A SPECIAL PARAMETERIZED INEXACT UZAWA ALGORITHM FOR SYMMETRIC SADDLE POINT PROBLEM

by

Jun-Feng LU* and Li MA

Hangzhou Institute of Commerce, Zhejiang Gongshang University, Hangzhou, China

Original scientific paper
https://doi.org/10.2298/TSCI1804715L

In this paper, we consider a symmetric saddle point problem arising in the fluid dynamics. A special parameterized inexact Uzawa algorithm is proposed for solving the symmetric saddle point problem. The convergence of this special algorithm is considered. Sufficient conditions for the convergence are given. Numerical experiments resulting from stokes problem are presented to show the efficiency of the algorithm.

Key words: *saddle point problem, Stokes problem, Uzawa algorithm, preconditioner*

Introduction

We consider the symmetric saddle point problem:

$$\begin{bmatrix} A & B^T \\ B & -C \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} f \\ g \end{bmatrix} \tag{1}$$

where $A \in R^{n\times n}$ is symmetric positive definite, $B \in R^{m\times n}(m \le n)$ is of full row rank, and $C \in R^{m\times m}$ is symmetric positive definite, and $f \in R^n$ and $g \in R^m$ are two column vectors. Linear systems of the form (1) have widely applications in fluid dynamics, it results from the mixed finite element approximation of Stokes problem describing a slow incompressible viscous flow [1-4]. System (1) also arises from the 3-D coupled consolidation equation for modeling the displacement and pressure field of saturated porous media [5].

Since the coefficient matrix of eq. (1) is often large and sparse, iterative methods are more attractive than direct methods for solving system (1). In previous decades, a number of effective iterative methods have been discussed in the literature, such as Uzawa-type methods [6-8], SOR-like methods [9], HSS-type methods [10], and preconditioned Krylov subspace iteration method [11]. As a classical iteration method, Uzawa method has been paid many attentions to. Due to their simple implementation and minimal memory requirement, Uzawa-type algorithms are very suitable for solving the large and sparse saddle point problem [6-8, 12]. The efficiency of Uzawa-type algorithms depend on the choice of preconditioners and relaxed parameters. In this paper, we propose a strategy for choosing the preconditioners in the inexact Uzawa algorithm, which results in a special parameterized inexact Uzawa method (GPIUS). The sufficient conditions for the convergence of this special algorithm are given. Finally, a numerical example arising from stokes problem is provided to verify the efficiency of the GPIUS.

* Corresponding author, e-mail: ljfblue@hotmail.com

Notations. In the rest of this paper, $R^{m\times n}$ denotes the space of real $m\times n$ matrix. For any matrix $X \in R^{m\times n}$, X^T stands for its transpose. The norm $\|\cdot\|_2$ is 2-norm of a vector or matrix. Besides, diag(a, b) denotes the diagonal matrix with diagonal elements a and b, and tridiag(a, b, c) represents the tridiagonal matrix with subdiagonal a, diagonal b, and superdiagonal c.

The special parameterized inexact Uzawa algorithm

Theoretically, (1) is equivalent to:

$$\mathrm{A}\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} A & B^T \\ -B & C \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} f \\ -g \end{bmatrix} \tag{2}$$

Consider the following matrix splitting A = M – N with:

$$\mathrm{M} = \begin{bmatrix} P & O \\ -B+Q_1 & Q_2 \end{bmatrix} \text{ and } \mathrm{N} = \begin{bmatrix} P-A & -B^T \\ Q_1 & Q_2-C \end{bmatrix} \tag{3}$$

where $P \in R^{n\times n}$ and $Q_2 \in R^{m\times m}$ are prescribed symmetric positive definite matrices, and $Q_1 \in R^{m\times n}$ is an arbitrary matrix. Based on the splitting (3), Zhou and Zhang [13] proposed the following generalized parameterized inexact Uzawa (GPIU) algorithm.

