Research Article

A New Global Optimization Algorithm for Solving a Class of Nonconvex Programming Problems

Xue-Gang Zhou[1,2] and Bing-Yuan Cao[1]

[1] *School of Mathematics and Information Science, Key Laboratory of Mathematics and Interdisciplinary Sciences of Guangdong,*
Higher Education Institutes, Guangzhou University, Guangzhou, Guangdong 510006, China
[2] *Department of Applied Mathematics, Guangdong University of Finance, Guangzhou, Guangdong 510521, China*

Correspondence should be addressed to Bing-Yuan Cao; caobingy@163.com

A new two-part parametric linearization technique is proposed globally to a class of nonconvex programming problems (NPP). Firstly, a two-part parametric linearization method is adopted to construct the underestimator of objective and constraint functions, by utilizing a transformation and a parametric linear upper bounding function (LUBF) and a linear lower bounding function (LLBF) of a natural logarithm function and an exponential function with e as the base, respectively. Then, a sequence of relaxation lower linear programming problems, which are embedded in a branch-and-bound algorithm, are derived in an initial nonconvex programming problem. The proposed algorithm is converged to global optimal solution by means of a subsequent solution to a series of linear programming problems. Finally, some examples are given to illustrate the feasibility of the presented algorithm.

1. Introduction

In this paper, we consider a class of nonconvex programming problems as follows:

$$(\text{NPP}) \begin{cases} \min & f_0(x) \\ \text{s.t.} & Ax \leqslant b, \\ & f_j(x) \leqslant e_j, \\ & x \in X^0 = [\underline{x}, \overline{x}] \subset R^n, \end{cases} \tag{1}$$

where

$$f_j(x) = \sum_{k=1}^{K_j} f_{jk}(x) = \sum_{k=1}^{K_j} t_{jk} \prod_{m=1}^{K_{jk}} \left(c_{jkm}^T x + d_{jkm}\right)^{\alpha_{jkm}}, \tag{2}$$

$$j = 0, 1, \dots, p,$$

and $b \in R^q$, $c_{jkm} \in R^n$, e_j, t_{jk}, d_{jkm}, α_{jkm} are real numbers, A is $q \times n$ matrix, $\underline{x} > 0$, and $\underline{x}$, $\overline{x}$ are finite. $c_{jkm}^T x + d_{jkm} > 0$ for all $x \in X^0$. (NPP) contains various variants such as a sum or product of a finite number of ratios in linear functions, generalized linear multiplicative programs,

general polynomial programming, quadratic programming, and generalized geometric programming. So, (NPP) with its special form has attracted considerable attention to the literature because of its large number of practical applications in various fields of study, including transaction cost [1], financial optimization [2], robust optimization [3], VLISI chip design [4], data mining/pattern recognition [5], queueing-location problems [6, 7], bond portfolio optimization [8, 9], and elastic-plastic finite element analysis of metal forming processes [10]. From a researching point of view, (NPP) poses significant theoretical and computational challenges. It follows that it possesses multiple local optima that are not globally optimal. Recently, Jiao [11] and Shen et al. [12] have proposed a branch-and-bound algorithm globally to a class of nonconvex programming problems (NPP). By utilizing tangential hypersurfaces, convex envelope approximations of exponential function, and concave envelope approximations of logarithm function, a two-stage linear relaxation technique was given. Then, the relaxation linear programming of original problem can be constructed with a branch-and-bound algorithm proposed for globally solving (NPP).

For all $j = 0, 1, \ldots, p$, $k = 1, 2, \ldots, K_j$, if $\alpha_{jkm} = 1$, (NPP) can be reduced to the linear multiplicative programming (LMP) [15, 16]. When $K_j = 1$ for any $j = 0, 1, \ldots, p$, (NPP) is called multiplicative programming problems with exponent (MPE) [17, 18]; by utilizing logarithmic property, one can obtain an equivalent problem of (MPE), and a linear relaxation of equivalent problem is received by tangential hypersurfaces and concave envelope approximations. Then, a new branch-and-bound algorithm is given via solving a sequence of linear relaxations over partitioned subsets in order to find a global optimal solution to problem (MPE). If, for all $j = 0, 1, \ldots, p$, $k = 1, 2, \ldots, K_j$, $K_j > 1$ and $\alpha_{jkm} = 1$, the problem is called generalized linear multiplicative programs (GLMP) [19]. A greedy branching rule for rectangular branch-and-bound algorithms is proposed for solving problem (GLMP).

Assume that $K_{jk} = 2$ for all j, k, and, without loss of generality, let $\alpha_{jk1} = 1$ and $\alpha_{jk2} = -1$; (NPP) can be reduced to a linear sum-of-ratios fractional program. It is a global optimization problem; that is, it is known to generally possess multiple local optima that are not globally optimal [20]. Furthermore, it is NP-hard [21], and the objective function is neither quasiconvex nor quasiconcave. A number of algorithms have been proposed for globally solving a linear sum-of-ratios fractional program. They can be classified as follows: parametric simplex methods [22, 23], outer approximation methods [24, 25], the branch-and-bound approaches [13, 26–29], a duality-bounds method [30], an iteratively searching method [31], and so forth. Readers can find the applications, theory, and algorithms of the sum-of-ratios fractional programming in [32]. If there exist some $\alpha_{jkm} < 0$ and $K_{jk} > 2$, (NPP) is called generalized linear fractional programming problems. Shen and Wang [14] used a transformation and a two-part linearization technique to systematically convert the generalized linear fractional program into a series of linear programming problems.

When $\alpha_{jkm} = 1$ for all j, k, and m, (NPP) can be reduced to the general polynomial programming problem earlier investigated in [33–35]. Most recently, Lasserre [36, 37] developed a class of positive semidefinite relaxations for polynomial programming with the property that any polynomial program can be approximated as closely as desired by semidefinite program of this class.

In this paper, a new global optimization method is presented to (NPP) by solving a sequence of linear programming problems over partitioned subsets. By using a transformation and a two-part parametric linearization technique, we can systematically convert (NPP) into a series of linear programming problems. The solutions to these converted problems can be sufficiently closed to the global optimum of (NPP) by a successive refinement process. Some examples show that the proposed method can achieve all of the test problems in finding globally optimal solutions within a prespecified tolerance.

The organization and content of this paper can be summarized as follows. In Section 2, we first discuss parametric linear estimation of the natural logarithm function and the exponential function with e as the base, respectively.

Then, two-part parametric linearization method is presented for generating the relaxation lower linear programming of (NPP). In Section 3, the proposed branch-and-bound algorithm in which the relaxed subproblems are embedded is described, and the convergence of the algorithm is established. Some numerical results are reported in Section 4. Finally, concluding remarks are given in Section 5.

2. Parametric Linear Relaxation of (NPP)

Now, we derive an equivalent form of the function $f_j(x)$ by transformation. First, for any $x \in X^0$, since $c_{jkm}^T x + d_{jkm} > 0$, we assume that

$$c_{jkm}^T x + d_{jkm} = \exp\left(y_{jkm}\right). \tag{3}$$

Then, for all $j = 0, 1, \ldots, p$, the function $f_j(x)$ can be rewritten as

$$
\begin{aligned}
f_j(x) = f_j(y_j) &= \sum_{k=1}^{K_j} f_{jk}(y_{jk}) \\
&= \sum_{k=1}^{K_j} t_{jk} \exp\left(\sum_{m=1}^{K_{jk}} \alpha_{jkm} y_{jkm}\right) = \sum_{k=1}^{K_j} t_{jk} \exp\left(Y_{jk}\right),
\end{aligned}
\tag{4}
$$

where

$$y_{jkm} = \ln\left(c_{jkm}^T x + d_{jkm}\right), \qquad Y_{jk} = \sum_{m=1}^{K_{jk}} \alpha_{jkm} y_{jkm}. \tag{5}$$

In order to construct underestimator of function $f_j(x)$ for all j, we adopt two-part parametric linearization method. We will firstly derive a linear upper bounding function (LUBF) and a linear lower bounding function (LLBF) of $t_{jk} \exp(Y_{jk})$ about the variable y_{jk}, respectively. Then, in the second part, an LUBF about primal variable x will be constructed ultimately.

