ACTA MATHEMATICA VIETNAMICA
Volume 8 Number 1 (1983)

AN OUTER APPROXIMATION METHOD FOR GLOBALLY MINIMIZING A CONCAVE FUNCTION OVER A COMPACT CONVEX SET

TRAN VU THIEU, BUI THE TAM AND VU THIEN BAN
Institute of Mathematics,
Hanoi

1. INTRODUCTION

Since the appearance of Tuy's first paper [6] on concave programming, optimization problems involving the global minimization of a concave or, more generally, a quasi-concave function, have received a more and more intensive development. This paper is devoted to the study of an important problem of this class. More specifically, we shall be concerned with the global minimization of a quasi-concave function $f(x)$ over a compact convex set D. To our knowledge, R.Horst [3] was the first to give an algorithm for solving this problem. Later, other algorithms for this problem have been developed by Tuy and Thai [7] and by K.Hoffman [2].

Horst's algorithm is a branch and bound procedure based on compact partitions of the constraint set. It requires solving an auxiliary convex programming problem at each step. Tuy and Thai's algorithm — which is a further development of some basic ideas in Thoai and Tuy's earlier algorithm [5] for concave minimization over a polytope — uses, too, a branch and bound technique but incorporates it into a cone splitting scheme. Boundings in this algorithm are based on solving relaxed subproblems obtained by cutting the corresponding subcones by means of suitable supporting hyperplanes to the constraint set.

This idea of relaxing the constraint set by using suitable supporting hyperplanes plays also a crucial role in the recent paper of Hoffman [2]. But in constrast with the two above mentioned algorithms, Hoffman's algorithm does not follow the branch and bound approach. Instead, it is an outer approximation method which generates a decreasing sequence of polytopes, $S_1 \supset S_2 \supset \dots S_k \supset \dots \supset D$, all enclosing the given constraint set and approximating this set more and more closely. At each step k, the vertex x^k achieving the minimum of the objective function over the current polytope S_k is computed. If x^k happens to be feasible, it yields an optimal solution to the problem; otherwise a suitable supporting hyperplane to D is generated that cuts off x^k from S_k and produces a new polytope S_{k+1} giving a better approximation of D than S_k.

In the sequel we shall develop an algorithm which bears some similarities to Hoffman's algorithm, inasmuch as it proceeds according to the same outer approximation scheme. However, the proposed algorithm differs from Hoffman's in several essential points : 1) unlike Hoffman's algorithm, it does not require for its intialization the prior knowledge of any interior point of the constraint set; 2) it does not require solving any auxiliary problem in each step; instead, the determination of the hyperplane for cutting off x^k and producing S_{k+1} is quite easy; 3) the procedure for deriving the vertex set V_{k+1} of S^{k+1} from V_k seems to be simpler than in Hoffman's algorithm; 4) when specialized to the linear constraints case, our algorithm yields a finite algorithm different from that of Falk-Hoffman [1].

This paper is organized as follows. Section 2 is devoted to a detailed description of the algorithm. In section 3 the convergence of the algorithm is proved. In section 4 several computational aspects of the algorithm are discussed : finding an initial enclosing polytope S_1 and the set V_1 of its vertices; computing V_{k+1} from V_k $(k \geqslant 1)$, determining redundant constraints to S_{k+1}. In section 5 the algorithm is specialized to the case where D is a polytope and in the last section two illustrative examples are given.

2. DESCRIPTION OF THE ALGORITHM

We first state a basic property of quasi-concave function, which will play a key role in our algorithm. Recall that a function $f : R^n \to R$ is said to be *quasi-concave* if for any two points $y, z \in R^n$ and for any scalar λ: $0 < \lambda < 1$, we have

$$f(\lambda y + (1 - \lambda) z) \geqslant \min \{ f(y), f(z)\},$$

i. e. the minimum of f over any line segment is achieved at one endpoint of this segment.

As an immediate consequence of this definition we get the following proposition.

PROPOSITION 1. *The minimum of a quasi-concave and lower semi-continuous function* f (x) *over a compact convex set* $D \subset R^n$ *is always attained in at least one extreme point of the set* D.

Proof. Let $\bar{x}$ be any point achieving the minimum of f over D (such a point exists because of the lower semi-conti-nuity of f and the compactness of D). Let $F_{\bar{x}}$ be the smallest face of D containing $\bar{x}$. If dim $F_{\bar{x}} = 0$, i. e. $F_{\bar{x}} = \{\bar{x}\}$, then $\bar{x}$ is an extreme point of D. Otherwise, $\bar{x}$ is a relative interior point of $F_{\bar{x}}$, so there is in $F_{\bar{x}}$ a line segment Γ containing $\bar{x}$ in its relative interior. Let y and z be the two points where the line containing Γ meets the boundary of D. Suppose, for example, that $f(y) \geqslant f(z)$. Then from the above definition we have

$$f(\bar{x}) \geqslant \min \{f(y), f(z)\} = f(z),$$

which implies that z is also a minimum point of f over D. Further, if F_z denotes the smallest face of D containg z, then dim $F_z <$ dim $F_{\bar{x}}$. Thus, if dim $F_{\bar{x}} > 0$ then we can replace $\bar{x}$ by another minimum point z with dim $F_z <$ dim $F_{\bar{x}}$. Continuing this reasoning as long as needed, we shall arrive finally at a minimum point which is also an extreme point of D.

It follows from this Proposition that the minimum of f over a polytope is always achieved at one vertex.

We now formulate the algorithm.

THE ALGORITHM.

Recall that the problem we are concerned with is the following

$$\text{Minimize } f(x), \text{ s.t. } x \in D, \qquad \textbf{(P)}$$

where f is areal-valued, quasi-concave function on R^n, D a compact convex subset of R^n. We shall assume that the function f is continuous and that D is defined by a system of the form

$$g_i(x) \leqslant 0 \qquad (i = 1, \ldots, m),$$

where g_i are real-valued functions, defined and convex throught R^n (hence continuous).

