An alternating positive semidefinite splitting preconditioner for the three-by-three block saddle point problems*

Davod Khojasteh Salkuyeh[1,2,†], Hamed Aslani[3] and Zhao-Zheng Liang[4]

[1] *Faculty of Mathematical Sciences, University of Guilan, Rasht, Iran*
[2] *Center of Excellence for Mathematical Modelling, Optimization and Combinational Computing (MMOCC), University of Guilan, Rasht, Iran*
[3] *Faculty of Mathematical Sciences, University of Guilan, Rasht, Iran*
[4] *School of Mathematics and Statistics, Lanzhou University, Lanzhou, P. R. China*

Received September 22, 2020; accepted January 8, 2021

Abstract. Using the idea of the dimensional splitting method, we present an iteration method for solving three-by-three block saddle point problems which appear in linear programming and finite element discretization of the Maxwell equation. We prove that the method is unconditionally convergent. Then the induced preconditioner is used to accelerate the convergence of the GMRES method for solving the system. Numerical results are presented to compare the performance of the method with some existing ones.

AMS subject classifications: 65F10, 65F50, 65F08

Key words: saddle point, block, dimensional, split, preconditioner, GMRES

1. Introduction

We consider the block three-by-three saddle point problems of the form

$$\mathcal{K}\mathbf{x} \equiv \begin{pmatrix} A & B^T & 0 \\ B & 0 & C^T \\ 0 & C & 0 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} f \\ g \\ h \end{pmatrix}, \tag{1}$$

where $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{m\times n}$ and $C \in \mathbb{R}^{l\times m}$. Here, $f \in \mathbb{R}^n$, $g \in \mathbb{R}^m$ and $h \in \mathbb{R}^l$. In this case, the coefficient matrix of system (1) is of order $\mathbf{n} \times \mathbf{n}$, where $\mathbf{n} = n + m + l$. Systems of the form (1) appear in many applications such as finite element methods for solving time-dependent Maxwell equations with discontinuous coefficients in general three-dimensional Lipschitz polyhedral domains [10], solving quadratic programs [12] and the Picard iteration method for a class of mixed finite element scheme for stationary magnetohydrodynamics models [13].

*The work of Davod Khojasteh Salkuyeh is partially supported by the University of Guilan and the work of Zhao-Zheng Liang is supported by the National Natural Science Foundation of China (No 11801242).

†Corresponding author. *Email addresses:* `khojasteh@guialn.ac.ir` (D. K. Salkuyeh), `hamedaslani525@gmail.com` (H. Aslani), `liangzz@lzu.edu.cn` (Z.-Z. Liang)

System (1) can be equivalently rewritten in the following form:

$$\mathcal{A}\mathbf{x} \equiv \begin{pmatrix} A & B^T & 0 \\ -B & 0 & -C^T \\ 0 & C & 0 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} f \\ -g \\ h \end{pmatrix} = \mathbf{b}. \tag{2}$$

Unlike the coefficient matrix of system (1), the matrix $\mathcal{A}$ is nonsymmetric; however, since the matrix $\mathcal{A} + \mathcal{A}^T$ is symmetric positive semidefinite, we see that the matrix $\mathcal{A}$ has the following good properties:
1. $\mathcal{A}$ is semipositive real, that is, $v^T Av \geq 0$, for all $v \in \mathbb{R}^{\mathbf{n}}$;
2. $\mathcal{A}$ is positive semistable, which means that $\Re(\lambda) \geq 0$ for all $\lambda \in \sigma(\mathcal{A})$, where $\sigma(\mathcal{A})$ denotes the spectrum of $\mathcal{A}$.
The above results are significant for Krylov subspace methods like GMRES (see [2, p. 420] and [20, Theorem 6.30]).

Recently, some iteration methods and preconditioning techniques have been presented for system (2). In [15], Huang and Ma analyzed the spectral properties of the preconditioned matrix $\mathcal{P}_D^{-1}\mathcal{A}$, where the preconditioner $\mathcal{P}_D$ is of the form

$$\mathcal{P}_D = \begin{pmatrix} A & 0 & 0 \\ 0 & S & 0 \\ 0 & 0 & CS^{-1}C^T \end{pmatrix}, \tag{3}$$

with $S = BA^{-1}B^T$. They proved that when A is symmetric positive definite (SPD) and C and B are of full rank, then all the matrices A, S and $CS^{-1}C^T$ are SPD and the preconditioned matrix $\mathcal{P}_D^{-1}\mathcal{A}$ has six different eigenvalues. Therefore, the GMRES method for the system $\mathcal{P}_D^{-1}\mathcal{A}\mathbf{x} = \mathcal{P}_D^{-1}\mathbf{b}$ will converge in at most six iterations. In the implementation of the preconditioner $\mathcal{P}_D$ within a Krylov subspace iteration method, at each iteration, a system of linear equations of the form $\mathcal{P}_D w = r$ should be solved. To do so, three systems with coefficient matrices A, S and $CS^{-1}C^T$ should be solved. Hence, these systems can be solved directly using the Cholesky factorization, or inexactly using the conjugate gradient (CG) method. Obviously, forming the matrices S and $CS^{-1}C^T$ is too expensive. Moreover, these matrices would be dense even if the matrices A, B and C are sparse. On the other hand, in each matrix-vector multiplication of solving the system with the matrix S using the CG method a system with the coefficient matrix A should be solved, which is very costly. Furthermore, this would be even more complicated for the matrix $CS^{-1}C^T$. Here, we should also note that if the matrix A is ill-conditioned, then the matrix S as well as $CS^{-1}C^T$ are often ill-conditioned, too. To overcome this problem, in [15], the authors proposed to use $\hat{A}$ and $\hat{S}$ as approximations of A and S, respectively. However, they solved the obtained subsystems directly. In [23], Xie and Li presented the following three preconditioners:

$$\begin{aligned} &\mathcal{P}_1 = \begin{pmatrix} A & 0 & 0 \\ B & -S & C^T \\ 0 & 0 & -CS^{-1}C^T \end{pmatrix}, \quad \mathcal{P}_2 = \begin{pmatrix} A & 0 & 0 \\ B & -S & C^T \\ 0 & 0 & CS^{-1}C^T \end{pmatrix}, \\ &\mathcal{P}_3 = \begin{pmatrix} A & B^T & 0 \\ B & -S & 0 \\ 0 & 0 & -CS^{-1}C^T \end{pmatrix}. \end{aligned} \tag{4}$$

It was proved that (see [23])

$$\mathcal{T}_1 = \sigma(\mathcal{P}_1^{-1}\mathcal{K}) = \{-1, 1\}, \quad \mathcal{T}_2 = \sigma(\mathcal{P}_2^{-1}\mathcal{K}) = \{1\}, \quad \mathcal{T}_3 = \sigma(\mathcal{P}_3^{-1}\mathcal{K}) = \{\pm\frac{1}{2}, 1\},$$

and that under the assumptions that A is SPD and C and B are of full rank, all the preconditioned matrices $\mathcal{T}_i\,(i = 1, 2, 3)$ have a minimal polynomial of degree 3. Here, $\sigma(.)$ denotes the spectrum of the matrix. Therefore, the GMRES method for solving the corresponding preconditioned systems converges in at most three iterations. Obviously, the implementation of the preconditioners is quite similar to the preconditioner $\mathcal{P}_D$, and the one with $\mathcal{P}_3$ is even more complicated. In [8], Cao presented the shift-splitting (SS) iteration method for solving system (2). Next in [22], Wang and Zhang generalized the SS iteration method to solve system (2). It has been proved that the SS iteration method and its generalized version are unconditionally convergent (see [8, 22]). Huang [14] and Huang et al. [16] presented the Uzawa-type methods for solving system (1).

