Yokohama Publishers
ISSN 1349-8169 ONLINE JOURNAL
Pacific Journal of Optimization
Vol. 11, No. 4, pp. 705–719, October 2015

A LIMITED-MEMORY PROJECTION METHOD FOR VARIATIONAL INEQUALITY PROBLEMS

Qun Wang, Xingju Cai*† and Deren Han*‡

Abstract: The limited-memory technique is an efficient skill which has been used in many optimization methods. In this paper we present a limited-memory projection method for solving monotone variational inequality problems with simple constrains. By combining the information of the last $(m-1)$ iterations and the kth iteration, we construct a new descent direction and then design a new algorithm. The method can be viewed as a generalization of the conjugate gradient method for nonlinear programs. Under some suitable assumptions, we prove the global convergence of the method. We also report some preliminary numerical results, which illustrate that the method is reliable and efficient in practice.

Key words: *limited-memory, projection method, variational inequality problem, complementarity problem, generalized Nash equilibrium problem*

Mathematics Subject Classification: *90C25, 90C33, 90C30*

---

1 Introduction

Variational inequality (VI) is a classical problem and has many important applications in different fields such as mathematical programming, network economics, transportation research, game theory and regional sciences. VI was initiated as an analytic tool for studying free boundary problems defined by nonlinear partial differential operators arising from unilateral problems in elasticity and plasticity theory and in mechanics, which are infinite-dimensional (see [1, 5, 7]).

The variational inequality problem, denoted VI(F,Ω), is to find a vector x^* such that

$$\langle x-x^*, F(x^*)\rangle \geq 0, \quad \forall x\in\Omega, \tag{1.1}$$

where Ω is a nonempty closed convex subset of $\mathbb{R}^n$, F is a continuous operator from $\mathbb{R}^n$ into $\mathbb{R}^n$. In this paper, we focus on the case that F is a monotone operator on Ω, i.e.,

$$\langle x-y, F(x)-F(y)\rangle \geq 0, \quad \forall x,y\in\Omega.$$

*The last two authors were also supported by a project funded by PAPD of Jiangsu Higher Education Institutions.

†The research of the second author was supported by NSFC (11401315) and Jiangsu Provincial NSFC (BK20140914).

‡The research of the third author was supported by NSFC (11371197; 11431002).

If $F(x) = \nabla\theta(x)$, where $\theta(x) : \mathbb{R}^n \to \mathbb{R}$ is a continuously differentiable convex function, then $\mathrm{VI}(F, \Omega)$ is equivalent to the optimization problem:

$$\min\ \theta(x), \quad \text{s.t. } x \in \Omega. \tag{1.2}$$

In addition to optimization problems, many problems from various fields can be formulated as VIs, such as least absolute deviations problem, shortest path problem, spatial price equilibrium problem, machine learning and so on (see [14, 15, 27, 28]). In these problems, the feasible set Ω usually is a simple closed convex set, such as the nonnegative orthant $\mathbb{R}^n_+ = \{x \in \mathbb{R}^n \mid x_i \geq 0\}$, or a box $\{x \in \mathbb{R}^n \mid l_i \leq x_i \leq h_i\}$ or a ball $\{x \in \mathbb{R}^n \mid \|x\| \leq r\}$. In 1980, Dafermos found that the traffic network equilibrium problems can be transformed to finite-dimensional variational inequality problems [5].

There are some classic algorithms for solving the monotone $\mathrm{VI}(F, \Omega)$, such as projection methods, regularization methods, proximal point methods, operator splitting methods and so on (see [7]). Projection methods are simple methods for solving a monotone $\mathrm{VI}(F, \Omega)$, which neither require the use of the Jacobian of the operator F, nor involve any complex computation besides the projection onto the set Ω and the evaluation of F. Hence, once the projection is easy to implement, projection methods can be applied to solve very large problems because of their simplicity (see [7, 9, 22]).

The efficiency of the projection methods is mainly determined by the descent direction and the step size. In this paper, using the limited-memory technique, we construct a new descent direction by combining the information of the last $m-1$ iterations and the kth iteration. The idea of limited-memory has been widely used in many optimization methods (see [2–4, 8, 23, 26]), while there are very few similar results in solving VIs [11]. Just like the conjugate gradient method [8], this is a useful technique for solving large scale optimization problems, which uses previous gradient information to generate a new iteration and has the stability property. Cragg and Levy [4] proposed a supermemory gradient method (multi-step gradient method) for finding the minimum of a function $\theta(x)$ whose variables x are unconstrained. Then Shi and Shen [26] presented another multi-step memory gradient method with Goldstein line search. The limited-memory technology was also adopted in quasi-Newton methods, e.g., limited-memory BFGS (see [2, 3, 23]).

Given the current iterate $x^k \in \Omega$, a projection method first constructs a descent direction for the implicit merit function $\frac{1}{2}\|x - x^*\|^2$ and chooses a suitable step size, and the next iterate is generated by performing a projection onto Ω. Assuming we have already obtained a descent direction $g(x^k)$, we propose to improve the algorithm by constructing a new descent direction $d(x^k)$ by combining $g(x^k)$ and the information from the last $m-1$ steps for $m \geq 1$:

$$d(x^k, \beta_k) = \begin{cases} g(x^k, \beta_k), & \text{if } k \leq m-1, \\ (1-\alpha_k) g(x^k, \beta_k) + \alpha_k \left(\sum_{i=2}^m \alpha_{k-i+1} d(x^{k-i+1}, \beta_{k-i+1}) \right), & \text{if } k \geq m, \end{cases} \tag{1.3}$$

where the meaning of $g(x^k, \beta_k)$, α_{k-i+1} and β_k will be specified in the section 3. This idea is very similar to the classic conjugate gradient (CG) algorithm for solving unconstrained optimization problems, except that here it is for VIs. Note that in CG, the parameters can be obtained via some well-known formulae or via some line search strategies, while for VIs, there are no such formulae and line search strategies can not be used too, due to the fact that the merit function $\frac{1}{2}\|x - x^*\|^2$ is implicit in the sense that x^* is unknown.

