A General Solution Algorithm for The Linearly Constrained Global Optimization Problems

Hossein Arsham
University of Baltimore, Baltimore, Maryland 21201, USA
harsham@UBmail.ubalt.edu
Miro Gradišar, Mojca Indihar Štemberger
University of Ljubljana, Faculty of economics,
1000 Ljubljana, Slovenia
mojca.stemberger@uni-lj.si, miro.gradisar@uni-lj.si

Abstract:

The article presents a simple alternative approach to solve general linearly constraint optimization problems. This class of optimization problems includes fractional, nonlinear network models, quadratic, and linear programs. The unified approach is accomplished by converting the constrained optimization problem to an unconstrained optimization problem through a parametric representation of its feasible region. The proposed solution algorithm consists of three phases. In phase 1, the parametric representation of the feasible region (the polyhedron) is constructed. Since in the case of nonlinear objective function the optimum can appear at any point of the polyhedron, therefore, the edges and faces have to be identified. This is done by a modified version of an algorithm for finding the *V-representation* of the polyhedron. Then, in phase 2, the parametric representation of the feasible region is used for the parametric representation of objective function. Following this, the global optimal solution of the parametric objective function is found by computing the stationary points and evaluation of the objective function at these points as well as at the vertices.

Keywords:

Global optimization, Linearly constrained optimization, Polyhedra, Linear programming, Nonlinear programming

1 Introduction

Linearly constrained optimization problems are extremely varied. They differ in their functional form of the objective function, constraints, and in the number of variables. Although the structure of this problem is simple, finding a global solution – and even detecting a local solution is known to be difficult to solve. The simplest form of this problem is realized when the objective function is linear. The resulting model is a linear program (LP). Other problems include fractional, nonlinear network models, quadratic, separable, convex and nonconvex programs.

There are well over 400 different solution algorithms in solving different kinds of linearly constrained optimization problems. However, there is not one algorithm superior to others in all cases. For example in applying the Karush-Kuhn-Tucker (KKT) condition, it may be difficult, if not essentially impossible, to derive an optimal solution directly [20]. The most promising numerical solution algorithm is the feasible direction method, however, if objective function is nonconvex then the best one can hope for is that it converges to a local optimal point.

The paper develops a simple alternative approach to solve general continous optimization problems with linear constraints. The unified approach is accomplished by converting the constrained optimization problem to an unconstrained optimization problem through a parametric representation of its feasible region.

The remainder of this paper is organized as follows: Section 2 presents the main phases of the algorithm with some definitions and constructive theorems. The algebraic method for finding all vertices, extreme rays, edges and faces of the feasible region is developed in section 3. For better understanding of the presented concepts both sections are illustrated by numerical examples. Two numerical examples of the algorithm are presented in section 4 in the context of numerical problems solved by other methods for comparative purposes. The last section contains the conclusions with some useful remarks.

2 New Solution Algorithm

We want to solve the following problem with linear feasible region:

Problem P: Maximize $f(\mathbf{x})$

Subject to: $\mathbf{Ax} \leq \mathbf{b}$

with some variables x_i have explicit upper and/or lower bounds and some are unrestricted in sign, where $\mathbf{A}$ is $m \times n$ matrix, $\mathbf{b}$ is m-vector and f is a continuous function. Problem $\mathbf{P}$ is a subset of a larger set of problems known as Continuous Global Optimization [25] with diverse areas of applications [14, 19, 21, 24].

The feasible region of the problem $\mathbf{P}$ is obviously the set of points that defines the polyhedron. In the proposed solution we need to find all the stationary points of objective function f inside and at the bounds of the polyhedron.

A *polyhedron* with finite number of vertices can be represented in two equivalent ways [8]: H-representation and V-representation.

Definition 1 *An H-representation of the polyhedron is given by an $m \times n$ matrix $\mathbf{A} = (a_{i,j})$ and m-vector $\mathbf{b} = (b_i)$:*

$$S = \{\mathbf{x} \in R^n; \quad \mathbf{Ax} \leq \mathbf{b}\}.$$

Definition 2 *A vertex $\mathbf{v} \in R^n$ is a point of S that satisfies an affinely independent set of n inequalities as equations.*

An extreme ray $\mathbf{w} \in R^n$ is a direction such that for some vertex $\mathbf{v}$ and any positive scalar μ, $\mathbf{v} + \mu\mathbf{w}$ is in S and satisfies some set of $n-1$ affinely independent inequalities as equations.

Definition 3 *An V-representation of S is given by a minimal set of M vertices $\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_M}$ and N extreme rays $\mathbf{w_1}, \mathbf{w_2}, \ldots, \mathbf{w_N}$:*

$$S = \left\{ \mathbf{x} \in R^n; \quad \mathbf{x} = \sum_{i=1}^{M} \lambda_i \mathbf{v_i} + \sum_{j=1}^{N} \mu_j \mathbf{w_j}, \ \lambda_i, \mu_j \geq 0, \ \sum_{i=1}^{M} \lambda_i = 1 \right\}.$$

Definition 4 *A face of polyhedron S is a boundary set of S containing points on a line or plane (or hyper-plane). An edge is the line segment between any two adjacent vertices.*

If a feasible region is bounded then a corresponding polyhedron is called a polytope which has no extreme rays. Its V-representation is given by a convex combination of the vertices.

