Linear Programs with an Additional Rank Two Reverse Convex Constraint

ULRICH PFERSCHY[1] and HOANG TUY[2]
[1]*Institute of Mathematics, TU Graz, Graz, Austria;* [2]*Institute of Mathematics, Hanoi, Vietnam.*

(Received: 13 April 1992; accepted: 24 March 1993)

Abstract. An efficient algorithm is developed for solving linear programs with an additional reverse convex constraint having a special structure. Computational results are presented and discussed.

Key words. Rank two reverse convex constraints, parametric method.

1. Introduction

A typical problem of global optimization that has attracted a great deal of attention from researchers in recent years is the reverse convex programming problem:

$$\begin{aligned}&\text{minimize } cx\\ &\text{s.t. } x \in D \setminus E\,,\end{aligned}$$

where D is a polytope, E is an open convex set in $\mathbb{R}^n$ and $c \in \mathbb{R}^n$ (see, e.g. Tuy [15], Horst & Tuy [3] and the references therein). As it is known, when E is the unit ball, checking the feasibility of the constraint $x \in D\setminus E$ is a set containment problem, which has been shown to be NP-hard [1]. No wonder that the general purpose algorithms that have been developed so far for the above problem can usually handle only instances of a very limited size.

In this paper we are concerned with a special variant of the reverse convex programming problem where the set E has a particular structure which allows large scale problems to become tractable. Specifically, setting $E = \{x \mid f(x) \leq 1\}$, we consider the problem

$$\begin{aligned}(P)\quad &\text{minimize } cx\\ &\text{s.t. } x \in D &(1)\\ &f(x) \leq 1\,, &(2)\end{aligned}$$

where D is a polytope and $f: \mathbb{R}^n \to \mathbb{R}$ is a continuous function, quasiconcave on D and possessing the following *rank two property* ([14], [16]) on D:

(•) There exist two linearly independent vectors $c^1, c^2 \in \mathbb{R}^n$ such that $\forall x \in D$:

$$z \in \mathbb{R}^n, c^i z \geq 0 (i = 1, 2) \Rightarrow f(x + z) \geq f(x)\,.$$

Of particular interest is the case when $f(x)$ is the product of two affine functions, e.g. $f(x) = (c^1x + d_1)(c^2x + d_2)$. Condition (•) obviously holds for functions of this form and if $(c^1x + d_1) > 0$ and $(c^2x + d_2) > 0$, $\forall x \in D$, then, as can be checked easily, $\log f(x)$ is concave, hence $f(x)$ is quasiconcave on D. Functions that are the product of two affine functions occur in a number of real world problems. Among the applications reported in different fields: microeconomics, transportation, investment optimization, VLSI design, etc. let us mention some economic models presented in Henderson & Quandt [2], certain problems of optimizing floor plans for electronic packages discussed in Maling *et al.* [9] and problems arising from bond portfolios optimization discussed in Konno & Inori [5].

Aside from products of affine functions, the class of quasiconcave functions with rank two property includes all functions of the form $f(x) = g(x_1) + \Sigma_{i=2}^n h_i x_i$, where $g(\cdot)$ is a concave nondecreasing functions of one variable (condition (•) is fulfilled with $c^1 = (1, 0, \ldots, 0)$, $c^2 = (0, h_2, \ldots, h_n)$). In particular, it includes concave quadratic functions with just one eigenvalue (considered on an appropriate domain). To show one last nontrivial example let us consider a linear two level optimization problem:

$$\begin{aligned}
&\text{minimize } h_1u + h_2v\\
&\text{s.t. } A_1u + B_1v \leqslant g_1\,,\\
&u \geqslant 0, (u, v) \in \mathbb{R}^p \times \mathbb{R}^q \text{ and}\\
&v \in \operatorname{argmax}\{d_2v \mid A_2u + B_2v \leqslant g_2\,, v \geqslant 0\}\,,
\end{aligned}$$

(see [16] and the references therein), where the matrices A_2 and B_2 have a single row. If we define $\phi(u) = \max\{d_2v \mid A_2u + B_2v \leqslant g_2, v \geqslant 0\}$, $x = (u, v)$ and rewrite this problem as a reverse convex program

$$\begin{aligned}
&\text{minimize } h_1u + h_2v\\
&\text{s.t. } A_1u + B_1v \leqslant g_1\,, A_2u + B_2v \leqslant g_2\,,\\
&u \geqslant 0, v \geqslant 0, \text{and}\\
&\phi(u) - d_2v \leqslant 0\,,
\end{aligned}$$

then it can be seen easily that this is a special case of (P) because the function $f(x) = \phi(u) - d_2v + 1$ is concave and satisfies condition (•) with $c^1 = (0, -d_2)$, $c^2 = (-A_2, 0)$. Thus a variety of problems of relevant practical interest can be cast into the form (P).

