Journal of Global Optimization **19:** 345–361, 2001.

A New Extreme Point Algorithm and Its Application in PSQP Algorithms for Solving Mathematical Programs with Linear Complementarity Constraints ⋆

JIANZHONG ZHANG[1] and GUOSHAN LIU[2]
[1]*Department of Mathematics, City University of Hong Kong, Hong Kong, China (Corresponding author, e-mail: mazhang@cityu.edu.hk);* [2]*School of Business Administration, Renmin University of China, Beijing 100872, China*

(Received 14 November 2000; accepted in revised form 27 December 2000)

Abstract. In this paper, we present a new extreme point algorithm to solve a mathematical program with linear complementarity constraints without requiring the upper level objective function of the problem to be concave. Furthermore, we introduce this extreme point algorithm into piecewise sequential quadratic programming (PSQP) algorithms. Numerical experiments show that the new algorithm is efficient in practice.

Key words: Mathematical programs with equilibrium constraints, Linear complementarity, Extreme point

1. Introduction

A mathematical program with equilibrium constraints (MPEC) [5] is a mathematical program in which the constraints include parametric variational inequalities. This nonconvex, nonsmooth and hence very difficult problem has been attracting more and more attention of the operations research community due to its extensive application background and its close relationship with other branches of operations research. It generalizes the bilevel programmingproblems (BLPP) in which the lower level problem is convex (see [14] for a review about BLPP and [11] for some general introductions to BLPP). So, it is also called generalized bilevel programming problem by some researchers [10, 16].

Due to its complicated structure, the analysis of optimality conditions and the development of algorithms for MPEC encounter more difficulties than that for an ordinary nonlinear programming problem. However, some progress has been made and several new algorithms have been proposed in recent years. Among them, implicit programming approaches [1, 3, 12] were proposed to solve an important

⋆ This research is partially supported by City University of Hong Kong under its Strategic Research Grant #7000866 and the National Natural Science Foundation of China grant #19901002.

type of MPEC in which the solution set of the parametric variational inequalities in the constraints is a singleton for any fixed parameter value. The penalty interior-point algorithm [8] and smoothing methods [2, 4, 6] are all globally convergent under some suitable conditions, while piecewise sequential quadratic programming methods (PSQP) [8, 9, 13] are locally superlinearly convergent without requiring the strict complementarity condition. Numerical tests [7] show that these PSQP methods indeed converge very fast. For a MPEC whose data functions are linear (either in linear complementarity constraints or in affine variational inequality constraints), extreme point algorithms [15] are very reasonable and effective. However, existing extreme point algorithms can only deal with the MPEC in which the upper level objective function is concave.

In this paper, we concentrate on mathematical program with linear complementarity constraints (MPLCC). In this case, the feasible solution set of the problem consists of some faces of a polyhedral set. We propose a new extreme point algorithm to solve MPLCC without requiring the upper level objective function of the problem to be concave. At each iteration of the algorithm, we only need to find an extreme direction or an point which is adjacent to a feasible extreme point of MPLCC and satisfies some request, even if the iterative point may not be an extreme point. The extreme point to be found must satisfy that either this point has a smaller objective function value (we call this point a feasible descent extreme point), or the direction from the present iterative point to this extreme point is a feasible descent direction (we call this direction a feasible descent extreme point direction), while the extreme direction to be found must satisfy that from the present iterative point this extreme direction is a feasible descent direction (we call this descent direction a feasible descent extreme direction). In PSQP algorithms, it is needed to find a fesible descent face for a feasible solution of the problem at each iteration. Our extreme point algorithm can also be used for this purpose. As a result, some computing effort can be saved and the efficiency of PSQP algorithms can be enhanced. Moreover, our algorithms do not require any constraint qualification to hold at iterative points. Our numerical experiment shows that the new algorithms are very efficient.

The paper is organized as follows. Some preliminaries are introduced in Section 2. In Section 3, our extreme point algorithm for solving MPLCC is presented and an example is given to illustrate this algorithm. The convergence of this algorithm is analyzed in Section 4 and a subproblem of the algorithm is discussed in Section 5. In Section 6, we introduce the extreme point algorithm proposed in Section 3 into PSQP algorithm for solving MPLCC. Results of some numerical experiments are listed in Section 7. Finally, a few conclusions are given in Section 8.

2. Preliminaries

In this paper, we consider the following mathematical program with linear complementarity constraints (MPLCC) as follows:

$$\begin{aligned} \min_{x,y} \quad & F(x, y) \\ \text{s.t.} \quad & A_1x + B_1y \leqslant b_1, \\ & A_2x + B_2y \leqslant b_2, \\ & y^T(b_2 - A_2x - B_2y) = 0, \\ & y \geqslant 0, \end{aligned}$$

where $F : R^{n+m} \to R$ is convex and continuously differentiable, $b_1 \in R^p$, $b_2 \in R^m$, $A_1 \in R^{p\times n}$, $A_2 \in R^{m\times n}$, $B_1 \in R^{p\times m}$ and $B_2 \in R^{m\times m}$.

Denote the feasible solution set of MPLCC by $\mathcal{F}$. A feasible solution $(\bar{x}, \bar{y})$ of MPLCC is called an optimal solution to MPLCC, if $F(x, y)$ for any $(x, y) \in \mathcal{F}$, $(\bar{x}, \bar{y}) \in \mathcal{F}$ is called a locally optimal solution to MPLCC if there is a neighborhood $N(\bar{x}, \bar{y})$ of $(\bar{x}, \bar{y})$ such that $F(\bar{x}, \bar{y}) \leqslant F(x, y)$ for any $(x, y) \in \mathcal{F} \cap N(\bar{x}, \bar{y})$.

When the complementarity condition is waived, we denote the relaxed feasible region by D, i.e.,

$$D = \{(x, y) | A_1x + B_1y \leqslant b_1, A_2x + B_2y \leqslant b_2, y \geqslant 0\}.$$

In [9], an interesting result about the property of D was given and this result plays an important role in the algorithm proposed in this paper.