In [5], Benzi and Guo proposed the dimensional splitting method and the corresponding dimensional split preconditioner for saddle point problems. Then in [17], Ke and Ma corrected an error in [5]. Next, a modified version of the dimensional splitting method was presented by Cao et al. in [9] for solving generalized saddle point problems. When the (2,2) block of the matrix $\mathcal{A}$ is symmetric positive definite, using the idea of the dimensional splitting method [5], Liang and Zhang [18] proposed alternating positive semidefinite splitting (APSS) for solving the problem. However, in our problem, the (2,2) block of the matrix $\mathcal{A}$ is zero. In this paper, we apply the APSS iteration method for solving system (2) and investigate its convergence properties. We prove that the method is unconditionally convergent. Then we apply the induced preconditioner to accelerate the convergence of the GMRES method for solving the system.

The rest of the paper is organized as follows. In Section 2, we give conditions for the nonsingularity of the matrix $\mathcal{A}$. Section 3 is devoted to the application of the APSS method to solving system (2). Convergence of the method is presented in Section 4. We present some implementation issues for the APSS preconditioner in Section 5. Numerical results are presented in Section 6. Some concluding remarks are presented in Section 7.

2. Existence of the solution

We state the following theorem for the existence of a solution to system (2).

Theorem 1. *Assume that C has full row rank and A is symmetric positive semidefinite. If $\mathcal{N}(A) \cap \mathcal{N}(B) = \{0\}$, $\mathcal{N}(B^T) \cap \mathcal{N}(C) = \{0\}$ and $\mathcal{R}(B) \cap \mathcal{R}(C^T) = \{0\}$, then the matrix $\mathcal{A}$ defined in Eq. (2) is nonsingular.*

Proof. Let $\mathbf{x} = (u; v; p)$ and $\mathcal{A}\mathbf{x} = 0$. Therefore,

$$Au + B^T v = 0, \tag{5}$$

$$Bu + C^T p = 0, \tag{6}$$

$$Cv = 0. \tag{7}$$

Pre-multiplying Eq. (5) by u^T and by using Eq. (6) and Eq. (7) it follows

$$\begin{aligned} 0 &= u^TAu + u^TB^Tv = u^TAu + (Bu)^Tv \\ &= u^TAu - p^TCv \\ &= u^TAu. \end{aligned}$$

Now, since the matrix A is symmetric positive semidefinite, we deduce that $Au = 0$. Therefore, from Eq. (5) we get $B^Tv = 0$. This along with Eq. (7) shows that $v \in \mathcal{N}(B^T) \cap \mathcal{N}(C)$. Thus, $v = 0$. On the other hand, from Eq. (6) we have $Bu = -C^Tp \in \mathcal{R}(B) \cap \mathcal{R}(C^T)$. Hence, $Bu = 0$ and $C^Tp = 0$. So, $p = 0$, since C has full row rank. Finally, from $Bu = 0$ and $Au = 0$, we conclude that $u \in \mathcal{N}(A) \cap \mathcal{N}(B)$, which reveals that $u = 0$. Hereupon, $\mathbf{x} = (u; v; p) = 0$, which completes the proof. □

It is easy to prove that if $\mathcal{R}(B) \subseteq \mathcal{N}(C)$, then $\mathcal{R}(B) \cap \mathcal{R}(C^T) = \{0\}$. So in Theorem 1, one may replace the condition $\mathcal{R}(B) \cap \mathcal{R}(C^T) = \{0\}$ by $\mathcal{R}(B) \subseteq \mathcal{N}(C)$. The following corollaries can be easily deduced using Theorem 1 and its proof.

Corollary 1 (see [14, 16, 23]). *Assume that A is symmetric positive definite and the matrices B and C are of full row rank. Then the matrix $\mathcal{A}$ defined in Eq.* (2) *is nonsingular.*

Corollary 2 (see [23]).

(i) *Assume that A is a symmetric positive semidefinite and the matrices B and C are of full row rank. Then the matrix $\mathcal{A}$ defined in Eq.* (2) *is nonsingular if and only if $\mathcal{N}(A) \cap \mathcal{N}(B) = \{0\}$.*

(ii) *Assume that A is a symmetric positive definite matrix and the matrix C has full row rank. Then the matrix $\mathcal{A}$ defined in Eq.* (2) *is nonsingular if and only if $\mathcal{N}(B^T) \cap \mathcal{N}(C) = \{0\}$.*

3. The APSS method

We split the matrix $\mathcal{A}$ into $\mathcal{A} = \mathcal{A}_1 + \mathcal{A}_2$, where

$$\mathcal{A}_1 = \begin{pmatrix} A & B^T & 0 \\ -B & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad \mathcal{A}_2 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -C^T \\ 0 & C & 0 \end{pmatrix}.$$

Both matrices $\mathcal{A}_1$ and $\mathcal{A}_2$ are positive semidefinite. So, using the splittings

$$\mathcal{A} = (\alpha\mathcal{I} + \mathcal{A}_1) - (\alpha\mathcal{I} - \mathcal{A}_2) = (\alpha\mathcal{I} + \mathcal{A}_2) - (\alpha\mathcal{I} - \mathcal{A}_1),$$

where $\alpha > 0$ is a given real parameter and $\mathcal{I}$ is the identity matrix of order $\mathbf{n}$, we state the APSS iteration

$$\begin{cases} (\alpha\mathcal{I} + \mathcal{A}_1)\mathbf{x}^{(k+\frac{1}{2})} = (\alpha\mathcal{I} - \mathcal{A}_2)\mathbf{x}^{(k)} + \mathbf{b}, \\ (\alpha\mathcal{I} + \mathcal{A}_2)\mathbf{x}^{(k+1)} = (\alpha\mathcal{I} - \mathcal{A}_1)\mathbf{x}^{(k+\frac{1}{2})} + \mathbf{b}, \end{cases} \quad k = 0, 1, 2, \ldots, \tag{8}$$

where $\mathbf{x}^{(0)} \in \mathbb{R}^{\mathbf{n}}$ is a given parameter. It follows from Lemma 1 in [3] that both matrices $\alpha\mathcal{I} + \mathcal{A}_1$ and $\alpha\mathcal{I} + \mathcal{A}_2$ are nonsingular. Computing $\mathbf{x}^{(k+\frac{1}{2})}$ from the first equation in (8) and substituting in the second one results in the stationary iteration

$$\mathbf{x}^{(k+1)} = \mathcal{T}_\alpha \mathbf{x}^{(k)} + \mathbf{c}, \tag{9}$$

where

$$\mathcal{T}_\alpha = \left(\alpha\mathcal{I} + \mathcal{A}_2\right)^{-1} \left(\alpha\mathcal{I} - \mathcal{A}_1\right) \left(\alpha\mathcal{I} + \mathcal{A}_1\right)^{-1} \left(\alpha\mathcal{I} - \mathcal{A}_2\right),$$

and

$$\mathbf{c} = 2\alpha \left(\alpha\mathcal{I} + \mathcal{A}_2\right)^{-1} \left(\alpha\mathcal{I} + \mathcal{A}_1\right)^{-1} \mathbf{b}.$$

Similarly to the Hermitian and skew-Hermitian splitting (HSS) iteration method [1], if we set

$$\mathcal{P}_\alpha = \frac{1}{2\alpha}(\alpha\mathcal{I} + \mathcal{A}_1)(\alpha\mathcal{I} + \mathcal{A}_2), \qquad \mathcal{Q}_\alpha = \frac{1}{2\alpha}(\alpha\mathcal{I} - \mathcal{A}_1)(\alpha\mathcal{I} - \mathcal{A}_2),$$

then $\mathcal{A} = \mathcal{P}_\alpha - \mathcal{Q}_\alpha$ and

$$\mathcal{T}_\alpha = \mathcal{P}_\alpha^{-1}\mathcal{Q}_\alpha = \mathcal{I} - \mathcal{P}_\alpha^{-1}\mathcal{A}. \tag{10}$$