For the sake of convenience, any polytope $S \supset D$ will be called an «enclosing polytope».

Initialization. Start with an «enclosing polytope» S_1 whose vertex set V_1 is known.

Step $k = 1, 2, \ldots$ At this step we already have an enclosing polytope S_k, whose vertex set V_k is known. Let

$$x^k = \arg \min \{f(x) : x \in S_k\} \tag{1}$$

or, which amounts to the same by Proposition 1,

$$x^k = \arg\min \{f(v) : v \in V_k\}.$$

a) If $g_i(x^k) \leqslant 0$ for all $i = 1, \ldots, m$, i. e. $x^k \in D$, stop: x^k is an optimal solution of Problem (P).

b) Otherwise, let

$$I_k = \left\{ i : g_i(x^k) = \max_{1 \leqslant j \leqslant m} g_j(x^k),\ i = 1, \ldots, m \right\}.$$

Select $i_k \in I_k$ and $a_{i_k}(x^k) \in \partial g_{i_k}(x^k)$, the subdifferential of g_{i_k} at x^k (so $a_{i_k}(x^k)$ is a subgradient of g_{i_k} at x^k; the existence of such a subgradient is guaranteed by the continuity of every g_i throughout R^n). Set

$$S_{k+1} = S_k \cap \left\{ x : \langle a_{i_k}(x^k), x - x^k \rangle + g_{i_k}(x^k) \leqslant 0 \right\}. \tag{2}$$

Compute the vertex set V_{k+1} of S_{k+1} and go to step $k + 1$ (the question of how to compute V_{k+1} will be discussed later).

Remark 1. When solving the problem on a computer, it is more convenient to replace the stopping rule a) by the following one:

$$\text{Stop, if } g_i(x^k) \leqslant \varepsilon \ (i = 1, \ldots, m) \tag{*}$$

where $\varepsilon > 0$ is a given tolerance number.

From the convergence Theorem to be established in the next section, it will follow that under this stopping rule the algorithm necessarily terminates after finitely many steps.

A point x^k satisfying (*) is an approximate feasible point and in view of (1) it can be accepted as an approximate optimal solution. Of course, having obtained x^k, one could also compute the projection y^k of x^k on the convex set D, i. e. the feasible point that lies the nearest to x^k. This would require solving the convex program

$$\min \{ \| y - x^k \| : g_i(y) \leqslant 0 \quad (i = 1, \dots, m)\}.$$

Then

$$f(x^k) \leqslant \min \{f(x) : x \in D\} \leqslant f(y^k).$$

3. CONVERGENCE THEOREM

From (2) we have

$$S_1 \supset S_2 \supset \dots \supset S_k \supset S_{k+1} \supset \dots$$

and

$$S_k = S_1 \cap \{x : \langle a_{i_j}(x^j), x - x^j \rangle + g_{i_j}(x^j) \leqslant 0,\ j = 1, \dots, k-1\}. \quad (3)$$

Observe that $S_k \supset D$ for all $k = 1, 2, \dots$ Indeed, let $x \in D$, i. e. $g_i(x) \leqslant 0$ for all $i = 1, \dots, m$. Then, $x \in S_1$ (see Initialization) and since $a_{i_j}(x^j)$ is a subgradient of g_{i_j} at x^j,

$\langle a_{i_j}(x^j), x - x^j \rangle + g_{i_j}(x^j) \leqslant g_{i_j}(x) \leqslant 0$ for all $j = 1, \dots, k-1$, hence $x \in S_k$.

Furthermore, we always have $x^k \in S_k \setminus S_{k+1}$. Indeed, $x^k \in S_k$ by (1). On the other hand, from the definition of I_k it follows that

$$g_{i_k}(x^k) = \max_{1 \leqslant j \leqslant m} g_j(x^k) > 0$$

and hence,

$$\langle a_{i_k}(x^k), x^k - x^k \rangle + g_{i_k}(x^k) = g_{i_k}(x^k) > 0.$$

This implies in view of (2): $x^k \notin S_{k+1}$.

Thus, the inequality

$$h_k(x) = \langle a_{i_k}(x^k),\ x - x^k \rangle + g_{i_k}(x^k) \leqslant 0$$

excludes x^k from S_{k+1}, but does not exclude any feasible point.

With these observations in mind we now prove our basic convergence Theorem.

THEOREM. *The above algorithm either terminates after finitely many steps, yielding an optimal solution, or generates a sequence* $\{x^k\}$, *whose every limit point is an optimal solution of Problem* (P).

Proof. Let $\mu = \min\{f(x) : x \in D\}$. Suppose that at some step k we have $x^k \in D$. Then, $f(x^k) \geqslant \mu$. Since, on the other hand, $S_k \supset D$ it follows from (1) that $f(x^k) \leqslant \mu$. Therefore, $f(x^k) = \mu$, and so x^k is an optimal solution of (P).

Suppose now that the algorithm generates an infinite sequence $\{x^k\}$. We then have for all k

$$\max_{1 \leqslant j \leqslant m} g_j(x^k) > 0.$$

Since $\{x^k\} \subset S_1$ and S_1 is a compact set, we can select a subsequence $\{x^k\}_{k \in Q}$ and a natural number r such that

$$x^k \to \bar{x} \text{ as } k \underset{Q}{\to} \infty \text{ and } i_k = r \text{ for all } k \in Q.$$

