). In molecular conformation, one is interested in determining the spatial structure of a molecule from a set of geometric constraints; in wireless sensor network localization, one is interested in inferring the locations of sensor nodes in a sensor network from connectivity-imposed proximity constraints. Thus, in these contexts, an algorithm that finds a realization of the vertices in the required dimension will have interesting biochemical and engineering consequences. Unfortunately, unless P = NP, there is no efficient algorithm for solving the Graph Realization Problem for any fixed $k \geq 1$ ([23]; see also [3,4]). Nevertheless, many heuristics have been developed for the problem over the years, and various approaches have been taken to improve their efficiency (see, e. g., [1,2,13,14,15,18,20]). However, these approaches have their limitations. Specifically, either they solve the original problem only for a very restricted family of instances, or it is not clear when the algorithm would solve the original problem. Thus, an interesting question arises: given a relaxation of the Graph Realization Problem, can one derive reasonably general conditions under which the relaxation is *exact*?

We begin by examining a semidefinite programming (SDP) relaxation proposed by [10] in Section Formulation. We introduce the notion of unique k-realizability and show that the SDP relaxation is exact if and only if the input instance is uniquely k-realizable, where k is the given dimension. The notion of unique k-realizability is attractive, as it has a straightforward geometric interpretation and is also very suitable for the algorithmic treatment of the Graph Realization Problem.

Although we have formulated the Graph Realization Problem as a feasibility problem, it is clear that one can also formulate various optimization versions of it. One particularly useful objective is to maximize the sum of the distances between certain pairs of non-adjacent vertices. Such an objective essentially stretches apart pairs of non-adjacent vertices, and is more likely to flatten a high-dimensional realization into a lower dimensional one. Indeed, such a device has been proven to be very useful for finding low-dimensional realizations both in theory (see, e. g., [6,7]) and in practice (see, e. g., [9,29,30]). In Section Applications, we show how these ideas can be incorporated into the SDP model and demonstrate a connection between SDP theory and tensegrity theory in discrete geometry.

Formulation

We begin by introducing the semidefinite programming (SDP) relaxation proposed by [10]. Let $G = (V, E)$ be a graph, and let $k \geq 1$ be an integer. Let $V_1 = \{1, \dots, n\}$ and $V_2 = \{n+1, \dots, n+m\}$ be a partition of V. The vertices in V_1 (resp. V_2) are said to be *unpinned* (resp. *pinned*). Specifically, let $\mathbf{a} = (a_i)_{i \in V_2}$ be given, where $a_i \in \mathbb{R}^k$ for all $i \in V_2$. Then, the vertex $i \in V_2$ is constrained to be at a_i, while there are no such restrictions on the vertices in V_1. For our purposes, we may assume that $V_2 \neq \emptyset$, since we can always pin one vertex at the origin. We may also assume that $E' = \{(i, j) : i, j \in V_2\} \subset E$, since the distance between any two pinned vertices is trivially known. Now, let $E_1 = \{(i, j) \in E : i, j \in V_1\}$ be the set of edges between two unpinned vertices, and let $E_2 = \{(i, j) \in E : i \in V_2, j \in V_1\}$ be the set of edges between a pinned and an unpinned vertex. Let $\mathbf{d} = (d_{ij}^2)_{(i,j) \in E_1}$ (resp. $\bar{\mathbf{d}} = (\bar{d}_{ij}^2)_{(i,j) \in E_2}$) be a set of weights on the edges in E_1 (resp. E_2). We are then interested in finding vectors $x_1, \ldots, x_n \in \mathbb{R}^k$ such that:

$$\begin{array}{rcll} \|x_i - x_j\|^2 & = & d_{ij}^2 & \text{for } (i,j) \in E_1 \\ \|a_i - x_j\|^2 & = & \bar{d}_{ij}^2 & \text{for } (i,j) \in E_2 \end{array} \qquad (1)$$

Here, $\|\cdot\|$ is the Euclidean norm, i. e. $\|x\| = \left(\sum_{i=1}^k x_i^2\right)^{1/2}$ for $x \in \mathbb{R}^k$. We say that $\mathbf{p} = (p_1, \ldots, p_n) \in \mathbb{R}^{kn}$ is a *realization* of $(G, (\mathbf{d}, \bar{\mathbf{d}}), \mathbf{a})$ in $\mathbb{R}^k$ if it satisfies (1). One may obtain a semidefinite relaxation of (1) as follows. Let $X = [x_1\, x_2\, \ldots\, x_n]$ be the $k \times n$ matrix that needs to be determined. Then, for all $(i,j) \in E_1$, we have:

$$\begin{aligned} \|x_i - x_j\|^2 &= (e_i - e_j)^T X^T X (e_i - e_j) \\ &= (e_i - e_j)(e_i - e_j)^T \bullet (X^T X) \end{aligned}$$

and for all $(i,j) \in E_2$, we have:

$$\|a_i - x_j\|^2 = \begin{pmatrix} a_i \\ -e_j \end{pmatrix}^T [I_k\ X]^T [I_k\ X] \begin{pmatrix} a_i \\ -e_j \end{pmatrix} = \begin{pmatrix} a_i \\ -e_j \end{pmatrix} \begin{pmatrix} a_i \\ -e_j \end{pmatrix}^T \bullet \begin{bmatrix} I_k & X \\ X^T & X^T X \end{bmatrix}$$

Here, e_i is the ith standard basis vector of $\mathbb{R}^n$, I_k is the k-dimensional identity matrix, and $\bullet$ is the Frobenius inner product on the space of symmetric matrices, i. e. $A \bullet B = \operatorname{tr}(A^T B) = \sum_{i,j=1}^n a_{ij} b_{ij}$ for symmetric $n \times n$ matrices A and B. Thus, problem (1) becomes that of finding a symmetric matrix $Y \in \mathbb{R}^{n \times n}$ and a matrix $X \in \mathbb{R}^{k \times n}$ that satisfy the following system:

$$\begin{gathered} (e_i - e_j)(e_i - e_j)^T \bullet Y = d_{ij}^2 \\ \text{for } (i,j) \in E_1 \\ \begin{pmatrix} a_i \\ -e_j \end{pmatrix} \begin{pmatrix} a_i \\ -e_j \end{pmatrix}^T \bullet \begin{bmatrix} I_k & X \\ X^T & Y \end{bmatrix} = \bar{d}_{ij}^2 \\ \text{for } (i,j) \in E_2 \\ Y = X^T X \end{gathered} \qquad (2)$$

By relaxing $Y = X^T X$ to $Y \succeq X^T X$ and using Schur's complement (see, e. g., [11]), we obtain the following relaxed problem:

$$\begin{array}{lll} \sup & 0 & \\ \text{subject to} & E_{ij} \bullet Z = d_{ij}^2 & \text{for } (i,j) \in E_1 \\ & \bar{E}_{ij} \bullet Z = \bar{d}_{ij}^2 & \text{for } (i,j) \in E_2 \\ & Z \succeq \mathbf{0},\ Z_{1:k,1:k} = I_k & \end{array} \qquad (3)$$

where $Z_{1:k,1:k}$ is the $k \times k$ principal submatrix of Z indexed by the first k rows (columns),

$$E_{ij} = \begin{pmatrix} \mathbf{0} \\ e_i - e_j \end{pmatrix} \begin{pmatrix} \mathbf{0} \\ e_i - e_j \end{pmatrix}^T$$

$$\text{and} \quad \bar{E}_{ij} = \begin{pmatrix} a_i \\ -e_j \end{pmatrix} \begin{pmatrix} a_i \\ -e_j \end{pmatrix}^T$$

Note that this formulation forces any feasible solution matrix to have rank at least k. To derive the dual of (3), let $(\theta_{ij})_{(i,j) \in E_1}$ and $(w_{ij})_{(i,j) \in E_2}$ be the dual multipliers of the constraints on E_1 and E_2, respectively. Then, the dual of (3) is given by:

$$\begin{array}{ll} \inf & I_k \bullet V + \sum_{(i,j) \in E_1} \theta_{ij} d_{ij}^2 \\ & + \sum_{(i,j) \in E_2} w_{ij} \bar{d}_{ij}^2 \\ \text{subject to} & U \equiv \begin{bmatrix} V & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} + \sum_{(i,j) \in E_1} \theta_{ij} E_{ij} \\ & + \sum_{(i,j) \in E_2} w_{ij} \bar{E}_{ij} \succeq \mathbf{0} \\ & \theta_{ij} \in \mathbb{R} \text{ for all } (i,j) \in E_1; \\ & w_{ij} \in \mathbb{R} \text{ for all } (i,j) \in E_2 \end{array} \qquad (4)$$

Note that the dual is always feasible, as $V = \mathbf{0}$, $\theta_{ij} = 0$ for all $(i,j) \in E_1$ and $w_{ij} = 0$ for all $(i,j) \in E_2$ is a feasible solution. Moreover, this solution has a dual objective value of 0. Thus, by the SDP strong duality theorem, if the primal is also feasible, then there is no duality gap between (3) and (4). Moreover, if Z is feasible for (3) and U is optimal for (4), then by complementarity, we have $\operatorname{rank}(Z) + \operatorname{rank}(U) \le k + n$. In particular, since $\operatorname{rank}(Z) \ge k$, we must have $\operatorname{rank}(U) \le n$.