In each iteration of the APSS iteration method, two systems with the coefficient matrices $\alpha\mathcal{I} + \mathcal{A}_1$ and $\alpha\mathcal{I} + \mathcal{A}_2$ should be solved. The first step of the kth iteration of APSS can be written as

$$\begin{pmatrix} \alpha I + A & B^T & 0 \\ -B & \alpha I & 0 \\ 0 & 0 & \alpha I \end{pmatrix} \begin{pmatrix} x^{(k+\frac{1}{2})} \\ y^{(k+\frac{1}{2})} \\ z^{(k+\frac{1}{2})} \end{pmatrix} = \begin{pmatrix} \alpha x^{(k)} + f \\ \alpha y^{(k)} + C^T z^{(k)} - g \\ -C y^{(k)} + \alpha z^{(k)} + h \end{pmatrix} \equiv \begin{pmatrix} f^{(k)} \\ g^{(k)} \\ h^{(k)} \end{pmatrix}. \tag{11}$$

From Eq. (11), we immediately get $z^{(k+\frac{1}{2})} = h^{(k)}/\alpha$ and the reduced system

$$\begin{pmatrix} \alpha I + A & B^T \\ -B & \alpha I \end{pmatrix} \begin{pmatrix} x^{(k+\frac{1}{2})} \\ y^{(k+\frac{1}{2})} \end{pmatrix} = \begin{pmatrix} f^{(k)} \\ g^{(k)} \end{pmatrix}, \tag{12}$$

From the second equation in (12), we obtain

$$y^{(k+\frac{1}{2})} = \frac{1}{\alpha}(g^{(k)} + Bx^{(k+\frac{1}{2})}),$$

and substituting it into the first equation of (12) gives the system

$$(\alpha I + A + \frac{1}{\alpha}B^T B)x^{(k+\frac{1}{2})} = f^{(k)} - \frac{1}{\alpha}B^T g^{(k)},$$

for $x^{(k+\frac{1}{2})}$. Summarizing the above results gives the following algorithm for computing $\mathbf{x}^{(k+\frac{1}{2})}$:

Algorithm 1. *Computation of* $\mathbf{x}^{(k+\frac{1}{2})}$*:*

```
1. Set $z^{(k+\frac{1}{2})} = h^{(k)}/\alpha$.
2. Solve $(\alpha I + A + \frac{1}{\alpha}B^T B)x^{(k+\frac{1}{2})} = f^{(k)} - \frac{1}{\alpha}B^T g^{(k)}$ for $x^{(k+\frac{1}{2})}$.
3. Set $y^{(k+\frac{1}{2})} = \frac{1}{\alpha}(g^{(k)} + Bx^{(k+\frac{1}{2})})$.
4. Set $\mathbf{x}^{(k+\frac{1}{2})} = (x^{(k+\frac{1}{2})}; y^{(k+\frac{1}{2})}; z^{(k+\frac{1}{2})})$.
```

The second step of the kth iteration of APSS can be written as

$$\begin{pmatrix} \alpha I & 0 & 0 \\ 0 & \alpha I & -C^T \\ 0 & C & \alpha I \end{pmatrix} \begin{pmatrix} x^{(k+1} \\ y^{(k+1)} \\ z^{(k+1)} \end{pmatrix} = \begin{pmatrix} (\alpha I - A)x^{(k+\frac{1}{2})} - B^T y^{(k+\frac{1}{2})} + f \\ Bx^{(k+\frac{1}{2})} + \alpha y^{(k+\frac{1}{2})} - g \\ \alpha z^{(k+\frac{1}{2})} + h \end{pmatrix}$$
$$\equiv \begin{pmatrix} f^{(k+\frac{1}{2})} \\ g^{(k+\frac{1}{2})} \\ h^{(k+\frac{1}{2})} \end{pmatrix}.$$

From the first equation of the above system we get $x^{(k+1} = f^{(k+\frac{1}{2})}/\alpha$. After eliminating the variable $x^{(k+1)}$ from the above system we find the following reduced system

$$\begin{pmatrix} \alpha I & -C^T \\ C & \alpha I \end{pmatrix} \begin{pmatrix} y^{(k+1)} \\ z^{(k+1)} \end{pmatrix} = \begin{pmatrix} g^{(k+\frac{1}{2})} \\ h^{(k+\frac{1}{2})} \end{pmatrix}. \tag{13}$$

From the first equation in (13) we deduce that

$$y^{(k+1)} = \frac{1}{\alpha}(g^{(k+\frac{1}{2})} + C^T z^{(k+1)}),$$

and substituting $y^{(k+1)}$ in the second equation of (13) results in the following system for $z^{(k+1)}$:

$$(\alpha I + \frac{1}{\alpha} CC^T) z^{(k+1)} = h^{(k+\frac{1}{2})} - \frac{1}{\alpha} C g^{(k+\frac{1}{2})}.$$

Summarizing the above results gives the following algorithm for computing $\mathbf{x}^{(k+1)}$.

Algorithm 2. *Computation of* $\mathbf{x}^{(k+1)}$*:*

1. *Set* $x^{(k+1)} = f^{(k+\frac{1}{2})}/\alpha$*.*
2. *Solve* $(\alpha I + \frac{1}{\alpha} CC^T) z^{(k+1)} = h^{(k+\frac{1}{2})} - \frac{1}{\alpha} C g^{(k+\frac{1}{2})}$ *for* $z^{(k+1)}$*.*
3. *Set* $y^{(k+1)} = \frac{1}{\alpha}(g^{(k+\frac{1}{2})} + C^T z^{(k+1)})$*.*
4. *Set* $\mathbf{x}^{(k+1)} = (x^{(k+1)}; y^{(k+1)}; z^{(k+1)})$*.*

From algorithms 1 and 2 we see that two systems with the coefficient matrices $\alpha I + A + \frac{1}{\alpha} B^T B$ and $\alpha I + \frac{1}{\alpha} CC^T$ should be solved. Both of these matrices are symmetric positive definite. Therefore, the corresponding systems can be solved exactly using the Cholesky factorization, or inexactly using the conjugate gradient (CG) method.

4. Convergence of the APSS iteration method

It is well-known that the stationary iteration method (9) converges to the solution of system (2) for any initial guess $\mathbf{x}^{(0)}$ if and only if $\rho(\mathcal{T}_\alpha) < 1$. To prove the convergence, we first state the next lemma.

Lemma 1 (Kellogg's lemma, see [19])**.** *If* $A \in \mathbb{C}^{n \times n}$ *is positive semidefinite, then*

$$\|(\alpha I + A)^{-1}(\alpha I - A)\|_2 \leq 1,$$

for all $\alpha > 0$. Moreover, if A is positive definite, then

$$\|(\alpha I + A)^{-1}(\alpha I - A)\|_2 < 1,$$

for all $\alpha > 0$.

