Annals of Operations Research, 25 (1990) 19–42

A LEVEL SET ALGORITHM FOR A CLASS OF REVERSE CONVEX PROGRAMS

Sihem BEN SAAD

AT&T Bell Laboratories, Holmdel, NJ 07733, USA

Stephen E. JACOBSEN

Department of Electrical Engineering, University of California, Los Angeles, CA 90024, USA

Abstract

A new algorithm is presented for minimizing a linear function subject to a set of linear inequalities and one additional reverse convex constraint. The algorithm utilizes a conical partition of the convex polytope in conjunction with its facets in order to remain on the level surface of the reverse convex constraint. The algorithm does not need to solve linear programs on a set of cones which converges to a line segment.

1. Introduction

A constraint $g(x) \leqslant 0$ is called a reverse convex constraint if g is a quasi concave function. Optimization problems with reverse convex constraints generally induce nonconvex feasible regions which are often disconnected into several nonconvex parts. As a result, problems with such constraints generally have local optima which are not global optima.

Problems of this form were first studied by Rosen [15] in a control theoretic setting and subsequently, in an engineering setting by Avriel and Williams [1,2]. For a recent interesting application in VLSI design, see Vidigal and Director [24] and Thach [19]. Rosen developed a successive linearization technique which converges to a Kuhn–Tucker point. Meyer [9], in a more general setting, proves convergence to a Kuhn–Tucker point and it is from that paper that the term "reverse convex" is taken.

Ueing [23] was the first to consider global optimization for problems with reverse convex constraints. In fact, the problem Ueing considers is the minimization of a concave function subject to only reverse convex constraints and he develops a combinatorial approach based upon maximizing the objective subject to various combinations of reversals of the reverse convex constraints. Subsequently, Bansal and Jacobsen [4,3] studied the global optimization of a reverse convex program which represented the maximization of network flow capacity. In

© J.C. Baltzer A.G. Scientific Publishing Company

particular, the incremental capacity cost functions were concave and, hence, represented economies-of-scale. Hillestad [8] then developed an edge search procedure for a linear program with one additional reverse convex constraint. Subsequently, Hillestad and Jacobsen [6] studied optimization problems with only reverse convex constraints and showed the convex hull of the feasible region is a convex polytope. They also suggested a cutting plane procedure, based upon Tuy cuts [20], as a possibly useful procedure for finding a good solution; however, they demonstrated that such a method need not converge to a feasible solution. Hillestad and Jacobsen [7] then presented an algorithm, for linear programs with one additional reverse convex constraint, which relies on simplex pivots and vertex enumeration of the feasible region intersected with the hyperplane determined by the current objective value. Tuy [22] then developed a method for convex programs with one additional reverse convex constraint which relies upon being able to solve concave minimization problems. In the latter paper, Tuy also shows that several reverse convex constraints can be converted to one such constraint at the expense of introducing an additional convex constraint and an additional variable. Tuy [21] also shows that virtually any optimization problem can, theoretically, be approximated by a convex program with one additional reverse convex constraint. Indeed, this latter result justifies the importance of this class of optimization problems.

In addition to the above relevance of reverse convex programs, insight is gained into the computational complexity of this class by observing that the $0-1$ linear integer programming problem is equivalent to the associated bounded variable linear program with the additional reverse convex constraint $\Sigma(x_i - x_i^2) \leqslant 0$. Similarly, the minimization of a concave function f subject to linear constraints is obviously convertible to a linear program with one additional reverse convex constraint; in particular, choose an additional variable, x_{n+1}, and minimize x_{n+1} subject to the linear constraints and the additional reverse convex constraint $f(x) - x_{n+1} \leqslant 0$.

Because of the close relationship between reverse convex programs and concave minimization, the reader is referred to the recent survey article by Pardalos and Rosen [14]. For additional papers on reverse convex programming which use, in one way or another, Tuy cuts [20] see, for example, [16,18,11,5].

2. Definitions

We briefly introduce the notation to be used throughout the paper. We denote by F_A a bounded convex polytope in $\mathbb{R}^n$ defined by a system of linear inequalities; that is

$$F_A = \{ x \in \mathbb{R}^n \mid Ax \geqslant b \},$$

where A is an $m \times n$ matrix, $m \geqslant n$, and b is an m-vector. Let

$$G = \{ x \in \mathbb{R}^n \mid g(x) \leqslant 0 \}$$

and

$$G^c = \{ x \in \mathbb{R}^n \mid g(x) > 0 \},$$

where g is a continuous concave function on $\mathbb{R}^n$. We assume G and G^c are both non-empty. Let $F = F_A \cap G$. The problem to be considered in this paper is: $\min\{ c^T x \mid x \in F \}$.

DEFINITION 1

Let x^0 be a vertex of F_A. The function η_1 is defined to be:

$$\eta_1 : \mathbb{R}^n \to \mathbb{R} \cup \{ +\infty \}$$

$$z \mapsto \eta_1(z) = \alpha_1,$$

where $\alpha_1 = \min\{ \alpha \geqslant 0 : g(x^0 + \alpha(z - x^0)) = 0 \}$ if this minimum exists, and $\alpha_1 =$ some constant $\bar{\alpha}$ otherwise ($\bar{\alpha} > 1$).

DEFINITION 2

Let y be an interior point of F_A. The function η_y is defined to be:

$$\eta_y : \mathbb{R}^n \to \mathbb{R} \cup \{ +\infty \}$$

$$z \mapsto \eta_y(z) = \alpha_y,$$

where $\alpha_y = \min\{ \alpha \geqslant 0 : y + \alpha(z - y) \in \partial F_A \}$.
Note that the existence of an interior point y of F_A is equivalent to $\dim(F_A) = n$.

DEFINITION 3

Let $z^1, \ldots, z^n$ be n affinely independent vectors of $\mathbb{R}^n$ with respect to some point x^0 of $\mathbb{R}^n$. Then $H[z^1, \ldots, z^n]$ denotes the convex hull of $z^1, \ldots, z^n$, $\mathcal{H}[z^1, \ldots, z^n]$ denotes the plane defined by $z^1, \ldots, z^n$ and $\mathcal{H}^+[z^1, \ldots, z^n]$ denotes the half-space not containing x^0, whose bounding hyperplane is $\mathcal{H}[z^1, \ldots, z^n]$.

DEFINITION 4

$C[v^1, \ldots, v^n]$ denotes the cone originating at the origin and generated by the n linearly independent vectors $v^1, \ldots, v^n$ of $\mathbb{R}^n$ i.e., $C[v^1, \ldots, v^n] = \{ z \in \mathbb{R}^n$ s.t. $z = \alpha_1 v^1 + \ldots + \alpha_n v^n,\ \alpha_1 \geqslant 0, \ldots, \alpha_n \geqslant 0 \}$

We introduce in the following the notion of a "y-conical coverage" of the feasible region.

DEFINITION 5

Given an interior point y of F_A and n points $z^1, \ldots, z^n$ such that, for each $i = 1, \ldots, n$, the vectors $z^1, \ldots, z^{i-1}, y, z^{i+1}, \ldots, z^n$ are affinely independent with

respect to some vertex x^0 of F_A, a y-conical coverage of F is defined to be a set of cones C_i, $i = 1, \ldots, n$, where

$$C_i = \{x^0\} + C[z^1 - x^0, \ldots, z^{i-1} - x^0, y - x^0, z^{i+1} - x^0, \ldots, z^n - x^0]$$

and such that the following conditions are satisfied:

(i) $\cup_{i=1}^n C_i \supseteq F$;

(ii) $g(z^j) = 0$ for $j = 1, \ldots, n$.