We are interested in deriving the conditions under which the relaxation (3) is exact for (2). Towards that end, let us first introduce a definition:

Definition 1 We say that an instance $(G, (\mathbf{d}, \bar{\mathbf{d}}), \mathbf{a})$ is uniquely k-realizable if (i) there is a unique realization $\mathbf{p} = (p_1, \ldots, p_n)$ of $(G, (\mathbf{d}, \bar{\mathbf{d}}), \mathbf{a})$ in $\mathbb{R}^k$, and (ii) there does not exist $p'_1, \ldots, p'_n \in \mathbb{R}^l$, where $l > k$, such that:

$$\begin{array}{rcll} \|p'_i - p'_j\|^2 & = & d_{ij}^2 & \text{for } (i,j) \in E_1 \\ \left\| \begin{pmatrix} a_i \\ \mathbf{0} \end{pmatrix} - p'_j \right\|^2 & = & \bar{d}_{ij}^2 & \text{for } (i,j) \in E_2 \\ p'_i & \neq & \begin{pmatrix} p_i \\ \mathbf{0} \end{pmatrix} & \text{for some } 1 \le i \le n \end{array}$$

For the motivation of this definition, see [25]. We remark that Definition 1 can be viewed as a new notion of rigidity which takes into account both the combinatorial and the geometric aspects of the Graph Realization Problem.

At this point it is fair to ask whether Definition 1 is vacuous, i. e. whether uniquely k-realizable instances exist at all. It is not hard to see that they do exist for all $k \geq 1$. In fact, there exists a family of uniquely k-realizable instances in which the number of edges scales linearly with the number of vertices ([25]). This refutes a common belief in the literature (see, e. g., [2,5]) that the graph of any uniquely k-realizable instance must have $\Omega(n^2)$ edges.

Having established the existence of uniquely k-realizable instances, we are now ready to state the main theorem of this section. For a proof, see [25,27].

Theorem 1 *Let $G = (V, E)$ be connected, and let $\mathbf{d}$, $\bar{\mathbf{d}}$ and $\mathbf{a}$ be given. Then, the following are equivalent:*
(1) The instance $(G, (\mathbf{d}, \bar{\mathbf{d}}), \mathbf{a})$ is uniquely k-realizable.
(2) The max-rank solution matrix of (3) has rank k.
(3) The solution matrix of (3) satisfies $Y = X^T X$.

Although unique k-realizability is a useful notion in determining the solvability of the Graph Realization Problem, it is not stable under perturbation. Indeed, there exist instances that are uniquely k-realizable, but may no longer be so after small perturbation of the unpinned vertices; see [27]. This motivates us to define another notion called strong k-realizability:

Definition 2 We say that an instance $(G, (\mathbf{d}, \bar{\mathbf{d}}), \mathbf{a})$ is strongly k-realizable if (4) has a rank–n optimal dual slack matrix.

Note that if an instance is strongly k-realizable, then it is uniquely k-realizable by complementarity and Theorem 1, since the rank of any solution to (3) is equal to k.

Given an instance $\mathcal{I} = (G, (\mathbf{d}, \bar{\mathbf{d}}), \mathbf{a})$, we say that the instance $(G', (\mathbf{d}', \bar{\mathbf{d}}'), \mathbf{a})$ is a *sub–instance* of $\mathcal{I}$ if G' is a subgraph of G that includes all the pinned vertices, and $(\mathbf{d}', \bar{\mathbf{d}}')$ is the restriction of $(\mathbf{d}, \bar{\mathbf{d}})$ on G'. As indicated by the following theorem, the notion of strong k-realizability is very useful in identifying the uniquely k-realizable sub–instances of a given instance. Its proof can be found in [25,27].

Theorem 2 *Suppose that a given instance $\mathcal{I}$ contains a sub–instance $\mathcal{I}'$ that is strongly k-realizable. Then, in any solution to (3), the submatrix that corresponds to $\mathcal{I}'$ has rank k.*

Applications

It is often observed in practice that by "stretching apart" pairs of non-adjacent vertices, one is more likely to flatten a high-dimensional realization into a lower dimensional one. We now formalize this observation using elements of tensegrity theory (see, e. g., [12,21]). We begin with some definitions:

Definition 3 A *tensegrity* $G(\mathbf{p})$ is a graph $G = (V, E)$ together with a configuration $\mathbf{p} = (p_1, \dots, p_n) \in \mathbb{R}^{kn}$ such that each edge is labelled as a cable, strut, or bar; each vertex is labelled as pinned or unpinned; and vertex $i \in V$ is assigned the coordinates $p_i \in \mathbb{R}^k$ for $1 \leq i \leq n$.

The label on each edge is intended to indicate its functionality. Cables (resp. struts) are allowed to decrease (resp. increase) in length (or stay the same length), but not to increase (resp. decrease) in length. Bars are forced to remain the same length. As before, a pinned vertex is forced to remain where it is. Given a graph $G = (V, E)$ and a set $\mathbf{d}$ of weights on the edges, if (i, j) is a cable (resp. strut), then d_{ij} will be the upper (resp. lower) bound on its length. If (i, j) is a bar, then d_{ij} will simply be its length.

An important concept in the study of tensegrities is that of an *equilibrium stress*:

Definition 4 An *equilibrium stress* for $G(\mathbf{p})$ is an assignment of real numbers $\omega_{ij} = \omega_{ji}$ to each edge $(i, j) \in E$ such that for each unpinned vertex i of G, we have:

$$\sum_{j:(i,j)\in E} \omega_{ij}(p_i - p_j) = \mathbf{0} \qquad (5)$$

Furthermore, we say that the equilibrium stress $\omega = \{\omega_{ij}\}$ is *proper* if $\omega_{ij} = \omega_{ji} \geq 0$ (resp. ≤ 0) if (i, j) is a cable (resp. strut).

Clearly, the zero stress $\omega = \mathbf{0}$ is a proper equilibrium stress, but it is not too interesting. On the other hand, suppose that $G(\mathbf{p})$ has a non-zero equilibrium stress, and that at least one of the incident edges of vertex i has a non-zero stress. Then, Eq. (5) implies that the set of vectors $\{p_j - p_i : (i, j) \in E\}$ is linearly dependent, and hence those vectors span a lower dimensional space. Thus, it would be nice to have conditions that guarantee the existence of a non-zero proper equilibrium stress. It turns out that the concept of an *unyielding tensegrity* is useful for that purpose.

Definition 5 Let $G = (V, E)$ be a graph, and let $\mathbf{p}$ and $\mathbf{q}$ be two configurations of G. We say that $G(\mathbf{p})$ *dominates* $G(\mathbf{q})$ (denoted by $G(\mathbf{p}) \trianglerighteq G(\mathbf{q})$) if for every pinned vertex i, we have $p_i = q_i$, and for every edge $(i, j) \in E$, we have:

$$\|p_i - p_j\| \left\{ \begin{array}{c} \geq \\ = \\ \leq \end{array} \right\} \|q_i - q_j\| \quad \text{if } (i,j) \text{ is a } \left\{ \begin{array}{c} \text{cable} \\ \text{bar} \\ \text{strut} \end{array} \right\}$$

We call $G(\mathbf{p})$ an *unyielding tensegrity* and $\mathbf{p}$ an *unyielding configuration* if any other configuration $\mathbf{q}$ with $G(\mathbf{p}) \trianglerighteq G(\mathbf{q})$ satisfies $\|p_i - p_j\| = \|q_i - q_j\|$ for all $(i, j) \in E$.

We are now ready to state the following theorem due to [6], which plays a crucial role in the characterization of the so-called 3-realizable graphs (informally, a graph G is 3-realizable if, given *any* set $\mathbf{d}$ of edge weights, whenever $(G, \mathbf{d})$ is realizable at all, then it can also be realized in $\mathbb{R}^3$; for further details, see [7]):

Theorem 3 *If $G(\mathbf{p})$ is an unyielding tensegrity with exactly one strut or cable, then $G(\mathbf{p})$ has an equilibrium stress that is non-zero on at least one edge.*

Belk's proof of Theorem 3 uses the Inverse Function Theorem and hence is not constructive. It turns out that the problem of computing an unyielding configuration $\mathbf{p}$ of a graph G can be formulated as an SDP. What is even more interesting is that the optimal dual multipliers of the SDP will give rise to a non-zero proper equilibrium stress for $G(\mathbf{p})$. Consequently, we obtain a constructive proof of Theorem 3. In fact, the SDP-based proof yields more information than that offered by Belk's proof.

Specifically, let V_1, V_2, E_1, E_2 be as before, and set $E_1^c = \{(i,j) \notin E : i, j \in V_1\}$ and $E_2^c = \{(i,j) \notin E : i \in V_2, j \in V_1\}$. Let C_1, S_1 be disjoint subsets of E_1^c, and let C_2, S_2 be disjoint subsets of E_2^c. The pairs in C_i are intended to be *cables*, and those in S_i are intended to be *struts*. We remark that we do not assume the sets C_1, C_2, S_1, S_2 to be non-empty.

Now, consider the following SDP, where we augment the formulation (3) with an objective function:

$$\begin{array}{lll} \sup & \displaystyle\sum_{(i,j)\in S_1} E_{ij} \bullet Z + \sum_{(i,j)\in S_2} \bar{E}_{ij} \bullet Z & \\ & \displaystyle - \sum_{(i,j)\in C_1} E_{ij} \bullet Z - \sum_{(i,j)\in C_2} \bar{E}_{ij} \bullet Z & \\ \text{subject to} & E_{ij} \bullet Z = d_{ij}^2 & \text{for } (i,j) \in E_1 \\ & \bar{E}_{ij} \bullet Z = \bar{d}_{ij}^2 & \text{for } (i,j) \in E_2 \\ & Z \succeq \mathbf{0}, Z_{1:k,1:k} = I_k & \end{array} \tag{6}$$

The dual of (6) is given by:

$$\begin{array}{ll} \inf & \displaystyle I_k \bullet V + \sum_{(i,j)\in E_1} \theta_{ij} d_{ij}^2 \\ & \displaystyle + \sum_{(i,j)\in E_2} w_{ij} \bar{d}_{ij}^2 \\ \text{subject to} & \displaystyle U \equiv - \sum_{(i,j)\in S_1} E_{ij} - \sum_{(i,j)\in S_2} \bar{E}_{ij} \\ & \displaystyle + \sum_{(i,j)\in C_1} E_{ij} + \sum_{(i,j)\in C_2} \bar{E}_{ij} \\ & \displaystyle + \begin{bmatrix} V & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} + \sum_{(i,j)\in E_1} \theta_{ij} E_{ij} \\ & \displaystyle + \sum_{(i,j)\in E_2} w_{ij} \bar{E}_{ij} \succeq \mathbf{0} \end{array} \tag{7}$$

We then have the following theorem due to [26]:

Theorem 4 *Let $G = (V, E)$, $\mathbf{d}$, $\bar{\mathbf{d}}$ and $\mathbf{a}$ be given such that:*

(1) there is at least one pinned vertex, and

(2) the graph $G \backslash \{n+2, \ldots, n+m\}$ is connected.