Example 1:

The polyhedron in Figure 1, defined by

$$\begin{aligned} 5x_1 - x_2 &\leq 30 \\ x_1 &\leq 5 \\ x_1 &\geq 0 \end{aligned}$$

has two vertices and one extreme ray:

$$\mathbf{v_1} = \begin{bmatrix} 0 \\ -30 \end{bmatrix},\ \mathbf{v_2} = \begin{bmatrix} 5 \\ -5 \end{bmatrix},\ \mathbf{w} = \begin{bmatrix} 0 \\ 1 \end{bmatrix},$$

which indicates that its parametric representation is given by

$$\{(x_1, x_2);\ (x_1, x_2) = (5\lambda_2, -30\lambda_1 - 5\lambda_2 + \mu),\ \lambda_1, \lambda_2, \mu \geq 0,\ \lambda_1 + \lambda_2 = 1\}.$$

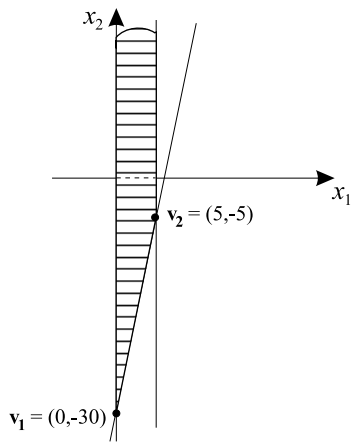

Figure 1: Polyhedron for the example 1

The edge connecting both vertices can be expressed as

$$\{(x_1, x_2);\ (x_1, x_2) = (5\lambda_2, -30\lambda_1 - 5\lambda_2),\ \lambda_1, \lambda_2 \geq 0,\ \lambda_1 + \lambda_2 = 1\}.$$

Definition 5 *The parametric representation of the objective function f is given by:*

$$f(\mathbf{x}) = f(\mathbf{x}(\lambda, \mu)) = f(\lambda, \mu).$$

For example, the parametric representation of the function $f(x_1, x_2) = x_1^3 + x_2$ defined on the polyhedron from Example 1 is given by

$$f(\lambda_1, \lambda_2, \mu) = 125\lambda_2^3 - 30\lambda_1 - 5\lambda_2 + \mu$$

The proposed algorithm for the global solution of problem **P** is based on the following theorem:

Theorem 1 *The maximum (minimum) points of an optimization problem* **P** *correspond to the maximization (minimization) of the parametric objective function* $f(\lambda, \mu)$.

Proof: The proof follows from the fact that the parametric representation of $x = x(\lambda, \mu)$ is a linear function.

The following result is applicable to LP problems.

Theorem 2 *Let the terms with the largest (smallest) coefficients in* $f(\lambda, \mu)$ *be denoted by* λ_L *and* λ_S *respectively.*

1. *If we are looking for the maximum of objective function and some coefficient at* μ_i *(if any) is positive then the problem* **P** *has no solution. If all the coefficients are negative the optimal value is obtained at the vertex corresponding to* λ_L.

2. *If we are looking for the minimum of objective function and some coefficient at* μ_i *(if any) is negative then the problem* **P** *has no solution. If all the coefficients are positive the optimal value is obtained at the vertex corresponding to* λ_S.

Proof: The proof follows from the fact that if we are looking for maximum and some coefficient at $\mu_i \geq 0$ is positive then the problem has no solution, since the objective function can have arbitrary large values. Otherwise the largest value of f is obtained by setting λ_L to 1 and all the others $\lambda_i = 0$ and $\mu_i = 0$. The proof for minimum is similar.

This is a new proof of the well-known fundamental result in the simplex algorithm.

Unlike linear programs, the optimal solution of nonlinear programs does not have to be a vertex of the polyhedron. Although there exist general solution algorithms for linear programs, for nonlinear programs more advanced solution algorithms are required. For example, the Lagrange Multiplier method [17] can be used to solve optimization problems

with equality constraints and the Karush-Kuhn-Tucker approach [16] can also be used to solve convex optimization problems. There are also special solution algorithms for special classes of problems, such as fractional and quadratic programs.

In the case of nonlinear objective function we are looking for its *stationary points*, that are the points, where gradient vanishes, or it fails to exist, over an open set domain. First, we find stationary points on the interior points of the feasible region. Next, we find stationary points on interior points of the faces of the feasible region, then on interior points of the edges (e.g., line segments) of the feasible region. Finally, by functional evaluation of the objective function at the stationary points and at the vertices of the feasible region the global optimal solution is found. Therefore, in solving an n dimension problem, we solve some unconstrained optimization problems in n, $n-1, \ldots, 1$ dimensions. Thus, removing the constraints by the proposed algorithm reduces the constrained optimization to unconstrained problems which can be more easily dealt with.

The following provides an overview of the algorithm's process strategy:

Phase 1: Find the V-representation of the feasible region (the polyhedron). If the objective function is nonlinear, find the edges and faces too.

Phase 2: Construct the parametric representation of the objective function.

Phase 3: If $f(\mathbf{x})$ is a linear function, then perform step a) otherwise perform step b).

a) If we are looking for the maximum (minimum) and some coefficient at μ_i (if any) is positive (negative) then the problem $\mathbf{P}$ has no solution. Otherwise the maximum (minimum) value is obtained at the vertex with the largest (smallest) coefficient.

b) Find the stationary points of the parametric version of the objective function over the interior, the faces, and the edges of the feasible region. Evaluate the objective function at the stationary points and the vertices. Select the global optimal point.

Since the first step of the algorithm is seems to be time consuming, it deserves special treatment. An efficient algorithm that implements it is described in next section.

3 Finding the vertices, extreme rays, edges and faces of the polyhedron

There are essentially two main approaches to the problem of generating all the vertices of the polyhedron, both with the origins in the 1950s. The double description method [23] involves building the polytope sequentially by adding the defining inequalities one at a time. Recent algorithms and practical implementations of this method have been developed by Fukuda and the others [12, 18]. This method seems very powerful for degenerate polytopes but has the disadvantage that it can require a lot of memory.