2.1. Parametric Linear Estimation of Logarithm and Exponential Functions. We first construct parametric linear overestimation and underestimation of a natural logarithm function and an exponential function with e as the base in interval vector $X \subseteq X^0$, respectively.

Let $X = [\underline{x}, \overline{x}] = \{x \in R^n \mid \underline{x} \leqslant x \leqslant \overline{x}\}$, for all $x = (x_i)_{n \times 1}$, where $\underline{x}$ and $\overline{x}$ are called the lower bound and upper bound, respectively. For any $x \in X$, we denote

$$x(\sigma) = \underline{x} + \sigma \cdot (\overline{x} - \underline{x}), \tag{6}$$

where $\sigma \in \{0, 1\}^n$ is an n-dimensional vector with components σ_i equal to 0 or 1. For convenience, we denote by $0 \in \{0, 1\}^n$ the vector with all components equal to 0 and by $1 \in \{0, 1\}^n$ the vector with all components equal to 1. Then, we have $x(0) = \underline{x}$ and $x(1) = \overline{x}$. The following theorem illustrates how to construct the lower and upper bound linear functions of natural logarithm function and the exponential function with e as the base, respectively.

Theorem 1. *For any interval vector X, $X \subseteq X^0 \subset R^n$, one assumes that the vertices of X are $x(\sigma)$, in form of (6). Let $\Phi(x) = \ln(\sum_{i=1}^n \gamma_i x_i + d)$ or $\Phi(x) = \exp(\sum_{i=1}^n \gamma_i x_i + d)$ and its gradient function $\Phi'(x) = ((\partial\Phi(x)/\partial x_1), \ldots, (\partial\Phi(x)/\partial x_n))^T$ over X. Then there exist vectors $\underline{z}, \overline{z} \in R^n$ such that the linear functions*

$$\Phi^l(x; X, \sigma) := z(\sigma)^T \cdot x + \Phi(x(\sigma)) - z(\sigma)^T \cdot x(\sigma),$$

$$\Phi^u(x; X, \sigma) := z(1-\sigma)^T \cdot x + \Phi(x(\sigma)) - z(1-\sigma)^T \cdot x(\sigma) \tag{7}$$

satisfy, for all $x \in X$, the inequalities

$$\Phi^l(x; X, \sigma) \leqslant \Phi(x) \leqslant \Phi^u(x; X, \sigma), \tag{8}$$

and moreover

$$\Phi^l(x(\sigma); X, \sigma) = \Phi^u(x(\sigma); X, \sigma) = \Phi(x(\sigma)), \tag{9}$$

where $z(\sigma)$, in form of (6), are vertices of the interval vector $z := [\underline{z}, \overline{z}]$, and the functions $\Phi^l(x; X, \sigma), \Phi^u(x; X, \sigma)$ show that Φ^l, Φ^u have the argument x and depend on the two parameters X and σ.

Proof. For function $\Phi(x) = \exp(\sum_{i=1}^n \gamma_i x_i + d)$, this result is shown in [38], and for $\Phi(x) = \ln(\sum_{i=1}^n \gamma_i x_i + d)$, the proof is similar. However, to provide a self-contained presentation, and because this result is central to this paper, we provide a direct proof for natural logarithm function.

By X and $\Phi'(x)$ it follows that there exist vectors $z = (\underline{z}_1, \ldots, \underline{z}_n)^T$ and $\overline{z} = (\overline{z}_1, \ldots, \overline{z}_n)^T$ satisfying

$$\underline{z} \leqslant \Phi'(x) \leqslant \overline{z}, \quad \text{for any } x \in X, \tag{10}$$

where, for $i = 1, 2, \ldots, n$,

$$\underline{z}_i = \min \left\{ \gamma_i \left(\sum_{j=1}^n \min\left(\gamma_j \underline{x}_j, \gamma_j \overline{x}_j\right) + d \right)^{-1}, \right.$$

$$\left. \gamma_i \left(\sum_{j=1}^n \max\left(\gamma_j \underline{x}_j, \gamma_j \overline{x}_j\right) + d \right)^{-1} \right\},$$

$$\tag{11}$$

$$\overline{z}_i = \max \left\{ \gamma_i \left(\sum_{j=1}^n \min(\gamma_j \underline{x}_j, \gamma_j \overline{x}_j) + d \right)^{-1}, \right.$$

$$\left. \gamma_i \left(\sum_{j=1}^n \max(\gamma_j \underline{x}_j, \gamma_j \overline{x}_j) + d \right)^{-1} \right\}.$$

By the mean value theorem, we have, for all $x \in X$,

$$\Phi(x) = \Phi(x(\sigma)) + \Phi'(\xi)^T \cdot (x - x(\sigma)), \tag{12}$$

where $\xi = \mu x + (1-\mu)x(\sigma)$ for some $\mu \in [0, 1]$. Then, (6) and (10) imply that, for $\sigma = 0$, the inequalities

$$\frac{\partial \Phi(\xi)}{\partial \xi_i} \geqslant z_i(\sigma), \qquad x_i - x_i(\sigma) \geqslant 0,$$

$$\forall x \in X, \ i = 1, 2, \ldots, n, \tag{13}$$

hold, where $x_i(\sigma)$ denotes the ith component of $x(\sigma)$. And for $\sigma = 1$ the inequalities

$$\frac{\partial \Phi(\xi)}{\partial \xi_i} \leqslant z_i(\sigma), \qquad x_i - x_i(\sigma) \leqslant 0,$$

$$\forall x \in X, \ i = 1, 2, \ldots, n, \tag{14}$$

are valid.

Consequently, it follows from the mean value theorem that

$$\Phi(x) = \Phi(x(\sigma)) + \sum_{i=1}^n \frac{\partial \Phi(\xi)}{\partial \xi_i} \cdot (x_i - x_i(\sigma))$$

$$\geqslant \Phi(x(\sigma)) + \sum_{i=1}^n z_i(\sigma) \cdot (x_i - x_i(\sigma)) \tag{15}$$

$$= z(\sigma)^T \cdot x + \Phi(x(\sigma)) - z(\sigma)^T \cdot x(\sigma).$$

So, $\Phi^l(x; X, \sigma) \leqslant \Phi(x)$, for all $x \in X$, and $\Phi^l(x(\sigma); X, \sigma) = \Phi(x(\sigma))$.

Similarly, we can prove that

$$\Phi^u(x; X, \sigma) \geqslant \Phi(x), \quad \forall x \in X,$$

$$\Phi^u(x(\sigma); X, \sigma) = \Phi(x(\sigma)). \tag{16} \qquad \Box$$

Now, we show how to construct a two-part parametric linearization method to systematically convert (NPP) into a series of linear programming problems by utilizing a transformation and a parametric linear upper bounding function (LUBF) and a linear lower bounding function (LLBF) of a natural logarithm function and an exponential function with e as the base, respectively.

2.2. First-Part Parametric Linear Relaxation. In this subsection, we discuss how to obtain the first-stage relaxation LLBF of $f_{jk}(y_{jk}) = t_{jk} \exp(Y_{jk})$ about the variable y_{jk} by using Theorem 1.