Let $t, s \in Q$ and $t > s$. Then $x^t \in S_s$ and consequently, by virtue of (3):

$$\langle a_r(x^s),\ x^t - x^s \rangle + g_r(x^s) \leqslant 0. \tag{4}$$

But the function g_r being convex and continuous throughout R^n and the polytope S_1 being a compact convex set, it follows from a known result on convex analysis (see e. g. [4], Theorem 24.7) that there exists a constant K satisfying $\| a_r(x) \| \leqslant K$ for all $x \in S_1$. Therefore, as $s \underset{Q}{\to} \infty$, the first term in (4) tends to zero and we get from (4) $g_r(\bar{x}) \leqslant 0$. Furthermore, the continuity of the function $\max_{1 \leqslant i \leqslant m} g_i(x)$ implies; as $x^k \to \bar{x}$ $(k \underset{Q}{\to} \infty)$:

$$\max_{1 \leqslant i \leqslant m} g_i(\bar{x}) \leqslant 0.$$

Thus $\bar{x} \in D$, and, consequently, $f(\bar{x}) \geqslant \mu$. But, from (1) and the fact $D \subset S_k$, we have $f(x^k) \leqslant \mu$ for all k, hence by setting $k \underset{Q}{\to} \infty$, $f(\bar{x}) \leqslant \mu$. This shows that $f(\bar{x}) = \mu$, and, therefore, $\bar{x}$ is an optimal solution of (P), as was to be proved.

Remark 2. The algorithm is still covergent if instead of (2) we set

$$S_{k+1} = S_k \cap \left\{ x : \langle a_k(x^k), x - x^k \rangle + \max_{1 \leqslant i \leqslant m} g_i(x^k) \leqslant 0 \right\}$$

with $\quad a_k(x^k) = \sum_{i \in I_k} \lambda_i \, a_i(x^k),\ \lambda_i \geqslant 0,\ \sum \lambda_i = 1$ and $a_i(x^k)$ being a subgradient of g_i at x^k. In fact this amounts to treating the system of constraints $g_i(x) \leqslant 0$ $(i = 1, \ldots, m)$ as a single constraint $g(x) \leqslant 0$ with $g(x) = \max_i g_i(x)$.

Remark 3. The proof of the Theorem makes use of the continuity of the functions f and g_i on S_1 only. Therefore, the Theorem still holds, provided the function f is quasi-concave and continuous on S_1, while the functions g_i are convex and continuous on S_1.

4. IMPLEMENTATION OF THE ALGORITHM

In order to implement the above algorithm, several questions must be examined: a) how to construct the initial polytope S_1 containing D; b) how to find at each step the vertex set V_k of S_k; c) how to identify the redundant constraints among those defining S_k (these redundant constraints could be deleted).

a) Finding S_1 and V_1. To construct the initial enclosing polytope we compute (by solving at most n + 1 convex programming problems):

$$\alpha_j = \min \{x_j : x \in D\},\ j = 1, \ldots, n, \tag{5'}$$

$$M = \max \left\{ \sum_{j=1}^{n} x_j : x \in D \right\}. \tag{5''}$$

Then we set

$$S_1 = \left\{ x : -x_j + \alpha_j \leqslant 0,\ j = 1, \ldots, n;\ \sum_{j=1}^{n} x_j - M \leqslant 0 \right\}. \tag{6}$$

It is easily seen that S_1 is a simplex containing D and having exactly $n+1$ vertices

$$v^0 = (\alpha_1, \dots, \alpha_n),$$

$$v^j = (\alpha_1, \dots, \alpha_{j-1}, \beta_j, \alpha_{j+1}, \dots, \alpha_n),\ j = 1, \dots, n$$

with $\beta_j = M - \sum_{i \neq j} \alpha_i$. Thus, the vertex set of S_1 is

$$V_1 = \{v^0, v^1, \dots, v^n\}.$$

If D is contained in the orthant R^n_+ (e. g. if the constraints include the inequalities $x_j \geqslant 0, j = 1, \dots, n$), it suffices to compute M (by solving (5'')) only. Then

$$S_1 = \left\{ x : x_j \geqslant 0,\ j = 1, \dots, n;\ \sum_{j=1}^{n} x_j \leqslant M \right\} \tag{7}$$

and the vertices are : $v^0 = 0,\ v^j = Me^j,\ j = 1, \dots, n$ (e^j is the j-th unit vector in R^n).

b) Computing V_{k+1} *from* V_k. Suppose we already know the constraints defining the polytope S_k along with the vertex set V_k of this polytope. Let S_{k+1} be constructed as indicated in Section 2, i. e. let S_{k+1} be obtained by adding to the constraints defining S_k the following one

$$h_k(x) = \langle a_{i_k}(x^k), x - x^k \rangle + g_{i_k}(x^k) \leqslant 0. \tag{8}$$

In view of (3), (6), (8), S_{k+1} consists of all x satisfying a system of linear inequalities of the form

$$p_j(x) \leqslant 0,\ j = 1, \dots, n + k + 1, \tag{9}$$

where

$$\begin{cases} p_j(x) = -x_j + \alpha_j \text{ or } -x_j \text{ for all } j = 1, \dots, n, \\ p_{n+1}(x) = x_1 + \dots + x_n - M, \\ p_{n+1+j}(x) = h_j(x) = \langle a_{i_j}(x^j), x - x^j \rangle + g_{i_j}(x^j) \end{cases}$$

for all $j = 1, \dots, k$.

$$S_k \subset \{x : h_k(x) \geqslant 0\}$$

and consequently,

$$S_{k+1} = S_k \cap \{x : h_k(x) \leqslant 0\}$$
$$= S_k \cap \{x : h_k(x) = 0\} = S_k \cap H_k .$$

This shows that S_{k+1} is a face of S_k . Therefore, every vertex of S_{k+1} is also a vertex of S_k and they all lie on H_k .
Thus

$$V_{k+1} = V_k \cap H_k = V_k \setminus V_k^+.$$

Case 2: $V_k^- \neq \phi$. For each pair (u, v) with $u \in V_k^-$, $v \in V_k^+$, we define the point $w = \lambda u + (1 - \lambda)v$ with $\lambda = h_k(v)/ (h_k(v) - h_k(u))$. Then $0 < \lambda < 1$ and

$$h_k(w) = \lambda\, h_k(u) + (1 - \lambda)h_k(v) = 0.$$

So w is nothing but the intersection of the hyperplane H_k with the line segment joining $u \in V_k^-$ and $v \in V_k^+$. Now consider those constraints defining S_{k+1} that are binding at w. Clearly. if the maximal number of linearly independent binding constraint at w is equal to n, then w is a vertex of S_{k+1}, i.e. $w \in V_{k+1}$. Otherwise, $w \notin V_{k+1}$.