Consider the SDP (6), where we assume that:

(3) it is strictly feasible, and

(4) the objective function is not vacuous, i. e. at least one of the sets C_1, C_2, S_1, S_2 is non-empty.

Let $\bar{\mathbf{x}} = (\bar{x}_1, \ldots, \bar{x}_n) \in \mathbb{R}^{ln}$ be the positions of the unpinned vertices in $\mathbb{R}^l$ (for some $l \geq k$), obtained from the optimal primal matrix $\bar{Z}$, and let $\{\bar{\theta}_{ij}, \bar{w}_{ij}\}$ be the optimal dual multipliers. Suppose that we assign the stress $\bar{\theta}_{ij}$ (resp. $\bar{w}_{ij}$) to the bar $(i, j) \in E_1$ (resp. $(i, j) \in E_2$), a stress of 1 to all the cables in $C_1 \cup C_2$, and a stress of -1 to all the struts in $S_1 \cup S_2$. Then, the resulting assignment yields a non-zero proper equilibrium stress for the *tensegrity* $G'(\bar{\mathbf{x}}, \bar{\mathbf{a}})$, *where* $G' = (V, E \cup C_1 \cup C_2 \cup S_1 \cup S_2)$ *and* $\bar{\mathbf{a}} = (\bar{a}_{n+1}, \dots, \bar{a}_{n+m})$, *where:*

$$\bar{a}_i = \begin{pmatrix} a_i \\ \mathbf{0} \end{pmatrix} \in \mathbb{R}^l$$

The intuition behind the proof of Theorem 4 is simple. Suppose that (6) and (7) achieve the same optimal value, and that the common optimal value is attained by the primal matrix $\bar{Z}$ and the dual matrix $\bar{U}$. Then, the desired result should follow from one of the conditions for strong duality, namely the identity $\bar{Z}\bar{U} = \mathbf{0}$. Of course, strong duality for SDP does not necessarily hold, and even when it does, there is no guarantee that the optimal value is attained by any matrix (see, e. g., [17] for some examples). Thus, some additional technical assumptions are needed, and items (2) and (3) in the statement of Theorem 4 turn out to be sufficient. In fact, the conclusion of Theorem 4 remains valid if we replace (3) by the following:

(3′) the optimal value of (7) *is attained by some dual feasible matrix*

We remark that in most applications of Theorem 4, there will only be one pinned vertex, namely $a_{n+1} = \mathbf{0}$. Thus, primal strict feasibility can be ensured if the given weights **d** admit a realization whose vertices are in general position, and the connectivity condition is simply the statement that G is connected. However, the strict feasibility assumption (or the dual attainment assumption) does weaken the applicability of Theorem 4. In particular, Theorem 4 is not as general as Theorem 3, although this can be fixed (see [25] for details).

Besides strict feasibility, it is also assumed that the given instance has at least one pinned vertex. Such an assumption is necessary in order to ensure that the entries of $\bar{Z}$ are bounded, but one can no longer argue that the net stress exerted on a pinned vertex is zero. However, if there is only one pinned vertex in the given instance, then the net stress exerted on it will be zero. Thus, one may assume without loss of generality that the given instance has one pinned vertex.

Finally, observe that the assumptions in the statement of Theorem 4 buy us some additional information that is not offered by Theorem 3. Specifically, the equilibrium stress obtained in Theorem 4 is non-zero on all the cables and struts, and the magnitudes of the stress on all the cables and struts can be prescribed (by assigning appropriate weights to each summand in the primal objective function).

Relation to the Maximum Variance Unfolding Method

The idea of stretching apart pairs of non-adjacent vertices has also been used in the artifical intelligence community to detect and discover low-dimensional structure in high-dimensional data. For instance, in [29] (see also [30]), the authors proposed the so-called *Maximum Variance Unfolding* (MVU) method for the problem of manifold learning. The idea is to map a given set of high-dimensional vectors $p_1, \dots, p_n \in \mathbb{R}^l$ to a set of low-dimensional vectors $q_1, \dots, q_n \in \mathbb{R}^k$ (where $1 \leq k \ll l$ are given) with maximum total variance, while at the same time preserves the local distances. More precisely, consider an n-vertex connected graph $G = (V, E)$, where the set E of edges represents the set of distances that need to be preserved. The desired set of low-dimensional vectors can then be obtained by solving the following quadratic program:

$$\begin{array}{ll} \text{maximize} & \displaystyle\sum_{i=1}^{n} \|x_i\|^2 \\ \text{subject to} & \displaystyle\sum_{i=1}^{n} x_i = \mathbf{0} \\ & \|x_i - x_j\|^2 = \|p_i - p_j\|^2 \\ & \text{for } (i,j) \in E \\ & x_i \in \mathbb{R}^k \qquad \text{for } 1 \leq i \leq n \end{array} \tag{8}$$

To explain the rationale behind the above formulation, we observe that the first constraint centers the solution vectors at the origin and eliminates the translational degree of freedom. Moreover, it implies that the objective function of (8) can be written as:

$$\sum_{i=1}^{n} \|x_i\|^2 = \frac{1}{2n} \sum_{i,j=1}^{n} \|x_i - x_j\|^2$$

Thus, we see that the MVU method attempts to “unfold” the manifold by pulling the data points as far apart as possible while preserving the local distances. We remark that such a technique has also been used for the problem of sensor network localization (see, e. g., [9,31]). Now, using the ideas in Section Formulation, we can formulate a semidefinite relaxation of (8) as follows:

$$\begin{aligned} \sup \quad & I \bullet X \\ \text{subject to} \quad & ee^T \bullet X = 0 \\ & E_{ij} \bullet X = \|v_i - v_j\|^2 \quad \text{for } (i,j) \in E \\ & X \succeq \mathbf{0} \end{aligned} \tag{9}$$

Here, $e = (1, 1, \dots, 1)$, $E_{ij} = (e_i - e_j)(e_i - e_j)^T$, and e_i is the ith standard basis vector of $\mathbb{R}^n$. It turns out that problem (9) and its dual are closely related to the problem of finding the fastest mixing Markov process on a graph, as well as to various spectral methods for dimensionality reduction. We shall not elaborate on these results here and refer the interested reader to [28,33] for further details. Instead, we will show that the MVU problem (9) can be viewed as a problem of finding an unyielding configuration of a certain tensegrity. To begin, suppose that we are given an n-vertex connected graph $G = (\{1, \dots, n\}, E)$ and a configuration $\mathbf{p} = (p_1, \dots, p_n) \in \mathbb{R}^{ln}$ of the vertices. Consider the tensegrity $G'(\mathbf{p}')$, where G' is obtained from G by adding a new vertex $n+1$ and connecting it to all the vertices of G, and $\mathbf{p}' = (\mathbf{p}, \mathbf{0}) \in \mathbb{R}^{l(n+1)}$, i. e. vertex $n+1$ is located at the origin. Furthermore, we label the edges in E as bars and the edges in $S \equiv \{(n+1, i) : 1 \le i \le n\}$ as struts. Suppose that we pin vertex $n+1$ at the origin, i. e. $a_{n+1} = \mathbf{0}$. Now, consider the following SDP:

$$\begin{aligned} \sup \quad & \sum_{i:(n+1,i)\in S} \bar{E}_{n+1,i} \bullet Z \\ \text{subject to} \quad & E_{ij} \bullet Z = \|p_i - p_j\|^2 \quad \text{for } (i,j) \in E \\ & Z \succeq \mathbf{0}, Z_{1:k,1:k} = I_k \end{aligned} \tag{10}$$

where:

$$E_{ij} = \begin{pmatrix} \mathbf{0} \\ e_i - e_j \end{pmatrix} \begin{pmatrix} \mathbf{0} \\ e_i - e_j \end{pmatrix}^T$$

$$\text{and} \quad \bar{E}_{n+1,i} = \begin{pmatrix} \mathbf{0} \\ -e_i \end{pmatrix} \begin{pmatrix} \mathbf{0} \\ -e_i \end{pmatrix}^T$$

It is clear that (10) is an instance of (6). Moreover, it can be shown ([25]) that the positions $\bar{\mathbf{x}} \in \mathbb{R}^{ln}$ of the unpinned vertices obtained from the optimal primal matrix $\bar{Z}$ are automatically centered at the origin, even though such a constraint is not explicitly enforced. Thus, we see that problem (10) is equivalent to the MVU problem (9).

From the above discussion, we see that the formulation (6) is more general than the MVU formulation (9). Moreover, the flexibility in the formulation (6) often allows one to achieve the desired dimensionality reduction which the MVU formulation cannot achieve. For instance, consider the case where the input graph G is a tree. It is not hard to show that there is a placement of struts such that *all* the optimal solutions to (6) have rank 1 and hence they all give rise to one-dimensional realizations. On the other hand, the MVU formulation may yield a two-dimensional realization; see [25] for an example.

