Journal of Global Optimization **18:** 1–15, 2000.

A New Approach to Optimization Under Monotonic Constraint

HOANG TUY and LE TU LUC
Institute of Mathematics, P.O. Box 631, Bo Ho, Hanoi, Vietnam
E-mail: htuy@hn.vnn.vn

(Received 8 June 1999; accepted in revised form 30 November 1999)

Abstract. A new efficient branch and bound method is proposed for solving convex programs with an additional monotonic nonconvex constraint. Computational experiments demonstrated that this method is quite practical for solving rank k reverse convex programs with much higher values of k than previously considered in the literature and can be applied to a wider class of nonconvex problems.

Key words: Monotonic constraint; Rank k reverse convex programs; Polyblock outer approximation; Convex multiplicative constraint

1. Introduction

Despite the difficulties of global optimization, significant progress has been achieved in recent years in the study of specially structured and yet important global optimization problems arising from applications. For instance, efficient decomposition methods are now available for solving many low rank nonconvex minimization problems, in which the objective function is quasiconcave and monotone increasing along any halfline parallel to a ray in the nonnegative orthant (see e.g. [2] and [9]).

In the present paper we will be concerned with the following problem

$$\max\{\langle c, x\rangle \,|\, x \in D,\ \varphi(g(x)) \leq 1\} \tag{Q}$$

where D is a compact convex set in R^n, $g : R^n_+ \to R^m_{++}$ is a component-wise convex mapping and $\varphi : R^m_+ \to R$ is a lower semi-continuous function, *increasing* on R^m_+, in the sense that $\varphi(y') \geqslant \varphi(y)$ whenever $y' \geqslant y$. Of course, the last constraint $\varphi(g(x)) \leqslant 1$ constitutes the main source of difficulty because without it problem (Q) would merely reduce to a convex program. A particular case which has already been studied in the literature is when $\varphi(y) = \Pi_{i=1}^m y_i$. The constraint set is then

$$\left\{ x \in D \,\middle|\, \prod_{i=1}^m y_i \leqslant 1,\ g_i(x) \leqslant y_i\ (i = 1, \ldots, m) \right\}$$

and since the set $\{y \,|\, \prod_{i=1}^m y_i \geqslant 1\}$ is convex, the problem is a convex program with an additional reverse convex constraint. For this class of problems several solution methods are available, some of which quite efficient if $m \leqslant 5$ [3–7]. Of particular

interest is the outer approximation method in [4] (see also [2, 9]), which can also be extended to problems with reverse convex constraints of a more general type than convex multiplicative constraints. However, to our knowledge so far no author has considered monotonic constraints described by means of arbitrary increasing functions $\varphi(\cdot)$.

Although nonconvex, the (lower) level sets of increasing functions enjoy a nice property analogous to the separation property of convex sets. Namely, whereas a convex set can be separated from any point outside by a halfspace, a level set of an increasing function can be separated from any point outside by a translate of the positive orthant. Using this specific separation property, a method analogous to the standard outer approximation method for convex maximization over convex sets (see e.g. [2] and [9]) was developed in [8] for maximizing an increasing function over a level set of another increasing function. Preliminary computational results suggested that the method is quite practical and could open the way to new promising developments of global optimization focussed on exploiting monotonicity properties.

It turns out that problem (Q) can be converted to a problem essentially of the same type as the one investigated in [8]. This allows problem (Q) to be approached by the same outer approximation method as was developed in the latter paper. Close scrutiny showed that when applied to problems with $\varphi(y) = \prod_{i=1}^{m} y_i$, our method coincides basically with the outer approximation method RECT of Kuno et al. [4] for convex programs with an additional constraint on the product of several convex functions. In fact, both methods use the same separation property of the feasible set to approximate the latter by a sequence of sets which are unions of rectangles (i.e. 'polyblocks', following the terminology introduced in [10]). In the sequel, however, we shall describe our method as a branch and bound rather than an outer approximation procedure, using the simple observation that if $f(x)$ is an increasing function on R^n_+ then an upper bound of $f(x)$ over the feasible solutions contained in a box $[a, b] = \{x \in R^n \mid a \leqslant x \leqslant b\}$ is obviously $f(b)$. Aside from being applicable to a wider range of problems, the present method differs from the method in [8] or RECT in [4] in several important aspects: (1) no special manipulation is required for the convergence of the algorithm, (2) a more systematic and efficient procedure is used for computing the new vertices of the current outer approximating polyblock (in particular, improper vertices and vertices no longer of interest are discarded, thus avoiding unnecessary computations), (3) a more rigorous theoretical foundation is provided, (4) reoptimization techniques are employed in solving the linear subproblems, to save computational efforts, and (5) restarts are made possible, to overcome memory space limitations and enhance computational efficiency.

Just as the earlier outer approximation method in [8], the present branch and bound method is a specialized version of a very general approach to monotonic optimization [10]. It is easy to implement and when applied to convex programs with an additional convex multiplicative constraint it seems to outperform most existing methods.

The paper is organized as follows. After the Introduction we shall describe in Section 2 preliminary transformations to reduce the problem to maximizing an increasing function over the level set of another increasing function. Section 3 is devoted to an analysis of basic properties of the objective function and the constraint set which can be exploited for devising efficient solution strategies. Section 4 describes the branch and bound method proposed for the solution of the problem. Section 5 discusses some features useful for the implementation, while Section 6 illustrates the Algorithm on selected numerical examples. Finally, we close the paper with a report on preliminary computational experiments.

2. Preliminary Transformations

For the sake of convenience let us reformulate the problem:

$$\max\{\langle c, x\rangle \mid x \in D,\ \varphi(g(x)) \leqslant 1\} \tag{Q}$$

where D is a compact convex set in R^n, $g = (g_1, \ldots, g_m)$ with each $g_i : D \to R_{++}$, $i = 1, \ldots, m$, being a continuous convex positive-valued function, and $\varphi : R^m_+ \to R$ is a lower semi-continuous function, increasing on R^m_+, i.e. such that for all $y, y' \in R^m_+$:

$$y \geqslant y' \Rightarrow \varphi(y) \geqslant \varphi(y')\,. \tag{1}$$

We also assume, naturally, that

$$\varphi(0) < 1\,. \tag{2}$$

We first make some preliminary remarks and show how to convert the problem into a form exhibiting monotonicity not only in the constraints but also in the objective function.

