A Variant of the Outer Approximation Method for Globally Minimizing a Class of Composite Functions

Takahito Kuno*

May 26, 1995

ISE-TR-95-120

Institute of Information Sciences and Electronics
University of Tsukuba
Tsukuba, Ibaraki 305, Japan
Phone: +81-298-53-5540, Fax: +81-298-53-5206, E-mail: takahito@is.tsukuba.ac.jp

* The author was partly supported by Grant-in-Aid for Scientific Research of the Ministry of Education, Science and Culture, Grant No. (C)07680447.

A variant of the outer approximation method for globally minimizing a class of composite functions

Takahito Kuno*

Institute of Information Sciences and Electronics

University of Tsukuba

May 1995

Abstract. In this paper, we consider a constrained optimization problem whose objective function is a composition of two functions $g : \mathbf{R}^n \to \mathbf{R}^p$ and $f : \mathbf{R}^p \to \mathbf{R}^1$. We show that a variant of the outer approximation method generates a globally ϵ-minimum point of $f \circ g = f(g(\cdot))$ on a convex set after finitely many iterations, if g is convex and f is continuous and coordinatewise increasing. Preliminary experiments indicate that the proposed algorithm is reasonably practical for two types of multiplicative programs if p is less than four.

Key words: Global optimization, composite function, outer approximation method, multiplicative program, multiple objective decision making.

1. Introduction

In a series of articles [5 – 11], Konno et al. studied multiplicative programming problems, whose objective functions can be expressed by the product of some convex functions. Although the class is a typical nonconvex program and hence has multiple local minima [6], one can generate a global minimum rather efficiently if the number of convex functions involved in the product term is much less than that of variables. Tuy [18] and Sniedovich et al. [14] showed that this nice characteristic is mainly due to a low-rank property possessed by multiplicative functions. In other words, minimizing a composition $f \circ g = f(g(\cdot))$ of two functions $g : \mathbb{R}^n \to \mathbb{R}^p$ and $f : \mathbb{R}^p \to \mathbb{R}^1$ over a convex set $X \subset \mathbb{R}^n$ is possibly as efficient as minimizing the product of p convex functions, if all components of g are convex on $\mathbb{R}^n$ and f *is coordinatewise increasing and quasiconcave* on $\{g(x) \mid x \in X\}$.

As stated in [7], the most important application of multiplicative programs is multiple objective decision making. When several objectives without a common scale need optimizing simultaneously, a handy approach is to optimize the product of these objectives (see e.g. [4]). This approach, however, assumes implicitly that the utility of the

*The author was partially supported by Grand-in-Aid for Scientific Research of the Ministry of Education, Science and Culture, Grant No. (C)07680447.

decision maker is quasiconcave on his criterion space, though the shape of the utility function is in general difficult to specify except that it is coordinatewise increasing [15].

In this paper, we will develop a method for minimizing $f \circ g$ over a convex set X without assuming that f is quasiconcave. More precisely, *f is continuous and coordinatewise increasing but needs to be neither quasiconcave nor quasiconvex* on some open set including $\{g(x) \mid x \in X\}$. This class of functions $f \circ g$ is a generalization of multiplicative functions and also contains rank-p quasiconcave functions studied by Tuy [18]. We will show that a variant of the outer approximation method can generate a global ϵ-minimum of this nonconvex function after finitely many iterations. Preliminary experiments indicate that the proposed algorithm is reasonably practical when p is less than four, even though n exceeds one hundred. This fact has an important implication in multiple objective decision making, since the number of objectives is usually less than five, and less than three in most practical applications (see e.g. [3]).

The organization of the paper is as follows: In Section 2, we will transform the problem into a p-dimensional minimization problem whose objective function is f. In Section 3, to solve the resultant problem, we will propose a variant of the outer approximation method. Unlike the usual one, our algorithm approximates the feasible region by using the union of finitely many rectangles in $\mathbb{R}^p$. We will discuss possible improvements on the algorithm in Section 4, and report the results of computational experiments in Section 5.

2. Master Problem in the p-Dimensional Space

Suppose a continuous function $f : \mathbb{R}^p \to \mathbb{R}^1$ satisfies

$$f(y) < f(y + d) \quad \text{if} \quad d \in \mathbb{R}^p_+ \quad \text{and} \quad d \neq 0 \tag{2.1}$$

for any $y \in S$, where S is an open subset of $\mathbb{R}^p$ and $\cdot_+$ stands for the nonnegative orthant. The problem we consider in this paper is to minimize a composition of f and a convex function $g : \mathbb{R}^n \to \mathbb{R}^p$ over a convex set $X \subset \mathbb{R}^n$, i.e.,

$$\text{(P)} \qquad \left|\begin{array}{ll} \text{minimize} & f \circ g(x) = f(g(x)) \\ \text{subject to} & x \in X. \end{array}\right.$$

We assume for simplicity that X is compact. Therefore the jth component g_j of g achieves a minimum and a maximum over X at some $\check{x}^j$ and $\hat{x}^j$ respectively for $j = 1, \ldots, p$. Also the objective function of (P) has a globally optimal solution in X, since the composition of two continuous functions is continuous. We further assume that

$$\{y \in \mathbb{R}^p \mid g_j(\check{x}^j) \leq y_j \leq g_j(\hat{x}^j),\ j = 1, \ldots, p\} \subset S. \tag{2.2}$$

Hence it holds for any two feasible solutions x', x'' of (P) that

$$f \circ \boldsymbol{g}(\boldsymbol{x}') < f \circ \boldsymbol{g}(\boldsymbol{x}'') \text{ if } g_j(\boldsymbol{x}') < g_j(\boldsymbol{x}''), \ j = 1, \ldots, p,$$

under condition (2.1).

We first define a univariate function:

$$f_j(y) = f(g_1(\check{\boldsymbol{x}}^1), \ldots, g_{j-1}(\check{\boldsymbol{x}}^{j-1}), y, g_{j+1}(\check{\boldsymbol{x}}^{j+1}), \ldots, g_p(\check{\boldsymbol{x}}^p)) \tag{2.3}$$

for $j = 1, \ldots, p$. Let

$$v = \min\{f \circ \boldsymbol{g}(\boldsymbol{x}) \mid \boldsymbol{x} = \check{\boldsymbol{x}}^1, \ldots, \check{\boldsymbol{x}}^p\}. \tag{2.4}$$

Lemma 2.1. *Let $\boldsymbol{x}' \in X$. If $f \circ \boldsymbol{g}(\boldsymbol{x}') \leq v$, then*

$$g_j(\check{\boldsymbol{x}}^j) \leq g_j(\boldsymbol{x}') \leq \max\{y \mid f_j(y) \leq v\}, \ j = 1, \ldots, p. \tag{2.5}$$

Proof: Since $g_j(\check{\boldsymbol{x}}^j) \leq g_j(\boldsymbol{x}')$ for every j, we have

$$f_j(g_j(\boldsymbol{x}')) \leq f \circ \boldsymbol{g}(\boldsymbol{x}') \leq v, \ j = 1, \ldots, p,$$

by definition. Each f_j is continuous and from (2.1) strictly nondecreasing. Hence the second inequality of (2.5) holds. The first one is obvious. □

Let us introduce a vector $\boldsymbol{y} \in \mathbb{R}^p$ of additional p variables y_j's, and consider the following problem:

$$\left|\begin{array}{ll} \text{minimize} & f(\boldsymbol{y}) \\ \text{subject to} & \boldsymbol{x} \in X, \\ & \boldsymbol{g}(\boldsymbol{x}) - \boldsymbol{y} \leq \boldsymbol{0}, \ \ \boldsymbol{\ell} \leq \boldsymbol{y} \leq \boldsymbol{u}, \end{array}\right. \tag{2.6}$$

where $\boldsymbol{\ell} = (\ell_1, \ldots, \ell_p)^{\mathrm{T}}$, $\boldsymbol{u} = (u_1, \ldots, u_p)^{\mathrm{T}}$ and

$$\ell_j = g_j(\check{\boldsymbol{x}}^j), \ u_j = \max\{y \mid f_j(y) \leq v\}, \ j = 1, \ldots, p. \tag{2.7}$$