LEMMA 2.1 [9]. *The feasible set $\mathcal{F}$ of MPLCC is a union of some faces of D.*

In this paper we use the symbol $E(\cdot)$ to represent the set of extreme points of the set to be concerned. In order to state our algorithm concisely, we assume that the following condition holds throughout this paper.

ASSUMPTION 2.1. The problem MPLCC has an optimal solution and for any face G of D we have $E(G) \neq \emptyset$.

Before ending this section, we recall the definition of two adjacent extreme points in a polyhedral set F. Assume z_1 and z_2 are extreme points of a polyhedral set F. If the convex hull $\text{conv}\{z_1, z_2\}$ of the set $\{z_1, z_2\}$ is a face of F, then we call z_1 and z_2 two adjacent extreme points of F.

3. An Extreme Point Algorithm for MPLCC

As we know, at least one local minimizer of MPLCC is an extreme point of D if the objective function F is concave. Unfortunately, this conclusion is not true when F

is not concave and the locally optimal solutions are often not extreme points of D. In this section, we propose a new extreme point algorithm to solve MPLCC. Our new algorithm utilizes a special property of an extreme point which is related to the iterative point at each iteration.

For every nonempty convex subset G of D, denote by $L(G)$ the set of extreme directions of G with unit length. For any $(\bar{x}, \bar{y}) \in E(D)$, denote by $A(\bar{x}, \bar{y})$ the set of all extreme points which are adjacent to $(\bar{x}, \bar{y})$ in D. We denote the smallest face of D containing $(\bar{x}, \bar{y})$ by $S(\bar{x}, \bar{y})$, and the smallest face of D containing G by $S(G)$.

In each iteration of the following algorithm, the iterative point may not be an extreme point of D. We choose an extreme point $\bar{z}$ of the smallest face $S(x_k, y_k)$ of D, which includes the iterative point (x_k, y_k). Then, we search extreme points adjacent to the point $\bar{z}$ and extreme directions of D in order to find a feasible descent point or a feasible descent direction. If no feasible descent point of feasible descent direction has been found, the algorithm stops and a locally optimal solution has been found, otherwise we can find the next iterative point by solving a subproblem.

ALGORITHM 3.1.

Step 0. Find an initial fesible solution (x_0, y_0) of MPLCC with $(x_0, y_0) \in \arg\min \{F(x, y) | (x, y) \in S(x_0, y_0)\}$ and let $k = 0$.
Step 1. Let $\tilde{z}_k \in E(S(x_k, y_k))$ and set $A_k = \emptyset$.
Step 2. If $A(\tilde{z}_k) \cap \mathcal{F} = A_k$, then go to Step 7.
Step 3. Take a $\bar{z} \in (A(\tilde{z}_k) \cap \mathcal{F}) \setminus A_k$.
Step 4. If $F(\bar{z}) < F(x_k, y_k)$, then set $(x_{k+1}, y_{k+1}) = \bar{z}$, $k = k + 1$ and go to Step 1.
Step 5. If $(\bar{z} - (x_k, y_k))^T \nabla F(x_k, y_k) \geqslant 0$, then set $A_k = A_k \cup \{\bar{z}\}$ and go to Step 2.
Step 6. If $\frac{1}{2}((x_k, y_k) + \bar{z}) \notin \mathcal{F}$, then set $A_k = A_k \cup \{\bar{z}\}$ and go to Step 2; otherwise go to Step 12.
Step 7. Let $L_k = \emptyset$.
Step 8. If $L_k = L(D)$, then stop.
Step 9. Take a $\bar{z} \in L(D) \setminus L_k$.
Step 10. If $\{(x_k, y_k) + \lambda \bar{z} | \lambda \geqslant 0\} \not\subseteq \mathcal{F}$, then let $L_k = L_k \cup \{\bar{z}\}$ and go to Step 8.
Step 11. If $\bar{z}^T \nabla F(x_k, y_k) \geqslant 0$, then let $L_k = L_k \cup \{\bar{z}\}$ and go to Step 8.
Step 12. Choose a feasible face T of D satisfying $\{(x_k, y_k), \bar{z}\} \subseteq T \subseteq \mathcal{F}$ if $\bar{z} \in E(D)$ or $(x_k, y_k) + \bar{z} \in T \subseteq \mathcal{F}$ if $\bar{z} \in L(D)$, and then solve the following subproblem $(MP)_T$

$$\begin{aligned} \min_{x,y} \quad & F(x, y) \\ \text{s.t.} \quad & (x, y) \in T. \end{aligned}$$

Let $(x_{k+1}, y_{k+1}) \in \arg\min \{F(x, y) | (x, y) \in T\}$, $k = k + 1$ and go to Step 1.

As we know, the difficulty to apply the extreme point technique to the problem MPLCC with nonconcave upper level objective function is that the iterative point may not be an extreme point of D. In Algorithm 3.1, we overcome this difficulty by finding an extreme point of the smallest face of D, which includes the iterative point.

In the k-th iteration of the above algorithm, we implicitly enumerate the extreme directions of D and the extreme points adjacent to an extreme point of the smallest face of D, which contains the iterative point. If a feasible descent extreme point or a feasible descent extreme point direction or a feasible descent extreme direction can be found, then the next strictly descent iterative point can be obtained, otherwise we can conclude that this iterative point is a locally optimal solution of MPLCC.

Algorithm 3.1 can be roughly divided into four parts, i.e., Step 0, Steps 1–6, Steps 7–11 and Step 12.

In Step 0, we find a feasible solution of MPLCC. This olution must be a minimizer of the upper level objection function $F(x, y)$ in the smallest face of D, which includes the solution.