Note that an initial y-conical coverage can be found as follows: Assume we can find an interior point y of F_A such that $g(y) = 0$, and let x^0 be a nondegenerate vertex of F_A which is infeasible for the reverse convex constraint. Then let $v^1, \ldots, v^n$ in F_A be the n neighboring vertices of x^0 and, hence, x^0, $v^1, \ldots, v^n$ are affinely independent. Let $z^1, \ldots, z^n$ be the n points where the rays

$$\{u \mid u = x^0 + \alpha(v^i - x^0),\ \alpha \geq 0\}$$

first intersect ∂G, i.e.,

$$z^i = x^0 + \eta_1(v^i)(v^i - x^0)$$

for $i = 1, \ldots, n$. Let

$$C_i = \{x^0\} + C[z^1 - x^0, \ldots, z^{i-1} - x^0, y - x^0, z^{i+1} - x^0, \ldots, z^n - x^0].$$

Then $\{C_i\}_{i=1}^n$ is an initial y-conical coverage for F.

Note that we have assumed the existence of intersection points of the rays $\{u \mid u = x^0 + \alpha(v^i - x^0)\}$ with ∂G. This assumption is not critical to the algorithm to be presented later.

The idea of the algorithm is to start with the initial coverage and for each cone $C_i = \{x^0\} + C[z^1 - x^0, \ldots, z^{i-1} - x^0, y - x^0, z^{i+1} - x^0, \ldots, z^n - x^0]$, and for each $z^j \notin F_A$, compute the point where the line segment $[y, z^j]$ intersects the boundary of F_A. That is, we compute $\eta_y(z^j)$ and set

$$u^j = y + \eta_y(z^j)(z^j - y).$$

We then redefine z^j by finding the point where the half-line

$$x^0 + \alpha(u^j - x^0), \quad \alpha \geq 0$$

intersects ∂G. That is, we compute $\eta_1(u^j)$ and redefine z^j by

$$z^j = x^0 + \eta_1(u^j)(u^j - x^0).$$

At this point, we collect the major assumptions which are to hold throughout the remainder of the paper:

(1) g is a continuous, concave function defined on $\mathbb{R}^n$.

(2) G^c is bounded. This assumption guarantees that certain line search problems, in the algorithm, will have solutions.

(3) $\text{Int}(F_A)$ is not empty.

(4) A point $y \in \text{Int}(F_A) \cap \partial G$ is available. This assumption is crucial and, in particular, rules out the $0-1$ linear integer programming problem. A procedure for heuristically finding y is discussed at the beginning of section 6.

The next section gives a detailed statement of the algorithm and provides a simple geometric example to explain the algorithm's steps.

3. Statement of the algorithm

Assume we have at hand a point $y \in \text{Int}(F_A) \cap \partial G$. Let $\tilde{F}_A = F_A$.

Step- I Solve the linear program

$$\min\{c^T x \mid x \in \tilde{F}_A\}$$

and assume x^0 is a nondegenerate vertex optimal solution.

Step- II If $x^0 \in G$, stop.
Otherwise, construct an initial y-conical coverage as described in section 2.
Set INFEAS $= \{j \mid z^j \notin \tilde{F}_A\}$.

Step- III DO WHILE INFEAS $\neq \varnothing$
FOR EACH $j \in$ INFEAS DO
find the last point on the line segment $[y, z^j]$ which is in F_A; that is, compute $\eta_y(z^j)$. Define

$$u^j = y + \eta_y(z^j)(z^j - y)$$

and redefine z^j as

$$z^j = x^0 + \eta_1(u^j)(u^j - x^0).$$

If $z^j \in F_A$, remove j from INFEAS.
END DO
END DO WHILE

Step- IV Let

$$C_i = \{x^0\} + C\left[z^1 - x^0, \ldots, z^{i-1} - x^0,\ y - x^0,\ z^{i+1} - x^0, \ldots, z^n - x^0\right],$$

$$\mathcal{H}_i^+ = \mathcal{H}^+\left[z^1, \ldots, z^{i-1},\ y,\ z^{i+1}, \ldots, z^n\right].$$

For $i = 1, \ldots, n$ solve the linear program

$$\min\{c^T x \mid x \in F_A \cap \mathcal{H}_i^+\}$$

and let x_*^i denote an optimal vector.
IF $(\min_{i=1,n}\{c^T z^i\} > \max_{i=1,n}\{c^T x_*^i\})$ THEN
let

$$H^+ = \left\{x \in \mathbb{R}^n \mid c^T x \geqslant \min_{i=1,\ldots,n} c^T x_*^i\right\},$$

let $\tilde{F}_A = F_A \cap H^+$,
let $x^0 = \text{argmin}\{c^T x_*^i,\ i = 1, \ldots, n\}$.

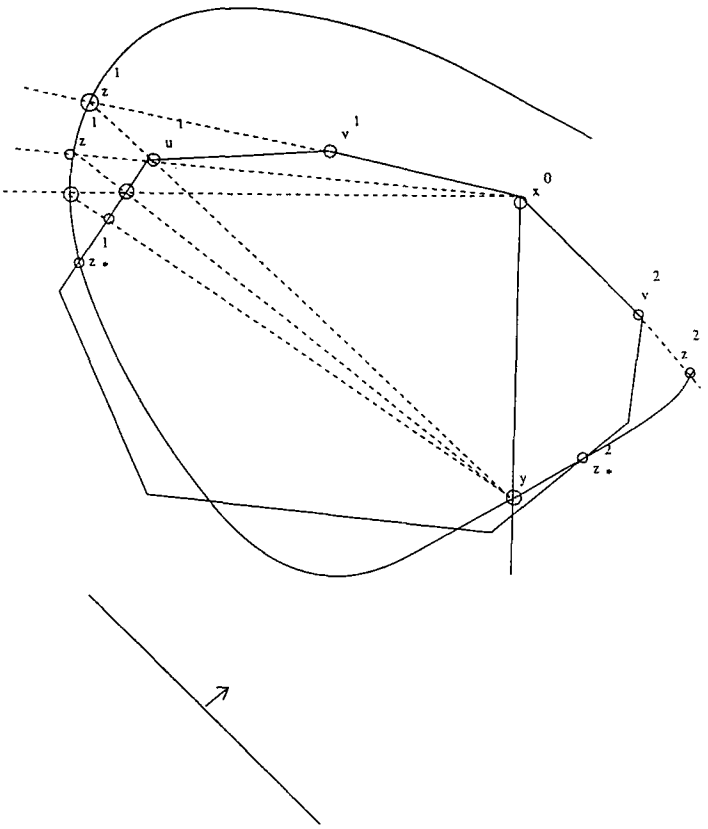

Fig. 1. Two-dimensional example – algorithm's steps.

Return to Step II
ELSEIF $(\min_{i=1,n}\{c^{\mathrm{T}}z^i\} \leqslant \max_{i=1,n}\{c^{\mathrm{T}}x^i_*\})$ THEN
let

$$H^+ = \left\{x \in \mathbb{R}^n \mid c^{\mathrm{T}}x \geqslant \min_{i=1,\ldots,n} c^{\mathrm{T}}z^i\right\},$$

let $\tilde{F}_A = F_A \cap H^+$,
let $x^0 = \operatorname{argmin}\{c^{\mathrm{T}}z^i,\ i = 1,\ldots,n\}$.
Return to Step II.

