Vietnam Journal of Mathematics **30**:4 (2002) 373-393

Vietnam Journal
of
MATHEMATICS

A Monotonicity Based Approach to Nonconvex Quadratic Minimization

Nguyen Thi Hoai Phuong and Hoang Tuy

Institute of Mathematics, P. O. Box 631, BoHo, Hanoi, Vietnam

Received July 20, 2001

Abstract. A new unified approach is proposed for minimizing a general (convex or concave, or indefinite) quadratic function under linear constraints. Unlike previously known approaches based on exploiting convexity and related properties, the proposed approach reformulates the problem as a generalized multiplicative program and solves it as a monotonic optimization problem of m variables, where m is the rank of the quadratic function. For the case $m = 1$ a finite algorithm can then be derived which improves upon the earlier known parametric algorithm. Numerical examples include, among others, a non trivial instance of the maximum clique problem.

1. Introduction

In this paper we are concerned with the general linearly constrained quadratic programming problem

$$\min\{\langle c, x\rangle + \langle x, Qx\rangle |\ x \in D\} \tag{QP}$$

where $D \subset R^n_+$ is a polyhedron, Q is a $n \times n$ matrix and $c \in R^n$. This problem is closely related to the "generalized multiplicative programming" problem

$$\min\{f_0(x) + \sum_{i=1}^{m} f_i(x)g_i(x) |\ x \in D\} \tag{MP}$$

where $f_0(x), f_i(x), g_i(x), i = 1, \ldots, m$, are affine functions. Actually, the two problems are equivalent, though this relationship has not been always explicitly noticed in the literature. In fact, (MP) is obviously a special case of (QP), while the converse readily follows from the relation $\langle x, Qx\rangle = \sum_{i=1}^{n} x_i(\sum_{j=1}^{n} q_{ij}x_j)$. On the other hand, any (QP) can be given the form (MP) with $m =$rank(Q) (see e.g. [16], Corollary 8.1). This relationship between (QP) and (MP) also shows that a problem (MP) with m arbitrary can always be reduced to the case $m \leq n$ where m is the rank of the equivalent quadratic objective function.

Due to their theoretical and practical interest, problems (QP) and (MP) have been the object of intensive research during the last four decades. It is well known that when Q is semidefinite positive (QP) is solvable in polynomial time but in other cases it is NP-hard. It has also been proved recently that (MP) is NP-hard, even for $m = 1$ and $f_0(x) \equiv 0$ [10]. In spite of that, both problems have received much attention since the appearance of global optimization methods ([3, 16]).

Specifically, (QP) with a semidefinite negative matrix Q is one of the most popular problems of concave minimization while the problem with a semidefinite matrix Q is a typical problem of d.c. optimization. We refer the interested reader to the books on global optimization (see e.g. [16]) for a detailed discussion on specific concave minimization and d.c. optimization methods for quadratic programming.

A special variant of (MP) when $m = 1$, namely the problem

$$\min\{f_0(x) + f_1(x)g_1(x)|\ x \in D\}, \tag{P_1}$$

was first studied by Konno and Kuno [5], Konno, Yajima and Matsui [8]. These authors assumed, however, that D is bounded (see also [6], where a method is given for solving an even more general problem than (MP), with $D = \{x \in R^n|\ h(x) \leq 0\}$ and all f_i, g_i, h being convex positive-valued on D). Subsequently, Schaible and Sodini [15] removed this boundedness assumption and proposed a finite method for solving (P_1). The latter method is similar to the previous method of Konno-Kuno in that both reduce the problem to a parametric linear program in which the objective function as well as the right-hand side of the constraints linearly depend upon a scalar parameter. The problem is then solved, in both methods, by means of a finite number of primal and/or dual simplex iterations performed on the parametric linear program. However, using different optimality conditions the two methods generate different sequences of intermediate solutions, as was shown on a numerical example discussed in [15]. Although the method in [15] does not assume boundedness of D, it does use a weaker assumption, namely that at least one of the two functions $f_1(x),\ g_1(x)$ achieves a minimum on D. The weakness of the two described methods is that the parametric approach that underlies each of them is efficient when the parameter is scalar ($m = 1$) but cannot be easily extended to the general case when the parameter is multidimensional ($m \geq 2$). Partly because of these difficulties problems (MP) with $m \geq 2$ have been less investigated (see, however,[7 - 9]). Aside from the above cited papers we should also mention the recent work of Nghia [12] where the monotonic approach of Tuy [18] has been successfully implemented to solve problems (MP) with $m = 1$.

The aim of the present paper is to develop a new unified approach to the problem (MP) which should be more efficient than the above mentioned methods when m is significantly smaller than n. It turns out that the optimal value of the parametric linear program used in these methods is an increasing function of the parameter, in the sense that it increases when every component of this parameter increases. Using this fact, the problem can be reformulated and solved as a monotonic optimization problem in R^m, of the type defined and studied in [18, 19]. Although the extension of this approach to convex multiplicative

programming should present no particular difficulty, we only briefly mention this possibility, reserving a more detailed study for a subsequent paper.

After the Introduction, in Sec. 2, we reformulate (MP) as a monotonic optimization problem. Next in Sec. 3, we discuss the computation of a lower bound for the objective function values over a hyperrectangle in R^m. Based on these results, in Sec. 4 a branch and bound method is proposed for solving (MP), in which bounds are obtained by solving linear relaxations of the corresponding subproblems. Sec. 5 shows how this method can be made finite by using a suitable subdivision rule, while Sec. 6 examines improvements to enhance efficiency of the method when dealing with higher rank problems (e.g. problems with $m \geq 10$). Finally, to illustrate how the method works in practice, Sec. 7 presents some nontrivial numerical examples including an instance of the maximum clique problem. The latter example also shows the importance of reformulation for improving the bound in order to speed up the process of verifying optimality of the best solution found.

Unlike the existing methods of quadratic programming based on convex and difference of convex (d.c.) properties, our approach systematically exploits monotonicity properties of the functions involved. Hopefully a new efficient tool is thus provided for handling a class of difficult, yet important and widely encountered, problems, especially when the rank is significantly smaller than the dimension.

2. Reduction to Monotonic Optimization

Throughout the following we shall make the blanket assumption

$$-\infty < \inf_{x \in D} \{g_i(x)\} < +\infty, \quad \forall i = 1, \ldots, m. \tag{1}$$

This condition is weaker than requiring that D be nonempty and compact. Note that it is assumed also in [15], though implicitly, for the special case $m = 1$.

Under assumption (1) the problem can always be reduced to the case when

$$g_i(x) > 0, \quad \forall x \in D, \quad \forall i = 1, \ldots, m. \tag{2}$$

Indeed, with $\beta_i = \min\{g_i(x) |\ x \in D\} - 1$, and $\tilde{g}_i(x) = g_i(x) - \beta_i$ we can write

$$f_i(x)g_i(x) = f_i(x)(\tilde{g}_i(x) + \beta_i) = f_i(x)\tilde{g}_i(x) + \beta_i f_i(x).$$

Therefore, setting $\tilde{f}_0(x) = f_0(x) + \sum_{i=1}^m \beta_i f_i(x)$, we reduce the problem to the equivalent problem

$$\min\{\tilde{f}_0(x) + \sum_{i=1}^m f_i(x)\tilde{g}_i(x) |\ x \in D\}$$

where all the affine functions $\tilde{g}_i(x),\ i = 0, 1, \ldots, m$ are positive-valued on D.

Proposition 1. *If* $\inf_{x \in D} f_i(x) > -\infty\ \forall i = 0, 1, \ldots, m$, *then the problem* (MP) *has a finite optimal solution. Otherwise* (MP) *is unbounded.*

Proof. Since $g_i(x) > 0$, if $a_i := \inf_{x\in D} f_i(x) > -\infty,\ \forall i = 0, 1, \dots, m$, then obviously $\inf_{x\in D}[f_0(x) + \sum_{i=1}^m f_i(x)g_i(x)] \geq \inf_{x\in D}[f_0(x) + \sum_{i=1}^m a_i g_i(x)] \geq \inf_{x\in D} f_0(x) + \sum_{i=1}^m [\inf_{x\in D} a_i g_i(x)] > -\infty$ (we used assumption (1) in the last inequality). Since D is closed and $f_0(x) + \sum_{i=1}^m f_i(x)g_i(x)$ is continuous, it follows that (MP) has a finite optimal solution. The second assertion is obvious. ■

Note in passing that a problem in [15] (Example 5.3 in [15]) is unbounded just because it fails to satisfy the just stated conditions. In view of the above Proposition, without loss of generality, in the following we shall assume, aside from (1), that

$$a_i = \inf_{x\in D} f_i(x) > -\infty, \quad \forall i = 0, 1, \dots, m. \tag{3}$$

Setting $b_i = \sup_{x\in D} f_i(x) \in R \cup \{+\infty\},\ i = 1, \dots, m$, we thus have

$$a \leq f(x) \leq b, \quad \forall x \in D. \tag{4}$$

Now, for brevity let us write $f(x)=(f_1(x), \dots, f_m(x)), g(x) = (g_1(x), \dots, g_m(x))$, $\langle f(x), g(x)\rangle = \sum_{i=1}^m f_i(x)g_i(x)$. Also it will be convenient to refer to a set of the form $[a, b] = \{y \in R^m |\ a \leq y \leq b\}$ as the rectangle $[a, b]$ (some or all components of b may be $+\infty$.) Assuming (2), (3), define

$$\begin{aligned} H &= \{y \in R^m |\ \ y \geq f(x),\ \ x \in D\} \\ \Phi(y) &= \inf\{f_0(x) + \langle f(x), g(x)\rangle |\ x \in D,\ y \leq f(x)\} \end{aligned}$$

with the usual convention $\inf \emptyset = +\infty$. Note that since (MP) has a finite optimal solution under the stated assumptions, we have $\Phi(y) > -\infty\ \forall y$.

