Performance Analysis of a Special GPU Method for Singular Saddle Point Problems

Jae Heon Yun, Member, IAENG

Abstract—In this paper, we first provide semi-convergence analysis for a special GPU (Generalized Parameterized Inexact Uzawa) method with singular preconditioners for solving singular saddle point problems. We next provide a methodology of how to choose nearly quasi-optimal parameters of the special GPU method. Lastly, numerical experiments are carried out to examine the effectiveness of the special GPU method with singular preconditioners by comparing its performance with that of other existing iterative methods for solving singular saddle point problems.

Index Terms—singular saddle point problem, GPU method, semi-convergence, singular splitting, Moore-Penrose inverse.

I. INTRODUCTION

We consider the following large sparse augmented linear system

\[
\begin{pmatrix}
A & B \\
-B^T & 0
\end{pmatrix}
\begin{pmatrix}
x \\
y
\end{pmatrix} =
\begin{pmatrix}
f \\
g
\end{pmatrix},
\]

(1)

where \(A \in \mathbb{R}^{m \times n}\) is a symmetric positive definite matrix, and \(B \in \mathbb{R}^{m \times n}\) is a rank-deficient matrix with \(m \geq n\). In this case, the coefficient matrix of (1) is singular and so the problem (1) is called a singular saddle point problem. This type of problem appears in many different scientific applications, such as constrained optimization problems [13], [18], [30], the finite element approximation for solving the Navier-Stokes equation [10], the constrained least squares parameter estimation problem [11], and MIAOR (Modified inexact AOR) method [22].

In case of \(B\) being of full rank, many relaxation iterative methods have been proposed for solving the augmented linear system (1), e.g., SOR-like method [11], GSOR (Generalized SOR) method [2], PIU (Parameterized Inexact Uzawa) method [3], GPU (Generalized Parameterized Inexact Uzawa) method [7], the modified SOR-like method [16], SSOR-like method [8], the modified SSOR-like method [17], Uzawa-SAOR method [21], GSSOR (Generalized SSOR) method [25], and MIAOR (Modified inexact AOR) method [22].


The purpose of this paper is to provide performance analysis of a special case of the GPU method with singular preconditioners for solving the singular saddle point problem (1). This paper is organized as follows. In Section 2, we provide preliminary results for semi-convergence analysis of the basic iterative methods. In Section 3, we provide semi-convergence results for a special case of the GPU method with singular preconditioners. In Section 4, we first provide a methodology of how to choose nearly quasi-optimal parameters of the special GPU method, and then we provide numerical experiments in order to examine the effectiveness of the special GPU method with singular preconditioners. Lastly, some conclusions are drawn.

II. PRELIMINARIES FOR SEMI-CONVERGENCE ANALYSIS

For simplicity of exposition, some notation and definitions are presented. For a vector \(x\), \(x^*\) denotes the complex conjugate transpose of the vector \(x\). \(\lambda_{\min}(H)\) and \(\lambda_{\max}(H)\) denote the minimum and maximum eigenvalues of the Hermitian matrix \(H\), respectively. For a square matrix \(G\), \(R(G)\) denotes the range space of \(G\), \(N(G)\) denotes the null space of \(G\), \(\sigma(G)\) denotes the set of all eigenvalues of \(G\), and \(\rho(G)\) denotes the spectral radius of \(G\).

Let us recall some useful results on iterative methods for solving singular linear systems based on matrix splitting. For a matrix \(E \in \mathbb{R}^{n \times n}\), the smallest nonnegative integer \(k\) such that \(\text{rank}(E^k) = \text{rank}(E^{k+1})\) is called the index of \(E\), and denoted by \(k = \text{index}(E)\). In other words, \(\text{index}(E)\) is the size of the largest Jordan block corresponding to the zero eigenvalue of \(E\). For a square matrix \(T\), the pseudo-spectral radius \(\nu(T)\) is defined by

\[
\nu(T) = \max\{|\lambda| \mid \lambda \in \sigma(T) - \{1\}\}
\]

where \(\sigma(T)\) is the set of eigenvalues of \(T\).