The rest of the paper is organized as follows. In Section 2, we summarize some basic definitions and properties to be used in the paper. In Section 3, we give the new limited-memory projection algorithm and analyze its convergence under some suitable conditions. In

Section 4, we report some numerical examples to demonstrate the feasibility and efficiency of the proposed method. We complete the paper with Section 5 by drawing some conclusions.

2 Preliminaries

In this section, we recall some basic concepts and useful properties that will play important roles in the following discussions.

First, we denote $\|x\| = \sqrt{\langle x, x\rangle}$ as the Euclidean norm. For a given vector $v \in \mathbb{R}^n$, the projection of v onto a convex set Ω under Euclidean norm, denoted by $P_\Omega(v)$, is defined as

$$P_\Omega(v) := \arg\min\{\|v-u\| \mid u \in \Omega\}.$$

It is well known that the projection operator $P_\Omega(\cdot)$ is nonexpansive [7], that is

$$\|P_\Omega(u) - P_\Omega(v)\| \le \|u-v\|, \qquad \forall u, v \in \mathbb{R}^n. \tag{2.1}$$

Moreover, we know that

$$\langle v - P_\Omega(v), w - P_\Omega(v)\rangle \le 0, \quad \forall v \in \mathbb{R}^n,\ \forall w \in \Omega. \tag{2.2}$$

Another well known result is that the VI(F,Ω) is equivalent to the projection equation

$$x = P_\Omega(x - \beta F(x)),$$

where $\beta > 0$ is an arbitrary constant. In other words, solving VI(F,Ω) is equivalent to finding a zero point of the residual function defined by

$$e(x,\beta) := x - P_\Omega(x - \beta F(x)). \tag{2.3}$$

That is, x^* is a solution of VI(F,Ω) if and only if x^* satisfies $e(x^*,\beta) = 0$.

Lemma 2.1. *For any $x \in \mathbb{R}^n$ and $\widetilde{\beta} \ge \beta > 0$, the following inequalities hold for the residual function*

$$\|e(x,\widetilde{\beta})\| \ge \|e(x,\beta)\|$$

and

$$\frac{\|e(x,\widetilde{\beta})\|}{\widetilde{\beta}} \le \frac{\|e(x,\beta)\|}{\beta}.$$

Proof. See [31] for a simple proof. □

Definition 2.2. The mapping $F(\cdot) : \mathbb{R}^n \to \mathbb{R}^n$ is said to be Lipschitz continuous on Ω with constant $L > 0$, if

$$\|F(x) - F(y)\| \le L\|x-y\|, \quad \forall x, y \in \Omega.$$

In order to describe the framework of limited-memory projection method, we give some definitions to make the description clear.

Definition 2.3. Let $c_0 > 0$ be a constant and $\varphi(x) : \mathbb{R}^n \to \mathbb{R}$ be a continuous function. We call $\varphi(x)$ an error measure function of VI(F,Ω) on Ω if it satisfies

$$\varphi(x) \ge c_0\|e(x,\beta)\|^2, \quad \forall x \in \Omega. \tag{2.4}$$

Definition 2.4. Let x^* be an arbitrary solution of $\mathrm{VI}(F,\Omega)$ and let $q(x)$ be a function from $\mathbb{R}^n$ into $\mathbb{R}^n$. We call $q(x)$ a profitable direction of $\mathrm{VI}(F,\Omega)$ if

$$\langle x - x^*, q(x)\rangle \geq \varphi(x), \quad \forall x \in \Omega, \tag{2.5}$$

where $\varphi(x)$ is an error measure function defined by (2.4).

Indeed, the profitable direction $q(x)$ can be viewed as an ascent direction of the distance function $\frac{1}{2}\|x-x^*\|^2$. It plays an important role in algorithm design and convergence analysis.

3 The Algorithm and its Convergence

In this section, we first describe the algorithmic framework of the limited-memory projection method. Then, we prove the global convergence of the new algorithm.

3.1 Algorithm

Before we present the new algorithm, for simplicity, we denote

$$g(x,\beta) = e(x,\beta) - \beta[F(x) - F(x - e(x,\beta))], \tag{3.1}$$

and

$$\varphi(x^k,\beta_k) = \langle e(x^k,\beta_k), g(x^k,\beta_k)\rangle.$$

According to the result in [16], $g(x^k,\beta_k)$ is a profitable direction at x^k. As like the conjugate direction method often has a desirable behavior, we introduce the limited-memory technique, and construct a new profitable direction by combining the information of the last $m-1$ iterations and the kth iteration.

The algorithm is described as follows.

Remark 3.1. Some remarks on Algorithm 1 are needed here:

For (unconstrained) optimization problems, there have been many results on limited-memory strategies (see [2–4,8,23,26]), as we have mentioned in the introduction. However, for VIs, our algorithm is the first one except [11]. The idea is very like a conjugate direction method, and here $g(x^k,\beta_k)$ is a profit direction of the (implicit) merit function $\frac{1}{2}\|x-x^*\|^2$ at x^k.

For the sequence $\{\beta_k\}$, it is bounded away from zero by the line search strategy and Lemma 3.1 in [10]. That is, there exits a real number $\beta_{\min} > 0$ such that $\beta_k \geq \beta_{\min} > 0$.

If we denote

$$\nu_k = \frac{\varphi(x^k,\beta_k)}{\sum\limits_{i=2}^{m} \alpha_{k-i+1}\,|\mu_{k-i+1}\varphi_{k-i+1} - \|d_{k-i+1}\|\cdot\|x^k - x^{k-i+1}\||},$$

then

$$\alpha_k = \xi_k\nu_k,$$

and

$$\mu_k = 1 - \alpha_k - \xi_k = 1 - (1+\nu_k)\xi_k.$$

To ensure that $\mu_k > 0$, we can select $\xi_k = \frac{\theta_k}{1+\nu_k}$, where $0 < \theta_k \leq \theta_{\max} < 1$. Then $\mu_k = 1 - \theta_k \geq 1 - \theta_{\max} = \mu_{\min}$, which means that $\{\mu_k\}$ is uniformly bounded away from zero. For θ_k, when ν_k is large, we select a smaller θ_k, otherwise we select a larger θ_k.

Algorithm 1 Limited-memory projection method (LMPM).