Following [18] a function $\Phi(y)$ defined on $[a, b]$ is said to be *increasing* if $a \leq y \leq y' \leq b$ implies that $\Phi(y) \leq \Phi(y')$. A set $H \subset R^m$ is said to be *reverse normal* if $y' \geq y \in H$ implies that $y' \in H$.

Proposition 2. *$\Phi(y)$ is an increasing function on $[a, b] \subset R^m$, and H is a reverse normal set in R^m.*

Proof. Let $D_y = \{x \in D |\ y \leq f(x)\}$. If $a \leq y \leq y' \leq b$ then $D_{y'} \subset D_y$, so $\Phi(y) = \min\{f_0(x) + \langle f(x), g(x)\rangle |\ x \in D_y\} \leq \min\{f_0(x) + \langle f(x), g(x)\rangle |\ x \in D_{y'}\} = \Phi(y')$. That the set H is reverse normal is obvious. ■

Proposition 3. *The problem* (MP) *is equivalent to*

$$\min\{\Phi(y) |\ \ y \in H,\ a \leq y \leq b\} \tag{G}$$

Proof. First observe that, since $\Phi(y)$ is increasing, if $y \in [a, b]$ is feasible to (G), i.e. $y \geq f(x)$ for some $x \in D$ then $y' = f(x) \leq y$ will still be feasible to (G), while $\Phi(y') \leq \Phi(y)$. Therefore the minimum of $\Phi(y)$ is attained on the set $\{y \in [a, b] \mid y = f(x),\ x \in D\}$ and consequently, (G) is equivalent to

$$\min\{\Phi(y) | y = f(x), \quad x \in D,\ \ a \leq y \leq b\}. \tag{G'}$$

Now, if $\bar{x} \in D$ solves (MP) then $\bar{y} = f(\bar{x})$ is feasible to (G'), and $\Phi(\bar{y}) \leq f_0(\bar{x}) + \langle f(\bar{x}), g(\bar{x})\rangle \leq f_0(x) + \langle f(x), g(x)\rangle\ \forall x \in D$, hence $\Phi(\bar{y}) \leq \Phi(y)$ for all

y such that $y = f(x),\ x \in D$, i.e. $\bar{y}$ solves (G'). Conversely, if $\bar{y}$ solves (G') , then $\Phi(\bar{y}) \leq \Phi(f(x))\ \forall x \in D$ (since $y = f(x)$ is feasible to (G')), hence, denoting by $\bar{x} \in D$ the optimal solution of the problem defining $\Phi(\bar{y})$, we have $\Phi(\bar{y}) = f_0(\bar{x}) + \langle f(\bar{x}), g(\bar{x})\rangle \leq f_0(x) + \langle f(x), g(x)\rangle\ \forall x \in D$, i.e. $\bar{x}$ solves (MP).■

Thus, solving (MP) amounts to solving (G) which is a monotonic optimization problem of the class considered in [18] or [19]. Of course, computing the exact value of $\Phi(y)$ for a given y is as difficult as solving the problem (MP) itself. However, it is not hard to find an easily computable underestimator of $\Phi(y)$ over any subrectangle $M = [p, q] \subset [a, b]$. Specifically, if we define $D_{[p,q]} = \{x \in D |\ p \leq f(x) \leq q\}$ then such an underestimator is provided by the function

$$\Psi_M(y) = \min\{f_0(x) + \langle y, g(x)\rangle |\ y \leq f(x),\ x \in D_{[p,q]}\}. \tag{5}$$

Like $\Phi(y)$ this function satisfies $\Psi_M(y) > -\infty\ \forall y$ and is increasing on $[p, q]$, but since (5) is a linear program, the value $\Psi_M(y)$ is easily computable. Using the underestimator $\Psi_M(y)$ one can derive lower bounds for $\Phi(y)$ over sets of the form $H \cap M$. It turns out that these bounds can be proved to be consistent with suitably defined subdivision rules, and can thus be incorporated into a convergent branch and bound method for solving (G) that works essentially in R^m (rather than R^n). In the next section we proceed to discuss this method in detail.

Remark 1. An alternative way of converting (MP) into a monotonic optimization problem is the following. Upon preliminary transformations similar to the above ones one can assume that $f_0(x), f_i(x), g_i(x)$ are nonnegative on D. Then the problem can be rewritten as

$$\min\{y_0 + \sum_{i=1}^{m} y_i z_i |\ y_0 \geq f_0(x),\ y_i \geq f_i(x),\ z_i \geq g_i(x)\ (i = 1, \ldots, m),\ x \in D\}.$$

Since the objective function $(y_0, y, z) \in R_+ \times R_+^m \times R_+^m \mapsto y_0 + \sum_{i=1}^m y_i z_i$ is increasing while the feasible set $C = \{(y_0, y, z) \in R_+ \times R_+^m \times R_+^m |\ y_0 \geq f_0(x), y \geq f(x), z \geq g(x),\ x \in D\}$ is reverse normal, this is a monotonic optimization problem in R_+^{2m+1}. This approach has been implemented [12] to solve problems with $m = 1$ by the reverse polyblock outer approximation procedure described in [18]. Although computational results presented in [12] show that this approach is quite practical for problems with $m = 1$ and up to 100 variables, its attractiveness sharply decreases as m increases due to the large dimension $(2m + 1)$ of the transformed problem.

3. Bounding Procedure

Given any subrectangle $M = [p, q] \subset [a, b]$, the problem (G) restricted to this subrectangle can be written as

$$\min\{\Phi(y)|\ y \geq f(x), \quad x \in D_{[p,q]}\}. \tag{G_M}$$

By Proposition 3 this subproblem is equivalent to

$$\min\{f_0(x) + \langle f(x), g(x)\rangle|\ \ x \in D_{[p,q]}\} \tag{P_M}$$

Proposition 4.

(i) *A lower bound for the optimal value of* (G_M) *is provided by*

$$\beta(M) := \min\{\Psi_M(y)|\ y \geq f(x),\ \ x \in D_{[p,q]}\} \tag{RG_M}$$

(ii) *Let* y^M *be an optimal solution of* (RG_M) *and* $x^M \in D$ *be an optimal solution of the linear program defining* $\Psi_M(y^M)$, *so that*

$$\beta(M) = \Psi_M(y^M), \quad \Psi_M(y^M) = f_0(x^M) + \langle y^M, g(x^M)\rangle.$$

Then an upper bound of the optimal value of (G_M) *is*

$$\alpha(M) = f_0(x^M) + \langle f(x^M), g(x^M)\rangle.$$

(iii) *If* $y^M = f(x^M)$ *then* y^M *is an optimal solution of* (G_M) *and* $\beta(M)$ *equals the optimal value of* (G_M):

$$\beta(M) = \alpha(M) = f_0(x^M) + \langle y^M, g(x^M)\rangle.$$

while x^M *is an optimal solution of* (P_M).

Proof. (i) and (iii) are obvious because $\Psi_M(y)$ is a underestimator of $\Phi(y)$ over M, while x^M is a feasible solution of (P_M) which is equivalent to (G_M).

For the proof of (ii) observe that if $f(x^M) = y^M$ then x^M is feasible to the problem defining $\Phi(y^M)$ so that $\Psi_M(y^M) = \Phi(y^M)$ and therefore,

$$\beta(M) = f_0(x^M) + \langle f(x^M), g(x^M)\rangle = \Psi_M(y^M) = \Phi(y^M)$$

because (RG_M) is a relaxation of (G_M). ∎

To obtain $\beta(M)$ we solve the monotonic optimization problem (RG_M) by using the reverse polyblock outer approximation (ROA) described in [18]. The exact optimal value of (G_M) is contained in the interval $[\beta(M), \alpha(M)]$ and we have

$$\alpha(M) - \beta(M) = \langle f(x^M) - y^M,\ g(x^M)\rangle \leq \|f(x^M) - y^M\|.\|g(x^M)\| \tag{6}$$

Let us recall from [18] that a reverse polyblock in $M = [p, q]$ is a subset S of M of the form $S = \cup_{i\in I}[p^i, q]$ where $\{p^i, i \in I\}$ is a finite set of points in M. Since $\Psi_M(y)$ is an increasing function on M it is clear that the minimum of $\Psi_M(y)$ on such a set S is attained at one of the points $p^i, i \in I$. The ROA method is an iterative procedure which consists in constructing a sequence of reverse polyblocks $S_\nu, \nu = 1, 2, \ldots$ such that

$$S_1 = M \subset S_2 \subset \cdots \subset H \cap M, \ \min_{y\in S_\nu} \Psi_M(y) \searrow \min_{H\cap M} \Psi_M(y). \tag{7}$$

Clearly the numbers $\beta_\nu(M) = \min_{y\in S_\nu} \Psi_M(y), \nu = 1, 2, \ldots$ form an increasing sequence of lower bounds for the minimum of $\Psi_M(y)$ over M. By stopping this iterative procedure ROA at some step ν we obtain a lower bound $\beta_\nu(M)$. Of

course the larger ν the tighter bound is obtained, but the more computations are needed, so in practice a trade-off should be resolved between the quality of the bound and the computational cost required. As is easily verified, $\beta_1(M) = \min_{y\in M} \Psi_M(y) = \Psi_M(p)$, so from Proposition 1 we can derive

Corollary 1.