Denote by V_{k+1} the set of all the vertices of S_{k+1} that are generated in that way. We have:

PROPOSITION 3. *If* $V_k^- \neq \phi$, *then* $V_{k+1} = (V_k \setminus V_k^+) \cup W_{k+1}$.

Falk and Hoffman [1] have suggested a rule for finding the vertex set of S_{k+1} by performing dual pivots on simplex tableaux. We present here a different rule for generating the new vertices of S_{k+1} which seems to be simpler than that of Falk — Hoffman. Our rule involves very simple operations and requires no simplex tableau. Let

$$V_k^- = \{u \in V_k : h_k(u) < 0\},\ V_k^+ = \{v \in V_k : h_k(v) > 0\}.$$

Recall that x^k violates the additional constraint, i.e.

$$h_k(x^k) = g_{i_k}(x^k) > 0 .$$

Therefore V_k^+ is not empty. Now it is easy to see that every $v \in V_k \setminus V_k^+$ still belongs to S_{k+1} , hence still is a vertex of S_{k+1} . So we have

$$V_k \setminus V_k^+ = V_k \cap V_{k+1} . \tag{10}$$

But, of course, besides the vertices that S_{k+1} shares with S_k , S_{k+1} may have some other new vertices lying on the hyperplane

$$H_k = \{ x : h_k (x) = 0\} .$$

To determine all these new vertices let us distinguish two cases:

Case 1: $V_k^- = \phi$.

PROPOSITION 2. *If $V_k^- = \phi$, then V_{k+1} consists of all vertices of S_k lying on the hyperplane H_k , i.e.*

$$V_{k+1} = \{v \in V_k : h_k (v) = 0\} = V_k \setminus V_k^+ .$$

Proof. Since $V_k^- = \phi$, i.e. $h_k(v) \geqslant 0$ for all $v \in V_k$, we have.

Proof. In view of (10), it suffices to show that every new vertex of S_{k+1}, i.e. every $w \in V_{k+1} \setminus V_k$. lies on some edge of S_k connecting a vertex $u \in V_k$ with a vertex $v \in V_k^+$. Indeed, since $w \in V_{k+1}$ there are among the constraints defining S_{k+1} (see (9)) n linearly independent constraints binding for w. Further since $w \notin V_k$, one of these n binding constraints must be the one that has just been added:

$$p_{n+k+1} (x) = h_k(x) \leqslant 0$$

Denote by R_k the set of indices of $n - 1$ remaining binding constraints. Of course, $R_k \subset \{1,..., n + k\}$. Let

$$Z = \left\{x : p_j(x) = 0, \ j \in R_k ; \ p_j(x) \leqslant 0, \ j \in \{1,..., n + k\} \setminus R_k\right\}$$

Then Z is a face of S_k and $w \in Z$. Certainly, $Z \neq \{w\}$, for otherwise w would be a vertex of S_k. Furthermore, since the constraints $p_j(x) = 0$, $j \in R_k$, are linearly independent, and $|R_k| = n - 1$ it follows that dim $Z = 1$. Therefore, Z is an edge of S_k. If u, v denote the two endpoints of Z, then they both belong to V_k. But, since $w \notin V_k$, w is distinct from u, v. So $w = \lambda u + (1 - \lambda)v$ for some $\lambda : 0 < \lambda < 1$.

Finally, from the relation

$$h_k(w) = \lambda h_k(u) + (1 - \lambda) h_k(v) = 0$$

it follows that $h_k(u) . h_k(v) < 0$, as was to be proved.

Remark 4. Instead of computing the maximal number of independent constraints binding for w, an alternative way to check whether $w \in W_{k+1}$ is the following.

Denote by $N_k(w)$ the index set of the constraints of the from (9) that define S_k and are binding for both u and v (consequently, for w), i. e.

$$N_k(w) = \{ j : p_j(u) = p_j(v) = 0, \ j = 1,\dots, n+k \}$$
$$= \{ j : p_j(w) = 0, \ j = 1,\dots, n+k \}$$

Then, as can easily be verified,

$Z(w) = \{x : p_j(x) = 0, \ j \in N_k(w), p_j(x) \leqslant 0, \ j \in \{1,\dots, n+k\} \setminus N(w)\}$ is the smallest face of S_k containing u and v. Therefore, if $|N_k(w)| < n-1$ or if $|N_k(w)| \geqslant n-1$ and there exists a vertex $z \in V_k \setminus \{u, v\}$ satisfying $p_j(z) = 0$ for all $j \in N_k(w)$ (i. e. $z \in Z(w) \cap V_k$), then dim $Z(w) > 1$ and hence, by Proposition 3, w cannot be a vertex of S_{k+1} : $w \notin W_{k+1}$. Otherwise, *dim* $Z(w) = 1$ and w is a vertex of S_{k+1} ; $w \in W_{k+1}$.

c) *Identifying the redundant constraints of* S_{k+1}. As previously indicated, S_{k+1} is defined by constraints of the form (9). A constraint $p_{j_0}(x) \leqslant 0$ is said to be *redundant* (for S_{k+1}) if the removal of it does not change set S_{k+1}, i.e.

$$S_{k+1} = \{x : p_j(x) \leqslant 0, \ j = 1,\dots, n+k+1\}$$
$$= \{x : p_j(x) \leqslant 0, \ j = 1,\dots, n+k+1 \text{ and } j \neq j_0\}.$$

A constraint non redundant for S_{k+1} is called *essential* for S_{k+1}. Denote by J_{k+1} the index set of the essential constraints to S_{k+1}. Of course, all the constraints defining S_1 are essential (see (6), (7)) : $J_1 = \{1,\dots, n+1\}$. Next, the constraints $p_{n+k+1}(x) = h_k(x) \leqslant 0$ is always essential for S_{k+1} (because $S_{k+1} = S_k \cap \{x : h_k(x) \leqslant 0\}$ and $x^k \in S_k \setminus S_{k+1}$), so $n+k+1 \in J_{k+1}$. Further, a constraint redundant for S_p will be redundant also for S_q, if $q > p$, i. e. $j \notin J_q$ implies $j \notin J_q$ for all $q > p$.