The second method for finding all the vertices and extreme rays of the polyhedron involves pivoting around the skeleton of the polytope. An efficient method using this approach is the reverse search method by Avis and Fukuda [10] and the revisited version [8].

If the intention is to find the solution of linearly constraint optimization problem it is not enough to determine only the V-representation of the polyhedron since the optimal value can appear at any point of the polyhedron not only in a vertex as in the case of linear programming. It means that in addition to vertices and extreme rays edges and faces of the polyhedron have to be found. They can be found with the modification of the algorithm for finding V-representation described in [13] that is based on the work of [15]. The algorithm belongs to the second group of the algorithms mentioned above since it systematically finds all vertices and extreme rays by using the breath first search method of the graph defined by the skeleton of the polyhedron. We suggest some modifications of that algorithm that enable finding the edges and faces of the polyhedron. The algorithm with the modifications is described below. The choice of the algorithm is made upon the fact that it enables us to extend it with the steps needed for finding the edges and faces.

Let S be the polyhedron expressed in H-representation. Since some inequalities can be explicit lower or upper bounds we can use a little different representation after adding slack variables as described in [13]:

$$\mathbf{Ax} = \mathbf{b}, \quad \mathbf{l} \leq \mathbf{x} \leq \mathbf{u}$$

where $\mathbf{l}$ and $\mathbf{u}$ are vectors describing lower and upper bounds of the variables.

If we investigate basic feasible solutions of above system we can see that every such solution x^* is fully determined by specifying which of the variables are basic, which nonbasic variables are at their lower bounds, and which nonbasic variables are at their upper

bounds. The resulting partition of x^* into three parts will be referred to as a basic feasible partition. If $\mathbf{A}$ has size $m \times n$ of a full row rank, then the basic feasible partition may be represented by the vector $\mathbf{p} = (p_1, p_2, \ldots, p_n)$ such that

$$\begin{aligned} p_k &= 1 \quad \text{if} \quad x_k \text{ is basic} \\ p_k &= 0 \quad \text{if} \quad x_k \text{ is nonbasic and } x_k^* = l_k < u_k \\ p_k &= 2 \quad \text{if} \quad x_k \text{ is nonbasic and } x_k^* = u_k \end{aligned}$$

Two basic feasible partitions are said to be neighbors of each other if they can be obtained from each other by a single pivot. It is assumed that the reader is familiar with the terms used by the algorithm that have the same meaning as by Revisited Simplex Method, where *basis matrix* $\mathbf{B}$ denotes a $m \times m$ matrix, which contains those columns of matrix $\mathbf{A}$ that correspond to basic variables. Similarly $\mathbf{x_B}$ is a m-dimensional vector composed of the values of basic variables in corresponding vertex. All the other details are described in [13].

The algorithm starts with a basic feasible partition $\mathbf{p}^*$ and produces all of it's neighbors. Then the neighbors of each new feasible partition are found until all the vertices are produced. To produce all the neighbors of a basic feasible partition $\mathbf{p}$ we have to consider only each nonbasic variable x_i with $l_i < u_i$ in turn to replace its value x_i^* by $x_i^* + t$ or $x_i^* - t$ with t as large as feasibility allows. If t can be arbitrarily large, then we discover an extreme ray that is the neighbor of $\mathbf{p}$; otherwise, we discover one or more basic feasible partitions that are neighbors of $\mathbf{p}$. The neighbors are found by pivoting that is similar as by the Revised Simplex Method with the exception that this algorithm finds all the neighbors of some vertex not only the most promising one.

More formally the algorithm, which has been modified by adding step 2 c) for finding the edges too, may be described as follows:

1. Set $\mathbf{p_1} = \mathbf{p}^*$, $M = 1$, $N = 0$, $L = 0$, $k = 1$, and let $\mathbf{v_1}$ be the basic feasible solution determined by $\mathbf{p_1}$.

2. Produce all the neighbors of $\mathbf{p_k}$.

   (a) Whenever a basic feasible partition $\mathbf{p}$ is produced that is not on the current list, set $M = M + 1$, $\mathbf{p_M} = \mathbf{p}$, $\mathbf{v_M} = \mathbf{v}$, where $\mathbf{v}$ is the corresponding basic feasible solution.

   (b) Whenever an extreme ray $\mathbf{w}$ is produced that is not on the current list, set $N = N + 1$, $\mathbf{w_N} = \mathbf{w}$.

(c) Whenever an edge $\mathbf{e}$ is produced that is not on the current list, set $L = L + 1$, $\mathbf{e_L} = \mathbf{e}$, and continue by producing the neighbors of $\mathbf{p_k}$.

3. If $k < M$ set $k = k + 1$ and return to step 2.

The algorithm finds the vertices, extreme rays and edges of the polyhedron. But in the case of nonlinear programming the faces of dimension $n-1$, $n-2, \ldots, 2$ have to be found too. It is easy to determine $(n-1)$ dimensional faces since we only have to find those basic feasible partitions that have some component equal to 0 or to 2. The basic feasible partition that have the k-th component equal to 0 correspond to the equation $x_k = l_k$ and those that have the k-th component equal to 2 correspond to the equation $x_k = u_k$.

The $(n-2)$ dimensional faces are determined by the basic feasible partitions that have two same components (equal to 0 or to 2) and the lower dimensional faces in similar way.

