Hindawi
Mathematical Problems in Engineering
Volume 2018, Article ID 9146309, 8 pages
https://doi.org/10.1155/2018/9146309

Research Article

Global Optimization for Generalized Linear Multiplicative Programming Using Convex Relaxation

Yingfeng Zhao and Ting Zhao

School of Mathematical Science, Henan Institute of Science and Technology, Xinxiang 453003, China

Correspondence should be addressed to Yingfeng Zhao; zhaoyingfeng6886@163.com

Received 10 February 2018; Accepted 1 April 2018; Published 10 May 2018

Academic Editor: Guillermo Cabrera-Guerrero

Copyright © 2018 Yingfeng Zhao and Ting Zhao. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Applications of generalized linear multiplicative programming problems (LMP) can be frequently found in various areas of engineering practice and management science. In this paper, we present a simple global optimization algorithm for solving linear multiplicative programming problem (LMP). The algorithm is developed by a fusion of a new convex relaxation method and the branch and bound scheme with some accelerating techniques. Global convergence and optimality of the algorithm are also presented and extensive computational results are reported on a wide range of problems from recent literature and GLOBALLib. Numerical experiments show that the proposed algorithm with a new convex relaxation method is more efficient than usual branch and bound algorithm that used linear relaxation for solving the LMP.

1. Introduction

This paper deals with finding global optimal solutions for generalized linear multiplicative programming problems of the form

$$(\mathrm{LMP}):\quad \begin{cases} \min & f_0(x) = \sum_{j=1}^{p_0} \phi_{0j}(x)\,\psi_{0j}(x) \\ \text{s.t.} & f_i(x) = \sum_{j=1}^{p_i} \phi_{ij}(x)\,\psi_{ij}(x) \le 0,\ i = 1,2,\ldots,M, \\ & f_i(x) = \sum_{j=1}^{p_i} \phi_{ij}(x)\,\psi_{ij}(x) \ge 0,\ i = M+1, M+2,\ldots,N, \\ & x \in D = \{x \in R^n \mid Ax \le b,\ x \ge 0\}, \end{cases} \tag{1}$$

where $\phi_{ij}(x), \psi_{ij}(x)$ in objective and constraints are linear functions with general forms $\phi_{ij}(x) = \sum_{k=1}^{n} a_{ijk}x_k + b_{ij}$, $\psi_{ij}(x) = \sum_{k=1}^{n} c_{ijk}x_k + d_{ij}$, a_{ijk}, c_{ijk}, b_{ij}, and d_{ij} are all arbitrary real numbers, $i = 0, 1, 2, \ldots, N$, $j = 1, 2, \ldots, p_i$, $k = 1, 2, \ldots, n$, $A \in R^{m\times n}$ is a matrix, $b \in R^m$ is a vector, and set $D \subset R^n$ is nonempty and bounded.

Generally, linear multiplicative programming problem (LMP) is a special case of nonconvex programming problem, known to be NP-hard. Linear multiplicative programming problem (LMP) has attracted considerable attention in the literature because of its large number of practical applications in various fields of study, including financial optimization in Konno et al. [1], VLSI chip design in Dorneich and Sahinidis [2], data mining and pattern recognition in Bennett and Mangasarian [3], plant layout design in Quesada and Grossmann [4], marketing and service planning in Samadi et al. [5], robust optimization in Mulvey et al. [6], multiple-objective decision in Benson [7], in Keeney and Raika [8], in Geoffrion [9], location-allocation problems in in Konno et al. [10], constrained bimatrix games in Mangasarian [11], three-dimensional assignment problems in Frieze [12], certain linear max-min problems in Falk [13], and many problems in engineering design, economic management, and operations research. Another reason why this problem attracts so much attention is that, by utilizing some techniques, many mathematical programs like general quadratic programming, bilinear programming, linear multiplicative programming, and quadratically constrained quadratic programming can be converted into the special form of LMP in Benson [14]; in Tuy [15], the important class of sum of ratios fractional problems can also be encapsulated by LMP; in fact, suppose that f_i are convex and g_i are concave and positive. Since each $1/g_i(x)$ is convex and positive, then the sum of ratios fractional problem

(with the objective function as $\sum(f_i(x)/g_i(x))$) reduces to LMP when g_i and f_i are linear. When g_i and f_i are negative, one can add a large real member to make negative be positive because of the compactness of the constraint set in Dai et al. [16].

Moreover, from the algorithmic design point of view, sum of products of two affine functions need not be convex (even not be quasi-convex), and hence linear multiplicative programming problem (LMP) may have multiple local solutions, which fail to be global optimal. Due to the facts above, developing practical global optimization algorithms for the LMP has great theoretical and the algorithmic significance.

In the past 20 years, many solution methods have been proposed for globally solving linear multiplicative programming problem (LMP) and its special cases. The methods are mainly classified as parameterization-based methods, outer-approximation methods, kinds of branch-and-bound methods, decomposition method, and cutting plane methods in Konno et al. [17], in Thoai [18], in Shen and Jiao [19, 20], in Wang et al. [21], in Jiao et al. [22], and in Shen et al. [23]. Most of these methods for linear multiplicative programming are designed only for problems in which the constraint functions are all linear or under the assumption that all linear functions that appeared in the multiplicative terms are nonnegative. And most branch-and-bound algorithms for this problem are developed based on two-phase linear relaxation scheme, which will take more computational time in the approximation process. For example, in Zhao and Liu [24] and in Jiao et al. [22], both algorithms utilize two-phase relaxation methods, and a large amount of computational time is consumed in the relaxation process. Compared with these algorithms, the main features of our new algorithm are threefold. (1) The problem investigated in this paper is linear multiplicative constrained multiplicative programming; it has a more general form than those linear multiplicative programming problems with linear constraints. (2) The relaxation programming problem is a convex programming which can be obtained with one-step relaxation; this will greatly improve the efficiency of approximation process. (3) The condition $\phi_{ij}(x) > 0$, $\psi_{ij}(x) > 0$, $\forall x \in X$, is a nonnecessitating requirement in our algorithm. (4) Extensive computational numerical examples and comparison from recent literature and GLOBALLib are performed to test our algorithm for LMP.