Lemma 2.2. *Let $(\boldsymbol{x}^*, \boldsymbol{y}^*)$ be an optimal solution of (2.6). Then $\boldsymbol{x}^*$ solves (P).*

Proof: Let $\boldsymbol{x}' \in X$ and assume that $f \circ \boldsymbol{g}(\boldsymbol{x}') < f \circ \boldsymbol{g}(\boldsymbol{x}^*)$. Let $\check{\boldsymbol{x}}^q = \operatorname{argmin}\{f \circ \boldsymbol{g}(\boldsymbol{x}) \mid \boldsymbol{x} = \check{\boldsymbol{x}}^1, \ldots, \check{\boldsymbol{x}}^p\}$. Then, by the previous lemma, $(\check{\boldsymbol{x}}^q, \boldsymbol{g}(\check{\boldsymbol{x}}^q))$ is feasible to (2.6) and satisfies

$$f \circ \boldsymbol{g}(\boldsymbol{x}') < f(\boldsymbol{y}^*) \leq f(\boldsymbol{g}(\check{\boldsymbol{x}}^q)) = v.$$

We again apply Lemma 2.1 and have $\boldsymbol{\ell} \leq \boldsymbol{g}(\boldsymbol{x}') \leq \boldsymbol{u}$. This is a contradiction, because $(\boldsymbol{x}', \boldsymbol{g}(\boldsymbol{x}'))$ is feasible to (2.6) and $f(\boldsymbol{g}(\boldsymbol{x}')) < f(\boldsymbol{y}^*)$ holds. □

Note that an ordinary line search algorithm can generate both the bounds ℓ_j and u_j of g_j. Since f_j is continuous and strictly nondecreasing, computing $u_j = \max\{y \mid f_j(y) \leq v\}$ amounts to minimizing a certain unimodal function of a single variable.

Let us denote by Z the feasible region of (2.6), i.e.,

$$Z = \{(\boldsymbol{x}, \boldsymbol{y}) \in \mathbb{R}^n \times \mathbb{R}^p \mid \boldsymbol{x} \in X,\ \boldsymbol{g}(\boldsymbol{x}) - \boldsymbol{y} \leq \boldsymbol{0},\ \boldsymbol{\ell} \leq \boldsymbol{y} \leq \boldsymbol{u}\}, \tag{2.8}$$

and let

$$Y = \{\boldsymbol{y} \in \mathbb{R}^p \mid \exists \boldsymbol{x} \in \mathbb{R}^n,\ (\boldsymbol{x}, \boldsymbol{y}) \in Z\}. \tag{2.9}$$

Then we have a problem in the p-dimensional space:

$$\text{(MP)} \quad \left| \begin{array}{ll} \text{minimize} & f(\boldsymbol{y}) \\ \text{subject to} & \boldsymbol{y} \in Y, \end{array} \right.$$

which is equivalent to (P) in the following sense:

Theorem 2.3. *Let $\boldsymbol{y}^*$ be an optimal solution of (MP). Then any $\boldsymbol{x}^*$ such that $(\boldsymbol{x}^*, \boldsymbol{y}^*) \in Z$ solves (P).*

Proof: It is obvious that any $(\boldsymbol{x}^*, \boldsymbol{y}^*) \in Z$ is an optimal solution of problem (2.6) if $\boldsymbol{y}^*$ is optimal to (MP). Hence $\boldsymbol{x}^*$ solves (P). □

By convexity of $\boldsymbol{g}$, we see that Z is a convex set in $\mathbb{R}^n \times \mathbb{R}^p$. The feasible region Y of (MP) is the orthogonal projection of Z onto the $\boldsymbol{y}$-space and hence a convex set in $\mathbb{R}^p$ [13]. We can also see that Y is compact as well as Z.

The above transformation from (P) into (MP) is based on a decomposition principle in global optimization [2, 12]. We refer to (MP) as the *master problem* of (P). If f is either convex or (quasi)concave, there are several solution methods for (MP) (e.g., [1, 16, 17]). These decomposition algorithms are known to be more promising than solving the original problem directly when p is much smaller than n. However, in applications such as multiple objective decision making, the shape of f is often difficult to specify except that it satisfies condition (2.1). In the rest of the paper, we will develop an algorithm for solving (MP), in which f is assumed to be neither convex nor (quasi)concave.

3. Outer Approximation Algorithm for the Master Problem

It is straightforward to see from (2.1) that there is a globally optimal solution $\boldsymbol{y}^*$ of the master problem (MP) among boundary points of the compact convex set Y. Hence outer approximation can still work for (MP) even though f is not (quasi)concave.

Let us denote

$$Y_0 = \{\boldsymbol{y} \in \mathbb{R}^p \mid \boldsymbol{\ell} \leq \boldsymbol{y} \leq \boldsymbol{u}\}. \tag{3.1}$$

Starting from Y_0 as the initial relaxation of Y, the class of outer approximation algorithms generates a sequence of relaxed problems (P_k), $k = 0, 1, \ldots,$ of the form:

$$
(\mathrm{P}_k) \quad \left| \begin{array}{ll} \text{minimize} & f(\boldsymbol{y}) \\ \text{subject to} & \boldsymbol{y} \in Y_k, \end{array} \right.
$$

where

$$
Y \subset Y_{k+1} \subset Y_k \subset \mathbb{R}^p, \quad k = 0, 1, \ldots. \tag{3.2}
$$

Let $\boldsymbol{y}^k$ be an optimal solution of (P_k). It follows from (2.1) and (3.2) that $\boldsymbol{y}^k \notin \operatorname{int} Y$ for every k, where $\operatorname{int}\cdot$ represents the set of interior points. If $\boldsymbol{y}^k$ happens to be a point of Y, then it is a globally optimal solution of (MP) and any $\boldsymbol{x}$ such that $(\boldsymbol{x}, \boldsymbol{y}^k) \in Z$ solves the original problem (P) (Theorem 2.3). Otherwise, we need to exclude some portion containing $\boldsymbol{y}^k$ from Y_k to obtain the next relaxation Y_{k+1} of Y. The usual procedures construct Y_{k+1} by adding some cutting-plane constraints to the system defining Y_k and generate a sequence of polytopes Y_k's. When f is (quasi)concave, we need only to search vertices of the polytope Y_k for an optimal solution $\boldsymbol{y}^k$ of (P_k). In our problem, however, such vertices might not provide an optimal solution. We will therefore present an alternative procedure for excluding $\boldsymbol{y}^k$ from Y_k in this section. The resultant Y_k turns out to be the union of finitely many rectangles in $\mathbb{R}^p$.

3.1. Approximation of the feasible region

Suppose an optimal solution $\boldsymbol{y}^k$ of the kth relaxed problem (P_k) is given. Regarding $\boldsymbol{y}^k$ as an ideal value of $\boldsymbol{g}$, let us consider the following minimax problem:

$$
(\mathrm{Q}(\boldsymbol{y}^k)) \quad \left| \begin{array}{ll} \text{minimize} & G(\boldsymbol{x}; \boldsymbol{y}^k) = \max\{c_j(g_j(\boldsymbol{x}) - y_j^k) \mid j = 1, \ldots, p\} \\ \text{subject to} & \boldsymbol{x} \in X, \quad \boldsymbol{g}(\boldsymbol{x}) \leq \boldsymbol{u}, \end{array} \right.
$$

where $\boldsymbol{c} = (c_1, \ldots, c_p)^{\mathrm{T}}$ is a positive vector of weights and $\boldsymbol{u} \in \mathbb{R}^p$ is defined by (2.7). The objective function $G(\cdot\,; \boldsymbol{y}^k)$ is convex and its minimum point $\boldsymbol{x}^*(\boldsymbol{y}^k)$ can be obtained if we apply any one of standard algorithms to an equivalent problem:

$$
\left| \begin{array}{ll} \text{minimize} & z \\ \text{subject to} & \boldsymbol{x} \in X, \quad \boldsymbol{g}(\boldsymbol{x}) \leq \boldsymbol{u}, \\ & g_j(\boldsymbol{x}) - z/c_j \leq y_j^k, \quad j = 1, \ldots, p, \end{array} \right. \tag{3.3}
$$

where z is a scalar variable. It is easy to check that $\boldsymbol{x}^*(\boldsymbol{y}^k)$ is feasible to (P) and that $\boldsymbol{\ell} \leq \boldsymbol{g}(\boldsymbol{x}^*(\boldsymbol{y}^k)) \leq \boldsymbol{u}$ holds. Hence, by letting $\boldsymbol{y}^*(\boldsymbol{y}^k) = \boldsymbol{g}(\boldsymbol{x}^*(\boldsymbol{y}^k))$, we have a feasible solution $\boldsymbol{y}^*(\boldsymbol{y}^k)$ of (MP), which satisfies

$$
f(\boldsymbol{y}^k) \leq f(\boldsymbol{y}^*) \leq f(\boldsymbol{y}^*(\boldsymbol{y}^k)). \tag{3.4}
$$