1: Choose a starting point $x^0 \in \mathbb{R}^n$. Take $\gamma \in (0,2)$, $m \geq 1$, $L \in (0,1)$, $\mu \in (0,1)$ and $\beta > 0$.

2: **while** $\|e(x^k, \beta_k)\| \geq \varepsilon$ **do**

3: Find the smallest nonnegative integer l_k, $\beta_k = \beta\mu^{l_k}$ satisfying

$$\beta_k\|F(x^k) - F(x^k - e(x^k, \beta_k))\| \leq L\|e(x^k, \beta_k)\|. \tag{3.2}$$

Update the next iteration via

$$x^{k+1} = P_\Omega(x^k - \gamma\rho_k d(x^k, \beta_k)), \tag{3.3}$$

where

$$d(x^k, \beta_k) = \begin{cases} g(x^k, \beta_k), & \text{if } k \leq m-1, \\ (1-\alpha_k)g(x^k, \beta_k) + \alpha_k\left(\sum\limits_{i=2}^{m} \alpha_{k-i+1} d(x^{k-i+1}, \beta_{k-i+1})\right), & \text{if } k \geq m. \end{cases} \tag{3.4}$$

$$\alpha_k = \frac{\xi_k \varphi(x^k, \beta_k)}{\sum\limits_{i=2}^{m} \alpha_{k-i+1} \left|\mu_{k-i+1}\varphi_{k-i+1} - \|d_{k-i+1}\| \cdot \|x^k - x^{k-i+1}\|\right|}, \tag{3.5}$$

$$\mu_k = 1 - \alpha_k - \xi_k,$$

and

$$\rho_k = \frac{\mu_k \varphi_k(x^k, \beta_k)}{\|d(x^k, \beta_k)\|^2}. \tag{3.6}$$

Adjust the parameter β via

$$\beta_{k+1} := \begin{cases} \beta_k/0.7, & \text{if } \beta_k\|F(x^k) - F(x^k - e(x^k, \beta_k))\| \leq 0.2\|e(x^k, \beta_k)\|; \\ \beta_k, & \text{otherwise.} \end{cases}$$

4: **end while**

3.2 Convergence analysis

In this subsection, we prove the global convergence of the proposed method under some standard assumptions in the variational inequality literature. We begin the analysis with a known lemma proved in [16] and skip the details.

Lemma 3.2. *Let x^* be a solution of VI(F, Ω) and the sequence $\{x^k\}$ be generated by Algorithm 1. Under the assumption that F is monotone, we have*

$$\langle x^k - x^*, g(x^k, \beta_k)\rangle \geq \varphi(x^k, \beta_k).$$

According to the result in [16], we can further derive that $\varphi(x^k, \beta_k) \geq (1-L)\|e(x^k, \beta_k)\|^2$. Hence, under the assumption $L \in (0,1)$, it is clear that $g(x^k, \beta_k)$ is a profitable direction at x^k by Lemma 3.2. Next, we prove that $d(x^k, \beta_k)$ is also a profitable direction at x^k.

Lemma 3.3. *Let x^* be a solution of VI(F, Ω). Then $d(x^k, \beta_k)$ defined by* (3.4) *satisfies*

$$\langle x^k - x^*,\ d(x^k, \beta_k)\rangle \geq \mu_k\varphi(x^k, \beta_k) \geq 0.$$

Proof. We divide the proof into two parts. First, we prove the case $k \le m-1$. By the definition of $d(x^k, \beta_k)$ in (3.4) and the Lemma 3.2, we have

$$d(x^k, \beta_k) = g(x^k, \beta_k),$$

and

$$\langle x^k - x^*, d(x^k, \beta_k)\rangle \ge \varphi(x^k, \beta_k) \ge \mu_k \varphi(x^k, \beta_k) \ge 0 \quad \text{for any} \quad \mu_k \le 1.$$

Second, if $k \ge m$, we prove the result by mathematical induction. Assume $d(x^{k-1}, \beta_{k-1})$ is a profitable direction. Then we have

$$\langle x^{k-1} - x^*, d(x^{k-1}, \beta_{k-1})\rangle \ge \mu_{k-1}\varphi(x^{k-1}, \beta_{k-1}) \ge 0,$$

and

$$d(x^k, \beta_k) = (1-\alpha_k) g(x^k, \beta_k) + \alpha_k \sum_{i=2}^{m} \alpha_{k-i+1} d(x^{k-i+1}, \beta_{k-i+1}).$$

For simplicity, we use the notations $d_{k-i+1} := d(x^{k-i+1}, \beta_{k-i+1})$ and $\varphi_{k-i+1} := \varphi(x^{k-i+1}, \beta_{k-i+1})$ in the following analysis. Using the Cauchy-Schwarz inequality, we get

$$\begin{aligned}
&\sum_{i=2}^{m} \alpha_{k-i+1} \langle d_{k-i+1}, x^k - x^*\rangle \\
&= \sum_{i=2}^{m} \alpha_{k-i+1} \left(\langle d_{k-i+1}, x^{k-i+1} - x^*\rangle + \langle d_{k-i+1}, x^k - x^{k-i+1}\rangle \right) \\
&\ge \sum_{i=2}^{m} \alpha_{k-i+1} \left(\mu_{k-i+1}\varphi_{k-i+1} - \|d_{k-i+1}\| \cdot \|x^k - x^{k-i+1}\| \right).
\end{aligned}$$

Then

$$\begin{aligned}
&\langle d(x^k, \beta_k), x^k - x^*\rangle \\
&= (1-\alpha_k)\langle g_k, x^k - x^*\rangle + \alpha_k \sum_{i=2}^{m} \alpha_{k-i+1} \langle d_{k-i+1}, x^k - x^*\rangle \\
&\ge (1-\alpha_k)\varphi_k + \alpha_k \sum_{i=2}^{m} \alpha_{k-i+1} \left(\mu_{k-i+1}\varphi_{k-i+1} - \|d_{k-i+1}\| \cdot \|x^k - x^{k-i+1}\| \right) \\
&\ge (1-\alpha_k)\varphi_k - \alpha_k \sum_{i=2}^{m} \alpha_{k-i+1} \left| \mu_{k-i+1}\varphi_{k-i+1} - \|d_{k-i+1}\| \cdot \|x^k - x^{k-i+1}\| \right|.
\end{aligned}$$