To illustrate the algorithm geometrically, consider the two-dimensional example in fig. 1. The point x^0 is optimal for the linear program and $x^0 \notin G$. Given the point $y \in \operatorname{Int}(F_A)$, the initial y-conical coverage is given by C_1 and C_2, where

$$C_1 = \{x^0\} + C[z^1 - x^0,\ y - x^0]$$

and

$$C_2 = \{x^0\} + C[z^2 - x^0,\ y - x^0].$$

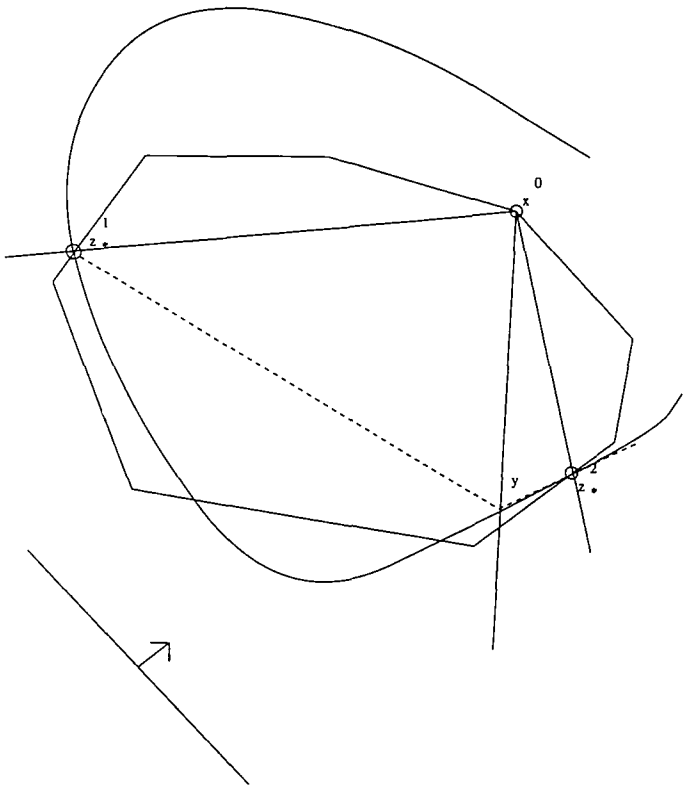

Fig. 2. Two-dimensional example – Step IV.

In what follows, we focus only on reductions of C_1. Given v^1, a neighboring vertex of x^0, we move to the boundary of G, ∂G, by computing $\eta_1(v^1)$. Since $z^1 \notin F_A$, we move from z^1 to the boundary of F_A, ∂F_A, in the direction $y - z^1$ and thereby obtain the point u^1; that is, we compute $\eta_y(z^1)$. The new $z^1 \in \partial G$ is then computed by moving from u^1 in the direction $u^1 - x^0$; that is, we compute $\eta_1(u^1)$. This process continues and, as shown in fig. 1, the resulting sequences of z's and u's converge to $z_*^1 \in \partial F_A \cap \partial G$. Similarly, one can easily see that this process carried out upon C_2 will quite rapidly, for this example, lead to $z_*^2 \in \partial F_A \cap \partial G$. Note that in the two-dimensional case, as depicted in figs. 1 and 2, an optimal point must be among the vectors $\{z_*^1, z_*^2\}$. That is, if one solved the two linear programs (there is no need to do so in two dimensions)

$$\min\{c^T x \mid x \in F_A \cap \mathscr{H}_i^+\}, \quad i = 1, 2,$$

an optimal solution (in this case z_*^2) for the reverse convex program is also optimal for one of these linear programs. In particular, this implies that in two dimensions Step II of the algorithm is not reentered.

4. Convergence of the algorithm

PROPOSITION 1

Consider the sequences $\{z_k^i\}_k$, $i = 1, \ldots, n$, generated by the algorithm. For each $i = 1, \ldots, n$, the entire sequence is contained in the cone originating at x^0 and generated only by the two vectors going respectively through the interior point y and the neighboring vertex v^i of x^0, i.e., for all $k = 1, 2, \ldots,$:

$$z_k^i \in \{x^0\} + C[y - x^0, v^i - x^0].$$

In particular, this result implies that any accumulation point z_*^i of $\{z_k^i\}_k$ is also in the above cone.

Proof

This proposition will subsequently allow us to carry out a proof of affine independence of the limiting vectors z_*^i, $i = 1, \ldots, n$ and also gives some insight into the geometry of this algorithm. Let us show by induction that each entire sequence $\{z_k^i\}_k$ is contained in a cone generated by two rays.

For $k = 1$, the claim is certainly true since $z_1^i - x^0 = \alpha(v^i - x^0)$ for some $\alpha \in \mathbb{R}^+$, $i = 1, \ldots, n$. Assume the result to be true for $k > 1$; the algorithm finds the intersection of the line segment $[y, z_k^i]$ with the polytope F_A, call it u_k^i. We have:

$$u_k^i = y + \eta_y(z_k^i)(z_k^i - y)$$

or

$$u_k^i = (1 - \eta_y(z_k^i))y + \eta_y(z_k^i)z_k^i,$$

i.e. u_k^i is a convex combination of y and z_k^i. Moreover,

$$y \in \{x^0\} + C[y - x^0, v^i - x^0],$$

$$z_k^i \in \{x^0\} + C[y - x^0, v^i - x^0] \quad \text{by the inductive hypothesis,}$$

and

$$\{x^0\} + C[y - x^0, v^i - x^0] \quad \text{is a convex set.}$$

Consequently, $u_k^i \in \{x\} + C[y - x^0, v^i - x^0]$. The next step of the algorithm extends the ray going through u_k^i so as to hit the reverse convex constraint, thus constructing z_{k+1}^i:

$$z_{k+1}^i = x^0 + \eta_1(u_k^i)(u_k^i - x^0)$$
$$= (1 - \eta_1(u_k^i))x^0 + \eta_1(u_k^i)u_k^i.$$

Therefore, since x^0 and u_k^i are both in the set $\{x^0\} + C[y - x^0, v^i - x^0]$, we have $z_{k+1}^i \in \{x^0\} + C[y - x^0, v^i - x^0]$. Since $\{x^0\} + C[y - x^0, v^i - x^0]$ is a closed subset of $\mathbb{R}^n$, we also have that $z_*^i \in \{x^0\} + C[y - x^0, v^i - x^0]$. This shows that

for each of the major iterations, $i = 1, \ldots, n$, of the algorithm, the sequence $\{z_k^i\}_k$ stays within a two-dimensional subspace of $\mathbb{R}^n$. □

A similar argument also shows that for each $i = 1, \ldots, n$, the sequence $\{z_k^i\}_k$ generated by the algorithm has the property

$$z_{k+1}^i \in \{x^0\} + C[y - x^0,\ z_k^i - x^0].$$

The proof of convergence of this algorithm consists of three parts: We will start by showing that for each index $i = 1, \ldots, n$, the sequence $\{z_k^i\}_k$ generated by the algorithm converges to an element z_*^i in $\partial F_A \cap \partial G$. Then we will show that the resulting n vectors z_*^i, $i = 1, \ldots, n$ have the property that $\{z_*^1, \ldots, z_*^{i-1},\ y,\ z_*^{i+1}, \ldots, z_*^n\}$ are affinely independent with respect to x^0 for $i = 1, \ldots, n$. The third part of the proof consists in showing that, from one iteration of the algorithm to the next, no part of the feasible region is deleted.

For the first part, since the proof is the same for each index $i = 1, \ldots, n$, we will drop the superscript.

The algorithm consists of the following main iteration: Given a current $z_k \in \partial G$, some intermediate point u_k is constructed such that:

$$u_k = y + \eta_y(z_k)(z_k - y), \tag{1}$$

$$z_{k+1} = x^0 + \eta_1(u_k)(u_k - x^0). \tag{2}$$

And consequently,

$$u_{k+1} = y + \eta_y(z_{k+1})(z_{k+1} - y). \tag{3}$$

Let us then look more closely at the sequence $\{u_k\}$. By combining (2) and (3) we can see that

$$u_{k+1} = \phi(u_k),$$

where

$$\phi(u) = y + \eta_y(x^0 + \eta_1(u)(u - x^0))(x^0 + \eta_1(u)(u - x^0) - y).$$

This relationship shows that the algorithm is of the fixed-point type.

LEMMA 1

The function ϕ is continuous on $\overline{G}^c$, the closure of G^c.

Proof

The functions η_1 and η_y can be written as

$$\eta_1(z) = \min\{\alpha \geqslant 0:\ g(x^0 + \alpha(z - x^0)) = 0\},$$

$$\eta_y(z) = \min\left\{\alpha \geqslant 0:\ \min_{j=1,n}\left\{\left(A_j(y + \alpha(z - y)) - b_j\right)^2\right\} = 0\right\}.$$

It is immediate that η_1 and η_y are continuous on $\overline{G}^c$ and, hence, so is ϕ. □

LEMMA 2

Let u_* be an accumulation point of $\{u_k\}$. Then, $u_* \in \partial F_A \cap \partial G$.