Algorithm GPIU

$$x_{i+1} = x_i + P^{-1}\left[f - (Ax_i + B^T y_i) \right]$$

$$y_{i+1} = y_i + Q_2^{-1}(Bx_{i+1} - Cy_i - g) - Q_2^{-1}Q_1(x_{i+1} - x_i)$$

The matrices P and Q_2 can be seen as the approximations of A and the Schur complement $S = BA^{-1}B^T + C$, respectively. By choosing different parameterized matrices in GPIU, it covers the preconditioned Uzawa method [9], the inexact Uzawa method [7] and the parameterized inexact Uzawa method [6].

In order to further improve the convergence of GPIU, we consider $Q_1 = (\omega I + \tau Q_2)B$ with ω and $\tau \in R$, and obtain the following special parameterized inexact Uzawa method.

Algorithm GPIUS

$$x_{i+1} = x_i + P^{-1}\left[f - (Ax_i + B^T y_i) \right]$$

$$y_{i+1} = y_i + Q_2^{-1}\left[(1-\omega)Bx_{i+1} + \omega Bx_i - Cy_i - g\right] - \tau B(x_{i+1} - x_i) \tag{4}$$

We note that GPIU-S is the inexact Uzawa method in [14] if $C = 0$.

Convergence analysis of GPIUS

One can easily verify that the iteration matrix of GPIUS reads as:

$$\mathrm{T} = \begin{bmatrix} P & O \\ -B+Q_1 & Q_2 \end{bmatrix}^{-1}\begin{bmatrix} P-A & -B^T \\ Q_1 & Q_2-C \end{bmatrix} \tag{5}$$

To guarantee the convergence of GPIUS, it requires that the spectral radius of T is less than one, *i. e.* $\rho(T) < 1$. So we need to estimate the eigenvalues of the matrix T. In the following section, we consider the generalized eigenvalue problem:

$$\begin{bmatrix} P-A & -B^T \\ Q_1 & Q_2-C \end{bmatrix}\begin{bmatrix} u \\ v \end{bmatrix} = \lambda \begin{bmatrix} P & O \\ -B+Q_1 & Q_2 \end{bmatrix}\begin{bmatrix} u \\ v \end{bmatrix} \tag{6}$$

where λ is an eigenvalue of T, and $(u^T, v^T)^T$ is the corresponding eigenvalue vector. We focus on considering the eigenvalue analysis for T when $C = \delta Q_2$ with $\delta \geq 0$. In order to illustrate the convergence of GPIUS, we show a preliminary lemma in [15].

Lemma 1. Consider the quadratic equation $x^2 + bx + c = 0$, where b and c are real numbers. Both roots of the equation are less than one in modulus if and only if $|c| < 1$ and $|b| < 1 + c$.

If $C = \delta Q_2$, eq. (6) can be represented:

$$\left[(1-\lambda)P - A\right]u = B^T v \tag{7}$$

$$(1-\lambda)(\omega I + \tau Q_2)Bu + \lambda Bu = (\lambda - 1 + \delta)Q_2 v \tag{8}$$

We show the convergence results based on the following two cases, one is $\lambda = 1 - \delta$, and the other is $\lambda \neq 1 - \delta$.

We first consider the simple case when $\lambda = 1 - \delta$. If $\lambda = 1 - \delta$, (7) reduces to $\delta(\omega I + \tau Q_2)Bu + (1 - \delta)Bu = 0$, which implies that $u \in N[\delta(\omega I + \tau Q_2)B + (1 - \delta)B]$, where N represents the null space of the corresponding matrix. By (7), we have $v = (BP^{-1}B^T)^{-1}(\delta B - BP^{-1}A)u$. Thus $\lambda = 1 - \delta$ is an eigenvalue of the iteration matrix T.