The Moore-Penrose inverse [4] of a singular matrix \(E \in \mathbb{R}^{n \times n}\) is defined by the unique matrix \(E^+\) which satisfies the
following equations
\[ E = EE^T E, \quad E^T = E^T EE^T, \quad (EE^T)^T = EE^T, \quad (E^T E)^T = E^T E. \]

Let \( A = M - N \) be a splitting of a singular matrix \( A \), where \( M \) is singular. Then an iterative method corresponding to this singular splitting for solving a singular linear system \( Ax = b \) is given by
\[ x_{i+1} = (I - M^T A)x_i + M^T b \text{ for } i = 0, 1, \ldots \]  
\[ (2) \]

Definition 2.1: The iterative method (2) is semi-convergent if for any initial guess \( x_0 \), the iteration sequence \( \{x_i\} \) produced by (2) converges to a solution \( x_\star \) of the singular linear system \( Ax = b \).

Notice that a matrix \( A \) is singular. Then an iterative method corresponding to this singular splitting for solving a singular linear system \( Ax = b \) is given by
\[ x_{k+1} = (I - M^T A)x_k + M^T b \text{ for } k = 0, 1, \ldots \]  
\[ (2) \]

Similarly, let the regularization parameter \( \tau > 0 \) and choose the singular value decomposition of the operator \( B^T M^{-1} B \), where \( B \) and \( V \) are unitary matrices, \( \Sigma = \text{diag}(\sigma_1, \sigma_2, \ldots, \sigma_r) \) and \( \sigma_i \)'s are positive singular values of \( B \). Let \( W = (W_1, W_2) \) be partitioned into \( W = (W_1, W_2) = (V_1, V_2) \) with \( W_1 \in \mathbb{C}^{m \times r} \), \( W_2 \in \mathbb{C}^{m \times (n-r)} \), \( V_1 \in \mathbb{C}^{n \times r} \), \( V_2 \in \mathbb{C}^{n \times (n-r)} \), respectively. Let us define an \((m+n)\times(m+n)\) unitary matrix \( \mathcal{P} \) as
\[ \mathcal{P} = \begin{pmatrix} W & 0 \\ 0 & V \end{pmatrix}. \]  
\[ (3) \]

Algorithm 1: SGPIU method with singular \( Q \)

Choose \( s \) and initial vectors \( x_0, y_0 \)

For \( k = 0, 1, \ldots \), until convergence
\[ x_{k+1} = x_k + P^{-1}(f - Az_k - B y_k) \]  
\[ y_{k+1} = y_k + Q^T(B^T x_{k+1} - g - s B^T(x_{k+1} - x_k)) \]  
\[ y_k + Q^T(B^T((1 - s)x_{k+1} + s x_k) - g) \]

End For

If \( s = 0, P \) is replaced by \( \frac{1}{\tau} P \) with \( \tau \in (0,2) \) and \( Q \) is replaced by \( \frac{1}{\tau} Q \), then the SGPIU method reduces to the PIU method.

Assume that the rank of \( B \) is \( r \), i.e., \( r = \text{rank}(B) < n \). Let
\[ \tilde{H}(s) = \mathcal{P}^* H_4(s) \mathcal{P}. \]  
\[ (4) \]

where \( \tilde{H}(s) = \Sigma_r W_1^T M^{-1} W_1 \) is an \( r \times r \) SPD matrix. Thus
\[ \tilde{Q}^T = \tilde{V}^* \tilde{Q} V \]  
\[ (5) \]

and
\[ \tilde{H}_4(s) = \begin{pmatrix} I_m & -\tilde{P}^{-1} A \\ \tilde{Q}^T \Sigma_r^T & I_n + (s-1)\tilde{Q}^T \Sigma_r \tilde{Q} \end{pmatrix} \]  
\[ (6) \]