Example 2:

Suppose we want to find the vertices, extreme rays, edges and faces of the polyhedron that is drawn in Figure 2 defined by

$$\begin{aligned} x_1 + x_2 + x_3 &\leq 10 \\ 3x_1 + x_3 &\leq 24 \\ x_1, x_2, x_3 &\geq 0 \end{aligned}$$

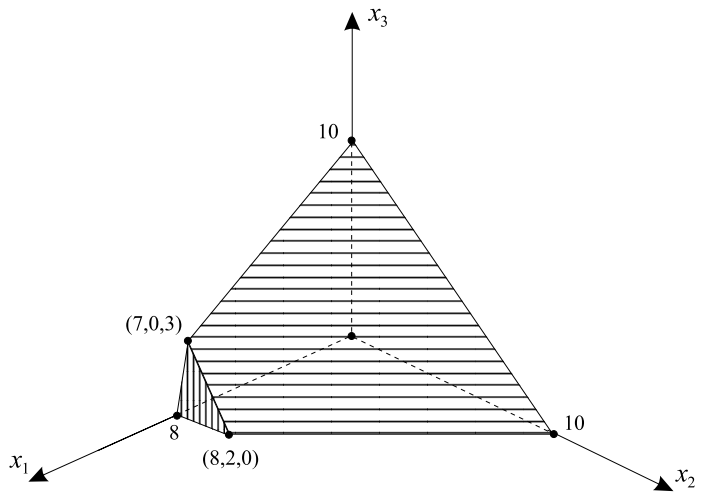

Figure 2: Feasible region (the polyhedron) for the Example 2

By adding slack variables x_4 and x_5 we can convert the problem to

$$\mathbf{Ax} = \mathbf{b}, \quad \mathbf{l} \leq \mathbf{x} \leq \mathbf{u}$$

where

$$\begin{aligned} \mathbf{A} &= \begin{bmatrix} 1 & 1 & 1 & 1 & 0 \\ 3 & 0 & 1 & 0 & 1 \end{bmatrix} \qquad \mathbf{b} = \begin{bmatrix} 10 \\ 24 \end{bmatrix} \\ \mathbf{l} &= [0,0,0,0,0]^T \qquad \mathbf{u} = [\infty,\infty,\infty,\infty,\infty]^T \end{aligned}$$

Since $\mathbf{v} = [0,0,0,10,24]^T$ is the basic feasible solution determined by the basic feasible partition $\mathbf{p}^* = [0,0,0,1,1]$ we can start with the first step of the algorithm by

$$\mathbf{v_1} = \mathbf{v}, \quad \mathbf{p_1} = \mathbf{p}^*, \quad M = 1, \quad k = 1$$

First we have to find all the neighbors of $\mathbf{p_1}$ with corresponding basic feasible solutions or extreme rays and edges. The basis matrix is

$$\mathbf{B} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Since we have three possible entering variables x_1, x_2 and x_3, three systems $\mathbf{Ad} = \mathbf{a_i}$, $i = 1,2,3$, where $\mathbf{a_i}$ denotes the i-th column of matrix $\mathbf{A}$, have to be solved. For each solution d we have to find the largest t for which $\mathbf{x_B} - t\mathbf{d} \geq 0$.

- For the entering variable x_1 we get the solution $\mathbf{d} = [1,3]^T$ and $t = 8$. It means that we have new basic feasible solution $\mathbf{v_2} = [8,0,0,2,0]^T$, the basic feasible partition $\mathbf{p_2} = [1,0,0,1,0]$ and the edge $\mathbf{e_1} = (\mathbf{v_1}, \mathbf{v_2})$ is added to the list of edges.
- For the entering variable x_2 we get the new basic feasible solution $\mathbf{v_3} = [0,10,0,0,24]^T$, the basic feasible partition $\mathbf{p_3} = [0,1,0,0,1]$ and the edge $\mathbf{e_2} = (\mathbf{v_1}, \mathbf{v_3})$ is added to the list of edges.
- For the entering variable x_3 we get the basic feasible solution $\mathbf{v_4} = [0,0,10,0,14]^T$, the basic feasible partition $\mathbf{p_4} = [0,0,1,0,1]$ and the edge $\mathbf{e_3} = (\mathbf{v_1}, \mathbf{v_4})$ is added to the list of edges.

We have found all the neighbors of the first basic feasible partition that determine all the neighbor vertices of the vertex $[0,0,0]^T$. These are the vertices

$$[8,0,0]^T, \quad [0,10,0]^T, \quad [0,0,10]^T.$$

Now we have to proceed by finding all the neighbors of the basic feasible partition $\mathbf{p_2}$ (vertex $[8,0,0]^T$). Since x_1 and x_4 are the basic variables the basis matrix is

$$\mathbf{B} = \begin{bmatrix} 1 & 1 \\ 3 & 0 \end{bmatrix}$$

- For the entering variable x_2 we get

$$\begin{aligned} \mathbf{v_5} &= [8,2,0,0,0]^T \\ \mathbf{p_5} &= [1,1,0,0,0] \\ \mathbf{e_4} &= (\mathbf{v_2}, \mathbf{v_5}) \end{aligned}$$

- For the entering variable x_3 we get

$$\begin{aligned} \mathbf{v_6} &= [7,0,3,0,0]^T \\ \mathbf{p_6} &= [1,0,1,0,0] \\ \mathbf{e_5} &= (\mathbf{v_2}, \mathbf{v_6}) \end{aligned}$$

- For the entering variable x_5 we do not get any new vertex or edge.

In the next step of the algorithm we are looking for the neighbors of basic feasible partition $\mathbf{p_3}$ ($k = 3$). Actually all the vertices of the polyhedron have been found so far and since the polyhedron is bounded it has no extreme rays. But the algorithm does not terminate until $k = M$. Therefore in the following steps only the new edges of the polyhedron will be added.