The rest of this paper is arranged in the following way. In Section 2, the construction process of the convex relaxation problem is detailed, which will provide a reliable lower bound for the optimal value of LMP. In Section 3, some key operations for designing a branch-and-bound algorithm and the global optimization algorithm for LMP are described. Convergence property of the algorithm is also established in this section. Numerical results are reported to show the feasibility and efficiency of our algorithm in Section 4 and some concluding remarks are reported in the last section.

2. Convex Relaxation of LMP

As is known to all, constructing a well-performed relaxation problem can bring great convenience for designing branch-and-bound algorithm of global optimization problems. In this section, we will show how to construct the convex relaxation programming problem (CRP) for LMP. For this, we first compute the initial variable bounds by solving the following linear programming problems:

$$\begin{aligned} x_i^l &= \min_{x\in D} x_i, \\ x_i^u &= \max_{x\in D} x_i, \\ & i = 1, 2, \ldots, n; \end{aligned} \tag{2}$$

then an initial hyperrectangle $X^0 = \{x \in R^n \mid x_i^l \le x_i \le x_i^u,\ i = 1, 2, \ldots, n\}$ can be obtained.

For convenience for introduction of relaxation convex programming problem for LMP over subrectangle $X \subset X^0$, we further solve the following set of linear programming problems:

$$\begin{aligned} l_{ij} &= \min_{x\in D\bigcap X} \phi_{ij}(x), \\ u_{ij} &= \max_{x\in D\bigcap X} \phi_{ij}(x), \\ L_{ij} &= \min_{x\in D\bigcap X} \psi_{ij}(x), \\ U_{ij} &= \max_{x\in D\bigcap X} \psi_{ij}(x). \end{aligned} \tag{3}$$

Based on the construction process of l_{ij} and L_{ij} and u_{ij} and U_{ij}, it is not hard to see that

$$\begin{aligned} &\left(\phi_{ij}(x) - u_{ij}\right)\left(\psi_{ij}(x) - U_{ij}\right) \ge 0, \\ &\left(\phi_{ij}(x) - l_{ij}\right)\left(\psi_{ij}(x) - L_{ij}\right) \ge 0, \end{aligned} \tag{4}$$

$$\begin{aligned} &\left(\phi_{ij}(x) - l_{ij}\right)\left(\psi_{ij}(x) - U_{ij}\right) \le 0, \\ &\left(\phi_{ij}(x) - u_{ij}\right)\left(\psi_{ij}(x) - L_{ij}\right) \le 0; \end{aligned} \tag{5}$$

by combining (4) with (5) and performing some equivalent transformation, we can easily obtain a lower and an upper bound of each bilinear term; that is,

$$\begin{aligned} \phi_{ij}(x)\psi_{ij}(x) &\le \min\left\{u_{ij}\psi_{ij}(x) + L_{ij}\phi_{ij}(x)\right. \\ &\quad \left. - u_{ij}L_{ij}, l_{ij}\psi_{ij}(x) + U_{ij}\phi_{ij}(x) - U_{ij}l_{ij}\right\}, \\ \phi_{ij}(x)\psi_{ij}(x) &\ge \max\left\{u_{ij}\psi_{ij}(x) + U_{ij}\phi_{ij}(x)\right. \\ &\quad \left. - U_{ij}u_{ij}, l_{ij}\psi_{ij}(x) + L_{ij}\phi_{ij}(x) - L_{ij}l_{ij}\right\}. \end{aligned} \tag{6}$$

To facilitate the narrative, $\forall i = 0, 1, 2, \ldots, N$, by denoting

$$\begin{aligned} \underline{g}_{ij}^1(x) &\triangleq u_{ij}\psi_{ij}(x) + U_{ij}\phi_{ij}(x) - U_{ij}u_{ij}, \\ \underline{g}_{ij}^2(x) &\triangleq l_{ij}\psi_{ij}(x) + L_{ij}\phi_{ij}(x) - L_{ij}l_{ij}, \\ \overline{g}_{ij}^1(x) &\triangleq u_{ij}\psi_{ij}(x) + L_{ij}\phi_{ij}(x) - u_{ij}L_{ij}, \\ \overline{g}_{ij}^2(x) &\triangleq l_{ij}\psi_{ij}(x) + U_{ij}\phi_{ij}(x) - U_{ij}l_{ij}, \end{aligned} \tag{7}$$

we can reformulate conclusion (6) as

$$\begin{aligned}\phi_{ij}(x)\,\psi_{ij}(x) &\le \min\left\{\overline{g}_{ij}^{1}(x), \overline{g}_{ij}^{2}(x)\right\} \triangleq \overline{g}_{ij}(x),\\ \phi_{ij}(x)\,\psi_{ij}(x) &\ge \max\left\{\underline{g}_{ij}^{1}(x), \underline{g}_{ij}^{2}(x)\right\} \triangleq \underline{g}_{ij}(x),\end{aligned} \tag{8}$$

respectively. With this, we obtain a lower bound function $\underline{g}_i(x)$ and upper bound function $\overline{g}_i(x)$ for $f_i(x)$, which satisfy $\underline{g}_i(x) \le f_i(x) \le \overline{g}_i(x)$, $i = 0, 1, 2, \ldots, N$, where

$$\begin{aligned}\underline{g}_i(x) &= \sum_{j=1}^{p_i} \underline{g}_{ij}(x),\\ \overline{g}_i(x) &= \sum_{j=1}^{p_i} \overline{g}_{ij}(x),\\ &\qquad i = 0, 1, 2, \ldots, N.\end{aligned} \tag{9}$$