Let $z(\boldsymbol{y}) = G(\boldsymbol{x}^*(\boldsymbol{y}); \boldsymbol{y})$ and let

$$
\overline{Y}_k = \{\boldsymbol{y} \in \mathbb{R}^p \mid c_j(y_j - y_j^k) < z(\boldsymbol{y}^k), \ j = 1, \ldots, p\}. \tag{3.5}
$$

Lemma 3.1. *Function* $z : \mathbb{R}^p \to \mathbb{R}^1$ *is convex and satisfies*

$$z(\boldsymbol{y}) \leq 0, \;\; \forall \boldsymbol{y} \in Y; \;\; z(\boldsymbol{y}^k) > 0 \;\; \textit{if} \;\; \boldsymbol{y}^k \notin Y. \tag{3.6}$$

Proof: Let $\boldsymbol{y}'$ be an arbitrary point of Y. Then by definition $\boldsymbol{g}(\boldsymbol{x}') - \boldsymbol{y}' \leq \boldsymbol{0}$ holds for some $\boldsymbol{x}' \in X$, and hence we have $h_k(\boldsymbol{y}') \leq \max_j\{c_j^k(g_j(\boldsymbol{x}') - y_j')\} \leq 0$ by noting $\boldsymbol{c} > 0$. If $\boldsymbol{y}^k \notin Y$, then no $\boldsymbol{y} \in Y$ satisfies $\boldsymbol{y} \leq \boldsymbol{y}^k$ under condition (2.1) because $\boldsymbol{y}^k$ is an optimal solution of a relaxed problem of (P). This implies that there is some index q such that $g_q(\boldsymbol{x}) > y_q^k$ for any feasible solution $\boldsymbol{x}$ of $(\mathrm{Q}(\boldsymbol{y}^k))$. Hence the optimal value $z(\boldsymbol{y}^k)$ of $(\mathrm{Q}(\boldsymbol{y}^k))$ is positive if $\boldsymbol{y}^k \notin Y$.

Convexity of z is shown as follows: Let $\boldsymbol{y}'$ and $\boldsymbol{y}''$ be any points in $\mathbb{R}^p$. Then for any $\lambda \in [0, 1]$ we have

$$\begin{aligned}
&(1-\lambda)z(\boldsymbol{y}') + \lambda z(\boldsymbol{y}'') \\
&\quad = (1-\lambda)\max_j\{c_j(g_j(\boldsymbol{x}^*(\boldsymbol{y}')) - y_j')\} + \lambda \max_j\{c_j(g_j(\boldsymbol{x}^*(\boldsymbol{y}'')) - y_j'')\} \\
&\quad \geq \max_j\{(1-\lambda)c_j(g_j(\boldsymbol{x}^*(\boldsymbol{y}')) - y_j') + \lambda c_j(g_j(\boldsymbol{x}^*(\boldsymbol{y}'')) - y_j'')\} \\
&\quad \geq \max_j\{c_j(g_j((1-\lambda)\boldsymbol{x}^*(\boldsymbol{y}') + \lambda \boldsymbol{x}^*(\boldsymbol{y}'')) - (1-\lambda)y_j' - \lambda y_j'')\} \\
&\quad \geq max_j\{c_j(g_j(\boldsymbol{x}^*((1-\lambda)\boldsymbol{y}' + \lambda \boldsymbol{y}'')) - (1-\lambda)y_j' - \lambda y_j'')\} \\
&\quad = z((1-\lambda)\boldsymbol{y}' + \lambda \boldsymbol{y}''),
\end{aligned}$$

since c_j's are positive and g_j's are convex. □

Lemma 3.2. *If* $\boldsymbol{y}^k \notin Y$, *then*

$$\boldsymbol{y}^k \in \overline{Y}_k, \;\; \overline{Y}_k \bigcap Y = \emptyset. \tag{3.7}$$

Proof: The first part of (3.7) follows from (3.6). To show the second, choose an arbitrary $(\boldsymbol{x}', \boldsymbol{y}') \in Z$. Then we have $z(\boldsymbol{y}^k) \leq \max_j\{c_j(g_j(\boldsymbol{x}') - y_j^k)\} \leq \max_j\{c_j(y_j' - y_j^k)\}$, which implies $\boldsymbol{y} \notin \overline{Y}_k$ for any $\boldsymbol{y} \in Y$. □

Thus by defining the $k+1$st relaxation of Y below:

$$Y_{k+1} = Y_k \setminus \overline{Y}_k, \tag{3.8}$$

we can gouge out some portion containing $\boldsymbol{y}^k$ from Y_k without losing any points of Y.

If we use the above procedure to generate every relaxed problem, the feasible region Y_k of (P_k) will not be any convex set but the union of a number of rectangles in $\mathbb{R}^p$:

$$Y_k = \bigcup_{i \in I} R_i, \tag{3.9}$$

where I is some index set and

$$R_i = \{\boldsymbol{y} \in \mathbb{R}^p \mid \boldsymbol{\ell}^i \leq \boldsymbol{y} \leq \boldsymbol{u}\}, \;\; i \in I. \tag{3.10}$$

However, only among the vertices $\boldsymbol{\ell}^i$'s exists an optimal solution $\boldsymbol{y}^k$ because the objective function f has the nondecreasing property (2.1). Hence we can solve (P_k) by performing at most $|I|$ comparisons:

$$\boldsymbol{y}^k \in \operatorname{argmin}\{f(\boldsymbol{y}) \mid \boldsymbol{y} = \boldsymbol{\ell}^i,\ i \in I\}. \tag{3.11}$$

Let I_k be a subset of indices $i \in I$ such that $\boldsymbol{\ell}^i \in \overline{Y}_k$. If $\boldsymbol{y}^k$ is not a point of Y, for each $i \in I_k$ we have to discard the portion of R_i included in $\overline{Y}_k$. This can easily be done in the following way:

Let J_k be an index set such that

$$\left.\begin{array}{ll} c_j(u_j - y_j^k) > z(\boldsymbol{y}^k), & j \in J_k, \\ c_j(u_j - y_j^k) \leq z(\boldsymbol{y}^k), & j \in \{1, \ldots, p\} \setminus J_k. \end{array}\right\} \tag{3.12}$$

For each $j \in J_k$ let

$$\boldsymbol{\ell}^{ij} = (\ell_1^i, \ldots, \ell_{j-1}^i, y_j^k + z(\boldsymbol{y}^k)/c_j, \ell_{j+1}^i, \ldots, \ell_p^i)^{\mathrm{T}} \tag{3.13}$$

and define

$$R_{ij} = \{\boldsymbol{y} \in \mathbb{R}^p \mid \boldsymbol{\ell}^{ij} \leq \boldsymbol{y} \leq \boldsymbol{u}\}. \tag{3.14}$$

If we replace R_i with $\bigcup_{j \in J_k} R_{ij}$ for every $i \in I_k$, all the portion of Y_k included in $\overline{Y}_k$ is deleted, and then the next relaxation Y_{k+1} of the same form as (3.9) is generated, i.e.,

$$Y_{k+1} = (\bigcup_{i \in I \setminus I_k} R_i) \bigcup (\bigcup_{i \in I_k} \bigcup_{j \in J_k} R_{ij}). \tag{3.15}$$

Note that we may remove any R_{ij} with vertex $\boldsymbol{\ell}^{ij}$ from this definition (3.15) unless $\boldsymbol{\ell}^{ij}$ is a vertex of Y_{k+1}.