By the definition of α_k, if ξ_k is a positive number and small enough, α_k always exists such that $1 - \alpha_k - \xi_k > 0$. Then, we conclude that

$$\langle d(x^k, \beta_k), x^k - x^*\rangle \ge (1 - \alpha_k - \xi_k)\varphi(x^k, \beta_k) = \mu_k \varphi(x^k, \beta_k) \ge 0.$$

This completes the proof. □

Next, we derive the concrete form of the step size ρ_k. Let

$$x^{k+1}(\rho) = P_\Omega(x^k - \rho d(x^k, \beta_k))$$

be the function of ρ dependent on (x^k, β_k), and let

$$\Psi(\rho) := \|x^k - x^*\|^2 - \|x^{k+1}(\rho) - x^*\|^2$$

be a progress-function to measure the improvement obtained at the k-th iteration. Clearly, larger $\Psi(\rho)$ results in better improvement. Thus, we hopefully maximize the $\Psi(\rho)$ to find a maximum improvement at each iteration. We have the following result to get an optimal ρ.

Lemma 3.4. *Let x^* be an arbitrary solution of* (1.1)*. Then we have*

$$\Psi(\rho) \geq \Phi(\rho),$$

where

$$\Phi(\rho) = 2\rho\mu_k\varphi(x^k, \beta_k) - \rho^2\|d(x^k, \beta_k)\|^2. \tag{3.7}$$

Proof. By invoking Lemma 3.3, it is easy to see that

$$\begin{aligned}\Psi(\rho) &= \|x^k - x^*\|^2 - \|x^{k+1}(\rho) - x^*\|^2 \\ &= \|x^k - x^*\|^2 - \|P_\Omega(x^k - \rho d(x^k, \beta_k)) - x^*\|^2 \\ &\geq \|x^k - x^*\|^2 - \|x^k - \rho d(x^k, \beta_k) - x^*\|^2 \\ &= 2\rho\langle d(x^k, \beta_k), x^k - x^*\rangle - \rho^2\|d(x^k, \beta_k)\|^2 \\ &\geq 2\rho\mu_k\varphi_k(x^k, \beta_k) - \rho^2\|d(x^k, \beta_k)\|^2,\end{aligned}$$

where the first inequality follows from the nonexpansiveness of the projection operator (2.1). The assertion is proved. □

Since $\Phi(\rho)$ is a quadratic function of ρ, we can find that $\Phi(\rho)$ attains the maximum at the point

$$\rho_k = \frac{\mu_k\varphi_k(x^k, \beta_k)}{\|d(x^k, \beta_k)\|^2}. \tag{3.8}$$

Accordingly, we can use the optimal choice of ρ_k in Algorithm 1. Moreover, for any relaxed factor $\gamma > 0$, it turns out that

$$\begin{aligned}\Phi(\gamma\rho_k) &= 2\gamma\rho_k\mu_k\varphi_k - \gamma^2\rho_k^2\|d(x^k, \beta_k)\|^2 \\ &= \gamma(2 - \gamma)\rho_k\mu_k\varphi_k.\end{aligned}$$

We should limit $\gamma \in (0, 2)$ to ensure that an improvement can be obtained at each iteration. Empirically, we suggest to take $[1, 2)$ for fast convergence in practice.

Theorem 3.5. *Suppose that x^* is an arbitrary solution of* (1.1)*. Then, the sequence $\{x^k\}$ generated by Algorithm 1 satisfies*

$$\|x^{k+1} - x^*\|^2 \leq \|x^k - x^*\|^2 - \gamma(2 - \gamma)\rho_k\mu_k\varphi(x^k, \beta_k). \tag{3.9}$$

Consequently, the sequence $\{x^k\}$ is bounded.

Proof. It follows from (3.3) and the nonexpansiveness of the projection operator (2.1) that

$$\begin{aligned}\|x^{k+1} - x^*\|^2 &= \|P_\Omega(x^k - \gamma\rho_k d(x^k, \beta_k)) - x^*\|^2 \\ &\leq \|x^k - x^* - \gamma\rho_k d(x^k, \beta_k)\|^2 \\ &= \|x^k - x^*\|^2 - 2\gamma\rho_k\langle d(x^k, \beta_k), x^k - x^*\rangle + \gamma^2\rho_k^2\|d(x^k, \beta_k)\|^2\end{aligned}$$

$$\leq \|x^k - x^*\|^2 - 2\gamma\rho_k\mu_k\varphi(x^k,\beta_k) + \gamma^2\rho_k\mu_k\varphi(x^k,\beta_k)$$
$$= \|x^k - x^*\|^2 - \gamma(2-\gamma)\rho_k\mu_k\varphi(x^k,\beta_k),$$

where the second inequality follows from Lemma 3.3 and the definitions of $\varphi(x^k,\beta_k)$ and ρ_k.

Since μ_k, $\varphi_k \geq 0$ and $\gamma \in (0,2)$, it follows that

$$\|x^{k+1} - x^*\|^2 \leq \cdots \leq \|x^0 - x^*\|^2. \tag{3.10}$$

The assertion then follows immediately. □

Theorem 3.6. *The sequence $\{x^k\}$ generated by Algorithm 1 converges to a solution of* (1.1)*.*

Proof. Since we have shown in Theorem 3.5 that $\{x^k\}$ is bounded, it follows from the continuity of $d(x^k,\beta_k)$ that there exists a constant $M > 0$ such that

$$\|d(x^k,\beta_k)\|^2 \leq M, \quad \forall k \geq 1.$$

From (3.9), we have

$$\|x^{k+1} - x^*\|^2 \leq \|x^k - x^*\|^2 - \gamma(2-\gamma)\frac{(1-L)^2\mu_{\min}^2\|e(x^k,\beta_k)\|^4}{M},$$

which means

$$\sum_{k=0}^{\infty}\|e(x^k,\beta_k)\|^4 < \infty.$$

Hence,

$$\lim_{k\to\infty}\|e(x^k,\beta_k)\| = 0.$$

By $\beta_k \geq \beta_{min} > 0$ and Lemma 2.1, we have that

$$\lim_{k\to\infty}\|e(x^k,\beta_{\min})\| = 0.$$

Since $\{x^k\}$ is bounded, it has at least a cluster point, denote by $\bar{x}$ and let $\{x^{k_j}\}$ be the subsequence converging to it. Taking limit along this subsequence and using the continuity of the residual function, we have

$$\|e(\bar{x},\beta_{\min})\| = \|e(\lim_{j\to\infty}x^{k_j},\beta_{\min})\| = \lim_{j\to\infty}\|e(x^{k_j},\beta_{\min})\| = 0,$$

indicating that $\bar{x}$ is a solution of (1.1). Since in (3.10), x^* is an arbitrary solution of (1.1), we can set $x^* := \bar{x}$ in it and get

$$\|x^{k+1} - x^*\|^2 \leq \|x^k - x^*\|^2$$

and the whole sequence $\{x^k\}$ converges to $\bar{x}$. This completes the proof. □