So far, based on the above discussion, it is not too hard to obtain the relaxation programming problem over X for the LMP which we formulated as follows:

(CRP):

$$\begin{cases}\min & \underline{g}_0(x) = \sum_{j=1}^{p_0} \underline{g}_{0j}(x)\\ \text{s.t.} & \underline{g}_i(x) = \sum_{j=1}^{p_i} \underline{g}_{ij}(x) \le 0,\ i = 1, 2, \ldots, M,\\ & \overline{g}_i(x) = \sum_{j=1}^{p_i} \overline{g}_{ij}(x) \ge 0,\ i = M+1, M+2, \ldots, N,\\ & x \in D \bigcap X = \{x \in X \mid Ax \le b,\ x \ge 0\}.\end{cases} \tag{10}$$

Remark 1. As one can easily confirm, program (CRP) is a convex program which can be effectively solved by some convex optimization tool-boxes such as CVX; that is, all functions that appeared in the objective and constraints of CRP are convex.

Remark 2. Both the lower and upper bound functions $\underline{g}_i(x)$ and $\overline{g}_i(x)$ will approximate function $f_i(x)$, as the diameter of rectangle converges to zero.

3. Branch-and-Bound Algorithm and Its Convergence

As we have known, branch-and-bound algorithms utilize tree search strategies to implicitly enumerate all possible solutions to a given problem, applying pruning techniques to eliminate regions of the search space that cannot yield a better solution. There are three algorithmic components in branch-and-bound scheme which can be specified to fine-tune the performance of the algorithm. These components are the search strategy, the branching strategy, and the pruning rules. This section presents a description of these components and the proposed algorithm for solving LMP.

3.1. Key Operation. There are two important phases of any branch-and-bound algorithm: search phase and verification phase.

The choice of search strategy primarily impacts the search phase and has potentially significant consequences for the amount of computational time required and memory used. In this paper, we will choose the depth-first search (DFS) strategy with node ranking techniques, which can reduce a lot of storage space.

The choice of branching strategy determines how children nodes are generated from a subproblem. It has significant impacts on both the search phase and the verification phase. By branching appropriately at subproblems, the strategy can guide the algorithm towards optimal solutions. In this paper, to develop the proposed algorithm for LMP, we adopt a standard range bisection approach, which is adequate to insure global convergence of the proposed algorithm. Detailed process is described as follows.

For any region $X = [x^l, x^u] \subset X^0$, let $r \in \operatorname{argmax}\{x_i^u - x_i^l \mid i = 1, 2, \ldots, n\}$ and $x_{\text{mid}} = (x_r^l + x_r^u)/2$, and then the current region X can be divided into the two following subregions:

$$\begin{aligned}\overline{X} &= \left\{x \in R^n \mid x_i^l \le x_i \le x_i^u,\ i \ne r,\ x_r^l \le x_r \le x_{\text{mid}}\right\},\\ \overline{\overline{X}} &= \left\{x \in R^n \mid x_i^l \le x_i \le x_i^u,\ i \ne r,\ x_{\text{mid}} \le x_r \le x_r^u\right\}.\end{aligned} \tag{11}$$

Another critical aspect of branch-and-bound algorithm is the choice of pruning rules used to exclude regions of the search space from exploration. The most common way to prune is to produce a lower and (or) upper bound on the objective function value at each subproblem and use this to prune subproblems whose bound is no better than the incumbents solution value. For each partition subset X generated by the above branching operation, the bounding operation is mainly concentrated on estimating a lower bound $\text{LB}(X)$ and an upper bound $\text{UB}(X)$ for the optimal value of linear multiplicative programming problem (LMP). $\text{LB}(X)$ can be obtained by solving the following convex relaxation programming problems:

$\text{CRP}(X)$:

$$\begin{cases}\min & \underline{g}_0(x) = \sum_{j=1}^{p_0} \underline{g}_{0j}(x)\\ \text{s.t.} & \underline{g}_i(x) = \sum_{j=1}^{p_i} \underline{g}_{ij}(x) \le 0,\ i = 1, 2, \ldots, M,\\ & \overline{g}_i(x) = \sum_{j=1}^{p_i} \overline{g}_{ij}(x) \ge 0,\ i = M+1, M+2, \ldots, N,\\ & x \in D \bigcap X = \{x \in X \mid Ax \le b,\ x \ge 0\}.\end{cases} \tag{12}$$

Moreover, since any feasible solution of linear multiplicative programming problem (LMP) will provide a valid upper bound of the optimal value, we can evaluate the initial objective value of the optimal solution of CRP to determine an upper bound (if possible) for the optimal value of linear multiplicative programming problem (LMP) over X.

3.2. Branch-and-Bound Algorithm. Based on the former discussion, the presented algorithm for globally solving the LMP can be summarized as follows.

Step 0 (initialization).

Step 0.1. Set iteration counter $k := 0$ and the initial partition set as $\Omega_0 = X^0$. The set of active nodes $\widetilde{X}^0 = \{X^0\}$; the upper bound UB $= \infty$; the set of feasible points $F = \phi$; the convergence tolerance $\epsilon = 1 \times 10^{-8}$.