LEMMA 1. *Under the stated assumptions, the feasible set of the problem (Q) is compact.*

Proof. Consider any sequence $\{x^k\}$ of feasible solutions. Since $x^k \in D$ and D is compact, there exists a subsequence $\{x^{k_q}\}$ such that $x^{k_q} \to x^0 \in D$. By continuity of $g(x)$ we have $g(x^{k_q}) \to g(x^0)$, then, since $\varphi(g(x^{k_q})) \leqslant 1$, the lower semicontinuity of $\varphi(y)$ implies that $\varphi(g(x^0)) \leqslant \liminf \varphi(g(x^{k_q})) \leqslant 1$. Therefore, any sequence of feasible solutions contains a subsequence converging to a feasible solution. □

Thus, the problem (Q) is either infeasible or has a finite optimal solution. Define

$$C = g(D) + R^m_+ = \{y \in R^m_+ \mid y \geqslant g(x) \text{ for some } x \in D\} \tag{3}$$

$$\theta(y) = \begin{cases} \sup\{\langle c, x\rangle \mid x \in D,\ g(x) \leqslant y\} & \text{if } y \in C \\ -M & \text{if } y \notin C \end{cases} \tag{4}$$

where M is an arbitrary positive number such that $-M < \min\{\langle c, x\rangle \mid x \in D\}$. Clearly $\theta(y) < +\infty\ \forall y \in R^m_+$ and $-M < \theta(y)$ if and only if $y \in C$.

LEMMA 2. *The set C is convex, closed, and the function $\theta(y)$ is increasing on R^m_+, upper semi-continuous and concave on C.*

Proof. Let $y^k \in C$, $y^k \to y^0$. Then $y^k \geq g(x^k)$ for some $x^k \in D$. Since D is compact, we can assume, by passing to a subsequence if necessary, that $x^k \to x^0 \in D$, hence, by continuity of $g(x)$: $y^0 \geq g(x^0)$, i.e. $y^0 \in C$. Therefore, C is closed. The convexity of C is obvious. To prove that $\theta(y)$ is increasing, just observe that if $y \leq y'$ and $y \notin C$ then $\theta(y) = -M$ so $\theta(y') \geq \theta(y)$; whereas, $y' \geq y \in C$ implies that $y' \in C$ and $\{x \in D \mid g(x) \leq y\} \subset \{x \in D \mid g(x) \leq y'\}$, hence $\theta(y) \leq \theta(y')$. To see that $\theta(y)$ is concave on C, let $y^1, y^2 \in C$, $\alpha \in [0, 1]$ and for each $i = 1, 2$ let $\langle c, x^i \rangle = \max\{\langle c, x\rangle \mid x \in D, \quad g(x) \leq y^i\}$. Then $y := \alpha y^1 + (1-\alpha)y^2 \in C$, $x := \alpha x^1 + (1-\alpha)x^2 \in D$, $g(\alpha x^1 + (1-\alpha)x^2) \leq \alpha y^1 + (1-\alpha)y^2 = y$, hence $\theta(y) \geq c(\alpha x^1 + (1-\alpha)x^2) = \alpha\theta(y^1) + (1-\alpha)\theta(y^2)$. Finally, to show the upper semi-continuity of $\theta(y)$ consider any sequence $y^k \to y^0$ such that $y^k \in C$, and let $\theta(y^k) = \langle c, x^k \rangle$ with $x^k \in D$ satisfying $g(x^k) \leq y^k$. Since D is compact, we can assume that $x^k \to x^0 \in D$, so that $g(x^k) \to g(x^0) \leq y^0$ by continuity of $g(x)$, hence $y^0 \in C$, which implies that $\langle c, x^0 \rangle \leq \theta(y^0)$, and consequently, $\limsup \theta(y^k) \leq \theta(y^0)$, as desired. □

PROPOSITION 1. *Problem (Q) is equivalent to*

$$\max\{\theta(y) \mid \varphi(y) \leq 1,\ y \in C \subset R^m_+\} \tag{MQ}$$

where both $\theta(y)$ and $\varphi(y)$ are increasing functions on R^m_+.

Proof. We observe that (Q) is equivalent to the problem

$$\max\{\langle c, x\rangle \mid x \in D,\ g(x) \leq y,\ \varphi(y) \leq 1\}. \tag{5}$$

Indeed, if $x \in D$, $g(x) \leq y$ and $\varphi(y) \leq 1$, then $\varphi(y) \geq \varphi(g(x))$, hence $\varphi(g(x)) \leq \varphi(y) \leq 1$, i.e. the feasible set of (Q) is actually the same as that of (5). It remains to show that in turn (5) is equivalent to (MQ). But if $\bar{y}$ is an optimal solution of (MQ) then $\varphi(\bar{y}) \leq 1$ and $\bar{y} \in C$, hence $\theta(\bar{y}) = \langle c, \bar{x}\rangle$ for some $\bar{x} \in D$, such that $g(\bar{x}) \leq \bar{y}$. This implies that $\theta(\bar{y}) \leq \max\{\langle c, x\rangle \mid x \in D,\ g(x) \leq \bar{y},\ \varphi(\bar{y}) \leq 1\}$. The converse is true because for any feasible solution x of (5) we have $x \in D$, $g(x) \leq y$, $\varphi(y) \leq 1$, hence y is feasible to (MQ) and $\langle c, x\rangle \leq \theta(y) \leq \theta(\bar{y})$. □

3. Basic Properties

Given two points y^1, y^2 in R^m such that $y^1 \leq y^2$ we write

$$[y^1, y^2] = \{y \mid y^1 \leq y \leq y^2\}, \qquad (y^1, y^2] = \{y \mid y^1 < y \leq y^2\}.$$