While looking for the neighbors of $\mathbf{p_3}$ edges

$$\mathbf{e_6} = (\mathbf{v_3}, \mathbf{v_5}), \quad \text{and} \quad \mathbf{e_7} = (\mathbf{v_3}, \mathbf{v_4})$$

are added to the list of edges.

Now we have to find the neighbors of three remaining basic feasible partitions $\mathbf{p_4}$, $\mathbf{p_5}$ and $\mathbf{p_6}$. The edges

$$\mathbf{e_8} = (\mathbf{v_4}, \mathbf{v_6}), \quad \text{and} \quad \mathbf{e_9} = (\mathbf{v_5}, \mathbf{v_6})$$

are found and added to the list of edges.

We have found so far all the vertices and edges of the polyhedron. If we summarize the vertices are

$$\mathbf{v_1} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \ \mathbf{v_2} = \begin{bmatrix} 8 \\ 0 \\ 0 \end{bmatrix}, \ \mathbf{v_3} = \begin{bmatrix} 0 \\ 10 \\ 0 \end{bmatrix}, \ \mathbf{v_4} = \begin{bmatrix} 0 \\ 0 \\ 10 \end{bmatrix}, \ \mathbf{v_5} = \begin{bmatrix} 8 \\ 2 \\ 0 \end{bmatrix}, \ \mathbf{v_6} = \begin{bmatrix} 7 \\ 0 \\ 3 \end{bmatrix}$$

and the edges

$$\mathbf{e_1} = (\mathbf{v_1}, \mathbf{v_2}), \quad \mathbf{e_2} = (\mathbf{v_1}, \mathbf{v_3}), \quad \mathbf{e_3} = (\mathbf{v_1}, \mathbf{v_4}),$$
$$\mathbf{e_4} = (\mathbf{v_2}, \mathbf{v_5}), \quad \mathbf{e_5} = (\mathbf{v_2}, \mathbf{v_6}), \quad \mathbf{e_6} = (\mathbf{v_3}, \mathbf{v_5}),$$
$$\mathbf{e_7} = (\mathbf{v_3}, \mathbf{v_4}), \quad \mathbf{e_8} = (\mathbf{v_4}, \mathbf{v_6}), \quad \mathbf{e_9} = (\mathbf{v_5}, \mathbf{v_6})$$

The faces of the polyhedron can be determined too if we examine the found basic feasible partitions. By finding all the basic feasible partitions $\mathbf{p_i}$ which have the first component equal to 0 the face that responds to the equation $x_1 = 0$ is determined. In our case the corresponding basic feasible partitions are $\mathbf{p_1}$, $\mathbf{p_3}$ and $\mathbf{p_4}$. It means that the first face of the polyhedron can be expressed as

$$\mathbf{f_1} = \{\mathbf{x};\ \mathbf{x} = \lambda_1\mathbf{v_1} + \lambda_3\mathbf{v_3} + \lambda_4\mathbf{v_4},\ \lambda_1, \lambda_3, \lambda_4 \geq 0,\ \lambda_1 + \lambda_3 + \lambda_4 = 1\}$$

Similarly the other faces can be determined. Basic feasible solutions with the second component equal to 0 (respond to $x_2 = 0$) determine the face

$$\mathbf{f_2} = \{\mathbf{x};\ \mathbf{x} = \lambda_1\mathbf{v_1} + \lambda_2\mathbf{v_2} + \lambda_4\mathbf{v_4} + \lambda_6\mathbf{v_6},\ \lambda_1, \lambda_2, \lambda_4, \lambda_6 \geq 0,\ \lambda_1 + \lambda_2 + \lambda_4 + \lambda_6 = 1\}$$

Basic feasible solutions with the third component equal to 0 (respond to $x_3 = 0$) determine the face

$$\mathbf{f_3} = \{\mathbf{x};\ \mathbf{x} = \lambda_1\mathbf{v_1} + \lambda_2\mathbf{v_2} + \lambda_3\mathbf{v_3} + \lambda_5\mathbf{v_5},\ \lambda_1, \lambda_2, \lambda_3, \lambda_5 \geq 0,\ \lambda_1 + \lambda_2 + \lambda_3 + \lambda_5 = 1\}$$

The remaining two faces are

$$\mathbf{f_4} = \{\mathbf{x};\ \mathbf{x} = \lambda_3\mathbf{v_3} + \lambda_4\mathbf{v_4} + \lambda_5\mathbf{v_5} + \lambda_6\mathbf{v_6},\ \lambda_3, \lambda_4, \lambda_5, \lambda_6 \geq 0,\ \lambda_3 + \lambda_4 + \lambda_5 + \lambda_6 = 1\}$$

and

$$\mathbf{f_5} = \{\mathbf{x};\ \mathbf{x} = \lambda_2\mathbf{v_2} + \lambda_5\mathbf{v_5} + \lambda_6\mathbf{v_6},\ \lambda_2, \lambda_5, \lambda_6 \geq 0,\ \lambda_2 + \lambda_5 + \lambda_6 = 1\}$$

The time complexity of the above algorithm is not high. The main computational burden involved in each execution of step 2 comes from solving the $n - m$ systems $\mathbf{Bd} = \mathbf{a}$ with $\mathbf{B}$ standing for basis matrix and $\mathbf{a}$ running through all the nonbasic columns. This task requires no more than $m^2(n - 2m/3)$ multiplications [13]. Finding out if the most recently produced vertex, extreme ray or edge is already on the list could be time consuming but the data structures such as balanced trees, such as those developed in network solution algorithms, can be used for the representation of the list. So the total running time of the algorithm is proportional to m^2nM.

