A Self-Adaptive Projection and Contraction Method for Monotone Symmetric Linear Variational Inequalities

LI-ZHI LIAO*
Department of Mathematics, Hong Kong Baptist University
Kowloon Tong, Kowloon, Hong Kong, P.R. China
lzliao@hkbu.edu.hk

SHENGLI WANG
Department of Mathematics, Nanjing University
Nanjing, 210093, P.R. China
slwangbb@sina.com

(Received February 2001; accepted March 2001)

Abstract—In this paper, we present a self-adaptive projection and contraction (SAPC) method for solving symmetric linear variational inequalities. Preliminary numerical tests show that the proposed method is efficient and effective and depends only slightly on its initial parameter. The global convergence of the new method is also addressed. © 2001 Elsevier Science Ltd. All rights reserved.

Keywords—Symmetric linear variational inequalities, Projection and contraction method, Self-adaptive rule.

1. INTRODUCTION

Let $\Omega$ be a nonempty closed convex subset of $\mathbb{R}^n$, $H$ be an $n \times n$ matrix, and $c \in \mathbb{R}^n$. The linear variational inequality problem, denoted by $\text{LVI}(\Omega, H, c)$, is to find a vector $x^* \in \Omega$ such that

$$\text{LVI}(\Omega, H, c) \quad (x - x^*)^T (Hx^* + c) \geq 0, \quad \forall x \in \Omega. \quad (1)$$

We call $\text{LVI}(\Omega, H, c)$ symmetric when $H$ is a symmetric matrix. Symmetric $\text{LVI}(\Omega, H, c)$ problems have been widely used in many fields [1]. It includes many special classes of problems, such as a system of linear equations (when $\Omega = \mathbb{R}^n$), linear complementarity problems (when $\Omega = \mathbb{R}^n_+$, the nonnegative orthant of $\mathbb{R}^n$), quadratic programs and least square problems with simple bounds (when $\Omega = \{x \in \mathbb{R}^n \mid x_L \leq x \leq x_U\}$), and some other basic problems in optimization (such as trust region problems when $\Omega = \{x \in \mathbb{R}^n \mid \|x\| \leq r\}$).

*Author to whom all correspondence should be addressed.

The research was supported in part by Grant FRG/99-00/II-23 of Hong Kong Baptist University (first author) and the NSFC Grant 19971040 (second author).

0898-1221/01/$ - see front matter © 2001 Elsevier Science Ltd. All rights reserved. Typeset by A4LATEX

PII: S0898-1221(01)00269-3
It is well known [2] that LVI(Ω, H, c) is equivalent to a class of linear projection equations (LPE)

\[ x = P_Ω[x - β(Hx + c)], \quad β > 0, \]

where \( P_Ω[·] \) denotes the projection on the set Ω. A basic property of the projection mapping on the closed convex set Ω is

\[ (z - P_Ω[z])^T(x - P_Ω[z]) \leq 0, \quad ∀z ∈ \mathbb{R}^n, \quad x ∈ Ω. \tag{2} \]

Let \( S \) be the solution set of problem (1) and denote

\[ e(x, β) = x - P_Ω[x - β(Hx + c)], \quad β > 0. \tag{3} \]

There are already a substantial number of algorithms for solving problem (1) (see [3–13] and references therein). In the PC method [9,10], the vector \( e(x, β) \) is used as the search direction. The recursion

\[ x_{k+1} = x_k - γα_k e(x_k, β), \quad β > 0, \tag{4} \]

with

\[ α_k = \frac{\|e(x_k, β)\|^2}{e(x_k, β) \cdot (I + βH)e(x_k, β)} \quad \text{and} \quad γ ∈ (0, 2) \tag{5} \]

generates a sequence \( \{x_k\} \) that satisfies

\[ \text{dist}_G(x_{k+1}, S) \leq \text{dist}_G(x_k, S) - c_0 \|e(x_k, β)\|^2. \]

Here

\[ \text{dist}_G(x, S) = \min \{\|x - x^*\|_G \mid x^* ∈ S\}, \quad G = I + βH, \quad \text{and} \quad c_0 = \frac{γ(2 - γ)}{\|I + βH\|}. \]

The main advantages of this PC method are its simplicity and ability to handle problem (1). Each iteration of this method consists essentially of only two main vector products and two projections of a vector onto Ω. Therefore, this method allows the optimal exploitation of the sparsity of the matrix \( H \) and may thus be effective for large sparse problems. Besides, by exploiting the symmetry of \( H \), the PC method is more efficient for problem (1) than some other methods such as [12].

