Journal of Global Optimization **6:** 87–105, 1995.

A Method for Solving D.C. Programming Problems. Application to Fuel Mixture Nonconvex Optimization Problem

THAI QUYNH PHONG, PHAM DINH TAO and LE THI HOAI AN
LMI-INSA Rouen, CNRS, URA 1378, B.P. 08, 76131, Mt St Aignan Cedex, France

(Received: 10 July 1992; accepted: 2 August 1994)

Abstract. We present a numerical method for solving the d.c. programming problem

$$c^* = \min\{\langle c, x\rangle \quad \text{s.t.} \quad f_i(x) \leq 0,\ i = 1, \ldots, m,\ x \in D\}$$

where $f_i\ i = 1, \ldots, m$ are d.c. (difference of two convex functions) and D is a convex set in $\mathbb{R}^n$. An (ε, η)-solution $x(\varepsilon, \eta)$ satisfying

$$x(\varepsilon, \eta) \in D, \quad \langle c, x(\varepsilon, \eta)\rangle \leq c^* + \varepsilon, \quad f_i(x(\varepsilon, \eta)) \leq \eta, \quad i = 1, \ldots, m,$$

can be found after a finite number of iterations. This algorithm combines an outer approximation procedure for solving a system of d.c. inequalities with a simple general scheme for minimizing a linear function over a compact set. As an application we discuss the numerical solution of a fuel mixture problem (encountered in the oil industry).

Key words: Nonlinear programming, global optimization, d.c. programming, outer approximation, system of d.c. inequalities, (ε, η)-solution, fuel mixture problem.

1. Introduction

In this paper we consider the multiextremal global optimization problem

$$(DCP) \quad \begin{cases} \min\langle c, x\rangle \\ \text{s.t. } f_i(x) \leq 0, \quad i = 1, \ldots, m, \\ \quad x \in D, \end{cases}$$

where $f_i\ i = 1, \ldots, m$ are d.c. functions (i.e. functions that can be expressed as a difference of two convex functions) on $\mathbb{R}^n$ and D is a compact convex set.

Problem *(DCP)* is important both from a practical and a theoretical point of view (see Global Optimization: Deterministic Approaches by Horst–Tuy [21]). D.c. functions have begun to be studied since a long time but have been introduced in optimization only recently, see e.g., the survey in Hiriart–Urruty [13], Tuy [36]. An important property of this class of functions is its stability with respect to operations frequently involved in optimization such as taking the pointwise maximum or pointwise minimum of a finite number of functions, etc. (Tuy [36]). Examples of d.c. functions often encountered in practice include convex, concave functions and indefinite quadratic forms.

Many economic models give rise to d.c. functions in the objective function as well as in the constraints (e.g., Hillestad [10], Zalessky [39]). In particular, the fuel mixture problem (cf. Pham Dinh–El Bernoussi [30]) which will be considered in Section 5 of this paper is of this type. D.c. programming problems are also encountered in engineering and physics (cf., e.g., Giannessi *et al.* [9], Nguyen *et al.* [26][25], Tuy [36][37], Vigidal–Director [38], Floudas–Visweswaran [7]). An interesting example of large scale d.c. programming problems is the Multidimensional Scaling problem which has been studied in Chine [5], where the subgradient method of Pham Dinh Tao was successfully applied (cf. also [29]).

To solve *(DCP)*, we note that, at the expense of an additional variable, this problem can be transformed into a canonical d.c. program

$$(CDCP) \quad \begin{cases} \min\langle c, x\rangle \\ \text{s.t. } h(x) \leq 0 \\ \quad\; g(x) \geq 0 \end{cases}$$

where $h, g : \mathbb{R}^{n+1} \to \mathbb{R}$ are convex functions (cf. [36]). The class of canonical d.c. programs was studied by many authors (e.g, [1],[2], [8], [12], [11], [19], [30], [27], [33]). Currently known algorithms for solving (CDCP) are either of branch and bound (Muu [23]) or outer approximation types (Tuy [35], Tuy and Thuong [34], Thoai [33]). Algorithms based on outer approximation and cutting plane techniques were proposed in Pham Dinh–El Bernoussi [30]. As noted in Horst *et al.* [19], the pure branch and bound or pure outer approximation algorithms are numerically expensive. For problems of large size, a more promising approach is to combine branch and bound with outer approximation. A first algorithm of this kind was developed in Horst–Thoai–Benson [16] for the concave minimization problem; later this approach was extended to (CDCP) in Horst–Phong–Thoai [19].

In general, finding an exact global solution is computationally expensive. However, in practice, given $\varepsilon > 0$ and $\eta > 0$, we can consider problem *(DCP)* solved if a vector $x(\varepsilon, \eta)$ has been found such that

$$x(\varepsilon, \eta) \in D, \quad \langle c, x(\varepsilon, \eta)\rangle - c^* \leq \varepsilon, \quad f_i(x(\varepsilon, \eta)) \leq \eta, \quad i = 1, \ldots, m,$$

where c^* is the optimal value of *(DCP)*. Such a vector will be called an (ε, η)-solution of *(DCP)*.

Our method for finding an (ε, η)-solution to *(DCP)* will proceed according to a dichotomy scheme borrowed from Nghia–Hieu [24]. Namely to determine the optimal value c^* we will start with an interval (γ_0, β_0) that is certain to contain c^*, then reduce it by a half from iteration to iteration until we reach an interval enclosing c^* whose length does not exceed ε. To see how the interval (γ_k, β_k) at iteration k should be reduced, a subproblem has to be solved to verify whether there is a feasible solution achieving a function value less than $\alpha_k = 1/2(\beta_k + \gamma_k)$. If such a solution exists, we set $\beta_{k+1} = \alpha_k, \gamma_{k+1} = \gamma_k$, otherwise we set $\beta_{k+1} = \beta_k, \gamma_{k+1} = \alpha_k$.

This scheme requires solving at each iteration a system of d.c. inequalities which are, in general, nondifferentiable. Recently, global methods have been developed

for solving a system of Lipschitzian equations-inequations. A branch and bound algorithm was proposed in Horst and Thoai [15] while an outer approximation algorithm was developed in Thach [31]. The method we will present in the sequel can be considered as an adaptation of Thach's procedure.

In view of the complexity inherent to global optimization, up to now only problems of reasonable size can be expected to be successfully solved. Fortunately, the fuel mixture problem belongs to this class, because the number of all possible components to be considered does not exceed 20. Note that the existing linearization methods can not generally find a global solution to this highly nonconvex problem. A first attempt to solve this problem globally was done in Pham Dinh–El Bernoussi [30], where the original problem *(DCP)* was converted into the canonical form. It turns out that applying our method to the fuel mixture problem enables us to obtain quite satisfactory numerical results.