Let X denote either the initial rectangle X^0 or some subrectangle of X^0 that is generated by the proposed algorithm. Without loss of generality, let $X = \{x \mid x_i^l \leqslant x_i \leqslant x_i^u, i = 1, 2, \ldots, n\}$. Denote the lower bound and the upper bound of y_{jkm} by y_{jkm}^l and y_{jkm}^u which can be derived on the presently considered rectangle X in the algorithm. For any j, k, fix a vector $\sigma \in \{0, 1\}^{K_{jk}}$, and for function $f_{jk}(y_{jk}) = \exp(\sum_{m=1}^{K_{jk}} \alpha_{jkm} y_{jkm})$ and interval vector $Y_{jk} = [y_{jk}^l, y_{jk}^u] \subseteq R^{K_{jk}}$, calculate interval vector $Z_{jk} = [z_{jk}^l, z_{jk}^u]$ satisfying inequalities (7) of Theorem 1 in [38], where $z_{jk}^l = (z_{jk1}^l, \ldots, z_{jkK_{jk}}^l)^T$, $z_{jk}^u = (z_{jk1}^u, \ldots, z_{jkK_{jk}}^u)^T$. That is, for any $j = 0, 1, \ldots, p$, $k = 1, \ldots, K_j$, and any $x \in X$, we calculate the following formulas:

$$Y_{jk}^l = \sum_{m=1}^{K_{jk}} \min\left(\alpha_{jkm} y_{jkm}^l, \alpha_{jkm} y_{jkm}^u\right),$$

$$Y_{jk}^u = \sum_{m=1}^{K_{jk}} \max\left(\alpha_{jkm} y_{jkm}^l, \alpha_{jkm} y_{jkm}^u\right),$$

$$z^l_{jkm} = \min \left\{ \alpha_{jkm} \exp \left(Y^l_{jk} \right), \alpha_{jkm} \exp \left(Y^u_{jk} \right) \right\},$$

$$m = 1, 2, \ldots, K_{jk},$$

$$z^u_{jkm} = \max \left\{ \alpha_{jkm} \exp \left(Y^l_{jk} \right), \alpha_{jkm} \exp \left(Y^u_{jk} \right) \right\},$$

$$m = 1, 2, \ldots, K_{jk}. \tag{17}$$

Thus, the vertices of the interval vectors Y_{jk}, Z_{jk} refer to

$$y_{jk}(\sigma) = y^l_{jk} + \sum_{m=1}^{K_{jk}} \sigma_m \left(y^u_{jkm} - y^l_{jkm} \right) e_m,$$

$$z_{jk}(\sigma) = z^l_{jk} + \sum_{m=1}^{K_{jk}} \sigma_m \left(z^u_{jkm} - z^l_{jkm} \right) e_m, \tag{18}$$

respectively, where e_i denotes the ith unit vector. Therefore, by Theorem 1, we can derive parametric linear lower bound functions $f^l_{jk}(y_{jk})$ of $f_{jk}(y_{jk})$ with respect to Y_{jk} as follows:

$$f^l_{jk}(y_{jk})$$

$$= \begin{cases} t_{jk} \left[z_{jk}(\sigma)^T \cdot y_{jk} + f_{jk}\left(y_{jk}(\sigma) \right) \right. \\ \qquad \left. - z_{jk}(\sigma)^T \cdot y_{jk}(\sigma) \right], & \text{if } t_{jk} > 0, \\ t_{jk} \left[z_{jk}(1-\sigma)^T \cdot y_{jk} + f_{jk}\left(y_{jk}(\sigma) \right) \right. \\ \qquad \left. - z_{jk}(1-\sigma)^T \cdot y_{jk}(\sigma) \right], & \text{if } t_{jk} < 0. \end{cases} \tag{19}$$

Let $X_{jk} = c^T_{jkm} x + d_{jkm}$. So, from (5), if $t_{jk} > 0$, the first-part LLBF of $f_{jk}(x)$ denoted by $f^l_{jk}(x)$ about x can get

$$f^l_{jk}(x) = t_{jk} \left[\sum_{m=1}^{K_{jk}} z_{jkm}(\sigma) \cdot \ln \left(X_{jk} \right) \right.$$

$$\left. + f_{jk}\left(y_{jk}(\sigma) \right) - z_{jk}(\sigma)^T \cdot y_{jk}(\sigma) \right], \tag{20}$$

where $z_{jkm}(\sigma)$ denotes the mth component of $z_{jk}(\sigma)$. And if $t_{jk} < 0$,

$$f^l_{jk}(x) = t_{jk} \left[\sum_{m=1}^{K_{jk}} z_{jkm}(1-\sigma) \cdot \ln \left(X_{jk} \right) \right.$$

$$\left. + f_{jk}\left(y_{jk}(\sigma) \right) - z_{jk}(1-\sigma)^T \cdot y_{jk}(\sigma) \right]. \tag{21}$$

2.3. Second-Part Parametric Linear Relaxation. Now, by Theorem 1, we construct the second-part LLBF of $t_{jk} \ln(X_{jk})$ about the variable x. For any interval vector $X = [\underline{x}, \overline{x}] \subseteq X^0 \subset R^n$ and any $x = (x_i)_{n\times 1} \in X$, let $\phi_{jkm}(x) =$

$\ln(c^T_{jkm} x + d_{jkm})$. For convenience, the following notations and functions of this paper are introduced:

$$X^l_{jkm} = \sum_{i=1}^{n} \min \left(c_{jkmi} \underline{x}_i, c_{jkmi} \overline{x}_i \right) + d_{jkm},$$

$$X^u_{jkm} = \sum_{i=1}^{n} \max \left(c_{jkmi} \underline{x}_i, c_{jkmi} \overline{x}_i \right) + d_{jkm},$$

$$\underline{w}_{jkmi} = \min \left\{ c_{jkmi} \left(X^l_{jkm} \right)^{-1}, c_{jkmi} \left(X^u_{jkm} \right)^{-1} \right\},$$

$$\overline{w}_{jkmi} = \max \left\{ c_{jkmi} \left(X^l_{jkm} \right)^{-1}, c_{jkmi} \left(X^u_{jkm} \right)^{-1} \right\}, \tag{22}$$

$$x(\sigma) = \underline{x} + \sum_{i=1}^{n} \sigma_i \left(\overline{x}_i - \underline{x}_i \right) e_i,$$

$$w_{jkm}(\sigma) = \underline{w}_{jkm} + \sum_{i=1}^{n} \sigma_i \left(\overline{w}_{jkmi} - \underline{w}_{jkmi} \right) e_i,$$

where e_i denotes the ith unit vector in R^n. Then, by Theorem 1, for any vector $\sigma \in \{0, 1\}^n$, we define the LLBF of $t_{jk} \phi_{jkm}(x)$ by $t_{jk} \phi^l_{jkm}(x)$ below:

$$t_{jk} \phi^l_{jkm}(x)$$

$$= \begin{cases} t_{jk} \left[w_{jkm}(\sigma)^T x + \phi_{jkm}\left(x(\sigma) \right) \right. \\ \qquad \left. - w_{jkm}(\sigma)^T x(\sigma) \right] & \text{if } t_{jk} > 0, \\ t_{jk} \left[w_{jkm}(1-\sigma)^T x + \phi_{jkm}\left(x(\sigma) \right) \right. \\ \qquad \left. - w_{jkm}(1-\sigma)^T x(\sigma) \right] & \text{if } t_{jk} < 0. \end{cases} \tag{23}$$