However, many applications have shown that if the fixed parameter \( β \) is chosen either too small or too large, the solution time of the PC method can increase significantly. Aiming at overcoming this shortfall, in this paper, we replace the constant \( β \) in (4),(5) by a sequence \( \{β_k\} \) and allow the parameter \( β_k \) to vary from iteration to iteration. In particular, we present a self-adaptive rule to adjust the parameter \( β_k \) automatically in each iteration based on previous iterates. The global convergence of this new method is also proved. Our preliminary numerical tests indicate that this new method is efficient and effective, and that its performance depends only slightly on the initial parameter \( β_0 \). So the proposed method is more practical.

In this paper, we assume that the projection on Ω is simple to carry out. Furthermore, it is assumed that the matrix \( H \) is positive semidefinite. Under this assumption, the solution set \( S \) of problem (1) is nonempty.

The rest of this paper is organized as follows. In Section 2, we propose a self-adaptive projection and contraction method for problem (1). The global convergence of the new method is also proved. In Section 3, some preliminary numerical results are reported. These results confirm that our new method is efficient and effective. Finally, some concluding remarks are given in Section 4.

A few words about our notation. A superscript such as in \( x_k \) refers to a special vector and usually denotes an iteration index. For any real vector \( v \), we denote its transpose by \( v^T \).
Euclidean norm and the max-norm will be denoted by $\| \cdot \|$ and $\| \cdot \|_\infty$, respectively. Given a positive definite matrix $G$, $\| v \|_G$ denotes $(v^T G v)^{1/2}$.

2. A SELF-ADAPTIVE PROJECTION AND CONTRACTION METHOD

In the projection and contraction (PC) method of [9,10], the parameter $\beta$ is fixed. The main difference between the proposing method and the PC method is that we introduce a parameter sequence $\{\beta_k\}$ in (4),(5) instead of a fixed $\beta$, and that we allow $\beta_k$ to vary in an acceptable region at each iteration. We further construct a self-adaptive rule for adjusting the parameter $\beta_k$ automatically. The global convergence of the proposed method is also proved in this section. Now we introduce our method as follows.

A Projection and Contraction Method with Variable Parameters

STEP 0. Given $\epsilon > 0$, $\gamma \in (0,2)$, $x^0 \in \mathbb{R}^n$, $\beta_0 > 0$ and a nonnegative sequence $\{\tau_k\}$ satisfying $\sum_{i=0}^{\infty} \tau_i < \infty$. Set $k := 0$.

STEP 1. If $\| e(x_k, \beta_k) \|_\infty < \epsilon$, then stop.

STEP 2. Compute

$$ x_{k+1} = x_k - \gamma \alpha_k \beta_k (x_k, \beta_k) $$

with

$$ \alpha_k = \frac{\| e(x_k, \beta_k) \|^2}{e(x_k, \beta_k)(1 + \beta_k \gamma) e(x_k, \beta_k)} . $$

STEP 3. Update parameter $\beta_{k+1}$ such that

$$ \frac{1}{1 + \tau_k} \beta_k \leq \beta_{k+1} \leq (1 + \tau_k) \beta_k . $$

Set $k := k + 1$ and go to Step 1.

REMARK 1. It follows from $\tau_k \geq 0$ and $\sum_{i=0}^{\infty} \tau_i < \infty$ that $\prod_{i=0}^{\infty} (1 + \tau_i) < \infty$. Let

$$ C_p := \prod_{i=0}^{\infty} (1 + \tau_i) \quad \text{and} \quad C_\gamma := \sum_{i=0}^{\infty} \tau_i . $$

Then, due to the update scheme in Step 3, the sequence $\{\beta_k\} \subset [\beta_0/C_p, C_p \beta_0]$ is bounded. That is to say,

$$ B_L := \inf \{ \beta_k \}^\infty_0 > 0 \quad \text{and} \quad B_U := \sup \{ \beta_k \}^\infty_0 < \infty . $$

Note that $\| e(x, \beta) \| \geq \| e(x, \beta') \|$ for any $\beta > \beta'$ (see [13]). Then we have

$$ \| e(x_k, B_L) \| \leq \| e(x_k, \beta_k) \| \leq \| e(x_k, B_U) \| , $$

which also justifies our stopping criterion.