Step 0.2. Solve the initial convex relaxation problem (CRP) over region X^0; if the CRP is not feasible, then there is no feasible solution for the initial problem. Otherwise, denote the optimal value and solution as f^0_{opt} and x^0_{opt}, respectively. If x^0_{opt} is feasible to the LMP, we can obtain an initial upper bound and lower bound of the optimal value for linear multiplicative programming problem (LMP); that is, UB $:= f_0(x^0_{\text{opt}})$, and LB $:= f^0_{\text{opt}}$. And then, if UB $-$ LB $< \epsilon$, the algorithm can stop, and x^0_{opt} is the optimal solution of the LMP; otherwise proceed to Step 1.

Step 1 (branching). Partition X^k into two new subrectangles according to the partition rule described in Section 3.1. Delete X^k and add the new nodes into the active nodes set $\widetilde{X}^k$; still denote the set of new partitioned sets as $\widetilde{X}^k$.

Step 2 (bounding). For each subregion still of interest $X^{k\mu} \subseteq X^0$, $\mu = 1, 2$, obtain the optimal solution and value for convex relaxation problem (CRP) by solving the convex relaxation programming problem over $X^{k\mu}$; if LB$(X^{k,\mu}) >$ UB, delete $X^{k\mu}$ from $\widetilde{X}^k$. Otherwise, we can update the lower and upper bounds: LB $=$ min{LB$(X^{k,\mu}) \mid \mu = 1, 2\}$ and UB $=$ min{UB, $f(x^{k,\mu}) \mid \mu = 1, 2\}$.

Step 3 (termination). If UB $-$ LB $\le \epsilon$, the algorithm can be stopped; UB is the global optimal value for LMP. Otherwise, set $k := k + 1$, and select the node with the smallest optimal value as the current active node, and return to Step 1.

3.3. Global Convergence of the Algorithm. The global convergence properties of the above algorithm for solving linear multiplicative programming problem (LMP) can be given in the sense of the following theorem.

Theorem 3. *The proposed algorithm will terminate within finite iterations or it will generate an infinite sequence* $\{x^k\}$*, any accumulation point of which is a global optimal solution for the LMP.*

Proof. If the proposed algorithm terminates finitely, assume that it terminates at the k_{th} iteration: $k \ge 0$. By the termination criteria, we know that

$$\text{UB} - \text{LB} \le \epsilon. \tag{13}$$

Based on the upper bounding technique described in Step 2, it is implied that

$$f\left(x^k\right) - \text{LB} \le \epsilon. \tag{14}$$

Let v_{opt} be the optimal value of linear multiplicative programming problem (LMP); then, by Section 3.1 and Step 2 above, we know that

$$\text{UB} = f\left(x^k\right) \ge v_{\text{opt}} \ge \text{LB}. \tag{15}$$

Hence, taking (14) and (15) together, it is implied that

$$v_{\text{opt}} + \epsilon \ge \text{LB} + \epsilon \ge f\left(x^k\right) \ge v_{\text{opt}}, \tag{16}$$

and thus the proof of the first part is completed.

If the algorithm is infinite, then it generates an infinite feasible solution sequence $\{x^k\}$ for the LMP via solving the CRP. Since the feasible region of LMP is compact, the sequence $\{x^k\}$ must have a convergent subsequence. For definiteness and without loss of generality, assume $\lim_{k\to\infty} x^k = x^*$, and then we have

$$\begin{aligned} \lim_{k\to\infty} \phi_{ij}\left(x^k\right) &= \phi_{ij}\left(x^*\right), \\ \lim_{k\to\infty} \psi_{ij}\left(x^k\right) &= \psi_{ij}\left(x^*\right). \end{aligned} \tag{17}$$

By the definitions of l_{ij} and L_{ij} and u_{ij} and U_{ij}, we know that

$$\begin{aligned} \lim_{k\to\infty} l_{ij}\left(x^k\right) &= \phi_{ij}\left(x^*\right), \\ \lim_{k\to\infty} L_{ij}\left(x^k\right) &= \psi_{ij}\left(x^*\right). \end{aligned} \tag{18}$$

Moreover, we have

$$\begin{aligned} \text{UB}_k &= \lim_{k\to\infty} f\left(x^k\right) = f\left(x^*\right), \\ \text{LB}_k &= \lim_{k\to\infty} \sum_{j=1}^{p_0} \underline{g}_{0j}(x) = f\left(x^*\right). \end{aligned} \tag{19}$$

Therefore, we have $\lim_{k\to\infty}(\text{UB}_k - \text{LB}_k) = 0$. This implies that x^* is a global optimal solution for linear multiplicative programming problem (LMP). □

4. Numerical Experiments

To test the proposed algorithm in efficiency and solution quality, we performed some computational examples on a personal computer containing an Intel Core i5 processor of 2.40 GHz and 4 GB of RAM. The code base is written in Matlab 2014a and all subproblems are solved by using CVX.

We consider some instances of linear multiplicative programming problem (LMP) from some recent literature in Wang and Liang [25], in Jiao [26], in Shen et al. [23], in Shen and Jiao [19, 20], and in Jiao et al. [22] and GLOBALLib at http://www.gamsworld.org/global/globallib.htm. Among them, Examples 1–6 are taken from some recent literature. Examples 7–11 are taken from GLOBALLib, a collection of nonlinear programming models. The last example is a nonlinear nonconvex mathematical programming problem with randomized linear multiplicative objective and constraint functions.