Then, if $t_{jk} > 0$, we can construct the LLBF of $t_{jk} \prod_{m=1}^{K_{jk}} \left(c^T_{jkm} x + d_{jkm} \right)^{\alpha_{jkm}}$ as follows:

$$LF^1_{jk}(x)$$

$$= t_{jk} \left\{ \sum_{m=1}^{K_{jk}} z_{jkm}(\sigma) \right.$$

$$\times \left[w_{jkm}(\sigma)^T x + \phi_{jkm}\left(x(\sigma) \right) - w_{jkm}(\sigma)^T x(\sigma) \right]$$

$$\left. + f_{jk}\left(y_{jk}(\sigma) \right) - z_{jk}(\sigma)^T \cdot y_{jk}(\sigma) \right\}$$

$$= t_{jk} \sum_{m=1}^{K_{jk}} z_{jkm}(\sigma) \left[w_{jkm}(\sigma)^T x \right]$$

$$+ t_{jk} \left[f_{jk}\left(y_{jk}(\sigma) \right) - z_{jk}(\sigma)^T \cdot y_{jk}(\sigma) \right]$$

$$+ t_{jk} \sum_{m=1}^{K_{jk}} z_{jkm}(\sigma) \left[\phi_{jkm}\left(x(\sigma) \right) - w_{jkm}(\sigma)^T x(\sigma) \right]. \tag{24}$$

And, for $t_{jk} < 0$, we can get the LLBF of $t_{jk}\prod_{m=1}^{K_{jk}}(c_{jkm}^T x + d_{jkm})^{\alpha_{jkm}}$ as

$$
\begin{aligned}
LF_{jk}^2(x) \\
= t_{jk} \Bigg\{ & \sum_{m=1}^{K_{jk}} z_{jkm}(1-\sigma) \\
& \times \left[w_{jkm}(1-\sigma)^T x + \phi_{jkm}(x(\sigma)) \right. \\
& \left. \qquad - w_{jkm}(1-\sigma)^T x(\sigma) \right] \\
& + f_{jk}\left(y_{jk}(\sigma)\right) - z_{jk}(1-\sigma)^T \cdot y_{jk}(\sigma) \Bigg\}
\end{aligned}
\tag{25}
$$

$$
\begin{aligned}
= t_{jk} & \sum_{m=1}^{K_{jk}} z_{jkm}(1-\sigma)\left[w_{jkm}(1-\sigma)^T x \right] \\
& + t_{jk}\left[f_{jk}\left(y_{jk}(\sigma)\right) - z_{jk}(1-\sigma)^T \cdot y_{jk}(\sigma) \right] \\
& + t_{jk} \sum_{m=1}^{K_{jk}} z_{jkm}(1-\sigma)\left[\phi_{jkm}(x(\sigma)) \right. \\
& \left. \qquad\qquad - w_{jkm}(1-\sigma)^T x(\sigma) \right].
\end{aligned}
$$

Taken together, the LLBF of function $f_j(x)$ with respect to x can be obtained as

$$
LF_j(x) = \sum_{t_{jk}>0} LF_{jk}^1(x) + \sum_{t_{jk}<0} LF_{jk}^2(x), \quad j = 0,1,\ldots,p.
\tag{26}
$$

Obviously, for all $x \in X \subseteq X^0$, $j = 0,1,\ldots,p$, $LF_j(x) \leqslant f_j(x)$.

2.4. Approximation Relaxation Linear Programming. Consequently, the approximation relaxation lower linear programming (LLP) of problem (NPP) with the parametric vector σ in interval vector $X = [\underline{x},\overline{x}] \subseteq X^0 \subset R^N$ is easily obtained like the following:

$$
\begin{aligned}
\text{(LLP)} \quad \min \quad & LF_0(x) \\
\text{s.t.} \quad & Ax \leqslant b, \\
& LF_j(x) \leqslant e_j, \\
& x \in X.
\end{aligned}
\tag{27}
$$

Based on the linear underestimators, every feasible point of (NPP) is feasible in (LLP), and the objective of (LLP) is smaller than or equal to that of (NPP) for all points in X. Thus, (LLP) provides a valid lower bound for the solution of (NPP) over the partition set X. It should be noted that problem (LLP) contains only the necessary constraints to guarantee convergence of the algorithm. The following results are key to the convergence of the proposed algorithm.

Lemma 2. *For all* $j = 0,1,\ldots,p$, $k = 1,2,\ldots,K_j$, $m = 1,2,\ldots,K_{jk}$, *and* $X = [\underline{x},\overline{x}] \subseteq X^0$, *let*

$$
\begin{aligned}
\Delta_{jkm}^1 = \phi_{jkm}(x) - & \left[w_{jkm}(\sigma)^T x + \phi_{jkm}(x(\sigma)) \right. \\
& \left. - w_{jkm}(\sigma)^T x(\sigma) \right], \\
\Delta_{jkm}^2 = & \left[w_{jkm}(1-\sigma)^T x + \phi_{jkm}(x(\sigma)) \right. \\
& \left. - w_{jkm}(1-\sigma)^T x(\sigma) \right] - \phi_{jkm}(x).
\end{aligned}
\tag{28}
$$

Then one has $\lim_{\|\overline{x}-\underline{x}\|\to 0}\Delta_{jkm}^1 = \lim_{\|\overline{x}-\underline{x}\|\to 0}\Delta_{jkm}^2 \to 0$.

Proof. From Theorem 1 and definition of function $\phi_{jkm}(x)$, for any $x \in X$, it follows that

$$
0 \leqslant \Delta_{jkm}^1 = \left(\phi_{jkm}'(\zeta) - w_{jkm}(\sigma) \right)^T (x - x(\sigma)), \quad \forall j,k,m,
\tag{29}
$$

where ϕ_{jkm}' is a gradient function of ϕ_{jkm}, $\zeta = \mu x + (1-\mu x(\sigma))$ for some $\mu \in [0,1]$, and $x(\sigma)$, $w_{jkm}(\sigma)$ are vertices of the interval vectors X and $W_{jkm} := [\underline{w}_{jkm}, \overline{w}_{jkm}]$, respectively. By (6) and proof of Theorem 1, the right-hand side in inequality (29) satisfies for arbitrarily fixed σ

$$
\begin{aligned}
& \left(\phi_{jkm}'(\zeta) - w_{jkm}(\sigma) \right)^T (x - x(\sigma)) \\
& \leqslant \sum_{i=1}^n \left| \frac{\partial \phi_{jkm}(\zeta)}{\partial \zeta_i} - w_{jkmi}(\sigma) \right| \cdot |x_i - x_i(\sigma)| \\
& \leqslant \sum_{i=1}^n \left| w_{jkmi}(0) - w_{jkmi}(1) \right| \cdot |x_i - x_i(\sigma)| \\
& = \sum_{i=1}^n \left| \overline{w}_{jkmi} - \underline{w}_{jkmi} \right| \cdot |x_i - x_i(\sigma)| \\
& = \sum_{i=1}^n \left| c_{jkmi} \right| \cdot \left| \left(X_{jkm}^l\right)^{-1} - \left(X_{jkm}^u\right)^{-1} \right| \cdot |x_i - x_i(\sigma)| \\
& = \sum_{i=1}^n \left| c_{jkmi} \right| \cdot |x_i - x_i(\sigma)| \cdot \left(\kappa_{jkm}\right)^{-1} \cdot \left| X_{jkm}^u - X_{jkm}^l \right| \\
& \leqslant \sum_{i=1}^n \left| c_{jkmi} \right| \cdot |\underline{x}_i - \overline{x}_i| \cdot \left(\kappa_{jkm}\right)^{-1} \cdot \left| X_{jkm}^u - X_{jkm}^l \right| \\
& = \left(\kappa_{jkm}\right)^{-1} \left(\sum_{i=1}^n \left| c_{jkmi} \right| \cdot |\underline{x}_i - \overline{x}_i| \right)^2 \\
& \leqslant \left(\kappa_{jkm}\right)^{-1} \left(\sum_{i=1}^n \left| c_{jkmi} \right| \right)^2 \cdot \|\overline{x} - \underline{x}\|,
\end{aligned}
\tag{30}
$$

where $\kappa_{jkm} = \mu X_{jkm}^l + (1-\mu)X_{jkm}^u$ for some $\mu \in [0,1]$. It shows that

$$
\lim_{\|\overline{x}-\underline{x}\|\to 0} \Delta_{jkm}^1 \longrightarrow 0.
\tag{31}
$$

Similarly, we can prove that $\lim_{\|\overline{x}-\underline{x}\|\to 0}\Delta_{jkm}^2 \to 0$. $\square$

Similarly, we have Lemma 3 (also see Lemma 1 in [38]).