If we let \( B_1 = \begin{pmatrix} \Sigma_r & 0 \\ 0 & 0 \end{pmatrix} \in \mathbb{R}^{m \times r} \), then using (10) to (12)
\[ \tilde{H}_4(s) = \begin{pmatrix} I_m & -\tilde{P}^{-1} A \\ \tilde{Q}^T \Sigma_r^T & I_n + (s-1)\tilde{Q}^T \Sigma_r \tilde{Q} \end{pmatrix} \]  
\[ (7) \]

then \( \tilde{H}(s) \) is the iteration matrix of the SGPIU method applied to the following nonsingular saddle point problem
\[ \left( \begin{array}{cc} \tilde{A} & B_1 \\ -B_1^T & 0 \end{array} \right) \begin{pmatrix} x \\ z \end{pmatrix} = \begin{pmatrix} \tilde{f} \\ -\tilde{g} \end{pmatrix} \]  
\[ (8) \]
Before proceeding to semi-convergence result analysis of the SGPIU method for solving the singular saddle point problem (1), we first consider convergence of the SGPIU method for solving the nonsingular saddle point problem (15) whose iteration matrix is $H(s)$. Note that $B_1$ has full column rank $r$. From the convergence analysis described in [7], [27], one can easily obtain the following three lemmas.

**Lemma 3.1:** Let $\lambda$ be an eigenvalue of $H(s)$ and $(\hat{x}, \hat{z})$ be the corresponding eigenvector. Then $\lambda \neq 1$ and $\hat{x} \neq 0$.

**Lemma 3.2:** Let $\lambda$ be an eigenvalue of $H(s)$ and $(\hat{x}, \hat{z})$ be the corresponding eigenvector. Then $\lambda$ satisfies the following quadratic equation

$$\lambda^2 + \frac{\hat{\beta} - 2\alpha + (1 - s)s}{\alpha} \lambda + \frac{\hat{\alpha} - \hat{\beta} + s\gamma}{\alpha} = 0,$$

where $\hat{\alpha} = \frac{x^T P x}{x^T x}$, $\hat{\beta} = \frac{x^T A z}{x^T x}$ and $\hat{\gamma} = \frac{x^T B_1 Q_1^{-1} B_1^T \hat{x}}{x^T x}$.

**Lemma 3.3:** Let $\lambda$ be an eigenvalue of $H(s)$ and $(\hat{x}, \hat{z})$ be the corresponding eigenvector. Then $|\lambda| < 1$ if and only if

$$\gamma - 4\alpha + 2\beta < 2s\gamma < 2\beta$$

and $\gamma > 0$,

where $\alpha = \frac{x^T P x}{x^T x}$, $\beta = \frac{x^T A z}{x^T x}$ and $\gamma = \frac{x^T B_1 Q_1^{-1} B_1^T \hat{x}}{x^T x}$.

**Theorem 3.4:** Let $\lambda \neq 1$ be an eigenvalue of $H_4(s)$ and $(\hat{x}, \hat{y})$ be the corresponding eigenvector. Then $|\lambda| < 1$ if and only if

$$\gamma - 4\alpha + 2\beta < 2s\gamma < 2\beta$$

and $\gamma > 0$.

Proof: Since $H_4(s)$ and $H_4(s)$ are similar, $\lambda$ is an eigenvalue of $H_4(s)$. Since $\lambda \neq 1$, from (13) and (14) $\lambda$ is also an eigenvalue of $H(s)$. Let $(\hat{x}, \hat{y})$ be an eigenvector of $H_4(s)$ corresponding to the $\lambda$. Then $(\hat{x}, \hat{y})$ is an eigenvector of $H(s)$ corresponding to the $\lambda$, where $\hat{y}$ is the subvector consisting of the first $r$ components $y_i$ from the relation $H_4(s) = P^T H_4(s) P$. Since $H_4(s)$ is symmetric positive definite, let $\lambda = \frac{\lambda_{\min}}{\rho(B_1 Q_1^{-1} B_1^T)}$.

From (16), $\hat{\gamma} = \gamma$ is also obtained. Therefore, the proof is complete.

**Corollary 3.5:** Let $\lambda$ be an eigenvalue of $H(s)$ and $(\hat{x}, \hat{y})$ be an eigenvector of $H_4(s)$ corresponding to the eigenvalue $\lambda$. Then $|\lambda| < 1$ if and only if

$$\gamma - 4\alpha + 2\beta < 2s\gamma < 2\beta$$

and $\gamma > 0$.