4 Numerical Experiments

In this section, we study the numerical performance of Algorithm 1 and denote it "LMPM" for short. Specifically, we apply "LMPM" to solve complementarity problems and generalized Nash equilibrium problems. In addition, we compare "LMPM" with some benchmark projection-like methods for complementarity problems, such as the extragradient method

[20,21] with an adaptive strategy introduced in [17] (denoted as "REGM"); the improved self-adaptive projection method proposed in [29] ("ISAPM" for short); two prediction-correction methods presented in [18] (denoted as "PC-I" and "PC-II", respectively). For generalized Nash equilibrium problems, we compare "LMPM" with the first projection algorithm developed in [30] (denoted as "ZQXA1") and the improved two-step method proposed in [12] (denoted as "HZQX").

All codes were written by MATLAB 2008b and run on a HP personal computer with Pentium Dual-Core processor 2.66 GHz and 2 GB memory. To demonstrate the efficiency of "LMPM", we report the numerical results in terms of the number of iterations ("Iter.") and computing time in seconds ("Time").

4.1 Complementarity problems

We first consider a special case of the VI(F, Ω) problem, the complementarity problem, which is to find a vector $x \in \mathbb{R}^n$ such that

$$x \geq 0, \qquad F(x) \geq 0 \qquad \text{and} \qquad \langle x,\ F(x)\rangle = 0.$$

Four problems are considered as well as [19] in this section. Below, we describe the details of the underlying mapping $F(x)$.

Example 4.1. The first example is a linear complementarity problem, that is

$$F(x) = Mx + q,$$

where $q = (-1, -1, \cdots, -1)^\top$, the matrix M is generated synthetically such that it has a preset condition number. This is accomplished by setting

$$M = V \Sigma V^\top \qquad \text{and} \qquad V = 2I_n - \frac{vv^\top}{\|v\|^2},$$

where V is a Householder matrix and $\Sigma = \mathrm{diag}(\sigma_1, \sigma_2, \cdots, \sigma_n)$ is a diagonal matrix. Here, each component σ_i $(i = 1, 2, \cdots, n)$ is generated such as follows

$$\sigma_i = \cos\frac{i\pi}{n+1} + 1 + \frac{(\cos\frac{\pi}{n+1} + 1) - \mathrm{cond}(M)(\cos\frac{n\pi}{n+1} + 1)}{\mathrm{cond}(M) - 1},$$

and the vector v is uniformly distributed in the interval $(-1, 1)$. In our test, we set $\mathrm{cond}(M) = 100$.

Example 4.2. The second one is an asymmetric nonlinear complementarity problem, whose $F(x)$ consists of a linear part and a nonlinear part. Concretely,

$$F(x) = Mx + D(x) + q,$$

where $Mx + q$ is the linear part and $D(x)$ is the nonlinear part. We form the linear part as described in [14] (see also [18]), that is $M = A^\top A + B$, where A is an $n \times n$ matrix whose entries are randomly generated in the interval $(-5, 5)$ and the skew-symmetric matrix B is generated in the same way; the vector q is generated randomly in the interval $(-500, 0)$. For the nonlinear part $D(x)$, each component of it is $D_j(x) = a_j \cdot \arctan(x_j)$ $(j = 1, 2, \cdots, n)$, where a_j is a uniformly random variable in $(0, 1)$.

Example 4.3. This example is same as Example 4.2, but with different q who is generated randomly in the interval $(-500, 500)$.

Example 4.4. The last complementarity problem under test has a known solution $x^* \in \mathbb{R}^n_+$. Specifically, let p be uniformly distributed in the interval $(-10, 10)$ and $x^* = \max(p, 0)$. By setting

$$w = \max(-p, 0) \qquad \text{and} \qquad q = w - (Mx^* + D(x^*)),$$

where the matrix M and the nonlinear part $D(x)$ are generated in the same way as Example 4.2. Therefore, it is clear that

$$F(x^*) = Mx^* + D(x^*) + q = w = \max(-p, 0),$$

and

$$\langle x^*, F(x^*)\rangle = \langle \max(p, 0),\ \max(-p, 0)\rangle = 0.$$

In this way, we get a nonlinear complementarity problem with a known solution x^* successfully.

Throughout the experiments on the four examples, we took $\nu = 0.9$ and $\mu = 0.3$ for "REGM", "PC-I" and "PC-II" methods, and $\gamma = 1.9$ for both "PC-I" and "PC-II" methods. The parameters in "ISAPM" are specified as $\gamma = 1.8$, $L = 0.95$, $\mu = 0.7$, and $\tau = 0.9$. Finally, we set $L = 0.9$, $\mu = 0.5$, and $\beta = 1$ for "LMPM". To ensure the fairness of comparison for the five methods, we terminated all the methods by setting the stopping criterion as $\|e(x^k, 1)\|_\infty \le 10^{-6}$.

Notice that two additional parameters γ and m are involved in "LMPM", we thus investigate the behaviors of different γ and m numerically. We consider four scenarios of the dimensionality with $n = \{100, 500, 1000, 2000\}$ and report the corresponding results in Tables 1 and 2.

The data in Table 1 show that larger γ performs better than smaller ones. However, since the global convergence is built up under the assumption $\gamma \in (0, 2)$, we suggest to take $\gamma \in [1, 2)$ for fast convergence in practice. Thus, we set $\gamma = 1.99$ in the rest of experiments. The numerical results reported in Table 2 clearly show that the "LMPM" weakly depends on the choice of m. In other words, the "LMPM" runs stably for these complementarity problems in this section.