Recall that V_{k+1} (the vertex set of S_{k+1}) consists generally of three kinds of vertices: the vertices belonging to V_k and satisfying $h_k(x) < 0$, the vertices beolonging to $V_k \cap H_k$, and the newly generated vertices (which all lie on H_k). We have

PROPOSITION 4. *Assume $V_k^- \neq \phi$ and $j_o \in J_k$. The constraint $p_{j_o}(x) \leqslant 0$ is redundant for S_{k+1} if and only if $p_{j_o}(u) < 0$ for all $u \in V_k^-$.*

Proof. To prove the «if» part, suppose $p_{j_o}(u) < 0$ for all $u \in V_k^-$. We first observe that $p_{j_o}(x) < 0$ for all $x \in S_{k+1} \setminus H_k$. Indeed, every point $x \in S_{k+1} \setminus H_k$ can be expressed in the form

$$x = \sum_{u \in V_k^-} \lambda_u u + \sum_{v \in V_{k+1} \cap H_k} \mu_v v$$

with $\lambda_u \geqslant 0$, $\mu_v \geqslant 0$ and $\Sigma\lambda_u + \Sigma \mu_v = 1$; moreover, there must be at least one $\lambda_u > 0$. From the hypotheses and the fact that $p_{j_o}(v) \leqslant 0$ for all $v \in V_{k+1} \cap H_k$,

$$p_{j_o}(x) = \Sigma \lambda_u p_{j_o}(u) + \Sigma \mu_v p_{j_o}(v) < 0.$$

This shows that

$$x \in S_{k+1} \text{ and } p_{j_o}(x) = 0 \text{ imply } h_k(x) = 0. \qquad (11)$$

Now let

$$S'_{k+1} = \{x : p_j(x) \leqslant 0,\ j \in J_k \setminus \{j_o\},\ h_k(x) \leqslant 0\}.$$

We shall show that $p_{j_o}(x) \leqslant 0$ for all $x \in S'_{k+1}$. Indeed, suppose the contrary that there is $z \in S'_{k+1}$ such that $p_{j_o}(z) > 0$. Take any vertex $u \in V_k^-$. From the hypotheses, $p_{j_o}(u) < 0$. Let

$$y = \lambda u + (1 - \lambda) z \text{ with } \lambda = p_{j_o}(z) / (p_{j_o}(z) - p_{j_o}(u)).$$

It is easily seen that $0 < \lambda < 1$ and

$$p_{j_o}(y) = \lambda p_{j_o}(u) + (1 - \lambda) p_{j_o}(z) = 0,$$

$$p_j(y) \leqslant 0 \text{ for all } j \in J_k \setminus \{j_o\} \text{ and } h_k(y) \leqslant 0.$$

This means that $y \in S_{k+1}$ and $p_{j_o}(y) = 0$. Hence, by virtue of (11),

$$h_k(y) = \lambda h_k(u) + (1 - \lambda) h_k(z) = 0,$$

conflicting with $h_k(u) < 0$, $h_k(z) \leqslant 0$. Therefore,

$$p_{j_o}(x) \leqslant 0 \text{ for all } x \in S'_{k+1}$$

This implies $S^{\bullet}_{k+1} = S_{k+1}$, and so the constraint $p_{j_o}(x) \leqslant 0$ is redundant for S_{k+1}.

To prove the «only if» part it suffices to show that if $p_{j_o}(u) = 0$ for some $u \in V_k^{-}$, then the constraint $p_{j_o}(x) \leqslant 0$ is essential for S_{k+1}. Indeed, since $u \in V_k^{-}$ (i. e. $h_k(u) > 0$), there exists a ball U around u such that

$$h_k(x) < 0 \text{ for all } x \in U.$$

Since $j_o \in J_k$, i.e. the constraint $p_{j_o}(x) \leqslant 0$ is essential for S_k, there exists a point z satisfying $p_{j_o}(z) > 0$ and $p_j(z) \leqslant 0$ for all $j \in J_k \setminus \{j_o\}$.

Let $y = \varepsilon z + (1 - \varepsilon) u$, where ε is some positive number so small that $y \in U$. Then

$$\begin{cases} p_{j_o}(y) = \varepsilon p_{j_o}(z) + (1 - \varepsilon) p_{j_o}(u) = \varepsilon p_{j_o}(z) > 0, \\ p_j(y) = \varepsilon p_j(z) + (1 - \varepsilon) p_j(u) \leqslant 0 \text{ for all } j \in J_k \setminus \{j_o\}, \\ h_k(y) > 0. \end{cases}$$

This shows that the constraint $p_{j_o}(x) \leqslant 0$ is also essential for S_{k+1}. The proof is complete.

Remark 5. The assumption $j_o \in J_k$ is required only in the «only if» part. Thus, Proposition 4 gives a sufficient condition to identify a redundant constraint for S_{k+1} in the case $V_k^{-} \neq \phi$.

The next Proposition can be used when $V_k^{-} = \phi$ (Then by Proposition 2, $V_{k+1} = V_k \setminus V_k^{+}$).