In the second part of the algorithm, we search in the set $A(\tilde{z}_k) \cap \mathcal{F}$ in order to find a feasible descent extreme point or a feasible descent extreme point direction, where $\tilde{z} \in S(x_k, y_k)$. First, we find a $\tilde{z}$ in $S(x_k, y_k)$ and let A_k be empty. A_k is used to store the searched points in $A(\tilde{z}_k) \cap \mathcal{F}$. Then, we implicitly enumerate all points in $A(\tilde{z}_k) \cap \mathcal{F}$. For a point $\bar{z}$ in the set $A(\tilde{z}_k) \cap \mathcal{F}$, if $F(\bar{z}) < F(x_k, y_k)$, then we set $\bar{z}$ as the next iterative point; otherwise we will check whether $\bar{z}$ can provide a descent direction. If this point can not provide a descent direction, we will try to search the next point in $A(\tilde{z}_k) \cap \mathcal{F}$; otherwise we will further check if this descent direction is feasible by verifying whether $\frac{1}{2}((x_k, y_k) + \bar{z}) \in \mathcal{F}$. If this direction is not feasible, we again try another point in $A(\tilde{z}_k) \cap \mathcal{F}$; otherwise a feasible descent direction has been found which will be used to find the next iterative point in the fourth part of the algorithm. If we have searched all points in $A(\tilde{z}_k) \cap \mathcal{F}$ and neither the next iterative point nor a feasible descent direction is found, then we will proceed to the next part of the algorithm.

In the third part of the algorithm, we search the extreme directions of D in order to find a feasible descent direction. The process of this part is similar to the second part of the algorithm. If a feasible descent direction is found in this part, then we will use it to produce the next iterative point in the fourth part of the algorithm; otherwise the algorithm stops and a locally optimal solution of MPLCC has been found.

If a feasible descent direction has been found in part two or part three, then we must have another feasible point in this direction from (x_k, y_k), and we simply construct a feasible face T of D which includes this point and (x_k, y_k). We search an optimal solution of F in T to get the next iterative point.

Before discussing the finite convergence of the algorithm, we illustrate this algorithm by a numerical example.

EXAMPLE 3.1. Consider the following quadratic bilevel programming problem:

$$\begin{aligned}\min_{x_1,x_2,y} \quad & \frac{1}{2}(x_1-1)^2+\frac{1}{2}(x_2-\frac{2}{5})^2+\frac{1}{2}(y-\frac{4}{5})^2\\ \text{s.t.} \quad & y\in \arg\min\left\{\frac{1}{2}y^2-y-x_1y+3x_2y \mid 0\leqslant y\leqslant 1\right\},\\ & 0\leqslant x_1\leqslant 1, 0\leqslant x_2\leqslant 1.\end{aligned}$$

We may change this problem into a MPLCC by replacing the lower level programming problem by its KKT optimality conditions.

$$\begin{aligned}\min_{x_1,x_2,y,u_1,u_2} \quad & \frac{1}{2}(x_1-1)^2+\frac{1}{2}(x_2-\frac{2}{5})^2+\frac{1}{2}(y-\frac{4}{5})^2\\ \text{s.t.} \quad & y-x_1+3x_2-u_1+u_2=1,\\ & u_1y=0\\ & u_2(1-y)=0,\\ & 0\leqslant y\leqslant 1,\\ & 0\leqslant x_1\leqslant 1,\\ & 0\leqslant x_2\leqslant 1,\\ & u_1\geqslant 0,\\ & u_1\geqslant 0.\end{aligned}$$

Take $(x_1, x_2, y, u_1, u_2) = (1, 0, 1, 0, 1)$ as the initial iterative point. It is easy to verify that (1,0,1,0,1) is an extreme point of D. In the first iteration, we get a feasible descent extreme point (1,$\frac{1}{3}$,1,0,0) of D. In the second iteration, the extreme point (1,$\frac{2}{3}$,0,0,0) of D is found to get a feasible descent direction. We choose

$$\begin{aligned}T=\{&(x_1, x_2, y, u_1, u_2)\mid y-x_1+3x_2=1, u_1=0, u_2=0,\\ &x_1=1, 0\leqslant x_2\leqslant 1, 0\leqslant y\leqslant 1\}.\end{aligned}$$

T is a face of D satisfying $\{(1,\frac{1}{3},1,0,0), (1,\frac{2}{3},0,0,0)\}\subseteq T\subseteq \mathcal{F}$. Solve the corresponding quadratic programming problem $(QP)_T$:

$$\begin{aligned}\min_{x_1,x_2,y,u_1,u_2} \quad & \frac{1}{2}(x_1-1)^2+\frac{1}{2}(x_2-\frac{2}{5})^2+\frac{1}{2}(y-\frac{4}{5})^2\\ & y-x_1+3x_2=1, u_1=0, u_2=0,\\ & x_1=1, 0\leqslant x_2\leqslant 1, 0\leqslant y\leqslant 1,\end{aligned}$$

we get the next iterative point (1,$\frac{2}{5}$,$\frac{4}{5}$,0,0). In the third iteration, there is no extreme point which is adjacent to an extreme point of the smallest face containing (1,$\frac{2}{5}$,$\frac{4}{5}$,0,0) and provides a feasible descent direction. So, the algorithm terminates at the locally optimal solution (1,$\frac{2}{5}$,$\frac{4}{5}$,0,0) of the MPLCC problem. In this way we obtain a locally optimal solution (1,$\frac{2}{5}$,$\frac{4}{5}$) of Example 3.1. In fact, (1,$\frac{2}{5}$,$\frac{4}{5}$) is a

global minimizer of the quadratic bilevel programming problem discussed in this example.

4. Finite Convergence

First, we prove some required lemmas.