Lemma 2. *Let A be SPD and let matrices B and C be of full row rank. Then the spectral radius of the iteration matrix $\mathcal{T}_\alpha$ in* (9) *is less than or equal to unity, that is, $\rho(\mathcal{T}_\alpha) \leq 1$, in which $\rho(.)$ denotes the spectral radius of the matrix.*

Proof. Clearly, the matrix $\mathcal{T}_\alpha$ is similar to

$$\hat{\mathcal{T}}_\alpha = (\alpha\mathcal{I} - \mathcal{A}_1)(\alpha\mathcal{I} + \mathcal{A}_1)^{-1}(\alpha\mathcal{I} - \mathcal{A}_2)(\alpha\mathcal{I} + \mathcal{A}_2)^{-1}.$$

So,

$$\rho(\mathcal{T}_\alpha) = \rho(\hat{\mathcal{T}}_\alpha) \leq \left\|(\alpha\mathcal{I} - \mathcal{A}_1)(\alpha\mathcal{I} + \mathcal{A}_1)^{-1}\right\|_2 \left\|(\alpha\mathcal{I} - \mathcal{A}_2)(\alpha\mathcal{I} + \mathcal{A}_2)^{-1}\right\|_2.$$

Since both matrices $\mathcal{A}_1$ and $\mathcal{A}_2$ are positive semidefinite, it follows from the Kellogg's lemma that

$$\left\|(\alpha\mathcal{I} - \mathcal{A}_1)(\alpha\mathcal{I} + \mathcal{A}_1)^{-1}\right\|_2 \leq 1, \quad \left\|(\alpha\mathcal{I} - \mathcal{A}_2)(\alpha\mathcal{I} + \mathcal{A}_2)^{-1}\right\|_2 \leq 1.$$

Therefore, we conclude that $\rho(\mathcal{T}_\alpha) = \rho(\hat{\mathcal{T}}_\alpha) \leq 1$. □

It follows from Lemma 2 that for the convergence of the APSS iteration method it is enough to prove that $\rho(\mathcal{T}_\alpha) = 1$ never happens. To do so, we state the following lemma.

Lemma 3. *Let A be SPD and matrices B and C be of full row rank. Then the following are equivalent:*

(*i*) *The matrix*

$$\mathcal{G}_\alpha = \begin{pmatrix} A & B^T + \frac{1}{\alpha^2}B^TC^TC & 0 \\ -B & 0 & -C^T \\ 0 & C & 0 \end{pmatrix}$$

does not have any purely imaginary eigenvalue.

(*ii*) *The spectral radius of the iteration matrix $\mathcal{T}_\alpha$ in* (9) *is strictly less than 1, that is, $\rho(\mathcal{T}_\alpha) < 1$.*

Proof. Similarly to the proof of Lemma 2 in [5], let λ be an eigenvalue of $\mathcal{T}_\alpha$. Then, it follows from Eq. (10) that $\lambda = 1 - \mu$, where μ is an eigenvalue of the matrix $\mathcal{P}_\alpha^{-1}\mathcal{A}$. Let $(\mu, \mathbf{x})$ be an eigenpair of $\mathcal{P}_\alpha^{-1}\mathcal{A}$. Then, we have $\mathcal{A}\mathbf{x} = \mu\mathcal{P}_\alpha\mathbf{x}$, which is equivalent to

$$\mathcal{A}\mathbf{x} = \frac{\mu}{2\alpha}(\mathcal{A}_1\mathcal{A}_2 + \alpha\mathcal{A} + \alpha^2\mathcal{I})\mathbf{x},$$

or

$$\left(1-\frac{1}{2}\mu\right)\mathcal{A}\mathbf{x}=\frac{\mu\alpha}{2}(\mathcal{I}+\frac{1}{\alpha^2}\mathcal{A}_1\mathcal{A}_2)\mathbf{x}. \tag{14}$$

Direct computation reveals that

$$\mathcal{H}_\alpha := \mathcal{I}+\frac{1}{\alpha^2}\mathcal{A}_1\mathcal{A}_2=\begin{pmatrix} I & 0 & -\frac{1}{\alpha^2}B^TC^T \\ 0 & I & 0 \\ 0 & 0 & I \end{pmatrix},$$

which is obviously nonsingular. Since both matrices $\mathcal{A}$ and $\mathcal{H}_\alpha$ are nonsingular we deduce that $\mu \neq 0$ and $\mu \neq 2$. Therefore, from (14) we have

$$\mathcal{H}_\alpha^{-1}\mathcal{A}\mathbf{x}=\frac{\mu\alpha}{2-\mu}\mathbf{x}.$$

This shows that

$$\theta := \frac{\mu\alpha}{2-\mu} \tag{15}$$

is an eigenvalue of

$$\mathcal{H}_\alpha^{-1}\mathcal{A}=\begin{pmatrix} A & B^T+\frac{1}{\alpha^2}B^TC^TC & 0 \\ -B & 0 & -C^T \\ 0 & C & 0 \end{pmatrix}=\mathcal{G}_\alpha. \tag{16}$$

From (15) we see that

$$\mu=\frac{2\theta}{\alpha+\theta},$$

and as a result

$$\lambda = 1-\mu = 1-\frac{2\theta}{\alpha+\theta}=\frac{\alpha-\theta}{\alpha+\theta}.$$

Now from Lemma 2 we have

$$|\lambda| = \left|\frac{\alpha-\theta}{\alpha+\theta}\right| \leq 1,$$

and $|\lambda| = 1$ if and only if $|\theta-\alpha| = |\theta+\alpha|$, which is itself equivalent to

$$(\Re(\theta)-\alpha)^2+\Im(\theta)^2=(\Re(\theta)+\alpha)^2+\Im(\theta)^2.$$

The latter equation is equivalent to $\Re(\theta)=0$. Therefore, $\rho(\mathcal{T}_\alpha)=1$ if and only if $\mathcal{G}_\alpha$ has at least one purely imaginary eigenvalue. □

Theorem 2. *Let A be SPD and matrices B and let C be of full row rank. Then the spectral radius of the iteration matrix $\mathcal{T}_\alpha$ in (9) is strictly less than 1, that is,*

$$\rho(\mathcal{T}_\alpha) < 1, \quad \forall \alpha > 0,$$

which means that the APSS iteration method unconditionally converges to the solution of (2).

Proof. According to Lemma 3, all we need is to prove that the matrix $\mathcal{G}_\alpha$ defined in (16) has no purely imaginary eigenvalue. Let $(\theta, \mathbf{x})$ be an eigenpair of the matrix $\mathcal{G}_\alpha$ with $\|\mathbf{x}\|_2 = 1$. Clearly, the matrix $\mathcal{G}_\alpha$ is nonsingular, therefore $\theta \neq 0$. Letting $\mathbf{x} = (u; v; p)$, it follows from $\mathcal{G}_\alpha \mathbf{x} = \theta \mathbf{x}$ that

$$Au + B^T(I + \frac{1}{\alpha^2}C^TC)v = \theta u, \tag{17}$$

$$-Bu - C^Tp = \theta v, \tag{18}$$

$$Cv = \theta p. \tag{19}$$

If $u = 0$, then it follows from $B^T(I + \frac{1}{\alpha^2}C^TC)v = 0$ that $v = 0$, since B^T is of full column rank and the matrix $I + \frac{1}{\alpha^2}C^TC$ is SPD. Substituting $v = 0$ in (19) yields $p = 0$. Therefore, $\mathbf{x} = (u; v; p) = 0$, which is impossible since $\mathbf{x}$ is an eigenvector. Therefore, $u \neq 0$.

If $v = 0$, then from Eq. (17) we obtain $Au = \theta u$. Thus,

$$\theta = \frac{1}{\|u\|_2^2} u^* A u > 0,$$

since $u \neq 0$ and A is SPD. This means that $\Re(\theta) = \theta > 0$.

If $p = 0$, then from Eq. (19) we get $Cv = 0$. Therefore, from eqs. (17) and (18) we obtain the eigenvalue problem

$$\begin{pmatrix} A & B^T \\ -B & 0 \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = \theta \begin{pmatrix} u \\ v \end{pmatrix}. \tag{20}$$

Eq. (20) shows that θ is an eigenvalue of the saddle matrix

$$\mathcal{S} := \begin{pmatrix} A & B^T \\ -B & 0 \end{pmatrix}.$$

Therefore, since A is SPD and B is of full rank, it follows from [3, Lemma 1.1 (iv)] that $\Re(\theta) > 0$.