Finally, we compare "LMPM" with other four benchmark projection methods mentioned at the beginning of this section. We consider six scenarios of the dimensionality with $n = \{50, 300, 700, 1000, 2000, 3000\}$, and set $m = 2$ in accordance to the data in Table 2. The results are reported in Table 3.

It can be easily seen from Table 2 that the "LMPM" outperforms the other four projection methods in term of taking the fewest iterations. However, the "LMPM" requires more computing time in some cases. The main reason is that "LMPM" needs to memory more information and it increases the amount of storage. Thus, we will pay our attention on reducing the storage of "LMPM" in the future.

4.2 Generalized Nash equilibrium problems

In this subsection, we consider an important application of variational inequality problem in characterizing equilibrium problems. Specifically, the problem under consideration is the generalized Nash equilibrium problem (GNEP), which is an extension of the classical Nash equilibrium problem and has been widely used in many fields. In the past decades, the GNEP has been studied theoretically and numerically in the literature, see, e.g., [6, 12, 13, 25, 30].

Table 1: Numerical performance of different γ for complementarity problems.

Dimension		$n = 100$		$n = 500$		$n = 1000$		$n = 2000$	
γ	Problem	Iter.	Time	Iter.	Time	Iter.	Time	Iter.	Time
$\gamma = 0.8$	Ex. 4.1	1573	0.234	1626	0.689	1642	4.650	1630	16.794
	Ex. 4.2	1168	0.183	1097	0.476	918	2.641	852	8.960
	Ex. 4.3	435	0.162	435	0.172	402	1.310	371	4.462
	Ex. 4.4	929	0.137	1165	0.480	1391	4.167	1413	15.961
$\gamma = 1.0$	Ex. 4.1	1234	0.176	1279	0.529	1293	3.668	1283	13.170
	Ex. 4.2	900	0.137	840	0.326	732	2.114	676	7.234
	Ex. 4.3	336	0.133	339	0.166	312	1.045	286	3.411
	Ex. 4.4	733	0.115	910	0.424	1082	3.294	1102	12.355
$\gamma = 1.3$	Ex. 4.1	912	0.133	963	0.421	955	3.048	965	9.761
	Ex. 4.2	669	0.099	605	0.222	528	1.537	485	5.218
	Ex. 4.3	241	0.101	251	0.114	230	0.772	210	2.621
	Ex. 4.4	543	0.090	672	0.312	800	2.437	829	9.370
$\gamma = 1.6$	Ex. 4.1	734	0.107	760	0.290	768	2.189	764	8.000
	Ex. 4.2	523	0.079	475	0.189	412	1.212	376	4.079
	Ex. 4.3	182	0.078	204	0.096	188	0.646	169	2.092
	Ex. 4.4	432	0.066	534	0.224	627	1.938	641	7.342
$\gamma = 1.9$	Ex. 4.1	584	0.081	600	0.232	608	1.729	602	6.215
	Ex. 4.2	435	0.078	392	0.159	335	0.992	305	3.345
	Ex. 4.3	150	0.057	158	0.086	146	0.496	132	1.710
	Ex. 4.4	344	0.056	435	0.191	513	1.588	530	6.109

Table 2: Numerical performance of different m for complementarity problems.

Dimension		$n = 100$		$n = 500$		$n = 1000$		$n = 2000$	
m	Problem	Iter.	Time	Iter.	Time	Iter.	Time	Iter.	Time
$m = 2$	Ex. 4.1	584	0.082	600	0.225	608	1.743	602	6.253
	Ex. 4.2	435	0.062	392	0.150	335	0.998	305	3.350
	Ex. 4.3	150	0.059	158	0.075	146	0.500	132	1.684
	Ex. 4.4	344	0.051	435	0.204	513	1.592	530	6.099
$m = 4$	Ex. 4.1	586	0.090	608	0.278	614	1.765	608	6.409
	Ex. 4.2	437	0.067	397	0.162	337	1.031	310	3.564
	Ex. 4.3	152	0.060	160	0.070	148	0.517	134	1.820
	Ex. 4.4	342	0.058	437	0.193	514	1.843	532	6.376
$m = 6$	Ex. 4.1	588	0.091	608	0.277	615	1.913	611	6.474
	Ex. 4.2	441	0.076	397	0.197	338	1.049	321	3.748
	Ex. 4.3	154	0.062	163	0.083	150	0.585	135	1.939
	Ex. 4.4	344	0.060	438	0.222	517	1.714	529	6.474
$m = 8$	Ex. 4.1	590	0.094	607	0.290	616	1.803	612	6.590
	Ex. 4.2	441	0.083	399	0.205	345	1.098	321	4.204
	Ex. 4.3	156	0.065	165	0.082	152	0.595	139	2.169
	Ex. 4.4	346	0.084	440	0.301	518	1.725	535	6.597

However, it is still a big challenge to design efficient algorithms for solving GNEP. In the rest of this section, we employ the "LMPM" to solve the GNEP and compare it with other two projection-like methods numerically.

We skip the background and description of GNEP and refer the reader to [6, 25] for

Table 3: Numerical results of the different projection methods for Example 4.1.