(i) *A lower bound for the optimal value of* (G_M) *is provided by the optimal value* $\beta_1(M)$ *of the linear program*

$$\min\{f_0(x) + \langle p, g(x)\rangle |\ x \in D_{[p,q]}\}. \qquad (\mathrm{LG}_M)$$

(ii) *If an optimal solution* x^M *of* (LG_M) *satisfies* $f(x^M) = p$ *then* p *is an optimal solution of* (G_M) *and* $\beta_1(M)$ *equals the optimal value of* (G_M)*, i.e.*

$$\beta_1(M) = \Phi(p) = f_0(x^M) + \langle p, g(x^M)\rangle.$$

Remark 2. The quality of the bound $\beta_1(M)$ computed as above often depends on how tightly the rectangle $M = [p, q]$ encloses the feasible portion in it. Therefore, whenever possible, one should try to reduce the size of the rectangle M by removing unnecessary infeasible portions of the set $M \setminus H$. Since $H \cap M = \{y|\ p \le y \le q, f(x) \le y, x \in D\}$, one way to achieve this is to replace p, q with $p' = (p'_1, \dots, p'_m), q' = (q'_1, \dots, q'_m)$, respectively, where

$$\begin{aligned} p'_i &= \min\{f_i(x)|\ p \le f(x) \le q, x \in D\}, \quad i = 1, \dots, m, \\ q'_i &= \max\{f_i(x)|\ p \le f(x) \le q, x \in D\}, \quad i = 1, \dots, m. \end{aligned}$$

For many problems this reduction operation may be essential for the quality of the bound and hence the speed of convergence.

Consider now a nested sequence of subrectangles $M_k \subset M$ such that $\cap_{k=1}^{\infty} M_k = \bar{M}$ and let $x^k = x^{M_k}$ be an optimal solution of (LG_{M_k}). The next proposition shows how to construct the sequence $M_k = [p^k, q^k]$ so as to ensure that the limit of x^k is an optimal solution of the problem $(\mathrm{P}_{\overline{M}})$.

Proposition 5. *Let* $\{M_k = [p^k, q^k],\ k \in K\}$ *be an infinite nested sequence of subrectangles of* $[a, b]$*, such that* M_{k+1} *is a child of* M_k *in the subdivision via the hyperplane* $y_{i_k} = \eta_k$ *where* $\eta_k = (p^k_{i_k} + f_{i_k}(x^k))/2$ *and* $x^k = x^{M_k}$ *is an optimal solution of* (LG_{M_k}) *while* $i_k \in \operatorname{argmax}(f_i(x^k) - p^k_i)$*. If* $[\bar{p}, \bar{q}] = \cap_{k=1}^{+\infty}[p^k, q^k]$ *then any cluster point* $\bar{x}$ *of the sequence* $\{x^k\}$ *is an optimal solution of* $(P_{[\bar{p},\bar{q}]})$*.*

Proof. Since $i_k \in \{1, \dots, m\}$, there exists a subsequence $K \subset \{1, 2, \dots\}$ such that $i_k = i_0\ \forall k \in K$, and $\eta_k = (p^k_{i_0} + f_{i_0}(x^k))/2,\ \forall k \in K$. By passing to subsequences if necessary we can assume that $x^k \to \bar{x},\ f_{i_0}(x^k) \to f_{i_0}(\bar{x})$. Then, by a well known property of rectangular subdivision (Lemma 5.4,[16]), $\eta_k \to \eta_0 \in \{\bar{p}_{i_0}, \bar{q}_{i_0}\}$, which, in view of the fact $\eta_k = (p^k_{i_0} + f_{i_0}(x^k)/2 \to (\bar{p}_{i_0} + f_{i_0}(\bar{x}))/2$ implies that $f_{i_0}(x^k) - p^k_{i_0} \to f_{i_0}(\bar{x}) - \bar{p}_{i_0} = 0$. But then from the definition of $i_0 = i_k$ we deduce that $f(x^k) - p^k \to 0$ and consequently, $f(x^k) \to \bar{p}$ as $k \to +\infty, k \in K$. Since x^k is an optimal solution of (LG_{M_k}), it follows that $\bar{x}$ is

an optimal solution of $(\mathrm{LG}_{[\bar{p},\bar{q}]})$ and finally, by Proposition 1 (property (ii)), the fact $f(\bar{x}) = \bar{p}$ implies that $\bar{x}$ solves $(\mathrm{P}_{[\bar{p},\bar{q}]})$. ∎

Remark 3. An increasing function $\varphi_M(y)$ is said to be a *tight underestimator* of $\Phi(y)$ over M if $\varphi_M(y) \leq \Phi(y)\ \forall y \in M$ and $\max_{y\in M}(\Phi(y) - \varphi_M(y)) \to 0$ as $\mathrm{diam}M = \|q - p\| \to 0$. It can easily be verified that Proposition 5 remains true when $(\mathrm{LG}(M_k)$ is taken to be the problem

$$\min\{\varphi_M(y)|\ y \geq f(x),\ x \in D_{[p,q]}\},$$

where $\varphi_M(y)$ is any tight underestimator of $\Phi(y)$ over M.

4. Branch and Bound Method

The above development leads to the following branch and bound method for solving (MP), in which branching is performed by rectangular subdivision of the initial rectangle $[a, b]$.

Algorithm 1. (for given tolerance $\varepsilon \geq 0$)

Step 0. Start with $\mathcal{P}_1 = \mathcal{S}_1 = \{M_1 = [a, b]\}$. Set $k = 1$.

Step 1. For each rectangle $M \in \mathcal{P}_k$ solve (LG_M) to obtain its optimal value $\beta_1(M)$ together with an optimal solution x^M when $\beta_1(M) < +\infty$. Update the incumbent by setting $CBS = \bar{x}^k$ with $\bar{x}^k \in \mathrm{argmin}\{f_0(x^M) + \langle f(x^M), g(x^M)\rangle|\ M \in \mathcal{P}_k\}$. Let $CBV = f_0(\bar{x}^k) + \langle f(\bar{x}^k), g(\bar{x}^k)\rangle$.

Step 2. Delete every $M \in \mathcal{S}_k$ such that $\beta_1(M) \geq CBV - \varepsilon$. Let $\mathcal{R}_k$ be the collection of remaining members of $\mathcal{S}_k$. If $\mathcal{R}_k = \emptyset$, then terminate: $\bar{x}^k$ is an ε-optimal solution.

Step 3. Select $M_k \in \mathrm{argmin}\{\beta_1(M)|\ M \in \mathcal{R}_k\}$. Let $x^k = x^{M_k}$, $y^k = f(x^k)$. Choose $i_k \in \mathrm{argmax}_i\{y^k_i - p^k_i\}$ and divide M_k into two subrectangles via the hyperplane $y_i = (p^k_{i_k} + y^k_{i_k})/2$. Reduce each rectangle in the partition of M_k as described in Remark 2 and let $\mathcal{P}_{k+1}$ be the collection of so reduced rectangles.

Step 4. Set $\mathcal{S}_{k+1} = (\mathcal{R}_k \setminus \{M_k\}) \cup \mathcal{P}_{k+1}$. Set $k \leftarrow k + 1$ and go back to Step 1.

Theorem 1. *The above algorithm can be infinite only if $\varepsilon = 0$ and in that case any cluster point $\bar{x}$ of the sequence $\{\bar{x}^k\}$ is an optimal solution.*

Proof. If the algorithm is infinite, it generates an infinite sequence of rectangles $M_{k_\nu} = [p^{k_\nu}, q^{k_\nu}]$, $\nu = 1, 2, \ldots$, such that each rectangle in this sequence is a child of its predecessor via the subdivision $(p^{k_\nu} + f_{i_{k_\nu}}(x^{k_\nu}))/2, i_{k_\nu})$. By Proposition 5 one can then assume that $p^{k_\nu} \to \bar{p}, q^{k_\nu} \to \bar{q}$, $x^{k_\nu} \to \bar{x}$ such that $\bar{x}$ solves $(\mathrm{P}_{\bar{M}})$ for $\bar{M} = [\bar{p}, \bar{q}]$. Since $f_0(x^{k_\nu}) + \langle f(x^{k_\nu}), g(x^{k_\nu})\rangle = \beta_1(M_{k_\nu}) \leq f_0(x) + \langle f(x), g(x)\rangle\ \forall x \in D$, it follows that $f_0(\bar{x}) + \langle f(\bar{x}), g(\bar{x})\rangle \leq f_0(x) + \langle f(x), g(x)\rangle\ \forall x \in D$, and hence, $\bar{x}$ solves (P) because obviously $\bar{x} \in D$. ∎

Remark 4. For problems where $f_0(x) + \langle a, g(x)\rangle \equiv 0$ (e.g. $a = 0, f_0(x) \equiv 0, f(x)$ linear), if $\beta_1(M)$ is used then at the first iteration we have $\beta_1(M_1) = 0$ since the

objective function in the linear program (LG_{M_1}) for $M_1 = [a, b]$ is identical to zero. Furthermore, in all subsequent iterations, $M_k = [0, q^k]$, hence $\beta_1(M_k) = 0$, so that the algorithm will stall right at the beginning. Therefore, for these problems, one should use $\beta_\nu(M)$ with $\nu > 1$ instead of $\beta_1(M)$, at least for every rectangle $M = [p, q]$ with $p = 0$. (except when $m = 1$ because in that case, as can easily be shown (see Sec. 5 below), $\beta_1(M)$ is already the optimal value of (RG_M)). Alternatively, one can reformulate the problem to avoid this situation, or use a tighter underestimator than $\Psi_M(y)$ as will be discussed in the next section.

Remark 5. The type of subdivision described in Proposition 5 and used in Algorithm 1 is often referred to as the "adaptive subdivision". The rationale behind it is that it tends to bring the distance $\|f(x^k) - p^k\|$ to zero, thus forcing $\{x^k\}$ to converge to an optimal solution of ($\mathrm{P}_{[\bar{p},\bar{q}]}$). Howewer, if $[a, b]$ is bounded, then instead of the adaptive subdivision one can also use the standard bisection, i.e. the subdivision of M_k via the hyperplane $y_{i_k} = (p^k_{i_k} + q^k_{i_k})/2$ with $i_k = \operatorname{argmax}(q^k_{i_k} - p^k_{i_k})$. Proposition 5 remains true with the standard bisection because its exhaustiveness (see e.g. [16]) implies that $\|q^k - p^k\| \to 0$ and hence $\|f(x^k) - y^k\| \to 0$. In practice, the choice of an efficient subdivision rule depends on the structure of the problem under study.