A problem similar to but more general than (P) has been studied recently by Thach *et al.* [14]. In the present context, the method of these authors reduces (P) to a quasiconcave minimization problem in $\mathbb{R}^2$, which is then solved by an outer approximation method. The resulting algorithm is quite practical, since it operates in two dimensions.

The aim of the present paper is to propose an efficient finite algorithm for solving problem (P), different from the one of Thach *et al.* Computational experiments with this algorithm have given very encouraging results. Basically, our approach consists in solving (P) by a sequence of iterations, involving each two phases as in the method of Tuy & Thuong [18] (see also [3]) for linear programs with an additional reverse convex constraint. In Phase I we use the same local moves as in the latter method to improve the current best solution, while in Phase II, to "transcend" the current best solution we use a parametric algorithm for rank two quasiconcave minimization developed in a recent work of Tuy & Tam [17]. To take full advantage of the specific structure of the problem, we will also make several improvements while carrying out each phase.

The paper is organized as follows. After the Introduction, we formulate in Section 2 a subproblem (to be called master problem) which has to be solved repeatedly in our approach. We also describe the procedure to be used for solving this master problem. In Section 3 we present the solution method for problem (P). In Section 4 we give the detailed algorithm and discuss its convergence as well as other related issues. An illustrative example is given in Section 5. Finally in Section 6 we present the computational results and compare our algorithm with the method in [14].

2. Master Problem and Main Subroutine

Let $\varepsilon > 0$ be a given tolerance. We will consider problem (P) solved if a feasible solution $\bar{x}$ of (P) has been found such that

$$c\bar{x} \leqslant cx + \varepsilon$$

for all feasible solutions x. For short, we will call such a feasible solution an *ε-optimal solution* of problem (P).

A basic question which arises when considering a global optimization problem such as (P) is the following:

Let $\bar{x}$ be a feasible solution. How to recognize whether $\bar{x}$ is an ε-optimal solution and if it is not, how to find a better feasible solution than $\bar{x}$?

If we denote $M = D \cap \{x \mid cx \leqslant c\bar{x} - \varepsilon\}$ then clearly the inequality $\min\{f(x) \mid x \in M\} > 1$ implies that $\bar{x}$ is ε-optimal while any $z \in M$ with $f(z) \leqslant 1$ will give a feasible solution such that $cz < c\bar{x}$. Thus, for our problem (P) an answer to the above question will be obtained by solving the following subproblem:

$$(Q) \quad \text{minimize } f(x) \quad \text{s.t. } x \in M\,.$$

We will refer to this subproblem as the *master problem*. As will be seen shortly, any procedure for solving this master problem can be used as the main subroutine in a method for solving (P).

When $f(x)$ is the product of two linear functions, problems of the form (Q) have been studied by Yajima & Konno [19], Konno *et al.* [7], Suzuki *et al.* [13], Pardalos [10], and especially by Konno & Kuno [6]. In the general case of an arbitrary rank two quasiconcave function $f(x)$, an efficient solution procedure for (Q) has been developed recently by Tuy & Tam [17]. In the sequel we will use the latter procedure to solve our master problem.

Let us state the theorem that serves as the foundation for the procedure of Tuy & Tam:

THEOREM 1. *Let c^1, c^2 be the two vectors in the rank two condition $(\bullet)$ and for each $\alpha \in [0, 1]$ let x_α be an optimal solution of the linear parametric program*

$$(R_\alpha) \quad \textit{minimize } \alpha c^1x + (1-\alpha)c^2x \quad \textit{s.t. } x \in M .$$

Then $\min\{f(x_\alpha) \mid 0 \leq \alpha \leq 1\} = \min\{f(x) \mid x \in M\}$.