n	Problem	REGM		PC-I		PC-II		ISAPM		LMPM	
		Iter.	Time	Iter.	Time	Iter.	Time	Iter.	Time	Iter.	Time
50	Ex. 4.1	1186	0.060	593	0.037	495	0.038	514	0.046	470	0.055
	Ex. 4.2	1361	0.069	736	0.047	577	0.042	578	0.052	365	0.042
	Ex. 4.3	437	0.022	255	0.018	197	0.014	181	0.018	151	0.019
	Ex. 4.4	745	0.041	468	0.035	329	0.025	318	0.032	288	0.039
300	Ex. 4.1	1415	0.260	722	0.168	596	0.160	610	0.168	572	0.200
	Ex. 4.2	1660	0.312	1056	0.279	762	0.195	712	0.210	405	0.151
	Ex. 4.3	626	0.112	432	0.109	300	0.075	291	0.077	158	0.068
	Ex. 4.4	950	0.186	584	0.148	428	0.109	409	0.114	381	0.155
700	Ex. 4.1	1413	3.732	648	2.148	591	2.075	604	2.100	562	2.576
	Ex. 4.2	1659	4.682	1020	3.703	764	2.709	713	2.288	346	1.616
	Ex. 4.3	637	1.732	430	1.579	311	1.064	272	0.966	154	0.821
	Ex. 4.4	969	2.857	621	2.389	451	1.576	415	1.399	356	1.823
1000	Ex. 4.1	1435	7.437	664	4.263	607	4.284	619	4.270	575	6.169
	Ex. 4.2	1612	8.265	1018	6.815	774	5.670	708	4.983	323	3.795
	Ex. 4.3	584	3.051	400	2.886	285	1.827	248	1.666	138	2.044
	Ex. 4.4	1290	6.918	828	5.866	584	3.852	561	3.541	483	5.359
2000	Ex. 4.1	1422	29.609	657	16.983	599	15.204	612	15.943	570	22.757
	Ex. 4.2	1567	32.063	1004	26.460	740	18.710	687	16.417	293	12.350
	Ex. 4.3	609	13.346	401	12.813	294	8.388	261	7.061	126	6.063
	Ex. 4.4	1298	27.044	834	24.226	585	15.862	556	14.209	500	21.280
3000	Ex. 4.1	1413	61.211	653	36.182	593	33.646	607	34.453	566	47.161
	Ex. 4.2	1519	69.771	964	59.735	725	42.777	658	36.362	266	28.884
	Ex. 4.3	586	26.095	399	24.214	286	15.903	251	14.200	119	12.395
	Ex. 4.4	1405	61.874	921	52.724	625	33.335	619	30.444	546	20.259

details. In this section, we borrow the notations used in [30] and consider a two-person game which comes from [13] and [24]. Specifically, in the two-person game, each player chooses a number x_i between 0 and 10 such that the sum of their numbers must be less than or equal to 15. The cost functions u_i and the set mappings K^i are given by

$$u_1(x_1,x_2) = (x_1)^2 + \frac{8}{3}x_1x_2 - 34x_1, \quad u_2(x_1,x_2) = (x_2)^2 + \frac{5}{4}x_1x_2 - 24.5x_2,$$
$$K^1(\bar{x}_2) = \{0 \le x_1 \le 10, x_1 \le 15 - \bar{x}_2\}, \quad K^2(\bar{x}_1) = \{0 \le x_2 \le 10, x_2 \le 15 - \bar{x}_1\}.$$

As pointed out in [30], the set of GNEP solution of this game is composed of the point $(5,9)^\top$ and the line segment $[(9,6)^\top,(10,5)^\top]$.

Throughout the experiments, we terminate the three compared methods at the same stopping criterion used in last section. For the parameters used in "ZQXA1", we took $\gamma = 1$, $l = 0.5$, $\lambda = 1.99$, and $\mu = 0.3$. For the method "HZQX", we set $\gamma = 1$, $l = 0.5$, $c = 0.3$, $\rho = 3.5$, $\lambda = 1.98$ and $\mu = 0.85$. Finally, we set $\beta = 1$, $L = 0.9$, $\mu = 0.5$, $m = 3$ $\xi = 0.1$and $\gamma = 1.99$ for our "LMPM". We compared the three methods by setting five different starting points and reported the results in Table 4.

Table 4: Numerical comparisons between the three methods for GNEP.

Starting point	Iter.			Time			Approximate solution		
	ZQXA1	HZQX	LMPM	ZQXA1	HZQX	LMPM	ZQXA1	HZQX	LMPM
$(0,0)^\top$	201	58	158	0.495	0.015	0.021	$(5,9)^\top$	$(5,9)^\top$	$(5,9)^\top$
$(5,10)^\top$	212	45	136	0.014	0.016	0.012	$(5,9)^\top$	$(5,9)^\top$	$(5,9)^\top$
$(10,10)^\top$	206	61	66	0.014	0.016	0.007	$(5,9)^\top$	$(5,9)^\top$	$(5,9)^\top$
$(0,10)^\top$	177	48	106	0.011	0.012	0.009	$(5,9)^\top$	$(5,9)^\top$	$(5,9)^\top$
$(5,5)^\top$	235	52	93	0.014	0.015	0.007	$(5,9)^\top$	$(5,9)^\top$	$(5,9)^\top$

From the data in Table 4, we can see that the "HZQX" outperforms the other two methods in term of taking fewest iterations. However, the global convergence of the "HZQX" method is built up under the co-coercive assumption, which is stronger than the condition of our "LMPM". Thus, the stronger requirement of "HZQX" may preclude its potential applications in some cases. Moreover, we observe that the "LMPM" takes the least computing time to obtain a good solution. Our new method is also efficient and reliable for this problem.

Below, we further study the numerical performance of "LMPM" with different m for the GNEP. We tested five different starting points and considered five cases of $m = \{2, 3, 4, 6, 8\}$. The corresponding results are summarized in Table 5.

Table 5: Numerical performance of "LMPM" with different m for GNEP.

Starting point	$m = 2$		$m = 3$		$m = 4$		$m = 6$		$m = 8$	
	Iter.	Time	Iter.	Time	Iter.	Time	Iter.	Time	Iter.	Time
$(0,0)^\top$	123	0.012	122	0.012	191	0.019	185	0.021	190	0.021
$(5,10)^\top$	102	0.010	169	0.019	164	0.018	162	0.017	156	0.018
$(10,10)^\top$	76	0.008	196	0.020	271	0.028	223	0.024	250	0.029
$(0,10)^\top$	60	0.006	163	0.018	90	0.010	88	0.009	85	0.011
$(5,5)^\top$	126	0.013	182	0.017	188	0.021	181	0.018	182	0.021

From Table 5 we can see that our "LMPM" performs well for different m. However, the data also clearly show that the choice of m can affect the convergence of the method for different starting points. Thus, we will further study how to choose a better m iteratively.

5 Conclusions

In this paper, we present a limited-memory projection method. The method can be viewed as generalization of conjugate gradient methods for solving unconstrained nonlinear programming problems. Under some suitable conditions, we prove that the proposed algorithm is globally convergent. Some preliminary numerical results demonstrate the proposed algorithm is efficient and reliable for solving monotone variational inequalities in practice.