LEMMA 4.1. *Let* $(\bar{x}, \bar{y}) \in \mathcal{F}$. *If* $F(\bar{x}, \bar{y}) \leqslant F(z)$ *for all* z *in every face* T *of* D *satisfying* $(\bar{x}, \bar{y}) \in T \subseteq \mathcal{F}$, *then* $(\bar{x}, \bar{y})$ *is a locally optimal solution of MPLCC.*

Proof. Suppose that $F(\bar{x}, \bar{y}) \leqslant F(z)$ for any face T of D satisfying $(\bar{x}, \bar{y}) \in T \subseteq \mathcal{F}$ and any $z \in T$. If $(\bar{x}, \bar{y})$ were not a locally optimal solution of MPLCC, there would be a sequence $\{(x_i, y_i)\}$ in $\mathcal{F}$ converging to $(\bar{x}, \bar{y})$ and satisfying

$$F(x_i, y_i) < F(\bar{x}, \bar{y}), \quad \text{for } i = 1, 2, \cdots .$$

Because D has only a finite number of faces, there must be a subsequence $\{(x_{i_k}, y_{i_k})\}$ of $\{(x_i, y_i)\}$ and a face $\bar{T}$ of D such that

$$S(x_{i_k}, y_{i_k}) = \bar{T}, \text{ for } k = 1, 2, \cdots .$$

Therefore, $(\bar{x}, \bar{y}) \in \bar{T}$ and $S(\bar{x}, \bar{y}) \subseteq \bar{T}$. From Lemma 2.1, we have $\bar{T} \subseteq \mathcal{F}$. This contradicts the assumption of this lemma. So $(\bar{x}, \bar{y})$ is a locally optimal solution of MPLCC. □

LEMMA 4.2. *Let* $(\bar{x}, \bar{y}) \in \mathcal{F}$ *with* $(\bar{x}, \bar{y}) \in$ *arg min* $\{F(x, y) | (x, y) \in S(\bar{x}, \bar{y})\}$ *and* $\bar{z} \in E(S(\bar{x}, \bar{y}))$. *If* $(z - (\bar{x}, \bar{y}))^T \nabla F(\bar{x}, \bar{y}) \geqslant 0$ *for any* $z \in A(\bar{z}) \cap \mathcal{F}$ *satisfying* $\frac{1}{2}(z + (\bar{x}, \bar{y})) \in \mathcal{F}$, *and* $z^T \nabla F(\bar{x}, \bar{y}) \geqslant 0$ *for any* $z \in L(D)$ *with* $(\bar{x}, \bar{y}) + z \in \mathcal{F}$, *then* $(\bar{x}, \bar{y})$ *is a locally optimal solution of MPLCC.*

Proof. It suffices to verify that the condition of Lemma 4.1 is satisfied. Let T be a face of D satisfying $(\bar{x}, \bar{y}) \in T \subseteq \mathcal{F}$, we will show that

$$F(\bar{x}, \bar{y}) \leqslant f(z)$$

for any $z \in T$. First, assume that $L(D) = \emptyset$. Let z be a point in T. Thus there are a positive integer t, $\lambda_i \geqslant 0$ and $z_i \in A(\bar{z}) \cap T$ for $i = 1, \ldots, t$ such that

$$z = \bar{z} + \sum_{i=1}^{t} \lambda_i (z_i - \bar{z}). \tag{4.1}$$

If $\sum_{i=1}^{t} \lambda_i \leqslant 1$, let $\lambda_0 = 1 - \sum_{i=1}^{t} \lambda_i$ and $z_0 = \bar{z}$. By the assumption of the lemma,

$$\begin{aligned} (z - (\bar{x}, \bar{y}))^T \nabla F(\bar{x}, \bar{y}) &= \sum_{i=0}^{t} \lambda_i (z_i - (\bar{x}, \bar{y}))^T \nabla F(\bar{x}, \bar{y}) \\ &\geqslant 0. \end{aligned}$$

By the convexity of F, we have

$$F(z) \geqslant F(\bar{x}, \bar{y}).$$

Now we suppose $\sum_{i=1}^{t} \lambda_i > 1$. We have

$$(\bar{x}, \bar{y}) = \mu_0 \bar{z} + \sum_{i=1}^{t_1} \mu_i z_i + \sum_{i=t+1}^{t_2} \mu_i z_i, \tag{4.2}$$

where

$$\begin{aligned} \mu_0 &> 0, \\ \mu_i &\geqslant 0, \quad i = 1, 2, \cdots, t_1, \\ z_i &\in E(S(\bar{x}, \bar{y})) \cap A(\bar{z}), \quad i = 1, 2, \cdots, t_1, \\ \mu_i &\geqslant 0, \quad i = t+1, \cdots, t_2, \\ z_1 &\in E(S(\bar{x}, \bar{y})) \setminus A(\bar{z}), \quad i = t+1, \cdots, t_2, \end{aligned}$$

and

$$\sum_{i=0}^{t_1} \mu_i + \sum_{i=t+1}^{t_2} \mu_i = 1.$$

Without loss of generality, we can assume $t_1 \leqslant t$. Let $\mu_i = 0$ for $i = t_1 + 1, \cdots, t$, $\lambda_i = 0$ for $i = t+1, \cdots, t_2$ and $M = \sum_{i=1}^{t} \lambda_i$. By (2), we can replace $\bar{z}$ in (1) by $(\bar{x}, \bar{y})$. Therefore,

$$z = \frac{1}{\mu_0} \sum_{i=1}^{t_2} (\mu_0 \lambda_i + (M-1)\mu_i)(z_i - (\bar{x}, \bar{y})) + (\bar{x}, \bar{y}).$$

Similarly, we can also get

$$F(z) \geqslant F(\bar{x}, \bar{y}).$$

So, in this case, the condition of Lemma 4.1 is satisfied.