Proof. See [17], where this theorem was derived from a polyhedral annexation algorithm for solving (Q). A direct proof has been given in [4]. In the important special case when $f(x) = (c^1x)(c^2x)$ an elementary proof can also be found in [11]. □

As a straightforward consequence of this Theorem we obtain the following *parametric procedure* for solving the master problem (Q):

Let $\varphi(\alpha)$ be the optimal value in (R_α) (as is known from parametric programming theory, $\varphi(\alpha)$ is a convex piecewise affine function). Compute the sequence of break points of $\varphi(\alpha)$:

$$0 = \alpha_1 < \cdots < \alpha_{j-1} < \alpha_j < \cdots < \alpha_{N+1} = 1 \tag{3}$$

along with, for each $j = 1, \ldots, N$, a vertex x^j of M which is a basic optimal solution of (R_α) for all $\alpha \in (\alpha_j, \alpha_{j+1}]$. Then an optimal solution of (Q) is given by the point x^{j^*} such that

$$f(x^{j^*}) = \min\{f(x^j) \mid j = 1, \ldots, N\} .$$

Thus solving the master problem (Q) reduces to computing the sequence of break points (3) together with the associated sequence $x^1, \ldots, x^{N+1}$ and the values $f(x^1), \ldots, f(x^{N+1})$. In the next section we will present a solution method for problem (P) using this parametric procedure as main subroutine.

3. Solution Method

Let u be an optimal solution of the linear program

$$\min\{cx \mid x \in D\}$$

which is obtained from (P) by omitting the reverse convex constraint (2). If u satisfies the reverse convex constraint as well, then it obviously solves (P), i.e. the reverse convex constraint is not essential (can be omitted). Therefore, it is natural to assume that every optimal solution u of the above linear program satisfies $f(u) > 1$, or equivalently, that

$$\min\{cx \mid x \in D\} < \min\{cx \mid x \in D, f(x) \leq 1\} . \tag{4}$$

To find an *ε-optimal solution* of (P), we propose to proceed in a number of iterations, consisting each of two phases. In Phase I we perform some inexpensive "local" moves in order to improve the best feasible solution available to us at this stage (for short this *current best feasible solution* will be denoted by *CBS*). When local moves are no longer possible, we pass to Phase II, the main task of which is to test *CBS* for ε-optimality and to "transcend" it i.e. to find a better feasible solution than *CBS* when it is not yet ε-optimal. In more detail each phase proceeds as follows.

PHASE I. This phase is entered with the knowledge of a feasible solution z^0 which is a vertex of the current polytope M (at the beginning, $M = D$). Consider the linear problem

$$(L)\quad \begin{aligned} &\text{minimize } cx \\ &\text{s.t. } x \in M . \end{aligned}$$

Since by assumption (4) $cz^0 > \min\{cx \mid x \in D\}$, z^0 is not optimal for (L). Hence, starting from z^0 we can perform simplex pivot operations to move from z^0 to a neighbouring vertex z^1 such that $cz^1 < cz^0$, then from z^1 to z^2 and so on. In this way we will obtain a sequence of better and better vertices z^0, z^1, $z^2, \ldots$ of M.

Eventually we must arrive at some vertex z^i having a neighbour z^{i+1} with $cz^{i+1} < cz^i$ but lying on the other side of the surface $S := \{x \mid f(x) = 1\}$. As soon as the latter situation occurs we compute the intersection point $\bar{x}$ of the surface S with the edge $[z^i, z^{i+1}]$. Then $\bar{x}$ will give a better feasible solution than z^0 (except in rare degenerate cases where z^0 already is such an intersection point, i.e. $i = 0$). We then set $CBS \leftarrow \bar{x}$ and enter Phase II.

PHASE II. In this phase we text $\bar{x} = CBS$ for ε-optimality and "transcend" it if necessary. For that purpose we solve the master problem (Q) with M redefined as $M = D \cap \{x \mid cx \leq c\bar{x} - \varepsilon\}$. This means we use ε to exclude *CBS* from the further computation. According to the procedure described in Section 2, we consider the associated linear parametric program

$$\begin{aligned} (R_\alpha)\quad & \text{minimize } \alpha c^1 x + (1-\alpha)c^2 x \\ & \text{s.t. } x \in D \\ & cx \leqslant c\bar{x} - \varepsilon\,. \end{aligned}$$

Starting from $\alpha_1 = 0$ we compute the successive break points $\alpha_i \in [0, 1]$ $i = 1, 2, \ldots)$ of (R_α) along with the basic optimal solutions x^i associated with the intervals $(\alpha_i, \alpha_{i+1}]$ and the values $f(x^j)$. There are two possibilities:

1. $f(x^j) \leqslant 1$ for a certain break point α_j. As soon as this occurs, we stop the parametric procedure at α_j: x^j is a better feasible solution than $\bar{x}$, so we go to Phase I of the next iteration with $z^0 = x^j$.
2. $f(x^i) > 1$ for every break point α_i (i.e. the parametric procedure is continued until the break point $\alpha_{N+1} = 1$). Then, by Theorem 1, $f(x) > 1$ for all $x \in D$ such that $cx \leqslant c\bar{x} - \varepsilon$, hence $\bar{x}$ is an ε-optimal solution and we terminate.