3.2. Description of the Algorithm

We are now ready to present an outer approximation algorithm for solving the master problem (MP). Here $\epsilon \geq 0$ stands for a given tolerance.

Algorithm 1.

Step 0. Compute both the bounds $\boldsymbol{\ell}$ and $\boldsymbol{u}$ of $\boldsymbol{g}$ according to (2.3), (2.4) and (2.7), and define the feasible region $Y_0 = \{\boldsymbol{y} \in \mathbb{R}^p \mid \boldsymbol{\ell} \leq \boldsymbol{y} \leq \boldsymbol{u}\}$ of the initial relaxed problem (P_0). Let $k = 0$ and go to Step 1.

Step 1. Compute an optimal solution $\boldsymbol{y}^k$ of (P_k). Solve a minimax problem $(\mathrm{Q}_k(\boldsymbol{y}^k))$ and let $\boldsymbol{x}^*(\boldsymbol{y}^k)$ and $z(\boldsymbol{y}^k)$ be an optimal solution and the optimal value respectively.

Step 2. Let $\boldsymbol{y}^*(\boldsymbol{y}^k) = \boldsymbol{g}(\boldsymbol{x}^*(\boldsymbol{y}^k))$. If

$$f(\boldsymbol{y}^*(\boldsymbol{y}^k)) - f(\boldsymbol{y}^k) \leq \epsilon, \tag{3.16}$$

then stop.

Step 3. Let $\overline{Y}_k = \{\boldsymbol{y} \in \mathbb{R}^p \mid c_j(y_j - y_j^k) < z(\boldsymbol{y}^k),\ j = 1, \ldots, p\}$ and update the relaxation of Y as $Y_{k+1} = Y_k \setminus \overline{Y}_k$. Return to Step 1 with $k = k + 1$. □

If this algorithm terminates, the stopping criterion (3.16) guarantees the ϵ-optimality of $\boldsymbol{y}^*(\boldsymbol{y}^k)$ to (MP). By the definition of $\boldsymbol{y}^*(\boldsymbol{y}^k)$ we have $(\boldsymbol{x}^*(\boldsymbol{y}^k),\ \boldsymbol{y}^*(\boldsymbol{y}^k)) \in Z$. Hence $\boldsymbol{x}^*(\boldsymbol{y}^k)$ is a globally ϵ-optimal solution of (P) in this case. Moreover, we should note that every $\boldsymbol{x}^*(\boldsymbol{y}^k)$ generated in the course of computation has a certain desirable property in vector optimization. Since $\boldsymbol{x}^*(\boldsymbol{y}^k)$ minimizes $\max_j\{c_j(g_j(\boldsymbol{x}) - y_j^k)\}$ on X for $\boldsymbol{c} > 0$, there are no $\boldsymbol{x} \in X$ such that $\boldsymbol{g}(\boldsymbol{x}) < \boldsymbol{g}(\boldsymbol{x}^*(\boldsymbol{y}^k))$. This implies that $\boldsymbol{x}^*(\boldsymbol{y}^k)$ is a *weakly efficient* solution of a vector minimization problem (see e.g. [15]):

$$\left|\begin{array}{ll} \text{minimize} & \boldsymbol{g}(\boldsymbol{x}) \\ \text{subject to} & \boldsymbol{x} \in X. \end{array}\right.$$

Theorem 3.3. *Algorithm 1 terminates after finitely many iterations if $\epsilon > 0$. If $\epsilon = 0$, Algorithm 1 generates a sequence of points $\boldsymbol{y}^k$'s, every accumulation point of which is a globally optimal solution of (MP).*

Proof: Suppose the algorithm does not terminate. Then an infinite sequence $\{\boldsymbol{y}^k\}$ is generated in the compact set Y_0. We can take a subsequence $\{\boldsymbol{y}^{k_q} \mid q = 0, 1, \ldots\}$ which converges to some point $\bar{\boldsymbol{y}} \in Y_0$. Let us assume the contrary to the assertion, i.e., there exists some constant $\sigma > \epsilon$ such that

$$f(\boldsymbol{y}^*(\boldsymbol{y}^{k_q})) - f(\boldsymbol{y}^{k_q}) \geq \sigma, \quad \forall q. \tag{3.17}$$

Let $h(\boldsymbol{y};\, \boldsymbol{y}^k) = \max_j\{c_j(y_j - y_j^k) - z(\boldsymbol{y}^k)\}$. We see from (3.5) that $\boldsymbol{y} \in \overline{Y}_k$ if and only if $h(\boldsymbol{y};\, \boldsymbol{y}^k) < 0$. Then by Lemma 3.2 we have $h(\boldsymbol{y}^{k_{q+1}};\, \boldsymbol{y}^{k_q}) \geq 0$ for every q and hence

$$\lim_{q\to\infty} h(\boldsymbol{y}^{k_{q+1}};\, \boldsymbol{y}^{k_q}) = \lim_{q\to\infty} h(\boldsymbol{y}^{k_q};\, \boldsymbol{y}^{k_q}) = -z(\bar{\boldsymbol{y}}) \geq 0$$

by continuity of z (Lemma 3.1). On the other hand, it follows from (3.6) that $z(\boldsymbol{y}^{k_q}) > 0$ for every q, which also implies $z(\bar{\boldsymbol{y}}) \geq 0$. Consequently, we have

$$z(\bar{\boldsymbol{y}}) = \max_j\{y_j^*(\bar{\boldsymbol{y}}) - \bar{y}_j\} = 0, \tag{3.18}$$

which contradicts assumption (3.17) under condition (2.1). If $\epsilon > 0$, then (3.16) holds after finitely many iterations and Algorithm 1 terminates. If $\epsilon = 0$, by continuity of f we have

$$f(\bar{\boldsymbol{y}}) = \lim_{q\to\infty} f(\boldsymbol{y}^{k_q}) \leq f(\boldsymbol{y}), \quad \forall \boldsymbol{y} \in Y.$$

It follows from (3.6) and (3.18) that $\bar{\boldsymbol{y}} \in Y$, and hence $\bar{\boldsymbol{y}}$ is a globally optimal solution of the master problem (MP). □

4. Some Improvements on the Algorithm

In this section we will develop two procedures for improving the efficiency of the algorithm presented in Section 3.

4.1. Determination of the weighting vector

We have not yet discussed how to determine the weighting vector $\boldsymbol{c}$ of the objective function of $(\mathrm{Q}(\boldsymbol{y}^k))$. As shown in Theorem 3.3, Algorithm 1 converges with any fixed $\boldsymbol{c} > 0$ and yields an ϵ-optimal solution of (MP) when $\epsilon > 0$. However, the choice of $\boldsymbol{c}$ will affect the speed of convergence considerably.

Our purpose in solving the minimax problem $(\mathrm{Q}(\boldsymbol{y}^k))$ is essentially to find a feasible solution $\boldsymbol{y}$ of (MP) such that $f(\boldsymbol{y})$ is the closest to an ideal value $f(\boldsymbol{y}^k)$. If f is differentiable at $\boldsymbol{y}^k$, we have a first-order approximation of f around $\boldsymbol{y}^k$:

$$f(\boldsymbol{y}) \approx f(\boldsymbol{y}^k) + \nabla f(\boldsymbol{y}^k)(\boldsymbol{y} - \boldsymbol{y}^k). \tag{4.19}$$

Also we have

$$\nabla f(\boldsymbol{y}^k)(\boldsymbol{y} - \boldsymbol{y}^k) \leq p \max\{\frac{\partial f(\boldsymbol{y}^k)}{\partial y_j}(y_j - y_j^k) \mid j = 1, \ldots, p\}. \tag{4.20}$$

Hence, to find a closest point $\boldsymbol{y} \in Y$ to $\boldsymbol{y}^k$, we may minimize the right-hand-side of (4.20), i.e.,

$$\left|\begin{array}{ll} \text{minimize} & \max\{\dfrac{\partial f(\boldsymbol{y}^k)}{\partial y_j}(g_j(\boldsymbol{x}) - y_j^k) \mid j = 1, \ldots, p\} \\ \text{subject to} & x \in X, \;\; \boldsymbol{g}(\boldsymbol{x}) \leq \boldsymbol{u}. \end{array}\right. \tag{4.21}$$

If f is continuously differentiable on S and $\nabla f(\boldsymbol{y}) > 0$ for all $\boldsymbol{y} \in S$, we can exploit $\nabla f(\boldsymbol{y}^k)$ as the weighting vector $\boldsymbol{c}$ of $(\mathrm{Q}(\boldsymbol{y}^k))$ in every iteration of the algorithm. Both $\boldsymbol{c}$ and z are continuous functions on S if we let $\boldsymbol{c}(\boldsymbol{y}) = \nabla f(\boldsymbol{y})$. We can therefore prove in just the same way as in the proof of Theorem 3.3 that a subsequence of $\boldsymbol{y}^k$'s generated by the algorithm converges to a globally optimal solution of (MP).