We now consider the case that $L(D) \neq \emptyset$. Note that for any $z \in T$, we have

$$z \in \text{conv}(E(d)) + \text{cone}(L(D)).$$

Then we have $\tilde{z}_1 \in \text{conv}(E(D))$ and $\tilde{z}_2 \in \text{cone}(L(D))$ such that $z = \tilde{z}_1 + \tilde{z}_2$. From the above proof, we know that

$$(\tilde{z}_1 - (\bar{x}, \bar{y}))^T \nabla F(\bar{x}, \bar{y}) \geqslant 0.$$

Similarly, we can also prove

$$(\tilde{z}_2 - (\bar{x}, \bar{y}))^T \nabla F(\bar{x}, \bar{y}) \geqslant 0.$$

Hence, we again have

$$F(z) \geqslant F(\bar{x}, \bar{y}).$$

This completes the proof. □

Now, we can state our main result as follows.

THEOREM 4.1. *Algorithm 3.1 finds a locally optimal solution of MPLCC in a finite number of iterations.*

Proof. At each iteration, the algorithm generates a new iterative point which is a globally optimal solution of the objective function $F(x, y)$ restricted in a face of D. Because D has only a finite number of faces and every face of D is checked at most once, the algorithm terminates after a finite number of iterations. By Lemmas 4.1 and 4.2, the final iterative point generated by the algorithm is a locally optimal solution of MPLCC. □

5. A Subproblem

In this section, we will discuss a subproblem of Algorithms 3.1, that is, when a convex subset H of $\mathcal{F}$ is given, how to construct $S(H)$, i.e., the smallest face T of D satisfying $H \subseteq T$?

For convenience, we rewrite D in the following form:

$$D = \{z | Gz \leqslant \bar{b}\},$$

where $z = (x, y)$ and G is a $(p + 2m) \times (n + m)$ matrix. For any $z \in D$, let

$$I(z) = \{i | G_i z = \bar{b}_i, i = 1, \cdots, p + 2m\}$$

and

$$J(z) = \{i | G_i z < \bar{b}_i, i = 1, \cdots, p + 2m\},$$

where G_i and $\bar{b}_i$ are the i-th row of G and $\bar{b}$, respectively.

We have the following theorem.

THEOREM 5.1. *For any convex subset H of D and a $\bar{z} \in \mathrm{ri}(H)$,*

$$S(H) = \{z | G_{I(\bar{z})} z = \bar{b}_{I(\bar{z})}, G_{J(\bar{z})} z \leqslant \bar{b}_{J(\bar{z})}\},$$

where $G_{J(\bar{z})}$ and $\bar{b}_{J(\bar{z})}$ are the submatrices consisting of the rows of G and $\bar{b}$, respectively, with the indexes in $J(\bar{z})$, and so are $G_{I(\bar{z})}$ and $\bar{b}_{I(\bar{z})}$.

Proof. Let

$$\bar{S} = \{z | G_{I(\bar{z})} z = \bar{b}_{I(\bar{z})}, G_{J(\bar{z})} z \leqslant \bar{b}_{J(\bar{z})}\}.$$

First, we will show that $S(H) \subseteq \bar{S}$. Certainly, we have $\bar{z} \in \bar{S}$. It is easy to verify that $\bar{S}$ is a face of D. Therefore, $S(H) \subseteq \bar{S}$. On the other hand, it is not difficult to show that $\bar{z} \in \mathrm{ri}(\bar{S})$. For any face $\tilde{S}$ of D satisfying $H \subseteq \tilde{S}$, we have that $\bar{S} \subseteq \tilde{S}$. Therefore, $\bar{S} = S(H)$. The proof is completed. □

According to the Theorem, when an interior point is known, the smallest face can be explicitly characterized, and then the subproblem can be solved. Generally speaking, it is not a trivial task to obtain an interior point of a convex set. Fortunately, we need to consider only some special convex sets such as points and segments in Algorithm 3.1.

6. A New PSQP Algorithm for Solving MPLCC

In Section 4, we proved that Algorithm 3.1 converges to a locally optimal solution of MPLCC in finite iterations. However, we possibly need to solve a nonlinear mathematical programming problem at each iteration. In this section, using the extreme point technique introduced in Section 3, we give a new PSQP algorithm for solving MPLCC in which we need only to solve a quadratic programming problem in each iteration.

ALGORITHM 6.1.

Step 0. Find an initial feasible solution (x_0, y_0) of MPLCC, give a positive definite matrix $B_0 \in R^{(n+m)\times(n+m)}$ and let $k = 0$.

Step 1. Solve the following linear programming problem:

$$\begin{aligned} \min_{x,y} \quad & (x - x_k, , y - y_k)^T \nabla F(x_k, y_k) \\ \text{s.t.} \quad & (x, y) \in S(x_k, y_k). \end{aligned}$$

Let $\tilde{z}_k = (\tilde{x}_k, \tilde{y}_k)$ be a basic solution of the above linear programming problem.

Step 2. Set $A_k = \emptyset$.

Step 3. If $A(\tilde{z}_k) \cap \mathcal{F} = A_k$, then go to Step 8.

Step 4. Take a $\bar{z} \in (A(\tilde{z}_k) \cap \mathcal{F}) \setminus A_k$.

Step 5. If $F(\bar{z}) < F(x_k, y_k)$, then set $\tilde{z}_{k+1} = (x_{k+1}, y_{k+1}) = \bar{z}$, $k = k + 1$ and go to Step 2.

Step 6. If $(\bar{z} - (x_k, y_k))^T \nabla F(x_k, y_k) \geqslant 0$, then set $A_k = A_k \cup \{\bar{z}\}$ and go to Step 3.

Step 7. If $\frac{1}{2}((x_k, y_k) + \bar{z}) \notin \mathcal{F}$, then set $A_k = A_k \cup \{\bar{z}\}$ and go to Step 3; otherwise go to Step 13.

Step 8. Let $L_k = \emptyset$.

Step 9. If $L_k = L(D)$, then stop.

Step 10. Take a $\bar{z} \in L(D) \setminus L_k$.

Step 11. If $\{(x_k, y_k) + \lambda\bar{z} | \lambda \geqslant 0\} \nsubseteq \mathcal{F}$, then let $L_k = L_k \cup \{\bar{z}\}$ and go to Step 9.