If $\lambda \neq 1 - \delta$, by (8), it follows that:

$$v = \frac{1}{\lambda - 1 + \delta} Q_2^{-1}\left[(1-\lambda)(\omega I + \tau Q_2)B + \lambda B\right]u \tag{9}$$

By substituting eq. (9) to eq. (7), we have:

$$\left[(1-\lambda)P - A\right]u = \frac{1}{\lambda - 1 + \delta} B^T Q_2^{-1}\left[(1-\lambda)(\omega I + \tau Q_2)B + \lambda B\right]u$$

Multiplying both sides of the previous equation by left $u^T/u^T u$, we obtain:

$$\alpha\lambda^2 + \left[\gamma - \eta + (\delta - 2)\alpha + \beta\right]\lambda + \left[\eta + (1-\delta)\alpha - \beta(1-\delta)\right] = 0 \tag{10}$$

where

$$\alpha = \frac{u^T P u}{u^T u}, \quad \beta = \frac{u^T A u}{u^T u}, \quad \gamma = \frac{u^T B^T Q_2^{-1} B u}{u^T u}, \quad \theta = \frac{u^T B^T B u}{u^T u}, \quad \eta = \omega\gamma + \tau\theta \tag{11}$$

By *Lemma 1* together with eq. (10), $|\lambda| < 1$ holds if and only if:

$$0 < \delta < 2 \quad \text{and} \quad \frac{\gamma}{2} + (\delta - 2)\alpha + \frac{2-\delta}{2}\beta < \eta < \delta\alpha + \beta(1-\delta)$$

We summarize the previous analysis as the convergence theorem for GPIUS.

Theorem 1. Assume that P and Q_2 are symmetric positive definite matrices, and let $Bu(\omega I + \tau Q_2)B$, then algorithm GPIUS converges for the special case with $C = \delta Q_2$ if:

$$0 < \delta < 2 \quad \text{and} \quad \frac{\gamma}{2} + (\delta - 2)\alpha + \frac{2-\delta}{2}\beta < \eta < \delta\alpha + \beta(1-\delta)$$

where $\alpha, \beta, \gamma, \theta, \eta$ are defined by eq. (11).

Corollary 1. Under the assumptions of *Theorem 1*, Algorithm GPIUS converges for the linear system (1) with $C = \delta Q_2$ ($0 < \delta < 2$), if:

$$\left(\omega - \frac{1}{2}\right)B^T Q_2^{-1} B + \gamma B^T B + (2-\delta)P + \frac{\delta - 2}{2}A \quad \text{and} \quad \delta P + (1-\delta)A - \omega B^T Q_2^{-1} B \ \ -\tau B^T B$$

are positive definite matrices.

Corollary 2. Under the assumptions of *Theorem 1*, if $C = \delta Q_2$ ($0 < \delta < 2$), then the eigenvalues of the iterative matrix of algorithm GPIUS are:

$$\lambda = \frac{\eta - \gamma + (2-\delta)\alpha - \beta \pm \sqrt{\left[\eta - \gamma + (2-\delta)\alpha - \beta\right]^2 - 4\alpha\left[\eta + (1-\delta)\alpha - \beta(1-\delta)\right]}}{2\alpha}$$

Numerical experiments

In this section, we test a simulated saddle point problem to show the effectiveness of the GPIUS method. All the numerical computations are performed by MATLAB 2014 on PC with an Intel Core 2 Duo CPU, 2.4 GHz, and 8 GB RAM.