PROPOSITION 2. *We have $g(D) \subset [a, b]$, where for $i = 1, \ldots, n$:*

$$0 < a_i \leq \min\{g_i(x) \mid x \in D\} \leq \max\{g_i(x) \mid x \in D\} \leq b_i < +\infty.$$

Proof. Immediate, since $g(D)$ is compact, as we already saw. □

From $b_i \geqslant g_i(x)\ \forall x \in D$ it follows that $b \in C := g(D) + R_+^m$ and hence $\theta(b) > -M$. Furthermore, if $\min\{g_i(x) \mid x \in D\} = \max\{g_i(x) \mid x \in D\} = \eta_i$ for some i then any feasible solution x of (Q) satisfies $g_i(x) = \eta_i$, so the constraint $y_i = \eta_i$ can be added to problem (MQ), i.e. the number of variables in (MQ) can be reduced by 1. Therefore, without loss of generality we can assume $\min\{g_i(x) \mid x \in D\} < \max\{g_i(x) \mid x \in D\}\ \forall i = 1, \ldots, m$. The problem can now be written as

$$\max\{\theta(y) \mid y \in C \cap G\}, \tag{MQ}$$

where

$$C = g(D) + R_+^m, \qquad G = \{y \in R_+^m \mid \varphi(y) \leqslant 1\}. \tag{6}$$

Since $\varphi(y)$ is lower semi-continuous, G is closed. The next properties of G (see (6)), even though very simple, will be fundamental for the solution method to be proposed.

PROPOSITION 3. *The sets C and G satisfy*

$$y \notin C \Rightarrow y' \notin C, \quad \forall y' \in [0, y] \tag{7}$$

$$y \in G \Rightarrow y' \in G, \quad \forall y' \in [0, y]. \tag{8}$$

Proof. If $y' \in C$, i.e. $y' \geqslant g(x)$ for some $x \in D$ then $y \geqslant y'$ implies that $y \geqslant g(x)$ for the same x, hence $y \in C$. This proves (7). To prove (8) observe that if $y \geqslant 0$, $\varphi(y) \leqslant 1$ and $0 \leqslant y' \leqslant y$ then obviously $\varphi(y') \leqslant \varphi(y) \leqslant 1$. □

The properties (8)-(7) are expressed by saying that G is a *normal set* and C a *reverse normal set*. A point $y \in R_+^m$ is called an *upper boundary point* of G if $\lambda y \in G\ \forall \lambda \in [0, 1)$ but $\lambda y \notin G\ \forall \lambda > 1$. The set of all upper boundary points of G is called the *upper boundary* of G and denoted by $\partial^+ G$.

PROPOSITION 4. *For every $y \in R_+^m \backslash G$ the ray from 0 through y meets $\partial^+ G$ at a unique point $\pi(y)$, defined by*

$$\pi(y) = \mu y \quad \textit{with } \mu = \max\{\alpha \mid \alpha y \in G\}. \tag{9}$$

Proof. Since $\varphi(0) < 1$ (see (2)) while $\varphi(y) > 1$ because $y \notin G$ and the function $\alpha \mapsto \varphi(\alpha y)$ is lower semi-continuous and increasing for $\alpha > 0$, the set $\{\alpha \mid \varphi(\alpha y) \leqslant 1\}$ is a segment. Hence, $\mu = \max\{\alpha \mid \varphi(\alpha y) \leqslant 1\}$ exists and is unique. □

PROPOSITION 5. *Let $K_z = \{y \in R_+^m \mid y_i > z_i\ \ \forall i \in I(z)\}$, where $I(z) = \{i \mid z_i > 0\}$. Then $z' \notin K_z$ for any $z, z' \in \partial^+ G$.*

Proof. For every $y \in K_z$ we have $y_i > z_i\ \forall i \in I(z)$, hence, $y \geqslant \lambda z$ for some $\lambda > 1$. But, since $z \in \partial^+ G$, it follows that $\lambda z \notin G$, i.e. $\varphi(\lambda z) > 1$. Consequently, $\varphi(y) \geqslant \varphi(\lambda z) > 1$. But $z' \in \partial^+ G$ implies that $\varphi(z') = 1$, hence $z' \notin K_z$. □

PROPOSITION 6. *If $y \notin G$ then $y \in K_{\pi(y)} \subset R^m_+ \backslash G$. Hence,*

$$G = \bigcap_{z \in \partial^+ G} (R^m_+ \backslash K_z)\,. \tag{10}$$

Proof. As we saw from the proof of Proposition 5, $\varphi(y') > 1\ \forall y' \in K_{\pi(y)}$, hence $K_{\pi(y)} \subset R^m_+ \backslash G$. To prove (10) denote by G' the set on the right-hand side of (10). If $y \notin G$ then $y \in K_{\pi(y)}$, where $\pi(y) \in \partial^+ G$, i.e. $y \notin G'$. Therefore, $G' \subset G$. The converse inclusion is plain because $G \subset R^m_+ \backslash K_z\ \forall z \in \partial^+ G$. □

The properties of G as a normal set (expressed in Proposition 6) are similar to corresponding properties of convex sets, except that, while the separation for a convex set is achieved with the help of supporting halfspaces, it is achieved for a normal set by means of translates of the orthant. This gives the basis for an outer approximation method for solving (Q) similar to the method proposed in [8].

We now show some properties of $\theta(y)$ (as an increasing function) which suggest a branch and bound method for solving (Q).

Let us agree to call *polyblock* in $[0, b]$ any set P which is the union of a finite number of boxes $[0, y]$, $y \in T$, where $T \subset [0, b]$. The vectors in T are called the *vertices* of the polyblock $P = \cup_{y \in T} [0, y]$. A vertex y is said to be *improper* if it is dominated by some other vertex, i.e. if there exists $y' \in T \backslash \{y\}$ such that $[0, y] \subset [0, y']$. Of course, an improper vertex can be deleted without changing the polyblock.