Step 12. If $\bar{z}^T \nabla F(x_k, y_k) \geqslant 0$, then let $L_k = L_k \cup \{\bar{z}\}$ and go to Step 9.

Step 13. Choose a face T of D satisfying $\frac{1}{2}((x_k, y_k) + \bar{z}) \in T \subseteq \mathcal{F}$ if $\bar{z} \in E(D)$ or $(x_k, y_k) + \bar{z} \in T \subseteq \mathcal{F}$ if $\bar{z} \in L(D)$, then solve the following quadratic programming problem $(QP)_T$

$$\begin{aligned} \min_{x,y} \quad & (x - x_k, y - y_k)^T \nabla F(x_k, y_k) + \frac{1}{2}(x - x_k, y - y_k)^T B_k (x - x_k, y - y_k) \\ \text{s.t.} \quad & (x, y) \in T. \end{aligned}$$

Let z_k be a solution of the above quadratic programming problem and set

$$(x_{k+1}, y_{k+1}) = (x_k, y_k) + z_k.$$

Adjust B_{k+1}, let $k = k + 1$ and go to Step 1.

In the above algorithm, we can adjust B_k according to the rules given in existing sequential quadratic programming algorithms for solving standard nonlinear programming problems. We have the following results.

THEOREM 6.1. *If Algorithm 6.1 stops in a finite number of iterations, then it must stop at a locally optimal point of MPLCC.*

The main idea of Algorithm 6.1 is the same as that of PSQP algorithms ([8, 9, 13]), i.e., they all solve a linearized subproblem on an appropriate piece of the original problem. Hence Algorithm 6.1 also possesses the advantage of the existing PSQP algorithms such as superlinear convergence under some conditions. The only difference of Algorithm 6.1 from the existing PSQP algorithms is that we give a new method for choosing the piece which is required in the algorithm and we do not need to search all pieces. However, just like other PSQP algorithms for solving mathematical programming problem with equilibrium constraints, Algorithm 6.1 may not be globally convergent, while Algorithm 3.1 is globally and finitely terminated. By introducing the extreme point technique into PSQP algorithms, the efficiency of the PSQP algorithms can be enhanced.

7. Numerical Tests

In order to investigate the efficiency of the algorithms proposed in this paper, in a PC with celeron 300A CPU, 64M ram and 6.4G hard disk, we have implemented the algorithms given in this paper by using MATLAB and tested some randonly generated QPLCC, i.e., quadratic programs with linear complementarity

constraints, which can be written as the following form:

$$
\begin{aligned}
(QPLCC)\ \min_{x,y} \quad & F(x,y) = \frac{1}{2}\begin{bmatrix} x \\ y \end{bmatrix}^T C \begin{bmatrix} x \\ y \end{bmatrix} + c^T \begin{bmatrix} x \\ y \end{bmatrix}, \\
\text{s.t.} \quad & A_1 x + B_1 y = b_1, \\
& A_2 x + B_2 y + z = b_2, \\
& y^T z = 0, \\
& x \geqslant 0,\ y \geqslant 0,\ z \geqslant 0,
\end{aligned}
$$

where $c \in R^{n+m}$, $C \in R^{(n+m)\times(n+m)}$, $b_1 \in R^p$, $b_2 \in R^m$, $A_1 \in R^{p\times n}$, $A_2 \in R^{m\times n}$, $B_1 \in R^{p\times m}$ and $B_2 \in R^{m\times m}$.

In fact, there is a well designed program [7] which can randomly produce QPLCC type of test problems. When we deal with QPLCC, Algorithm, 3.1 and Algorithm 6.1 reduce to the same form and both are globally convergent and stop in a finite number of iterations.

The program for solving QPLCC consists of the following four subprograms:

(i) A subprogram for generating test problems. First, we randonly generate x_0, y_0, z_0, c, C, A_1, A_2, B_1 and B_2 satisfying that $x_0 \geqslant 0$, $y_0 \geqslant 0$, $z_0 \geqslant 0$ with $y_0^T z_0 = 0$, and C is positive definite. Then, set $b_1 = A_1 x_0 + B_1 y_0$ and $b_2 = A_2 x_0 + B_2 y_0 + z_0$. Let $I(x_0) = \{i | (x_0)_i = 0, i = 1, \cdots, n\}$, $I(y_0) = \{i | (y_0)_i = 0, i = 1, \cdots, m\}$, $I(z_0) = \{i | (z_0)_i = 0, i = 1, \cdots, m\}$, and solve the quadratic programming problem:

$$
\begin{aligned}
\min_{x,y,z} \quad & \frac{1}{2}\begin{bmatrix} x \\ y \end{bmatrix}^T C \begin{bmatrix} x \\ y \end{bmatrix} + c^T \begin{bmatrix} x \\ y \end{bmatrix}, \\
\text{s.t.} \quad & A_1 x + B_1 y = b_1, \\
& A_2 x + B_2 y + z = b_2, \\
& x \geqslant 0,\ y \geqslant 0,\ z \geqslant 0, \\
& x_{I(x_0)} = 0,\ y_{I(y_0)} = 0,\ z_{I(z_0)} = 0
\end{aligned}
$$

by the subroutine in MATLAB to get the initial iterative point.

(ii) A subprogram for finding an extreme point. For a given iterative point (x_k, y_k, z_k), solve the following linear programming problem by Simplex method:

$$
\begin{aligned}
\min_{x,y,z} \quad & c^T \begin{bmatrix} x \\ y \end{bmatrix}, \\
\text{s.t.} \quad & A_1 x + B_1 y = b_1, \\
& A_2 x + B_2 y + z = b_2, \\
& x \geqslant 0,\ y \geqslant 0,\ z \geqslant 0, \\
& x_{I(x_k)} = 0,\ y_{I(y_k)} = 0,\ z_{I(z_k)} = 0,
\end{aligned}
$$

and get its basic solution $(\bar{x}_k, \bar{y}_k, \bar{z}_k)$ and the index set $\tilde{B}_k$ for the basic variables.