There are many other techniques for finding the vertices of the polyhedron [9]. Some algorithms that ware mentioned above are faster but they do not find the edges and faces. They can be used in the case if f is linear since then only vertices have to be found. Another algorithm for face and edge identifications can be found in [6].

4 Numerical examples

In the following two examples, we demonstrate how the proposed solution algorithm can be used to solve general linear and nonlinear programs over polyhedron S.

Example 3:

The following is a "test" problem:

$$\begin{aligned} \text{Min} \quad & -4x_1^3 + 3x_1 - 6x_2 \\ \\ \text{subject to:} \quad & x_1 + x_2 \leq 4 \\ & 2x_1 + x_2 \leq 5 \\ & -x_1 + 4x_2 \geq 2 \\ & x_1, x_2 \geq 0 \end{aligned}$$

This test problem is a nonconvex program from [26, pages 133-136], that was chosen as an example to demonstrate the difficulties for finding the global solution for this types of problems by any existing methods. In [26] the objective function behavior over the feasible region, that is drawn in Figure 3, is investigated by drawing some iso-values curves of the objective function to illustrate the difficulties involved in solving general optimization problems even for this small size test problem.

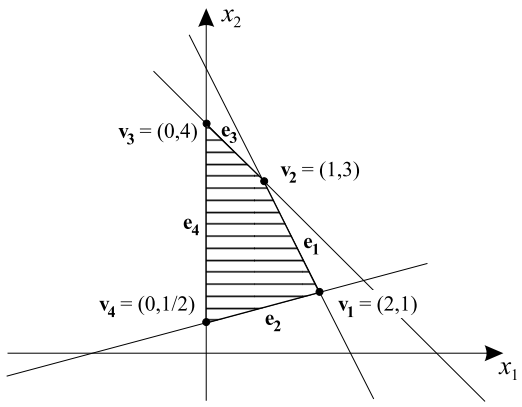

Figure 3: Feasible region for the example 3

In [26] an excellent and detailed discussion of serious difficulties is provided, such as:

- nonconvex programs are more difficult to solve than convex programs,

- even specialized algorithms such as separable programming methods can guarantee no more than a local optima to such problems, and
- although such problems may be handled by interior point methods such as the penalty functions algorithm, it is computationally expensive.

Accordingly [26, pages 133-136] detects the optimal solution only by graphical observation and not by any algorithmic approach. We solve this test problem, without referring to any graphical method using the proposed algorithm.

Applying the results of Section 3, the vertices of the feasible region are:

$$\mathbf{v_1} = \begin{bmatrix} 2 \\ 1 \end{bmatrix}, \ \mathbf{v_2} = \begin{bmatrix} 1 \\ 3 \end{bmatrix}, \ \mathbf{v_3} = \begin{bmatrix} 0 \\ 4 \end{bmatrix}, \ \mathbf{v_4} = \begin{bmatrix} 0 \\ 1/2 \end{bmatrix}$$

and the edges

$$\begin{aligned} \mathbf{e_1} &= (\mathbf{v_1}, \mathbf{v_2}), \quad \mathbf{e_2} = (\mathbf{v_1}, \mathbf{v_4}), \\ \mathbf{e_3} &= (\mathbf{v_2}, \mathbf{v_3}), \quad \mathbf{e_4} = (\mathbf{v_3}, \mathbf{v_4}). \end{aligned}$$

The parametric representation of the feasible region is:

$$\begin{aligned} \{(x_1, x_2); \quad & (x_1, x_2) = (2\lambda_1 + \lambda_2, \lambda_1 + 3\lambda_2 + 4\lambda_3 + 1/2\lambda_4), \\ & \lambda_1, \lambda_2, \lambda_3, \lambda_4 \geq 0, \ \lambda_1 + \lambda_2 + \lambda_3 + \lambda_4 = 1\}. \end{aligned}$$

By substituting the parametric version of the feasible region into the objective function we obtain:

$$\begin{aligned} f(\lambda) &= -4(2\lambda_1 + \lambda_2)^3 + 3(2\lambda_1 + \lambda_2) - 6(\lambda_1 + 3\lambda_2 + 4\lambda_3 + 1/2\lambda_4) \\ &= -32\lambda_1^3 - 4\lambda_2^3 - 48\lambda_1^2\lambda_2 - 24\lambda_2^2\lambda_1 - 15\lambda_2 - 24\lambda_3 - 3\lambda_4 \end{aligned}$$

over the closed domain $\lambda_1, \lambda_2, \lambda_3, \lambda_4 \geq 0$, and $\lambda_1 + \lambda_2 + \lambda_3 + \lambda_4 = 1$. Since gradient is not defined over a closed set, we need to treat interior points of the feasible region separately from edges and vertices. By substituting $\lambda_4 = 1 - \lambda_1 - \lambda_2 - \lambda_3$ we get the objective function over the interior points of the feasible region:

$$f(\lambda_1, \lambda_2, \lambda_3) = -32\lambda_1^3 - 4\lambda_2^3 - 48\lambda_1^2\lambda_2 - 24\lambda_2^2\lambda_1 + 3\lambda_1 - 12\lambda_2 - 21\lambda_3 - 3.$$

The gradient does not vanish since the derivative with respect to λ_3 is -21.

The next step to do is to find stationary points over the four edges. The parametric representation of the edge $\mathbf{e_1}$ is

$$\mathbf{e_1} = \{(x_1, x_2); \ (x_1, x_2) = (2\lambda_1 + \lambda_2, \lambda_1 + 3\lambda_2), \ \lambda_1, \lambda_2 \geq 0, \ \lambda_1 + \lambda_2 = 1\}.$$

After substituting $\lambda_2 = 1 - \lambda_1$, the parametric objective function is:

$$f(\lambda_1) = -4\lambda_1^3 - 12\lambda_1^2 - 3\lambda_1 - 19$$

over the open set $0 < \lambda_1 < 1$, which has a stationary point at $\lambda_1 = 0.118$ with $f(\lambda_1) = -18.82$.