Lemma 3. *For all* $j, k, Y_{jk} = [y_{jk}^l, y_{jk}^u] \subseteq R^{K_{jk}}$, *let*

$$\Delta_{jk}^3 = \exp\left(\sum_{m=1}^{K_{jk}} \alpha_{jkm} y_{jkm}\right) - \left[z_{jk}(\sigma)^T \cdot y_{jk} + f_{jk}\left(y_{jk}(\sigma)\right)\right.$$

$$\left. - z_{jk}(\sigma)^T \cdot y_{jk}(\sigma)\right],$$

$$\Delta_{jk}^4 = z_{jk}(1-\sigma)^T \cdot y_{jk} + f_{jk}\left(y_{jk}(\sigma)\right) - z_{jk}(1-\sigma)^T$$

$$\cdot y_{jk}(\sigma) - \exp\left(\sum_{m=1}^{K_{jk}} \alpha_{jkm} y_{jkm}\right). \tag{32}$$

Then $\lim_{\|y_{jk}^u - y_{jk}^l\| \to 0} \Delta_{jk}^3 = \lim_{\|y_{jk}^u - y_{jk}^l\| \to 0} \Delta_{jk}^4 \to 0$.

Theorem 4. *For any* j, k, m, *let* $\delta_{jk} = y_{jk}^u - y_{jk}^l, X = [\underline{x}, \overline{x}] \subseteq X^0$, *and* $Y_{jk} = [y_{jk}^l, y_{jk}^u] \subseteq R^{K_{jk}}$. *Then, when* $\|\overline{x} - \underline{x}\| \to 0$, *for any* $x \in X$, *the difference of* $LF_j(x)$ *and* $f_j(x)$ *satisfies* $f_j(x) - LF_j(x) \to 0$.

Proof. Firstly, notice that $\|y_{jk}^u - y_{jk}^l\| \to 0$ when $\|\overline{x} - \underline{x}\| \to 0$. Then, for any $x \in X$, and for any j, let

$$\Delta = f_j(x) - LF_j(x) = \sum_{t_{jk} > 0} \left(f_j(x) - LF_j^1(x)\right)$$

$$+ \sum_{t_{jk} < 0} \left(f_j(x) - LF_j^2(x)\right), \tag{33}$$

and let $\Delta_{jk}^1 = f_j(x) - LF_j^1(x)$ and $\Delta_{jk}^2 = f_j(x) - LF_j^2(x)$. Therefore, we only need to prove $\Delta_{jk}^1 \to 0, \Delta_{jk}^2 \to 0$ as $\|\overline{x} - \underline{x}\| \to 0$.

We first prove $\Delta_{jk}^1 \to 0$. Since

$$\Delta_{jk}^1 = \left(f_j(x) - f_{jk}^l(x)\right) + \left(f_{jk}^l(x) - LF_{jk}^1(x)\right)$$

$$= \Delta_{jk1}^1 + \Delta_{jk2}^1, \tag{34}$$

it is obvious that we only need to prove $\Delta_{jk1}^1 \to 0$ and $\Delta_{jk2}^1 \to 0$. We first consider the difference Δ_{jk1}^1. By the definition of $LF_{jk}^1(x), f_{jk}^l(x)$, it follows that

$$\Delta_{jk1}^1 = f_j(x) - f_{jk}^l(x)$$

$$= t_{jk} \left[\prod_{m=1}^{K_{jk}} \left(c_{jkm}^T x + d_{jkm}\right)^{\alpha_{jkm}} - z_{jk}(\sigma)^T \cdot y_{jk}\right.$$

$$\left. - f_{jk}\left(y_{jk}(\sigma)\right) + z_{jk}(\sigma)^T \cdot y_{jk}(\sigma)\right]$$

$$= t_{jk} \left[\exp\left(\sum_{m=1}^{K_{jk}} \alpha_{jkm} y_{jkm}\right) - z_{jk}(\sigma)^T \cdot y_{jk}\right.$$

$$\left. - f_{jk}\left(y_{jk}(\sigma)\right) + z_{jk}(\sigma)^T \cdot y_{jk}(\sigma)\right]$$

$$= t_{jk} \Delta_{jk}^3, \tag{35}$$

where $X_{jk} = c_{jkm}^T + d_{jkm}$. Then, by Lemma 2, $\Delta_{jk1}^1 \to 0$ as $\|\overline{x} - \underline{x}\| \to 0$.

Now, the difference $\Delta_{jk2}^1 = f_{jk}^l(x) - LF_{jk}^1(x)$ is considered. From the definition of $f_{jk}^l(x), LF_{jk}^1(x)$, we can obtain

$$\Delta_{jk2}^1 = f_{jk}^l(x) - LF_{jk}^1(x)$$

$$= t_{jk} \sum_{m=1}^{K_{jk}} z_{jkm}(\sigma) \left[\ln\left(X_{jk}\right) - w_{jkm}(\sigma)^T x\right.$$

$$\left. - \phi_{jkm}(x(\sigma)) + w_{jkm}(\sigma)^T x(\sigma)\right]$$

$$= t_{jk} \sum_{m=1}^{K_{jk}} z_{jkm}(\sigma) \cdot \Delta_{jkm}^1. \tag{36}$$

Then, by Lemma 3, $\Delta_{jk2}^1 \to 0$ as $\|\overline{x} - \underline{x}\| \to 0$. Therefore, when $\|\overline{x} - \underline{x}\| \to 0$, we can get

$$\Delta_{jk}^1 = f_j(x) - LF_j^1(x) = \Delta_{jk1}^1 + \Delta_{jk2}^1 \longrightarrow 0. \tag{37}$$

By similar discussion as above, we can get

$$\Delta_{jk}^2 = f_j(x) - LF_j^2(x) \longrightarrow 0, \quad \text{as } \|\overline{x} - \underline{x}\| \longrightarrow 0. \tag{38}$$

It follows from (37) and (38) that $f_j(x) - LF_j(x) \to 0$ when $\|\overline{x} - \underline{x}\| \to 0$. □

Theorem 4 shows that as the subhyperrectangle $X \subseteq X^0$ is small enough, the solution to $(LLP)(X)$ is sufficiently approaching the solution of $(NPP)(X)$ and this guarantees the global convergence of the method.

3. Algorithm and Its Convergence

In this section, a branch-and-bound algorithm is developed to solve (NPP) based on the relaxation lower linear programming in Section 2. This algorithm needs to solve a sequence of linear programming over partitioned subsets of X in order to find a global optimum. Consequently, this method needs partitioning the set X^0 into subhyperrectangles, each concerned with a node of the branch-and-bound tree, and each node is associated with a relaxation linear subproblem in each subhyperrectangle.

First, at any stage k of the algorithm, suppose that we have a collection of active nodes denoted by Q_k, say, each

associated with a subhyperrectangle $X \subset X^0$, for all $X \in Q_k$. For each node X, we will have computed a lower bound of the optimal value of the problem $((NPP)(X))$ via solution $LB(X)$ of problem (LLP) so that the lower bound of optimal value of (NPP) on the whole initial box region X^0 is given by $LB_k = \min\{LB(X) \mid \forall X \in Q_k\}$ at stage k. Whenever the lower bounding solution to any node subproblem; that is, the solution to the relaxation linear programming (LLP), turns out to be feasible to (NPP), we update the upper bound of incumbent solution UB if necessary. Then, the active nodes collection Q_k will satisfy $UB \geqslant LB(X)$, for all $X \in Q_k$, for each stage k. We now select an active node $X \in Q_k$ such that $LB(X) = LB_k$ for further considering. The active node X is partitioned into two subhyperrectangles according to the following branching rules. For these two subhyperrectangles, the fathoming step is applied in order to identify whether the subhyperrectangles should be eliminated. Finally, we obtain a collection of active nodes for the next stage, and this process is repeated until convergence is obtained.