The paper is orgainized as follows. First a general scheme for minimizing a linear function over a compact set is outlined in the next section. The Section 3 presents an outer approximation procedure for solving a system of d.c. inequalities. Section 4 contains the detailed description of a finite algorithm for finding an (ε, η)-solution of *(DCP)*. Numerical results for the fuel mixture problem are presented in Section 5. Finally, a small example is given in the Appendix to illustrate the algorithm in the general case.

2. Outline of General Dichotomy Scheme

Let us begin with a conceptual scheme for solving the general problem

$$(P) \qquad \min\langle c, x\rangle \quad \text{subject to} \quad x \in S$$

where S is a compact convex set given by a system of nonlinear constraints $S = \{x \in \mathbb{R}^n : f_i(x) \leq 0, i = 1, \ldots, m\}$. The idea of this scheme is taken from Nghia–Hieu [24], where it has been originally proposed for solving a canonical d.c. program.

Let γ_0, β_0 be a lower bound and an upper bound for the optimal value c^* of (P). If $x^0 \in S$ is available then set $\beta_0 = \langle c, x^0\rangle$. Otherwise set $x^0 = \emptyset$.

– *Iteration* $k = 0, 1, \ldots$

At the beginning of iteration k we already have lower and upper bounds γ_k, β_k for c^* and $\beta_k = \langle c, x^k\rangle$ if $x^k \neq \emptyset$.

If $\beta_k - \gamma_k \leq \varepsilon$ then stop.

Otherwise, solve the following subproblem, noted (P_k),

check whether the set $D_k = \{x \in S : \langle c, x\rangle \leq \alpha_k\}$ *is empty or not and in the latter case find a point* $x^k \in D_k$.

where $\alpha_k = \frac{1}{2}(\beta_k + \gamma_k)$. Two cases can occur:

(i) if D_k is empty then set $\beta_{k+1} = \beta_k, \gamma_{k+1} = \alpha_k, x^{k+1} = x^k$ and go to iteration $k + 1$.

(ii) if there exists a point $\bar{x}^k \in D_k$ then set $\beta_{k+1} = \langle c, \bar{x}^k \rangle$, $\gamma_{k+1} = \gamma_k$, $x^{k+1} = \bar{x}^k$ and go to iteration $k+1$.

The following theorem can be proven exactly as Theorem 2 in [24].

THEOREM 1. *The above scheme terminates after at most*

$$k_\varepsilon = \max\{0, [\log_2(M/\varepsilon)] + 1\}$$

iterations, were $M = \beta_0 - \gamma_0$ *and* $[\lambda]$ *denotes the integral part of* λ. *If* $x^k \neq \emptyset$ *then* x^k *is an* ε*-optimal solution.*

As applied to *(DCP)*, the above scheme requires solving at each iteration a subproblem (P_k) of the form

$$x \in D, \quad f_i(x) \leq 0, \quad i = 1, \ldots, m, \tag{1}$$

$$\langle c, x \rangle \leq \alpha_k, \tag{2}$$

where all f_i are d.c. This is itself a very hard problem which cannot be handled by Newton-type methods using derivatives, gradients, subgradients. Observe, however, that (P_k) differs from (P_{k+1}) only by the linear constraint (2). So it suggests choosing for solving the subproblem an algorithm which could employ the information obtained in solving (P_k) for the solution of (P_{k+1}). The outer approximation approach (e.g. [31]) meets this requirement. In the next section, we shall describe an outer approximation method following the scheme of Thach [31].

3. An Outer Approximation Procedure for Solving a System of d.c. Inequalities

Let us consider the problem of finding a point x satisfying

$$f_i(x) \leq 0 \quad i = 1, \ldots, m, \tag{3}$$

$$h(x) \leq 0 \tag{4}$$

where f_i are d.c. and $D = \{x : h(x) \leq 0\}$ is a nonempty, compact convex set.

Obviously, the system (3) is equivalent to the single inequality $f(x) \leq 0$ where $f(x) = \max\{f_i(x) : i = 1, \ldots, m\}$. Let $f_i = p_i - q_i$, where $p_i, q_i, (i = 1, \ldots, m)$ are finite convex functions over $\mathbb{R}^n$. Following [36] we can write

$$f(x) = p(x) - q(x) \tag{5}$$

where

$$p(x) = \max_{i=1,\ldots,m} \{p_i(x) + \sum_{\substack{j=1 \\ j \neq i}}^{m} q_j(x)\}, \quad q(x) = \sum_{j=1}^{m} q_j(x). \tag{6}$$

Thus, system (3)–(4) is equivalent to the system

$$x \in D, \quad p(x) - q(x) \leq 0. \tag{7}$$

Solving (3)–(4) is now reduced to finding $(x,t) \in \Omega_1 \backslash \Omega_2$ where Ω_1 and Ω_2 are convex sets in $\mathbb{R}^n \times \mathbb{R}$ defined by

$$\Omega_1 = \{(x,t) \in \mathbb{R}^n \times \mathbb{R} : x \in D, p(x) - t \leq 0\},$$

$$\Omega_2 = \{(x,t) \in \mathbb{R}^n \times \mathbb{R} : q(x) - t < 0\}.$$

To solve the latter problem according to the outer approximation scheme in Thach [31], we will construct a nested sequence of polyhedrons

$$\Omega_1 \subset \cdots \subset S_k \subset \cdots S_1 \subset S_0$$

in the following manner. At iteration k we choose a point $(x^k, t_k) \in S_k \backslash \Omega_2$; if $(x^k, t_k) \in \Omega_1$ or if $S_k \backslash \Omega_2 = \emptyset$ we are done, otherwise we construct a hyperplane strictly separating (x^k, t_k) from Ω_2, i.e. a linear inequality $l_k(x,t) \leq 0$ satisfied by all $(x,t) \in \Omega_2$ but not by (x^k, t_k). The next polyhedron is then defined to be

$$S_{k+1} = S_k \cap \{(x,t) : l_k(x,t) \leq 0\}.$$

Two main points in this scheme are how to choose (x^k, t_k) and how to construct the inequalities $l_k(x^k, t_k) \leq 0$ in order to ensure convergence. First note that since D is bounded the polyhedron Ω_1 has just one recession direction, namely the halfline $t \geq 0$. Therefore we can take S_0 to be a polyhedron with the halfline $t \geq 0$ as unique recession direction. Then, since $\Omega_1 \subset S_k \subset S_0$, the polyhedron S_k, too, has the halfline $t \geq 0$ as unique recession direction. Let V_k be the vertex set of S_k and $W_k = \{(x,t) \in V_k : t - q(x) \leq 0\}$.