Acknowledgements

We thank the two anonymous referees for the helpful comments and suggestions, which help us improve the paper greatly. The first author is grateful to Dr. Hongjin He for his help and comments on this paper.

References

[1] D.P. Bertsekas and E.M. Gafni, Projection method for variational inequalities with applications to the traffic assignment problem, *Math. Program.* 17 (1982) 139–159.

[2] R.H. Byrd, J. Nocedal, and R.B. Schnabel, Representations of quasi-newton matrices and their use in limited memory methods, *Math. Program.* 63 (1994) 129–156.

[3] R.H. Byrd, P.H. Lu, J. Nocedal, and C.Y. Zhu, A limited memory algorithm for bound constrained optimization, *SIAM J. Sci. Comput.* 16 (1995) 1190–1208.

[4] E.E. Cragg and A.V. Levy, Study on a supermemory gradient method for the minimization of functions, *J. Optim. Theory Appl.* 4 (1969) 191–205.

[5] S. Dafermos, Traffic equilibrium and variational inequalities, *Transpor Sci.* 14 (1980) 42–54.

[6] F. Facchinei and C. Kanzow, Generalized Nash equilibrium problems, *Ann. Oper. Res.* 175 (2010) 177–211.

[7] F. Facchinei and J.S. Pang, *Finite-Dimensional Variational Inequalities and Complementarity Problems*, Springer, New York, 2003.

[8] J.C. Gilbert and J. Nocedal, Global convergence properties of conjugate gradient methods for optimization, *SIAM J. Control Optim.* 2 (1992) 21–42.

[9] A.A. Goldstein, Convex programming in Hilbert space, *Bull. Amer. Math. Soc. (N.S.)* 70 (1964) 709–710.

[10] D.R. Han and H.K. Lo, Two new self-adaptive projection methods for variational inequality problems, *Comput. Math. Appl.* 43 (2002) 1529–1537.

[11] D.R. Han, A new class of projection and contraction methods for solving variational inequality problems, *Comput. Math. Appl.* 51 (2006) 937–950.

[12] D.R. Han, H.C. Zhang, G. Qian, and L.L. Xu, An improved two-step method for solving generalized nash equilibrium problems, *European J. Oper. Res.* 216 (2012) 613–623.

[13] P.T. Harker, Generalized Nash games and quasi-variational inequalities, *European J. Oper. Res.* 54 (1991) 81–94.

[14] P.T. Harker and J.S. Pang, A damped-Newton method for the linear complementarity problem, *Lect. Appl. Math.* 26 (1990) 265–284 .

[15] T. Hastie, R. Tibshirani, and J. Friedman, The Elements of Statistical Learning: Data Mining, Inference and Prediction, *Math. Intelligencer.* 27 (2005) 83-85.

[16] B.S. He, A class of projection and contraction methods for monotone variational inequalities, *Appl. Math. Optim.* 35 (1997) 69–76.

[17] B.S. He and L.Z. Liao, Improvements of some projection methods for monotone nonlinear variational inequalities, *J. Optim. Theory Appl.* 112 (2002) 111–128.

[18] B.S. He, X.M. Yuan, and J.Z. Zhang, Comparison of two kinds of prediction-correction methods for monotone variational inequalities, *Comput. Optim. Appl.* 27 (2004) 247–267.

[19] H.J. He, D.R. Han, and Z.B. Li, Some projection methods with the BB step sizes for variational inequalities, *J. Comput. Appl. Math.* 236 (2012) 2590–2604.

[20] E.N. Khobotov, Modification of the extragradient method for solving variational inequalities and certain optimization problems, *Comput. Math. Math. Phys.* 27 (1987) 120–127.

[21] G.M. Korpelevich, The extragradient method for finding saddle points and other problems, *Matecon.* 12 (1976) 747–756.

[22] E.S. Levitin and B.T. Polyak, Constrained minimization problems, *Comput. Math. Math. Phys.* 6 (1966) 1–50.

[23] D.C. Liu and J. Nocedal, On the limited memory method for large scale optimization, *Math. Program.* 45 (1989) 503–528.

[24] J. Outrata and J. Zowe, A numerical approach to optimization problems with variational inequality constraints, *Math. Program.* 68 (1995) 105–130.

[25] J.S. Pang and M. Fukushima, Quasi-variational inequalities, generalized Nash equilibria, and multi-leader-follower games, *Comput. Manag. Sci.* 2 (2005) 21–56.

[26] Z.J. Shi and J. Shen, Memory gradient method with Goldstein line search, *Comput. Math. Appl.* 53 (2006) 28–40.

[27] J.M. Wooldridge, *Introductory Econometrics: A Modern Approach*, South Western College Publications, Canada, 2009.

[28] G.L. Xue and Y.Y. Ye, An efficient algorithm for minimizing a sum of euclidean norms with applications, *SIAM J. Control Optim.* 7 (1997) 1017–1039.

[29] X.H. Yan, D.R. Han, and W.Y. Sun, A self-adaptive projection method with improved step-size for solving variational inequalities, *Comput. Math. Appl.* 55 (2008) 819–832.

[30] J.Z. Zhang, B. Qu, and N.H. Xiu, Some projection-like methods for the generalized Nash equilibria, *Comput. Optim. Appl.* 45 (2010) 89–109.

[31] T. Zhu and Z.G. Yu, A simple proof for some important properties of the projection mapping, *Math. Inequal. Appl.* 7 (2004) 453–456.

Manuscript received 27 January 2014
revised 5 April 2015
accepted for publication 19 April 2015

Qun Wang
School of Mathematical Sciences, Nanjing Normal University
1# Wenyuan Road, Nanjing, P.R. China
E-mail address: `0wangqun0@163.com`

Xingju Cai
School of Mathematical Sciences, Nanjing Normal University
1# Wenyuan Road, Nanjing, P.R. China
E-mail address: `caixingju@njnu.edu.cn`

Deren Han
School of Mathematical Sciences, Nanjing Normal University
1# Wenyuan Road, Nanjing, P.R. China
E-mail address: `handeren@njnu.edu.cn`