Example 1 (references in Shen et al. [23]).

$$\begin{aligned} \min \quad & -4x_1^2 - 5x_2^2 + x_1x_2 + 2x_1 \\ \text{s.t.} \quad & x_1 - x_2 \geq 0, \\ & \frac{1}{3}x_1^2 - \frac{1}{3}x_2^2 \leq 1, \\ & \frac{1}{2}x_1x_2 \leq 1, \\ & 0 \leq x_1 \leq 3, \\ & x_2 \geq 0. \end{aligned} \tag{20}$$

Example 2 (references in Wang and Liang [25] and in Jiao [26]).

$$\begin{aligned} \min \quad & x_1^2 + x_2^2 \\ \text{s.t.} \quad & 0.3x_1x_2 \geq 1, \\ & 2 \leq x_1 \leq 5, \\ & 1 \leq x_2 \leq 3. \end{aligned} \tag{21}$$

Example 3 (references in Jiao [26]).

$$\begin{aligned} \min \quad & x_1^2 + x_2^2 - x_3^2 \\ \text{s.t.} \quad & 0.3x_1x_2 + 0.3x_2x_3 + 0.6x_1x_3 \geq 4, \\ & 2 \leq x_1 \leq 5, \\ & 1 \leq x_2 \leq 3, \\ & 1 \leq x_3 \leq 3. \end{aligned} \tag{22}$$

Example 4 (references in Shen and Jiao [19, 20] and in Jiao et al. [22]).

$$\begin{aligned} \min \quad & x_1x_2 - 2x_1 + x_2 + 1 \\ \text{s.t.} \quad & 8x_2^2 - 6x_1 - 16x_2 \leq -11, \\ & -x_2^2 + 3x_1 + 2x_2 \leq 7, \\ & 1 \leq x_1 \leq 2.5, \\ & 1 \leq x_2 \leq 2.225. \end{aligned} \tag{23}$$

Example 5 (references in Shen and Jiao [19, 20] and in Jiao et al. [22]).

$$\begin{aligned} \min \quad & x_1 \\ \text{s.t.} \quad & \frac{1}{4}x_1 + \frac{1}{2}x_2 - \frac{1}{16}x_1^2 - \frac{1}{16}^2 x_2 \leq 1, \\ & \frac{1}{14}x_1^2 + \frac{1}{14}x_2^2 - \frac{3}{7}x_1 - \frac{3}{7}x_2 \leq -1, \\ & 1 \leq x_1 \leq 5.5, \\ & 1 \leq x_2 \leq 5.5. \end{aligned} \tag{24}$$

Example 6 (references in Jiao et al. [22]).

$$\begin{aligned} \min \quad & x_1 + (2x_1 - 3x_2 + 13)(x_1 + x_2 - 1) \\ \text{s.t.} \quad & -x_1 + 2x_2 \leq 8, \\ & -x_2 \leq -3, \\ & x_1 + 2x_2 \leq 12, \\ & x_1 - 2x_2 \leq -5, \\ & x_1 \geq 0, \\ & x_2 \geq 0. \end{aligned} \tag{25}$$

Examples 1–6 are taken from some literature, where they are solved by branch-and-bound algorithm with linear relaxation techniques. Numerical experiment demonstrates that the above method is more efficient than other methods in the sense that our algorithm requires rather less iterations and CPU time for solving the same problems. Specific results of numerical Examples 1–6 are listed in Table 1, where the notations used in the headline have the following meanings: Exam.: serial number of numerical examples in this paper; Ref.: serial number of numerical examples in the references; Iter.: iterative times; Time: CPU time in seconds; Prec.: precision used in the algorithm; Opt. val. and Opt. sol. denote the optimal value and solution of the problem, respectively.

Example 7 (st-qpk1).

$$\begin{aligned} \min \quad & 2x_1 - 2x_1^2 + 2x_1x_2 + 3x_2 - 2x_2^2 \\ \text{s.t.} \quad & -x_1 + x_2 \leq 1, \\ & x_1 - x_2 \leq 1, \\ & -x_1 + 2x_2 \leq 3, \\ & 2x_1 - x_2 \leq 3, \\ & 0 \leq x_1, x_2. \end{aligned} \tag{26}$$

Example 8 (st-z).

$$\begin{aligned} \min \quad & -x_1^2 - x_2^2 - x_3^2 + 2x_3 \\ \text{s.t.} \quad & x_1 + x_2 - x_3 \leq 0, \\ & -x_1 + x_2 - x_3 \leq 0, \\ & 12x_1 + 5x_2 + 12x_3 \leq 22.8, \\ & 12x_1 + 12x_2 + 7x_3 \leq 17.1, \\ & -6x_1 + x_2 + x_3 \leq 1.9, \\ & x_2 \geq 0. \end{aligned} \tag{27}$$

Example 9 (ex5-4-2).

$$\begin{aligned} \min \quad & x_1 + x_2 + x_3 \\ \text{s.t.} \quad & x_4 + x_6 \leq 400, \\ & -x_4 + x_5 + x_7 \leq 300, \end{aligned}$$

Table 1: Results of the numerical contrast experiments 1–6.