PROPOSITION 7. *Let P be a polyblock covering $C \cap G$* (the feasible set of problem (MQ)) *and let V be the set of all proper vertices of P that belong to C. Then the polyblock with vertex set V still covers $C \cap G$ and an upper bound of $\theta(.)$ over $C \cap G$ is given by* $\max\{\theta(y) \mid y \in V\}$.

Proof. If $y \notin C$, i.e. if $\{x \in D \mid g(x) \leqslant y\} = \emptyset$, then for every $y' \in [0, y]$ we also have $\{x \in D \mid g(x) \leqslant y'\} = \emptyset$, i.e. $y' \notin C$. Therefore, $[0, y] \cap C = \emptyset$ for all $y \notin C$. The conclusion follows by noting that the maximum of $\theta(.)$ over any box $[0, y]$ is obviously $\theta(y)$. □

Now, with the notations of Proposition 7 suppose that $y \in V \backslash G$ and let $z = \pi(y)$. Since $y \geqslant a > 0$ (Proposition 2), we have $0 < z < y$. Let e^i be the ith unit vector of R^m_+.

PROPOSITION 8. *Let $y \in [0, b]$ and $0 < z < y$. Then $[0, y] \backslash K_z$ is a polyblock with vertices*

$$z^i = y - (y_i - z_i)e^i\,, \quad i = 1, \ldots, m\,. \tag{11}$$

Proof. Let $H_i = \{u \in R^m_+ \mid u_i > z_i\}$. Since $K_z = \cap_{i=1}^m H_i$ we have $[0, y] \backslash K_z = \cup_{i=1}^m ([0, y] \backslash H_i) = \cup_{i=1}^m \{u \in R^m_+ \mid u_i \leqslant z_i,\ u_j \leqslant y_j\ \forall j \neq i\} = \cup_{i=1}^m [0, z^i]$. □

COROLLARY 1. *The polyblock with vertex set* $(V\backslash\{y\}) \cup \{z^1, \ldots, z^m\}$ *still covers* $C \cap G$ *but does not contain* y.

Thus, giving any polyblock covering $C \cap G$, and a proper vertex $y \in C\backslash G$, we can construct a smaller polyblock still covering $C \cap G$ but excluding y.

4. Solution Method

For any given $\varepsilon \geqslant 0$, we wish to find an ε-optimal solution of (MQ), i.e. a feasible solution $\bar{y}$ of (MQ) such that

$$\theta(\bar{y}) \geqslant \max\{\theta(y) \mid y \in C \cap G\} - \varepsilon$$

By (4) we then have $\theta(\bar{y}) = \langle c, \bar{x}\rangle$ for some $\bar{x} \in D$ satisfying $g(\bar{x}) \leqslant \bar{y}$ and hence $\varphi(g(\bar{x})) \leqslant 1$, so $\bar{x}$ is a feasible solution of (Q) such that

$$\langle c, \bar{x}\rangle \geqslant \max\{\langle c, x\rangle \mid x \in D,\ \varphi(g(x)) \leqslant 1\} - \varepsilon\,,$$

i.e. $\bar{x}$ solves (Q) with tolerance ε.

Based on the properies discussed in the preceding section, a branch and bound method for solving (Q) with tolerance ε can be outlined as follows.

Start with the polyblock $P_1 = [0, b] \supset C \cap G$ with vertex set $T_1 = \{y^1\}$, $y^1 = b$, $T_1 = V_1$. If $y^1 = b \in G$, b solves (Q). Otherwise, we can compute $z^1 = \pi(y^1) \in G$. If $z^1 \in C$ let $\bar{y}^1 = z^1$, $\gamma_1 = \theta(\bar{y}^1)$. Otherwise, let $\gamma_1 = -M$. An upper bound for $\theta(.)$ over the feasible set is

$$\theta(y^1) = \max\{\theta(y) \mid y \in V_1\}\,.$$

At iteration k we have a polyblock $P_k = \cap_{y \in T_k} [0, y] \supset C \cap G$ with vertex set T_k. For each box $[0, y]$, $y \in T_k$, an upper bound of $\theta(.)$ over the feasible points in this box is $\theta(y)$. Also a 'current best value' γ_k is known, along with a 'current best solution' $\bar{y}^k \in G$ with $\theta(\bar{y}^k) = \gamma_k$ if $\gamma_k > -M$. Let V_k be the set that remains from T_k after removing the improper elements and all y such that $\theta(y) \leqslant \theta(\bar{y}^k)$ (including points $y \notin C$, i.e. points y with $\theta(y) = -M$). Choose

$$y^k \in \operatorname{argmax}\{\theta(y) \mid y \in V_k\}\,.$$

Compute $z^k = \pi(y^k)$, and the points $y^{k,i} = y^k - (y_i^k - z_i^k)e^i$, $i = 1, \ldots, m$. By Corollary 1, the polyblock P_{k+1} with vertex set $T_{k+1} = (V_k\backslash\{y^k\}) \cup \{y^{k,i} \mid i = 1, \ldots, m\}$ still covers $C \cap G$ but is smaller than P_k because $y^k \notin P_{k+1}$. Determine the new current best value γ_{k+1} and current best solution $\bar{y}^{k+1}$ and repeat the procedure with P_{k+1} in place of P_k.

Clearly the sequence $\theta(y^k)$ is decreasing, while $\theta(\bar{y}^k)$ is increasing. The procedure is stopped when $\theta(y^k) - \theta(\bar{y}^k) \leqslant \varepsilon$ (then $\bar{y}^k$ is an ε-optimal solution) or $V_k = \emptyset$ (then $\bar{y}^k$ is an exact optimal solution of (MQ) if $\gamma_k > -M$, or the problem is infeasible if $\gamma_k = -M$).