Remark 6. Also note that when using the standard bisection it may happen that the midpoint of the longest side of $M_k = [p^k, q^k]$ lies on the right of $f_{i_k}(x^k)$ (x^k being an optimal solution of $\mathrm{LG}(_{M_k})$), i.e. $f_{i_k}(x^k) \le (p^k_{i_k} + q^k_{i_k})/2$: in that event, if M_k is bisected via the mentioned midpoint then the lower bound over the left half of M_k will remain equal to $\beta(M_k)$ (because x^k will still be optimal for the corresponding linear program), so this bisection will not improve the lower bound. Therefore, the best strategy is to use a hybrid rule: if the midpoint of the longest edge of M_k lies on the left of $f_{i_k}(x^k)$ then apply the standard bisection, otherwise apply the adaptive subdivision.

Remark 7. With minor modifications Algorithm 1 is still valid if a tight underestimator other than $\Psi_M(y)$ is used for bounding. Also the above method can in principle be extended to solve the generalized convex multiplicative programming problem considered in [5]:

$$f_0(x) + \min\{\sum_{i=1}^{m} f_i(x)g_i(x) |\ x \in D\}$$

where $D = \{x \in R^n |\ h(x) \le 0\}$ and the functions $f_0(x), f_i(x), g_i(x)$ are all convex and positive-valued on D. For instance, if the underestimator $\Psi_M(y)$ is used for bounding, the subproblems (LP_M) are the convex minimization problems

$$\min\{f_0(x) + \sum_{i=1}^{m} p_i g_i(x) |\ p_i \le f_i(x) \le q_i\ (i = 1, \dots, m),\ h(x) \le 0\}$$

which are solvable by currently available efficient algorithms.

5. Finite Algorithm when $m = 1$

When $m = 1$, the problem (MP) is

$$\min\{f_0(x) + f(x)g(x)|\ x \in D\} \tag{8}$$

with $f, g : R^n \to R$. It is then easily verified that for any $M = [p, q] \subset [a, b] \subset R$ with $a = \min_{x\in D} f(x)$, an optimal solution of (RG_M) is precisely $p = \min\{f(x)|\ x \in D, p \le f(x) \le q\}$. In other words, $\beta_1(M)$ is already the best lower bound obtained by the above method (optimal value of RG_M). However, in this case the algorithm can be made finite by adopting a more sophisticated subdivision rule.

To describe this subdivision let us consider the problem (MP) where $y = f(x)$ is viewed as a parameter:

$$\min\{f_0(x) + yg(x)|\ f(x) = y, x \in D\}. \tag{MP(y)}$$

Proposition 6. *There exist a finite set of intervals $\Delta^B \subset [a, b]$ $(B \in \mathcal{B})$ and affine mappings $\varphi^B : \Delta^B \to R^n$, such that $\cup\{\Delta^B|\ B \in \mathcal{B}\} = [a, b]$ and $\varphi^B(y)$ is a basic optimal solution of* (MP(y)) *for every $y \in \Delta^B$.*

Proof. Rewrite the problem (MP(y)) for any $y \in [a, b]$, as

$$\min\{\langle c + yc^0, x\rangle + \alpha(y)|\ Ax = d + yd^0,\ x \ge 0\} \tag{9}$$

where $\alpha(y) = f_0(0) + yg(0)$ and $A \in R^{k\times n}$ with $\operatorname{rank} A = k$. Let B be any $k \times k$ nonsingular submatrix of A and N the submatrix formed by the $n - k$ columns of A not belonging to B. Write $A = (B, N)$ and, accordingly, $x = (x_B, x_N)$ for any $x \in R^n$. Then the system $Ax = d + yd^0$ is equivalent to

$$x_B = B^{-1}(d + yd^0) - B^{-1}Nx_N. \tag{10}$$

Denote by $\varphi^B(y)$ the vector $x = (x_B, x_N)$ such that $x_B = B^{-1}(d + yd^0), x_N = 0 \in R^{n-k}$. This vector will be a feasible solution of (MP(y)) if

$$B^{-1}(d + yd^0) \ge 0. \tag{11}$$

On the other hand, since $\langle c+yc^0, x-\varphi^B(y)\rangle = (c+yc^0)_N x_N - (c+yc^0)_N B^{-1}Nx_N$ for any feasible solution $x = (x_B, x_N)$ of (MP(y)), it follows that $\varphi^B(y)$ is an optimal solution of (MP(y)) if

$$(c + yc^0)_N - (c + yc^0)_B B^{-1}N \ge 0. \tag{12}$$

Now define

$$\Delta^B = \{y|\ B^{-1}(d + yd^0) \ge 0,\ (c + yc^0)_N - (c + yc^0)_B B^{-1}N \ge 0\}. \tag{13}$$

Clearly Δ^B is a line segment contained in $[a, b]$ and for every $y' \in \Delta^B$ the vector $\varphi^B(y')$ satisfies conditions analogous to (11) and (12), hence is a basic optimal solution of (MP(y')). It is also immediate that, for any $y \in [a, b]$, since (MP(y) is solvable, we have $y \in \Delta^B$, where B is the basis matrix associated with a basic optimal solution of (MP(y)). Therefore, if $\mathcal{B}$ denotes the family of all basis matrices B that correspond each to a basic optimal solution of (MP(y)) for some

$y \in [a, b]$, then the union of all $\Delta^B, B \in \mathcal{B}$, covers the whole segment $[a, b]$. Since the collection $\mathcal{B}$ is finite, and the mapping $\varphi^B : \Delta^B \to R^n$ is affine (see (11)), the proof is complete. ∎

Note that for $y \in \Delta^B$ the optimal value of (MP(y)) is $\theta^B(y) = f_0(\varphi^B(y)) + yg(\varphi^B(y))$. Since this is a quadratic function of $y \in R$, a minimizer y^B of $\theta^B(y)$ over Δ^B can easily be computed and if $z^B = \varphi^B(y^B)$ then an optimal solution of (MP) is obtained by taking the minimum of $f_0(z^B) + f(z^B)g(z^B)$ over all $B \in \mathcal{B}$.

The parametric algorithm for (MP) when $m = 1$ ([5]; see also [16]) consists precisely in generating the collection of intervals $\Delta^B,\ B \in \mathcal{B}$ together with the associated vectors z^B. Since the set $\mathcal{B}$ is finite, the parametric algorithm is finite, in contrast to the branch and bound Algorithm 1 which, theoretically, may be infinite. However, a drawback of the parametric approach is that it requires computing the whole collection of intervals $\Delta^B, B \in \mathcal{B}$.

To overcome this drawback we can make our branch and bound procedure finite by using a suitable subdivision strategy based on Proposition 6. Denote the set of endpoints of all intervals Δ^B by T. Clearly $a, b \in T$. If $M = [p, q]$ is any interval candidate for subdivision, with $p, q \in T$, then we subdivide M as follows. Let x^M be a basic optimal solution of the bounding subproblem (LG_M) written in the form (9) (with $y = p$) and let $B = B(M)$ be the basis matrix associated with x^M. It can easily be verified that $\Delta^B = [p, r]$, where $r = \min\{r_1, r_2\}$ with

$$r_1 = \max\{y|\ (c+yc^0)_N - (c+yc^0)_B B^{-1}N \geq 0\},\ \ r_2 = \max\{y|\ \{y|\ B^{-1}(d+yd^0) \geq 0\}.$$

Then we divide M into two subintervals $M_1 = [p, r], M_2 = [r, q]$. Clearly $\beta(M_1) = \beta(M)$, so only $\beta(M_2)$ must be computed. Furthermore, a minimizer y^B of the above described quadratic function $\theta^B(y)$ over Δ^B can be computed which can be used to update the current best solution and the current best value. If $\beta(M_2)$ does not exceed the current best value, then the algorithm terminates because all previously generated intervals have been fathomed. Therefore, if the algorithm continues, the interval to be divided must be $M_2 = [r, q]$, so that the basis matrix corresponding to the subdivision point s for M_2 can be obtained merely by performing a primal simplex privoting procedure if $r = r_1$ or a dual simplex pivoting procedure if $r = r_2$. With this subdivision strategy the algorithm differs from the parametric approach of [5] only in that a lower bound $\beta(M)$ is computed at each iteration for the current interval which allows us to decide whether or not to terminate the algorithm even before the whole collection of intervals $\Delta^B, B \in \mathcal{B}$ has been generated.

An alternative subdivision strategy is to slightly modify the standard bisection in the following way. Starting from the initial interval $[p_0, q_0] = [a, b]$, assume that the current interval $M = [p, q]$ to be subdivided has its endpoints in T. For $y = (p+q)/2$ let B be the basis matrix associated with a basic optimal solution of (MP(y)), and define Δ^B by (13). If $\Delta^B = [r, s]$ then split M into 3 intervals : $M_1 = [p, r],\ \Delta^B = [r, s]$, and $M_2 = [s, q]$. Clearly $\beta(M_1) = \beta(M)$, while a minimizer y^B of $\varphi(y)$ over $[r, s]$ can be computed which can be used to

update the current best solution and current best value. It remains to compute $\beta(M_2) = \min\{f_0(x) + sg(x) |\ f(x) = s, x \in D_{[s,q]}\}$. In the next iteration M is thus replaced by the pair of intervals $[p, r], [s, q]$. This branch and bound algorithm may perform better than Algorithm 1 although a few more computations are required at each iteration.