Proof: Since $\lambda \neq 1$ from Lemma 3.1, this corollary follows from Theorem 3.4.

The following theorem shows that the condition $\gamma > 0$ in (17) can be omitted.

**Theorem 3.6:** Let $\lambda$ be an eigenvalue of $H(s)$ and $(\hat{x}, \hat{y})$ be an eigenvector of $H_4(s)$ corresponding to the eigenvalue $\lambda$. Then $|\lambda| < 1$ if and only if

$$\gamma - 4\alpha + 2\beta < 2s\gamma < 2\beta,$$

where $\alpha = \frac{x^T P x}{x^T x}$, $\beta = \frac{x^T A z}{x^T x}$ and $\gamma = \frac{x^T B_1 Q_1^{-1} B_1^T \hat{x}}{x^T x}$.

Proof: From Theorem 3.4, it was shown that $\alpha = \alpha > 0$, $\beta = \beta > 0$ and $\gamma = \gamma > 0$. If $\gamma = 0$ (i.e., $x \in N(B_1^T)$), then Lemma 3.2 implies that $\lambda$ satisfies the following quadratic equation

$$\lambda^2 - (2 - \frac{\beta}{\alpha})\lambda + 1 - \frac{\beta}{\alpha} = 0.$$ 

Hence, this corollary follows from Theorem 3.6 and the inequality (19).

Now we provide semi-convergence result for the SGPIU method for solving the singular saddle point problem (1).

**Theorem 3.7:** Assume that $2P - A$ is symmetric positive definite. Then $\rho(H(s)) < 1$ if the following inequality holds

$$\frac{2}{\rho(B_1 Q_1^{-1} B_1^T)} \lambda_{\min} (2P - A) < s < \frac{\lambda_{\min}(A)}{\rho(B_1 Q_1^{-1} B_1^T)}.$$ 

Proof: By Theorem 3.7, $\rho(H(s)) < 1$. From (13) and (14), it is clear that the matrix $H_4(s)$ is semi-convergent. Since $H_4(s) = P^T H_4(s) P$, $H_4(s)$ is also semi-convergent. Notice that $H_4(s) = I - (D + (s - 1)\mathcal{L})^T A$. From Theorem 2.2, we need to show that $N(A) = N((D + (s - 1)\mathcal{L})^T A)$. Hence, it is sufficient to show that $N((D + (s - 1)\mathcal{L})^T A) \subset N(A)$. Suppose that $(\hat{x}, \hat{y}) \in N((D + (s - 1)\mathcal{L})^T A)$. Then

$$P^{-1}(Ax + By) = 0 \quad \text{and} \quad -Q^T B_1 x + (s - 1)Q^T B_1 P^{-1}(Ax + By) = 0.$$ 

From these equations, $Ax + By = 0$ and $-Q^T B_1 x = 0$. Since $Q^T B_1 = B_1$, one obtains $(\hat{x}, \hat{y}) \in N(A)$. Therefore, the proof is complete.

**Corollary 3.9:** Let $\hat{P}$ be a SPD matrix which approximates $A$, $\hat{P} = \frac{1}{\alpha} P$, $Q = B_1^T M^{-1} B$ and $Q = \frac{1}{\alpha} Q$, where $0 < \omega < 2$, $\tau > 0$ and $M$ is a SPD matrix which approximates $P$. Assume that $2P - A$ is symmetric positive definite. Then the SGPIU method with the singular
Q for solving the singular saddle point problem (1) is semi-convergent if the following inequality holds
\[
\frac{1}{2} - \frac{\lambda_{\min}(\frac{2}{\tau} \tilde{P} - A)}{\rho(BQ^T B^T)} < s < \frac{\lambda_{\min}(A)}{\rho(BQ^T B^T)}.
\]