Proof

Because of proposition 1 and the fact that $y \in \mathrm{Int}(F_A)$, the intersection of the cone $\{x^0\} + C[y - x^0, v - x^0]$ with the bounding faces of F_A, which contain elements of $\{u_k\}$, must be the union of line segments. We denote this intersection by $\bigcup_{i=1}^{p} L_i$, which is a compact set because of the assumed boundedness of F_A. Since $\{u_k\} \subset \bigcup_{i=1}^{p} L_i$, an accumulation point u_* exists, and $\phi(u_k) = u_{k+1}$. Let j be the index such that $u_* \in L_j$. By construction of the algorithm we have, for all sufficiently large k, $u_{k+1} \in (u_*, u_k) \subset L_j$, which implies that $\{u_k\}$ and $\{u_{k+1}\}$ have the same accumulation point u_*. By continuity of ϕ we have $u_* = \phi(u^*)$ and therefore $u_* \in \partial F_A \cap \partial G$. □

THEOREM 1

For each $i = 1, \ldots, n$, the sequence $\{z_k^i\}_k$ generated by the algorithm converges towards $z_*^i \in \partial F_A \cap \partial G$.

Proof

Again, we drop the superscript which denotes a particular cone. We know there exists $u_* \in \partial F_A \cap \partial G$ such that $\{u_k\}$ converges to u_*. We have

$$z_{k+1} = x^0 + \eta_1(u_k)(u_k - x^0).$$

Consequently, $\{z^k\}$ must have a limit point z_* such that

$$z_* = x^0 + \eta_1(u_*)(u_* - x^0).$$

Since $\eta_1(u_*) = 1$, we have $z_* = u_*$. Thus $z^* \in \partial F_A \cap \partial G$. □

Having found the limiting vectors z_*^i, $i = 1, \ldots, n$, the last step of the algorithm constructs the hyperplanes going through y and $n - 1$ of these points. Consequently, we need to prove the affine independence of these vectors.

LEMMA 3

For each $i = 1, \ldots, n$, $\{v^1 - x^0, \ldots, v^{i-1} - x^0, y - x^0, v^{i+1} - x^0, \ldots, v^n - x^0\}$ are linearly independent. $v^1, \ldots, v^n$ are the n neighboring vertices that are used for the original cone coverage.

Proof

Assume there exist $\alpha_1, \ldots, \alpha_n$ such that

$$\alpha_1(v^1 - x^0) + \ldots + \alpha_{i-1}(v^{i-1} - x^0) + \alpha_i(y - x^0) + \alpha_{i+1}(v^{i+1} - x^0) + \ldots + \alpha_n(v^n - x^0) = 0.$$

Since $y \in \text{Int}(F_A)$, there exist $a_1, \ldots, a_n > 0$ such that

$$y = a_1(v^1 - x^0) + \ldots + a_n(v^n - x^0).$$

Consequently, from the linear independence of $(v^1 - x^0), \ldots, (v^n - x^0)$, we have

$$\begin{aligned}
\alpha_1 + \alpha_i a_1 &= 0,\\
&\ldots\\
\alpha_{i-1} + \alpha_i a_{i-1} &= 0,\\
\alpha_{i+1} + \alpha_i a_{i+1} &= 0,\\
&\ldots\\
\alpha_n + \alpha_i a_n &= 0,\\
\alpha_i a_i &= 0.
\end{aligned}$$

From the last equality, since $a_i > 0$, we get $\alpha_i = 0$, which implies $\alpha_1 = \ldots = \alpha_n = 0$. □

PROPOSITION 2

The limiting vectors z_*^i, $i = 1, \ldots, n$ generated by the algorithm are such that, for each $i = 1, \ldots, n$, $z_*^1, \ldots, z_*^{i-1}$, y, $z_*^{i+1}, \ldots, z_*^n$ are affinely independent with respect to x^0.

Proof

It is sufficient to show that $z_*^1 - x^0$, $z_*^2 - x^0 \ldots$, $z_*^{i-1} - x^0$, $y - x^0$, $z_*^{i+1} - x^0, \ldots, z_*^n - x^0$ are linearly independent. Assume there exist $\alpha_1, \ldots, \alpha_n$ such that:

$$\alpha_1(z_*^1 - x^0) + \ldots + \alpha_{i-1}(z_*^{i-1} - x^0) + \alpha_i(y - x^0) + \alpha_{i+1}(z_*^{i+1} - x^0) + \ldots + \alpha_n(z_*^n - x^0) = 0.$$

We also have, since $z_*^i \in \{x^0\} + C[y - x^0, v^i - x^0]$:

$$\forall j = 1, \ldots, n, \exists \lambda_1^j \geqslant 0, \lambda_2^j \geqslant 0, \text{ s.t. } z_*^j - x^0 = \lambda_1^j(y - x^0) + \lambda_2^j(v^j - x^0).$$

Consequently, we have

$$\begin{aligned}
&\left(\alpha_1\lambda_1^1 + \ldots + \alpha_{i-1}\lambda_1^{i-1} + \alpha_i + \alpha_{i+1}\lambda_1^{i+1} + \ldots + \alpha_n\lambda_1^n\right)(y - x^0) + \alpha_1\lambda_2^1(v^1 - x^0)\\
&\quad + \ldots + \alpha_{i-1}\lambda_2^{i-1}(v^{i-1} - x^0) + \alpha_{i+1}\lambda_2^{i+1}(v^{i+1} - x^0) + \ldots + \alpha_n\lambda_2^n(v^n - x^0) = 0.
\end{aligned}$$

Since $v^1 - x^0, \ldots, v^{i-1} - x^0$, $y - x^0$, $v^{i+1} - x^0, \ldots, v^n - x^0$ are linearly independent, the latter equality implies

$$\begin{aligned}
\left(\alpha_1\lambda_1^1 + \ldots + \alpha_{i-1}\lambda_1^{i-1} + \alpha_i + \alpha_{i+1}\lambda_1^{i+1} + \ldots + \alpha_n\lambda_1^n\right) &= 0,\\
\alpha_1\lambda_2^1 &= 0,\\
&\ldots\\
\alpha_{i-1}\lambda_2^{i-1} &= 0,\\
\alpha_{i+1}\lambda_2^{i+1} &= 0,\\
&\ldots\\
\alpha_n\lambda_2^n &= 0.
\end{aligned}$$

Since $z_*^1, \ldots, z_*^n \in \partial F_A$, and $y \in \mathrm{Int}(F_A)$, we have

$$\lambda_2^1 \neq 0, \ldots, \lambda_2^n \neq 0.$$

Thus, from the last $n-1$ equalities, we have

$$\alpha_1 = \ldots = \alpha_{i-1} = \alpha_{i+1} = \ldots = \alpha_n = 0.$$

Therefore, the first equality yields $\alpha_i = 0$. □

We now show that, at each iteration, no feasible point is deleted.