Remark 8. Proposition 6 can be extended to the general case $m > 1$ as follows: *There exist a finite set of polyhedrons* $\Delta^B \subset [a, b]$, *and affine mappings* $\varphi^B : \Delta^B \to R^n$, *such that the union of all these polyhedrons covers the entire rectangle* $[a, b]$ *and* $\varphi^B(y)$ *is a basic optimal solution of (MP(y)) for every* $y \in \Delta^B$.

The proof of this proposition for the special case $m = 1$ can be carried over to the general case. However, as was mentioned in the introduction, the parametric approach derived from this proposition for $m > 1$ is of little practical interest.

6. Improved Bounds for Higher Rank Problems

For hard problems, especially for those with high rank m (typically $m \geq 10$) the bounds computed as previously described may not be efficient enough to secure fast convergence. To alleviate the difficulty in these cases, one may try to improve the bounds in different ways, for instance:
(i) *Solving* (RG_M) *more accurately.* Instead of $\beta_1(M)$ in Step 1 we can take $\beta_\nu(M)$ with $\nu > 1$ as explained in Section 3. Then $\beta_\nu(M) = \Psi_M(y^M)$ for some $y^M \in M$ and x^M is an optimal solution of the linear program

$$\min\{f_0(x) + \langle y^M, g(x)\rangle |\ p \leq f(x) \leq q, x \in D\}.$$

Note that the value $\Psi_M(y)$ of the objective function of (RG_M) at each y is not given explicitly but is defined as the optimal value of a linear program (5). This means that computing $\beta_2(M)$ involves solving a sequence of m linear programs, corresponding to the m vertices of S_2 (see (7)). Each of these vertices differs from p by one coordinate only, so after solving the linear program associated with p, one can derive an optimal solution of any other program by using a reoptimization technique instead of starting from scratch. This allows a substantial saving of time. Nevertheless, even with this technique, it should be borne in mind that the computation of $\beta_\nu(M)$ with $\nu > 1$ may become so time consuming that the advantage of having a better bound may be offset by its cost. Thus, as was already mentioned in Sec. 3, there is a trade-off between the computational cost and the quality of the bound.
(ii) *Using a tighter underestimator for* $\Phi(y)$. Consider any rectangle $M = [p, q]$. Let $r = (r_1, \ldots, r_m)$ where $r_i = \min\{g_i(x) |\ x \in D_{[p,q]}\}$. Then for $y \leq f(x),\ x \in D_{[p,q]}$ we have $\langle f(x) - y, g(x) - r\rangle \geq 0$, hence

$$\langle f(x), g(x)\rangle \geq \langle y, g(x)\rangle + \langle r, f(x)\rangle - \langle r, y\rangle.$$

So if we define

$$\tilde{\Psi}_M(y) = \min\{f_0(x) + \langle y, g(x) - r\rangle + \langle r, f(x)\rangle |\ y \leq f(x), x \in D, p \leq f(x) \leq q\}$$

then $\Psi_M(y) \leq \tilde{\Psi}_M(y) \leq \Phi(y)$ for all y belonging to the set $\{y|\ y \leq f(x), x \in D_{[p,q]}\}$. It follows that a lower bound of $\Phi(y)$ over $H \cap M$ (see (G_M)) can be taken to be

$$\tilde{\beta}(M) = \min\{\tilde{\Psi}_M(y)|\ y \in H \cap M\}. \tag{14}$$

Since $g(x) - r \geq 0$ the function $\tilde{\Psi}_M(y)$ is also increasing, so the problem on the right hand side is again a monotonic optimization problem. If the ROA procedure for solving this problem is stopped at iteration ν then a value $\tilde{\beta}_\nu(M)$ is obtained that gives a lower bound for $\min\{\Phi(y)|\ y \in H \cap M\}$. In particular for $\nu = 1$ we have $\tilde{\beta}_1(M) = \tilde{\Psi}_M(p)$, so

$$\tilde{\beta}_1(M) = \min\{f_0(x) + \langle p, g(x) - r\rangle + \langle r, f(x)\rangle|\ y \leq f(x),\ \ x \in D, p \leq f(x) \leq q\}$$

should give a lower bound generally tighter than $\beta_1(M)$.

(iii) *Reformulating the problem appropriately.* A problem may be hard for a given method just because its formulation is not suitable. In such cases a reformulation may help to alleviate or resolve the difficulty. In general, when the objective as well as the constraint functions are already increasing or decreasing functions, one may be tempted to think that no further transformation is worthwhile, except, possibly, for reducing the dimension. However, there are cases when a reformulation is useful even though it may convert a monotonic objective function into a d.m. function (difference of two monotonic functions). The last example in the next section will show a situation where such a transformation may improve the algorithm drastically.

Finally an alternative approach to high rank problems (MP) is to rewrite the problem as a monotonic optimization problem of the form

$$\min\{F^+(x) - F^-(x)|\ \langle c^+, x\rangle - \langle c^-, x\rangle \geq 0,\ x \geq 0\},$$

where F^+, F^- are quadratic polynomials with positive coefficients and $c^+, c^- \in R^n_+$. Then a lower bound over any rectangle $M = \{x \in R^n_+|\ p \leq x \leq q\}$ can be computed by general methods of monotonic optimization. For detail we refer the interested reader to the recent work [20].

7. Numerical Examples

In this section we give some numerical examples, just to illustrate how the method works in practice. In these examples the problem has the form (MP) where

$$\begin{aligned} f_i(x) &= \langle c^i, x\rangle + r_i, \quad i = 1, \ldots, m, \\ g_i(x) &= \langle d^i, x\rangle + s_i, \quad i = 1, \ldots, m, \\ D &= \{x|\ Ax \leq h, \quad x \geq 0\}, \end{aligned}$$

and the functions $g_i(x)$ are positive on D. The computational results have been obtained using a code written in Pascal and run on a PC Pentium III 450 MHz. The first four examples are taken from the literature. The fifth and the sixth

examples are chosen to show that the computational time depends much more on m (the number of products) than on n; they could also serve as test problems for future research in nonconvex optimization. The last example is an instance of the maximum clique problem taken from [4]. It is given to show how a good reformulation of the problem may sometimes help to improve the efficiency of the method significantly.

Example 1. (Example 5.2 in [15]) (minimization) $n = 2, m = 1,\ A \in R^{4\times 2}$,

$$c^0 = (1,0), \quad r_0 = 0$$
$$c^1 = (1,1), \ \ r_1 = -1, \quad d^1 = (\ 2,-3), \ \ s_1 = 13$$
$$A = \begin{bmatrix} -1 & 2 \\ 0 & -1 \\ 1 & 2 \\ 1 & -2 \end{bmatrix}, \quad h = (8,-3,\ 12,-5).$$

Computation results (tolerance $\varepsilon = 0.001$) :
Optimal solution: $x = (0,4)$;
Optimal value: 3;
Computation time: 0.01 sec;
Optimal solution found at iteration 1 and confirmed at iteration 1;
Maximal number of active nodes: 2.

Example 2. (Taken from [1]) (minimization) $n = 3, m = 1,\ A \in R^{8\times 3}$

$$c^0 = (0.000000, \ \ 0.000000, \ \ 0.000000) \ \ r_0 = 0.000000,$$
$$c^1 = (1.000000, \ \ 0.000000, \ \ 0.111111) \ \ r_1 = 0.000000,$$
$$d^1 = (0.000000, \ \ 1.000000, \ \ 0.111111) \ \ s_1 = 0.000000,$$
$$h = (81,\ 72,\ 72,\ -9,\ -9,\ -9,\ 8,\ 8),$$
$$A = \begin{bmatrix} 9 & 9 & 2 \\ 8 & 1 & 8 \\ 1 & 8 & 8 \\ -7 & -1 & -1 \\ -1 & -7 & -1 \\ -1 & -1 & -7 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

Computational results (tolerance $\varepsilon = 0.001$) :
Optimal solution: $x = (8,\ 0,\ 1)$;
Optimal value: 0.901234;
Computation time: 0.22 sec;
Optimal solution found at iteration 3 and confirmed at iteration 24;
Maximal number of active nodes: 4.

Example 3. (Example 5.1 in [16]; quadratic concave minimization)

$$\begin{aligned}
\text{minimize}\quad & x_1 - 10x_2 + 10x_3 + x_8 \\
& - x_1^2 - x_2^2 - x_3^2 - x_4^2 - 7x_5^2 - 4x_6^2 - x_7^2 - 2x_8^2 \\
& 2x_1x_2 + 6x_1x_5 + 6x_2x_5 + 2x_3x_4, \\
\text{s.t.}\quad & x_1 + 2x_2 + x_3 + x_4 + x_5 + x_6 + x_7 + x_8 \le 8, \\
& 2x_1 + x_2 + x_3 \le 9, \\
& x_3 + x_4 + x_5 \le 5, \\
& 0.5x_5 + 0.5x_6 + x_7 + 2x_8 \le 3, \\
& 2x_2 - x_3 - 0.5x_4 \le 5, \\
& x_1 \le 6,\ x_i \ge 0\ i = 1, \dots, 8.
\end{aligned}$$

Computational results (tolerance $\varepsilon = 0.001$) :
Optimal solution: $x = (0, 0, 0, 0, 5, 1, 0, 0)$;
Optimal value: -179;
Optimal solution found at iteration 2 and confirmed at iteration 56;
Computation time: 0.88 sec;
Maximal number of active nodes: 3.