In a formal way, we can state:

ALGORITHM *Initialization*. Take a, b such that $0 < a_i \leqslant \min\{y_i \mid y \in g(D)\} \leqslant \max\{y_i \mid y \in g(D)\} \leqslant b_i$, $i = 1, \ldots, m$. If $\varphi(a) > 1$, the problem is infeasible. If $\varphi(b) \leqslant 1$, b is an optimal solution. If $\varphi(a) \leqslant 1 < \varphi(b)$, let $P_1 = [0, b]$, $y^1 = b$, $T_1 = V_1 = \{y^1\}$, $z^1 = \pi(y^1)$. If $z^1 \in C$, let $\bar{y}^1 = z^1$, $\gamma_1 = \theta(\bar{y}^1)$; if $z^1 \notin C$, let $\gamma_1 = -M$. Set $k = 1$.

Step 1. If $\theta(y^k) - \theta(\bar{y}^k) \leqslant \varepsilon$, terminate: $\bar{y}^k$ is an ε-optimal slution.

Step 2. Compute

$$y^{k,i} = y^k - (y_i^k - z_i^k)e^i \quad i = 1, \ldots, m \tag{12}$$

and set

$$T_{k+1} = (V_k \backslash \{y^k\}) \cup \{y^{k,1}, \ldots, y^{k,m}\}. \tag{13}$$

Step 3. For every new $y \in T_{k+1}$ solve the convex program

$$\max\{\langle c, x\rangle \mid x \in D,\ g(x) \leqslant y\} \tag{14}$$

to obtain $\theta(y)$. Let V_{k+1} be the set that remains from T_{k+1} after removing all improper vertices and all vertices y such that $\theta(y) \leqslant \theta(\bar{y}^k)$.

Step 4. If $V_{k+1} = \emptyset$ terminate: $\bar{y}^k$ is optimal if $\gamma_k > -M$, or the problem is infeasible if $\gamma_k = -M$.

Step 5. If $V_{k+1} \neq \emptyset$, compute

$$y^{k+1} \in \operatorname{argmax}\{\theta(y) \mid y \in V_{k+1}\} \tag{15}$$

$$\mu_{k+1} = \max\{\alpha \mid \varphi(\alpha y^{k+1}) \leqslant 1\}, \quad z^{k+1} = \mu_{k+1} y^{k+1}. \tag{16}$$

Let $\bar{y}^{k+1} = \operatorname{argmax}\{\theta(z^{k+1}),\ \theta(\bar{y}^k)\}$, $\gamma_{k+1} = \theta(\bar{y}^{k+1})$.

Set $k \leftarrow k + 1$ and return to Step 1.

THEOREM 1. *If the above Algorithm is infinite, each of the generated sequences* $\{y^k\}$ *and* $\{z^k\}$ *contains a subsequence converging to an optimal solution.*

Proof. We first show that there is a sequence $\{k_q\} \subset \{1, 2, \ldots\}$ such that

$$\lim_{q \to +\infty} (y^{k_q} - z^{k_q}) = 0. \tag{17}$$

Since $z^k = \lambda_k y^k$, i.e. $y^k - z^k = (1 - \lambda_k) y^k$ with $0 < \lambda_k \leqslant 1$, this amounts to saying that $1 - \lambda_{k_q} \to 0$ for some sequence $\{k_q\} \subset \{1, 2, \ldots\}$. Suppose this is not true, so there is $\eta > 0$ such that $1 - \lambda_k \geqslant \eta > 0$ $\forall k$. By Proposition 2, $y^k \geqslant a > 0$ $\forall k$ because $y^k \in C$ $\forall k$. Hence

$$y_i^k - z_i^k \geqslant \eta y_i^k \geqslant \eta a_i > 0 \quad \forall i = 1, \ldots, m, \quad \forall k = 1, 2, \ldots. \tag{18}$$

Now the branching process can be represented by a tree rooted at $y^1 = b$. A node y^k of this tree is connected to its successors $y^{k,i}$, $i = 1, \ldots, m$ defined by (12). Consider

any path on this tree passing through $y^{k_1}, y^{k_2}, \ldots, y^{k_p}$, where $y^{k_1} = b$ and for $q = 1, \ldots, p-1$, $y^{k_{q+1}}$ is a successor of y^{k_q}, i.e.

$$y^{k_{q+1}} = y^{k_q} - (y^{k_q}_{i_q} - z^{k_q}_{i_q})e^{i_q}$$

for $i_q \in \{1, \ldots, m\}$. By virtue of (18) this yields, for $q = 1, 2, \ldots, p-1$:

$$y^{k_q}_i - y^{k_{q+1}}_i \begin{cases} \geqslant \eta a_i & i = i_q \\ = 0 & i \neq i_q \end{cases} \tag{19}$$

Let $i_q = 1$ for N_1 values of q. Then by adding all the inequalities (19) for $i = 1$ we have

$$b_1 - y^{k_p}_1 = \sum_{q=1}^{p-1} (y^{k_q}_1 - y^{k_{q+1}}_1) \geqslant N_1 \eta a_1 .$$

Since $b_1 - y^{k_p}_1 \leqslant b_1$ this implies $N_1 \eta a_1 \leqslant b_1$, i.e. $N_1 \leqslant b_1/(\eta a_1)$. Analogously, $N_i \leqslant b_i/(\eta a_i)$, $i = 2, \ldots, m$. Hence $p = 1 + N_1 + N_2 + \cdots + N_m \leqslant \sum_{i=1}^m b_i/(\eta a_i)$, which leads to a contradiction for p sufficiently large. We have thus proved (17). Since the sequence $\{y^{k_q}\} \subset [0, b]$ is bounded, by passing to a subsequence if necessary, one can assume that $y^{k_q} \to \tilde{y}$. From (17) we then have

$$\tilde{y} = \lim_{q \to +\infty} y^{k_q} = \lim_{q \to +\infty} z^{k_q} . \tag{20}$$