Note that in this way, if a further iteration is needed, then Phase I of the next iteration will always be entered with a feasible solution z^0 which is a vertex of M.

REMARKS.

(*i*) *Starting Vertex.*

One point that remains to be clarified in the above method is how to define the feasible solution z^0 for Phase I of the first iteration. In certain cases such a z^0 may be readily available. Otherwise, we carry out a Phase II with $M = D$, i.e. we compute the break points of the linear parametric program $\min\{\alpha c^1 x + (1-\alpha)$ $c^2 x \mid x \in D\}$ together with the associated vertices of D and the values of $f(x)$ at these vertices: either a vertex of D is obtained with a value of $f(x) \leqslant 1$, in which case this vertex is used as z^0 for the first iteration; or no such vertex is found, in which case problem (P) is infeasible.

(*ii*) *Selection of* ε.

We use ε only to transcend *CBS*, hence ε can be chosen very small as its choice hardly effects the performance of the method. Only in some rare cases the number of iterations can slightly increase.

(*iii*) *Improvement of the parametric procedure.*

Let us call *lastalpha* the last break point of the current linear parametric program (R_α) reached at the completion of Phase II. Practical experiments show that the same sequence of break points $\{\alpha_0 = 0, \ldots, \textit{lastalpha}\}$ in an iteration k is found again very often in iteration $k+1$, in which usually further break points greater than *lastalpha* are calculated until a new better feasible solution than *CBS* is found (naturally the associated basic optimal solutions x^j may differ because the polytope M is not the same).

To take advantage of this phenomenon, instead of computing again all the break points $\{\alpha_0 = 0, \ldots, \mathit{lastalpha}\}$ one enters Phase II with $\alpha_0 = \mathit{lastalpha}$ and proceeds in the same way as before. If $\alpha_{N+1} = 1$ is reached (without discovering any vertex which is a better feasible solution than *CBS*) then the break points between 0 and *lastalpha* together with the associated vertices of M and the values of $f(x)$ at these vertices have to be computed to make sure that all the break points have been scanned. To do this we go back to the situation at *lastalpha* and generate the remaining break points moving backwards from *lastalpha* to 0. This requires storing the matrix of coefficients and all arrays defining the current basic solution after the first minimization program of Phase II in every iteration.

In most examples we have solved this improvement proved to be remarkable, sometimes very important.

4. The Algorithm

From the above we can now state the following detailed algorithm, assuming condition (4).

Algorithm:

```
Initial Step:
  Set $M := D$;
  $\mathit{lastalpha} := 0$;
  If a feasible vertex $z^0$ of $D$ is available then
    goto Step 1;
  Otherwise
    goto Step 2;
Step 1:
  1.1.: Set $i := 1$;
  1.2.: Perform a pivot operation of the simplex method for minimizing $cx$ over
        $M$ to obtain a new basic solution $z^i$.
  1.3.: If $f(z^i) \geq 1$ then
          $\bar{x} := \operatorname{argmin}\{cx \mid x \in [z^{i-1}, z^i];\ f(x) = 1\}$;          (5)
          $CBS := \bar{x}$;
          $M := M \cap \{x \mid cx \leq c(CBS) - \varepsilon\}$;
          goto Step 2;
        If $f(z^i) < 1$ then
          increment $i$;
          goto 1.2.;
Step 2:
  2.1.: Set $\alpha_0 := \mathit{lastalpha}$;
        move := up;
        $x^0 := \operatorname{argmin}\{\mathit{lastalpha}\ c^1x + (1 - \mathit{lastalpha})c^2x \mid x \in M\}$;
        Set $j := 0$;
        If $f(x^j) > 1$ then **store**;
          goto 2.3.;
  2.2.: If $f(x^j) \leqslant 1$ then
          $z^0 := x^j$;
          *lastalpha* $:= \alpha_j$;
          goto Step 1;
  2.3.: If *move* = up then
          $\alpha_{j+1} := \max\{\alpha \mid \alpha \in (a_j, 1]$ s.t.: $x^j$ is optimal for $(R_\alpha)\}$;
          If $\alpha_{j+1} = 1$ then
            **restore**;
            $x^j := x^0$;
            $\alpha_j :=$ *lastalpha*;
            *move* := down;
            goto 2.3.;
        Otherwise $\{\alpha_{j+1} < 1\}$
          Let $x^{j+1}$ be a basic optimal solution of $(R_\alpha)$
            for $\alpha = \alpha_{j+1} + \delta$; $\{\delta > 0$, arbitrary small$\}$
        Otherwise {*move* = down}
          $\alpha_{j+1} := \min\{\alpha \mid \alpha \in [0, \alpha_j)$ s.t.: $x^j$ is optimal for $(R_\alpha)\}$;
          If $\alpha_{j+1} = 0$ then **stop**.
            {*CBS* is an $\varepsilon$-optimal solution}
          Otherwise
            Let $x^{j+1}$ be a basic optimal solution of $(R_\alpha)$
              for $\alpha = \alpha_{j+1} - \delta$; $\{\delta > 0$, arbitrary small$\}$
        increment $j$;
        goto 2.2.;
```