If f has no positive gradients at some points of S, we may instead employ

$$c_j(\boldsymbol{y}) = \frac{f(\boldsymbol{y} + \delta \boldsymbol{e}^j) - f(\boldsymbol{y})}{\delta}, \;\; j = 1, \ldots, p, \tag{4.22}$$

where δ is a sufficiently small positive constant and $\boldsymbol{e}^j \in \mathbb{R}^p$ is the jth unit vector. Note that $\boldsymbol{c}$ defined by (4.22) is also continuous and positive valued at any $\boldsymbol{y}^k$, since f is a continuous function satisfying (2.1).

4.2. Modified Algorithm using Branch-and-bound procedure

The efficiency of the algorithm will also depend on the number $|I|$ of vertices $\boldsymbol{\ell}^i$'s of Y_k, but in particular on the number $|I_k|$ of those contained in $\overline{Y}_k$. If $\overline{Y}_k$ contains only one vertex, say $\boldsymbol{\ell}^q$, at most p vertices $\boldsymbol{\ell}^{qj}$'s of Y_{k+1} are newly generated. Then we can obtain an optimal solution $\boldsymbol{y}^{k+1}$ of (P_{k+1}) only by performing at most p comparisons if $f(\boldsymbol{\ell}^i)$'s are sorted beforehand. However, such a favorable situation will not be expected in general so long as we discard $\overline{Y}_k$ from the whole of Y_k.

Let $Y_k = \bigcup_{i\in I} R_i$ and suppose a vertex $\boldsymbol{\ell}^{i_k}$ of $R_{i_k} = \{\boldsymbol{y} \in \mathbb{R}^p \mid \boldsymbol{\ell}^{i_k} \leq \boldsymbol{y} \leq \boldsymbol{u}\}$ $(i_k \in I)$ provides an optimal solution of (P_k). We define the following set:

$$\tilde{Y}_k = R_{i_k} \bigcap \overline{Y}_k. \tag{4.23}$$

Lemma 4.4. *If $\boldsymbol{y}^k \notin Y$, then*

$$\boldsymbol{y}^k \in \tilde{Y}_k, \quad \tilde{Y}_k \bigcap Y = \emptyset. \tag{4.24}$$

Proof: Since we are assuming that $\boldsymbol{y}^k = \boldsymbol{\ell}^{i_k}$, the first relation of (4.24) is obvious. The second follows from Lemma 3.2 and the relation $\tilde{Y}_k \subset \overline{Y}_k$. □

If we discard the portion of Y_k only included in $\tilde{Y}_k$, then we have an alternative $k+1$st relaxation of Y:

$$Y_{k+1} = Y_k \setminus \tilde{Y}_k = (Y_k \setminus R_{i_k}) \bigcup (R_{i_k} \setminus \overline{Y}_k), \tag{4.25}$$

or an equivalent expression:

$$Y_{k+1} = (\bigcup_{i\neq i_k} R_i) \bigcup (\bigcup_{j\in J_k} R_{i_k,j}),$$

where J_k and $R_{i_k,j}$'s are defined by (3.12) – (3.14). This relaxation of Y is not so tight as the one based on (3.8). However, there is still a merit in using it. If we update the relaxation of Y according to (4.25), then only one of the vertices is removed and at most p vertices are newly generated. This leads us to a p-tree underlying a branch-and-bound method.

We incorporate the above two procedures into the algorithm. Here $\epsilon \geq 0$ is a given tolerance, $\boldsymbol{y}^\circ$ and v° are the incumbent and its objective function value of (MP) respectively.

Algorithm 2.

Step 0. Compute the bounds $\boldsymbol{\ell}$ and $\boldsymbol{u}$ of g according to (2.3), (2.4) and (2.7), and define the feasible region $Y_0 = \{\boldsymbol{y} \in \mathbb{R}^p \mid \boldsymbol{\ell} \leq \boldsymbol{y} \leq \boldsymbol{u}\}$ of the initial relaxed problem (P_0). Let $\mathcal{Y} = \{\boldsymbol{\ell}\}$ and initialize the incumbent: $\boldsymbol{y}^\circ = \boldsymbol{u}$, $v^\circ = f(\boldsymbol{y}^\circ)$. Let $k = 0$ and go to Step 1.

Step 1. Select $y^k \in \mathcal{Y}$ with the least $f(y^k)$ and let $\mathcal{Y} = \mathcal{Y} \setminus \{y^k\}$. If f is continuously differentiable on S and $\nabla f(y) > 0$ for all $y \in S$, let $c(y^k) = \nabla f(y^k)$. Otherwise, define $c(y^k)$ according to (4.22). Solve $(\mathrm{Q}(y^k))$ with the weighting vector $c(y^k)$ and let $x^*(y^k)$ and $z(y^k)$ be an optimal solution and the optimal value respectively.

Step 2. Let $y^*(y^k) = g(x^*(y^k))$. If $f(y^*(y^k)) < v^\circ$, then update the incumbent: $y^\circ = y^*(y^k)$, $v^\circ = f(y^*(y^k))$. If $v^\circ - f(y^k) \leq \epsilon$, then stop.

Step 3. For each $j = 1, \ldots, p$, do the following: If $c_j(u_j - y_j^k) > z(y^k)$, then let

$$y^{kj} = (y_1^k, \ldots, y_{j-1}^k, y_j^k + z(y^k)/c_j, y_{j+1}^k, \ldots, y_p^k)^{\mathrm{T}}, \tag{4.26}$$

and let $\mathcal{Y} = \mathcal{Y} \bigcup \{y^{kj}\}$. Return to Step 1 with $k = k + 1$. □

The following is analogous to Theorem 3.3:

Theorem 4.5. *Algorithm 2 terminates after finitely many iterations if $\epsilon > 0$. If $\epsilon = 0$, Algorithm 2 generates a sequence $\{y^k\}$, every accumulation point of which is a globally optimal solution of (MP).* □

To save the memory required by Algorithm 2, we can employ the depth first rule in selecting y^k from $\mathcal{Y}$ instead of the best bound rule. Since $f(y^k)$ gives a lower bound of f on the rectangle $R_{i_k} = \{y \in \mathbb{R}^p \mid y^k \leq y \leq u\}$, the sign of $v^\circ - f(y^k)$ indicates if the subproblem with R_{i_k} is fathomed or should be branched. Although the convergence is somewhat slower, this alteration causes no trouble if $\epsilon > 0$. However, if $\epsilon = 0$, the sequence $\{y^k\}$ might converge to some locally but not globally optimal solution of (MP).