References

1. Alfakih AY, Khandani A, Wolkowicz H (1999) Solving Euclidean Distance Matrix Completion Problems via Semidefinite Programming. Comput Optim Appl 12:13–30
2. Aspnes J, Eren T, Goldenberg DK, Morse AS, Whiteley W, Yang YR, Anderson BDO, Belhumeur PN (2006) A Theory of Network Localization. IEEE Trans Mobile Comput 5(12):1663–1678
3. Aspnes J, Goldenberg D, Yang YR (2004) On the Computational Complexity of Sensor Network Localization. In: Nikoletseas S, Rolim JDP (eds) In: Proc. 1st Int Workshop Algorithmic Aspects Wirel Sens Netw (ALGOSENSORS 2004) Lecture Notes in Computer Science, vol 3121. Springer, Berlin, pp 32–44
4. Bădoiu M, Demaine ED, Hajiaghayi M, Indyk P (2006) Low–Dimensional Embedding with Extra Information. Discret Comput Geom 36(4):609–632
5. Basu A, Gao J, Mitchell JSB, Sabhnani G (2006) Distributed Localization Using Noisy Distance and Angle Information. In: Proc. 7th ACM Int Symp Mobile Ad Hoc Netw Comput (MobiHoc 2006), pp 262–273
6. Belk M (2007) Realizability of Graphs in Three Dimensions. Discret Comput Geom 37(2):139–162
7. Belk M, Connelly R (2007) Realizability of Graphs. Discret Comput Geom 37(2):125–137
8. Biswas P, Lian T-C, Wang T-C, Ye Y (2006) Semidefinite Programming Based Algorithms for Sensor Network Localization. ACM Trans Sensor Netw 2(2):188–220
9. Biswas P, Liang T-C, Toh K-C, Wang T-C, Ye Y (2006) Semidefinite Programming Approaches for Sensor Network Localization with Noisy Distance Measurements. IEEE Trans Autom Sci Eng 3(4):360–371
10. Biswas P, Ye Y (2004) Semidefinite Programming for Ad Hoc Wireless Sensor Network Localization. In: Proc. 3rd Int Symposium on Information Processing in Sensor Networks (IPSN 2004), pp 46–54

11. Boyd S, El Ghaoui L, Feron E, Balakrishnan V (1994) Linear Matrix Inequalities in System and Control Theory, volume 15 of SIAM Stud Appl Numer Math. Soc Ind Appl Math, Philadelphia, Pennsylvania
12. Connelly R (1982) Rigidity and Energy. Invent Math 66:11–33
13. Crippen GM and Havel TF (1988) Distance Geometry and Molecular Conformation. Chemometrics Series, vol 15, Res Stud Press Ltd., Taunton, Somerset, England
14. Doherty L, Pister KSJ, El Ghaoui L (2001) Convex Position Estimation in Wireless Sensor Networks. In: Proc. 20th Annu IEEE Conference Comput Commun (INFOCOM 2001), vol 3, pp 1655–1663
15. Havel TF (2003) Metric Matrix Embedding in Protein Structure Calculations, NMR Spectra Analysis, and Relaxation Theory. Magn Reson Chem 41(S1):37–50
16. Havel TF, Wüthrich K (1985) An Evaluation of the Combined Use of Nuclear Magnetic Resonance and Distance Geometry for the Determination of Protein Conformations in Solution. J Mol Biol 182(2):281–294
17. Helmberg C (2000) Semidefinite Programming for Combinatorial Optimization. Technical Report ZR–00–34, Konrad–Zuse–Zentrum für Informationstechnik Berlin, Berlin, Germany
18. Hendrickson B (1995) The Molecule Problem: Exploiting Structure in Global Optimization. SIAM J Optim 5(4):835–857
19. Kaptein R, Boelens R, Scheek RM, van Gunsteren WF (1988) Protein Structures from NMR. Biochemistry 27(15):5389–5395
20. Laurent M (2000) Polynomial Instances of the Positive Semidefinite and Euclidean Distance Matrix Completion Problems. SIAM J Matrix Analysis Appl 22(3):874–894
21. Roth B, Whiteley W (1981) Tensegrity Frameworks. Trans Am Math Soc 265(2):419–446
22. Savvides A, Han C-C, Strivastava MB (2001) Dynamic Fine–Grained Localization in Ad–Hoc Networks of Sensors. In: Proc. 7th Annu Int Conference Mobile Comput Netw (MobiCom 2001), pp 166–179
23. Saxe JB (1979) Embeddability of Weighted Graphs in *k*–Space is Strongly NP–Hard. In: Proc. 17th Allerton Conference Commun, Control, and Comput, pp 480–489
24. Shang Y, Ruml W, Zhang Y, Fromherz M (2004) Localization from Connectivity in Sensor Networks. IEEE Trans Parallel Distrib Syst 15(11):961–974
25. So AM-C (2007) A Semidefinite Programming Approach to the Graph Realization Problem: Theory, Applications and Extensions. PhD thesis, Department of Computer Science, Stanford University, Stanford
26. So AM-C, Ye Y (2006) A Semidefinite Programming Approach to Tensegrity Theory and Realizability of Graphs. In: Proc. 17th Annu ACM–SIAM Symposium Discrete Algorithm (SODA 2006), pp 766–775
27. So AM-C, Ye Y (2007) Theory of Semidefinite Programming for Sensor Network Localization. Math Program Ser B 109(2):367–384
28. Sun J, Boyd S, Xiao L, Diaconis P (2006) The Fastest Mixing Markov Process on a Graph and a Connection to a Maximum Variance Unfolding Problem. SIAM Rev 48(4):681–699
29. Weinberger KQ, Saul LK (2006) Unsupervised Learning of Image Manifolds by Semidefinite Programming. Int J Comput Vision 70(1):77–90
30. Weinberger KQ, Sha F, Saul LK (2004) Learning a Kernel Matrix for Nonlinear Dimensionality Reduction. In: Proc. 21st Int Conference Mach Learn (ICML 2004), pp 839–846
31. Weinberger KQ, Sha F, Zhu Q, Saul LK (2007) Graph Laplacian Regularization for Large–Scale Semidefinite Programming. In: Schölkopf B, Platt J, Hofmann T (eds) Advances in Neural Information Processing Systems 19. MIT Press, Cambridge, pp 1489–1496
32. Wüthrich K (1989) The Development of Nuclear Magnetic Resonance Spectroscopy as a Technique for Protein Structure Determination. Acc Chem Res 22(1):36–44
33. Xiao L, Sun J, Boyd S (2006) A Duality View of Spectral Methods for Dimensionality Reduction. In: Proc. 23rd Int Conference Mach Learn (ICML 2006), pp 1041–1048

Greedy Randomized Adaptive Search Procedures

GPASP

MAURICIO G.C. RESENDE
Information Sci. Res., AT&T Labs Res.,
Florham Park, USA

MSC2000: 90-01, 90B40, 90C10, 90C35, 90C27, 94C15, 65H20, 65K05

Article Outline

Keywords

Combinatorial optimization; GRASP; Metaheuristics; Search heuristic

Optimization problems that involve a large finite number of alternatives often arise in industry, government and science. In these problems, one is given a finite solution set X and a real-valued function $f\colon X \to \mathbf{R}$, and one seeks a solution $x^* \in X$ with $f(x^*) \leq f(x)$, $\forall x \in X$. Common examples include designing efficient telecommunication networks and constructing cost effective airline crew schedules. To find the optimal solution in a com binatorial optimization problem it is theoretically possible to enumerate the solutions and evaluate each with respect to the stated objective. However, from a practical perspective, it is infeasible to follow such a strategy of complete enumeration because the number of combinations often grows exponentially with the size of problem.

Much work has been done over the last five decades to develop optimal seeking methods that do not explicitly require an examination of each alternative. This research has given rise to the field of *combinatorial optimization* (see [55]), and an increasing capability to solve ever larger real-world problems. Nevertheless, most problems found in industry and government are either computationally intractable by their nature, or sufficiently large so as to preclude the use of exact algorithms. In such cases, *heuristic methods* are usually employed to find good, but not necessarily guaranteed optimal solutions. The effectiveness of these methods depends upon their ability to adapt to a particular realization, avoid entrapment at local optima, and exploit the basic structure of the problem, such as a network or a natural ordering among its components. Furthermore, restart procedures, controlled randomization, efficient data structures, and preprocessing are also beneficial. Building on these notions, various heuristic search techniques have been developed that have demonstrably improved our ability to obtain good solutions to difficult combinatorial optimization problems. The most promising of such techniques include simulated annealing [35], tabu search [27,28,29], genetic algorithms [30] and GRASP (greedy randomized adaptive search procedures) [21,22].

In this article, we review GRASP. The components of a basic GRASP heuristic are addressed and enhancements proposed to the basic heuristic are discussed. The paper concludes with a brief literature review of applications of GRASP.

A Basic GRASP

A GRASP is a *multistart* or iterative process, in which each GRASP iteration consists of two phases, a *construction phase*, in which a feasible solution is produced, and a *local search phase*, in which a local optimum in the neighborhood of the constructed solution is sought. The best overall solution is kept as the result. The pseudocode below illustrates a GRASP procedure for minimization in which maxitr GRASP iterations are done.

```
$x^* = \infty$;
FOR $k = 1, \ldots,$ maxitr DO
    construct $(g(\cdot), \alpha, x)$;
    local $(f(\cdot), x)$;
    IF $f(x) < f(x^*)$ DO
        $x^* = x$;
    END IF;
END FOR
```

Procedure grasp($f(\cdot), g(\cdot)$, maxitr, x^*)

In the construction phase, a feasible solution is iteratively constructed, one element at a time. The basic GRASP construction phase is similar to the *semigreedy heuristic* proposed independently by J.P. Hart and A.W. Shogan [31]. At each construction iteration, the choice of the next element to be added is determined by ordering all candidate elements (i. e. those that can be added to the solution) in a candidate list C with respect to a *greedy function* $g\colon C \to \mathbf{R}$. This function measures the (myopic) benefit of selecting each element. The heuristic is adaptive because the benefits associated with every element are updated at each iteration of the construction phase to reflect the changes brought on by the selection of the previous element. The probabilistic component of a GRASP is characterized by randomly choosing one of the best candidates in the list, but not necessarily the top candidate. The list of best candidates is called the *restricted candidate list* (RCL). This choice technique allows for different solutions to be obtained at each GRASP iteration, but does not necessarily compromise the power of the adaptive greedy component of the method. Let $\alpha \in [0, 1]$ be a given parameter. The pseudocode below describes a basic GRASP construction phase.

```
$x = \emptyset$;
Initialize candidate set $C$;
WHILE $C \neq \emptyset$ DO
    $\underline{s} = \min\{g(t) : t \in C\}$;
    $\bar{s} = \max\{g(t) : t \in C\}$;
    RCL= $\{s \in C : g(s) \leq \underline{s} + \alpha(\bar{s} - \underline{s})\}$;
    Select $s$, at random, from the set RCL;
    $x = x \cup \{s\}$;
    Update candidate set $C$;
END WHILE
```

Procedure construct($g(\cdot)$, α, x)

The pseudocode shows that the parameter α controls the amounts of greediness and randomness in the algorithm. A value $\alpha = 0$ corresponds a *greedy construction* procedure, while $\alpha = 1$ produces *random construction*.