THEOREM 2. *The above algorithm terminates after finitely many steps.*

Proof. Phase I is finite because it is only part of a simplex procedure and Phase II in the worst case involves computing all the break points of a linear parametric program. Therefore, each iteration is finite.

At the completion of Phase I in iteration k we have a point $\bar{x}^k$ which is at the intersection of an edge of D with the surface S (more precisely, $\bar{x}^k$ is the point of this intersection that has the smallest value of cx, see (5)). But $c\bar{x}^{k+1} \leqslant c\bar{x}^k - \varepsilon < c\bar{x}^k$. Therefore the sequence $\{\bar{x}^k\}$ contains no repetition. Since each edge can generate at most one point like $\bar{x}^k$ and the number of edges is finite, the finiteness of the algorithm follows. □

REMARKS.

1. The intersection $[z^{i-1}, z^i] \cap \{x \mid f(x) = 1\}$ (see (5) in Step 1.3) usually is a singleton but in certain cases may consist of two points or even of an entire segment. We always take $\bar{x}$ to be the point of this intersection that lies the farthest from z^{i-1}, so it is uniquely defined.

2. The above algorithm assumes condition (4). However, in practice we may not know whether or not this condition holds. In this case, we should begin with computing a basic optimal solution u of the linear program

   $$\text{minimize } cx \text{ s.t. } x \in D\ .$$

   If $f(u) \leq 1$ then u solves (P) and we terminate. If $f(u) > 1$, then every time a new feasible solution z better than the current *CBS* is obtained ($z = \bar{x}$ in Step 1.3 or $z = x^j$ in Step 2.2) we must check whether $cz = cu$, in which case z solves (P) and we terminate.

5. Illustrative Example

To illustrate how the method works, we consider an example in $\mathbb{R}^2$ which can be visualized easily:

$$\begin{aligned}&\text{minimize } cx\\ &\text{s.t. } Ax \leq b\\ &f(x) := (c^1x + d_1)(c^2x + d_2) \leq 1\ ,\end{aligned}$$

with the following data:

$$x \in \mathbb{R}^2; c = (-4, -1)\ ;$$

$$c^1 = (0.002, -0.02); c^2 = (0.01, 0.03); d_1 = 0.8; d_2 = 1.4\ ;$$

$$A = \begin{bmatrix} -2 & -5 \\ 0 & -1 \\ 3 & -5 \\ 2 & -1 \\ 2 & 1 \\ 3 & 5 \\ -2 & 7 \\ -5 & 3 \\ -1 & 0 \end{bmatrix}\ ; \quad b = \begin{bmatrix} 185 \\ 45 \\ 345 \\ 160 \\ 160 \\ 310 \\ 155 \\ 25 \\ 5 \end{bmatrix}\ .$$

As can be seen in Figure 1 the feasible domain is disconnected (it consists of two disjoint parts). Assumption (4) holds because $u := (80, 0) \in \operatorname{argmin}\{cx \mid x \in D\}$ and $f(u) = 2.112 > 1$.

To find a starting vertex we solve the problem $\{\text{minimize } cx \text{ s.t. } Ax \leq b\}$. Thereby we get the vertex $(-5, -35)$, which we denote as first *CBS*. The further execution of the algorithm yields the following values: (The first index represents the iteration number.)