LEMMA 1. *If $W_k = \emptyset$ then $\Omega_1 \backslash \Omega_2 = \emptyset$ or, equivalently, (3)–(4) has no solution.*

Proof. If $W_k = V_k \backslash \Omega_2 = \emptyset$ then $V_k \subset \Omega_2$. Since Ω_2 is convex open and has the same direction of recession as S_k, we have $S_k \subset \Omega_2$. Hence $\Omega_1 \backslash \Omega_2$, i.e. $\Omega_1 \backslash \Omega_2 = \emptyset$ as was to be proved.

If $W_k \neq \emptyset$ we will choose $(x^k, t_k) \in W_k$. In general W_k may contain more than one element, and (x^k, t_k) can be chosen in either of the following alternative ways:

1. x^k is the *closest* point to the solution set (cf. [31]). More specifically, $x^k \in \arg\min\{F(x) : (x,t) \in W_k\}$ where $F(x) = \max\{h(x), f(x)\}$.
2. (x^k, t_k) solves the problem
$$\min\{t - q(x) : (x,t) \in V_k\}. \tag{8}$$
Obviously $W_k = \emptyset$ is equivalent to $\min\{t - q(x) : (x,t) \in V_k\} > 0$. Thus by solving (8) we can also verify whether $W_k = \emptyset$.

The above process is in general infinite. Since we are in fact interested in obtaining an η-solution, i.e. a point x^* such that

$$x^* \in D \quad \text{and} \quad f_i(x^*) \leq \eta \quad i = 1, \ldots, m, \tag{9}$$

we are led to the following algorithm.

OA PROCEDURE

- *Initialization*: Let $D \subset \Delta$ be a simple polytope with known vertex set (for example, Δ is a simplex containing D). Take $s \in \Delta$ and construct the polyhedral convex set
$$S_0 = \{(x,t) : x \in \Delta, \langle y^0, x - s\rangle + p(s) - t \leq 0\} \tag{10}$$
where $y^0 \in \partial p(s)$. It is clear that $\Omega_1 \subset S_0$. Let V_0 be the vertex set of S_0. Set $k = 0$.
- *Step* $k = 0, 1, \ldots$
(i) Construct $W_k = \{(x,t) \in V_k : t - q(x) \leq 0\}$. If $W_k = \emptyset$ then stop: system (3)–(4) has no solution.
(ii) Find $(x^k, t_k) \in \arg\min\{F(x) : (x,t) \in W_k\}$ where
$$F(x) = \max\{h(x), f(x)\}. \tag{11}$$
(iii) If $x^k \in D$ and $f(x^k) \leq \eta$ then stop: x^k is an η-solution. If $x^k \in D$ solve the convex programming problem
$$(Q_k)\ \min\{\| x^k - z \|^2 : z \in D\}$$
obtaining z^k. If $f(z^k) \leq \eta$ then stop: z^k is an η-solution.
(iv) Let $\mu = \max\{h(x^k), p(x^k) - t_k\}$ then we have $\mu > 0$. In fact, if $h(x^k) \leq 0$, i.e. $x^k \in D$ then
$$0 < \eta < f(x^k) = p(x^k) - q(x^k) \leq p(x^k) - t_k$$
since $t_k - q(x^k) \leq 0$. If $\mu = h(x^k)$ then set
$$l_k(x,t) = \langle y^k, x - x^k\rangle + h(x^k) \tag{12}$$
with $y^k \in \partial h(x^k)$. If $\mu = p(x^k) - t_k$ then set
$$l_k(x,t) = \langle y^k, x - x^k\rangle + p(x^k) - t \tag{13}$$
with $y^k \in \partial p(x^k)$.
(v) Form the new polyhedral convex set
$$S_{k+1} = S_k \cap \{(x,t) : l_k(x,t) \leq 0\}.$$
Compute the vertex set V_{k+1} of S_{k+1} and go to step $k+1$.

COMMENTS

1. in step (ii), we could alternatively choose (x^k, t_k) according to (8). Then the OA procedure is very similar to the one for solving the concave minimization problem
$$\min\{t - q(x) : x \in D, p(x) - t \leq 0\}.$$
2. in step (iii) (Q_k) is used to find an approximate solution such that $x^* \in D, f_i(x^*) \leq \eta$. It should be noted that using only linear cuts as introduced in step (iv), we could only obtain an approximate solution x^* such that
$$h(x^*) \leq \eta, \quad f_i(x^*) \leq \eta, \quad i = 1, \ldots, m.$$

If a point $\omega \in intD$ is available then instead of solving (Q_k) a point z^k can be chosen as the intersection point of the segment $[\omega, x^k]$ with the boundary of D.

THEOREM 2. *The OA Procedure terminates after a finite number of steps either yielding an η-approximate solution (case (ii)–(iii)) of the system or showing that it has no solution (case (i)).*

Proof. The procedure can be infinite only if step (iv) occurs for all sufficiently large k. But then the algorithm reduces to a standard outer approximation for the convex set

$$\max\{h(x), p(x) - t\} \leq 0.$$

By the convergence theorem in [18], every cluster point $(\bar{x}, \bar{t})$ of the bounded sequence $\{(x^k, t_k)\}$ satisfies

$$h(\bar{x}) \leq 0, \quad p(\bar{x}) - \bar{t} \leq 0,$$

i.e. $\bar{x} \in D, p(\bar{x}) - \bar{t} \leq 0$. Therefore, for k large enough, $p(x^k) - t_k \leq \eta/2$, hence

$$f(x^k) = p(x^k) - q(x^k) = (p(x^k) - t_k) + (t_k - q(x^k)) \leq \frac{\eta}{2}. \tag{14}$$

Since the algorithm neither stops at step (i) nor step (iii) we must have $f(z^k) > \eta$. But $\bar{x} \in D$ implies that $\| x^k - z^k \| \to 0$, hence, for k large enough, $| f(x^k) - f(z^k) | \leq \eta/2$. This, together with (14), implies $f(z^k) \leq \eta$, a contradiction.

4. Finite Algorithm for Finding an (ε, η)-Solution of *(DCP)*

By incorporating the above OA Procedure into the dichotomy scheme described in Section 2, we obtain the following algorithm for solving *(DCP)*. Let f, p, q be defined by (5), (6) respectively.