LEMMA 4

Let $\mathscr{H}_i^{+(k)} = \mathscr{H}^+[z_k^1, \ldots, z_k^{i-1}, y, z_{i+1}^k, \ldots, z_n^k]$. Then, for each k,

$$\left\{\bigcap_{i=1}^n \left(\mathscr{H}_i^{+(k)}\right)^c\right\} \cap F = \varnothing.$$

Proof

By construction, the claim is true for $k = 1$, i.e., when $z_1^i = x^0 + \alpha(v^i - x^0)$, for some α greater than zero. Assume the claim is true for k; i.e., assume

$$\left\{\bigcap_{i=1}^n \left(\mathscr{H}_i^+\left[z_k^1, \ldots, z_k^{i-1}, y, z_k^{i+1}, \ldots, z_k^n\right]\right)^c\right\} \cap F = \varnothing$$

and also assume:

$$\left\{\bigcap_{i=1}^n \left(\mathscr{H}_i^+\left[z_{k+1}^1, z_k^2, \ldots, z_k^{i-1}, y, z_k^{i+1}, \ldots, z_k^n\right]\right)^c\right\} \cap F \neq \varnothing.$$

Note that, since one iteration of the algorithm corresponds to a move with respect to one of the z's, it is sufficient to prove the claim for the index $i = 1$. Let $x \in F$ be an element of the above non-empty intersection. We start by noticing that

$$\bigcap_{i=1}^n \left(\mathscr{H}_i^+\left[z_k^1, z_k^2, \ldots, z_k^{i-1}, y, z_k^{i+1}, \ldots, z_k^n\right]\right)^c = \mathrm{Int}\left(\{y\} + C\left[z_k^1 - y, \ldots, z_k^n - y\right]\right).$$

We have

$$g(x) \geqslant 0,$$

$$x \notin \mathrm{Int}\left(\{y\} + C\left[u_k^1 - y, z_k^2 - y, \ldots, z_k^n - y\right]\right),$$

$$x \in \mathrm{Int}\left(\{y\} + C\left[z_{k+1}^1 - y, z_k^2 - y, \ldots, z_k^n - y\right]\right).$$

Consequently, $\exists v \in (u_k^1, z_{k+1}^1)$, $\exists a_1 \in (0, 1)$, $\exists a_2, \ldots, a_n \geqslant 0$, such that

$$x - y = a_1(v - y) + a_2\left(z_k^2 - y\right) + \ldots + a_n\left(z_k^n - y\right),$$

Note that g assumes non-negative values on the half-line originating at y and with direction $x - y$. This fact, combined with $g(y) = g(x) = 0$, implies, because of the concavity of g, that $g(\tilde{x}) = 0$ for all $\tilde{x}$ on this half-line. Therefore, let $\tilde{x}$ be such that

$$g(\tilde{x}) = 0,$$

$$\tilde{x} - y = (x - y) \Big/ \left(\sum_{i=1}^n a_i\right).$$

That is,

$$\tilde{x} = \frac{a_1}{\sum_{i=1}^{n} a_i} v + \frac{a_2}{\sum_{i=1}^{n} a_i} z_2^k + \ldots + \frac{a_n}{\sum_{i=1}^{n} a_i} z_n^k.$$

Thus, by concavity of g

$$g(\tilde{x}) \geqslant \frac{a_1}{\sum_{i=1}^{n} a_1} g(v) + \frac{a_2}{\sum_{i=1}^{n} a_i} g(z_k^2) + \ldots + \frac{a_n}{\sum_{i=1}^{n} a_i} g(z_k^n).$$

Since $g(v) \geqslant 0$ and $g(z_k^2), \ldots, g(z_k^n) = 0$,

$$g(\tilde{x}) \geqslant \frac{a_1}{\sum_{i=1}^{n} a_i} g(v) \geqslant 0.$$

Therefore,

$$g(\tilde{x}) = 0 \Rightarrow a_1 g(v) = 0 \Rightarrow g(v) = 0,$$

which contradicts the fact that z_{k+1}^1 is by construction the first point on the line, originating at x^0 and going through u_k^1, where g attains the zero value. Therefore one cannot have $g(x) = 0$ and, hence, $x \notin F$. □

THEOREM 2

The algorithm converges to an optimal solution of the reverse convex problem.

Proof

The last step of the algorithm solves the following n linear problems:

$$\min_{x \in \mathbb{R}^n} c^{\mathrm{T}} x,$$

$$x \in F_A \cap \mathscr{H}^+ \left[z_*^1, \ldots, z_*^{i-1}, y, z_*^{i+1}, \ldots, z_*^n \right].$$

Let x_*^i be a solution thus obtained to the above linear problem. As seen previously, no feasible point has been eliminated and, consequently,

$$\bigcup_{i=1}^{n} \mathscr{H}_i^+ \supset F.$$

Therefore, for any $x \in F$,

$$c^{\mathrm{T}} x \geqslant \min\{ c^{\mathrm{T}} x_*^1, \ldots, c^{\mathrm{T}} x_*^n \},$$

and the algorithm resumes, setting $x^0 := \operatorname{argmin}\{ c^{\mathrm{T}} x_*^i,\ i = 1, \ldots, n \}$, or $x^0 := \operatorname{argmin}\{ c^{\mathrm{T}} z_*^i,\ i = 1, \ldots, n \}$, and $F_A := F_A \cap \{ x \in \mathbb{R}^n \mid c^{\mathrm{T}} x \geqslant c^{\mathrm{T}} x^0 \}$. The algorithm stops if $x^0 \in F$ (i.e., $g(x^0) = 0$). This has to happen since, at each execution of Step II of the algorithm, the neighboring edges of x^0, one of which being an edge of the original F_A, is eliminated from further searches.

5. Construction of test problems

This section develops a method for constructing linear programs with an additional reverse convex constraint whose optimal solution is known. The method is an application of the procedure of Sung and Rosen [17] for constructing a concave minimization problem whose vertex optimal solution is known.

Testing the algorithm against known sample problems led to the realization that very few numerical examples of significantly high dimension exist in the literature for the obvious reason that they are hard to solve. The question that naturally follows then is: How does one develop a set of reverse convex problems whose solution is known beforehand? The same question has been raised in the context of the minimization of a concave function on a bounded polytope and a variety of methods have been proposed. Sung and Rosen [17] developed a method based on the following geometric fact: since the minimum of a concave function over a polytope is attained at one of the vertices, one can choose, in advance, a vertex and then construct a concave function for which the chosen vertex is optimal. To achieve this, a sphere containing the polytope and passing through this vertex is constructed. Then one can build a quadratic concave function whose minimum over the sphere is attained at this prespecified vertex. The minimum of this function over the polytope will also then occur at this vertex. Pardalos generalizes this idea to indefinite quadratic functions in [13].

For the application of the above idea to reverse convex programs we simply proceed as follows. Consider the linear program

$$\min_{x \in \mathbb{R}^n} \{ c^T x \mid Ax \leqslant b \},$$

where A is an $m \times n$ matrix with $m \geqslant n+1$, $c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$. Let x^* be a point on an edge of the polytope $F_A = \{ x \in \mathbb{R}^n \mid Ax \leqslant b \}$ and let $\alpha = c^T x^*$. The idea is to construct a sphere passing through x^* and which contains

$$F_A \cap \{ x \in \mathbb{R}^n \mid c^T x < \alpha \}.$$

The construction of such a sphere is the same as that of [17] once we have used the objective to redefine the convex polytope. We assume x^* is on an edge of F_A and is not a vertex of F_A. Then x^* is a vertex of the following polyhedron

$$F_A \cap \{ x \in \mathbb{R}^n \mid c^T x \leqslant \alpha \}.$$

We assume the following linear equalities define x^*

$$a_i^T x = b_i, \quad i = 1, \ldots, n-1,$$

$$c^T x = c^T x^*,$$

where the first $n-1$ equalities are, without loss of generality, assumed to define the edge on which x^* lies and a_i^T is the ith row of A. Let B denote the matrix formed by the first $n-1$ rows of A and by the row c^T. Also let $\tilde{b}$ denote the right hand side corresponding to the first $n-1$ components of b and let $\tilde{b} = (\tilde{b}, c^T x^*)^T$.

As observed by Sung and Rosen [17], one can find the radius of a ball containing $F_A \cap \{ x \in \mathbb{R}^n \mid c^T x \leqslant c^T x^* \}$ by solving the following n linear programs: For $i = 1, \ldots, n-1$

$$\min\{ a_i^T x \mid Ax \leqslant b,\ c^T x \leqslant c^T x^* \}$$

and

$$\min\{ c^T x \mid Ax \leqslant b,\ c^T x \leqslant c^T x^* \}.$$

Let $v_1, \ldots, v_n$ denote the respective optimal objective values of these n linear programs and define

$$v^T(\epsilon) = (v_1 - \epsilon,\ v_2 - \epsilon, \ldots, v_n - \epsilon),$$

where ϵ is an arbitrarily small positive number. Let the jth component of the vector r be defined as

$$r_j = 1/2\left(\tilde{\tilde{b}}_j + v_j(\epsilon) \right).$$

We can now state the following theorem.