PROPOSITION 5. *If $p_{j_o}(v) < 0$ for all $v \in V_{k+1}$, then the constraint $p_{j_o}(x) \leqslant 0$ is redundant for S_{k+1}.*

Proof. For every $x \in S_{k+1}$ we have

$$x = \sum_{v \in V_{k+1}} \lambda_v v \text{ with } \lambda_v \geqslant 0, \ \Sigma \lambda_v = 1,$$

hence

$$p_{j_o}(x) = \Sigma \lambda_v p_{j_o}(v) < 0.$$

This means that $p_{j_o}(x) \leqslant 0$ is redundant for S_{k+1}.

Remark 6. The following example shows that the condition stated in Proposition 5 may not be necessary (even if $j_o \in J_k$). Let S_k be the polytope obtained when cutting a 3-dimensional cube by the plane DEG (see Fg. 1). S_k has seven faces corresponding to seven constraints. It is easily seen that all these constraints are essential for S_k. Suppose S_{k+1} is the base $ABCD$. Then, V_{k+1} consists of four vertices A, B, C, D and the constraint corresponding to the section DEG is redundant for S_{k+1} although it does not fulfil the condition stated in Proposition 5 (because it is binding at D).

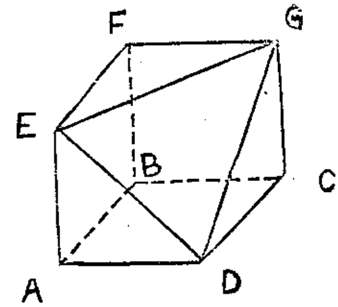

Fig. 1

The above results justify the following procedure for identifying the redundant constraints of S_{k+1}.

Procedure T. Consider the set V_k^- (if it is not empty) or the set $V_{k+1} = V_k \setminus V_k^+$ (otherwise). Let this set consist of elements $u^1, \ldots, u^T$ (in arbitrary order). Mark the constraints defining S_k that have not been deleted up to step k and that are binding for u^1. Then, mark the ones binding for u^2 that have not yet been marked. And so on, until the last vertex u^T has been reached or all the constraints have been marked. The constraints that remnin unmarked after this procedure will be redundant for S_{k+1} and hence can be deleted. Therefore, S_{k+1} is defined by the system of all constraints that have been marked, plus the constraint $h_k(x) \leqslant 0$.

Remark 7. Since in the case $V_k^- = \phi$ we have only a sufficient condition for identifying redundant constraints, the above procedure is not able to delete all the redundant constraints for S_{k+1}. But if $V_h^- \neq \phi$ for all $h = 1, \ldots, k$, then the procedure actually deletes all the redundant constraints for S_{k+1}. For instance, if D has a nonempty interior (i. e. dim $D = n$), then V_k^- is nonempty for all k (for otherwise, as seen above, $S_{k+1} = S_k \cap H_k$ is a face of S_k and hence, dim $S_{k+1} < n$, conflicting with $S_{k+1} \supset D$). Therefore, in that case for every k, the above procedure deletes all the redundant constraints for S_{k+1}.

5. SPECIALIZATION TO THE CASE OF LINEAR CONSTRAINTS.

In this section we specialize the above described algorithm to the case where D is a polytope, i.e. to the following problem:

$$\left.\begin{array}{c}\text{Minimize } f(x), \text{ subject to} \\ A_i x + b_i \leqslant 0, \; i = 1, \ldots, m, \\ x_j \geqslant 0, \; j = 1, \ldots, n,\end{array}\right\} \qquad (12)$$

where $A_i = (a_{i1}, \ldots a_{in})$ are n-dimensional row vectors, b_i are real numbers, such that the polyhedral set

$$D = \left\{ x \in R^n_+ : A_i x + b_i \leqslant 0, \; i = 1, \ldots m \right\}$$

is non-empty and bounded.

Since $g_{i_k}(x)$ is now affine, we have

$$h_k(x) \equiv g_{i_k}(x) = A_{i_k} x + b_{i_k} .$$

Finite algorithm for solving Problem (12).

1. Solve the linear problem

$$M = \max \left\{ \sum_{j=1}^{n} x_j : A_i x + b_i \leqslant 0, i = 1, \ldots, m; \; x_j \geqslant 0, \; j = 1, \ldots, n \right\}.$$

Set

$$S_1 = \left\{ x : \sum_{j=1}^{n} x_j \leqslant M, x_j \geqslant 0, \; j = 1, \ldots, n \right\}.$$

Let $J_1 = \{ m+1, \ldots, m+n+1 \}$ be the index set of the constraints defining S_1 (the index $m+j$ corresponds to the constraint $x_j \geqslant 0$, the index $m+n+1$ to the constraint $\Sigma x_j \leqslant M$).

Let

$$V_1 = \{ v^0, v^1, \ldots, v^n \}$$

with $v^0 = 0$, $v^j = Me^j$, $j = 1, \ldots, n$, (e^j is the j-th unit vector in R^n).

Set $k = 1$.