According to the above results, here we assume that $u \neq 0$, $v \neq 0$ and $p \neq 0$. From $\mathcal{G}_\alpha \mathbf{x} = \theta \mathbf{x}$ and $\|\mathbf{x}\|_2 = 1$, we get $\theta = \mathbf{x}^* \mathcal{G}_\alpha \mathbf{x}$. Hence,

$$\begin{aligned} \Re(\theta) &= \frac{1}{2}\mathbf{x}^*(\mathcal{G}_\alpha + \mathcal{G}_\alpha^T)\mathbf{x} = \frac{1}{2}\begin{pmatrix} u^* & v^* & p^* \end{pmatrix} \begin{pmatrix} 2A & \frac{1}{\alpha^2}B^TC^TC & 0 \\ \frac{1}{\alpha^2}C^TCB & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} u \\ v \\ p \end{pmatrix} \\ &= u^*Au + \frac{1}{2\alpha^2}\left(u^*B^TC^TCv + v^*C^TCBu\right). \end{aligned} \tag{21}$$

On the other hand, from Eq. (17) we deduce

$$u^*Au + u^*B^T(I + \frac{1}{\alpha^2}C^TC)v = \theta u^*u, \tag{22}$$

and, taking the conjugate of both sides, the latter equation gives

$$u^*Au + v^*(I + \frac{1}{\alpha^2}C^TC)Bu = \bar{\theta}u^*u. \tag{23}$$

From eqs. (22) and (23) we get

$$u^*Au + \frac{1}{2}(u^*B^Tv + v^*Bu) + \frac{1}{2\alpha^2}\left(u^*B^TC^TCv + v^*C^TCBu\right) = \Re(\theta)u^*u. \tag{24}$$

From eqs. (24) and (21) we obtain

$$\Re(\theta) + \frac{1}{2}(u^*B^Tv + v^*Bu) = \Re(\theta)u^*u.$$

Now, by contradiction we assume that $\Re(\theta) = 0$. In this case, from the above equation we deduce that

$$u^*B^Tv + v^*Bu = 0. \tag{25}$$

On the other hand, we have

$$\begin{aligned}
\Re(\theta) &= u^*Au + \frac{1}{2\alpha^2}\left(u^*B^TC^TCv + v^*C^TCBu\right) \\
&= u^*Au + \frac{1}{2\alpha^2}\left(\theta u^*B^TC^Tp + \bar{\theta}p^*CBu\right) && \text{(from Eq. (19))} \\
&= u^*Au + \frac{1}{2\alpha^2}\left(\theta u^*B^T(-Bu - \theta v) + \bar{\theta}(-Bu - \theta v)^*Bu\right) && \text{(from Eq. (18))} \\
&= u^*Au - \frac{1}{2\alpha^2}\left(2\Re(\theta)\|Bu\|_2^2 + \theta^2u^*B^Tv + \bar{\theta}^2v^*Bu\right).
\end{aligned}$$

Now, if $\Re(\theta) = 0$, then $\theta = i\xi$, where $\xi \neq 0$. Therefore, from the above equation we see that

$$\begin{aligned}
0 &= u^*Au + \frac{\xi^2}{2\alpha^2}\left(u^*B^Tv + v^*Bu\right) \\
&= u^*Au, \qquad \text{(from Eq. (25))},
\end{aligned}$$

which is a contraction since $u \neq 0$ and A is SPD. Therefore, the proof is completed. □

5. APSS preconditioner and its implementation issues

In the previous section we have proved that under the conditions of Theorem 2 we have $\rho(\mathcal{T}_\alpha) < 1$. Therefore, the eigenvalues of $\mathcal{T}_\alpha$ are included in a circle centered at the origin with radius 1. On the other hand, from (10) we have

$$\mathcal{P}_\alpha^{-1}\mathcal{A} = \mathcal{I} - \mathcal{T}_\alpha.$$

Hence, the eigenvalues of $\mathcal{P}_\alpha^{-1}\mathcal{A}$ are contained in a circle centered at $(1,0)$ with radius 1. Therefore, $\mathcal{P}_\alpha$ can be used as a preconditioner for the system $\mathcal{A}\mathbf{x} = \mathbf{b}$. Since the pre-factor $1/(2\alpha)$ in the matrix $\mathcal{P}_\alpha$ has no effect on the preconditioned system, we drop it. Hence, the matrix

$$\mathcal{M}_\alpha = (\alpha\mathcal{I} + \mathcal{A}_1)(\alpha\mathcal{I} + \mathcal{A}_2)$$

can be used to accelerate the convergence of the Krylov subspace methods like GMRES or its restarted version to solve the system $\mathcal{A}\mathbf{x} = \mathbf{b}$. In each iteration of the GMRES method a linear system of the equations of the form $\mathcal{M}_\alpha v = r$ should be solved. This can be done in two steps. In the first step, the system

$$(\alpha\mathcal{I} + \mathcal{A}_1)w = r$$

is solved for w and then the system

$$(\alpha\mathcal{I} + \mathcal{A}_2)v = w$$

for v. These systems can be solved using algorithms 1 and 2.

Remark 1. *According to part (i) of Lemma 2, if the matrix A is symmetric positive semidefinite and the matrices B and C are of full row rank with $\mathcal{N}(A) \cap \mathcal{N}(B) = \{0\}$, then the matrix $\mathcal{A}$ is nonsingular. In this case, the proposed iteration method is still well-defined; however, convergence of the method is not guaranteed. Nevertheless, from Lemma 2 we have $\rho(\mathcal{T}_\alpha) \leq 1$ and the GMRES method can be applied to the corresponding preconditioned system. On the other hand, using the idea of [3], one may state a convergent iteration method using the extrapolation technique.*

6. Numerical experiments

We present some numerical results to show the efficiency of the proposed preconditioner. We present three examples and in all examples we first apply a symmetric diagonal scaling for the matrix $\mathcal{A}$. To do so, we replace the coefficient matrix $\mathcal{A}$ by the matrix $\mathcal{D}^{-\frac{1}{2}}\mathcal{A}\mathcal{D}^{-\frac{1}{2}}$, where $\mathcal{D} = \operatorname{diag}(\|\mathcal{A}_1\|_2, \ldots, \|\mathcal{A}_{\mathbf{n}}\|_2)$, in which $\mathcal{A}_j$ is the jth column of the matrix $\mathcal{A}$. The right-hand side vector of the system is set to $\mathbf{b} = \mathcal{A}\mathbf{e}$, where $\mathbf{e}$ is a vector of all ones. We use the flexible version of the GMRES(50) [20, 21], FGMRES(50), for solving the systems. The iteration is started from a zero vector and terminated as soon as the residual 2-norm is reduced by a factor of 10^{-6}. The maximum number of iterations is set to be 20 000.

For the APSS preconditioner we use the implementation method presented in Section 5. The subsystems are solved using the conjugate gradient (CG) method. The CG method is started from a zero vector and the iteration is stopped as soon as the residual 2-norm is reduced by a factor of 10^3. The maximum number of CG iterations is set to be 200. We compare the numerical results of the APSS preconditioner with those of the preconditioners $\mathcal{P}_D$ and $\mathcal{P}_2$ defined in eqs. (3) and (4), respectively. In the implementation of the preconditioners $\mathcal{P}_D$ and $\mathcal{P}_2$ in the FGMRES(50) method three systems with the coefficient matrices A, $S = BA^{-1}B$ and $M = CS^{-1}C^T$ should be solved, which are symmetric positive definite. The system with the coefficient matrix A can be solved using the CG method; however, there is no obvious implementation method to solve inexactly the systems with the coefficient matrices S and $CS^{-1}C^T$. Hence, using the idea of [14], we set

$$\hat{A} = LL^T,$$

where L is computed using the incomplete Cholesky factorization with dropping tolerance 10^{-8}. To do so, we use the Matlab command:

```
L=ichol(A, struct('type','ict','droptol',1e-8)).
```

We also set

$$\hat{S} = \mathrm{diag}(B\hat{A}^{-1}B^T).$$

Now, instead of using the preconditioners $\mathcal{P}_D$ and $\mathcal{P}_2$, we use

$$\hat{\mathcal{P}}_D = \begin{pmatrix} \hat{A} & 0 & 0 \\ 0 & \hat{S} & 0 \\ 0 & 0 & C\hat{S}^{-1}C^T \end{pmatrix}, \quad \hat{\mathcal{P}}_2 = \begin{pmatrix} \hat{A} & 0 & 0 \\ B & -\hat{S} & C^T \\ 0 & 0 & C\hat{S}^{-1}C^T \end{pmatrix},$$

respectively. In the implementation of these preconditioners the matrix $C\hat{S}^{-1}C^T$ is formed explicitly and the corresponding system is solved using the Cholesky factorization. All runs are implemented in Matlab R2017, equipped with a laptop with 1.80 GHz central processing unit (Intel(R) Core(TM) i7-4500), 6 GB memory and Windows 7 operating system.