As is the case for many deterministic methods, the solutions generated by a GRASP construction are not guaranteed to be locally optimal with respect to simple neighborhood definitions. Hence, it is almost always beneficial to apply a local search to attempt to improve each constructed solution. A local search algorithm works in an iterative fashion by successively replacing the current solution by a better solution in the neighborhood of the current solution. It terminates when no better solution is found in the neighborhood. The *neighborhood structure* N for a problem P relates a solution s of the problem to a subset of solutions $N(s)$. A solution s is said to be *locally optimal* if there is no better solution in $N(s)$. The key to success for a local search algorithm consists of the suitable choice of a neighborhood structure, efficient neighborhood search techniques, and the starting solution.

While such local optimization procedures can require exponential time from an arbitrary starting point, empirically their efficiency significantly improves as the initial solution improves. Through the use of customized data structures and careful implementation, an efficient construction phase can be created which produces good initial solutions for efficient local search. The result is that often many GRASP solutions are generated in the same amount of time required for the local optimization procedure to converge from a single random start. Furthermore, the best of these GRASP solutions is generally significantly better than the single solution obtained from a random starting point. The pseudocode below describes a basic local search procedure.

```
$H = \{y \in N(x) : f(y) < f(x)\}$;
WHILE $|H| > 0$ DO
    Select $x \in H$;
    $H = \{y \in N(x) : f(y) < f(x)\}$;
END WHILE
```

Procedure local($f(\cdot)$, $N(\cdot)$, x)

It is difficult to formally analyze the quality of solution values found by using the GRASP methodology. However, there is an intuitive justification that views GRASP as a repetitive sampling technique. Each GRASP iteration produces a sample solution from an unknown distribution of all obtainable results. The mean and variance of the distribution are functions of the restrictive nature of the candidate list. For example, if the cardinality of the restricted candidate list is limited to one, then only one solution will be produced and the variance of the distribution will be zero. Given an effective greedy function, the mean solution value in this case should be good, but probably suboptimal. If a less restrictive cardinality limit is imposed, many different solutions will be produced implying a larger variance. Since the greedy function is more compromised in this case, the mean solution value should degrade. Intuitively, however, by order statistics and the fact that the samples are randomly produced, the best value found should outperform the mean value. Indeed, often the best solutions sampled are optimal.

An especially appealing characteristic of GRASP is the ease with which it can be implemented. Few parameters need to be set and tuned, and therefore development can focus on implementing efficient data structures to assure quick GRASP iterations. Finally, GRASP can be trivially implemented in parallel. Each processor can be initialized with its own copy of the procedure, the instance data, and an independent random number sequence. The GRASP iterations are then performed in parallel with only a single global variable required to store the best solution found over all processors.

Enhancements to the Basic GRASP

A number of enhancements to the basic GRASP, presented in the previous section, have been proposed in the literature. In this section we review the use path relinking, long-term memory, the proximate optimality principle, and bias functions in a GRASP. We discuss a parallelization scheme and the use of GRASP in hybrid metaheuristics.

Path Relinking

M. Laguna and R. Martí [43] adapted the concept of *path relinking* for use within a GRASP. To test their concept, they im plemented a GRASP with path relinking for the 2-layer straight line crossing minimization problem. A small set of high-quality, or elite, solutions is stored to serve as guiding solutions for path relinking. Each GRASP iteration produces a locally optimal solution x^*. A solution y^* is chosen at random from the elite set and a path of solutions linking x^* to y^* is constructed by applying a series of changes to the original solution. For example, let $x^* = (1, 0, 0, 0)$ and $y^* = (0, 1, 0, 1)$. A path relinking of x^* and y^* is $x^* = (1, 0, 0, 0) \rightarrow (0, 0, 0, 0) \rightarrow (0, 1, 0, 0) \rightarrow (0, 1, 0, 1) = y^*$. Each of these path solutions is evaluated for solution quality. Laguna and Martí report that often improvements to the incumbent are found in this path relinking.

Long-Term Memory

Long-term memory is the basis for tabu search. Besides path relinking, which can thought of as a form of long-term memory, other uses of long term memory have been proposed for use in a GRASP. C. Fleurent and F. Glover [26] observe the fact that the basic GRASP does not make use of information gathered in previous iterations and propose a long term memory scheme to address this issue. M. Prais and C.C. Ribeiro [64] propose a scheme to learn an appropriate value for the RCL parameter α.

Fleurent and Glover introduced a way to use long-term memory in multistart heuristics such as GRASP. Their scheme maintains a set S of elite solutions to be used in the construction phase. To become an elite solution a solution s must be either better than the best member of S, or better than the worst member of S and sufficiently different from the other elite solutions. For example, one can count identical solution vector components and set a threshold for rejection. A *strongly determined variable* is one that cannot be changed without eroding the objective or changing significantly other variables. A *consistent variable* is one that receives a particular value in a large portion of the elite solution set. Let $I(e)$ be a measure of the strongly determined and consistent features of choice e, i. e. $I(e)$ becomes larger as e resembles solutions in elite set S. The intensity function $I(e)$ is used in the construction phase as follows. Recall that $g(e)$ is the greedy function. Let $E(e) = F(g(e), I(e))$ be a function of the greedy and the intensification functions. For example, $E(e) = \lambda\ g(e) + I(e)$. The intensification scheme biases selection from the RCL to those elements e with a high value of $E(e)$ by setting the probability of selecting e to be $p(e) = E(e)/ \sum_{s \in \mathrm{RCL}} E(s)$. The function $E(e)$ can vary with time by changing the value of λ, e. g. initially λ is set to a large value and when diversification is called for, λ is decreased. A procedure for changing the value of λ is given by Fleurent and Glover. See also [11] for an application of this long-term memory strategy.

Reactive GRASP

The term ‘reactive GRASP’ was introduced by Prais and Ribeiro [64] for a GRASP that reacts to solutions produced by different settings of the RCL parameter α and seeks to adjust α to give the GRASP an appropriate level of greediness and randomness. At each GRASP iteration, the value of α is chosen from a discrete set of values $\{\alpha_1, \ldots, \alpha_m\}$. The probability of selecting the value α_k is $p(\alpha_k)$, for $k = 1, \ldots, m$. Reactive GRASP adaptively changes the probabilities $\{p(\alpha_1), \ldots, p(\alpha_m)\}$ to favor values that produce good solutions. Consider applying Reactive GRASP to a minimization problem. Initially the probabilities are set as $p(\alpha_k) = 1/m$, for $i = 1, \dots, m$, so that the values are selected uniformly. To adaptively redefine the probabilities, define $F(S^*)$ to be the value of the best solution found so far and let A_i be the average value of the solutions obtained with α_i. Prais and Ribeiro propose a period of warm-up iterations to initialize the A_i values. Periodically (say every N_α iterations) the quantities $q_i = (F(S^*)/A_i)^\delta$ are computed for $i = 1, \dots, m$ and the probabilities are updated to $p(\alpha_i) = q_i / \sum_{j=1}^{m} q_j$, for $i = 1, \dots, m$. Observe that the more suitable a value α_i is, the larger the value of q_i is and, consequently, the higher the value of $p(\alpha_i)$, making α_i more likely to be selected. The parameter δ can be used as an attenuation parameter. See also [16] for an application of reactive GRASP.

Proximate Optimality Principle

The *proximate optimality principal* is based on the idea that 'good solutions at one level are likely to be found close to good solutions at an adjacent level' [29]. Fleurent and Glover [26] provide a GRASP interpretation of this principle. They suggest that imperfections introduced during steps of GRASP construction can be 'ironed-out' by applying local search during (and not only at the end of) GRASP construction. Because of efficiency considerations, a practical implementation of POP to GRASP is to apply local search during a few points in the construction phase and not during each construction iteration. See also [11] for an application of the proximate optimality principle.

Global Convergence

In [52] it was pointed out that GRASP with a fixed nonzero RCL parameter α is not asymptotically convergent to a global optimum. During construction, a fixed RCL parameter may rule out a candidate that is present in all optimal solutions. Several remedies have been proposed to get around this problem. The most straightforward is the use of a randomly selected α [72]. In this approach, the parameter is selected at random from the continuous interval [0, 1] at the start of each GRASP iteration. That value is used during the entire iteration. Since a subset of the iterations are random, the algorithm becomes asymptotically globally convergent. Reactive GRASP, as described above, can also be made asymptotically globally convergent by making $\alpha_m = 1$, i. e. allowing the choice of a value that produces a random GRASP iteration. J.L. Bresina [13] introduced the concept of a *bias function* to select a candidate element to be included in the solution. Bresina's method, which is directly applicable to GRASP construction, also allows for purely random construction and is therefore asymptotically globally convergent. At each construction step, the elements in the candidate set C are ranked by their greedy function values. A bias value bias(r) is assigned to the rth ranked element. Bresina proposes several bias functions. In logarithmic bias, bias(r) = $1/\log(r + 1)$. In linear bias, bias(r) = $1/r$. In polynomial bias of order n, bias(r) = $1/r^n$. In exponential bias, bias(r) = $1/e^r$. Finally, in random bias, bias(r) = 1. During construction, the probability of selecting the rth ranked candidate is bias(r) / $\sum_{i=1}^{|C|}$ bias(i). See also [11] for an application of this bias function strategy.