Representation of $f(\lambda)$ on the interior of the edge $\mathbf{e_2}$ joining the vertices $\mathbf{v_1}$ and $\mathbf{v_4}$ after substituting $\lambda_4 = 1 - \lambda_1$ is expressed by

$$f(\lambda_1) = -32\lambda_1^3 + 3\lambda_1 - 3$$

over the open set $0 < \lambda_1 < 1$, which has a stationary point at $\lambda_1 = \sqrt{1/32}$ with $f(\lambda_1) = -2.65$.

Similarly we get the stationary point on the edge $\mathbf{e_3}$ joining the vertices $\mathbf{v_2}$ and $\mathbf{v_3}$ for $\lambda_2 = \sqrt{3/4}$ with $f(\lambda_2) = -18.80$. On the remaining edge $\mathbf{e_4}$ is no stationary point.

The values of the objective function at the vertices are presented in Table 1.

	λ_1	λ_2	λ_3	λ_4	$f(\mathbf{x})$
v_1	1	0	0	0	-32
v_2	0	1	0	0	-19
v_3	0	0	1	0	24
v_4	0	0	0	1	-3

Table 1: Function values at the vertices for Example 3

Now, comparing all the functional evaluations of $f(\lambda)$ at all the stationary points and at the vertices, the optimal solution has $\lambda_1 = 1$ and all other $\lambda_i = 0$. This gives the global optimal solution ($x_1 = 2$, $x_2 = 1$) yielding the global optimal value of -32.

Next example shows the use of proposed algorithm in the case of unbounded feasible region.

Example 4:

The following linear program is from [27, page 155] which is solved therein with Simplex Method by first converting the unrestricted variable to two non-negative variables [11].

The problem is

$$\text{Max} \qquad 30x_1 - 4x_2$$

$$\begin{aligned} \text{subject to:} \quad 5x_1 - x_2 &\leq 30 \\ x_1 &\leq 5 \\ x_1 &\geq 0 \\ x_2 &\text{ is unrestricted in sign} \end{aligned}$$

Figure 1 in Example 1 presents feasible region, which is the polyhedron with two vertices and one extreme ray:

$$\mathbf{v_1} = \begin{bmatrix} 0 \\ -30 \end{bmatrix}, \ \mathbf{v_2} = \begin{bmatrix} 5 \\ -5 \end{bmatrix}, \ \mathbf{w} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}.$$

It's V-representation is given by

$$\{(x_1, x_2);\ (x_1, x_2) = (5\lambda_2, -30\lambda_1 - 5\lambda_2 + \mu),\ \lambda_1, \lambda_2, \mu \geq 0,\ \lambda_1 + \lambda_2 = 1\}$$

and parametric representation of objective function by

$$\begin{aligned} f(\lambda_1, \lambda_2, \mu) &= 150\lambda_2 + 120\lambda_1 + 20\lambda_2 - 4\mu \\ &= 120\lambda_1 + 170\lambda_2 - 4\mu. \end{aligned}$$

Since we are looking for maximum and the coefficient at μ is negative, the optimal value occurs by setting $\lambda_2 = 1$ and $\lambda_1 = \mu = 0$. It means that the optimal value of objective function 170 is obtained at the vertex $\mathbf{v_2}$, which is $x_1 = 5$, and $x_2 = -5$.

Suppose we wish to maximize $x_1 - x_2^2 + 4x_2$, subject to same feasible region. The gradient of the objective function is $(1, -2x_2 + 4)$, which is nonzero. Therefore, there is no stationary point in the interior of the feasible region. Also, it can be shown that there is no stationary point on the edge joining the two vertices $\mathbf{v_1}$ and $\mathbf{v_2}$.

To find the stationary points on the interior points of extreme ray rooted at $\mathbf{v_1}$ in the direction of $\mathbf{w}$ we first have to construct its parametric version as:

$$\{(x_1, x_2);\ (x_1, x_2) = (0, -30) + \mu(0, 1) = (0, -30 + \mu),\ \mu \geq 0\}.$$

Then we have to construct the parametric objective function

$$f(\mu) = -(-30 + \mu)^2 + 4(-30 + \mu) = -\mu^2 + 64\mu - 1020$$

over the open domain $\mu > 0$. The derivative vanishes at $\mu = 32$, therefore a stationary point is at $(0, 2)$, with the objective function value of 4.

Similarly, to find the stationary points on the interior points of extreme ray rooted at vertex $\mathbf{v_2}$ in the direction of $\mathbf{w}$ we have to construct its parametric representation as:

$$\{(x_1, x_2);\ (x_1, x_2) = (5, -5) + \mu(0, 1) = (5, -5 + \mu),\ \mu \geq 0\}.$$

Now we can construct the parametric objective function, which is

$$f(\mu) = 5 - (-5 + \mu)^2 + 4(-5 + \mu) = -\mu^2 + 14\mu - 40$$

over the open domain $\mu > 0$. The derivative vanishes at $\mu = 7$, which implies that a stationary point is at $(5, 2)$, with the objective function value of 9.

The objective function values at both vertices are -1020 at $\mathbf{v_1}$ and -40 at $\mathbf{v_2}$. Since we are looking for a maximum, the global optimal solution is: $x_1 = 5$ and $x_2 = 2$ with the optimal value 9.