3.1. Branching Rule.
The critical element in guaranteeing convergence to a global minimum means the choice of a suitable partitioning strategy. In our paper, we choose a simple and standard bisection rule. This method is sufficient to ensure convergence since it drives all the intervals to zero for the variables that are associated with the term yielding the greatest discrepancy in the employed approximation along with any infinite branch of a branch-and-bound tree.

Consider any node subproblem identified by the hyperrectangle $X = \{\underline{x}, \overline{x}\} \subseteq X^0$ and the selection of branching variable x_p and partitioning of X is then done by using the following rule (see also [39, 40]). Let $p = \arg\max\{\overline{x}_i - \underline{x}_i; i = 1, 2, \ldots, n\}$, partitioning X by bisectioning the interval $[\underline{x}_p, \overline{x}_p]$ into the subintervals $[\underline{x}_p, (\underline{x}_p + \overline{x}_p)/2]$ and $[(\underline{x}_p + \overline{x}_p)/2, \overline{x}_p]$.

3.2. Algorithmic Statement.
The deterministic global optimization algorithm is summarized as follows.

Step 0 (initialization).

(0.1) Initialize the iteration counter $k := 0$, the set of all active nodes $Q_0 = \{X^0\}$, the upper bound $\beta = +\infty$, and the set of feasible points $F := \emptyset$.

(0.2) Solve (LLP) with $X = X^0$ in order to find an optimal solution x^0 and the optimal value $\alpha(X^0)$. If x^0 is feasible to (NPP), then set $\beta = f(x^0)$, $F = F \cup \{x^0\}$, and $\alpha_0 = \alpha(X^0)$, if necessary.

(0.3) If $\beta \leqslant \alpha_0 + \varepsilon$, where $\varepsilon > 0$ is some accuracy tolerance, then stop. x^0 is global ε-optimal solution to (NPP). Otherwise, set $k = 1$ and proceed to Step 1.

Step 1 (partitioning step). According to the rectangle bisection rule, select a branching variable x_p to partition X^k to get two

new subhyperrectangles $X^{k,1}, X^{k,2} \subseteq R^n$. Call the set of new partition rectangles as $\overline{X}^k = \{X^{k,1}, X^{k,2}\}$.

Step 2 (feasibility check for (NPP) in subhyperrectangles). For each new node $X = [\underline{x}, \overline{x}] \in \overline{X}^k$, for each $l = 1, 2, \ldots, q$, compute the lower bound for any linear constraint function $\sum_{i=1}^{n} a_{li} x_i$ only according to the present considered rectangle; that is, compute lower bound $\sum_{a_{li}>0} a_{li}\underline{x}_i + \sum_{a_{li}<0} a_{li}\overline{x}_i$. If there exists some $l \in \{1, \ldots, q\}$ such that

$$\sum_{a_{li}>0} a_{li}\underline{x}_i + \sum_{a_{li}<0} a_{li}\overline{x}_i > b_l, \tag{39}$$

then the corresponding subrectangle X is eliminated from $\overline{X}^k$; that is, $\overline{X}^k = \overline{X}^k \setminus \{X\}$, and skip to next element of $\overline{X}^k$.

Step 3 (bounding step). If $\overline{X}^k = \emptyset$, go to Step 5. If $\overline{X}^k \neq \emptyset$, solve LLP$(X)$ to obtain $\alpha(X)$ and $x(X)$ for each $X \in \overline{X}^k$. If $\alpha(X) > \beta$, set $\overline{X}^k = \overline{X}^k \setminus \{X\}$. Otherwise, if $x(X)$, is feasible to (NPP), then update β and F, if necessary.

Step 4 (updating the upper bound). Select the midpoint x^{mid} of X^k; if x^{mid} is feasible to (NP)(X^k), then $F := F \cup \{x^{\mathrm{mid}}\}$. Define the upper bound $\beta := \min_{x \in F} f(x)$. If $F \neq \phi$, the best known feasible point is denoted by $x^{\mathrm{best}} := \arg\min_{x \in F} f(x)$.

Step 5 (updating the lower bound). The partition set remaining is now $Q_k := (Q_k \setminus \{X^k\}) \cup \overline{X}^k$ and a new lower bound is $\alpha_k := \min_{X \in Q_k} \beta(X)$.

Step 6 (convergence checking). Set $Q_{k+1} = Q_k \setminus \{X : \beta - \alpha_k \leqslant \varepsilon, X \in Q_k\}$. If $Q_{k+1} = \phi$, then stop with β as the solution of (NPP) and x^{best} as an optimal solution. Otherwise select an active node X^{k+1} such that $X^{k+1} = \arg\min_{X \in Q_{k+1}} \alpha(X)$, $x^k = x(X^k)$. Set $k := k + 1$ and go to Step 1.

3.3. Convergence of the Algorithm.
By Theorem 4, global algorithm convergence will be given in Theorem 5.

Theorem 5. *The above algorithm either terminates finitely with the incumbent solution being optimal to (NPP) or generates an infinite sequence of iterations such that, along with any infinite branch of the branch-and-bound tree, any accumulation point of sequence x^k will be global solution to (NPP).*

Proof. If the above proposed algorithm terminates finitely, obviously α is a global optimal value and x^{best} is optimal solution for the (NPP). If the algorithm is infinite, it generates at least one infinite sequence $\{X^k\}$ such that $X^{k+1} \subset X^k$ for any k. Then, from [39, 40], $\bigcap_k X^k = \{\widehat{x}\}$ for some point

TABLE 1: Computational results of Examples 1–5.

Example	Methods	Optimal solution	Optimal value	Iter.	$L_{\max}$
Example 1	[13]	(0, 0, 1.25)	2.9311923	25	
	Ours	(0, 0, 1.25)	2.9311923	15	4
Example 2	[13]	(0.000, 1.111, 0.000)	4.217	31	
	Ours	(0.000, 1.111, 0.000)	4.217	20	5
Example 3	[14]	(1.5, 1.5, 0)	7.96324	52	3
	Ours	(3.0, 4.0, 0)	5.7606445	55	8
Example 4	[11]	(1.0, 1.0)	1.3463824	26	4
	[12]	(1.0, 1.0)	1.3463824	1	0
	Ours	(1.0, 1.0)	1.3463824	24	2
Example 5	[11]	(1.0, 1.0)	288.0	43	5
	[12]	(1.0, 1.0)	288.0	1	0
	Ours	(1.0, 1.0)	288.0	42	5

$\hat{x} \in R^n$. For every iteration of the algorithm, the following results are true:

$$\alpha_k \leqslant \min_{x \in X^0} f(x), \qquad X^k \in \arg\min_{X \in Q_k} \alpha(X),$$

$$x^k = x\left(X^k\right) \in X^k, \quad k = 0, 1, \ldots. \tag{40}$$

Since $\{x^k\}$ is contained in a compact set X^0, there must be one convergent subsequence $\{x^s\} \subseteq \{x^k\}$ and assume $\lim_{s \to \infty} x^s = \hat{x}$. Then from the proposed algorithm, there exists a decreasing subsequence $\{X^r\} \subset \{X^s\}$ where $X^r \in Q_r$ with $x^r \in X^r$, $\alpha_r = \alpha(X^r) = LF_0(x^r)$, and $\lim_{r \to \infty} X^r = \{\hat{x}\}$. According to Theorem 5, we have $\lim_{r \to \infty} \alpha_r = \lim_{r \to \infty} LF_0(x^r) = \lim_{r \to \infty} f_0(x^r) = f_0(\hat{x})$.