(iii) A subprogram for finding a feasible descent direction. As we know, $(\bar{x}_k, \bar{y}_k, \bar{z}_k)$ above is also a basic solution of the following linear programming problem:

$$\begin{aligned} \min_{x,y,z} \quad & (c + C\begin{bmatrix} x_k \\ y_k \end{bmatrix})^T \begin{bmatrix} x \\ y \end{bmatrix}, \\ \text{s.t.} \quad & A_1 x + B_1 y = b_1, \\ & A_2 x + B_2 y + z = b_2, \\ & x \geqslant 0, y \geqslant 0, z \geqslant 0. \end{aligned}$$

With Simplex method, for each feasible basic solution adjacent to $(\bar{x}_k, \bar{y}_k, \bar{z}_k)$ of the above problem, verify that if the basic solution can provide a feasible descent direction for the problem QPLCC, and for each extreme direction, verify that if this extreme direction provides a feasible descent direction. If no feasible descent direction for the problem QPLCC can be found, then the algorithm terminates and a locally optimal solution of QPLCC is found. Otherwise a new feasible solution $(\tilde{x}_k, \tilde{y}_k, \tilde{z}_k)$ of QPLCC has been found, which provides a descent direction at (x_k, y_k, z_k).

(iv) The generation of the next iterative point. For given $(\tilde{x}_k, \tilde{y}_k, \tilde{z}_k)$, solve the following quadratic programming problem

$$\begin{aligned} \min_{x,y,z} \quad & \frac{1}{2}\begin{bmatrix} x \\ y \end{bmatrix}^T C \begin{bmatrix} x \\ y \end{bmatrix} + c^T \begin{bmatrix} x \\ y \end{bmatrix}, \\ \text{s.t.} \quad & A_1 x + B_1 y = b_1, \\ & A_2 x + B_2 y + z = b_2, \\ & x \geqslant 0, y \geqslant 0, z \geqslant 0, \\ & x_{I(\tilde{x}_k)} = 0, y_{I(\tilde{y}_k)} = 0, z_{I(\tilde{z}_k)} = 0, \end{aligned}$$

and let its solution be the next iterative point.

In the above, we stated the main idea of our code for solving QPLCC. Now, we list the result of our numerical tests (Table 1).

In the numerical experiment, problems are randomly produced in the computer. From the numerical results, we know that the time spent on finding descent faces is between 10–20% of the total running time. In fact, at each iteration of our PSQP algorithm, the computation effort is mainly spent on solving three subproblems, one of them is to solve a linear program, the second is to find a descent face or to conclude that a locally optimal solution is found. The third is to solve a quadratic program. Hence, the main difficulty to solve a QPLCC is to solve the quadratic program that occurs in the algorithm.

In order to observe the impact of degeneracy of the linear complementarity constraint in QPLCC to the efficiency of the algorithm, we have made another experiment by giving an initial feasible solution of QPLCC with a fixed degree of degeneracy. Then we try to find a new feasible descent direction or conclude that this solution is a locally optimal solution. We choose $m = 40$, $n = 60$, $p = 30$ and

Table 1. Numerical results for the first experiment

		min	max	mean	std
n=10	time I	0.1300	0.5500	0.2054	0.0755
m=15	time II	0.0200	0.1000	0.0451	0.0148
p=8	ratio	0.1111	0.3478	0.2250	0.0480
r=100	iterations	1.0000	4.0000	1.1700	0.4935
		min	max	mean	std
n=20	time I	0.4100	4.8400	1.0431	0.8727
m=30	time II	0.0600	0.7500	0.1926	0.1450
p=15	ratio	0.1258	0.3455	0.1957	0.0326
r=100	iterations	1.0000	9.0000	2.2200	1.7383
		min	max	mean	std
n=40	time I	2.4100	124.02	13.024	13.915
m=60	time II	0.3500	15.360	1.6661	1.7084
p=30	ratio	0.1000	0.1935	0.1347	0.0188
r=100	iterations	1.0000	40.000	6.1000	5.6309
		min	max	mean	std
n=60	time I	6.9400	510.11	108.35	94.580
m=80	time II	0.7400	58.810	11.791	10.727
p=40	ratio	0.0895	0.1257	0.1080	0.0083
r=100	iterations	1.0000	75.000	18.860	14.536
		min	max	mean	std
n=80	time I	26.220	917.07	353.49	234.12
m=100	time II	3.6200	135.90	40.679	29.301
p=60	ratio	0.0994	0.1482	0.1131	0.0098
r=50	iterations	1.0000	52.000	20.020	13.461
		min	max	mean	std
n=100	time I	104.06	936.92	480.70	360.16
m=120	time II	11.150	106.99	54.595	41.137
p=80	ratio	0.1071	0.1190	0.1126	0.0042
r=50	iterations	5.0000	48.000	21.833	41.137

time I – the total time needed to solve a QPLCC (unit:second);
time II – the total time needed to find descent faces for solving a QPLCC (unit:second);
ratio – (time II)/(time I);
iterations – the number of iterations needed to solve a QPLCC;
r – the number of problems solved in the experiment;
std – standard deviation.

Table 2. Numerical results for the second experiment

		num	min	max	mean	std
degree=0	total	200	1.8400	2.6000	2.1190	0.1477
	optimal	38	2.0800	2.6000	2.2716	0.1161
degree=4	total	200	1.6400	3.6800	1.9089	0.3686
	optimal	2	2.1000	2.2000	2.1500	0.0707
degree=8	total	100	1.4600	3.1900	1.9085	0.5532
	optimal	1	1.8000	1.8000	1.8000	0
degree=10	total	100	1.3900	1.7400	1.4908	0.0743
	optimal	1	1.7400	1.7400	1.7400	0

consider different degrees of degeneracy. We have the following results about the time needed to verify a locally optimal solution or to find a new descent direction (Table 2).