4.3. Numerical example

Before concluding this section, let us illustrate Algorithm 2 using a three-dimensional problem (see also Figure 4.1):

$$\left|\begin{array}{ll} \text{minimize} & (5 - 1.25x_1) \cdot (5 - 0.75x_2) \\ \text{subject to} & -3x_1 + 3x_2 + 6x_3 \leq 8, \\ & 17x_1 - 3x_2 + 14x_3 \leq 48, \\ & 27x_1 + 15x_2 - 24x_3 \leq 96, \\ & x_1 \geq 0,\ x_2 \geq 0,\ x_3 \geq 0. \end{array}\right. \tag{4.1}$$

Let us define

$$f(y) = y_1 \cdot y_2, \quad g(x) = (g_1(x), g_2(x)) = (5 - 1.25x_1, 5 - 0.75x_2).$$

If we let $S = \{y \in \mathbb{R}^2 \mid y > 0\}$, then f satisfies condition (2.1) on S. Moreover, assumption (2.2) is fulfilled, since

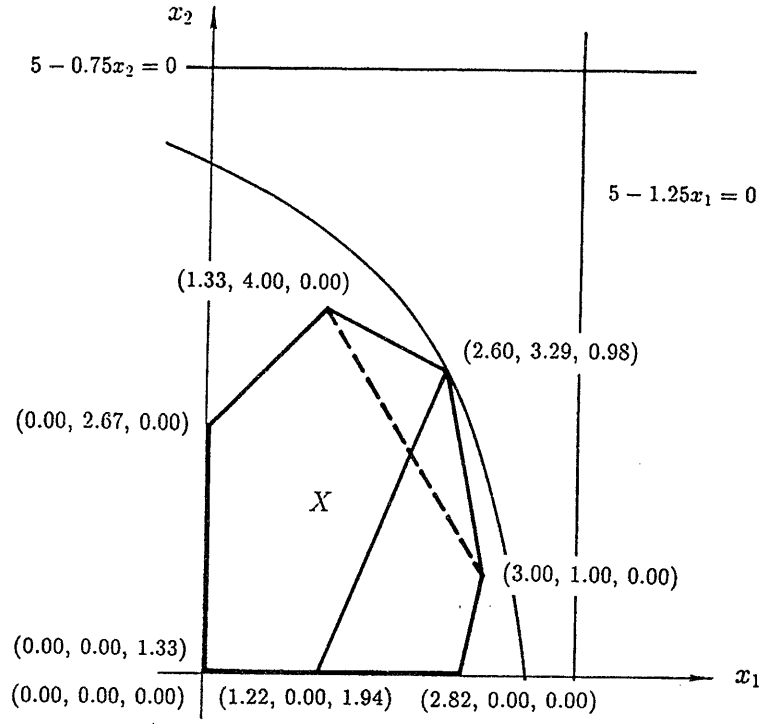

Figure 4.1. Three-dimensional example (4.1) of (P).

$$\ell_1 = g_1(\check{\boldsymbol{x}}^1) = 1.250 > 0, \quad \ell_2 = g_2(\check{\boldsymbol{x}}^2) = 2.000 > 0,$$

where $\check{\boldsymbol{x}}^1 = (3.000,\ 1.000,\ 0.000)$ and $\check{\boldsymbol{x}}^2 = (1.333,\ 4.000,\ 0.000)$ are minimizers of g_1 and g_2 respectively. Upper bounds of g_1 and g_2 are given as follows:

$$u_1 = \max\{y \mid 2y \leq v\} = 2.656, \quad u_2 = \max\{y \mid 1.25y \leq v\} = 4.250,$$

where $v = \min\{f \circ \boldsymbol{g}(\boldsymbol{x}) \mid \boldsymbol{x} = \check{\boldsymbol{x}}^1,\ \check{\boldsymbol{x}}^2\} = f \circ \boldsymbol{g}(\check{\boldsymbol{x}}^1) = 5.313$. Thus we have

$$Z = \left\{ (\boldsymbol{x},\ \boldsymbol{y}) \in \mathbb{R}^3 \times \mathbb{R}^2 \;\middle|\; \begin{array}{l} \boldsymbol{x} \in X, \\ 5.000 - x_1 - y_1 \leq 0, \quad 5.000 - x_2 - y_2 \leq 0, \\ 1.250 \leq y_1 \leq 2.656, \quad 2.000 \leq y_2 \leq 4.250 \end{array} \right\},$$

where X is the feasible region of (4.1). Figure 4.2 depicts the feasible region $Y = \{\boldsymbol{y} \in \mathbb{R}^2 \mid \exists \boldsymbol{x} \in \mathbb{R}^2,\ (\boldsymbol{x},\ \boldsymbol{y}) \in Z\}$ of the master problem (MP).

To solve the master problem, we generate a sequence of its relaxed problems. The feasible region of the initial relaxed problem (P_0) is $Y_0 = \{\boldsymbol{y} \in \mathbb{R}^2 \mid 1.250 \leq y_1 \leq 2.656,\ 2.000 \leq y_2 \leq 4.250\}$, and hence

$$\boldsymbol{y}^0 = (1.250,\ 2.000)$$

is optimal to (P_0). Regarding $\boldsymbol{y}^0$ as an ideal value of $\boldsymbol{g}$, we solve a minimax problem:

$$(\mathrm{Q}(\boldsymbol{y}^0)) \left| \begin{array}{ll} \text{minimize} & z = \max\{c_1(3.750 - 1.250x_1),\ c_2(3.000 - 0.750x_2)\} \\ \text{subject to} & \boldsymbol{x} \in X,\ \ x_1 \geq 1.875,\ \ x_2 \geq 1.000. \end{array} \right.$$

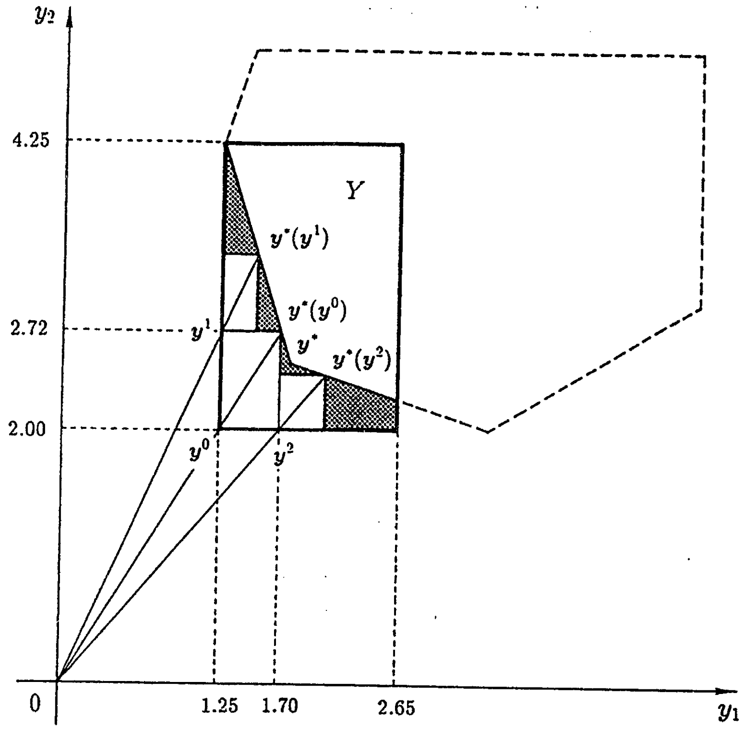

Figure 4.2. The master problem of (4.1).

If we choose $c_1 = \partial f(\boldsymbol{y}^0) \,/\, \partial y_1 = 2.000$ and $c_2 = \partial f(\boldsymbol{y}^0) \,/\, \partial y_2 = 1.250$, then

$$\boldsymbol{x}^*(\boldsymbol{y}^0) = (2.641,\ 3.043,\ 0.874), \quad z(\boldsymbol{y}^0) = 0.898$$

is optimal to $(\mathrm{Q}(\boldsymbol{y}^0))$. We also obtain a feasible solution of (MP):

$$\boldsymbol{y}^*(\boldsymbol{y}^0) = \boldsymbol{g}(\boldsymbol{x}^*(\boldsymbol{y}^0)) = (1.698,\ 2.718),$$

which gives an incumbent value:

$$v^\circ = f(\boldsymbol{y}^*(\boldsymbol{y}^0)) = 4.616.$$

According to (4.26) we generate

$$\begin{aligned} \boldsymbol{y}^{01} &= (y_1^0 + z(\boldsymbol{y}^0) \,/\, c_1,\ y_2^0) = (1.698,\ 2.000), \\ \boldsymbol{y}^{02} &= (y_1^0,\ y_2^0 + z(\boldsymbol{y}^0) \,/\, c_2) = (1.250,\ 2.718), \end{aligned}$$

and let $\mathcal{Y} = \{\boldsymbol{y}^{01},\ \boldsymbol{y}^{02}\}$ (see Figure 4.2).