1. Iteration:

Step 1:

$z^{10} := (-5, -35)$;

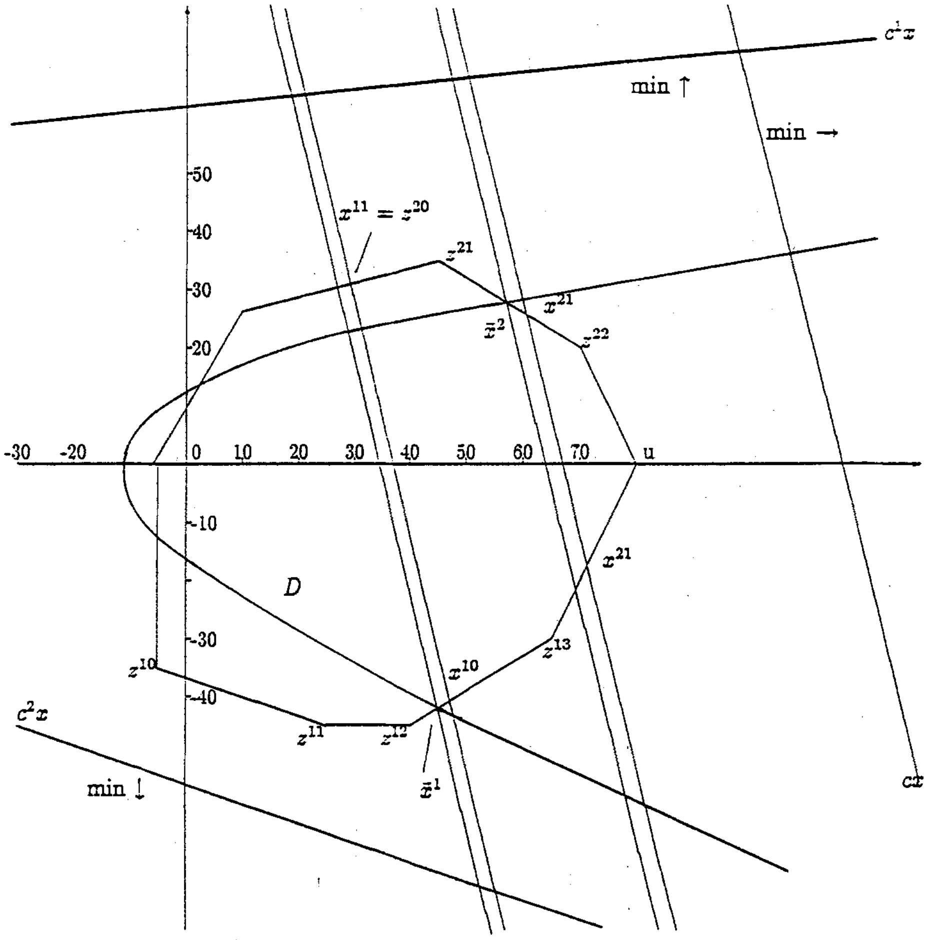

Fig. 1. The two iterations to solve the example.

$z^{11} := (25, -45);$
$z^{12} := (40, -45);$
$z^{13} := (65, -30);$ $\quad f(z^{13}) = 1.76 > 1;$
$CBS := \bar{x}^{-1} := (44.52, -42.29);$ $\quad c(CBS) = -135.77;$
$M := D \cap \{x \mid cx \leqslant -140.77\};$

Step 2:
$x^{10} := (45.6, -41.64);$
$\alpha^{11} := 0.57;$
$CBS := x^{11} := (27.68, 30.05);$ $\quad f(x^{11}) = 0.656 < 1;$

2. *Iteration:*

```
Step 1:
    $z^{20} := CBS$;
    $z^{21} := (45, 35)$;
    $z^{22} := (70, 20)$;                         $f(z^{22}) = 1.46 > 1$;
    $CBS := \bar{x}^2 := (56.9, 27.86)$;          $c(CBS) = -255.44$;
    $M := M \cap \{x \mid cx \leqslant -260.44\}$;
Step 2:
    $x^{20} := (58.36, 26.98)$;
    $\alpha^{21} := 1$; {restore}
    $\alpha^{21} := 0.57$;
    $x^{21} := (70.07, -19.85)$;
    $\alpha^{22} := 0$; stop.
```

Hence $CBS = (56.9,\ 27.86)$ is the global ε-optimal solution. For reasons of graphical representation we chose $\varepsilon = 5$ unusually large.