Since $y^q \in C$ while $z^q \in G$ for every q, this yields $\tilde{y} \in C \cap G$. Furthermore, since $\theta(y^q) \geqslant \bar{\gamma} := \max\{\theta(y) \mid y \in C \cap G\}$, it follows from the upper semicontinuity of $\theta(y)$ that $\theta(\tilde{y}) \geqslant \bar{\gamma}$, and hence, $\theta(\tilde{y}) = \bar{\gamma}$. Thus $\tilde{y}$ is an optimal solution of (MQ). This completes the proof of the Proposition because by (20) $\tilde{y}$ is also an accumulation point of the sequence $\{z^k\}$. □

5. Discussion

(1) Each iteration of the above Algorithm involves solving an univariate equation $\varphi(\alpha y^{k+1}) = 1$ (see (16)) and solving m convex programs (14) (for finding m new points $y^{k,i}$, $i = 1, \ldots, m$). To solve univariate equations like (16) the best method is perhaps Bolzano's bisection (see [8]). As for the convex programs (14), since they differ only by the vector y, reoptimization techniques could be used to save computational efforts and time, instead of starting from scratch for solving each of them. Especially when these programs are linear (which is the case if D is polyhedral and $g(x)$ is affine), reoptimization techniques could be very efficient.

(2) To delete the improper elements of T_{k+1} (see (13)) note that, since V_k has no improper element and $y^{k,i} \leqslant z^k \ \forall i$, only some of the new elements $y^{k,i}$ can be improper for T_{k+1}. Therefore, all improper elements can be deleted in the following way: for every $y \in V_k$ check whether $y \geqslant z^k$, and whether $y_i < y^k_i$ for a unique $i \in \{1, \ldots, m\}$; if yes, then delete y^{ki}. This procedure requires at most $2m(|V_k| - 1)$ comparisons.

(3) Even though at most $m-1$ new points may be added to V_k at each iteration, storage problems may arise because of the growth of V_k. To overcome the difficulty in this case, it is advisable to make a **restart** every time V_k is going to reach a critical size. Specifically, if L is the maximum size allowed for V_k, then Step 1 is modified as follows:

Step 1. If $\theta(y^k) - \theta(\bar{y}^k) \leq \varepsilon$, terminate: $\bar{y}^k$ is an ε-optimal solutions. If $|V_k| \leq L$, go to Step 2. Otherwise, reset $V_k = \{b\}$, (i.e. $P_k = [0, b]$), $y^k = b$, and go to Step 2.

In this way, the algorithm is restarted from the polyblock $[0, b] \backslash (z^k, b]$.

(4) The above Algorithm can also be viewed as a special outer approximation procedure, in which a nested sequence of polyblocks is constructed to approximate the feasible set more and more closely. At every iteration of this procedure a set T_k (vertex set of the current polyblock) has to be computed. However, while in standard polyhedral outer approximation (OA) procedures, the vertex set of the current outer approximating polytope exponentially grows in size and becomes more and more difficult to compute accurately, the set T_k in the above Algorithm increases at most by m at each iteration and its new elements are computed by extremely simple formulas (see (12)). Furthermore, the function $\varphi(y)$ is assumed only to be increasing. In view of these positive features, the Algorithm should be able to handle problems of much larger size and of a larger class than standard OA procedures. In particular, as applied to convex programs with additional multiplicative constraint, it is expected to perform better than existing algorithms.

6. Illustrative Examples

To illustrate how the algorithm works we present some numerical examples. For the sake of simplicity these examples as well as the test problems used in the computational experiments to be discussed in the next section are taken in the form

$$\max\left\{\langle c, x\rangle \,|\, Mx \leq q,\ x \geq 0,\ \prod_{i=1}^{m} g_i(x) \leq 1\right\} \tag{21}$$

where $M \in R^{h\times n}$, $q \in R^h$, and $g_1(x), \ldots, g_m(x)$ are affine functions, i.e. $g(x) = Ex + d$ with $E \in R^{m\times n}$, $d \in R^m$.

EXAMPLE 1. Solve the problem:

$$\begin{aligned} \max\quad & -x_1 + x_2 \\ \text{s.t.}\quad & x_1 + x_2 \leq 4,\ x_1 \leq 2,\ x_1 \geq 0,\ x_2 \geq 0, \\ & (0.2x_1 - 0.1x_2 + 1.2)(0.1x_1 + 0.2x_2 + 0.3) \leq 1 \end{aligned}$$

For initialization we take

$$a = (0.800, 0.300), \quad b = (1.600, 1.100), \quad V_1 = \{(1.600, 1.100)\},$$

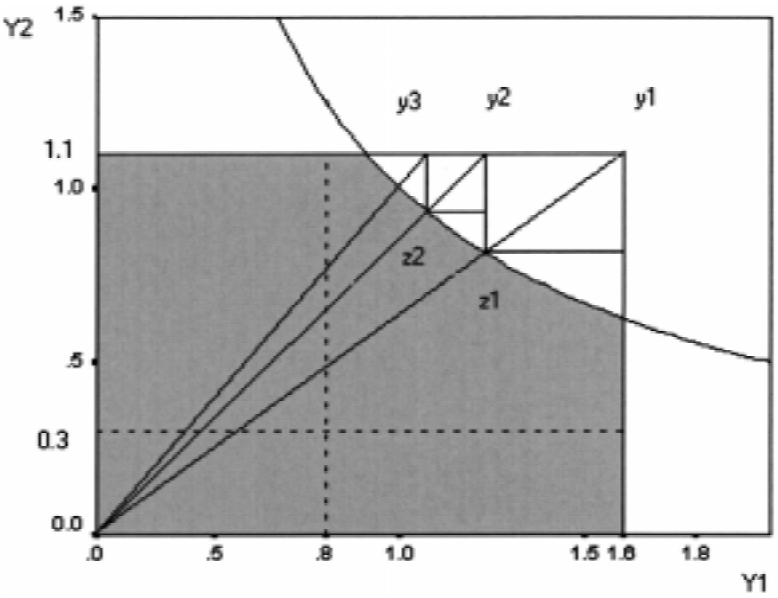

Figure 1. Polyblock Approximation Algorithm.