THEOREM 3 [17]

x^* is the unique optimal solution to the following reverse convex problem:

$$\begin{aligned} \min \quad & c^T x \\ \text{s.t.} \quad & Ax \leqslant b, \\ & - \| Bx - r \|^2 \leqslant -1/4 \| \tilde{\tilde{b}} - v(\epsilon) \|^2, \end{aligned}$$

where the norm is the Euclidean norm.

Example 1: A three-dimensional example

$$\begin{aligned} \min_{x \in \mathbb{R}^3} \quad & -2x_1 + x_2 - x_3, \\ \text{s.t.} \quad & x_1 + x_2 + x_3 \leqslant 4, \\ & x_1 \leqslant 2, \\ & x_3 \leqslant 3, \\ & 3x_2 + x_3 \leqslant 6, \\ & x_1,\ x_2,\ x_3 \geqslant 0. \end{aligned}$$

The above three-dimensional polytope is found in [12]. The prespecified edge point is $x^* = (0.5, 0, 3)$ with objective value -4. This enables us to construct a reverse convex constraint to be appended to the above linear problem. Recall that the equation of such a reverse constraint is

$$- \| Bx - r \|^2 \leqslant -1/4 \| \tilde{\tilde{b}} - v(\epsilon) \|^2,$$

Table 1
Three-dimensional example

y	1.269053724	0.730946276	1.269053724
x^0	2	0	0
z_*^1	2	1.326639	0.67336
z_*^2	1.9054	0.09455	0
z_*^3	0.5625699	0.218715	3
x_*^1	0.58817	0	3
x_*^2	0.68017	0.31983	3
x_*^3	1.8598	0	0
x^0	0.58817	0	3
z_*^1	1.46271215	1.1583201	1.37896
z_*^2	1.99953758	0.3509639	0
z_*^3	0.5	0	3
x^*	0.5	0	3

where, in this case,

$$B = \begin{pmatrix} 0 & 0 & 1 \\ 0 & -1 & 0 \\ -2 & 1 & -1 \end{pmatrix}$$

and

$$\tilde{b} = \begin{pmatrix} 3 \\ 0 \\ -4 \end{pmatrix}, \quad v(\epsilon) = \begin{pmatrix} 0-\epsilon \\ -1-\epsilon \\ -6-\epsilon \end{pmatrix}, \quad r = \begin{pmatrix} 1.5-\epsilon/2 \\ -0.5-\epsilon/2 \\ -5-\epsilon/2 \end{pmatrix}.$$

If one sets $\epsilon = 0.2$ and allows MINOS 5.1 [10] to begin at its own computed starting point, the solution (1.8772, 0.0, 0.0) with objective value -3.7544 is found and, hence, this problem has non-optimal local minima. Solving the resulting problem using the algorithm of section 3 yields an optimal solution $x^* = (0.5, 0, 3)$ while taking 67 iterations with a CPU time of 0.1 seconds on a Sun SPARCstation 1.

In this paper, an iteration is defined as a step between two successive points z_k^i, z_{k+1}^i on the boundary of G. Hence, the total number of iterations is the sum of all such steps in all n cones. Table 1 is a summary of the last points produced by the algorithm of section 3. The first row is the interior point y; the second row is the initial vertex x^0. The next three rows comprise the z_*^i for $i = 1, 2, 3$ and the following three rows contain the solutions x_*^i of the resulting linear programs. There is one return to Step II and the next row of the table is the new x^0 at which the algorithm is initiated. The new z_*^i are produced and the last row contains the optimal solution, $x^* = z_*^3$.

6. Numerical results

In this section, we present examples of concave minimization problems as well as other types of reverse convex programs. An iteration is defined as a step between two successive points z_k^i, z_{k+1}^i on the boundary of G. Hence, the total number of iterations is defined to be the total number of such steps over all iterations and all cones. In all of the numerical work presented, we used a tolerance parameter, for the calculation of η_1 and η_y, equal to 10^{-8}. We also report the CPU times for each example and all examples were run on a Sun SPARCstation 1. While such CPU times are reported, not too much importance should be placed upon them since we made little effort to make the reverse convex code numerically efficient. For instance, we used a bisection algorithm to compute η_1 and η_y.

Some words of caution regarding numerical implementation are in order. In particular, the convergence proof assumes that the z_k^i and the z_*^i are computed exactly. Of course, each such computation is the result of an infinite process and, as a result, numerical accuracy is extremely important. In fact, under relatively inaccurate computation of the z_k^i and the z_*^i, the final answer to the problem may be infeasible. This behavior has not occurred in any of our examples using 10^{-8} as the numerical tolerance for the calculation of η_1 and η_y. The numerical implementation of such nonconvex problems will be the subject of another paper.

6.1. CONCAVE MINIMIZATION EXAMPLES

Note that the construction of the interior point y is trivial for concave minimization problems subject to linear inequality constraints because of the assumption that $\dim(F_A) = n$.

Example 2: Falk-Hoffman example

$$
\begin{aligned}
\min_{x \in \mathbb{R}^3} \quad & -(x_1 - 1)^2 - x_2^2 - (x_3 - 1)^2, \\
\text{s.t.} \quad & x_1 + x_2 - x_3 \leqslant 1, \\
& -x_1 + x_2 - x_3 \leqslant -1, \\
& 12x_1 + 5x_2 + 12x_3 \leqslant 34.8, \\
& 12x_1 + 12x_2 + 7x_3 \leqslant 29.1, \\
& -6x_1 + x_2 + x_3 \leqslant -4.1, \\
& x_1, x_2, x_3 \geqslant 0.
\end{aligned}
$$

In order to use the reverse convex algorithm and a linear programming sub-

Table 2
Falk – Hoffman example

y	1.0006	0.498152	1.012933	0.981522	1.229846
x^0	1.0333	0.4	1.7	0	40
z_*^1	1.0333	0.4	1.7	39.348	40
z_*^2	0.982627	0.694459	0.71183	0.9736	1.5393
z_*^3	0.95891	0	1.2591	0.9859	1.05479
z_*^4	1.4236058	0.265115	0.688721	0.97909	1.325718
z_*^5	1.03333	0.4	1.7	0	0.65111
x_*^1	1	0	0	0	2.5281
x_*^2	1.76	0	1.14	39.184	40
x_*^3	1	0.9	0.9	38.53	40
x_*^4	0.72857	0	0.27143	37.583	40
x_*^5	1	0	0	37.47	40
x^0	1	0	0	37.47	40
z_*^1	1.000531	0.72967	1.173508	6.0109	6.5734
z_*^2	0.943949	0.49681	1.0668808	1.079436	1.333876
z_*^3	1.237667	0.479859	1.212811	2.32154	2.65355
z_*^4	1.000615	0.498152	1.012933	0	0.248324
z_*^5	1	0	0	39	40
x^*	1	0	0	39	40

routine, we introduce two new nonnegative variables η_1, η_2 thus rewriting this problem as:

$$
\begin{aligned}
\min_{\eta_1,\eta_2\in\mathbb{R},\,x\in\mathbb{R}^3} \quad & \eta_1 - \eta_2, \\
\text{s.t.} \quad & x_1 + x_2 - x_3 \leqslant 1, \\
& -x_1 + x_2 - x_3 \leqslant -1, \\
& 12x_1 + 5x_2 + 12x_3 \leqslant 34.8, \\
& 12x_1 + 12x_2 + 7x_3 \leqslant 29.1, \\
& -6x_1 + x_2 + x_3 \leqslant -4.1, \\
& \eta_1 \leqslant 40, \\
& \eta_2 \leqslant 40, \\
& x_1,\ x_2,\ x_3,\ \eta_1,\ \eta_2 \geqslant 0, \\
& -(x_1-1)^2 - x_2^2 - (x_3-1)^2 - \eta_1 + \eta_2 \leqslant 0.
\end{aligned}
$$

Note that bounds on η_1 and η_2 have also been introduced due to the original assumption that the polytope is bounded. Table 2 is a summary of the intermediate points generated by the algorithm. The first row is the interior point $y \in \mathrm{Int}(F_A) \cap \partial G$. The second row is the initial vertex x^0. The next 5 rows provide the coordinates of $z_*^1, \ldots, z_*^5$ (the two last columns indicate the values of η_1 and η_2), and the following five rows provide the coordinates of the solutions of the 5

linear problems solved at Step IV of the algorithm. We return to Step II with the new $x^0 = x_*^5$. The new z_*^i, $i = 1, \ldots, 5$ are computed and it is found that $x^* = z_*^5$ is optimal. Finally, the last row contains the optimal vector in the higher dimensional space considered (because of the addition of two variables, the space is $\mathbb{R}^5$ rather than $\mathbb{R}^3$). The method required 61 iterations with a resulting CPU time of 0.4 seconds.