The following lemma plays an important role in both the convergence analysis of the proposed method and the construction of the self-adaptive rule. Although a similar result can be found in [10], we detail its proof here for the sake of completeness.

LEMMA 1. Let $x^* \in S$. Then we have

$$ (x - x^*)^T (I + \beta H) e(x, \beta) \geq \| e(x, \beta) \|^2, \quad \forall x \in \mathbb{R}^n, \quad \beta > 0 . $$

PROOF. By setting $z := x - \beta (Hx + c)$ and $x := x^*$ in (2), we obtain

$$ \{ e(x, \beta) - \beta (Hx + c) \}^T (P_0 [x - \beta (Hx + c)] - x^*) \geq 0 . $$

(13)
Since \( P_h[\cdot] \in \Omega \), it follows from (1) that
\[
\beta (Hx^* + c)^T \{ P_h[x - \beta (Hx + c)] - x^* \} \geq 0.
\] (14)

Adding (13) and (14), we get
\[
\{ e(x, \beta) - \beta H (x - x^*) \}^T \{ (x - x^*) - e(x, \beta) \} \geq 0.
\]

Note that \( H \) is positive semidefinite. Then the above inequality implies
\[
(x - x^*)^T (I + \beta H)e(x, \beta) \geq \| e(x, \beta) \|^2 + (x - x^*)^T \beta H (x - x^*) \geq \| e(x, \beta) \|^2.
\]

Hence, the assertion of this lemma is proved. \( \square \)

Now we prove the global convergence of our proposed method.

**Theorem 1.** Let \( \{ x_k \} \) be the sequence generated by our proposed method. Then we have

(i) \( \| x^{k+1} - x^* \|^2_{(I + \beta_k H)} \leq (1 + \tau_k) \| x^k - x^* \|^2_{(I + \beta_k H)} - (\gamma (2 - \gamma)/\| I + B_U H \|) \| e(x_k, B_L) \|^2 \),

\( \forall x^* \in S \), where \( B_U \) and \( B_L \) are defined in (10).

(ii) \( \{ x_k \} \) converges to a solution point of problem (1).

**Proof.** By a simple calculation, we obtain
\[
\left\| x^{k+1} - x^* \right\|^2_{(I + \beta_k H)} = \left\| (x^{k+1} - x^*) - \gamma \alpha_k e(x^k, \beta_k) \right\|^2_{(I + \beta_k H)}
\]
\[
= \left\| x^k - x^* \right\|^2_{(I + \beta_k H)} - 2 \gamma \alpha_k \left\| e(x^k, \beta_k) \right\|^2 + \gamma^2 \alpha_k^2 \left\| e(x^k, \beta_k) \right\|^2.
\] (15)

Then it follows from (7) and (12), we have
\[
\left\| x^{k+1} - x^* \right\|^2_{(I + \beta_k H)} \leq \left\| x^k - x^* \right\|^2_{(I + \beta_k H)} - 2 \gamma \alpha_k \left\| e(x^k, \beta_k) \right\|^2 + \gamma^2 \alpha_k \left\| e(x^k, \beta_k) \right\|^2.
\]

On the other hand, it follows from (8) that
\[
\left\| x^{k+1} - x^* \right\|^2_{(I + \beta_k H)} = (x^{k+1} - x^*)^T (I + \beta_k + H) (x^{k+1} - x^*) \leq (1 + \tau_k) \left\| x^{k+1} - x^* \right\|^2_{(I + \beta_k H)},
\] (16)

where \( \tau_k \) is nonnegative. Combining (15) and (16), we obtain
\[
\left\| x^{k+1} - x^* \right\|^2_{(I + \beta_k H)} \leq (1 + \tau_k) \left\| x^k - x^* \right\|^2_{(I + \beta_k H)} - \gamma (2 - \gamma) \alpha_k \left\| e(x^k, \beta_k) \right\|^2.
\]