Table 1

Iter	y^k	$z^k = \pi(y^k)$	CBS $\bar{y}^k$	$\theta(y^k) - \theta(\bar{y}^k)$	New vertices
1	(1.600, 1.100)	(1.206, 0.829)	(1.206, 0.829)	1.354	(1.206, 1.100), (1.600, 0.829)
2	(1.206, 1.100)	(1.047, 0.955)	(1.047, 0.955)	0.725	(1.047, 1.100) (1.206, 0.955)
3	(1.047, 1.100)	(0.976, 1.025)	(0.976, 1.025)	0.375	(0.976, 1.100), (1.047, 1.025)
4	(0.976, 1.100)	(0.942, 1.062)	(0.942, 1.062)	0.191	(0.942, 1.100), (0.976, 1.062)
5	(0.942, 1.100)	(0.925, 1.081)	(0.925, 1.081)	0.096	(0.925, 1.100), (0.942, 1.081)
6	(0.925, 1.100)	(0.917, 1.090)	(0.917, 1.090)	0.048	(0.917, 1.100), (0.925, 1.090)
7	(0.917, 1.100)	(0.913, 1.095)	(0.913, 1.095)	0.024	(0.913, 1.100), (0.917, 1.095)
8	(0.913, 1.100)	(0.911, 1.098)	(0.911, 1.098)	0.012	(0.911, 1.100), (0.913, 1.098)
9	(0.911, 1.100)	(0.910, 1.099)	(0.910, 1.098)	0.006	(0.910, 1.100), (0.911, 1.099)
10	(0.910, 1.100)	(0.910, 1.099)	(0.910, 1.099)	0.003	(0.910, 1.100), (0.910, 1.099)
11	(0.910, 1.100)	(0.909, 1.100)	(0.909, 1.100)	0.002	(0.909, 1.100), (0.910, 1.100)
12	(0.909, 1.100)	(0.909, 1.100)	(0.909, 1.100)	0.001	

so $y^1 = (1.600, 1.100)$. The line through 0 and y^1 intersects the upper boundary of the feasible region at the point $z^1 = (1.206, 0.829)$ and we have $\theta(y^1) - \theta(z^1) = 1.354$. The vertices of the rectangle $[z^1, y^1]$ which are adjacent to y^1 are (1.206, 1.100) and (1.600, 0.829). Hence we have $V_2 = \{(1.206, 1.100), (1.600, 0.829)\}$. In the next step, we compute $y^2 = \text{argmax}\{\theta(y) \mid y \in V_2\}$. Therefore, $y^2 = (1.206, 1.100)$ and so on. With a tolerance $\epsilon = 0.001$ the algorithm terminates after 12 iterations, yielding $y_{\text{opt}} = (0.909, 1.100)$ as an optimal solution of problem (MQ), and $x_{\text{opt}} = (0.000, 3.999)$ as an optimal solution of problem (Q), respectively. The computation results for all iterations are summarized in Table 1.

EXAMPLE 2. Consider problem (21) with the following data:

Vector $c \in R^{15}$:

$c = (0.700, 0.500, -0.300, 0.400, 0.300, -0.900, 0.400, -0.100,$
$0.600, 0.600, -0.100, 0.000, 0.200, 0.400, -0.200)$.

Matrix M (10×15):

−6.30	3.30	4.20	6.50	9.40	−1.80	8.00	5.40
	−2.20	6.30	1.80	−2.10	−9.90	−7.90	−4.00
9.50	−8.10	−1.30	6.80	−6.20	−4.60	−3.60	−9.60
	0.00	5.30	4.10	−3.10	0.00	−5.70	4.30
3.60	−2.70	−3.70	7.90	5.10	−7.80	−5.90	2.50
	8.40	2.50	2.10	−8.60	4.50	−2.10	9.30
−0.70	6.80	−3.80	2.10	−3.20	3.70	7.60	4.70
	−9.50	9.00	−4.10	−9.10	7.30	9.10	6.70
−2.00	−2.70	−5.00	6.30	−8.90	2.10	2.60	−4.80
	2.60	3.90	8.10	−3.30	4.30	−7.50	−5.10
−3.90	7.60	0.80	7.40	−3.80	−4.00	2.60	2.40
	−6.30	1.80	5.00	−0.30	6.20	0.70	−7.40
0.40	−8.30	9.90	5.40	−8.00	7.50	1.40	7.80
	−4.30	9.50	−0.10	−0.60	4.10	−3.90	1.70
−0.80	1.20	5.10	−7.90	6.20	−7.00	−2.60	−5.20
	5.20	2.50	−3.20	−6.60	7.60	−6.70	7.50
1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
	1.00	1.00	1.00	1.00	1.00	1.00	1.00
1.30	6.20	−5.10	−5.20	1.50	1.00	−4.20	3.20
	2.20	−3.40	3.80	1.40	−3.10	8.60	−6.20

Vector $q \in R^{10}$:

$q = (2.25, 8.20, 10.44, 15.37, 5.23, 12.19, 23.36, 1.70, 6.50, -5.82)$.

Matrix E (5×15):

0.06	0.06	−0.11	0.16	0.06	0.19	−0.18	0.16
	−0.09	0.11	−0.12	−0.22	0.02	−0.02	0.22
0.21	0.25	−0.23	0.22	0.09	−0.01	0.19	−0.19
	0.17	0.01	0.07	0.14	−0.25	0.08	0.03
−0.12	−0.01	−0.05	−0.12	0.14	−0.24	0.09	−0.09
	−0.18	0.21	0.21	0.08	0.02	0.21	0.21
−0.24	−0.18	0.02	0.00	0.09	−0.18	0.03	0.16
	−0.12	−0.13	0.17	−0.01	0.06	0.19	−0.16
−0.25	0.04	−0.19	0.21	0.10	0.00	−0.02	0.07
	−0.04	0.00	0.22	0.13	0.18	0.22	0.17

Vector $d \in R^5$:
$d = (1.19,\ 0.59,\ 0.99,\ 1.19,\ 0.96)$.