6. Computational Results and Comparison

The algorithm was tested on various problems of the form (P) where the polytope D was defined by

$$D := \{x \mid Ax \leqslant b, x \geqslant 0\},$$

and $f(x) = (c^1x + d_1)(c^2x + d_2)$. $A \in \mathbb{R}^{m \times n}$ was generated randomly between -50 and 50 such that $\mathcal{O} \in D$. In the same way c, c^1 and c^2 were generated, then d_1 and d_2 were chosen such that each function $c^ix + d_i$, $i = 1, 2$ had a positive sign over D. At least the reverse convex constraint was scaled such that assumption (A) held and the feasible area was not empty.

For each set of constraints five different objective functions c were chosen randomly. About 20 constraint sets were generated yielding 100 examples. The size of the problems varied between $n = 40$, $m = 40$ and $n = 100$, $m = 60$. We set ε to 0.0001. In practice very small values of ε can be used because we only have to make sure that at each iteration CBS is deleted from the feasible area. In our experiments variation of ε did not change the number of iterations.

The program was coded in Turbo Pascal and run on a 386 microcomputer with 20 MHZ.

It could be observed that the local improvement in Step 1 took place predominantly in the first few iterations where sometimes more than 100 pivot operations were executed, whereas after three or four iterations Step 1 tended very strongly to take only two pivots. This means that to find a vertex better than CBS (with respect to cx) we immediately have to cross the border of the set

$$T = \{x \mid (c^1x + d_1)(c^2x + d_2) = 1\} . \tag{6}$$

We also noted a moderate tendency of a growing number of iterations if the number of pivots in Step 1 in the first few iterations was rather small but this property could not be generalized.

With very few exceptions all examples showed that the effort spent in Step 2 was considerably larger than in Step 1.

The break points *lastalpha* computed in each iteration were distributed without any striking pattern but a noticeable tendency to form clusters, i.e. many break points in a small interval.

Most of the pivot operations performed in Step 2 occurred during the first few iterations. In the later iterations a new feasible solution for passing to the next break point was found frequently with a single pivot operation. As mentioned above Step 1 often computed just an intersection point adjacent to the starting vertex, hence in many iterations we got a total number of only three pivots which can be interpreted as a "zig-zaging" of the algorithm along the surface (6).

Naturally the total number of iterations varied widely with each objective function. The maximum number of iterations encountered was 63.

The running time of the program execution was proportional to the number of pivot operations with only a small amount of time necessary for the storage operations in Step 2.1.

A method for a larger class of problems also applicable to solve (P) is presented by Thach *et al.* [14]. The problem is reduced to a quasiconcave minimization problem in $\mathbb{R}^2$ which is solved by applying the outer approximation method (OAM). The calculation requires a number of iterations with two linear programs. This approach to solve (P) was implemented by Pferschy [12] and several experiments of the same form and in the same environment as described

Table I.

Problem Nr.	1	2	3	4	5
m	40	40	40	40	50
n	40	50	60	70	40
new method:					
av. iterations	8.2	13.4	13.8	20.2	13.4
av. pivots Step 1	23.4	52.8	89.4	63.8	48.0
av. pivots Step 2	36.4	88.2	89.2	128.8	71.4
av. total pivots	109.8	198.0	257.6	284.6	152.4
av. total time (sec)	5.7	12.5	19.1	24.5	9.6
(OAM) from [14]					
av. iterations	9.8	9.2	8.6	9.2	9.4
av. total pivots	1233.4	1950.8	2438.6	2506.2	1275.6
av. total time (sec)	57.6	110.7	161.8	193.1	72.2

above were performed. The size of the problems varied between $n = 30$, $m = 30$ and $n = 70$, $m = 60$. The computation of an ε-optimal solution for the same small ε as in our algorithm poses no problem in this case as we have a linear objective function.

We observed that the number of iterations varied only in a surprisingly small interval usually between 8 and 11 and was hardly dependent on the size of the problem but more closely related to the choice of η, the relaxation parameter of the reverse convex constraint. Hence the effort spent to solve the linear programs was most important for the total performance of the algorithm. The necessary calculation of a considerable number of linear programs caused this algorithm to take generally much longer than the method in this paper for all choices of η. For reasonable values of η the number of pivot operations of this (OAM) algorithm was about ten times higher. However it is applicable to more general problems.