ALG 1

- *Initialization*: Let $D \subset \Delta$ be a simple polytype with known vertex set and $s \in \Delta$. Select $\varepsilon > 0$ and $\eta > 0$. Compute
  $$\beta_0 = \max\{\langle c, x\rangle : x \in \Delta\}, \qquad \gamma_0 = \min\{\langle c, x\rangle : x \in \Delta\}.$$
  Construct the polyhedral convex set
  $$S_0 = \{(x, t) : x \in \Delta, \langle y_0, x - s\rangle + p(s) - t \leq 0, \}$$
  where $y_0 \in \partial p(s)$. Let V_0 be the vertex set of S_0.
  If $s \in D : f(s) \leq \eta$ set $\beta_0 = \langle c, s\rangle$ and $x^{opt} = s$. Otherwise $x^{opt} = \emptyset$. Set $k = 0$.
- *Iteration* $k = 0, 1, \ldots$
  k.1 If $\beta_k - \gamma_k \leq \varepsilon$ then stop.
  k.2 Otherwise compute the vertex set V_k^{α} of the set
  $$\{(x, t) \in S_k : \langle c, x\rangle \leq \alpha_k\}$$

where $\alpha_k = \frac{1}{2}(\beta_k + \gamma_k)$. Find
$W_k = \{(x,t) \in V_k^\alpha : t - q(x) \leq 0\}$.
k.3 If $W_k = \emptyset$ then set
$S_{k+1} = S_k, \quad \beta_{k+1} = \beta_k, \quad \gamma_{k+1} = \alpha_k$
and go to iteration $k+1$.
k.4 Otherwise find $(x^k, t_k) \in \arg\min\{F(x) : (x,t) \in W_k\}$ where
$F(x) = \max\{h(x), f(x)\}$.
k.5 If $x^k \in D$ and $f(x^k) \leq \eta$ then set
$S_{k+1} = S_k, \quad \beta_{k+1} = \langle c, x^k \rangle, \quad \gamma_{k+1} = \alpha_k, \quad x^{opt} = x^k$
and go to iteration $k+1$.
k.6 If $x^k \in D$ solve the convex program
$(Q_k) \qquad \min\{\| z - x^k \|^2 : z \in D\}$
obtaining an optimal solution z^k. Two subcases may occur:
(k.6.1) $f(z^k) \leq \eta$: Set $\beta_{k+1} = \min\{\beta_k, \langle c, z^k \rangle\}$ and let x^{opt} be the corresponding solution. Set $\gamma_{k+1} = \gamma_k$.
(k.6.2) $f(z^k) > \eta$: Set $\beta_{k+1} = \beta_k, \gamma_{k+1} = \gamma_k$.
k.7 Let $\mu = \max\{h(x^k), p(x^k) - t_k\}$.
If $\mu = h(x^k)$ then compute $y^k \in \partial h(x^k)$ and set
$l_k(x,t) = \langle y^k, x - x^k \rangle + h(x^k)$.
If $\mu = p(x^k) - t_k$ then compute $y^k \in \partial p(x^k)$ and set
$l_k(x,t) = \langle y^k, x - x^k \rangle + p(x^k) - t$.
k.8 Form the new polyhedral convex set
$S_{k+1} = S_k \cap \{(x,t) : l_k(x,t) \leq 0\}$.
Compute the vertex set V_{k+1} of S_{k+1} and go to iteration $k+1$.

THEOREM 3. *Algorithm ALG 1 terminates after a finite number of iterations yielding an (ε, η)-solution of problem (DCP) (provided $x^{opt} \neq \emptyset$).*

This follows from the finiteness of the dichotomy scheme and of the OA Procedure.

COMMENTS

(i) The computationally most expensive part of the algorithm is the calculation of vertex sets V_k^α (step k.2) and V_{k+1} (step k.8). The problem of finding all vertices of a polytope that is defined by a finite system of linear inequalities has been discussed extensively in the literature (e.g., [22]). Some algorithms for finding V_{k+1} via the knowledge of V_k have been proposed by Thieu–Tam–Ban [32], Horst–Thoai–Vries [17], Pham Dinh–El Bernoussi [30], Chen–Hansen–Jaumard [4]. It is worthwhile noting that in our case S_k is a polyhedral convex set with a special structure, namely with only one extreme direction along the axis t.

(ii) When D is a polytope defined by the system of linear inequalities
$$\langle A_i, x \rangle \leq b_i, \quad i = 1, \ldots, r, \tag{15}$$
$$x_j \geq 0, \quad j = 1, \ldots, n, \tag{16}$$

where $A_i \in \mathbb{R}^n, b_i \in \mathbb{R}$, the algorithm can be simplified. Specifically, for initialization the set Δ can be taken to be a simplex

$$\Delta = \{x : \sum_{j=1}^{n} x_j \leq N, \quad x_j \geq 0, \quad j = 1, \ldots, n\}$$

where $N = \max\{\sum_j x_j : x \in D\}$. There is no Step 6, while in Step 7 the cut is constructed as follows. Let

$$h(x) = \max\{\langle A_i, x\rangle - b_i, \quad i = 1, \ldots, r\}.$$

Then

– if $x^k \in D$ and $f(x^k) > \eta$ then compute $y^k \in \partial p(x^k)$ and set

$$l_k(x,t) = \langle y^k, x - x^k\rangle + p(x^k) - t,$$

– if $x^k \in D$ then select the index $i_k = \arg\max\{\langle A_i, x^k\rangle - b_i, i = 1, \ldots, r\}$ and set

$$l_k(x,t) = \langle A_{i_k}, x\rangle - b_{i_k}.$$

We will refer to ALG 1 with the above modifications as ALG 2.

5. Application to the Fuel Mixture Nonconvex Optimization Problem

We now discuss the application of d.c. optimization to the fuel mixture problem encountered in the oil industry. This problem was earlier studied in Pham Dinh–El Bernoussi [30] where some preliminary numerical results were reported. The method proposed in this paper is to convert the problem into the canonical form, then solve the canonical d.c. program by outer approximation and cutting plane technique. It is worthwhile noting that originally the fuel mixture problem is formulated as *(DCP)* so it is more convenient to solve it directly by the methods in the preceding sections.

5.1. Fuel mixture problem as a d.c. programming problem

A fuel is the mixture of several components obtained after refining, (e.g. Isomat, Alkulat, Reformat, etc.). Each component is characterized by a number of parameters, (e.g. volume mass, steam pressure, etc.). A commercialized fuel (e.g. Summer Super or Winter Super,...) must satisfy certain specified conditions which set the bounds on its characteristics. The problem is to fabricate a cheapest fuel mixture satisfying the required conditions.