Example 4. (Test problem 10, Chapter 2, [4])

$$\min\{-0.5\sum_{i=1}^{10}\lambda_i(x_i+\alpha_i)^2 + 0.5\sum_{i=1}^{10}\mu_i(y_i+\beta_i)^2 |\ A(x,y) \le h,\ (x,y) \in R_+^{10} \times R_+^{10}\}$$

where

$$\begin{gathered}
\lambda = (63, 15, 44, 91, 45, 50, 89, 58, 86, 82), \\
\mu = (-42, -98, -48, -91, -11, -63, -61, -61, -38, -26), \\
\alpha = (19, 27, 23, 53, 42, -26, 33, 23, -41, -19), \\
\beta = (52, 3, -81, -30, 85, -68, -27, 81, -97, 73), \\
h = (380, 415, 385, 405, 470, 415, 400, 460, 400, 200)^T,
\end{gathered}$$

$$A = \begin{bmatrix}
3 & 5 & 5 & 6 & 4 & 4 & 5 & 6 & 4 & 4 & 8 & 4 & 2 & 1 & 1 & 1 & 2 & 1 & 7 & 3 \\
5 & 4 & 5 & 4 & 1 & 4 & 4 & 2 & 5 & 2 & 3 & 6 & 1 & 7 & 7 & 5 & 8 & 7 & 2 & 1 \\
1 & 5 & 2 & 4 & 7 & 3 & 1 & 5 & 7 & 6 & 1 & 7 & 2 & 4 & 7 & 5 & 3 & 4 & 1 & 2 \\
3 & 2 & 6 & 3 & 2 & 1 & 6 & 1 & 7 & 3 & 7 & 7 & 8 & 2 & 3 & 4 & 5 & 8 & 1 & 2 \\
6 & 6 & 6 & 4 & 5 & 2 & 2 & 4 & 3 & 2 & 7 & 5 & 3 & 6 & 7 & 5 & 8 & 4 & 6 & 3 \\
5 & 5 & 2 & 1 & 3 & 5 & 5 & 7 & 4 & 3 & 4 & 1 & 7 & 3 & 8 & 3 & 1 & 6 & 2 & 8 \\
3 & 6 & 6 & 3 & 1 & 6 & 1 & 6 & 7 & 1 & 4 & 3 & 1 & 4 & 3 & 6 & 4 & 6 & 5 & 4 \\
1 & 2 & 1 & 7 & 8 & 7 & 6 & 5 & 8 & 7 & 2 & 3 & 5 & 5 & 4 & 5 & 4 & 2 & 2 & 8 \\
8 & 5 & 2 & 5 & 3 & 8 & 1 & 3 & 3 & 5 & 4 & 5 & 5 & 6 & 1 & 7 & 1 & 2 & 2 & 4 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1
\end{bmatrix}$$

By rewriting the objective function as $\sum_{i=1}^{10} f_i(x,y)g_i(x,y)$ and solving the problem by Algorithm 1 the following results have been obtained (tolerance $\varepsilon = 0.001$) :

Optimal solution:
$x = (0, 0, 0, 62.608696, 0, 0, 0, 0, 0, 0),\ y = (0, 0, 0, 0, 0, 4.347826, 0, 0, 0, 0)$;
(the solution given in [4] is wrong, probably due to a misprint)
Optimal value: 49318.017957;
Computation time: 3.44 sec;
Optimal solution found at iteration 5 and confirmed at iteration 121;
Maximal number of active nodes: 4.

Example 5. (minimization) $n = 12, m = 6,\ A \in R^{15\times 12}$

$c^1 = (-0.2, -0.7, -0.1,\ 0.4,\ 0.0,\ 0.8,\ 0.1, -0.8, -0.2,\ 0.0,\ 0.1,\ 0.4),\ r_1 = 21.0,$
$d^1 = (0.2,\ 0.5, -0.6,\ 0.1,\ 0.6,\ 0.4, -0.4, -0.3,\ 0.7,\ 0.5,\ 0.4, -0.1),\quad s_1 = 13.3,$
$c^2 = (-0.1,\ 0.1, -0.4, -0.1, -0.1,\ 0.4,\ 0.2,\ 0.5,\ 0.3, -0.4, -0.3,\ 0.3), r_2 = 16.3,$
$d^2 = (-0.3, -0.2, -0.7,\ 0.1,\ 0.2, -0.2, -0.5,\ 0.4,\ 0.3,\ 0.0,\ 0.6, -0.5), s_2 = 16.0$
$c^3 = (0.8,\ 0.0, -0.1,\ 0.4,\ 0.2,\ 0.1, -0.5,\ 0.0,\ 0.5,\ 0.6, -0.3, -0.4),\quad r_3 = 3.7$
$d^3 = (0.1,\ 0.0,\ 0.0,\ 0.3,\ 0.2,\ 0.7,\ 0.4,\ 0.2, -0.1, -0.5,\ 0.6, -0.1),\quad s_3 = 16.7$
$c^4 = (0.6,\ 0.2,\ 0.2, -0.3,\ 0.5,\ 0.4,\ 0.1,\ 0.6, -0.3,\ 0.3,\ 0.4,\ 0.3),\quad r_4 = -1.8$
$d^4 = (-0.3,\ 0.0,\ 0.0, -0.5, -0.1,\ 0.2,\ 0.6, -0.6,\ 0.1, -0.2,\ 0.8, -0.3), s_4 = 21.5$
$c^5 = (-0.3, -0.3,\ 0.5,\ 0.1,\ 0.2, -0.5,\ 0.1,\ 0.2,\ 0.0,\ 0.6,\ 0.3, -0.2),\quad r_5 = 5.0$
$d^5 = (0.3,\ 0.0,\ 0.3,\ 0.0, -0.8, -0.3,\ 0.3, -0.9, -0.1, -0.6, -0.1,\ 0.2), s_5 = 18.7$
$c^6 = (0.2, -0.1,\ 0.0,\ 0.0, -0.2, -0.4,\ 0.0, -0.6,\ 0.8, -0.2,\ 0.0, -0.1), r_6 = 12.7$
$d^6 = (0.0,\ 0.6,\ 0.0,\ 0.1,\ 0.0, -0.2,\ 0.0, -0.5,\ 0.2, -0.3,\ 0.3,\ 0.1),\quad s_6 = 19.2$

$h = (-20.1, -1.0, 82.6, 14.6, 37.7, 40.7, -23.0, 47.4, 83.0, 9.9, 33.7, 14.0, -45.6, 30.4)$

$$A = \begin{bmatrix}
1.9 & 0.0 & -0.2 & -1.5 & 1.8 & 0.9 & -1.0 & 4.5 & 4.5 & -3.5 & -1.8 & -4.8 \\
2.9 & 3.7 & -4.8 & -1.9 & 1.8 & -3.7 & 1.8 & 2.5 & -2.9 & 1.9 & -3.0 & 3.2 \\
3.3 & 2.4 & 3.3 & 4.8 & -0.3 & 3.9 & 0.8 & -1.7 & 2.0 & -0.3 & -1.8 & 2.2 \\
-4.3 & 1.8 & 2.1 & -4.5 & -0.5 & 2.4 & 1.4 & -0.3 & -2.0 & -2.8 & 0.4 & 4.5 \\
1.5 & -0.3 & 0.4 & 1.2 & 1.1 & 1.9 & 1.5 & -1.2 & -3.3 & 4.4 & 3.2 & -4.3 \\
-3.2 & 2.4 & -4.5 & -1.0 & -2.7 & 3.7 & -0.1 & 3.9 & -1.9 & 3.2 & 2.1 & 1.3 \\
0.9 & 0.5 & 4.0 & -1.5 & 1.2 & -1.5 & 1.2 & -3.7 & -0.1 & 0.0 & -2.4 & -4.1 \\
-4.1 & -4.5 & 2.2 & -3.1 & 4.4 & 4.8 & -3.4 & 2.2 & -2.1 & 2.3 & 2.6 & -1.4 \\
2.4 & 2.3 & 4.7 & -1.7 & -1.6 & 3.8 & -4.0 & 1.3 & -0.4 & -0.4 & 2.9 & 1.2 \\
0.0 & -3.2 & -0.2 & 2.0 & -2.9 & 2.7 & 3.1 & 2.9 & -2.6 & -4.3 & 0.2 & 4.6 \\
-1.3 & -0.9 & 3.4 & 3.9 & 4.9 & 2.3 & -3.0 & -1.5 & 2.5 & -1.7 & 1.7 & -2.9 \\
3.5 & 3.4 & 2.5 & -0.4 & -4.5 & 2.8 & -1.7 & 2.1 & -2.9 & -4.7 & 1.3 & 4.5 \\
1.9 & -0.9 & -3.3 & -2.3 & 1.6 & -0.5 & -4.9 & 3.0 & -4.9 & 3.6 & -3.7 & 2.2 \\
-1.4 & 3.5 & -2.8 & -1.2 & -4.7 & -3.2 & 2.2 & -4.0 & 2.8 & 3.3 & 4.4 & -3.1 \\
-2.1 & 2.6 & -3.9 & 1.0 & 2.3 & 1.8 & 4.2 & 1.8 & 2.7 & 0.9 & 3.3 & 1.7
\end{bmatrix}$$

Computational results (tolerance $\varepsilon = 0.001$) :
Optimal solution:

$x = (0.000000,\ 3.191426,\ 15.976649,\ 7.637881,\ 0.000000,\ 0.000000,$
$2.089829,\ 14.934635,\ 1.673765,\ 14.492286,\ 0.865866,\ 5.377971);$

Optimal value: 550.937336;
Computation time: 2.91 sec;
Optimal solution found at iteration 19 and confirmed at iteration 144;
Maximal number of active nodes: 15.

For this example the adaptive subdivision rule performed better than the standard bisection. Also, it is worth noting that if the problem were written and solved as a problem (QP) (with objective function in the form $\langle x, Qx\rangle$), then the algorithm would work in dimension 12 rather than 6 and would require a much larger number of iterations (2752 iterations, with maximally 514 active nodes), and a much longer computation time (62.01 sec).