Tables I and II show the performance of both methods on 10 selected examples. Beside the size of the problems at first the number of iterations for the new algorithm is shown followed by the number of all pivot operations in Step 1 and in Step 2, the sum of the pivot operations (including the computation of a starting vertex) and the total running time in seconds. Then the number of iterations and pivots and the total amount of time in seconds needed for the method in [14] are given. All values are taken in average over the five different objective functions for each constraint set. Additional tables, remarks and a sourcecode listing can be found in Pferschy [11].

REMARK. It was pointed out by an anonymous referee that a problem of the same class has been studied recently by Kuno *et al.* [8], who applied outer approximation to compute a lower bound of the optimal value with a not necessarily feasible solution.

Table II.

Problem Nr.	6	7	8	9	10
m	50	50	60	60	60
n	50	70	40	50	70
new method:					
av. iterations	17.0	18.0	13.8	7.8	12.8
av. pivots Step 1	72.2	83.4	50.2	67.2	96.6
av. pivots Step 2	81.0	120.2	67.8	47.6	161.2
av. total pivots	216.2	308.6	197.8	231.0	451.8
av. total time (sec)	16.7	32.1	13.5	19.5	54.7
(OAM) from [14]					
av. iterations	9.6	9.2	8.8	8.0	8.2
av. total pivots	1951.2	3649.0	1800.6	1945.2	3353.8
av. total time (sec)	137.0	340.1	121.1	163.9	379.8

Acknowledgement

We would like to thank Prof. R. E. Burkard for valuable suggestions and discussions.

References

1. R. M. Freund and J. N. Orlin (1985), On the complexity of four set containment problems, *Mathematical Programming* **33**, 139–145.
2. J. M. Henderson and R. E. Quandt (1971), *Microeconomic Theory*, McGraw-Hill, New York.
3. R. Horst and H. Tuy (1990), *Global Optimization*, Springer-Verlag, Berlin Heidelberg New York.
4. B. Klinz and H. Tuy (1993), Minimum concave-cost network flow problems with a single nonlinear arc cost, in Ding-Zhu Du and P. M. Pardalos (eds.), *Network Optimization Problems*, World Scientific, 125–143.
5. H. Konno and M. Inori (1988), Bond portfolio optimization by bilinear fractional programming, *Journal of the Operations Research Society Japan* **32**, 143–158.
6. H. Konno and T. Kuno (1992), Linear Multiplicative Programming, *Mathematical Programming*, to appear.
7. H. Konno, Y. Yajima and T. Matsui (1991), Parametric simplex algorithms for solving a special class of nonconvex minimization problems, *Journal of Global Optimization* **1**, 65–82.
8. T. Kuno, H. Konno and Y. Yamamoto (1991), Parametric successive underestimation method for convex programming problems with an additional convex multiplicative constraint, *Journal of Global Optimization* **1**, 267–286.
9. K. Maling, S. H. Mueller and W. R. Heller (1982), On finding most optimal rectangular package plans, *Proc. of the 19th Design Automation Conference*, 663–670.
10. P. M. Pardalos (1988), On the global minimization of the product of two linear functions over a polytope, Preprint, Computer Science Department, Pennsylvania State University.
11. U. Pferschy (1991), Mathematical programs with a two-dimensional reverse convex constraint, diploma thesis, Institute of Mathematics, University of Technology, Graz.
12. U. Pferschy (1991), Reverse convex programs dealing with the product of two affine functions, Numerical experiments, Institute of Mathematics, University of Technology, Graz.
13. S. Suzuki, P. T. Thach and T. Tanaka (1989), On methods for minimizing the product of two convex functions, Preprint, Institute of Mathematics, Hanoi.
14. P. T. Thach, R. E. Burkard and W. Oettli (1991), Mathematical programs with a two-dimensional reverse convex constraint, *Journal of Global Optimization* **1**, 145–154.
15. H. Tuy (1992), The complementary convex structure in global optimization, *Journal of Global Optimization* **2**, 21–40.
16. H. Tuy, A. Migdalas and P. Värbrand (1993), A global optimization approach for the linear two level program, *Journal of Global Optimization* **3**, 1–24.
17. H. Tuy and B. T. Tam (1992), An efficient solution method for the rank two quasiconcave minimization problems, *Optimization* **24**, 43–56.
18. H. Tuy and N. V. Thuong (1984), A finite algorithm for solving linear programs with an additional reverse convex constraint, *Lecture Notes in Economics and Mathematical Systems* **255**, 291–304, Springer.
19. Y. Yajima and H. Konno (1991), Efficient algorithms for solving rank two and rank three bilinear programming problems, *Journal of Global Optimization* **1**, 155–171.