Then all what remains is to prove that $\hat{x}$ is feasible to (NPP)(X^0). First, it is obvious that $\hat{x} \in X^0$ since X^0 is closed. Secondly, by the algorithm, we can obtain that, for all r, x^r is feasible solution to (NPP); that is, $Ax^r \leqslant b$. Taking limits over r in this inequality yields $A\hat{x} \leqslant b$. The remainder of the proof will be by contradiction. Assume that $f_j(\hat{x}) > e_j$ for some $j = 1, 2, \ldots, p$. Because function $LF_j(x)$ is continuous and again from Theorem 4, the sequence $\{LF_j(x^r)\}$ converges to $f_j(\hat{x})$; then by definition of convergence, there must be $\bar{r}$, such that $|LF_j(x^r) - f_j(\hat{x})| < f_j(\hat{x}) - e_j$ for any $r > \bar{r}$. Therefore, for any $r > \bar{r}$, we have $f_j(\hat{x}) > e_j$, which implies that LLP(X^r) is infeasible and violating the assumption that $x^r = x(X^r)$. This is a contradiction, and thus the theorem is completed. □

4. Numerical Experiments

To verify performance of the proposed global optimization algorithm, some test problems were implemented. The test problems are coded in C++ and the experiments are conducted on a Pentium IV (3.06 GHZ) microcomputer. Set $\varepsilon = 0.000001$. The results of Examples 1–5 are summarized in Table 1. In Table 1, the notations have been used for row headers: Iter.: number of algorithm iterations; $L_{\max}$: the maximal length of the enumeration tree.

Example 1 (see [13]). Consider

$$\min \quad y = \frac{3x_1 + 5x_2 + 3x_3 + 50}{3x_1 + 4x_2 + 5x_3 + 50} + \frac{3x_1 + 4x_2 + 50}{4x_1 + 3x_2 + 2x_3 + 50}$$

$$+ \frac{4x_1 + 2x_2 + 4x_3 + 50}{5x_1 + 4x_2 + 3x_3 + 50}$$

s.t. $\quad 6x_1 + 3x_2 + 3x_3 \leqslant 10, \qquad 10x_1 + 3x_2 + 8x_3 \leqslant 10,$

$$x_1, x_2, x_3 \geqslant 0. \tag{41}$$

Example 2 (see [13]). Consider

$$\min \quad y = \frac{4x_1 + 3x_2 + 3x_3 + 50}{3x_2 + 3x_3 + 50} + \frac{3x_1 + 4x_3 + 50}{4x_1 + 4x_2 + 5x_3 + 50}$$

$$+ \frac{x_1 + 2x_2 + 5x_3 + 50}{x_1 + 5x_2 + 5x_3 + 50} + \frac{x_1 + 2x_2 + 4x_3 + 50}{5x_2 + 4x_3 + 50}$$

s.t. $\quad 2x_1 + x_2 + 5x_3 \leqslant 10, \qquad x_1 + 6x_2 + 3x_3 \leqslant 10,$

$$5x_1 + 9x_2 + 2x_3 \leqslant 10, \qquad 9x_1 + 7x_2 + 3x_3 \leqslant 10,$$

$$x_1, x_2, x_3 \geqslant 0. \tag{42}$$

Example 3 (see [14]). Consider

$$\min \quad \left(\frac{13x_1 + 13x_2 + 13}{37x_1 + 73x_2 + 13}\right)^{-1.4} \times \left(\frac{63x_1 - 18x_2 + 39}{13x_1 + 26x_2 + 13}\right)^{1.2}$$

$$- \left(\frac{x_1 + 2x_2 + 5x_3 + 50}{x_1 + 5x_2 + 5x_3 + 50}\right)^{0.5}$$

$$\times \left(\frac{x_1 + 2x_2 + 4x_3 + 50}{5x_2 + 4x_3 + 50}\right)^{-2}$$

s.t. $\quad 2x_1 + x_2 + 5x_3 \leqslant 10, \qquad 5x_1 - 3x_2 = 3,$

$$1.5 \leqslant x_1 \leqslant 3, \ x_1, x_2, x_3 \geqslant 0. \tag{43}$$

TABLE 2: Computational results of Example 6.

Example	(m, n)	Iter.	Time
1	(10, 10)	41.8	33.7
2	(10, 20)	43.2	35.3
3	(20, 20)	69.1	46.5
4	(20, 30)	75.6	49.9
5	(30, 10)	88.7	57.0
6	(30, 20)	91.8	61.7
7	(40, 10)	118.2	78.4
8	(40, 20)	125.0	82.5
9	(50, 10)	138.6	88.9
10	(50, 20)	146.7	92.0

Example 4 (see [11, 12]). Consider

$$
\min \quad \left(\frac{x_1 + x_2 + 1}{x_1 + x_2 + 2}\right)^{1.1} \times \left(\frac{x_1 + x_2 + 3}{x_1 + x_2 + 4}\right)^{1.2}
$$

$$
+ \left(\frac{x_1 + x_2 + 5}{x_1 + x_2 + 6}\right)^{1.1} \times \left(\frac{x_1 + x_2 + 7}{x_1 + x_2 + 8}\right)^{1.2} \tag{44}
$$

$$
\text{s.t.} \quad x_1 x_2^2 + x_1^2 x_2 \leq 10,
$$

$$
1.0 \leq x_1 \leq 2.0, \quad 1.0 \leq x_2 \leq 2.0.
$$

Example 5 (see [11, 12]). Consider

$$
\min \quad (2x_1 + x_2 + 1)^2 (2x_1 + 2x_2 + 1)^2
$$

$$
- (x_1 + 2x_2 + 1)^2 (x_1 + 3x_2 + 3)
$$

$$
\text{s.t.} \quad (2x_1 + 2x_2 + 1)(x_1 + 2x_2 + 1)^2
$$

$$
+ 2(x_1 + x_2 + 1)^{1.5}(2x_1 + 2x_2)^2 \leq 200, \tag{45}
$$

$$
(x_1 + x_2 + 1)^{1.1}(1.5x_1 + x_2 + 2)^{1.2}
$$

$$
- (2x_1 + 2x_2 + 1)(2x_1 + x_2 + 3) \leq 30,
$$

$$
1 \leq x_1 \leq 3, \quad 1 \leq x_2 \leq 3.
$$

Example 6. In this example, we solve 10 different random instances:

$$
\min \quad \sum_{k=1}^{K} t_k \prod_{j=1}^{K_k} \left(c_{kj}^T x + d_{kj}\right)^{\alpha_{kj}}
$$

$$
\text{s.t.} \quad Ax \leq b, \tag{46}
$$

$$
x \in X^0 = [\underline{x}, \overline{x}] \subset R^n,
$$

where $K = 4$, $K_k = 3$, A is $m \times n$ matrix, and all elements of t_k, c_{kj}, d_{kj}, α_{km}, A, b, $\underline{x}$, and $\overline{x}$ are randomly generated, whose ranges are $[-2, 2]$. Table 2 summarizes our computational results. In Table 2, the following indices characterize performance in algorithm: (m, n): the dimensions of the matrix A; Iter.: the average number of iterations; time: the average execution time in seconds.

5. Conclusion

In this paper, a global optimization algorithm is presented to a class of nonconvex programming problems (NPP). A transformation and a two-part parametric linearization technique are employed to initial (NPP), and (NPP) is reduced to a parametric relaxation in lower linear programming based on the linear lower bounding of the objective function and nonlinear constraint functions. Thus the initial (NPP) is reduced to a sequence of linear programming problems through the successive refinement in a linear relaxation of feasible region in an objective function. The algorithm can obtain finite convergence to the global minimum through the successive refinement of the feasible region and the subsequent solutions to a series of linear programming problems. The proposed algorithm is applied to several test problems. In all cases, convergence to the global minimum is achieved.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

References

[1] S. Schaible and T. Ibaraki, "Fractional programming," *European Journal of Operational Research*, vol. 12, no. 4, pp. 325–338, 1983.