To have an idea of the effect of using reoptimization techniques for solving the subproblems (14) we solved the above example by two different implementation versions of the Algorithms, one with incorporated reoptimization techniques for linear programs and the other without. For a tolerance of 0.000001 and using a PC computer Pentium II, the optimal value of 3.364 was found after 59 iterations, requiring a total amount of time about 1.6 seconds by the first version, and 6.1 seconds by the second version. This shows that using reoptimization techniques may substantially reduce the computational time. In fact, an important part of the computational burden is due to the solution of these subproblems (m subproblems in each iteration). Note, however, that although theoretically, the two procedures (with or without reoptimization techniques) must provide the same optimal solution and require the same number of iterations, in practice, the intermediary results may not always coincide. These possible differences are due to the fact that the computing errors may not be quite the same when solving a linear program from scratch and when reoptimizing it from an optimal solution of a related linear program. For instance, for the present example the two implementation versions of the Algorithm gave two different optimal solutions

$$y_{\text{opt}} = (1.954, 1.681, 0.876, 0.768, 0.453)$$

in the first version (with incorporated reoptimization techniques) and

$$y_{\text{opt}} = (1.383, 1.686, 0.905, 1.046, 0.453)$$

in the other version.

7. Computational Experiments

The algorithm described in Section 5 for problem (21) was coded in Pascal language and tested on a PC computer Pentium II.

More than 200 tested problems were randomly generated, with different values of n (number of variables), h (numbers of rows of M), and m (number of functions g_i). For each given size 12 problems were generated. The experiments were carried out on problems with m up to 10, and a tolerance of $\varepsilon = 0.01$ for the relative error of the optimal value. The restart version of the algorithm was implemented, with $L = 500$ for problems with $m \leqslant 4$ and $L = 1900$ for problems with $m > 4$ (L being the maximum size allowed for $|V_k|$, see Section 5). The procedure works basically in R^m space (the number of vertices to be stored increases by $m - 1$ at each iteration). The computational burden depends essentially on the value of m, and much less on the values of h and n which only determine the sizes of the linear programs to be solved for computing $\theta(y^k)$.

The results of the experiments are reported in Table 2. Column Iter indicates the maximum number of iterations, Rest: the maximum number of restarts and t: the maximum computational time (in seconds), for the 12 tested problems of each size.

These results show that the method is quite practical even on conventional PC's, at least for solving problems with $m \leqslant 10$. A feature worth noticing is that even though the number of iterations may seem unusually large in certain cases, the total computational time is quite reasonable because each iteration involves few and simple computations, and requires on the average no more than 0.7 seconds even for problems with $m = 10$.

We emphasize the role of reoptimization and restarting in the experiments performed. As was mentioned in Section 6, the use of reoptimization technique for solving the subproblems (14) enabled one to significantly reduce the computation

Table 2

Prob.	h	n	m	Iter	Rest	t (in seconds)
1–12	10	15	2	23	0	0.16
13–24	15	20	2	16	0	0.39
25–36	10	15	3	94	0	0.88
37–48	15	20	3	403	1	12.90
49–60	10	15	4	483	2	23.23
61–72	15	20	4	587	3	36.58
73–84	10	15	5	1110	2	489.66
85–96	15	20	5	608	1	338.24
97–108	10	15	6	1334	3	684.64
109–120	15	20	6	884	2	485.05
121–132	10	15	7	1202	3	621.37
133–144	15	20	7	1064	3	511.83
145–156	10	15	8	1067	4	617.15
157–168	15	20	8	2006	7	1250.10
169–180	10	15	9	1252	5	817.51
181–192	15	20	9	826	3	634.24
193–204	10	15	10	1538	7	938.67
205–216	15	20	10	1890	8	1307.40

time. Also, many tested problems with $m = 10$ could not have been solved successfully (premature termination would have been necessary for these cases), were the algorithm implemented without restart.

8. Conclusion

We have presented a new approach to a class of convex programs with an additional monotonic constraint. This approach is based on the approximation of the level sets of increasing functions by means of sets of a particular kind called 'polyblocks'. Results in this and the paper [8] have been further generalized in [10] to apply to a broad class of nonconvex optimization problems described by means of *increasing functions and differences of increasing functions*.

References

1. Horst, R. and Tuy, H. (1996), *Global Optimization: Deterministic Approaches*, Springer, 3rd edition.
2. Konno, H., Thach, P.T. and Tuy, H. (1997), *Optimization on Low Rank Nonconvex Structures*, Kluwer.
3. Kuno, T., Konno, H. and Yamomoto, Y. (1992), A parametric successive underestimation method for convex programming problems with an additional convex multiplicative constraint, *Journal of the Operations Research Society of Japan* 35: 290–299.
4. Kuno, T., Yajima, Y., Yamamoto, Y. and Konno, H. (1994), Convex programs with an additional constraint on the product of several convex functions, *European Journal of Operational Research* 77: 314–324.
5. Kuno, T. and Yamamoto, Y. (1998), A finite algorithm for globally optimizing a class of rank-two reverse convex constraints, *Journal of Global Optimization* 12: 247–265.
6. Pferschy, U. and Tuy, H. (1994), Linear programs with an additional rank two reverse convex constraint, *Journal of Global Optimization* 4: 441–454.
7. Thach, P.T., Burkard, R. and Oettli, W. (1991), Mathematical programs with a two dimensional reverse convex constraint, *Journal of Global Optimization* 1: 145–154.
8. Rubinov, A., Tuy, H. and Mays, H. (1998), *Algorithm for a Monotonic Global Optimization Problem*. Preprint, SIMS, University of Ballarat, Australia. To appear in *Optimization*.
9. Tuy, H. (1998), *Convex Analysis and Global Optimization*, Kluwer.
10. Tuy, H. (1999), *Monotonic Optimization: Problems and Solution Approaches*. Preprint, Institute of Mathematics, Hanoi. To appear in *SIAM Journal on Optimization*.
11. Tuy, H. and Nghia, N.D. Decomposition algorithms for reverse convex programs. To appear in *Vietnam Journal of Mathematics* (Springer-Verlag).