**Corollary 3.10:** Let \( P = \frac{1}{\tau} A, \tilde{Q} = B^T M^{-1} B \) and \( Q = \frac{1}{\tau} Q \), where \( 0 < \omega < 2, \tau > 0 \) and \( M \) is a SPD matrix which approximates \( P \). Then the SGPIU method with the singular \( Q \) for solving the singular saddle point problem (1) is semi-convergent if the following inequality holds
\[
\frac{1}{2} - \left( 2 - \frac{1}{\omega} \right) \frac{\lambda_{\min}(A)}{\rho(BQ^T B^T)} < s < \frac{\lambda_{\min}(A)}{\rho(BQ^T B^T)}.
\]

**Proof:** Since \( 0 < \omega < 2, 2 P - A \) is symmetric positive definite. Hence this corollary follows from Corollary 3.9. \( \blacksquare \)

## IV. Numerical Results

In this section, we first provide a methodology of how to choose nearly quasi-optimal parameters of the special GPU method, and then we provide numerical experiments in order to examine the effectiveness of the SGPIU method with singular preconditioners for solving the singular saddle point problem (1). Performance of the SGPIU method with singular preconditioners is compared with that of the SGPIU method with nonsingular preconditioners and the PU or PIU methods with singular or nonsingular preconditioners.

In Tables II to V, \( \text{Iter} \) denotes the number of iteration steps, and \( \text{CPU} \) denotes the elapsed CPU time in seconds excluding the computational time of \( Q^T Q \) for the singular case of \( Q \) or the Cholesky factorization time of \( Q \) for the nonsingular case of \( Q \). In all experiments, the right hand side vector \((f^T, -g^T)^T \in \mathbb{R}^{m+n}\) was chosen such that the exact solution of the saddle point problem (1) is \((x^T, y^T)^T = (1, 1, \ldots, 1)^T \in \mathbb{R}^{m+n}\), and the initial vector was set to the zero vector. All iterations for the singular saddle point problem are terminated if the current iteration satisfies \( \text{RES} < 10^{-6} \), where \( \text{RES} \) is defined by
\[
\text{RES} = \frac{\sqrt{\|f - Ax_k - B y_k\|^2 + \|g - B^T x_k\|^2}}{\sqrt{\|f\|^2 + \|g\|^2}},
\]
where \( \| \cdot \| \) denotes the \( L_2 \)-norm.

All numerical tests are carried out on a PC equipped with Intel Core i5-4570 3.2GHz CPU and 8GB RAM using MATLAB R2014b. For the elapsed CPU time, every experiment is repeated five times. The best and the worst ones out of 5 CPU times are discarded, and then the average of the remaining 3 CPU times is reported in Tables II to V.

**Example 4.1 (28)**: We consider the saddle point problem (1), in which
\[
A = \begin{pmatrix} I \otimes T + T \otimes I & 0 \\ I \otimes T + T \otimes I & 0 \end{pmatrix} \in \mathbb{R}^{2p^2 \times 2p^2},
\]
\[
B = \begin{pmatrix} \tilde{B} & \tilde{B} \end{pmatrix} = \begin{pmatrix} B_1 & b_1 \\ B_2 & b_2 \end{pmatrix} \in \mathbb{R}^{2p^2 \times (p^2 + 2)},
\]
\[
\tilde{B} = \begin{pmatrix} I \otimes F \\ P \otimes I \end{pmatrix} \in \mathbb{R}^{2p^2 \times p^2}, \quad b_1 = \tilde{B} \begin{pmatrix} c_{p^2/2} \\ 0 \end{pmatrix},
\]
\[
b_2 = \tilde{B} \begin{pmatrix} 0 \\ c_{p^2/2} \end{pmatrix} = (1, 1, \ldots, 1)^T \in \mathbb{R}^{p^2/2},
\]
\[
T = \frac{1}{h^2} \cdot \text{tridiag}(-1, 2, -1) \in \mathbb{R}^{p \times p},
\]
\[
F = \frac{1}{h} \cdot \text{tridiag}(-1, 1, 0) \in \mathbb{R}^{p \times p},
\]
with \( \otimes \) denoting the Kronecker product and \( h = \frac{1}{p^2} \) the discretization mesh size. For this example, \( m = 2p^2 \) and \( n = p^2 + 2 \). Thus the total number of variables is \( 3p^2 + 2 \). Numerical results for this example are listed in Tables II and III.