Example 3: Six-dimensional cube example

$$\begin{aligned} \min_{x \in \mathbb{R}^6} \quad & 10.5x_1 - 3.95x_2 + 3.0x_3 + 5.0x_4 \\ & + 1.5x_5 - 1.5x_6 - 1.5x_1^2 - x_2^2 \\ & - x_3^2 - 2x_4^2 - x_5^2 - 2.5x_6^2, \\ \text{s.t.} \quad & 0 \leqslant x_1 \leqslant 99, \\ & 0 \leqslant x_2 \leqslant 99, \\ & 0 \leqslant x_3 \leqslant 99, \\ & 0 \leqslant x_4 \leqslant 99, \\ & 0 \leqslant x_5 \leqslant 99, \\ & 0 \leqslant x_6 \leqslant 99. \end{aligned}$$

Table 3
Six-dimensional cube example

y	0.999945	0.999945	0.999945	0.999945
	0.999945	0.999945	100.00005	94.449866
x^0	99	99	99	99
	99	99	0	10000000
z_*^1	99	99	99	99
	99	99	9913231.4499	10000000
z_*^2	0	1.000033	1.000033	1.000033
	1.000033	1.000033	99.999965	103.45033
z_*^3	0.999897	0	0.999897	0.999897
	0.999897	0.999897	100	89.500364
z_*^4	0.999965	0.999965	0	0.999965
	0.999965	0.999965	100	96.449879
z_*^5	0.999975	0.999975	0.999975	0
	0.999975	0.999975	100	97.44991
z_*^6	0.999950	0.999950	0.999950	0.999950
	0	0.9999505	100	94.9499078
z_*^7	0.9999064	0.9999064	0.9999064	0.9999064
	0.9999064	0	100	90.45037
z_*^8	99	99	99	99
	99	99	0	86768.55

This example is used as a test example since it has the obvious answer $x^* = (99, 99, 99, 99, 99, 99)$. Table 3 is a summary of the intermediate points generated by the algorithm. The first row is the interior point $y \in \mathrm{Int}(F_A) \cap \partial G$. The next row is the initial vertex x^0. The next 8 rows give the coordinates of $z_*^1, \ldots, z_*^8$ and table 4 provides the coordinates of the solutions of the 8 linear problems solved by Step IV of the algorithm. Finally, the last row of table 4 contains the optimal vector in the higher dimensional space considered (because of the addition of two variables, the space is $\mathbb{R}^8$ rather than $\mathbb{R}^6$). There were no returns to Step II. The algorithm used 40 iterations with a CPU time of 0.6 seconds.

Example 4: Six-dimensional sliced cube example

$$
\begin{aligned}
\min_{x \in \mathbb{R}^6} \quad & 10.5x_1 - 3.95x_2 + 3.0x_3 + 5.0x_4 \\
& + 1.5x_5 - 1.5x_6 - 1.5x_1^2 - x_2^2 \\
& - x_3^2 - 2x_4^2 - x_5^2 - 2.5x_6^2, \\
\text{s.t.} \quad & 0 \leqslant x_1 \leqslant 99, \\
& 0 \leqslant x_2 \leqslant 99, \\
& 0 \leqslant x_3 \leqslant 99, \\
& 0 \leqslant x_4 \leqslant 99, \\
& 0 \leqslant x_5 \leqslant 99, \\
& 0 \leqslant x_6 \leqslant 99, \\
& x_1 + x_2 + x_3 + x_4 + x_5 + x_6 \leqslant 500, \\
& x_1 + 3x_2 + 6x_3 + 2x_4 \geqslant 50, \\
& 3x_5 + 4x_6 \geqslant 50, \\
& x_3 + 2x_4 + 3x_5 + x_6 \leqslant 350.
\end{aligned}
$$

After 67 iterations the algorithm produced the vector $x^* = (99, 99, 53, 99, 0, 99)$ with $f(x^*) = -70262.05$. As a means of comparison with existing codes for nonlinear programming, we ran the same problem using the well-known MINOS 5.1 [10] package, obtaining the vector $x^{**} = (99, 99, 99, 76, 0, 99)$ with an objective value of approximately -69181.04.

6.2. REVERSE CONVEX TEST PROBLEMS

The following test problems have been generated using the theoretical results of section 5. Given a linear problem with a bounded feasible region, and given a point x^* on an edge of this polytope, a subroutine first builds a quadratic function such that, if the corresponding reverse convex constraint is appended to the linear problem, x^* is optimal for the resulting reverse convex problem. The construction of the interior point y is done heuristically. Two different vertices of the polytope are chosen, say by maximizing and minimizing a linear objective,

Table 4
Linear program solutions; six-dimensional cube example

x_*^1	99	99	99	99
	99	99	0	86768.5473
x_*^2	99	99	99	99
	99	99	991320	10000000
x_*^3	99	99	99	99
	99	99	991320	10000000
x_*^4	99	99	99	99
	99	99	991320	10000000
x_*^5	99	99	99	99
	99	99	991320	10000000
x_*^6	99	99	99	99
	99	99	991320	10000000
x_*^7	99	99	99	99
	99	99	991320	10000000
x_*^8	99	99	99	99
	99	99	991320	10000000
x^*	99	99	99	99
	99	99	0	86768.5473

and then the line between them is searched for an intersection with the boundary of G.

Example 5: A twenty-dimensional example

$$\min_{x \in \mathbb{R}^{20}} \quad -9x_1 - 2x_5 - 8x_9 - 6x_{13} - 3x_{17} - 6x_2 - 7x_6 - 9x_{10} - x_{14} - 5x_{18} - 7x_3 - 8x_7 - 6x_{11} - 3x_{15} - 4x_{19} - 5x_4 - 4x_8 - 3x_{12} - 2x_{16} - x_{20},$$

subject to:

$$x_1 + x_2 + x_3 + x_4 \leqslant 1,$$
$$x_5 + x_6 + x_7 + x_8 \leqslant 2,$$
$$x_9 + x_{10} + x_{11} + x_{12} \leqslant 3,$$
$$x_{13} + x_{14} + x_{15} + x_{16} \leqslant 3,$$
$$x_{17} + x_{18} + x_{19} + x_{20} \leqslant 4,$$
$$x_1 + x_5 + x_9 + x_{13} + x_{17} \leqslant 3,$$
$$x_2 + x_6 + x_{10} + x_{14} + x_{18} \leqslant 3,$$
$$x_3 + x_7 + x_{11} + x_{15} + x_{19} \leqslant 3,$$
$$x_4 + x_8 + x_{12} + x_{16} + x_{20} \leqslant 3,$$
$$0 \leqslant x_i \leqslant 1, \ i = 1, \ldots, 20.$$

The prechosen edge vector is:

$$x^* = (0.5, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1),$$

with objective value of $v^* = -17.5$. Then a reverse convex constraint is constructed as described in section 5. The reverse convex algorithm finds, in 82 iterations with CPU time = 2.1 seconds, the vector

$$x^{**} = (1, 0, 0, 0, 0, 0, 0, 0, 0.33762972396, 0, 0.9665335136736, 0, 0, 0, 0, 0, 0, 0, 0, 0)$$

with optimal objective value $v^* = -17.5$. Note that $x^{**} \neq x^*$. This occurs since we deliberately took ϵ, of the previous section, equal to zero and, therefore, an alternative optimal solution can be produced.