Numerical results are presented in the tables. In the tables, “Iters” stands for the number of iterations for the convergence. The elapsed CPU time (in seconds) is denoted by “Time”. The value of R_k defined by

$$R_k = \frac{\|\mathbf{b} - \mathcal{A}\mathbf{x}^{(k)}\|_2}{\|\mathbf{b}\|_2}$$

is also reported, where $\mathbf{x}^{(k)}$ is the computed solution at iteration k. In the tables, *nnz* stands for the number of nonzero entries of the matrix. A dagger (†) shows that the method has not converged in the maximum number of iterations. Also, § shows that the preconditioner has not been computed in 500 seconds. Finally, “NA” (for not applicable) means that the coefficient matrix does not satisfy the assumptions of Theorem 2. We now present the examples.

Example 1. *We consider problem* (2) *with (see [14, 23])*

$$A = \begin{pmatrix} I\otimes T + T\otimes I & 0 \\ 0 & I\otimes T + T\otimes I \end{pmatrix} \in \mathbb{R}^{2p^2\times 2p^2},$$
$$B = (I\otimes F, F\otimes I) \in \mathbb{R}^{p^2\times 2p^2} \text{ and } C = E\otimes F \in \mathbb{R}^{p^2\times p^2},$$

where

$$T = \frac{1}{h^2}\,\mathrm{tridiag}(-1,2,-1) \in \mathbb{R}^{p\times p}, \quad F = \frac{1}{h}\,\mathrm{tridiag}(0,1,-1) \in \mathbb{R}^{p\times p},$$

and $E = \mathrm{diag}\left(1, p+1, \ldots, p^2-p+1\right)$ *in which* $\otimes$ *denotes the Kronecker product and* $h = 1/(p+1)$ *the discretization meshsize. We present the numerical results for* $p = 16, 32, 64, 128,$ *and* 256. *We may use the idea of [7] to estimate the optimal value of the parameter* α. *However, for the APSS preconditioner the value of* α *is set to be* 0.005 *for all dimensions in this example. We have experimentally found that this choice often gives quite suitable results for this problem. It is necessary to mention that* $\alpha = 0.005$ *is not the optimal value of the parameter. Numerical results are presented in Table 1.*

$p, \mathbf{n}, nnz$		No Prec.	$\mathcal{M}_\alpha$	$\hat{\mathcal{P}}_D$	$\hat{\mathcal{P}}_2$
$p = 16$	Iters	425	15	29	22
$\mathbf{n} = 1024$	Time	4.02	0.04	0.04	0.04
$nnz = 5408$	R_k	8.6e-7	3.1e-7	5.9e-7	8.8e-7
$p = 32$	Iters	949	13	32	25
$\mathbf{n} = 4096$	Time	83.43	0.09	0.32	0.25
$nnz = 22080$	R_k	9.9e-7	5.3e-7	6.9e-7	7.5e-7
$p = 64$	Iters	†	13	34	28
$\mathbf{n} = 16384$	Time	-	0.47	8.69	5.99
$nnz = 89216$	R_k	-	9.1e-7	7.3e-7	5.7e-7
$p = 128$	Iters	†	22	68	69
$\mathbf{n} = 65536$	Time	-	3.10	174.16	175.89
$nnz = 358656$	R_k	-	6.4e-7	4.8e-7	5.6e-7
$p = 256$	Iters	†	51	§	§
$\mathbf{n} = 262144$	Time	-	37.47	-	-
$nnz = 1438208$	R_k	-	6.6e-7	-	-

Table 1: *Numerical results for FGMRES(50) to solve Example 1*

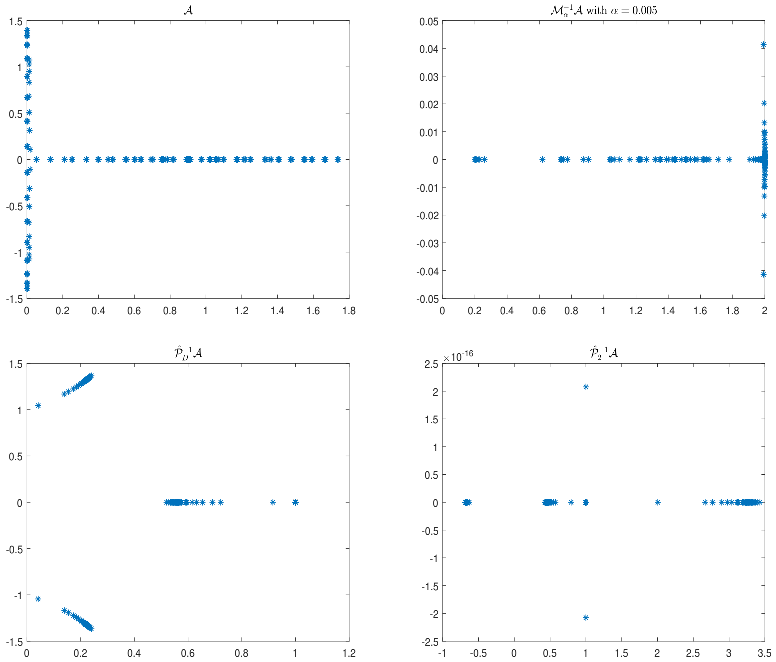

Figure 1: *Eigenvalue distribution $\mathcal{A}$, $\mathcal{M}_\alpha^{-1}\mathcal{A}$, $\hat{\mathcal{P}}_D^{-1}\mathcal{A}$ and $\hat{\mathcal{P}}_2^{-1}\mathcal{A}$ with $p = 8$ for Example 1*

Numerical results show that the three preconditioners significantly reduce the number of iterations and the CPU time of FGMRES(50). However, the APSS preconditioner outperforms the others from both the number of iterations and the CPU time point of view. The main problem with the preconditioners $\hat{\mathcal{P}}_D$ and $\hat{\mathcal{P}}_2$ is their need for a *large amount of CPU time to set up.*

Fig. 1 displays the eigenvalues of matrices $\mathcal{A}$, $\mathcal{M}_\alpha^{-1}\mathcal{A}$, $\hat{\mathcal{P}}_D^{-1}\mathcal{A}$ *and* $\hat{\mathcal{P}}_2^{-1}\mathcal{A}$ *for* $p = 8$. *In this case, the matrix* A *is of dimension* $\mathbf{n} = 256$. *It is seen that the eigenvalues of* $\mathcal{M}_\alpha^{-1}\mathcal{A}$ *are clustered in a circle centered at* $(1,0)$ *with radius 1. Another observation which can be posed here is that the eigenvalues of* $\mathcal{M}_\alpha^{-1}\mathcal{A}$ *are more clustered around the point* $(1,0)$ *than the others.*