Since $f(\boldsymbol{y}^{01}) = f(\boldsymbol{y}^{02}) = 3.397$, both $\boldsymbol{y}^{01}$ and $\boldsymbol{y}^{02}$ are optimal to the second relaxed problem (P_1). We select an arbitrary $\boldsymbol{y}^1$ from $\mathcal{Y}$, say $\boldsymbol{y}^1 = \boldsymbol{y}^{02}$, and solve

$$(\mathrm{Q}(\boldsymbol{y}^1)) \left| \begin{array}{ll} \text{minimize} & z = \max\{c_1(3.750 - 1.250x_1),\ c_2(2.282 - 0.750x_2)\} \\ \text{subject to} & \boldsymbol{x} \in X, \quad x_1 \geq 1.875, \quad x_2 \geq 1.000, \end{array} \right.$$

where $c_1 = \partial f(\boldsymbol{y}^1) \,/\, \partial y_1 = 2.718$ and $c_2 = \partial f(\boldsymbol{y}^1) \,/\, \partial y_2 = 1.250$. Then we have

$$\begin{aligned}
&\boldsymbol{x}^*(\boldsymbol{y}^1) = (2.781,\ 2.249,\ 0.534), && z(\boldsymbol{y}^1) = 0.745,\\
&\boldsymbol{y}^*(\boldsymbol{y}^1) = (1.524,\ 3.313), && f(\boldsymbol{y}^*(\boldsymbol{y}^1)) = 5.050,\\
&\boldsymbol{y}^{11} = (1.524,\ 2.718), && \boldsymbol{y}^{12} = (1.250,\ 3.313),
\end{aligned}$$

and let $\mathcal{Y} = \{\boldsymbol{y}^{01},\ \boldsymbol{y}^{11},\ \boldsymbol{y}^{12}\}$.

Since $f(\boldsymbol{y}^{01}) = 3.397$ is smaller than $f(\boldsymbol{y}^{11}) = f(\boldsymbol{y}^{12}) = 4.142$, we select $\boldsymbol{y}^{01}$ as $\boldsymbol{y}^2$ and solve

$$(\mathrm{Q}(\boldsymbol{y}^2))\ \left|\ \begin{array}{ll} \text{minimize} & z = \max\{c_1(3.302 - 1.250x_1),\ c_2(3.000 - 0.75x_2)\} \\ \text{subject to} & \boldsymbol{x} \in X,\quad x_1 \geq 1.875,\quad x_2 \geq 1.000, \end{array}\right.$$

where $c_1 = \partial f(\boldsymbol{y}^2) / \partial y_1 = 2.000$ and $c_2 = \partial f(\boldsymbol{y}^2) / \partial y_2 = 1.698$. Then we have

$$\begin{aligned}
&\boldsymbol{x}^*(\boldsymbol{y}^2) = (2.353,\ 3.434,\ 0.793), && z(\boldsymbol{y}^2) = 0.722,\\
&\boldsymbol{y}^*(\boldsymbol{y}^2) = (2.059,\ 2.425), && f(\boldsymbol{y}^*(\boldsymbol{y}^2)) = 4.993,\\
&\boldsymbol{y}^{21} = (2.059,\ 2.000), && \boldsymbol{y}^{22} = (1.698,\ 2.425),
\end{aligned}$$

and let $\mathcal{Y} = \{\boldsymbol{y}^{11},\ \boldsymbol{y}^{12},\ \boldsymbol{y}^{21},\ \boldsymbol{y}^{22}\}$.

In the next iteration, we select either $\boldsymbol{y}^{21}$ or $\boldsymbol{y}^{22}$ as $\boldsymbol{y}^3$, say $\boldsymbol{y}^3 = \boldsymbol{y}^{22}$, since $f(\boldsymbol{y}^{21}) = f(\boldsymbol{y}^{22}) = 4.118 < f(\boldsymbol{y}^{11}) = f(\boldsymbol{y}^{12}) = 4.142$. Solving $(\mathrm{Q}(\boldsymbol{y}^3))$, we have

$$\begin{aligned}
&\boldsymbol{x}^*(\boldsymbol{y}^3) = (2.587,\ 3.304,\ 0.975), && z(\boldsymbol{y}^3) = 0.165,\\
&\boldsymbol{y}^*(\boldsymbol{y}^3) = (1.767,\ 2.522), && f(\boldsymbol{y}^*(\boldsymbol{y}^3)) = 4.456,\\
&\boldsymbol{y}^{31} = (1.767,\ 2.425), && \boldsymbol{y}^{32} = (1.698,\ 2.522).
\end{aligned}$$

and let $\mathcal{Y} = \{\boldsymbol{y}^{11},\ \boldsymbol{y}^{12},\ \boldsymbol{y}^{21},\ \boldsymbol{y}^{31},\ \boldsymbol{y}^{32}\}$. Since $f(\boldsymbol{y}^*(\boldsymbol{y}^3)) < v^0 = 4.616$, we have to revise the incumbent:

$$v^0 = f(\boldsymbol{y}^*(\boldsymbol{y}^3)) = 4.456.$$

In the same way, we can generate a sequence of $\boldsymbol{y}^k$, $k = 4, 5, \ldots$, which converges to a point $\boldsymbol{y}^* = (1.750,\ 2.530)$. Hence a globally optimal solution of (4.1) is given by $\boldsymbol{x}^*(\boldsymbol{y}^*) = (2.600,\ 3.293,\ 0.983)$, where the objective function value is $f(\boldsymbol{y}^*) = 4.428$.

5. Computational Experiments

We will report the results of computational experiments on Algorithm 2 presented in the previous section. We solved two simple subclasses of (P):

$$(\mathrm{TP1})\ \left|\ \begin{array}{ll} \text{minimize} & \displaystyle\prod_{j=1}^{p}(M - \boldsymbol{d}_j^{\mathrm{T}}\boldsymbol{x}) \\ \text{subject to} & A\boldsymbol{x} \leq \boldsymbol{b},\quad \boldsymbol{x} \geq 0, \end{array}\right.$$

$$(\mathrm{TP2})\ \left|\ \begin{array}{ll} \text{minimize} & (M_1 - \boldsymbol{d}_1^{\mathrm{T}}\boldsymbol{x})(M_1 - \boldsymbol{d}_p^{\mathrm{T}}\boldsymbol{x}) + \displaystyle\sum_{j=2}^{p}(M_j - \boldsymbol{d}_{j-1}^{\mathrm{T}}\boldsymbol{x})(M_j - \boldsymbol{d}_j^{\mathrm{T}}\boldsymbol{x}) \\ \text{subject to} & A\boldsymbol{x} \leq \boldsymbol{b},\quad \boldsymbol{x} \geq 0, \end{array}\right.$$

Table 5.1. Comparison between Programs A and B for (TP1) when $\epsilon = 10^{-4}$

m	10	30	30	70	70	150	150
n	20	20	50	50	100	100	200
p	2	2	2	2	2	2	2
# of branching operations (standard deviation)							
Program A:	24.4 (15.8)	22.5 (12.9)	34.9 (23.5)	25.4 (19.1)	43.7 (17.4)	36.9 (33.4)	56.3 (29.3)
Program B:	16.6 (11.0)	14.4 (7.1)	22.6 (16.0)	16.6 (10.5)	31.4 (15.0)	26.0 (19.4)	38.2 (19.0)
CPU time in seconds (standard deviation)							
Program A:	0.05 (0.02)	0.20 (0.11)	0.66 (0.62)	1.68 (0.82)	5.54 (4.77)	14.97 (16.13)	38.48 (30.16)
Program B:	0.05 (0.02)	0.22 (0.10)	0.58 (0.42)	1.68 (0.86)	5.04 (3.94)	11.56 (7.38)	29.76 (14.81)

where $A \in \mathbb{R}^{m\times n}$, $\boldsymbol{b} \in \mathbb{R}^m$, $\boldsymbol{d}_j \in \mathbb{R}^n$, $j = 1, \ldots, p$. We drew every component of A and $\boldsymbol{d}_j$'s randomly from the uniform distribution over $[-1.000, 1.000]$ and that of $\boldsymbol{b}$ from $[0.000, 1.000]$, and let

$$M = 1.1 \cdot \max\{\boldsymbol{d}_j^{\mathrm{T}}\hat{\boldsymbol{x}}^j \mid j = 1, \ldots, p\}, \quad M_1 = 1.1 \cdot \max\{\boldsymbol{d}_1^{\mathrm{T}}\hat{\boldsymbol{x}}^1, \boldsymbol{d}_p^{\mathrm{T}}\hat{\boldsymbol{x}}^p\},$$
$$M_j = 1.1 \cdot \max\{\boldsymbol{d}_{j-1}^{\mathrm{T}}\hat{\boldsymbol{x}}^{j-1}, \boldsymbol{d}_j^{\mathrm{T}}\hat{\boldsymbol{x}}^j\}, \quad j = 2, \ldots, p,$$

where $\hat{\boldsymbol{x}}^j = \mathrm{argmax}\{\boldsymbol{d}_j^{\mathrm{T}}\boldsymbol{x} \mid A\boldsymbol{x} \leq b, \ \boldsymbol{x} \geq 0\}$. While the objective function of (TP1) is quasiconcave, that of (TP2) is in general neither quasiconcave nor quasiconvex [6, 7].