2. Select $x_k = \arg\min \{ f(v) : v \in V_k \}$ (if there are several candidates, take any one of them). Compute

$$\gamma_k = \max_{1 \leqslant i \leqslant m} \{ A_i x^k + b_i \}$$

a) If $\gamma_k \leqslant 0$, i. e. $A_i x^k + b_i \leqslant 0$ for all $i = 1, \ldots, m$, stop: x^k is an optimal solution of Problem (12).

b) Otherwise, select

$$i_k = \arg\max \{ A_i x^k + b_i \, , \; i = 1, \dots, m\}$$

and set

$$S_{k+1} = S_k \cap \{ x : A_{i_k} x + b_{i_k} \leqslant 0 \},$$

$$J_{k+1} = J_k \cup \{i_k\}.$$

Let

$$V_k^- = \{ u \in V_k : A_{i_k} u + b_{i_k} < 0 \}, \; V_k^+ = \{ v \in V_k : A_{i_k} v + b_{i_k} > 0 \}.$$

3. a) If $V_k^- = \varnothing$, set $V_{k+1} = V_k \setminus V_k^+$. Go to 4.

b) Otherwise, for each pair $(u, v) \in V_k^- \times V_k^+$ compute

$$\lambda = \frac{A_{i_k} v - b_{i_k}}{(A_{i_k} v - b_{i_k}) - (A_{i_k} u - b_{i_k})}.$$

Let $w = \lambda u + (1 - \lambda) v$. Define

$$J_{k+1}(w) = \left\{ i \in J_k : \begin{array}{c} A_i w + b_i = 0 \text{ if } i = 1, \dots, m \\ w_{i-m} = 0 \text{ if } i = m + 1, \dots, m + n \\ \sum_{j=1}^{n} w_j = M \text{ if } i = m + n + 1 \end{array} \right\}$$

and let W_{k+1} be the set of all w corresponding to all pairs (u, v) for which $|J_{k+1}(w)| \geqslant n - 1$ and there exists no $z \in V \setminus \{u, v\}$ satisfying

$$\begin{cases} A_i z + b_i = 0 \text{ for } i \in J_{k+1}(w) \cap \{1, \dots, m\}, \\ z_{i-m} = 0 \text{ for } i \in J_{k+1}(w) \cap \{m + 1, \dots, m + n\}. \\ \sum_{j=1}^{n} z_j = M \text{ if } m + n + 1 \in J_{k+1}(w). \end{cases}$$

Set

$$V_{k+1} = (V_k \setminus V_k^+) \cup W_{k+1}.$$

4. Set $k + 1 \leftarrow k$ and return to 2.

Since at each step the current polytope S_k is obtained from the previous one, S_{k-1}, by adding one new constraint, since all these constraints are taken from the system (12), it is easily seen that the algorithm stops after at most m steps.

6. ILLUSTRATIVE EXAMPLES

We first consider Hoffman's example [2] (nonlinear constraint case):

Minimize $f(x_1, x_2) = -(x_1 - x_2)^2 / 2x_1$,

subject to $g_1(x_1, x_2) = -28x_1 + 9x_2 + 21 \leqslant 0$,

$$g_2(x_1, x_2) = 9x_1^2 - 72x_1 + 16x_2^2 \leqslant 0,$$

$$g_3(x_1, x_2) = 64x_1^2 - 192x_1 - 36x_2 + 153 \leqslant 0.$$

Initialization. The initial enclosing polytope S_1 is chosen as follows

$$S_1 = \{(x_1, x_2) : 0.5 \leqslant x_1,\ 0 \leqslant x_2,\ x_1 + x_2 \leqslant 6\}.$$

The vertices of S_1 are (0. 5, 0); (6,0); (0. 5, 5. 5).

Step 1. The vertex of S_1 having minimum objective function value is $x^1 = (0.5, 5.5)$ with $f(x^1) = -25$ and $g_1(x^1) = 47.5$; $g_2(x^1) = 450.25$; $g_3(x^1) = -125$. We select $i_1 = 2$. The new constraint is

$$h_1(x_1, x_2) = -63x_1 + 176x_2 - 486.25 \leqslant 0. \qquad \text{(I)}$$

Thus

$$S_2 = S_1 \cap \{(x_1, x_2) : h_1(x_1, x_2) \leqslant 0\}.$$

The vertices of S_2 are $(0.5, 0)$; $(6, 0)$: $(0.5, 2.94176)$ and (2.38389, 3.61611).

Step 2. The minimum of f over S_2 is achieved at the vertex x^2 $(0.5, 2.94176)$ with $f(x^2) = -5.9621919$ and $g_1(x^2) = 30.77584$; $g_2(x^2) = 104.71323$; $g_3(x^2) = -32.90336 < 0$.

Again $i_2 = 2$. The new constraint is

$$h_2(x_1, x_2) = -63x_1 + 94.13632x_2 - 140.71323 \leqslant 0 \qquad \text{(II)}$$

and

$$S_3 = S_2 \cap \{(x_1, x_2) : h_2(x_1, x_2) \leqslant 0\}.$$

S_3 has four vertices: $(0.5, 0)$; $(6, 0)$; $(0.5, 1.829403)$ and $(2.69896, 3.30104)$.

After twelve more steps we arrive at the solution $x^{14} = (1.6578327, 0.2942215)$ with objective function value $-0,5608031$ and

$$\max \{g_1(x^{14}),\ g_2(x^{14}),\ g_3(x^{14})\} = 0,00234.$$

Fig. 2 depicts the feasible set D, the initial enclosing polytope S_1 and the constraints added up to the step 9.

The data relative to these steps can be found in Table 1.

Table 1

k	$x^k = (x_1^k, x_2^k)$	$f(x^k)$	$\max g_i(x^k) : i = 1, 2, 3$
1	(0.5,5.5)	—25	450.25
2	(0.5,2.94176)	—5.9621919	104.71323
3	(6,0)	—3	1305
4	(3.73437, 0)	—1.867185	328.5228
5	(0.5, 1.829403)	—1.767304	23.4646
6	(2.58572, 0.0)	—1.29286	84.442426
7	(1.97808, 0.0)	—0.98904	23.62787
8	(1.5913674, 0.0)	—0.7256837	9.5343
9	(1.78472, 0.327657)	—0.594778	2.392538
10	(1.6880449, 0.296248)	—0.5737694	0.598151
11	(1.6397072, 0.2805448)	—0.5633086	0.149546
12	(1.6638756, 0.2967041)	—0.561688	0.037386
13	(1.6759594, 0.3047833)	—0.5609097	0.009352
14	(1.6578327, 0.2942215)	—0.5608031	0.00234