In Table 2 'num' means the number of problems tested, 'total' means for all problems tested and 'optimal' means for the problems in which the given initial point is a locally optimal solution. The time unit in the table is still second.

As shown in the above table, the degree of degeneracy does not affect too much the time spent on verifying local optimality or finding a new descent direction.

8. Conclusions

In Algorithm 3.1, the basic idea of the extreme point algorithm for linear-quadratic bilevel programming problems such as that discussed in Vicente et al. [15] is extended to deal with MPLCC in which the upper level objective function is not concave. In Algorithm 3.1, in order to produce a feasible descent face of D, we do not need to check every feasible face which includes the iterative point. Instead, we need to check only all feasible extreme directions and the extreme points adjacent to an extreme point which is relevant to the iterative point.

By introducing this extreme point technique to PSQP algorithms, we proposed a new PSQP algorithm, i.e., Algorithm 6.1, which possesses the advantages of both PSQP algorithms for solving MPEC and the extreme point algorithms for solving linear MPEC.

More important, we proposed a method to choose a descent face or a linear piece which is needed in some algorithms for solving MPLCC, and the efficiency of some existing algorithms for solving MPLCC can be improved. This method may also be hopeful to provide an alternative way to find a descent direction for PSQP algorithms in more general cases. The numerical tests given in Section 7 reveal

that, although the descent direction finding subproblem in PSQP algorithms is NP-Hard, we spent only about 15% of the total time in the descent direction finding subproblem when we solved the randomly generated middle scale problems. This also means that PSQP algorithms are very promising for solving MPEC.

9. Acknowledgements

The authors heartily thank Professor S. Dempe and Professor J.Y. Han for their careful reading of this paper and their helpful suggestions which lead to the present form of the paper. We also give our thanks to Dr. H.Y. Jiang and Dr. D.F. Sun for their stimulating discussion with us.

10. Appendix

In the proof of Lemma 4.2, the following fact has been used and we restate it without giving the proof here.

LEMMA. *Assume that D is a polyhedral set and $\bar{z} \in E(D)$, for any $z \in D$, there are positive integers t_1 and t_2 with $t_1 \leqslant t_2$, $\lambda_i \geqslant 0$ for $i = 1, \ldots, t_2$, $z_i \in A(\bar{z}) \cap D$ for $i = 1, \ldots, t_1$ and $z_i \in L(D)$ for i with $t_1 + 1 \leqslant i \leqslant t_2$ such that*

$$z = \bar{z} + \sum_{i=1}^{t_1} \lambda_i (z_i - \bar{z}) + \sum_{i=t_1+1}^{t_2} \lambda_i z_i.$$

References

1. Dempe, S. (1995), On generalized differentiability of optimal solutions and its application to an algorithm for solving bilevel optimization problems, in Du, D.Z., Qi, L. and Womersley, R.S. (eds), *Recent Advances in Nonsmooth Optimization*, World Scientific, Singapore. (pp 36–56).
2. Facchinei, F., Jiang, H.Y. and Qi, L. (1999), A smoothing method for mathematical programs with equilibrium constraints, *Mathematical Programming* 85: 107–134.
3. Falk, J.E. and Liu, J. (1995), On bilevel programming, Part I: General cases, *Mathematical Programming* 70: 47–72.
4. Fukushima, M., Luo, Z.Q. and Pang, J.S. (1998), A globally convergent sequential quadratic programming algorithm for mathematical programs with linear complementarity constraints, *Computational Optimization and Applications* 10: 5–34.
5. Harker, P.T. and Pang, J.S. (1988), On the existence of optimal solution to mathematical program with equilibrium constraints, *Operations Research Letters* 7: 61–64.
6. Jiang, H.Y. and Ralph, D., Smooth SQP methods for mathematical programs with nonlinear complementarity constraints, *SIAM Journal on Optimization* (to appear).
7. Jiang, H.Y. and Ralph, D. (1999), QPECgen, a MATLAB generator for mathematical programs with quadratic objectives and affine variational inequality constraints, *Computational Optimization and Applications* 33: 25–59.
8. Luo, Z.Q., Pang, J.S. and Ralph, D. (1996), *Mathematical Programs with Equilibrium Constraints*, Cambridge University Press, Cambridge, UK.

9. Luo, Z.Q., Pang, J.S. and Ralph, D. (1997) Piecewise sequential quadratic programming for mathematical programs with nonlinear complementarity constraints, in Ferris, M.C. and Pang, J.S. (eds), *Complementarity and Variational Problems: State of the Art*, SIAM, Philadelphia, PA.
10. Marcotte, P. and Zhu, D.L. (1996), Exact and inexact penalty methods for the generalized bilevel programming problems, *Mathematical Programming* 74: 141–157.
11. Migdalas, A., Pardalos, P. and Värbrand, P. (eds) (1997), *Multilevel Optimization: Algorithms and Applications*, Kluwer Academic Publishers, Boston, Ma.
12. Outrata, J. and Zowe, J. (1995), A numerical approach to optimization problems with variational inequality constraints, *Mathematical Programming* 68: 105–130.
13. Ralph, D. (1996), Sequential quadratic programming for mathematical programs with linear complementarity constraints, in May, R.L. and Easton A.K. (eds), CTAC95 *Computational Techniques and Applications*, World Scientific, Singapore.
14. Vicente, L.N. and Calamai, P.H. (1994), Bilevel and multilevel programming: A bibliography review, *Journal of Global Optimization* 3: 291–306.
15. Vicente, L.N., Savard, G. and Judice, J. (1994), Descent approaches for quadratic bilevel programming, *Journal of Optimization Theory and Applications* 81: 379–399.
16. Ye, J., Zhu D.L. and Zhu, Q, (1997), Generalized bilevel programming problems, *SIAM Journal on Optimization* 7: 481–507.