From (7) and (10), we get
\[
\alpha_k \geq \frac{1}{\| I + \beta_k H \|} \geq \frac{1}{\| I + B_U H \|}.
\]

Then from (11), we further have
\[
\left\| x^{k+1} - x^* \right\|^2_{(I + \beta_k H)} \leq (1 + \tau_k) \left\| x^k - x^* \right\|^2_{(I + \beta_k H)} - \frac{\gamma (2 - \gamma)}{\| I + B_U H \|} \left\| e(x^k, \beta_k) \right\|^2.
\] (17)

Therefore, Part (i) holds for any \( x^* \in S \).
Now we prove Part (ii). Given any \( x^* \in S \), it follows from (17) and (9) that

\[
\|x^k - x^*\|_{(I + \beta_k H)}^2 \leq \prod_{i=0}^{k-1} (1 + \tau_i) \|x^0 - x^*\|_{(I + \beta_0 H)}^2 \leq C_p \|x^0 - x^*\|_{(I + \beta_0 H)}^2.
\]

So there exists a constant \( C > 0 \) such that

\[
\|x^k - x^*\|_{(I + \beta_k H)}^2 \leq C, \quad \forall k \geq 0. \tag{18}
\]

Since the sequence \( \{\beta_k\} \) is bounded, so is \( \{x^k\} \) from (18). Also from (17) and (18), we obtain

\[
\frac{\gamma(2 - \gamma)}{\|I + B_U H\|} \sum_{i=0}^{\infty} \|e(x^k, B_L)\|^2 \leq \sum_{i=0}^{\infty} \tau_i \|x^i - x^*\|_{(I + \beta_i H)}^2 + \|x^0 - x^*\|_{(I + \beta_0 H)}^2 \leq (1 + C_s)C.
\]

Hence,

\[
\lim_{k \to \infty} \|e(x^k, B_L)\| = 0.
\]

Since \( \{x^k\} \) is bounded, there exists a subsequence \( \{x^{k_j}\} \) converging to \( \tilde{x} \). From the continuity of \( e(x, B_L) \), we have

\[
\|e(\tilde{x}, B_L)\| = \lim_{j \to \infty} \|e(x^{k_j}, B_L)\| = 0.
\]

So \( \tilde{x} \in S \). Assume that \( \hat{x} \) is another cluster point of \( \{x^k\} \). Then there exists a \( k_0 \geq 0 \) such that

\[
\|x^{k_0} - \hat{x}\| \leq \frac{1}{2\sqrt{C_p\|I + B_U H\|}} \|\tilde{x} - \hat{x}\|. \tag{19}
\]

Since \( H \) is positive semidefinite, it follows from (17) and (19) that for any \( k > k_0 \),

\[
\begin{align*}
\|x^k - \tilde{x}\| &\leq \|x^k - \hat{x}\|_{(I + \beta_k H)} \\
&\leq \prod_{i=k_0}^{k-1} (1 + \tau_i) \|x^{k_0} - \hat{x}\|_{(I + \beta_{k_0} H)} \leq \sqrt{C_p} \|x^{k_0} - \hat{x}\|_{(I + \beta_{k_0} H)} \\
&\leq \sqrt{C_p\|I + B_U H\|} \|x^{k_0} - \hat{x}\| \leq \frac{1}{2} \|\tilde{x} - \hat{x}\|.
\end{align*}
\]

Thus, we immediately have

\[
\|x^k - \tilde{x}\| \geq \|\tilde{x} - \hat{x}\| - \|x^k - \hat{x}\| \geq \frac{1}{2} \|\tilde{x} - \hat{x}\|, \quad \forall k > k_0.
\]

So we conclude that the sequence \( \{x^k\} \) has exactly one cluster point, i.e., \( \lim_{k \to \infty} x^k = \tilde{x} \in S \). \hfill \blacksquare

Theorem 1 guarantees the global convergence of the proposed method. However, from a numerical point of view, \( \beta_{k+1} \) in Step 3 cannot be chosen arbitrarily from the interval \( [\beta_k/(1 + \tau_k), (1 + \tau_k)\beta_k] \). In practice, to make our method efficient and applicable, we usually develop some self-adaptive rules to adjust \( \beta_{k+1} \) automatically. In the following discussion, we term the proposed method together with a self-adaptive rule as the self-adaptive projection and contraction (SAPC) method. Actually, there are various ways of constructing such a rule. Here we propose one as an example which is given as follows.