[2] C. D. Maranas, I. P. Androulakis, C. A. Floudas, A. J. Berger, and J. M. Mulvey, "Solving long-term financial planning problems via global optimization," *Journal of Economic Dynamics & Control*, vol. 21, no. 8-9, pp. 1405–1425, 1997.

[3] J. M. Mulvey, R. J. Vanderbei, and S. A. Zenios, "Robust optimization of large-scale systems," *Operations Research*, vol. 43, no. 2, pp. 264–281, 1995.

[4] M. C. Dorneich and N. V. Sahinidis, "Global optimization algorithms for chip design and compaction," *Engineering Optimization*, vol. 25, no. 2, pp. 131–154, 1995.

[5] K. P. Bennett and O. L. Mangasarian, "Bilinear separation of two sets in *n*-space," *Computational Optimization and Applications*, vol. 2, no. 3, pp. 207–227, 1993.

[6] Z. Drezner, S. Schaible, and D. Simchi-Levi, "Queueing-location problems on the plane," *Naval Research Logistics*, vol. 37, no. 6, pp. 929–935, 1990.

[7] S. Zhang, "Stochastic Queue Location Problems," Tinbergen Institute Research Series 14, Econometric Institute, Erasmus University, Rotterdam, The Netherlands, 1991.

[8] H. Konno and M. Inori, "Bond portfolio optimization by bilinear fractional programming," *Journal of the Operations Research Society of Japan*, vol. 32, no. 2, pp. 143–158, 1989.

[9] H. M. Markowitz, *Portfolio Selection*, Basil Blackwell, Oxford, UK, 2nd edition, 1991.

[10] H. W. Zhang, W. L. Xu, S. L. Di, and P. F. Thomson, "Quadratic programming method in numerical simulation of metal forming process," *Computer Methods in Applied Mechanics and Engineering*, vol. 191, no. 49-50, pp. 5555–5578, 2002.

[11] H. Jiao, "A branch and bound algorithm for globally solving a class of nonconvex programming problems," *Nonlinear Analysis. Theory, Methods & Applications*, vol. 70, no. 2, pp. 1113–1123, 2009.

[12] P. Shen, X. Bai, and W. Li, "A new accelerating method for globally solving a class of nonconvex programming problems," *Nonlinear Analysis. Theory, Methods & Applications*, vol. 71, no. 7-8, pp. 2866–2876, 2009.

[13] T. Kuno, "A revision of the trapezoidal branch-and-bound algorithm for linear sum-of-ratios problems," *Journal of Global Optimization*, vol. 33, no. 2, pp. 215–234, 2005.

[14] P.-P. Shen and C.-F. Wang, "Global optimization for sum of generalized fractional functions," *Journal of Computational and Applied Mathematics*, vol. 214, no. 1, pp. 1–12, 2008.

[15] H. Konno and T. Kuno, "Generalized linear multiplicative and fractional programming," *Annals of Operations Research*, vol. 25, no. 1-4, pp. 147–161, 1990.

[16] H. Konno and T. Kuno, "Linear multiplicative programming," *Mathematical Programming*, vol. 56, no. 1, pp. 51–64, 1992.

[17] P. Shen and H. Jiao, "Linearization method for a class of multiplicative programming with exponent," *Applied Mathematics and Computation*, vol. 183, no. 1, pp. 328–336, 2006.

[18] X.-G. Zhou and K. Wu, "A method of acceleration for a class of multiplicative programming problems with exponent," *Journal of Computational and Applied Mathematics*, vol. 223, no. 2, pp. 975–982, 2009.

[19] H.-S. Ryoo and N. V. Sahinidis, "Global optimization of multiplicative programs," *Journal of Global Optimization*, vol. 26, no. 4, pp. 387–418, 2003.

[20] S. Schaible, "A note on the sum of a linear and linear-fractional function," *Naval Research Logistics Quarterly*, vol. 24, pp. 691–693, 1977.

[21] T. Matsui, "NP-hardness of linear multiplicative programming and related problems," *Journal of Global Optimization*, vol. 9, no. 2, pp. 113–119, 1996.

[22] H. Konno, Y. Yajima, and T. Matsui, "Parametric simplex algorithms for solving a special class of nonconvex minimization problems," *Journal of Global Optimization*, vol. 1, no. 1, pp. 65–81, 1991.

[23] A. Cambini, L. Martein, and S. Schaible, "On maximizing a sum of ratios," *Journal of Information & Optimization Sciences*, vol. 10, no. 1, pp. 65–79, 1989.

[24] H. Konno, T. Kuno, and Y. Yajima, "Global minimization of a generalized convex multiplicative function," *Journal of Global Optimization*, vol. 4, no. 1, pp. 47–62, 1994.

[25] H. Konno and H. Yamashita, "Minimizing sums and products of linear fractional functions over a polytope," *Naval Research Logistics*, vol. 46, no. 5, pp. 583–596, 1999.

[26] M. Dür, R. Horst, and N. Van Thoai, "Solving sum-of-ratios fractional programs using efficient points," *Optimization*, vol. 49, no. 5-6, pp. 447–466, 2001.

[27] H. Konno and K. Fukaishi, "A branch and bound algorithm for solving low rank linear multiplicative and fractional programming problems," *Journal of Global Optimization*, vol. 18, no. 3, pp. 283–299, 2000.

[28] I. Quesada and I. E. Grossmann, "A global optimization algorithm for linear fractional and bilinear programs," *Journal of Global Optimization*, vol. 6, no. 1, pp. 39–76, 1995.

[29] T. Kuno, "A branch-and-bound algorithm for maximizing the sum of several linear ratios," *Journal of Global Optimization*, vol. 22, no. 1–4, pp. 155–174, 2002.

[30] H. P. Benson, "A simplicial branch and bound duality-bounds algorithm for the linear sum-of-ratios problem," *European Journal of Operational Research*, vol. 182, no. 2, pp. 597–611, 2007.

[31] N. T. H. Phuong and H. Tuy, "A unified monotonic approach to generalized linear fractional programming," *Journal of Global Optimization*, vol. 26, no. 3, pp. 229–259, 2003.

[32] S. Schaible and J. Shi, "Fractional programming: the sum-of-ratios case," *Optimization Methods & Software*, vol. 18, no. 2, pp. 219–229, 2003.

[33] H. D. Sherali, "Global optimization of nonconvex polynomial programming problems having rational exponents," *Journal of Global Optimization*, vol. 12, no. 3, pp. 267–283, 1998.

[34] H. D. Sherali and C. H. Tuncbilek, "New reformulation linearization/convexification relaxations for univariate and multivariate polynomial programming problems," *Operations Research Letters*, vol. 21, no. 1, pp. 1–9, 1997.

[35] N. Z. Shor, *Nondifferentiable Optimization and Polynomial Problems*, Kluwer Academic Publishers, Dordrecht, The Netherlands, 1992.

[36] J. B. Lasserre, "Global optimization with polynomials and the problem of moments," *SIAM Journal on Optimization*, vol. 11, no. 3, pp. 796–817, 2000/01.

[37] J. B. Lasserre, "Semidefinite programming versus LP relaxations for polynomial programming," *Mathematics of Operations Research*, vol. 27, no. 2, pp. 347–360, 2002.

[38] P. Shen, "Linearization method of global optimization for generalized geometric programming," *Applied Mathematics and Computation*, vol. 162, no. 1, pp. 353–370, 2005.

[39] H. Tuy, *Convex Analysis and Global Optimization*, vol. 22, Kluwer Academic Publishers, Dordrecht, The Netherlands, 1998.

[40] R. Horst and H. Tuy, *Global Optimization: Deterministic Approaches*, Springer, Berlin, Germany, 2nd edition, 1993.