Parallel GRASP

Parallel implementation of GRASP is straightforward. Two general strategies have been proposed. In search space decomposition, the search space is partitioned into several regions and GRASP is applied to each in parallel. An example of this is the GRASP for maximum independent set [23,69] where the search space is decomposed by fixing two vertices to be in the independent set. In iteration parallelization, the GRASP iterations are partitioned and each partition is assigned to a processor. See [54,56,57,58,67] for examples of parallel implementations of GRASP. Some care is needed so that different random number generator seeds are assigned to the different iterations. This can be done by running the random number generator through an entire cycle, recording all N_g seeds in a seed array. Iteration i is started with seed(i). GRASP has been implemented on distributed architectures. In [58] a *PVM-based implementation* is described. Two *MPI-based implementations* are given in [4,50]. A.C.F. Alvim [4] proposes a general scheme for MPI implementations. A master process manages seeds for slave processors. It passes blocks of seeds to each slave processor and awaits the slaves to indicate that they have finished processing the block and need another block. Slaves also pass back to the master the best solution found for each block of iterations.

GRASP in Hybrid Metaheuristics

GRASP has been used in *hybrid metaheuristic* schemes. Laguna and J.L. González-Velarde [41] proposed a GRASP in which local search is done by tabu search. See also [16,46] for implementations of GRASP using tabu search as the local search procedure. Simulated annealing can also be used as a GRASP local search procedure if the initial temperature is low so that it remains near the neighborhood of the constructed solution. R.K. Ahuja, J.B. Orlin and A. Tiwari [3] use GRASP construction as a mechanism for generating the initial population in a genetic algorithm. GRASP is used in [45] in a genetic algorithm to implement a type of crossover called *perfect offspring*.

Applications of GRASP

We now turn our attention to a number of GRASP implementations that have appeared in the literature, covering a wide range of applications. An early tutorial on GRASP appears in [22]. We group the work into two categories, applications to operations research problems and to industrial applications.

Operations Research Problems

Applications of GRASP to operations research problems can be classified into eight categories: scheduling problems, routing problems, logic, partitioning problems, location problems, graph theoretic problems, assignment problems, and nonconvex network flow problems.

GRASP has been applied to several scheduling problems, including operations sequencing in discrete parts manufacturing [7], flight scheduling [18], just-in-time scheduling in parallel machines [41], printed wire assembly scheduling [9,19], single machine scheduling with sequence dependent setup costs and delay penalties [24], field technician scheduling [79], flowshop with setup costs [76,77], and bus-driver scheduling [45].

Applications of GRASP to routing problems include vehicle routing with time windows [38], vehicle routing [32], aircraft routing [5], inventory routing problem with satellite facilities [10], and permanent virtual circuit (PVC) routing [66].

Problems in logic have been approached with GRASP. These include the satisfiability problem [68], maximum satisfiability [58,71,72], and inference of logical clauses from examples [15].

GRASP has been applied to partitioning problems, including graph two partition [40] and number partitioning [6].

Applications of GRASP to location problems include p-hub location [36], pure integer capacitated plant location [14], location with economies of scale [33], single source capacitated plant location [16], location of concentrators in network access design [74], and maximum covering [67].

GRASP has been used for finding approximate solutions to a number of graph theoretic problems, including set covering [21], maximum independent set [23,69], maximum clique with weighted edges [48], graph planarization [73,75], 2-layer straight line crossing minimization [43], sparse graph coloring [42], maximum weighted edge subgraph [47], the Steiner tree problem in graphs [49,50], feedback vertex set in directed graphs [60], maximum clique [1,61], and the capacitated minimum spanning tree problem [2].

Several assignment problems have been approached with GRASP. A GRASP was introduced for the quadratic assignment problem in [44]. A parallel version of this GRASP is described in [57]. Fortran subroutines for dense and sparse quadratic assignment problems can be found respectively in [70] and [59]. A modified local search for the GRASP for quadratic assignment problems is proposed in [65]. GRASP has been used to generate the initial population of a genetic algorithm for the quadratic assignment problem [3]. Long term memory schemes have been adapted to a GRASP for the quadratic assignment problem in [26]. A GRASP for the biquadratic assignment problem is described in [51]. GRASP has been applied to two multidimensional assignment problems [53,78] and to the radio link frequency assignment problem [62]. A GRASP for the generalized assignment problem was proposed in [46].

GRASP has been used for finding approximate solutions to a concave-cost network flow problem [34].

Industrial Applications

Industrial applications of GRASP can be classified into seven categories: manufacturing, transportation, telecommunications, automatic drawing, electrical power systems, military, and biology.

GRASP has been applied to several manufacturing problems, including operations sequencing in discrete parts manufacturing [7], cutting path and tool selection in computer-aided process planning [17], manufacturing equipment selection [8], component grouping [37], and printed wire assembly scheduling [9,19].

Applications of GRASP in transportation include flight scheduling and maintenance base planning [18], intermodal trailer assignment [20], and aircraft routing in response to groundings and delay [5].

In telecommunications, GRASP has been applied to the design of SDH mesh-restorable networks [63], the Steiner tree problem in graphs [49,50], permanent virtual circuit (PVC) routing [66], location of concentrators in network access design [74], traffic scheduling in satellite switched time division multi-access (SS/TDMA) systems [64], location of points of presence (PoPs) [67], and to the multicriteria radio link frequency assignment problem [62].

GRASP has been applied to automatic drawing problems, including seam drawing in mosaicing of aerial photographic maps [25], graph planarization [73,75], and 2-layer straight line crossing minimization [43].

GRASP has been applied to other industrial problems. An application to *electrical power systems* is transmission expansion planning [12]. A military application of GRASP is in multitarget multisensor tracking [53]. GRASP has been applied in biology for protein structure prediction [39].

Conclusion

We have surveyed the literature on greedy randomized adaptive search procedures (GRASP) in the 1990s. In these years many enhancements to the basic GRASP introduced in 1988 have been proposed. The number and variety of applications has grown and continues to grow.

See also

- ▶ Feedback Set Problems
- ▶ Generalized Assignment Problem
- ▶ Graph Coloring
- ▶ Graph Planarization
- ▶ Heuristics for Maximum Clique and Independent Set
- ▶ Maximum Satisfiability Problem
- ▶ Quadratic Assignment Problem
- ▶ Quadratic Semi-assignment Problem

References

1. Abello J, Pardalos PM, Resende MGC (1999) On maximum clique problems in very large graphs. In: Abello J, Vitter J (eds) External memory algorithms and visualization. 50DIMACS, Amer Math Soc, pp 119–130
2. Ahuja RK, Orlin JB, Sharma D (1998) New neighborhood search structures for the capacitated minimum spanning tree problem. Techn Report Dept ISE Univ Florida
3. Ahuja RK, Orlin JB, Tiwari A (2000) A greedy genetic algorithm for the quadratic assignment problem. Comput Oper Res 27:917–934
4. Alvim ACF (Apr. 1998) Parallelization strategies for the metaheuristic GRASP. MSc Thesis Dept Computer Sci Catholic Univ Rio de Janeiro
5. Argüello MF, Bard JF, Yu G (1997) A GRASP for aircraft routing in response to groundings and delays. J Combin Optim 1:211–228
6. Argüello MF, Feo TA, Goldschmidt O (1996) Randomized methods for the number partitioning problem. Comput Oper Res 23(2):103–111
7. Bard JF, Feo TA (1989) Operations sequencing in discrete parts manufacturing. Managem Sci 35:249–255
8. Bard JF, Feo TA (1991) An algorithm for the manufacturing equipment selection problem. IIE Trans 23:83–92
9. Bard JF, Feo TA, Holland S (1996) A GRASP for scheduling printed wiring board assembly. IIE Trans 28:155–165
10. Bard JF, Huang L, Jaillet P, Dror M (1998) A decomposition approach to the inventory routing problem with satellite facilities. Transport Sci 32:189–203
11. Binato S, Hery WJ, Loewenstern D, Resende MGC (1999) Approximate solution of the job shop scheduling problem using GRASP. Techn Report AT&T Lab Res
12. Binato S, Oliveira GC, Araújo JL (1998) A greedy randomized adaptive search procedure for transmission expansion planning. IEEE Trans Power Systems
13. Bresina JL (1996) Heuristic-biased stochastic sampling. In: Proc. AAAI-96, pp 271–278