**Remark 2.** Note that inequality (12) in Lemma 1 is the foundation of the proposed method. It can be rewritten as

\[
\{(x - x^*) + \beta H (x - x^*)\}^T e(x, \beta) \geq \|e(x, \beta)\|^2.
\]
For the sake of balance, it is natural to expect \( \| x - x^* \| \approx \| \beta H(x - x^*) \| \). Since \( x^* \) is unknown, we cannot reach this objective. Instead, from given \( x^k \) and the new iterate \( x^{k+1} \), we can choose a \( \beta_k \) such that
\[
\| x^{k+1} - x^k \| \approx \| \beta_k H(x^{k+1} - x^k) \|. \tag{20}
\]
Because \( x^{k+1} - x^k \) is a proportion of \( -e(x^k, \beta_k) \) (see (6)), this can be written as
\[
\| e(x^k, \beta_k) \| \approx \| \beta_k H e(x^k, \beta_k) \|. \tag{20}
\]
Such consideration provides some useful information on how to choose a proper multiplicative factor \( \beta_k \). For the current point \( x^k \), if \( \| e(x^k, \beta_k) \| \gg \| \beta_k H e(x^k, \beta_k) \| \), it means that the current \( \beta_k \) is too small and we should increase it. Conversely, we decrease \( \beta_k \) when \( \| e(x^k, \beta_k) \| \ll \| \beta H e(x^k, \beta_k) \| \). This is the basic idea in our self-adaptive rule.

Strategy of a Self-Adaptive Rule
\[
\beta_{k+1} = \begin{cases} 
(1 + \tau_k)\beta_k, & \text{if } \mu \| \beta_k H e(x^k, \beta_k) \| < \| e(x^k, \beta_k) \|, \\
\beta_k, & \text{otherwise}, \\
\end{cases} \tag{21}
\]
where \( \mu > 1 \).

It should be noted that our self-adaptive rule can be easily implemented.

3. NUMERICAL IMPLEMENTATION

In this section, we present the results of some numerical experiments. Our main interest is in showing the necessity and effectiveness of the proposed method in this paper.

For this purpose, we consider those problems which are formed in the similar way as in [7]. The symmetric positive semidefinite matrix \( H \) is defined as \( H = A^T A \), where \( A \) is an \( n \times n \) matrix whose entries are randomly generated from a uniform distribution in the interval \((-500, +500)\) as in [7]. We test problems with sizes \( n \) ranging from 100 to 1000.

To compare the numerical performance between the PC method and the SAPC method, we use the same convergence criterion in all our tests as
\[
\| e(x,1) \|_\infty \leq 10^{-6}.
\]
The starting point is a random vector, whose components are randomly generated in the interval \((0, 10)\), and we let \( \gamma = 1.8 \). For the SAPC method, we use
\[
\beta_{k+1} = \begin{cases} 
2\beta_k, & \text{if } 4 \| \beta_k H e(x^k, \beta_k) \| \leq \| e(x^k, \beta_k) \| \text{ and } k \leq k_{\text{max}}, \\
\frac{\beta_k}{2}, & \text{if } 4 \| \beta_k H e(x^k, \beta_k) \| \leq 4 \| e(x^k, \beta_k) \| \text{ and } k \leq k_{\text{max}}, \\
\beta_k, & \text{otherwise}, \\
\end{cases} \tag{22}
\]
as its self-adaptive rule in Step 3. In fact, it is easy to check that (22) is a special case of (21). All codes are written in Matlab and run on a P-III 600 personal computer.