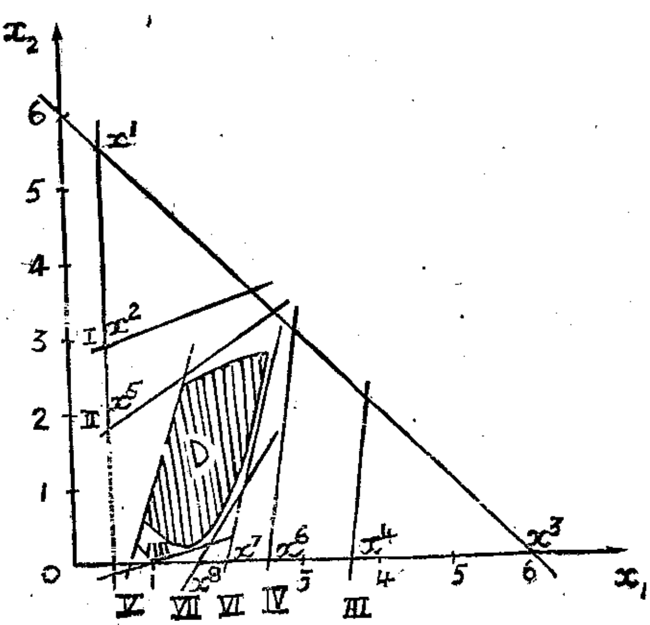

Fig 2

As a second example, we solve the problem (linear constraint case):

$$f = -3x_1^2 - 2r_2^2 \to min$$

$$-2x_1 - 3x_2 + 6 \leqslant 0 \quad (1)$$

$$x_1 + x_2 - 10 \leqslant 0 \quad (2)$$

$$-x_1 + 2x_2 - 8 \leqslant 0 \quad (3)$$

$$x_1 - x_2 - 4 \leqslant 0 \quad (4)$$

$$x_1 \geqslant 0 \quad (5)$$

$$x_2 \geqslant 0 \quad (6)$$

Initialization.

$M = 10 = max \{x_1 + x_2 : (x_1, x_2) \text{ satisfying } (1) - (6)\}$,

$V_1 = \{(0,0); (10,0); (0,10)\}$.

Associated objective function values

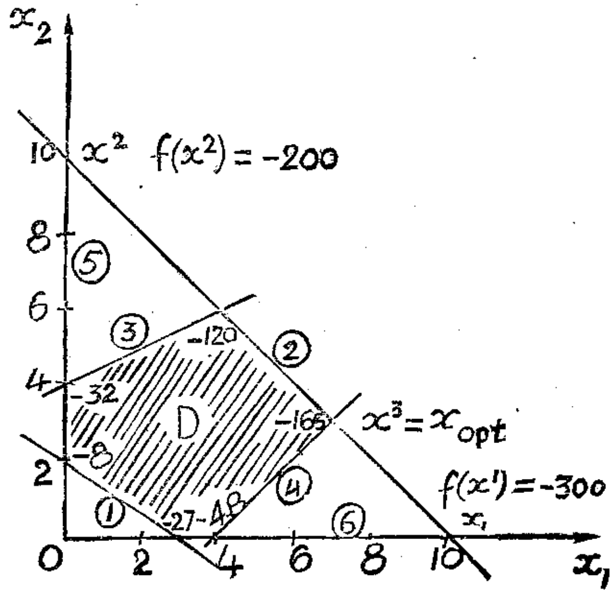

Fig. 3

$$f = \{ 0, -300, -200 \}$$

Step 1. $x^1 = (10,0)$; $f(x^1) = -300$,

$\gamma_1 = max \{ -14,0, -18, 6 \} = 6 > 0$,

$i_1 = 4$,

$V_1^- = \{ (0,0); (0,10) \}$; $V_1^+ = \{(10,0) \}$,

$V_2 = \{ (0,0); (0,10); (4,0); (7,3) \}$,

$f = \{ 0, -200, -48, -165 \}$.

Step 2. $x^2 = (0,10)$; $f(x^2) = -200$,

$\gamma_2 = max \{ -24,0, 12, -14 \} = 12 > 0$,

$i_2 = 3$,

$V_2^- = \{ (0,0); (4,0); (7,3) \}$; $V_2^+ = \{ (0,10) \}$,

$V_3 = \{ (0,0); (4,0); (7,3); (0,4); (4,6) \}$.

$f = \{ 0, -48, -165, -32, -120\}$

Step 3. $x^3 = (7,3)$; $f(x^3) = -165$,

$\gamma_3 = max \{-17, 0, -9,0\} = 0$.

Stop: $x^3 = (7,3)$ is an optimal solution, with objective function value $f(x_{opt}) = -165$.

Received May 28, 1982

REFERENCES

[1] J. Falk, K. L. Hoffman. *A Successive Underestimation Method for Concave Minimization Problems*. Math. Oper. Res. 1 (1976), 251 — 259.

[2] K. L. Hoffman. *A Method for Globally Minimizing Concave Functions over Convex Sets*. Math. Programming 20 (1981), 22 — 32.

[3] R. Horst. *An Algorithm for Nonconvex Programming Problems*. Math. Programming 10 (1976), 312 — 321.

[4] R. T. Rockafellar. *Convex Analysis*. Princeton University Press, Princeton, N. J., 1970.

[5] Ng. V. Thoai, H. Tuy. *Convergent Algorithms for Minimizing a Concave Function*. Math. Oper. Res. 5 (1980), 556 — 566.

[6] H. Tuy. *Concave Programming under Linear Constraints*. Doklady Acad. Nauk 159 (1964), 32 — 35. Translated Soviet Math. 5 (1964), 1437 — 1440.

[7] H. Tuy, Ng. Q. Thai. *Minimizing a Concave Function over a Compact Convex Set*. Procedeings of a Conferenee on Optimization held in Vitte / Hiddensee, May 1981.