Exam.	Ref.	Opt. val.	Opt. sol.	Iter.	Time (s)	Prec.
1	in Shen et al. [23]	−15.000	(2.0, 1)	1657	120.58	10^{-6}
	Ours	−15.0000	(2.0000, 1.0000)	31	0.34449	10^{-8}
2	In Wang and Liang [25]	6.7780	(2.00003, 1.66665)	44	0.18	10^{-4}
	In Jiao [26]	6.77778	(2.0, 1.666667)	58	<1	10^{-8}
	Ours	6.77778	(2.0000, 1.6667)	1	0.0137	10^{-8}
3	In Jiao [26]	−4.0	(2.0, 1.0, 3.0)	43	-	10^{-8}
	Ours	−4.0	(2.0000, 1.0000, 3.0000)	1	0.0214	10^{-8}
4	In Shen and Jiao [19, 20]	0.0000	(2.00, 1.00)	24	-	10^{-3}
	In Jiao et al. [22]	0.00000003	(2.0000061, 1.0)	16	0.018	10^{-8}
	Ours	0.0000	(2.0000, 1.0000)	1	0.0351	10^{-8}
5	In Shen and Jiao [19, 20]	1.1771	(1.17709, 2.1772)	434	1	10^{-3}
	In Jiao et al. [22]	1.17708	(1.17709, 2.1772)	189	0.226	10^{-6}
	Ours	1.1770	(1.177088, 2.17718)	3	0.1534	10^{-8}
6	In Jiao et al. [22]	3.0000	(0.0000, 4.0000)	25	0.750	10^{-8}
	Ours	3.0000	(0.0000, 4.0000)	1	0.01436	10^{-8}

$$
\begin{aligned}
&-x_5 + x_8 \le 100,\\
&x_1 - x_1x_6 + 833.333333333333x_4\\
&\le 83333.3333333333,\\
&x_2x_4 - x_2x_7 - 1250x_4 + 1250x_5 \le 0,\\
&x_3x_5 - x_3x_8 - 2500x_5 \le -1250000\\
&100 \le x_1 \le 10000,\\
&1000 \le x_2 \le 10000,\\
&1000 \le x_3 \le 10000,\\
&10 \le x_4 \le 1000,\\
&10 \le x_5 \le 1000,\\
&10 \le x_6 \le 1000,\\
&10 \le x_7 \le 1000,\\
&10 \le x_8 \le 1000.
\end{aligned}
\tag{28}
$$

Example 10 (st-qpc-m1).

$$
\begin{aligned}
\min\quad & 10x_1 - 0.34x_1x_1 - 0.28x_1x_2 + 10x_2\\
& - 0.22x_1x_3 + 10x_3\\
& -0.24x_1x_4 + 10x_4 - 0.51x_1x_5 + 10x_5\\
& - 0.28x_2x_1 - 0.34x_2x_2 - 0.23x_2x_3\\
& -0.24x_2x_4 - 0.45x_2x_5 - 0.22x_3x_1 - 0.23x_3x_2\\
& - 0.35x_3x_3 - 0.22x_3x_4\\
& -0.34x_3x_5 - 0.24x_4x_1 - 0.24x_4x_2 - 0.22x_4x_3\\
& - 0.2x_4x_4 - 0.38x_4x_5\\
& -0.51x_5x_1 - 0.45x_5x_2 - 0.34x_5x_3 - 0.38x_5x_4\\
& - 0.99x_5x_5\\
\text{s.t.}\quad & x_1 + x_2 + 2x_3 + x_4 + x_5 \ge 10,\\
& 2x_1 + 3x_2 + x_5 \ge 8,\\
& x_2 + 4x_3 - x_4 + 2x_5 \ge 12,\\
& 8x_1 - x_2 - x_3 + 6x_4 \ge 20,\\
& 2x_1 + x_2 + 3x_3 + x_4 + x_5 \le 30,\\
& x_1, x_2, x_3, x_4, x_5 \ge 0.
\end{aligned}
\tag{29}
$$

Example 11 (st-e26).

$$
\begin{aligned}
\min\quad & -3x_1^2 - 5x_1 - 3x_2^2 - 5x_2\\
\text{s.t.}\quad & 0.7x_1 + x_2 \le 6.3,\\
& 0.5x_1 + 0.8333x_2 \le 6,\\
& x_1 + 0.6x_2 \le 7.08,\\
& 0.1x_1 + 0.25x_2 \le 1.35,\\
& 0 \le x_1 \le 10,\\
& 0 \le x_2 \le 30.
\end{aligned}
\tag{30}
$$

These five examples are taken from GLOBALLib; all of them are generalized linear multiplicative programs with different types of constraints. Computational results and some known results are listed in Table 2, where the notations used in the headline have the following meanings: Exam. denotes the serial number of the example tested in this paper; Best sol. and Best val. represent the best optimal solution and optimal value currently known; Our sol. and Our val. are the optimal solution and optimal value obtained by our algorithm described in this paper. From the results summarised in the table, we can see that our algorithm can effectively solve the LMP.

Table 2: Results of numerical experiments 7–11.

Exam.	Best sol.	Our sol.	Best val.	Our val.
7	Unknown	$(1, 0)$	Unknown	0
8	Unknown	$(0.9, 0, 0.9)$	Unknown	−1
9	x_{glob}	x_{our}	7512.2301449	7512.2301446
10	Unknown	$(0, 0, 0, 3.333, 26.6667)$	Unknown	−473.778
11	Unknown	$(7.08, 0)$	Unknown	−185.779196

Table 3: Numerical results of Example 12.

p	m	n	Avr. iter.	Std. dev.	Avr. time (s)
5	10	10	5.7	28.2	0.5157
5	10	20	11.4	13.4	1.315
5	10	30	19.0	14.7	5.528
10	10	20	31.4	18.4	41.2140
10	20	40	35.7	17.7	43.541
20	30	40	49.6	19.5	75.208
20	30	50	56.1	22.3	88.233
30	50	80	117.0	14.8	168.213
50	80	100	327.0	24.5	216.209
50	100	150	530.0	58.5	436.139

One has

$$\begin{aligned} x_{\text{glob}} &= (1026.9484, 1000, 5485, 265.0597, 280.58873, 134.9403, 284.47098, 380.5887) \\ x_{\text{our}} &= (1026.9484, 1000, 5485.282, 265.06, 280.589, 134.94, 284.471, 380.589)\,. \end{aligned} \tag{31}$$