**Example 4.2:** Consider the Stokes equations of the following form: find \( u \) and \( v \) such that
\[
\begin{align*}
- \Delta u + \nabla w &= f \quad \text{in } \Omega, \\
- \nabla \cdot u &= 0 \quad \text{in } \Omega,'
\end{align*}
\]
where \( \Omega = (0, 1) \times (0, 1) \), \( u \) is a vector-valued function representing the velocity, and \( w \) is a scalar function representing the pressure. The boundary conditions are \( u = (0, 0)^T \) on the three fixed walls (\( x = 0, y = 0, x = 1 \)) and \( u = (1, 0)^T \) on the moving wall (\( y = 1 \)). Dividing \( \Omega \) into a uniform grid with mesh size \( h = \frac{1}{p} \) and discretizing (20) by using MAC (marker and cell) finite difference scheme [9], [12], the singular saddle point problem (1) is obtained, where \( A \in \mathbb{R}^{2p(p-1) \times 2p(p-1)} \) is a symmetric positive definite matrix and \( B = (\tilde{B} \tilde{B}) = \begin{pmatrix} 2(q^2-1) \quad -q^2 \end{pmatrix} \) is a rank-deficient matrix of \( \text{rank}(B) = p^2 - 1 \) with \( \tilde{B} \in \mathbb{R}^{2p(p-1) \times (p^2-1)} \) and \( \tilde{B} \in \mathbb{R}^{2p(p-1)}. \) For this example, \( m = 2p(p-1) \) and \( n = p^2. \) Thus the total number of variables is \( 3p^2 - 2p. \) Numerical results for this example are listed in Tables IV and V.

For the SGPIU method, the symmetric positive definite matrices \( P \) are chosen as \( P = \frac{1}{\tau} \hat{P} \) with a positive parameter \( \omega \in (0, 2) \) in three different ways. The first choice is \( \hat{P} = A \), the second choice is \( \hat{P} = (E - F)E^{-1}(E - F)^T \), where \( A = E - F - F^T \) is a splitting of the symmetric positive definite matrix \( A \) with \( E \) a diagonal matrix and \( F \) a strictly lower triangular matrix, and the third choice is \( \hat{P} = L_0 \tilde{P} \), where \( A = L_0 \tilde{P} - R_0 \) is a splitting of the symmetric matrix \( A \) obtained by an incomplete Cholesky factorization of \( A \) with no fill-in. The singular or nonsingular preconditioning matrices \( Q \) are chosen as \( Q = \frac{1}{\tau} \hat{Q} \) with a positive parameter \( \tau \), where the matrices \( Q \) are chosen as in Table I. In Table I, \( \text{Diag}(B^T A^{-1} B, B^T B) \) denotes a block diagonal matrix consisting of two submatrices \( B^T A^{-1} B \) and \( B^T B \). The SGPIU algorithm for the nonsingular case of \( Q \) is the same as that for the singular case of \( Q \) except that \( Q^{-1} \) is used instead of \( Q^T \).

For these choices of \( P \) and \( Q \), the SGPIU method with \( s = 0 \) reduces to the PU method for \( P = \frac{1}{\tau} A \) or the PIU method for other two choices of \( P \). For \( s = 0 \), the parameters \( \omega \) and \( \tau \) are chosen as the optimal or quasi-optimal parameters which are computed using the formulas given in [28] or [19], respectively (see data reported in the first line of Tables II - V for each case of \( P \)). For \( s \neq 0 \), the parameters are chosen in two different ways: One choice is that \( \omega \) and \( \tau \) are chosen first as the optimal or quasi-optimal parameters and then \( s \) is chosen as the best one by tries (see data reported in the second line of Tables II - V for each case of \( P \)), and the other choice is the experimentally chosen optimal parameters \( s, \omega \) and \( \tau \) (see data reported in the third line of Tables II - V for each case of \( P \)). For singular matrix \( Q, Q^T \) is computed only once using the Matlab function \texttt{pinv} with a drop tolerance \( 10^{-13} \), and then it is stored for later use. For nonsingular matrix \( Q \), the Cholesky factorization of \( Q \) is computed only.