Example 6: A forty-dimensional example

$$\begin{aligned}\min_{x \in \mathbb{R}^{40}} \quad & -215x_1 - 116x_2 - 670x_3 - 924x_4 - 510x_5 \\ & -600x_6 - 424x_7 - 942x_8 - 43x_9 - 369x_{10} \\ & -408x_{11} - 52x_{12} - 319x_{13} - 214x_{14} - 851x_{15} \\ & -394x_{16} - 88x_{17} - 124x_{18} + 17x_{19} - 779x_{20} \\ & -278x_{21} - 258x_{22} - 271x_{23} - 281x_{24} - 326x_{25} \\ & -819x_{26} - 485x_{27} - 454x_{28} - 297x_{29} - 53x_{30} \\ & -136x_{31} - 796x_{32} - 114x_{33} - 43x_{35} \\ & -268x_{36} - 179x_{37} - 78x_{38} - 105x_{39} - 281x_{40},\end{aligned}$$

subject to

$$\begin{aligned}& 8x_1 + 11x_2 + 6x_3 + x_4 + 7x_5 + 9x_6 + 10x_7 + 3x_8 \\ & + 11x_9 + 11x_{10} + 2x_{11} \\ & + x_{12} + 16x_{13} + 18x_{14} + 2x_{15} + x_{16} + x_{17} + 2x_{18} + \\ & 3x_{19} + 4x_{20} + 7x_{21} + 6x_{22} \\ & 2x_{23} + 2x_{24} + x_{25} + 2x_{26} + x_{27} + 8x_{28} + 10x_{29} \\ & + 2x_{30} + x_{31} + 9x_{32} + x_{33} \\ & + 9x_{34} + 2x_{35} + 4x_{36} + 10x_{37} + 8x_{38} + 6x_{39} + x_{40} \leqslant 5000; \\ & 5x_1 + 3x_2 + 2x_3 + 7x_4 + 7x_5 + 3x_6 + 6x_7 + 2x_8 + 15x_9 \\ & + 8x_{10} + 16x_{11} \\ & + x_{12} + 2x_{13} + 2x_{14} + 7x_{15} + 7x_{16} + 2x_{17} + 2x_{18} + 4x_{19} \\ & + 3x_{20} + 2x_{21} \\ & + 13x_{22} + 8x_{23} + 2x_{24} + 3x_{25} + 4x_{26} + 3x_{27} + 2x_{28} + x_{29} \\ & + 10x_{30} + 6x_{31} \\ & + 3x_{32} + 4x_{33} + x_{34} + 8x_{35} + 6x_{36} + 3x_{37} + 4x_{38} + 6x_{39} \\ & + 2x_{40} \leqslant 5000; \\ & 3x_1 + 4x_2 + 6x_3 + 2x_4 + 2x_5 + 3x_6 + 7x_7 + 10x_8 + \\ & 3x_9 + 7x_{10} + 2x_{11} \\ & + 16x_{12} + 3x_{13} + 3x_{14} + 9x_{15} + 8x_{16} + 9x_{17} + 7x_{18} \\ & + 6x_{19} + 16x_{20} + 12x_{21} \\ & + x_{22} + 3x_{23} + 14x_{24} + 7x_{25} + 13x_{26} + 6x_{27} + 16x_{28} \\ & + 3x_{29} + 2x_{30} + x_{31} \\ & + 2x_{32} + 8x_{33} + 3x_{34} + 2x_{35} + 7x_{36} + x_{37} + 2x_{38} + 6x_{39}\end{aligned}$$

$$+5x_{40} \leqslant 5000;$$
$$0 \leqslant x_i \leqslant 99,\ i = 1, \ldots, 40.$$

Again, by running a procedure that creates a reverse convex constraint to be appended to this linear program, we build a reverse convex problem with prechosen optimal vector

$$x^* = (49.5, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)$$

and objective value $v^* = -10642.5$. The algorithm finds x^* after 189 iterations and a CPU time of 5.2 seconds.

References

[1] M. Avriel and A.C. Williams, Complementary geometric programming, SIAM J. Appl. Math. 19 (1970) 125–141.

[2] M. Avriel and A.C. Williams, An extension of geometric programming with applications in engineering optimization, J. Eng. Math. 5 (1971) 187–194.

[3] P.P. Bansal and S.E. Jacobsen, An algorithm for optimizing network flow capacity under economies of scale, J. Optimization Theory Appl. 15 (5) (1975) 565–586.

[4] P.P. Bansal and S.E. Jacobsen, Characterization of local solutions for a class of nonconvex programs, J. Optimization Theory Appl. 15 (5) (1975).

[5] J. Fulop, A finite cutting plane method for solving linear programs with an additional reverse convex constraint, Working Paper, Department of Operations Research, Hungarian Academy of Science, Budapest (1988).

[6] R.J. Hillestad and S.E. Jacobsen, Reverse convex programming, Appl. Math. Optimization 6 (1980) 63–78.

[7] R.J. Hillestad and S.E. Jacobsen, Linear programs with an additional reverse convex constraint, Appl. Math. Optimization 6 (1980) 257–269.

[8] R.J. Hillestad, Optimization problems subject to a budget constraint with economies of scale, Oper. Res. 23 (6) (1975) 1091–1098.

[9] R. Meyer, The validity of a family of optimization methods, SIAM J. Control 8 (1970) 41–54.

[10] B.M. Murtagh and M.A. Saunders, MINOS 5.1 user's guide, Technical Report SOL 83-20R, Systems Optimization Laboratory, Department of Operations Research, Stanford University (January 1987).

[11] L.D. Muu, A convergent algorithm for solving linear programs with an additional reverse convex constraint, Kybernetika 21 (1985) 428–435.

[12] C. Papadimitriou and K. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity* (Prentice Hall, Englewood Cliffs, NJ, 1982).

[13] P.M. Pardalos, Generation of large-scale quadratic programs for use as global optimization test problems, ACM Trans. Math. Software 13 (2) (1987) 133–137.

[14] P.M. Pardalos and J.B. Rosen, Methods for global concave minimization: A bibliographic survey, SIAM Rev. (1986).

[15] J.B. Rosen, Iterative solution of nonlinear optimal control problems, SIAM J. Control 4 (1966) 223–244.

[16] S. Sen and H.D. Sherali, Nondifferentiable reverse convex programs and facetial convexity cuts via a disjunctive characterization, Math. Programming 37 (1987) 169–183.

[17] Y.Y. Sung and J.B. Rosen, Global minimum test problem construction, Math. Programming 24 (1982) 353–355.
[18] P.T. Thach, Convex programs with several additional reverse convex constraints, Preprint Series, Institute of Mathematics, Hanoi (1985).
[19] P.T. Thach, The design centering problem as a d.c. programming problem, Preprint Series, Institute of Mathematics, Hanoi (1986).
[20] H. Tuy, Concave programming under linear constraints, Sov. Math. 5 (1964) 1437–1440.
[21] H. Tuy, A general deterministic approach to global optimization via d.c. programming, *Fermat Days 1985: Mathematics for Optimization* (1986) pp. 98–118.
[22] H. Tuy, Convex programs with an additional reverse convex constraint, J. Optimization Theory Appl. 52 (3) (1987) 463–485.
[23] U. Ueing, A combinatorial method to compute a global solution of certain non-convex optimization problems, in: *Numerical Methods for Non-linear Optimization*, ed. F.A. Lootsma (Academic Press, 1972) pp. 223–230.
[24] L.M. Vidigal and S.W. Director, A design centering algorithm for nonconvex regions of acceptability, IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems, CAD-1(1) (Jan. 1982).