Example 12 (random test).

$$\begin{aligned} \min \quad & f(x) = \sum_{i=1}^{p} \left(\left(a^{0i}\right)^T x + b_{0i}\right)\left(\left(c^{0i}\right)^T x + d_{0i}\right) \\ \text{s.t.} \quad & \sum_{i=1}^{p} \left(\left(a^{1i}\right)^T x + b_{1i}\right)\left(\left(c^{1i}\right)^T x + d_{1i}\right) \le 0, \\ & \sum_{i=1}^{p} \left(\left(a^{2i}\right)^T x + b_{2i}\right)\left(\left(c^{2i}\right)^T x + d_{2i}\right) \ge 0, \\ & x \in D = \left\{x \in R^n \mid Ax \le b\right\}, \end{aligned} \tag{32}$$

where the real numbers b_{0j} and d_{0j} are randomly generated in the range $[-1, 1]$, b_{1j} is randomly generated in interval $[-1, 0]$, d_{1j}, b_{2j}, d_{2j} are randomly generated in the range $[0, 1]$, and the real elements of a_{ij}, c_{ij}, A, and b are randomly generated in the range $[0, 1]$.

For this problem, we tested 8 groups of instances with different dimension. For each group, we performed 10 instances for a total of 80 instances. The computational results are listed in Table 3, where the notations used in the headline have the following meanings: Avr. iter.: average numbers of iterations in the algorithm; Std. dev.: standard deviation; Avr. time: average CPU time in seconds; m and n denote the numbers of linear constraints and variables, respectively.

5. Concluding Remarks

This paper presents a new relaxation method for designing branch-and-bound algorithm for generalized linear multiplicative programming problem with linear multiplicative constraints. The relaxation problem is a convex programming which can be easily obtained with one-step relaxation; it has better approximation effect than usual two-phase linear relaxation method. The presented algorithm can efficiently work without nonnegative restriction to linear function in multiplicative terms, while this restriction is a necessary condition to most branch and bound algorithms described in a lot of literatures. Extensive results of numerical experiments from recent literature show that our method is feasible and effective for this kind of problems.

Conflicts of Interest

The authors declare that there are no conflicts of interest.

Acknowledgments

This paper is supported by the Science and Technology Key Project of Education Department of Henan Province (14A110024 and 15A110023), the National Natural Science Foundation of Henan Province (152300410097), the Science and Technology Projects of Henan Province (182102310941), the Cultivation Plan of Young Key Teachers in Colleges and Universities of Henan Province (2016GGJS-107), the Higher School Key Scientific Research Projects of Henan Province (18A110019 and 17A110021), and the Major Scientific Research Projects of Henan Institute of Science and Technology (2015ZD07).

References

[1] H. Konno, H. Shirakawa, and H. Yamazaki, "A mean-absolute deviation-skewness portfolio optimization model," *Annals of Operations Research*, vol. 45, no. 1-4, pp. 205–220, 1993.

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

[3] K. P. Bennett and O. L. Mangasarian, "Bilinear separation of two sets in nspace," *Computational optimization and applications*, vol. 2, no. 3, pp. 207–227, 1993.

[4] I. Quesada and I. E. Grossmann, "Alternative bounding approximations for the global optimization of various engineering design problems," in *Global Optimization in Engineering Design, Nonconvex Optimization and Its Applications*, I. E. Grossmann, Ed., vol. 9, Kluwer Academic Publishers, Norwell, MA, 1996.

[5] F. Samadi, A. Mirzazadeh, and M. . Pedram, "Fuzzy pricing, marketing and service planning in a fuzzy inventory model: a geometric programming approach," *Applied Mathematical Modelling: Simulation and Computation for Engineering and Environmental Systems*, vol. 37, no. 10-11, pp. 6683–6694, 2013.

[6] 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.

[7] H. P. Benson, "Vector maximization with two objective functions," *Journal of Optimization Theory and Applications*, vol. 28, no. 2, pp. 253–257, 1979.

[8] R. L. Keeney and H. Raika, *Decisions with Multiple Objective*, Cambridge University Press, Cambridge, 1993.

[9] A. M. Geoffrion, "Solving bicriterion mathematical programs," *Operations Research*, vol. 15, pp. 39–54, 1967.

[10] H. Konno, P. T. Thach, and H. Tuy, *Optimization on Low Rank Nonconvex Structures*, vol. 15 of *Nonconvex Optimization and its Applications*, Kluwer Academic Publishers, Dordrecht, The Netherlands, 1997.

[11] O. L. Mangasarian, "Equilibrium points of bimatrix games," *Journal of the Society for Industrial and Appl ied Mathematics*, vol. 12, pp. 778–780, 1964.

[12] A. M. Frieze, "A bilinear programming formulation of the 3-dimensional assignment problem," *Mathematical Programming*, vol. 7, no. 1, pp. 376–379, 1979.

[13] J. E. Falk, "A linear max-min problem," *Mathematical Programming*, vol. 5, pp. 169–188, 1973.

[14] H. P. Benson, "Global maximization of a generalized concave multiplicative function," *Journal of Optimization Theory and Applications*, vol. 137, no. 1, pp. 105–120, 2008.

[15] H. Tuy, *Convex Analysis and Global Optimization*, Springer International Publishing AG, Switzerland, 2nd edition, 2016.

[16] Y. Dai, J. Shi, and S. Wang, "Conical partition algorithm for maximizing the sum of dc ratios," *Journal of Global Optimization*, vol. 31, no. 2, pp. 253–270, 2005.

[17] 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.