The Stokes problem is given by:

$$\begin{cases} -\Delta u + \nabla p = \tilde{f} \\ \nabla u = \tilde{g} \end{cases} \tag{12}$$

with the vector-valued velocity, u, and the scalar pressure function, p, over the square domain $\Omega = (0.1)\times(0,1)$. We set $u = 0$ on the boundary of Ω, and let $\int_\Omega p(x)\mathrm{d}x = 0$. By discretizing (12) with the upwind scheme, we obtain the linear system (1) with:

$$A = \begin{bmatrix} I_m \otimes T_m + T_m \otimes I_m & O \\ O & I_m \otimes T_m + T_m \otimes I_m \end{bmatrix} \in R^{2m^2 \times 2m^2}, \quad B = \begin{bmatrix} I_m \otimes F_m \\ F_m \otimes T_m \end{bmatrix} \in R^{2m^2 \times m^2}$$

where $T_m = 1/h^2 \mathrm{tridiag}(-1, 2, -1) \in R^{m\times m}$, $F_m = 1/h\mathrm{tridiag}(-1, 1, 0) \in R^{m\times m}$ with $\otimes$ being the Kronecker product symbol and $h = 1/(m + 1)$ the discretization mesh size [10]. Different with [10], we construct the sub-matrices A and B by setting $h = m + 1$. The matrix C is set as $C = \mathrm{diag}(m^2 : -1:1) \in R^{m^2\times m^2}$. The right-hand side vectors $\vec{f}$ and $\vec{g}$ are chosen such that the exact solution of (1) is the vectors $\vec{x}$ and $\vec{y}$ of all ones. Both the initial vectors $\vec{x}_0$ and $\vec{y}_0$ are set to be zero in this example. The iteration schemes are terminated if the current iteration satisfies:

$$\frac{\| r_i \|_2}{\| r_0 \|_2} \le 10^{-6}$$

where

$$r_i = \begin{pmatrix} f - Ax_i - B^T y_i \\ g - Bx_i + Cy_i \end{pmatrix}$$

is the residual vector of system (1) in the i^{th} iteration.

Table 1. The parameters for the preconditioners P and Q_1

Preconditioner	Q	γ	w	t	δ
A	diag(A)	0.2	0.49	–0.01	1.3333
B	tridiag(A)	0.1	0.45	–0.01	1.3333
C	RR^T	1.05	0.50	–0.01	1.1111

We provide two strategies for setting the preconditioner P. In the first case, the preconditioner P is set as $P = A + \gamma Q$, where $\gamma > 0$, and Q is symmetric positive definite. The matrix Q is defined in the following two ways: (a) Q = diag(A); (b) Q = tridiag(A). The preconditioner P is chosen as $P = \gamma RR^T$ in the second case (denoted by C), where R results from the incomplete Cholesky factorization of A with the drop tolerance 0.01 [16]. The preconditioner Q_1 is determined by $Q_1 = (\omega I + \tau Q_2)B$ with Q_2 satisfying $C = \delta Q_2$. Table 1 shows the parameters of the previous preconditioners for the considered grids. Table 2 lists the iteration numbers, CPU times, and the relative errors $\|r_i\|_2/\|r_0\|_2$ of the tested algorithms. We see that GPIU-S works well for this example. Specially, the algorithm GPIU-S with Q = tridiag(A) performs better than the GPIU-S with Q = diag(A). Figure 1 plots the error curves of the algorithm GPIUS when m = 64 or 128, respectively.

Table 2. Numerical results for GPIUS with different preconditioners

GPIUS	m = 64			m = 128		
	iteration	CPU	error	iteration	CPU	error
A	13	0.57	$9.53\cdot10^{-7}$	13	7.93	$8.01\cdot10^{-7}$
B	13	0.56	$8.64\cdot10^{-7}$	13	7.57	$7.46\cdot10^{-7}$
C	11	0.38	$9.57\cdot10^{-7}$	10	5.54	$9.43\cdot10^{-7}$