Many researches have been done in order to establish the law of mixture, i.e. the determination of a characteristic of a mixture as a function of those of its components. In the additive model widely used in practice, every characteristic of the fuel depends linearly upon the characteristics of the components. Namely, if $x_j (j = 1, \ldots, n)$ is the fraction of component j and a_j its characteristic then the characteristic of the mixture is determined by the formula

$$y = a_1 x_1 + a_2 x_2 + \cdots + a_n x_n \tag{17}$$

The fuel mixture problem is formulated as a linear programming problem. Unfortunately, the solutions obtained in this model are quite often unsatisfactory.

More recent researches have led to another model with takes into account the (first order) interactions between each pair of components. Numerous experiments have been carried out in order to determine the quadratic interaction coefficient α_{ij} between components i and j. The characteristic of the mixture is then computed as follows

$$y = \sum_{j=1}^{n} a_j x_j + \sum_{i,j=1, i<j}^{n} \alpha_{ij} x_i x_j. \tag{18}$$

This model is more adequate than the additive model. The price to be paid, however, is that the fuel mixture problem becomes a nonconvex optimization problem with quadratic constraints.

Note that, by definition, $x_j \geq 0$ and $\sum_j x_j = 1$. Other constraints may be either linear or quadratic.

- *Linear* constraints are due to the bounds which are imposed on the volume of certain components and to the fact that for certain characteristics, the additive model (17) can still be used. Summarizing all these linear constrains, we can write

  $$Ax \leq b$$

  where A is a $(r \times n)$-matrix and $b \in \mathbb{R}^r$.
- *Quadratic* constraints occur to take account of the interactions between components. These constraints have the form

  $$\underline{a} \leq \sum_{j=1}^{n} a_j x_j + \sum_{i,j=1, i<j}^{n} \alpha_{ij} x_i x_j \leq \bar{a}.$$

  and so can be written as

  $$1/2\langle A_i x, x\rangle + \langle b_i, x\rangle + d_i \leq 0, \quad i = 1, \ldots, s$$

  where $A_i, i = 1, \ldots, s$ are $(n \times n)$-symmetric matrices, $b_i \in \mathbb{R}^n$ and $d_i \in \mathbb{R}$.

Thus, mathematically, the problem of the cheapest fuel mixture can be formulated as

$$(FM) \quad \begin{cases} \min\langle c, x\rangle, \\ \sum_{j=1}^{n} x_j = 1, \quad x_j \geq 0, \\ f_i(x) = \frac{1}{2}\langle A_i x, x\rangle + \langle b_i, x\rangle + d_i \leq 0 \quad (i = 1, \ldots, s) \end{cases}$$

($c = (c_1, \ldots, c_n)$ is the cost vector).

Since quadratic functions are d.c. functions, problem *(FM)* is a *(DCP)*. However to be able to apply the above developed methods to solve this problem, we need an explicit representation of the quadratic functions $f_i(x)(i = 1, \ldots, s)$ as d.c. functions. The following proposition, firstly established in [30], may be useful in other contexts.

PROPOSITION 1. *Any system of quadratic constraints*

$$f_i(x) = \frac{1}{2}\langle A_i x, x\rangle + \langle b_i, x\rangle + d_i \leq 0 \quad (i = 1, \ldots, s) \tag{19}$$

where A_i, $i = 1, \ldots, s$ *are symmetric matrices,* $b_i \in \mathbb{R}^n$, $d_i \in \mathbb{R}$, *is equivalent to a single d.c. constraint*

$$p(x) - \frac{1}{2}\lambda\langle x, x\rangle \leq 0$$

where $p(x)$ *is a convex function and* λ *is a constant such that* $\lambda \geq \max\{\rho(A_i) : i = 1, \ldots, s\}$, $\rho(A_i)$ *being the spectral radius of* A_i.

Proof. We can write $f_i(x)$ as

$$f_i(x) = \frac{1}{2}\langle (A_i + \lambda I)x, x\rangle + \langle b_i, x\rangle + d_i - \frac{\lambda}{2}\langle x, x\rangle = p_i(x) - \frac{\lambda}{2}\langle x, x\rangle \tag{20}$$

where

$$\lambda \geq \max\{\rho(A_i) : i = 1, \ldots, s\},$$

and $\rho(A_i)$ is the spectral radius of A_i. It is obvious that $A_i + \lambda I, i = 1, \ldots, s$, are positive semi-definite so all $p_i(x)$ are convex. Then

$$f(x) = \max_{i=1,\ldots,s} f_i(x) = \max_{i=1,\ldots,s} p_i(x) - \frac{\lambda}{2}\langle x, x\rangle$$

and (19) is reduced to the constraint

$$p(x) - \frac{\lambda}{2}\langle x, x\rangle \leq 0$$

where $p(x) = \max\{p_i(x), i = 1, \ldots, s\}$ and $\lambda/2\langle x, x\rangle$ are convex functions.

Note that if $\rho(A_i), i = 1, \ldots, s$, are not available, we can take

$$\lambda \geq \max\{\| A_i \|_1 : i = 1, \ldots, s\}$$

where for any matrix $M = (M_{ij})$: $\| M \|_1 = \max\{\sum_{i=1}^n | M_{ij} | : j = 1, \ldots, n\}$.

Thus, finally, problem *(FM)* can be rewritten as

$$(FM') \quad \begin{cases} \min\langle c, x\rangle, \\ \sum_{j=1}^n x_j = 1, \quad x_j \geq 0, \\ Ax \leq b, \\ p(x) - \frac{\lambda}{2}\langle x, x\rangle \leq 0. \end{cases}$$

This is a problem *(DCP)* where D is a polytope. Therefore ALG 2 can be applied to solve it.

5.2. IMPLEMENTATION ISSUES

The equality constraint $\sum_{j=1}^{n} x_j = 1$ could be used to eliminate one variable and thus, to transform the problem into one of $n-1$ variables as was done in [30]. However, it may be more convenient to proceed directly as follows.

Let

$$\Delta = \{x : \sum_{j=1}^{n} x_j = 1, x_j \geq 0, \quad j = 1, \ldots, n\}$$

(an $(n-1)$-simplex containing D). As the initial polyhedron, take

$$S_0 = \{(x,t)\} : x \in \Delta, \langle y^0, x - v\rangle + p(v) - t \leq 0\}$$

where $v = (1/n, \ldots, 1/n)$ and $y^0 \in \partial p(v)$. Then every subsequent polyhedron S_k will be defined by a system of linear inequalities, plus the linear equation $\sum_{j=1}^{n} x_j = 1$.

For the impliementation of ALG 2, some modifications should be take in the calculation of vertex sets in order to deal with this kind of polyhedral sets. We discuss below a modification of Horst–Thoai–Vrie's method [17] which seems to be well suited to our application. The idea of the method is borrowed from the algorithm for solving general linear programs by Pham Dinh Tao [28].