5 Conclusion

We have presented a new solution algorithm for the general linearly constrained optimization problems with continuous objective function. For a polyhedron specified by a set of linear equalities and/or inequalities, the proposed solution algorithm utilizes its parametric representation. This parametric representation of the feasible region enables us to solve a large class of optimization problems including fractional linear programming and quadratic linear programming. The key to this generalized solution algorithm is that the constrained optimization problem is converted to an unconstrained optimization problem through a parametric representation of the feasible region.

It favorably compares with other methods for this type of problems. The proposed algorithm, unlike other general purpose solution methods, such as KKT conditions, guarantees global optimal solution, it has simplicity, potential for wide adaptation, and deals with all cases. However, this does not imply that all distinction among problems should be ignored. One must incorporate the special characteristic of the problem to modify the proposed algorithm in solving them.

While the Lagrange and KKT (penalty-based) methods "appear" to remove the constraints by using a linear (or nonlinear) combination of the constraints in a penalty function, the proposed solution algorithm, however, uses the linear convex combination of vertices to remove the constraints. The main drawback for the proposed algorithm is that

all the vertices of the feasible region have to be found. However, there are quite efficient techniques to apply which may be in the Reference section.

Some areas for future research include development of possible refinements such as, decomposition by way of not the convex combination of all vertices, but some, say $k, k > n+1$. An immediate work is development of an efficient computer code to implement the approach, and performing a comparative computational study. A convincing evaluation for the reader would be the application of this approach to a problem he/she has solved by any other method.

References

[1] W. Altherr, An algorithm for enumerating the vertices of a convex polyhedron, Computing, 15 (1975) 181-193.

[2] H. Arsham, A comprehensive simplex-like algorithm for network optimization and perturbation analysis, Optimization, 32 (1995) 211-267.

[3] H. Arsham, A refined algebraic method for linear programs and their geometric representation, Congressus Numerantium 120 (1996) 65-81.

[4] H. Arsham, Affine geometric method for linear programs, Journal of Scientific Computing, 12 (1997) 289-303.

[5] H. Arsham, Initialization of the simplex algorithm: An artificial-free approach, SIAM Review, 39 (1997) 736-744.

[6] H. Arsham, Computational geometry and linear programs, International Journal of Mathematical Algorithms, 1 (1999) 251-266.

[7] H. Arsham , M. Gradišar, M. Oblak, A direct solution algorithm for linear systems of inequalities with application to optimization, to appear.

[8] D. Avis, Computational experience with the reverse search vertex enumeration algorithm, Optimization Methods and Software, to appear.

[9] D. Avis, D. Bremner, R. Seidel, How good are convex hull algorithms?, Computational Geometry: Theory and Applications, 7 (1997) 265-301.

[10] D. Avis, K. Fukuda, A pivoting algorithm for convex hulls and vertex enumeration of arrangements and polyhedra, Discrete and Computational Geometry, 8 (1992) 295-313.

[11] M. Best, R. Caron, The simplex method and unrestricted variables, Journal of Optimization Theory and Applications, 45 (1985) 33-39.

[12] D. Bremner, K. Fukuda,A. Marzetta, Primal-dual methods for vertex and facet enumeration, In Proc. 13th Annu. ACM Sympos. Comput. Geom. (1997) 49-56.

[13] V. Chvatal, Linear Programming, Freeman and Co., New York, 1983.

[14] E. Dravnieks, J. Chinneck, Formulation assistance for global optimization problems, Computers & Operations Research 24 (1997) 1157-1168.

[15] M.E. Dyer, L.G Proll, An algorithm for determing all extreme points of a convex polytope, Mathematical Programming 12 (1977) 81-96.

[16] A. Fiacco, G. McCormick, Nonlinear Programming, Sequential Unconstrained Minimization Techniques, Classics in Applied Mathematics, SIAM, Philadelphia, PA, 1990.

[17] R. Fletcher, Practical Methods of Optimization, Wiley, New York, 1987.

[18] K. Fukuda, T. Liebling, F. Margot, Analysis of backtrack algorithms for listing all vertices and all faces of a convex polyhedron, Computational Geometry: Theory and Applications, 8 (1997) 1-12.

[19] L. Heeseok, Simultaneous determination of capacities and load in parallel M/M/1 queues, European Journal of Operational Research, 73(1)(1994) 95-102.

[20] F. Hillier, G. Lieberman, Introduction To Operations Research, 6th edition, McGraw-Hill Co., New York, NY, 1995.

[21] L. Kin-nam , L. Pui-lam, T. Ka-kit, A mathematical programming approach to clusterwise regression model and its extensions, European Journal of Operational Research, 116(3)(1999) 640-652.

[22] C. Kelley, Iterative Methods for Linear and Nonlinear Equations, SIAM, Philadelphia, PA, 1995.

[23] T.S. Motzkin, H. Raifa, G.L. Thompsonm, R.M. Thrall, The double description method, Annals of Math. Studies, 8 (1953) 51-73.

[24] C. Nilotpal, Some results concerning post-infeasibility analysis, European Journal of Operational Research, 73(1)(1994) 139-143.

[25] J. Pintér, Continuous global optimization: an introduction to models, solution approaches, tests and applications, 2, ITORMS, 1999.

[26] H. Williams, Model Building in Mathematical Programming, Wiley, Chichester, UK, 1999.

[27] W. Winston, Introduction to Mathematical Programming: Applications and Algorithms, Wadsworth Pub. Co., New York, NY, 1997.

[28] T. Yamada, J. Yoruzuya K. Kataoka, Enumerating extreme points of a highly degenerate polytope, Computers & Operations Research, 21 (1994) 397-410.

[29] G. Ziegler , Lectures on Polytopes, Springer-Verlag, New York, NY, 1995.