[18] N. V. Thoai, "A global optimization approach for solving the convex multiplicative programming problem," *Journal of Global Optimization*, vol. 1, no. 4, pp. 341–357, 1991.

[19] P. Shen and H. Jiao, "A new rectangle branch-and-pruning approach for generalized geometric programming," *Applied Mathematics and Computation*, vol. 183, no. 2, pp. 1027–1038, 2006.

[20] 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.

[21] C.-F. Wang, S.-Y. Liu, and P.-P. Shen, "Global minimization of a generalized linear multiplicative programming," *Applied Mathematical Modelling: Simulation and Computation for Engineering and Environmental Systems*, vol. 36, no. 6, pp. 2446–2451, 2012.

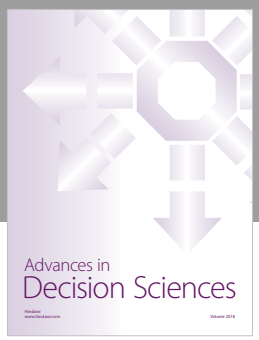
[22] H.-W. Jiao, S.-Y. Liu, and Y.-F. Zhao, "Effective algorithm for solving the generalized linear multiplicative problem with generalized polynomial constraints," *Applied Mathematical Modelling: Simulation and Computation for Engineering and Environmental Systems*, vol. 39, no. 23-24, pp. 7568–7582, 2015.

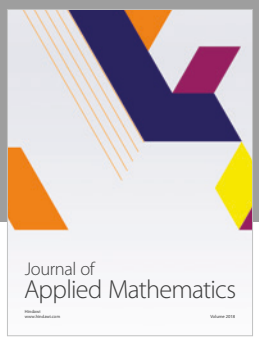
[23] P. Shen, Y. Duan, and Y. Ma, "A robust solution approach for nonconvex quadratic programs with additional multiplicative constraints," *Applied Mathematics and Computation*, vol. 201, no. 1-2, pp. 514–526, 2008.

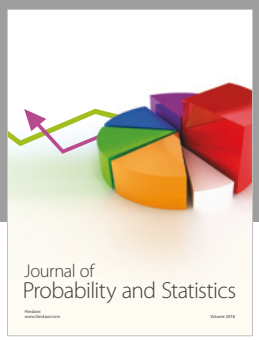
[24] Y. Zhao and S. Liu, "An efficient method for generalized linear multiplicative programming problem with multiplicative constraints," *SpringerPlus*, vol. 5, no. 1, article no. 1302, 2016.

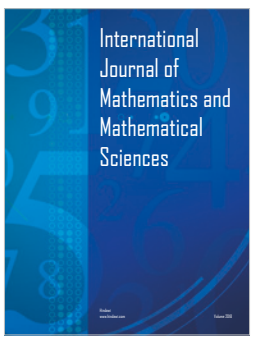
[25] Y. Wang and Z. Liang, "A deterministic global optimization algorithm for generalized geometric programming," *Applied Mathematics and Computation*, vol. 168, no. 1, pp. 722–737, 2005.

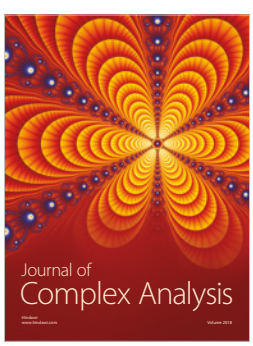
[26] H. Jiao, "A branch and bound algorithm for globally solving a class of nonconvex programming problems," *Nonlinear Analysis. Theory, Methods & Applications. An International Multidisciplinary Journal*, vol. 70, no. 2, pp. 1113–1123, 2009.

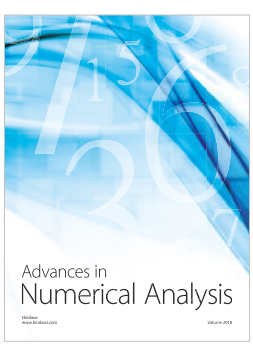


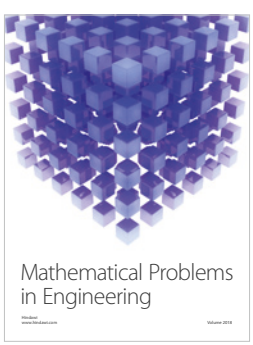


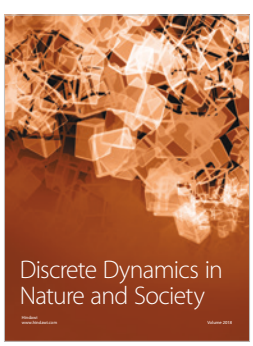


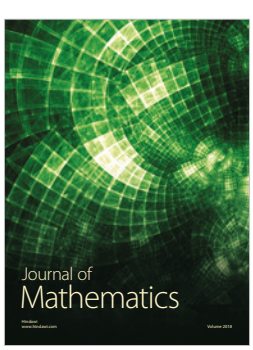


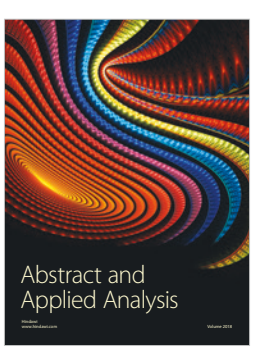


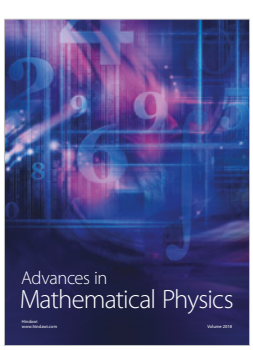












Hindawi

Submit your manuscripts at
www.hindawi.com