Consider a polytope defined by the following system of linear equations and inequalities

$$K = \{x \in \mathbb{R}^n : Ax = a, Bx \leq b\}$$

where A is an $(m \times n)$ matrix of rang m and B is a $(p \times n)$ matrix. Suppose that the vertex set $V(K)$ of K is known, we want to compute the vertex set $V(K')$ of K' defined by:

$$K' = \{x \in K : \langle \alpha, x\rangle + \beta \leq 0\}.$$

Let

$$H = \{x : \langle \alpha, x\rangle + \beta = 0\}, \tag{21}$$

$$V^+(K) = \{v \in V(K) : \langle \alpha, x\rangle + \beta > 0\}, \tag{22}$$

$$V^-(K) = \{v \in V(K) : \langle \alpha, x\rangle + \beta < 0\}. \tag{23}$$

It is obvious that $V(K') = V^-(K) \cup V(S)$ where $S = K \cap H$. All available methods for determining $V(S)$ via the knowledge of $V(K)$ are based on the following characterization (cf. Falk–Hoffman [6]):

LEMMA 2. *$w \in V(S)$ if and only if w is either a vertex of K lying in H or a point where an edge $[u, v]$ of $K, u \in V^-(K), v \in V^+(K)$ intersects H.*

For very small problems with $| V(K) | \leq 100$, the straightforward method of Thieu–Tam–Ban [32] can be used. But for $| V(K) | > 100$ the following method of Horst–Thoai–Vries [17] (see also [30]) is superior: Let $| V | = \min\{| V^+(K) |, | V^-(K) |\}$. For each $u \in V$ denote by $E(u)$ the set of halflines emanating from u, each contains an edge of K. It suggests to compute for each $e \in E(u)$ the intersection w of e with the hyperplane H and to check whether $w \in K$ or $w \notin K$. In [17] this was done by performing pivot operations in a simplex tableau involving slack variables. Thus we have to handle $(n+1) \times (2n+2)$-matrices.

Following [28], rather than represent a vertex u by a simplex tableau, we will represent it as a solution of the linear system

$$Ax = a, \tag{24}$$

$$B_I x = b_I, \tag{25}$$

such that

$$B_{I*} x \leq b_{I*}, \tag{26}$$

where $I \subset \{1, \ldots, p\}, | I | = n - m$ and the rows of A and B_I are linearly independent. I^* is the complement of I in $\{1, \ldots, m\}$. Assuming nondegeneracy, it is well-known then there are exactly $n - m$ halflines emanating from u whose direction is the solution $d^{(i)}$ of the following linear system

$$Ax = 0, \tag{27}$$

$$B_{I(i)} x = 0, \tag{28}$$

$$B_{r(i)} x = -1 \tag{29}$$

where $i \in \{1, \ldots, | I |\}, r(i) \in I$ and $I(i) = I \backslash \{r(i)\}$. If $\langle \alpha, d^{(i)} \rangle < 0$ then the halfline corresponding to $d^{(i)}$ will intersect H at the point $w = u + \lambda_i d^{(i)}$ where

$$\lambda_i = -\frac{\langle \alpha, u \rangle + \beta}{\langle \alpha, d^{(i)} \rangle}.$$

Analogously, if $u \in V^-(K)$ then the halfline will intersect H if $\langle \alpha, d^{(i)} \rangle > 0$. Thus, we can compute all newly generated vertices by solving linear systems (27)–(29).

It should be noted that the above procedure is valid only if K is bounded. Although in our case $K = S_k$ is unbounded, it possesses only one infinite direction along the axis t. Furthermore, every cut has the form

$$\langle \alpha, x \rangle + \beta \leq 0$$

or

$$\langle \alpha, x \rangle + \beta - t \leq 0,$$

and $K' = S_{k+1}$ has the same direction as K. Also, $w \in V(S)$ if and only if w is either a vertex of K lying in H or the intersection point of an edge (possible infinite) of K, emanating from a vertex $u \in V^+(K)$, with the hyperplane H. Thus, to carry out the calculation of the vertex set $V(K')$ it suffices to take $V = V^+(K)$.

Another efficient algorithm for the on-line vertex enumeration problem has been proposed by Chen–Hansen–Jaumard [4] based on exploiting adjacency lists between vertices. Note that degeneracy does not occur frequently within cutting plane algorithms. However, degeneracy can be handled as in [4], i.e., by checking whether a vertex of K is on H and giving a small perturbation to β if it is the case. For the question of detecting redundant constraints see discussions in [17], [14].

5.3. Finding a better solution

In actual practice even the computational cost of obtaining an (ε, η)-solution might be so high that one should be content with one reasonably good solution. For instance, sometimes a feasible solution may be available at the beginning, and one may wish to compute a feasible solution with the price reduced by a certain percentage. Our approach will allow to obtain easily such a solution if it exists. Specifically, if x^* is the given feasible solution with the price $\langle c, x^* \rangle$ and we wish to reduce the price by $\theta\%$ (θ is some prescribed positive number), then the problem is to solve the system

$$(P^*) \quad \begin{cases} \langle c, x \rangle \leq \alpha^*, \\ \sum_{j=1}^{n} x_j = 1, \quad x_j \geq 0, \\ Ax \leq b, \\ f_i(x) - \frac{1}{2}\langle A_i x, + \langle b_i, x \rangle + d_i \leq 0 \quad (i = 1, \ldots, s), \end{cases}$$

where $\alpha^* = (1 - \theta/100)\langle c, x^* \rangle$. This problem can be solved by the above method.

5.4. Numerical results

Algorithms ALG 1, ALG 2 were programmed in PASCAL under UNIX system. An illustrative example for solving *(DCP)* will be given in the Appendix. We present here some numerical solutions to problem *(FM)*.