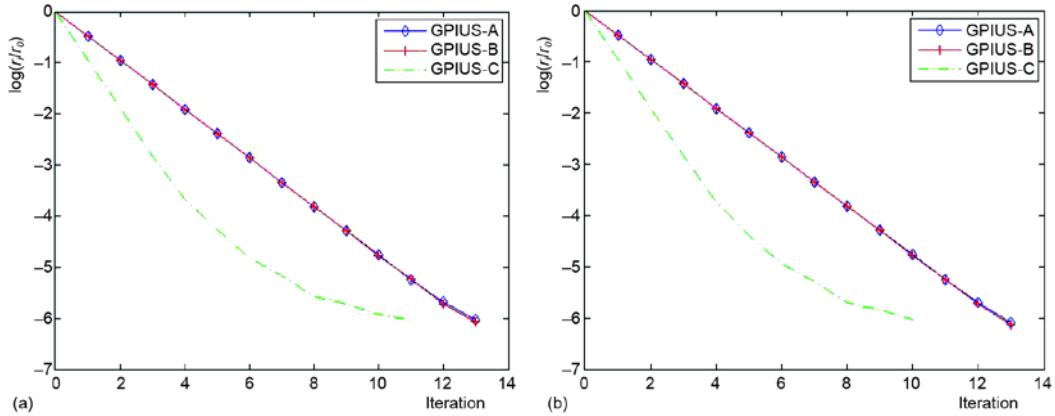

Figure 1. Error curves of GPIUS with m = 64 (a) and m = 128 (b)

In order to reduce the computational cost of GPIUS, we test the inexact solves for the cases A and B. The solution $\Phi = [f - (Ax_i + B^T y_i)]$ is determined by conjugate gradient method for the linear system $Ph = [f - (Ax_i + B^T y_i)]$. The numerical results for GPIUS with in-

exact solves are shown in tab. 3. The error curves of GPIUS are presented in fig. 2. We see that GPIUS with inexact solves works better than GPIUS with exact solves. In particular, GPIUS with Q = diag(A) works better than GPIUS with tridiag(A) when the inexact solves are used.

Table 3. Numerical results for GPIUS with inexact solves

GPIUS	m = 64			m = 128		
	iteration	CPU	error	iteration	CPU	error
A	13	0.53	$9.59\cdot10^{-7}$	13	6.89	$8.10\cdot10^{-7}$
B	13	0.54	$8.77\cdot10^{-7}$	13	7.10	$7.51\cdot10^{-7}$

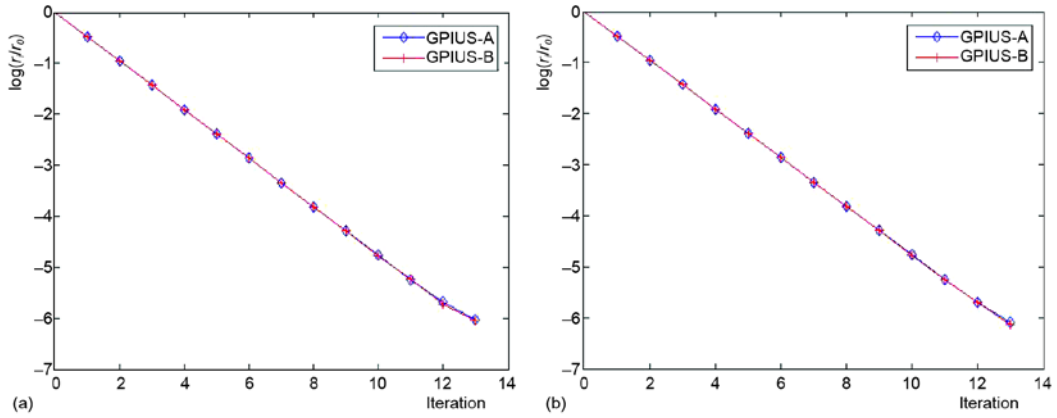

Figure 2. Error curves of GPIUS with inexact solves when m = 64 (a) and m = 128 (b)

Conclusion

This paper focuses on solving the symmetric saddle point problem by a special parameterized inexact Uzawa method. We give the convergence analysis of this special Uzawa algorithm. Numerical results show that GPIUS is efficient for solving the symmetric saddle point problem. However, it still remains an open problem that how to choose the optimal parameters such that the total computational cost of GPIUS can be as small as possible. We will investigate this topic in our future work.