14. Delmaire H, Díaz JA, Fernández E, Ortega M (1997) Comparing new heuristics for the pure integer capacitated plant location problem. Techn Report Dept Statist and Oper Res Univ Politecn Catalunya, Barcelona, no. DR97/10
15. Deshpande AS, Triantaphyllou E (1998) A greedy randomized adaptive search procedure (GRASP) for inferring logical clauses from examples in polynomial time and some extensions. Math Comput Modelling 27:75–99
16. Díaz JA, Fernández E (1998) A hybrid GRASP-tabu search algorithm for the single source capacitated plant location problem. Techn Report Dept Statist and Oper Res Univ Politecn Catalunya, Barcelona
17. Feo TA, Bard JF (1989) The cutting path and tool selection problem in computer-aided process planning. J Manufacturing Systems 8:17–26
18. Feo TA, Bard JF (1989) Flight scheduling and maintenance base planning. Managem Sci 35:1415–1432
19. Feo TA, Bard J, Holland S (1995) Facility-wide planning and scheduling of printed wiring board assembly. Oper Res 43:219–230
20. Feo TA, González-Velarde JL (1995) The intermodal trailer assignment problem: Models, algorithms, and heuristics. Transport Sci 29:330–341
21. Feo TA, Resende MGC (1989) A probabilistic heuristic for a computationally difficult set covering problem. Oper Res Lett 8:67–71
22. Feo TA, Resende MGC (1995) Greedy randomized adaptive search procedures. J Global Optim 6:109–133
23. Feo TA, Resende MGC, Smith SH (1994) A greedy randomized adaptive search procedure for maximum independent set. Oper Res 42:860–878
24. Feo TA, Sarathy K, McGahan J (1996) A GRASP for single machine scheduling with sequence dependent setup costs and linear delay penalties. Comput Oper Res 23:881–895
25. Fernández E, Martí R (1999) GRASP and tabu search for seam drawing in mosaicking of aerial photographic maps. J Heuristics 5:181–197
26. Fleurent C, Glover F (1999) Improved constructive multistart strategies for the quadratic assignment problem using adaptive memory. INFORMS J Comput 11:198–204
27. Glover F (1989) Tabu search – Part I. ORSA J Comput 1:190–206
28. Glover F (1990) Tabu search – Part II. ORSA J Comput 2:4–32
29. Glover F, Laguna M (1997) Tabu search. Kluwer, Dordrecht
30. Goldberg DE (1989) Genetic algorithms in search, optimization and machine learning. Addison-Wesley, Reading
31. Hart JP, Shogan AW (1987) Semi-greedy heuristics: An empirical study. Oper Res Lett 6:107–114
32. Hjorring CA (1995) The vehicle routing problem and local search metaheuristics. PhD Thesis, Univ. Auckland
33. Holmqvist K, Migdalas A, Pardalos PM (1997) Greedy randomized adaptive search for a location problem with economies of scale. In: Bomze IM et al (eds) Developments in Global Optimization. Kluwer, Dordrecht, pp 301–313
34. Holmqvist K, Migdalas A, Pardalos PM (1998) A GRASP algorithm for the single source uncapacitated minimum concave-cost network flow problem. In: Pardalos PM, Du D-Z (eds) Network design: Connectivity and facilities location. DIMACS 40. Amer. Math. Soc., Providence, pp 131–142
35. Kirkpatrick S (1984) Optimization by simulated annealing: Quantitative studies. J Statist Phys 34:975–986
36. Klincewicz JG (1992) Avoiding local optima in the p-hub location problem using tabu search and GRASP. Ann Oper Res 40:283–302
37. Klincewicz JG, Rajan A (1994) Using GRASP to solve the component grouping problem. Naval Res Logist 41:893–912
38. Kontoravdis G, Bard JF (1995) A GRASP for the vehicle routing problem with time windows. ORSA J Comput 7:10–23
39. Krasnogor N, Pelta DA, Russo W, Terrazas G (1998) A GRASP approach to the protein structure prediction problem. Techn Report LIFIA Lab Univ La Plata
40. Laguna M, Feo TA, Elrod HC (1994) A greedy randomized adaptive search procedure for the two-partition problem. Oper Res 42:677–687
41. Laguna M, González-Velarde JL (1991) A search heuristic for just-in-time scheduling in parallel machines. J Intelligent Manufacturing 2:253–260
42. Laguna M, Martí R (1998) A GRASP for coloring sparse graphs. Techn Report Graduate School Business, Univ Colorado
43. Laguna M, Martí R (1999) GRASP and path relinking for 2-layer straight line crossing minimization. INFORMS J Comput 11:44–52
44. Li Y, Pardalos PM, Resende MGC (1994) A greedy randomized adaptive search procedure for the quadratic assignment problem. In: Pardalos PM, Wolkowicz H (eds) Quadratic Assignment and Related Problems. DIMACS 16. Amer. Math. Soc., Providence, pp 237–261
45. Lourenço H Ramalhinho, Paixao JP, Portugal R (1998) Metaheuristics for the bus-driver scheduling problem. Techn Report Dept Economics and Management Univ Pompeu Fabra, Barcelona
46. Lourenço H Ramalhinho, Serra D (May 1998) Adaptive approach heuristics for the generalized assignment problem. Techn Report Dept Economics and Management Univ Pompeu Fabra, Barcelona
47. Macambira EM, Meneses CN (1998) A GRASP algorithm for the maximum weighted edge subgraph problem. Techn Report Dept Statist and Computation Univ Ceará, Fortaleza, CE 60740-000
48. Macambira EM, Souza CC de (Oct. 1997) A GRASP for the maximum clique problem with weighted edges. In: Proc. XXIX Brazilian Symp. Operations Research, p 70 In Portuguese

49. Martins SL, Pardalos PM, Resende MGC, Ribeiro CC (1999) Greedy randomized adaptive search procedures for the Steiner problem in graphs. In: Pardalos PM, Rajasejaran S, Rolim J (eds) Randomization methods in algorithmic design. DIMACS 43. Amer. Math. Soc., Providence, pp 133–145
50. Martins SL, Ribeiro CC (1998) A parallel GRASP for the Steiner problem in graphs. Proc. Irregular'98, In: Lecture Notes Computer Sci, vol 1457. Springer, Berlin, pp 285–297
51. Mavridou T, Pardalos PM, Pitsoulis LS, Resende MGC (1997) A GRASP for the biquadratic assignment problem. Europ J Oper Res 105:613–621
52. Mockus J, Eddy E, Mockus A, Mockus L, Reklaitis GV (1997) Bayesian discrete and global optimization. Kluwer, Dordrecht
53. Murphey RA, Pardalos PM, Pitsoulis LS (1998) A greedy randomized adaptive search procedure for the multitarget multisensor tracking problem. In: Pardalos PM, Du D-Z (eds) Network design: Connectivity and facilities location. DIMACS 40. Amer. Math. Soc., Providence, pp 277–301
54. Murphey RA, Pardalos PM, Pitsoulis LS (1998) A parallel GRASP for the data association multidimensional assignment problem. In: Pardalos PM (ed) Parallel processing of discrete problems. IMA vol Math Appl, vol 106. Springer, Berlin, pp 159–180
55. Papadimitriou CH, Steiglitz K (1982) Combinatorial optimization: Algorithms and complexity. Prentice-Hall, Englewood Cliffs
56. Pardalos PM, Pitsoulis L, Mavridou T, Resende MGC (1995) Parallel search for combinatorial optimization: Genetic algorithms, simulated annealing and GRASP. In: Ferreira A, Rolim J (eds) Parallel Algorithms for Irregularly Structured Problems, Proc. 2nd Internat. Workshop –Irregular'95, Lecture Notes Computer Sci. Springer, Berlin, pp 317–331
57. Pardalos PM, Pitsoulis LS, Resende MGC (1995) A parallel GRASP implementation for the quadratic assignment problem. In: Ferreira A, Rolim J (eds) Parallel Algorithms for Irregularly Structured Problems – Irregular'94. Kluwer, Dordrecht, pp 111–130
58. Pardalos PM, Pitsoulis LS, Resende MGC (1996) A parallel GRASP for MAX-SAT problems. In: Lecture Notes Computer Sci, vol 1180. Springer, Berlin, pp 575–585
59. Pardalos PM, Pitsoulis LS, Resende MGC (1997) Algorithm 769: Fortran subroutines for approximate solution of sparse quadratic assignment problems using GRASP. ACM Trans Math Softw 23:196–208
60. Pardalos PM, Qian T, Resende MGC (1998) A greedy randomized adaptive search procedure for the feedback vertex set problem. J Combin Optim 2(3)
61. Pardalos PM, Resende MGC, Rappe J (1998) An exact parallel algorithm for the maximum clique problem. In: De Leone R et al (eds) High performance algorithms and software in nonlinear optimization. Kluwer, Dordrecht, pp 279–300
62. Pasiliao EL (1998) A greedy randomized adaptive search procedure for the multi-criteria radio link frequency assignment problem. Techn Report Dept ISE Univ Florida
63. Poppe F, Pickavet M, Arijs P, Demeester P (1997) Design techniques for SDH mesh-restorable networks. Proc. European Conf. Networks and Optical Communications (NOC'97), Volume 2: Core and ATM Networks, pp 94–101
64. Prais M, Ribeiro CC (2000) Reactive GRASP: An application to a matrix decomposition problem in TDMA traffic assignment. INFORMS J Comput 12(3):164–176
65. Rangel MC, Abreu NMM de, Boaventura-Netto PO, Boeres MCS (1998) A modified local search for GRASP in the quadratic assignment problem. Techn Report Production Engin Program, COPPE, Federal Univ Rio de Janeiro
66. Resende LIP, Resende MGC (1997) A GRASP for frame relay PVC routing. Techn Report AT&T Lab Res
67. Resende MGC (1998) Computing approximate solutions of the maximum covering problem using GRASP. J Heuristics 4:161–171
68. Resende MGC, Feo TA (1996) A GRASP for satisfiability. In: Johnson DS, Trick MA (eds) Cliques, Coloring and Satisfiability: The Second DIMACS Implementation Challenge. DIMACS 26. Amer. Math. Soc., Providence, pp 499–520
69. Resende MGC, Feo TA, Smith SH (1998) Fortran subroutines for approximate solution of maximum independent set problems using GRASP. ACM Trans Math Softw 24:386–394
70. Resende MGC, Pardalos PM, Li Y (1996) Algorithm 754: Fortran subroutines for approximate solution of dense quadratic assignment problems using GRASP. ACM Trans Math Softw 22:104–118
71. Resende MGC, Pitsoulis LS, Pardalos PM (1997) Approximate solution of weighted MAX-SAT problems using GRASP. In: Gu J, Pardalos PM (eds) Satisfiability problems. DIMACS 35. Amer. Math. Soc., Providence, pp 393–405
72. Resende MGC, Pitsoulis LS, Pardalos PM (2000) Fortran subroutines for computing approximate solutions of MAX-SAT problems using GRASP. Discrete Appl Math 100:95–113
73. Resende MGC, Ribeiro CC (1997) A GRASP for graph planarization. Networks 29:173–189
74. Resende MGC, Ulular O (1997) SMART: A tool for AT&T Worldnet access design – Location of Cascade 9000 concentrators. Techn Report AT&T Lab Res
75. Ribeiro CC, Resende MGC (1999) Algorithm 797: Fortran subroutines for approximate solution of graph planarization problems using GRASP. ACM Trans Math Softw 25:341–352
76. Ríos-Mercado RZ, Bard JF (1998) Heuristics for the flow line problem with setup costs. Europ J Oper Res 110:76–98
77. Ríos-Mercado RZ, Bard JF (1999) An enhanced TSP-based heuristic for makespan minimization in a flow shop with setup costs. J Heuristics 5:57–74