EXAMPLE. $n = 6, r = 8, m = 3$

1. cost vector c: 1.4 1.6 1.4 1.6 1.0 1.0
2. linear constraints: $Ax \leq b$

-1271	-263	-439	-37	-44	-50	-500
1271	263	439	37	44	50	860
-630	-824	-789	-866	-586	-804	-730
630	824	789	866	586	804	730
-100	-5	-35	30	-130	15	-10
100	5	35	-30	130	-15	10
-105	-16	-35	10	-100	-1	-40
105	16	35	-10	100	1	70

3. quadratic constraints: $\frac{1}{2}\langle A_i x, x\rangle + \langle b_i, x\rangle + d_i \leq 0$

0	1	1	13	0	2	-81.9	
1	0	-1	1	1	1	-90.4	
1	-1	0	6	13	1	-88.4	
13	1	6	0	33	7	-99.8	
0	1	13	33	0	1	-91.0	
2	1	1	7	1	0	-78.4	$d_1 = 85.0$

0	-1	-1	-13	0	-2	81.9	
-1	0	1	-1	-1	-1	90.4	
-1	1	0	-6	-13	-1	88.4	
-13	-1	-6	0	-33	-7	99.8	
0	-1	-13	-33	0	-1	91.0	
-2	-1	-1	-7	-1	0	78.4	$d_2 = 95.0$

0	-3	2	0	-9	4	84.6	
-3	0	0	4	-12	-2	-101.9	
2	0	0	3	-7	-1	-98.6	
0	4	3	0	17	3	-110.7	
-9	-12	-7	17	0	-8	-93.0	
4	-2	-1	3	-8	0	-88.2	$d_3 = 95.4$

With $\varepsilon = 0.0001, \eta = 0.001$, the algorithm terminates after 30 iterations and 6.1 seconds (on SUN SPARC station). The optimal solution is

$$x^{opt} = (0.151352, 0.000000, 0.682859, -0.000000, 0.072080, 0.093709)$$

TABLE I. Numerical results for the fuel mixture problem

n	ε	η	ITER	CUT	VER	TIME
7	0.001	0.001	17	8	156	1.0
7	-	-	14	4	44	0.6
7	-	-	22	12	248	3.2
8	-	-	26	16	1078	26.4
8	-	-	19	9	224	4.3
8	-	-	19	9	221	2.1
9	-	-	33	23	3671	284.8
9	-	-	16	6	156	2.2
9	-	-	21	11	476	8.5
11	0.01	0.01	21	14	3275	108.8
11	-	-	22	15	4323	168.3
12	0.05	0.01	23	18	8896	489.8
12	-	-	21	16	6089	286.5
13	-	-	20	15	4592	163.4
13	-	-	22	17	7104	504.7

with function value $c^* = 1.333684$. The maximal number of generated vertices is 225. Suppose given a fixed function value $\bar{c} = 1.35$. After 3 iterations the algorithm finds a solution

$$x = (0.151374, 0.000000, 0.682788, -0.000000, 0.072086, 0.093751)$$

with the objective function value 1.333665 which is less than $\bar{c}$ by 1.21%. But there is no solution achieving an objective function value less than $\bar{c}$ by 1.22%. The algorithm has been run for several problems of different size, generated from the real-word data. The computational results are given in Table I: "n" is the number of components, "ITER" the number of iterations, "CUT" the number of cutting plans, "VER" the maximal number of generated vertices to be stored and "TIME" the CPU-time in the seconds (on SPARC station).

It is worth noting that in practice the number of components of a fuel mixture is not very large. The computational results given above show that the algorithm is quite efficient for solving the fuel mixture problem.

Acknowledgement

The authors would like to thank the referees for their insightful comments and suggestions. They are grateful to Prof. Hoang Tuy for his valuable suggestions and corrections which have substantially improved the presentation of the revised paper.

Appendix

We illustrate ALG 1 by the following example:

$$\begin{aligned} &\min -2x_1 + x_2, \\ &\text{s.t.} \quad x_1^2 + x_2^2 \leq 1, \\ &\qquad x_1 - x_2^2 \leq 0. \end{aligned}$$

Setting $c = (-2, 1)$ and

$$\begin{aligned} h(x) &= x_1^2 + x_2^2, \\ p(x) &= x_1, \\ q(x) &= x_2^2, \end{aligned}$$

the problem becomes

$$\min\{\langle c, x\rangle \quad \text{s.t.} \quad h(x) \leq 0, \quad p(x) - q(x) \leq 0\},$$

where $D = \{x : h(x) \leq 0\}$ is included in the simplex

$$T = \{(x_1, x_2) : x_1 > -1, x_2 > -1, x_1 + x_2 \leq 4\}.$$

Initialization: We start with $S_0 = \{(x, t) : x \in T, x_1 - t \leq 0\}$ which has 3 vertices

$$v^1 = (-1, -1, -1), \quad v^2 = (3, -1, 3), \quad v^2 = (-1, 3, -1).$$

Solving the linear programs yields a lower bound $\gamma_0 = -7$ and an upper bound $\beta = 5$. We take $\varepsilon = 0.0001, \eta = 0.001$.

Iteration 0: We obtain $\alpha_0 = -1$. We compute the set

$$W_0 = \{(-1, -1, -1), (0, -1, 0), (1, 1, 1)\}$$

and by solving $\min\{F(x) : x \in W_0\}$ we obtain $(x^0, t_0) = (0, -1, 0)$. Since $h(x^0) = p(x^0) - t_0 = 0$ we have $\beta_1 = -1, \gamma_1 = -7, x^{opt} = x_0$ and $S_1 = S_0$.

Iteration 1: We compute $\alpha_1 = -4$. The set $W_1 = \emptyset$ so we set $\beta_2 = -1, \gamma_2 = -4, S_2 = S_1$.

Iteration 2: We obtain $\alpha_2 = -2.5$. We compute the set

$$W_2 = \{(-1, -1, -1), (0.75, -1, 0.75), (1.5, 0.5, 1.5)\}$$

and by solving $\min\{F(x) : x \in W_2\}$ we obtain $(x^2, t_2) = (0.75, -1.075)$. Since $x^2 \notin D$ we compute the projection of x^2 on D and obtain $z^2 = (0.6, -0.8)$ with $p(z^2) - q(z^2) = -0.04 < \eta$. Therefore $\beta_3 = -1, \gamma_3 = -2.5$. We construct a new polytope $S_3 = S_2 \cap \{(x, t) : 1.2x_1 - 1.6x_2 + 2 \leq 0\}$ which has 4 vertices:

$$\begin{aligned} v^1 &= (-1, -1, -1), \\ v^2 &= (-1, 3, -1), \\ v^3 &= (0.333333, -1, 0.333333) \\ v^4 &= (1.857143, 0.142857, 0.142857). \end{aligned}$$

With $\varepsilon = 0.0001, \eta = 0.001$, the algorithm terminates after 26 iterations at a solution $x^* = (0.618002, -0786182)$ with the objective function value -2.022186.