Example 2. *We consider the three-by-three block saddle point problem* (2), *where (see [14, 23])*

$$A = \operatorname{diag}\left(2W^TW + D_1, D_2, D_3\right) \in \mathbb{R}^{n\times n},$$

is a block-diagonal matrix,

$$B = [E, -I_{2\widetilde{p}}, I_{2\widetilde{p}}] \in \mathbb{R}^{m\times n} \text{ and } \quad C = E^T \in \mathbb{R}^{l\times m},$$

are both full row-rank matrices, where $\tilde{p} = p^2$, $\hat{p} = p(p+1)$; $W = (w_{ij}) \in \mathbb{R}^{\hat{p}\times\hat{p}}$ *with* $w_{ij} = e^{-2\left((i/3)^2+(j/3)^2\right)}$; $D_1 = I_{\widehat{p}}$ *is an identity matrix;* $D_i = \operatorname{diag}(d_j^{(i)}) \in \mathbb{R}^{2\tilde{p}\times 2\tilde{p}}$, $i = 2,3$, *are diagonal matrices, with*

$$d_j^{(2)} = \begin{cases} 1, & \text{for} \quad 1 \le j \le \tilde{p}, \\ 10^{-5}(j-\tilde{p})^2, & \text{for } \tilde{p}+1 \le j \le 2\tilde{p}, \end{cases}$$

$$d_j^{(3)} = 10^{-5}(j+\tilde{p})^2 \text{ for } 1 \le j \le 2\tilde{p},$$

and

$$E = \begin{pmatrix} \widehat{E} \otimes I_p \\ I_p \otimes \widehat{E} \end{pmatrix}, \quad \widehat{E} = \begin{pmatrix} 2 & -1 & & & \\ & 2 & -1 & & \\ & & \ddots & \ddots & \\ & & & 2 & -1 \end{pmatrix} \in \mathbb{R}^{p\times(p+1)}.$$

The numerical results are listed in Table 2 for different values of p ($p = 16, 32, 64, 128, 256$). *In the APSS preconditioner, the value of* α *is set to be* 0.4 *for all dimensions. We have experimentally found that this value often gives good results for this problem.*

As the numerical results show, the APSS preconditioner improves significantly the number of the iterations and the CPU time of the GMRES method. As we see, the preconditioner $\hat{\mathcal{P}}_D$ *improves neither the number of iterations nor the CPU time of the GMRES for all sizes, except for* $p = 16$. *On the other hand,* $\hat{\mathcal{P}}_2$ *improves slightly the number of iterations; however, this is not the case for the CPU time. Therefore, the APSS preconditioner is superior to the others from both the number of iterations and the CPU time point of view.*

Fig. 2 shows the eigenvalues of matrices $\mathcal{A}$, $\mathcal{M}_\alpha^{-1}\mathcal{A}$, $\hat{\mathcal{P}}_D^{-1}\mathcal{A}$ *and* $\hat{\mathcal{P}}_2^{-1}\mathcal{A}$ *for* $p = 8$. *In this case, the matrix* A *is of dimension* $\mathbf{n} = 256$. *It is seen that the eigenvalues of* $\mathcal{M}_\alpha^{-1}\mathcal{A}$ *are clustered in a circle centered at* $(1,0)$ *with radius 1. We also see that the eigenvalues of* $\mathcal{M}_\alpha^{-1}\mathcal{A}$ *are more clustered around the point* $(1,0)$ *than the others.*

$p, \mathbf{n}, nnz$		No Prec.	$\mathcal{M}_\alpha$	$\mathcal{P}_D$	$\mathcal{P}_2$
$p = 16$	Iters	150	31	96	70
$\mathbf{n} = 2080$	Time	0.08	0.04	0.07	0.06
$nnz = 9972$	R_k	9.4e-7	8.8e-7	1.0e-6	9.8e-7
$p = 32$	Iters	176	32	222	158
$\mathbf{n} = 4096$	Time	0.36	0.14	0.58	0.47
$nnz = 22080$	R_k	1.0e-6	8.0e-7	9.3e-7	9.9e-7
$p = 64$	Iters	213	31	275	186
$\mathbf{n} = 16384$	Time	1.53	0.42	3.16	2.59
$nnz = 89216$	R_k	9.8e-7	9.9e-7	1.0e-6	9.8e-7
$p = 128$	Iters	204	30	263	178
$\mathbf{n} = 65536$	Time	7.59	1.67	26.22	21.94
$nnz = 358656$	R_k	9.7e-7	8.4e-7	9.9e-7	9.8e-7
$p = 256$	Iters	195	29	248	167
$\mathbf{n} = 262144$	Time	32.01	8.41	853.62	832.01
$nnz = 1438208$	R_k	9.6e-7	7.5e-7	9.7e-7	9.6e-7

Table 2: *Numerical results of FGMRES(50) to solve Example 2*

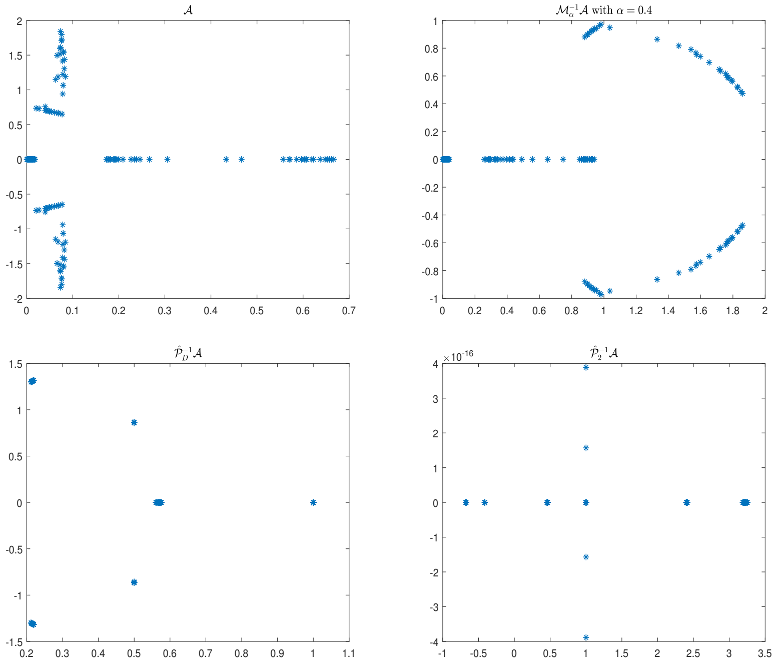

Figure 2: *Eigenvalue distribution $\mathcal{A}$, $\mathcal{M}_\alpha^{-1}\mathcal{A}$, $\hat{\mathcal{P}}_D^{-1}\mathcal{A}$ and $\hat{\mathcal{P}}_2^{-1}\mathcal{A}$ with $p = 8$ for Example 2*

Example 3. *In this example, we consider a quadratic program [12, 14]*

$$\begin{aligned} &\min_{x\in\mathbb{R}^n, y\in\mathbb{R}^l} \frac{1}{2}x^T Ax + r^T x + q^T y \\ &s.t.: \quad Bx + C^T y = b, \end{aligned} \tag{26}$$

where $r \in \mathbb{R}^n$ and $q \in \mathbb{R}^l$. To solve the above problem we define the Lagrange function

$$L(x, y, \lambda) = \frac{1}{2}x^T Ax + r^T x + q^T y + \lambda^T (Bx + C^T y - b),$$

where the vector $\lambda \in \mathbb{R}^m$ is the Lagrange multiplier. Then the Karush-Kuhn-Tucker necessary conditions of (26) are as follows (see [6]):

$$\nabla_x L(x, y, \lambda) = 0, \quad \nabla_y L(x, y, \lambda) = 0 \quad \textit{and} \quad \nabla_\lambda L(x, y, \lambda) = 0.$$

It is easy to see that these equations give a system of linear equations of the form (1). In this example, we have chosen the matrices A, B and C from the CUTEr collection [11]. Numerical results along with the generic properties of the test matrices are presented in Table 3. For the test matrices MOSARQP1 and STCQP2, the matrix C is not full rank. Hence, the matrix $C\hat{S}^{-1}C^T$ is symmetric positive semidefinite, which shows that the preconditioners $\mathcal{P}_D$ and $\mathcal{P}_2$ are singular. Therefore, they can not be used for preconditioning the system. Similarly, for the test matrices UBH1, AUG2D, CONT201 and CONT300, the matrix A is symmetric positive semidefinte. Hence the matrix S as well as the matrices $\mathcal{P}_D$ and $\mathcal{P}_2$ can not be formed. For all test matrices in this example, the parameter α is the optimal one (the one with a minimum number of iterations) which has been found experimentally.