78. Robertson AJ (1998) A set of greedy randomized adaptive local search procedure (GRASP) implementations for the multidimensional assignment problem
79. Xu J, Chiu S (1996) Solving a real-world field technician scheduling problem. Proc. Internat. Conf. Management Sci. and the Economic Development of China, Hong-Kong, July 1996. pp 240–248

Gröbner Bases for Polynomial Equations

P. O. Lindberg[1], Lars Svensson[2]
[1] Linköping University, Linköping, Sweden
[2] KTH, Stockholm, Sweden

MSC2000: 12D10, 12Y05, 13P10

Article Outline

Keywords

Polynomial equations; Zeros; Gröbner basis

Polynomial equations (in several variables) arise in many areas connected to management science. They could describe the feasible set of an optimization problem, the Karush–Kuhn–Tucker conditions for the same problem, or maybe constraints on the positions of the links of a robot arm in a flexible manufacturing system.

There are many analogies between polynomial equations and their special case, linear equations.

- One might want to solve the equations, i. e. find one or all solutions, determine whether a solution is unique or determine whether the system in inconsistent.
- One might want to answer more abstract questions, such as whether a given equation is a consequence of a given set of equations (cf. ► Farkas lemma; ► Farkas lemma: Generalizations).

For linear equations a fundamental concept is that of a (linear) basis and the fundamental tool is that of Gaussian elimination, by which one can construct a basis from a given set of vectors. Similarly, for polynomials there is the corresponding concepts of a *Gröbner basis* and the *Buchberger algorithm*, which for a given set of polynomials constructs a Gröbner basis. In particular one can convert a system of polynomial equations to triangular form, which allows for a solution by back substitution. In Gaussian elimination, the variables/columns have an ordering that influences the end result. Similarly, for Gröbner bases we need an order, not only for the variables, but for *monomials*, i. e. the simplest possible polynomials, such as $x_1^3x_4$, that are products of variables. In this short note we will review Gröbner basis for polynomial equations.

Before defining a Gröbner base we will give an example.

Example 1 Suppose we want to find the local optima of the following optimization problem ([4, Problem 337]; also used in [3]), by solving the KKT-conditions:

$$(P)\begin{cases}\min & f(x) = 9x_1^2 + x_2^2 + 9x_3^2\\ \text{s.t.} & g_1(x) = 1 - x_1x_2 \le 0\\ & g_2(x) = 1 - x_2 \le 0\\ & g_3(x) = x_3 - 1 \le 0.\end{cases}$$

The KKT conditions for (P) are:

$$(KKT)\begin{cases}18x_1 - \lambda_1x_2 = 0\\ 2x_2 - \lambda_1x_1 - \lambda_2 = 0\\ 18x_3 + \lambda_3 = 0\\ \lambda_1(1 - x_1x_2) = 0\\ \lambda_2(1 - x_2) = 0\\ \lambda_3(x_3 - 1) = 0.\end{cases}$$

Further suppose we use a *lexicographical order* of the monomials such that $x_1 > x_2 > x_3 > \lambda_1 > \lambda_2 > \lambda_3$. Then, computing the Gröbner basis for the set of polynomials in the above system and forming the corresponding equation system, we get

$$\begin{cases} 18x_1 - x_2\lambda_1 & = 0 \\ x_2\lambda_1 - 36x_3 + 18\lambda_2 & = 0 \\ x_2\lambda_2 - \lambda_2 & = 0 \\ 2x_2 - \lambda_1 - \lambda_2 & = 0 \\ 18x_3 - \lambda_3 & = 0 \\ \lambda_1^3 - 36\lambda_1 - 18\lambda_2^2 + 36\lambda_2 & = 0 \\ \lambda_1\lambda_2 + \lambda_2^2 - 2\lambda_2 & = 0 \\ \lambda_2^3 + 14\lambda_2^2 - 32\lambda_2 & = 0 \\ \lambda_3^2 + 18\lambda_3 & = 0. \end{cases}$$

This system has an obvious triangular structure, that we have tried to display graphically. The last equation contains only λ_3. Then comes equations in λ_2 (and possibly λ_3) and so on. In a similar way as in Gaussian elimination, the system can thus be solved by back substitution. In each step, one then has to solve a single variable polynomial equation, giving possibly several solutions, each of which is substituted into the preceding equations. Thus the solution process evolves in a tree-like structure. It might happen, that one has to solve for a variable that is already computed. Then of course the solutions have to agree, else they are discarded.

The above type of structure will always occur if there are finitely many solutions. It might happen, though, that the system allows a manifold of solutions. In this case it might e. g. happen that the last equation contains two variables or that you in the back substitution process comes to an equation with two (or more) undetermined variables. These equations then give a parametrization of the manifold.

What is a Gröbner Basis

In Gaussian elimination the variables are ordered and the basic reduction rule is to replace the equations $f = 0$, $g = 0$ by $f = 0$, $g - cf = 0$ where the constant c is chosen so that the leading terms in g and cf coincide.

In systems of polynomial equations we do something quite similar. First we extend the ordering of the variables to a total ordering of all monomials in a way such that $m' < \mathrm{m}'' \Rightarrow mm' < mm''$ for all monomials m, m' and m'' and so that 1 is the least one.

The basic reduction rule is now to replace the equations $f = 0$, $g = 0$ by $f = 0$, $g - cmf = 0$ where the constant c and the monomial m are chosen so that the leading terms of g and cmf coincide. This implies that $h = g - cmf$ is 'smaller' than g in the ordering. If such a reduction of g with f is possible and $h = g - cmf$ we will write $g \rightarrow_f h$.

Definition 2 A finite set G of polynomials is a *Gröbner basis* if for every polynomial q there exist a *unique* r and a finite reduction chain $q \rightarrow_{g_1} q_1 \rightarrow_{g_2} \cdots \rightarrow_{g_k} q_k = r$ for some $g_1, \ldots, g_k$ in G and such that r cannot be reduced further. The unique polynomial r is called the *normal form* of q modulo G.

Given a finite set of vectors we can use Gaussian elimination to compute a basis of vectors spanning the same linear space. Given a finite set P of polynomials (and an admissible monomial ordering), one can use the Buchberger algorithm to compute a Gröbner basis G, spanning the same 'space' of polynomials as P. (By the space of polynomial spanned by P is meant the *ideal* generated by P, i. e. the set of finite linear combinations $q_1p_1 + \cdots + q_sp_s$ where the p_i-s are in P and the q_i-s are arbitrary polynomials.) We say that G is a Gröbner basis for P. Moreover, the common zeros of P are the same as those of G.

What are Gröbner Bases good for

Roughly speaking, all questions concerning a system of polynomial equations $f_1 = \cdots = f_s = 0$ can be answered if we have a corresponding Gröbner basis. Here we list just a few of them.

- Is the system solvable?
- If the system is solvable, how many solutions are there, and which are they?
- How many real solutions are there? (in case the coefficients are real). Here we can also allow for inequalities.
- Is it possible to eliminate some of the variables?
- Given some polynomial f, does f vanish whenever $f_1 \cdots f_s$ does? This can be used for automated proofs in geometry.
- Given some polynomial f, does there exist polynomials $q_1, \ldots, q_s$ such that $f = q_1f_1 + \cdots + q_sf_s$?
- Is it possible to describe the algebraic relations between the f_i-s, i. e. the set of polynomials q in s variables such that $q(f_1, \ldots, f_s)$ is the zero polynomial.

- Can a given polynomial f be written as $f = q(f_1, \dots, f_s)$ for some polynomial q in s variables, and in case it can, is it possible to compute q?
- Can we compute a vector space basis for the vector space of polynomials modulo $f_1 \cdots f_s$?

Using Gröbner Bases and Learning more About them

Essentially all major mathematical computer packages with symbolic capabilities contain modules for Gröbner bases. The main examples are Maple and Mathematica. For a short but more detailed introduction to Gröbner bases, see [3]. The book [2] gives a rather short introduction to the field. One standard textbook is [1]

See also

- ► Contraction-mapping
- ► Fundamental Theorem of Algebra
- ► Global Optimization Methods for Systems of Nonlinear Equations
- ► Interval Analysis: Systems of Nonlinear Equations
- ► Nonlinear Least Squares: Newton-type Methods
- ► Nonlinear Systems of Equations: Application to the Enclosure of All Azeotropes

References

1. Cox D, O'Shea D (1992) Ideals, varieties and algorithms. An Introduction to computational algebraic geometry and commutative algebra. Springer, Berlin
2. Fröberg R (1997) An introduction to Gröbner bases. Wiley, New York
3. Hägglöf K, Lindberg PO, Svensson L (1995) Computing global minima to polynomial optimization problems using Gröbner bases. J Global Optim 7:115–125
4. Schittkowski K (1987) More test examples for nonlinear programming codes. Springer, Berlin