(Advance online publication: 23 August 2017)
SGPIU method with an appropriately chosen number $s$ of $PQ$ vector operation after constructing $Q$

$\text{tridiag} \left( \text{Diag} (\hat{B}^T \tilde{A}^{-1} \hat{B}, \hat{B}^T \tilde{B}) \right)$ $\hat{A} = \text{tridiag}(A)$ nonsingular

TABLE II

| Performance of SGPIU method with $P = \frac{1}{2} P$ and singular $Q = \frac{1}{2} \hat{Q}$ for Example 4.1 |

<table>
<thead>
<tr>
<th>$m$</th>
<th>$n$</th>
<th>$\hat{P}$</th>
<th>Case I of $Q$</th>
<th>Case II of $Q$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>$s$</td>
<td>$\omega$</td>
</tr>
<tr>
<td>1152</td>
<td>578</td>
<td>$A$</td>
<td>0.002</td>
<td>0.2488</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-0.04</td>
<td>0.2488</td>
<td>0.1423</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.04</td>
<td>1.7657</td>
<td>0.0626</td>
</tr>
<tr>
<td>2048</td>
<td>1026</td>
<td>$\left(E - F\right) E^{-1} \left(E - F\right)^T$</td>
<td>0.1325</td>
<td>0.0910</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$\tilde{L}_0 \hat{L}_0^T$</td>
<td>-0.25</td>
<td>1.3256</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.15</td>
<td>1.7657</td>
<td>0.0626</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-0.15</td>
<td>1.65</td>
<td>0.12</td>
</tr>
</tbody>
</table>

TABLE III

| Performance of SGPIU method with $P = \frac{1}{2} P$ and nonsingular $Q = \frac{1}{2} \hat{Q}$ for Example 4.1 |

<table>
<thead>
<tr>
<th>$m$</th>
<th>$n$</th>
<th>$\hat{P}$</th>
<th>Case III of $Q$</th>
<th>Case IV of $Q$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>$s$</td>
<td>$\omega$</td>
</tr>
<tr>
<td>1152</td>
<td>578</td>
<td>$A$</td>
<td>0.002</td>
<td>0.2489</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-0.04</td>
<td>0.2489</td>
<td>0.1423</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.04</td>
<td>1.7657</td>
<td>0.0626</td>
</tr>
<tr>
<td>2048</td>
<td>1026</td>
<td>$\left(E - F\right) E^{-1} \left(E - F\right)^T$</td>
<td>0.1325</td>
<td>0.0910</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$\tilde{L}_0 \hat{L}_0^T$</td>
<td>-0.25</td>
<td>1.3256</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.15</td>
<td>1.7657</td>
<td>0.0626</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-0.15</td>
<td>1.65</td>
<td>0.12</td>
</tr>
</tbody>
</table>

Once using the Matlab function chol, and then it is stored for later use.

For singular $Q$, $Q^T b$ is computed using matrix-times-vector operation after constructing $Q^T$ explicitly, which is very time-consuming. For nonsingular $Q$, $Q^{-1} b$ is computed using the forward and backward substitutions after constructing the Cholesky factorization of $Q$ explicitly. As can be seen in Tables II to V, $P = \frac{1}{2} \tilde{L}_0 \hat{L}_0^T$ provides better performance and faster convergence rate than other two cases of $P$. From Tables II to V, it can be also seen that the SGPIU method with an appropriately chosen number $s$ and optimal or quasi-optimal parameters $\omega$ and $\tau$ performs better than the PU or PIU methods with optimal or quasi-optimal parameters $\omega$ and $\tau$ (i.e., the SGPIU methods with $s = 0$).