Example 6. (minimization) $n = 10,\ m = 7,\ A \in R^{15\times 10}$.

$$
\begin{aligned}
c^0 &= (0.0,\ 0.0,\ 0.0,\ 0.0,\ 0.0,\ 0.0,\ 0.0,\ 0.0,\ 0.0,\ 0.0), && r_0 = 0.0\\
c^1 &= (-0.7,\ 0.0,\ 0.4,\ 0.7,\ 0.4, -0.1, -0.4,\ 0.9, -0.1,\ 0.0), && r_1 = 19.3\\
d^1 &= (0.4, -0.6,\ 0.0,\ 0.4, -0.1, -0.3, -0.1,\ 0.6,\ 0.7,\ 0.1), && s_1 = 5.4\\
c^2 &= (0.3,\ 0.3,\ 0.1, -0.3, -0.3,\ 0.1, -0.1,\ 0.0, -0.1, -0.5), && r_2 = 25.0\\
d^2 &= (0.5, -0.4, -0.4, -0.1, -0.3,\ 0.4, -0.1,\ 0.2, -0.1, -0.4), && s_2 = 34.7\\
c^3 &= (0.3,\ 0.3,\ 0.0, -0.1,\ 0.1, -0.1,\ 0.6,\ 0.1,\ 0.4, -0.1), && r_3 = -1.0\\
d^3 &= (0.2, -0.2,\ 0.4,\ 0.6, -0.6, -0.5,\ 0.2,\ 0.5, -0.6, -0.5), && s_3 = 43.4\\
c^4 &= (0.6,\ 0.3,\ 0.4, -0.3, -0.2,\ 0.8,\ 0.1,\ 0.2, -0.1, -0.1), && r_4 = 12.8\\
d^4 &= (0.2, -0.4, -0.5,\ 0.1,\ 0.9,\ 0.0,\ 0.4, -0.9, -0.5, -0.2), && s_4 = 19.3\\
c^5 &= (0.0,\ 0.5, -0.2, -0.1,\ 0.2,\ 0.0,\ 0.3,\ 0.7, -0.8,\ 0.5), && r_5 = 6.2\\
d^5 &= (0.5, -0.1,\ 0.6,\ 0.8,\ 0.6,\ 0.0, -0.7, -0.5, -0.1, -0.5), && s_5 = 51.7\\
c^6 &= (0.2, -0.5, -0.4, -0.2, -0.2,\ 0.7, -0.9, -0.2,\ 0.3,\ 0.5), && r_6 = 26.6\\
d^6 &= (0.5, -0.2,\ 0.0,\ 0.1, -0.3,\ 0.2, -0.1,\ 0.3,\ 0.0,\ 0.3), && s_6 = 7.5\\
c^7 &= (-0.5,\ 0.0,\ 0.1,\ 0.4,\ 0.4,\ 0.9,\ 0.6,\ 0.0,\ 0.2,\ 0.2), && r_7 = -0.6\\
d^7 &= (-0.5,\ 0.4,\ 0.4, -0.3,\ 0.1,\ 0.3, -0.7, -0.2,\ 0.2,\ 0.1), && s_7 = 12.8
\end{aligned}
$$

$$
\begin{aligned}
h = (&161.1, -65.6,\ 45.7,\ 35.1,\ 3.5, -18.8, -46.4, -37.7, -59.4,\\
&-117.0,\ 24.8,\ 1.5,\ 55.6, -124.8, -27.2)
\end{aligned}
$$

$$
A = \begin{bmatrix}
4.9 & 3.5 & 3.7 & 0.7 & 2.1 & 1.9 & -2.3 & 2.4 & 3.1 & 3.3\\
1.6 & 1.5 & -0.4 & 4.8 & -4.7 & -0.1 & -1.4 & -2.9 & -2.7 & -4.7\\
-1.0 & -1.0 & -2.7 & -0.6 & 2.9 & -0.5 & -2.5 & 4.8 & 0.8 & 3.5\\
-0.7 & -0.3 & -0.8 & -1.9 & 4.3 & 1.1 & -2.0 & -2.7 & 4.4 & -1.4\\
4.4 & -4.9 & 2.7 & 1.4 & -0.3 & 2.0 & 3.4 & -0.2 & -0.4 & -3.0\\
2.6 & 1.5 & 2.2 & -1.8 & -2.7 & 0.4 & 0.3 & -0.9 & 2.5 & -4.1\\
-0.8 & -4.4 & -1.3 & -0.8 & 1.8 & 0.8 & 1.4 & -2.6 & 1.4 & -3.8\\
1.6 & 2.1 & -1.0 & -2.2 & 2.5 & -3.0 & -1.5 & 2.7 & -2.2 & -4.1\\
-4.5 & -2.2 & 3.1 & 2.8 & -2.1 & 4.5 & 3.1 & 2.6 & -4.4 & -4.0\\
-4.0 & 2.6 & -2.3 & 0.6 & -4.8 & -2.5 & 3.2 & 0.5 & -3.0 & -3.6\\
3.2 & -1.5 & -2.8 & 2.0 & 1.6 & 3.8 & -1.6 & 0.9 & 2.6 & -3.9\\
0.6 & 0.1 & 1.1 & 3.2 & -3.8 & -3.2 & 1.7 & 2.6 & 4.0 & -4.4\\
-4.8 & 1.8 & 3.7 & 2.5 & -3.4 & 0.2 & -3.5 & 3.0 & 1.1 & 4.4\\
2.0 & 1.3 & -3.9 & -3.9 & -1.1 & -4.3 & 2.5 & -1.2 & -4.7 & -2.4\\
1.9 & -3.6 & 4.3 & -0.9 & -0.1 & -1.0 & 0.1 & 0.1 & 0.0 & -4.1
\end{bmatrix}
$$

Computational results (tolerance $\varepsilon = 0.001$) :
Optimal solution:

$$x = (0.000000,\ 5.015611,\ 2.479073,\ 0.000000,\ 4.209541,\\ 5.717726,\ 35.160715,\ 0.000000,\ 28.710187,\ 32.283983);$$

Optimal value: 1181.273353;
Computation time: 63.55 sec;
Optimal solution found at iteration 1 and confirmed at iteration 2744;
Maximal number of active nodes: 381.

Example 7. ($n = m = 20$)

$$\min\left\{F(x) | \sum_{i=1}^{20} x_i = 23,\ \ 0 \le x_i \le 23\ (i = 1, \dots, 20)\right\} \tag{15}$$

where

$$F(x) = \langle x, Qx\rangle,\ \ Q = H + \operatorname{diag}(a),$$
$$a = (3, 2, 10, 6, 4, 1, 10, 10, 10, 7, 3, 3, 2, 3, 2, 6, 10, 5, 2, 9)$$

$$H = \begin{bmatrix}
0 & 9 & 4 & 6 & 9 & 4 & 5 & 4 & 6 & 6 & 6 & 7 & 3 & 6 & 5 & 4 & 3 & 6 & 6 & 2 \\
9 & 0 & 5 & 0 & 1 & 5 & 1 & 1 & 7 & 8 & 4 & 9 & 7 & 4 & 7 & 4 & 5 & 2 & 1 & 5 \\
4 & 5 & 0 & 7 & 7 & 4 & 4 & 0 & 6 & 6 & 4 & 7 & 3 & 0 & 7 & 9 & 0 & 6 & 7 & 4 \\
6 & 0 & 7 & 0 & 3 & 3 & 5 & 6 & 0 & 5 & 7 & 4 & 9 & 6 & 6 & 3 & 0 & 6 & 7 & 5 \\
9 & 1 & 7 & 3 & 0 & 1 & 1 & 8 & 4 & 9 & 9 & 3 & 4 & 2 & 1 & 1 & 9 & 9 & 5 & 9 \\
4 & 5 & 4 & 3 & 1 & 0 & 7 & 2 & 4 & 6 & 2 & 9 & 7 & 1 & 0 & 8 & 9 & 6 & 6 & 9 \\
5 & 1 & 4 & 5 & 1 & 7 & 0 & 8 & 7 & 4 & 5 & 3 & 2 & 9 & 9 & 6 & 4 & 8 & 1 & 1 \\
4 & 1 & 0 & 6 & 8 & 2 & 8 & 0 & 4 & 0 & 0 & 7 & 0 & 1 & 6 & 1 & 0 & 1 & 9 & 1 \\
6 & 7 & 6 & 0 & 4 & 4 & 7 & 4 & 0 & 6 & 4 & 4 & 7 & 6 & 0 & 9 & 5 & 0 & 6 & 4 \\
6 & 8 & 6 & 5 & 9 & 6 & 4 & 0 & 6 & 0 & 4 & 8 & 9 & 5 & 1 & 6 & 5 & 2 & 3 & 9 \\
6 & 4 & 4 & 7 & 9 & 2 & 5 & 0 & 4 & 4 & 0 & 1 & 4 & 1 & 1 & 8 & 8 & 7 & 9 & 5 \\
7 & 9 & 7 & 4 & 3 & 9 & 3 & 7 & 4 & 8 & 1 & 0 & 8 & 4 & 5 & 4 & 3 & 1 & 9 & 3 \\
3 & 7 & 3 & 9 & 4 & 7 & 2 & 0 & 7 & 9 & 4 & 8 & 0 & 7 & 4 & 0 & 0 & 2 & 0 & 4 \\
6 & 4 & 0 & 6 & 2 & 1 & 9 & 1 & 6 & 5 & 1 & 4 & 7 & 0 & 3 & 0 & 3 & 7 & 7 & 0 \\
5 & 7 & 7 & 6 & 1 & 0 & 9 & 6 & 0 & 1 & 1 & 5 & 4 & 3 & 0 & 9 & 2 & 5 & 3 & 7 \\
4 & 4 & 9 & 3 & 1 & 8 & 6 & 1 & 9 & 6 & 8 & 4 & 0 & 0 & 9 & 0 & 2 & 3 & 7 & 1 \\
3 & 5 & 0 & 0 & 9 & 9 & 4 & 0 & 5 & 5 & 8 & 3 & 0 & 3 & 2 & 2 & 0 & 2 & 7 & 0 \\
6 & 2 & 6 & 6 & 9 & 6 & 8 & 1 & 0 & 2 & 7 & 1 & 2 & 7 & 5 & 3 & 2 & 0 & 1 & 1 \\
6 & 1 & 7 & 7 & 5 & 6 & 1 & 9 & 6 & 3 & 9 & 9 & 0 & 7 & 3 & 7 & 7 & 1 & 0 & 4 \\
2 & 5 & 4 & 5 & 9 & 9 & 1 & 1 & 4 & 9 & 5 & 3 & 4 & 0 & 7 & 1 & 0 & 1 & 4 & 0
\end{bmatrix}$$

We solve this problem using the bounds indicated in (ii), Sec. 6.
With tolerance 0.01 the results are as follows:
Optimal solution:
(0, 0, 0, 0, 0, 15.333333, 0, 0, 0, 0, 0, 0, 0, 0, 7.666667, 0, 0, 0, 0, 0);
Optimal value: 352.666667;

Computational time: 65.424 sec.;
Optimal solution found at iteration 352 and confirmed at iteration 369;
Maximal number of active nodes: 48.