The numerical results presented in Table 3 show that the APSS preconditioner is very effective in reducing both the CPU time and the number of iterations for the convergence. Moreover, for the matrices YAO and liswet12 we see that the APSS preconditioner outperforms the two other preconditioners. It is necessary to mention that, although in some cases the matrices A and C are not of full rank, the APSS preconditioner works well.

Matrix, $\mathbf{n}, nnz$		No Prec.	$\mathcal{M}_\alpha$	$\mathcal{P}_D$	$\mathcal{P}_2$
YAO	α	-	0.5	-	-
$\mathbf{n} = 6004$	Iters	85	24	41	33
$nnz = 18006$	Time	0.17	0.07	0.08	0.07
	R_k	7.7e-7	8.6e-7	7.6e-7	8.2e-7
MOSARQP1	α	-	0.05	-	-
$\mathbf{n} = 5700$	Iters	115	6	NA	NA
$nnz = 14434$	Time	0.32	0.03	-	-
	R_k	8.9e-7	5.2e-7	-	-
STCQP2	α	-	0.25	-	-
$\mathbf{n} = 10246$	Iters	316	45	NA	NA
$nnz = 83979$	Time	1.30	0.48	-	-
	R_k	9.9e-7	7.1e-7	-	-
liswet12	α	-	0.5	-	-
$\mathbf{n} = 30004$	Iters	77	21	38	29
$nnz = 90006$	Time	0.82	0.22	1.28	1.04
	R_k	9.9e-7	9.0e-7	9.1e-7	7.4e-7
UBH1	α	-	0.095	-	-
$\mathbf{n} = 48018$	Iters	15969	1582	NA	NA
$nnz = 138021$	Time	261.32	66.78	-	-
	R_k	1.0e-7	1.0e-6	-	-
AUG2D	α	-	0.5	-	-
$\mathbf{n} = 50400$	Iters	77	25	NA	NA
$nnz = 140200$	Time	1.03	0.43	-	-
	R_k	9.6e-7	8.3e-7	-	-
CONT201	α	-	0.25	-	-
$\mathbf{n} = 120992$	Iters	183	26	NA	NA
$nnz = 489592$	Time	6.30	1.95	-	-
	R_k	9.4e-7	9.5e-7	-	-
CONT300	α	-	0.25	-	-
$\mathbf{n} = 271492$	Iters	143	19	NA	NA
$nnz = 1101892$	Time	1542	3.22	-	-
	R_k	9.6e-7	7.7e-7	-	-

Table 3: *Numerical results for GMRES to solve Example 3*

7. Conclusions

We have applied the APSS iteration method for solving three-by-three block saddle point problems. Convergence of the method has been proved. The induced preconditioner has been used to accelerate the convergence of the FGMRES method for solving the system. Numerical results showed that the proposed preconditioner is very effective. Moreover, we have seen that it outperforms two recently proposed preconditioners.

Acknowledgement

The authors would like to thank the referee for careful reading of the paper and giving useful comments and suggestions.

References

[1] Z. Z. Bai, G. H. Golub, M. K. Ng, *Hermitian and skew-Hermitian splitting methods for non-Hermitian positive definite linear systems*, SIAM J. Matrix Anal. Appl. **24**(2003), 603–626.

[2] M. Benzi, *Preconditioning techniques for large linear systems: A survey*, J. Comput. Phys. **182**(2002) 418–477.

[3] M. Benzi, G. H. Golub, *A preconditioner for generalized saddle point problems*, SIAM J. Matrix Anal. Appl. **26**(2004), 20–41.

[4] M. Benzi, G. H. Golub, J. Liesen, *Numerical solution of saddle point problems*, Acta Numer. **14**(2005), 1–137.

[5] M. Benzi, X. -P. Guo, *A dimensional split preconditioner for Stokes and linearized Navier-Stokes equations*, Appl. Numer. Math. **61**(2011), 66–76.

[6] D. P. Bertsekas, *Nonlinear Programming*, 2nd Ed., Athena Scientific, Athena, 1999.

[7] Y. Cao, *A block positive-semidefinite splitting preconditioner for generalized saddle point linear systems*, J. Comput. Appl. Math. **374**(2020), 112787.

[8] Y. Cao, *Shift-splitting preconditioners for a class of block three-by-three saddle point problems*, Appl. Math. Lett. **96**(2019), 40–46.

[9] Y. Cao, L. -Q. Yao, M. -Q. Jiang, *A modified dimensional split preconditioner for generalized saddle point problems*, J. Comput. Appl. Math. **250**(2013), 70–82.

[10] Z. Chen, Q. Du, J. Zou, *Finite element methods with matching and nonmatching meshes for Maxwell equations with discontinuous coefficients*, SIAM J. Numer. Anal. **37**(2000), 1542–1570.

[11] N. I. M. Gould, D. Orban, P. L. Toint, *CUTEr and SifDec, a constrained and unconstrained testing environment*, revisited, ACM Trans. Math. Softw. **29**(2003), 373–394.

[12] D. Han, X. Yuan, *Local linear convergence of the alternating direction method of multipliers for quadratic programs*, SIAM J. Numer. Anal. **51**(2013), 3446–3457.

[13] K. B. Hu, J. C. Xu, *Structure-preserving finite element methods for stationary MHD models*, arXiv:1503.06160.

[14] N. Huang, *Variable parameter Uzawa method for solving a class of block three-by-three saddle point problems*, Numer. Algor. **8**5(2020), 1233–1254.

[15] N. Huang, C. -F. Ma, *Spectral analysis of the preconditioned system for the* 3×3 *block saddle point problem*, Numer. Algor. **81**(2019), 421–444.

[16] N. Huang, Y. -H. Dai, Q. Hu, *Uzawa methods for a class of block three-by-three saddle-point problems*, Numer. Linear Algebra Appl. **26**(2019), e2265.

[17] Y. Ke, C. F. Ma, *A note on "A dimensional split preconditioner for Stokes and linearized Navier-Stokes equations"*, Appl. Numer. Math. **108**(2016), 223–225.

[18] Z. -Z. Liang, G. -F. Zhang, *Alternating positive semidefinite splitting preconditioners for double saddle point problems*, Calcolo **56**(2019), 26.

[19] G. I. Marchuk, *Methods of Numerical Mathematics*, Springer-Verlag, New York, 1984.

[20] Y. Saad, *Iterative methods for sparse linear systems*, PWS Press, New York, 1995.

[21] Y. Saad, *A flexible inner-outer preconditioned GMRES algorithm*, SIAM J. Sci. Comput. **14**(1993), 461–469.

[22] L. Wang, K. Zhang, *Generalized shift-splitting preconditioner for saddle point problems with block three-by-three structure*, OALib. Journal **6**(2019), e5968.

[23] X. Xie, H. -B. Li, *A note on preconditioning for the* 3×3 *block saddle point problem*, Comput. Math. Appl. **79**(2020), 3289–3296.