Acknowledgment

The work is supported by the Natural Science Foundation of Zhejiang Province (LY17A010001), the project 17NDJC212YB of Philosophy and Social Science of Zhejiang Province and the project Y201636537 of Education Department of Zhejiang Province.

References

[1] Elman, H. C., *et al.*, *Finite Elements and Fast Iterative Solvers: With Applications in Incompressible Fluid Dynamics, Numerical Mathematics and Scientific Computation*, Oxford University Press, New York, USA, 2005

[2] Silvester, D., Wathen, A., Fast Iterative Solution of Stabilised Stokes Systems, Part II: Using General Block Preconditioners, *SIAM Journal on Numerical Analysis, 31* (1994), 5, pp. 1352-1367

[3] Hristov, J., Integral-Balance Solution to the Stokes' First Problem of a Viscoelastic Generalized Second Grade Fluid, *Thermal Science*, *16* (2012), 2, pp. 395-410
[4] Qiu, Y. Y., Solving a Class of Boundary Value Problems by LSQR, *Thermal Science*, *21* (2017), 4, pp. 1719-1724
[5] Boiti, M., *et al.*, Integrable Two-Dimensional Generalization of the Sine-and Sinh-Gordon Equations, *Inverse Problems*, *3* (1987), 1, pp. 37-49
[6] Bai, Z. Z., Wang, Z. Q., On Parameterized Inexact Uzawa Methods for Generalized Saddle Point Problems, *Linear Algebra and its Applications*, *428* (2008), 11-12, pp. 2900-2932
[7] Bramble, J. H., *et al*., Analysis of the Inexact Uzawa Algorithm for Saddle Point Problems, *SIAM Journal on Numerical Analysis*, *34* (1997), 3, pp. 1072-1092
[8] Lu, J. F., Zhang, Z. Y., A Modified Nonlinear Inexact Uzawa Algorithm with a Variable Relaxation Parameter for the Stabilized Saddle Point Problem, *SIAM Journal on Matrix Analysis and Applications*, *31* (2010), 4, pp. 1934-1957
[9] Chen, X. J., On Preconditioned Uzawa Methods and SOR Methods for Saddle-Point Problems, *Journal of Computational & Applied Mathematics*, *100* (1998), 2, pp. 207-224
[10] Bai, Z. Z., *et al*., Preconditioned Hermitian and Skew-Hermitian Splitting Methods for Non-Hermitian Positive Semidefinite Linear Systems, *Numerische Mathematik*, *98* (2004), 1, pp. 1-32
[11] Benzi, M., Golub, G. H., A Preconditioner for Generalized Saddle Point Problems, *SIAM Journal on Matrix Analysis and Applications*, *26* (2004), 1, pp. 20-41
[12] Benzi, M., *et al*., Numerical Solution of Saddle Point Problems, *Acta Numerica*, *14* (2005), May, pp. 1-137
[13] Zhou, Y. Y., Zhang, G. F., A Generalization of Parameterized Inexact Uzawa Method for Generalized Saddle Point Problems, *Applied Mathematics and Computation*, *215* (2009), 2, pp. 599-607
[14] Dou, Q. Y., Yin, J. F., A Class of Generalized Inexact Uzawa Methods for Saddle Point Problems, *Mathematica Numerica Sinica*, *34* (2011), 1, pp. 37-48
[15] Young, D. M., *Iterative Solution for Large Linear Systems*, Academic Press, New York, USA, 1971
[16] Elman, H. C., *et al*., Algorithm 866: IFISS, a Matlab Toolbox for Modelling Incompressible Flow, *ACM Transactions on Mathematical Software*, *33* (2007), 2, pp. 1-18

Paper submitted: January 25, 2017
Paper revised: December 16, 2017
Paper accepted: December 16, 2017