Remark 9. The computation of a lower bound $\alpha(M)$ for (16) deserves some additional comments, since the method used may present some general insterest. Noting that the elements of H are nonnegative and $x \geq p$ we have $Hx \geq Hp$, hence $\langle x-p, Hx-Hp\rangle \geq 0$, and so $\langle x, Hx\rangle \geq \langle p, Hx\rangle + \langle x, Hp\rangle - \langle p, Hp\rangle = 2\langle Hp, x\rangle - \langle Hp, p\rangle$. Therefore, a lower bound for $F(x)$ over $p \leq x \leq q$ can be taken to be

$$\min\Big\{\sum_{i=1}^{20} a_i x_i^2 + 2\langle Hp, x\rangle - \langle Hp, p\rangle \Big| \sum_{i=1}^{20} x_i = 23,\ p \leq x \leq q\Big\}. \tag{16}$$

The problem (16) is a simple convex quadratic program and can be efficiently solved by many currently available algorithms. Here we choose to solve this convex program by the now almost forgotten Frank-Wolfe (FW) method. The reason of this choice in the present case is that, aside from being very simple, this method has the advantage of reducing the convex program (16) to a sequence of extremely simple knapsack problems. However, FW method is known to be numerically instable near the optimum. Since we need only a good lower bound and not necessarily the exact minimum of (16), to get round this difficulty, we simply stop the FW procedure at some sufficiently advanced iteration k. Let $f(x) = \sum_{i=1}^{20} a_i x_i^2 + 2\langle Hp, x\rangle - \langle Hp, p\rangle$, $D_{[p,q]} = \{x|\ \sum_{i=1}^{20} x_i = 23,\ p \leq x \leq q\}$, $\nabla f(x) = 2(\mathrm{diag}(a)x + Hp)$. If the last obtained solution for (16) is x^k with $t_k = \min\{\langle \nabla f(x^k), x - x^k\rangle |\ x \in D_{[p,q]}\}$ then $f(x^k) + t_k \leq \min\{f(x)|\ x \in D_{[p,q]}\} \leq f(x^k)$, so we can take $\alpha(M) = f(x^k) + t_k$.

To describe the FW procedure, assume $\sum_{i=1}^{20} p_i \leq 23 \leq \sum_{i=1}^{20} q_i$ (otherwise $D_{[p,q]} = \emptyset$, i.e. $\alpha(M) = +\infty$). Let $\theta > 0$ be user supplied (typically $0.001 \leq \theta \leq 0.1$).

FW Procedure

0. Start with a feasible solution $x^0 \in D_{[p,q]}$. Set $k = 0$.

1. Compute $y^k = \mathrm{argmin}\{2\langle \mathrm{diag}(a)x^k + Hp, x - x^k\rangle |\ x \in D_{[p,q]}\}$ (by the knapsack subroutine). If $t_k = 2\langle\ \mathrm{diag}(a)x^k + Hp, y^k - x^k\rangle \geq -\theta$, then stop: $\alpha(M) = f(x^k) + t_k$. Otherwise, go to **2**.

2. Compute $r_k = -t_k / \langle 2(y^k - x^k),\ \mathrm{diag}\,(a)(y^k - x^k)\rangle$ (this is the value of r where the derivate of the univariate function $\varphi(r) := f(x^k) + r(y^k - x^k)$ vanishes). If $r_k \leq 1$ set $x^{k+1} = x^k + r_k(y^k - x^k)$, otherwise set $x^{k+1} = y^k$. With $k \leftarrow k+1$ go back to **1**.

Knapsack Subroutine (for finding $\min\{2\langle\ \mathrm{diag}\,(a)x^k + Hp, x - x^k\rangle |\ x \in D_{[p,q]}\}$): Sort the components of $2(\ \mathrm{diag}\,(a)x^k + Hp)$ in increasing order: $c_{i_1} \leq c_{i_2} \leq \cdots \leq c_{i_{20}}$. Set $j = 1, y^k = p$. If $\sum_{i=1}^{20} y_i^k = 23$, stop. Otherwise, reset $y_{i_j}^k = \min\{q_{i_j}, 23 - \sum_{i=1}^{20} y_i^k + p_{i_j}\}$. Repeat for $j \leftarrow j+1$.

8. Conclusion

We have presented an approach to nonconvex quadratic optimization under linear constraints, based on analyzing and exploiting monotonicity rather than convexity properties. As usual in nonconvex optimization the branch and bound strategy is best suitable for high rank problems, typically for problems of rank $m > 5$ in the present state of knowledge. The polyblock approximation developed in [18] can then be used for computing reasonably efficient bounds, especially when other methods fail or may be difficult to apply. For problems where monotonicity is combined with other structures as in the last example, a proper reformulation may sometimes help to improve the bounds and speed up the convergence drastically. As demonstrated by the numerical examples, quite often the global optimum can be found very fast, but usually much longer time is needed to confirm it. Furthermore, since no global optimization method can pretend to be uniformly best for all problems, and since, on the other hand, a given problem can be most efficiently solved only by exploiting its mathematical structure, in practice the best results are usually obtained by some hybrid method combining different solution approaches in a clever way.

References

1. H. P.Benson and G. M. Boger, Outcome-space cutting plane algorithm for linear multiplicative programming, *J. Optim. Theory Appl.* **104** (2000) 301–322.
2. J. E. Falk and S. W. Palocsay, *Optimizing the Sum of Linear Fractional Functions*, in *Global Optimization* C. Floudas and P. Pardalos (eds.), Princeton University Press, 1992, 221–258.
3. R. Horst and H. Tuy, *Global Optimization - Deterministic Approaches*, 3rd ed., Springer - Verlag, Berlin - New York, 1996.
4. C. A. Floudas et al.,*Handbook of Test Problems in Local and Global Optimization*, Kluwer Academic Publishers, Dordrecht - Boston - London, 1999.
5. H. Konno and T. Kuno, Generalized linear multiplicative and fractional programming, *Annals of Operations Research* **25** (1990) 147–162.
6. H. Konno and T. Kuno, Linear multiplicative programming, *Math. Program.* **56** (1992) 51–64.
7. H. Konno, P. T. Thach, and H. Tuy, *Optimization on Low Rank Nonconvex Structures*, Kluwer Academic Publishers, Dordrecht - Boston - London, 1997.
8. H. Konno, Y. Yajima and T. Matsui, Parametric simplex algorithms for solving a special class of nonconvex minimization problems, *J. Global Optim.* **1** (1991) 65–81.
9. H. Konno and Y. Yamashita, Minimization of the sum and the product of several linear fractional functions', Tokyo Institute of Technology, Technical Report, Department of IE & Management, 1997.
10. T. Matsui, NP-hardness of linear multiplicative programming and related problems, *J. Global Optim.* **9** (1996) 113–119.
11. T. S. Motzkin and E. G. Strauss, Maxima for graphs and a new proof of a theorem of Turan, *Canad. J. Math.* **17** (1965) 533–540.

12. N. D. Nghia, Reverse polyblock approximation algorithm for generalized multiplicative programming: Implementation and computational experience, Preprint, Institute of Mathematics, Hanoi, 2000.
13. P. M. Pardalos and J. B. Rosen, *Constrained Global Optimization Algorithms: Algorithms and Applications*, Lecture Notes in Computater Science, 268, Springer, 1987.
14. P. M. Pardalos and S. A. Vavasis, Quadratic programming with one negative eigenvalue is NP-Hard, *J. Global Optim.* **1** (1991) 15–22.
15. S. Schaible and C. Sodini, Finite algorithm for generalized linear multiplicative programming, *J. Optim. Theory Appl.* **87** (1995) 441–455.
16. H. Tuy, *Convex Analysis and Global Optimization*, Kluwer Academic Publishers, Dordrecht - Boston - London, 1998.
17. H. Tuy, Normal Sets, Polyblocks and monotonic optimization, *Vietnam J. Math.* **27** (1999) 277–300.
18. H. Tuy, Monotonic optimization: Problems and solution approaches, *SIAM J. Optim.* **11** (2000) 464–494.
19. H. Tuy, Convexity and monotonicity in global optimization, Proceedings, International Conference on *Advances in Convex Analysis and Global Optimization*, honoring the memory of Caratheodory, Pythagorian, Samos, 5-9 June 2000.
20. H. Tuy, P. T. Thach, and H. Konno, Optimization of polynomial fractional functions, preprint, Institute of Mathematics, Hanoi, 2001 (submitted).