References

1. R. Benacer (1986), *Contribution à l'étude des algorithmes de l'optimisation non convexe et non differentiable.* Thèse de doctorat en math. appl., Université J. Fourier, Grenoble, France.
2. M.C. Bohringer and S.E. Jacobsen (1983), Two general algorithms for solving linear programs with an additional reverse convex constraint. In *Lectures Notes in Control and Inform. Sc,* number 59 in Sys. Mod. and Optim., Copenhagen. 11th of IFIP Working Conference.
3. J. Chaarani (1989), *Etude d'une classe d'algorithmes d'optimisation non convexe. Implementation et Applications.* Thèse de doctorat, Univ. Joseph Fourier, Grenoble, France.
4. P.C. Chen, P. Hansen, and B. Jaumard (1991), On-line and off-line vertex enumeration by adjacency lists. *Operations Research Letters* **10**, 403–409.
5. A. Chine (1991), *Algorithmes robustes en optimisation non convexe. Codes et Simulations numeriques en grande dimension.* Thèse de doctorat, Univ. Joseph Fourier, Grenoble, France.
6. J.E. Falk and K.L. Hoffman (1976), A successive underestimating method for concave programming problems. *Mathematics of Operations Research* **1**, 251–259.
7. C.A. Floudas and V. Visweswaran (1990), A global optimization algorithm for certain classes of nonconvex NLPs -I. Theory. *Computers and Chemical Engineering* **14**, 1397, 1990.
8. J. Fülöp (1990), A finite cutting plane method for solving linear programs with an additional reverse convex constraint. *European J. Oper. Research* **44**, 395–409.
9. F. Giannessi, L. Jurina, and G. Maier (1979), Optimal excavation profile for a pipeline freely resting on the sea-floor. *Engineering Structures* **1**, 81–91.
10. R.J. Hillestad (1975), Optimization problems subject to a budget constraint with economies of scale. *Operations Research* **23**, 1091–1098.
11. R.J. Hillestad and Jacobsen S.E. (1980), Linear programs with an additional reverse convex constraint. *Applied Mathematics and Optimization* **6**, 257–269.
12. R.J. Hillestad and Jacobsen S.E. (1980), Reverse convex programming. *Applied Mathematics and Optimization* **6**, 63–78.
13. J.B. Hiriart-Urruty (1985), Generalized differentiability, duality and optimization for problems dealing with difference of two convex functions. In *Lectures Notes in Economics and Mathematical Systems* **256**, 37–69. Springer-Verlag, Berlin.
14. R. Horst (1988), Deterministic global optimization: Recent advances and new fields of application. *Naval Res. Log. Quar.* **37**, 433–471.
15. R. Horst and Thoai N.V. (1988), Branch and bound methods for solving systems of equations and inequalities. *J. of Optimization Theory and Applications* **134**, 426–430.
16. R. Horst, Thoai N.V., and H. Benson (1991), Concave minimization via conical partitions and polyhedral outer approximation. *Mathematical Programming* **50**, 259–274.
17. R. Horst, Thoai, N.V., and J. de Vries (1988), On finding new vertices and redudant constraints in cutting plane algorithms for global optimization. *Operations Research Letters* **7**(2), 85–90.
18. R. Horst, Thoai N.V., and H. Tuy (1987), Outer approximation by polyhedral convex sets. *Operations Research Spektrum* **9**, 153–159.
19. R. Horst, T.Q. Phong, and N.V. Thoai (1990), On solving general reverse convex programming problems by a sequence of linear programs and linear searches. *Annals of Operations Research* **25**, 1–18.
20. R. Horst, T.Q. Phong, N.V. Thoai, and J. de Vries (1991), On solving a d.c. programming problem by a sequence of linear programs. *J. of Global Optimization* **1**, 183–203.
21. R. Horst and H. Tuy (1993), *Global Optimization: Deterministic Approaches.* Springer-Verlag, Berlin New York, 2 edition.
22. T.H. Matheis and D.S Rubin (1980), A survey and comparison of methods for finding all vertices of convex polyhedral sets. *Mathematics of Operations Research* **5**, 167–185.
23. L.D. Muu (1985), Convergent algorithm for solving linear programs with an additional reverse convex constraint. *Kybernetica* **21**, 428–435.
24. N.D. Nghia and N.D. Hieu (1986), A method for solving reverse convex programming problems. *Acta Mathematica Vietnamica* **11**(2), 241–252.
25. V.H. Nguyen and J.J. Strodiot (1992), Computing a global optimal solution to a design centering problem. *Mathematical Programming* **53**, 111–123.

26. V.H. Nguyen, J.J. Strodiot, and N.V. Thoai (1985), On an optimum shape design problem. Tech. Report 85/5, Department of Mathematics, Univ. of Namur.
27. S. Sen and H.D. Sherali (1987), Nondifferentiable reverse convex programs and facial convexity cuts via disjunctive characterization. *Mathematical Programming* **37**, 169–183.
28. Pham D. Tao (1991) Un algorithme pour la résolution du programme linéaire général. *RAIRO-Recherche opérationnelle* **25**, 183–201.
29. Pham D. Tao and El Bernoussi (1988), Duality in d.c. (difference of convex functions) optimization. Subgradient methods. In K.H. Hoffmann *et al.*, eds, *Trends in Mathematical Optimization*, volume 84 of *International Series of Numerische Mathematik*. Birkhauser.
30. Pham D. Tao and El Bernoussi (1989), Numerical method for solving a class of global nonconvex optimization problems. *International Series of Numerische Mathematik* **87**, 97–132.
31. P.T. Thach (1993), D.c. sets, d.c. functions and nonlinear equations. *Mathematical programming* **58**, 415–428.
32. T.V. Thieu, B.T. Tam, and V.T. Ban (1985), An outer approxmation method for globally minimizing a concave function over a compact convex set. *Acta Mathematica Vietnamica* **2**.
33. N.V. Thoai (1988), A modified version of Tuy's method for solving d.c. programming problems. *Optimization* **19**, 665–674.
34. N.V. Thuong and H. Tuy (1984), A finite algorithm for solving linear programs with an additional reverse convex constraint. In V. Demyannov and H. Pallaschlke, editors, *Lecture Notes in Economics and Mathematical Systems* **225**, 291–302. Springer.
35. H. Tuy (1983), On outer approximation methods for solving concave minimization problems. *Acta Mathematica Vietnamica* **8**, 3–34.
36. H. Tuy (1986), A general deterministic approach to global optimization via d.c. programming. In J.B. Hiriart-Urruty, editor, *Fermats Days 1985: Mathematics for Optimization*, 137–162. North-Holland, Amsterdam.
37. H. Tuy (1987), Global minimization of a difference of two convex functions. *Mathematical Programming Study* **30**, 150–182.
38. L. Vigidal and S. Director (1982), A design centering algorithm for nonconvex regions of acceptability. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 13–24.
39. A.B. Zalessky (1980), Nonconvexity of feasible domains and optimization of management decisions. *Ekonomika i Mat. Metody*.