Our numerical results are summarized in Tables 1–3. The results in Table 1 illustrate that solution times of the PC method depend significantly on the initial parameter \( \beta_0 \), while the performance of the SAPC method is fairly stable. Although the PC method did work well when \( \beta_0 \) equals \( 10^{-3} \), yet the selection of a proper initial parameter is still unresolved. So it is necessary for us to introduce the self-adaptive technique for the original method. The results in Table 2 show that the SAPC method is independent of \( k_{\text{max}} \). After several iterations, the SAPC method should generate a relatively suitable parameter \( \beta_k \), though this \( \beta_k \) might not be the best.
Table 1. Iteration numbers and computation times of the PC method and the SAPC method with different $\beta_0$ ($n = 200, k_{\text{max}} = 100$).

<table>
<thead>
<tr>
<th>$\beta_0$</th>
<th>PC Method</th>
<th>SAPC Method</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>No. of Iterations</td>
<td>CPU (in Sec.)</td>
</tr>
<tr>
<td>$10^{-10}$</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>$10^{-7}$</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>&gt;10000</td>
<td>&gt;247.00</td>
</tr>
<tr>
<td>$10^{-5}$</td>
<td>7585</td>
<td>199.76</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>762</td>
<td>15.99</td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>347</td>
<td>7.25</td>
</tr>
<tr>
<td>$10^{-2}$</td>
<td>475</td>
<td>9.94</td>
</tr>
<tr>
<td>0.1</td>
<td>4108</td>
<td>86.07</td>
</tr>
<tr>
<td>1</td>
<td>&gt;10000</td>
<td>&gt;206.52</td>
</tr>
<tr>
<td>10</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>$10^{4}$</td>
<td>---</td>
<td>---</td>
</tr>
</tbody>
</table>

'---' means the iteration number > 10000 and CPU time > 300 sec.

Table 2. Numerical results of the SAPC method with different $k_{\text{max}}$ ($n = 200, \beta_0 = 1$).

<table>
<thead>
<tr>
<th>$k_{\text{max}}$</th>
<th>5</th>
<th>7</th>
<th>10</th>
<th>50</th>
<th>100</th>
<th>200</th>
<th>400</th>
</tr>
</thead>
<tbody>
<tr>
<td>No. of Iterations</td>
<td>720</td>
<td>349</td>
<td>357</td>
<td>357</td>
<td>357</td>
<td>357</td>
<td>357</td>
</tr>
<tr>
<td>CPU (in Sec.)</td>
<td>15.16</td>
<td>7.36</td>
<td>7.53</td>
<td>7.52</td>
<td>7.52</td>
<td>7.58</td>
<td>7.63</td>
</tr>
</tbody>
</table>

Table 3. Iteration numbers and computation times of the SAPC method with different sizes ($\beta_0 = 1, k_{\text{max}} = 20$).

<table>
<thead>
<tr>
<th>$n$</th>
<th>100</th>
<th>150</th>
<th>200</th>
<th>250</th>
<th>300</th>
<th>400</th>
<th>500</th>
<th>700</th>
<th>1000</th>
</tr>
</thead>
<tbody>
<tr>
<td>No. of Iterations</td>
<td>296</td>
<td>332</td>
<td>323</td>
<td>331</td>
<td>334</td>
<td>255</td>
<td>334</td>
<td>275</td>
<td>359</td>
</tr>
<tr>
<td>CPU (in Sec.)</td>
<td>0.83</td>
<td>3.51</td>
<td>6.92</td>
<td>11.15</td>
<td>16.42</td>
<td>22.58</td>
<td>46.19</td>
<td>73.65</td>
<td>193.95</td>
</tr>
</tbody>
</table>

The results in Table 3 indicate that the number of iterations of the SAPC method is relatively independent of problem sizes. The results in the above three tables clearly indicate that our SAPC method is quite stable and efficient.

5. CONCLUDING REMARKS

In this paper, we propose a self-adaptive projection and contraction (SAPC) method for solving problem (1). The SAPC method successfully overcomes the difficulty of estimating the initial parameter in the PC method. The global convergence of the SAPC method is also proved. Our preliminary numerical tests clearly indicate that the proposed SAPC method is quite stable and efficient in practice. In brief, our work in this paper is practically necessary and can be viewed as a meaningful and practical extension of the PC method in [9,10].

REFERENCES