The branching rule we employed was a compromise between the best bound and depth first rules, i.e., among the last twenty $\boldsymbol{y}^k$'s of $\mathcal{Y}$ we selected one with the least $f(\boldsymbol{y}^k)$ when $|\mathcal{Y}| > 20$, where $f(\boldsymbol{y}^k) = \prod_j^p y_j^k$ for (TP1) and $f(\boldsymbol{y}^k) = y_1^k y_p^k + \sum_{j=2}^p y_{j-1}^k y_j^k$ for (TP2). Then we tried different weighting vectors for $(\mathrm{Q}(\boldsymbol{y}^k))$, i.e., $\boldsymbol{c} = (1, \ldots, 1)^{\mathrm{T}}$ in Program A and $\boldsymbol{c} = (\prod_{j\neq 1} y_j^k, \ldots, \prod_{j\neq p} y_j^k)^{\mathrm{T}}$ in Program B. The minimax problem $(\mathrm{Q}(\boldsymbol{y}^k))$ of both (TP1) and (TP2) can be reduced to a linear programming problem. We solved it by using a dual simplex algorithm, where we took the solution of $(\mathrm{Q}(\boldsymbol{y}^{k-1}))$ as an initial dual feasible point. We coded both Programs A and B in C language and tested them on a microSPARC II computer (70 MHz).

Table 5.1 shows the comparison between Programs A and B for (TP1) when $\epsilon = 10^{-4}$ and $p = 2$. (Note that (TP1) is equivalent to (TP2) in this case.) The size of (m, n) ranges from (10, 20) to (150, 200). Tables 5.2 and 5.3 show the results on Program B for (TP1) and (TP2) respectively, when $\epsilon = 10^{-4}$, $p = 3, 4$ and (m, n) is between (10, 20) and (70, 100). Each column of the tables gives the average number of branching operations and CPU time in seconds (and their standard deviations in the brackets)

Table 5.2. Computational results on Program B for (TP1) when $\epsilon = 10^{-4}$

m	10	30	30	70	70	10	30
n	20	20	50	50	100	20	20
p	3	3	3	3	3	4	4
# of branching operations (standard deviation)							
	259.8	163.6	491.5	657.9	1293.1	1079.7	2107.2
	(457.5)	(107.8)	(480.6)	(1399.3)	(1245.1)	(1593.4)	(2609.0)
CPU time in seconds (standard deviation)							
	0.95	2.13	7.88	32.44	73.18	5.32	25.59
	(1.58)	(1.44)	(6.13)	(54.34)	(52.48)	(7.27)	(30.81)

Table 5.3. Computational results on Program B for (TP2) when $\epsilon = 10^{-4}$

m	10	30	30	70	70	10	30
n	20	20	50	50	100	20	20
p	3	3	3	3	3	4	4
# of branching operations (standard deviation)							
	577.5	698.2	981.0	1375.5	2506.6	2498.6	3195.8
	(887.9)	(1282.0)	(1597.3)	(1480.3)	(2636.1)	(2775.8)	(3093.8)
CPU time in seconds (standard deviation)							
	2.22	7.41	14.75	71.93	155.50	13.75	47.70
	(3.82)	(12.74)	(19.61)	(72.25)	(140.41)	(17.33)	(47.67)

needed for solving ten examples. The number of branching operations corresponds to that of $(\mathrm{Q}(y^k))$'s solved in the course of computation.

We see from Table 5.1 that the performance of the algorithm considerably depends on the choice of the weighting vector. Program A requires more branching operations than Program B. This would affect the total computational time seriously when $p > 2$. We also see from Tables 5.1, 5.2 and 5.3 that Algorithm 2 is very sensitive to the size of p. The number of branching operations sharply increases as a function of p. However, we should emphasize that the number is rather insensitive to the size of (m, n) for each p. This implies that the total computational time is dominated by that for solving a linear programming problem, i.e., $(\mathrm{Q}(y^k))$, if p is a fixed number.

We conclude from these results that our algorithm is reasonably practical for the randomly generated classes (TP1) and (TP2) when p is less than 4. In this case, Algorithm 2 can generate globally 10^{-4}-optimal solutions of fairly large scale problems. We can expect the algorithm to solve more general classes if p is fixed at 2 or 3 and a practically efficient algorithm for $(\mathrm{Q}(y^k))$ is available.

References

[1] Benders, J.F., "Partitioning procedures for solving mixed variables programming problems," *Numerishe Mathematik* **4** (1962), 238 – 252.

[2] Horst, R. and H.Tuy, *Global Optimization: Deterministic Approaches*, Springer-Verlag (Berlin, 1990).

[3] Keeney, R.L. and H.Raiffa, *Decisions with Multiple Objectives: Preferences and Value Tradeoffs*, John Wiley and Sons (New York, 1976).

[4] Konno, H. and M.Inori, "Bond portfolio optimization by bilinear fractional programming," *Journal of the Operations Research Society of Japan* **32** (1988), 143 – 158.

[5] Konno, H. and T. Kuno, "Generalized linear multiplicative and fractional programming," *Annals of the Operations Research* **25** (1990), 147 – 162.

[6] Konno, H. and T. Kuno, "Linear multiplicative programming," *Mathematical Programming* **56** (1992), 51 – 64.

[7] Konno, H. and T. Kuno, "Multiplicative programming problems," in R. Horst and P.M. Pardalos (eds.), *Handbook of Global Optimization*, Kluwer Academic Publishers (Dortrecht, 1995).

[8] Konno, H., T. Kuno and Y. Yajima, "Global minimization of a generalized convex multiplicative function," *Journal of Global Optimization* **4** (1994), 47 – 62.

[9] Konno, H., Y. Yajima and T. Matsui, "Parametric simplex algorithms for solving a special class of nonconvex minimization problems," *Journal of Global Optimization* **1** (1991), 65 – 82.

[10] Kuno, T. and H. Konno, "A parametric successive underestimation method for convex multiplicative programming problems," *Journal of Global Optimization* **1** (1991), 267 – 285.

[11] Kuno, T., Y. Yajima and H. Konno, "An outer approximation method for minimizing the product of several convex functions on a convex set," *Journal of Global Optimization* **3** (1993), 325 – 335.

[12] Lasdon, L.S., *Optimization Theory for Large Systems*, Macmillan Company (London, 1970).

[13] Rockafellar, R.T., *Convex Analysis*, Princeton University Press (Princeton, NJ, 1970).

[14] Sniedovich, M., Maçalalag, E. and S. Findlay, "The simplex method as a global optimizer: a c-programming perspective," *Journal of Global Optimization* **4** (1994), 89 – 109.

[15] Steuer, R.E., *Multiple Criteria Optimization: Theory, Computation, and Application*, John Wiley & Sons (NY, 1986).

[16] Thoai, N.V., "A global optimization approach for solving the convex multiplicative programming problem," *Journal of Global Optimization* **1** (1991), 145 – 154.

[17] Tuy, H., "Concave minimization under linear constraints with special structure," *Optimization* **16** (1985), 335 – 352.

[18] Tuy, H., "The complementary convex structure in global optimization," *Journal of Global Optimization* **2** (1992), 21 – 40.