More specifically, when $P = \frac{1}{2} A$, SGPIU method with an appropriately chosen number $s$ corresponding to optimal parameters $\omega$ and $\tau$ of the PU method performs significantly better than PU method with the optimal parameters $\omega$ and $\tau$ for all types of preconditioners $Q$ used in this paper. When $P = \frac{1}{2} \left(E - F\right) E^{-1} \left(E - F\right)^T$ or $\frac{1}{2} \tilde{L}_0 \hat{L}_0^T$, SGPIU method with an appropriately chosen number $s$ corresponding to quasi-optimal parameters $\omega$ and $\tau$ of the PIU method performs much better than PIU method with the quasi-optimal parameters $\omega$ and $\tau$ for the preconditioners $Q$ of types I and III. Clearly, the SGPIU method with experimentally chosen optimal parameters $s$, $\omega$ and $\tau$ performs best. However, we do not have a formula for finding optimal parameters of the SGPIU method, which should be done in the future work.
In this paper, we provided semi-convergence analysis of the SGPIU method with singular preconditioners for solving singular saddle point problems. Numerical experiments show that the SGPIU method with an appropriately chosen number \( s \) and optimal or quasi-optimal parameters \( \omega \) and \( \tau \) performs better than the PU or PIU methods with optimal or quasi-optimal parameters \( \omega \) and \( \tau \). More specifically, when \( P = \frac{1}{2}A \), SGPIU method with an appropriately chosen number \( s \) corresponding to optimal parameters \( \omega \) and \( \tau \) of the PU method performs significantly better than the PU method for all types of preconditioners \( Q \) used in this research for finding optimal parameters of the SGPIU method.

The author is grateful to the anonymous reviewers for their valuable comments and suggestions which improved the quality and the clarity of the paper.

**ACKNOWLEDGMENT**

The author is grateful to the anonymous reviewers for their valuable comments and suggestions which improved the quality and the clarity of the paper.

**REFERENCES**


---

**TABLE IV**

<table>
<thead>
<tr>
<th>( m )</th>
<th>( n )</th>
<th>( \hat{P} )</th>
<th>Case I of ( Q )</th>
<th>Case II of ( Q )</th>
<th>Case III of ( Q )</th>
<th>Case IV of ( Q )</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>( s )</td>
<td>( \omega )</td>
<td>Iter</td>
<td>CPU</td>
</tr>
<tr>
<td>1104</td>
<td>576</td>
<td>((E - F)E^{-1}(E - F)^T)</td>
<td>0</td>
<td>1.7489</td>
<td>0.0971</td>
<td>197</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-0.25</td>
<td>1.0304</td>
<td>0.0971</td>
<td>75</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-0.25</td>
<td>1.0304</td>
<td>0.0971</td>
<td>75</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-0.30</td>
<td>0.1416</td>
<td>0.0599</td>
<td>102</td>
</tr>
</tbody>
</table>

**TABLE V**

<table>
<thead>
<tr>
<th>( m )</th>
<th>( n )</th>
<th>( \hat{P} )</th>
<th>Case III of ( Q )</th>
<th>Case IV of ( Q )</th>
<th>Case V of ( Q )</th>
<th>Case VI of ( Q )</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>( s )</td>
<td>( \omega )</td>
<td>Iter</td>
<td>CPU</td>
</tr>
<tr>
<td>1104</td>
<td>576</td>
<td>((E - F)E^{-1}(E - F)^T)</td>
<td>0</td>
<td>1.7489</td>
<td>0.0971</td>
<td>197</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-0.25</td>
<td>1.0304</td>
<td>0.0971</td>
<td>75</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-0.25</td>
<td>1.0304</td>
<td>0.0971</td>
<td>75</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-0.30</td>
<td>0.1416</td>
<td>0.0599</td>
<td>102</td>
</tr>
</tbody>
</table>

---

**V. CONCLUSION**

In this paper, we provided semi-convergence analysis of the SGPIU method with singular preconditioners for solving singular saddle point problems. Numerical experiments show that the SGPIU method with an appropriately chosen number \( s \) and optimal or quasi-optimal parameters \( \omega \) and \( \tau \) performs better than the PU or PIU methods with optimal or quasi-optimal parameters \( \omega \) and \( \tau \). More specifically, when \( P = \frac{1}{2}A \), SGPIU method with an appropriately chosen number \( s \) corresponding to optimal parameters \( \omega \) and \( \tau \) of the PU method performs significantly better than the PU method for all types of preconditioners \( Q \) used in this research for finding optimal parameters of the SGPIU method.


