Journal of Computational Mathematics
Vol.31, No.4, 2013, 398–421.

http://www.global-sci.org/jcm
doi:10.4208/jcm.1304-m4209

A RELAXED HSS PRECONDITIONER FOR SADDLE POINT PROBLEMS FROM MESHFREE DISCRETIZATION*

Yang Cao
School of Transportation, Nantong University, Nantong 226019, China
Email: caoyangsqq@163.com
Linquan Yao
School of Urban Rail Transportation, Soochow University, Suzhou 215006, China
Email: lqyao@suda.edu.cn
Meiqun Jiang
School of Mathematical Sciences, Soochow University, Suzhou, 215006, China
Email: mqjiang@suda.edu.cn
Qiang Niu
Mathematics and Physics Center, Xi'an Jiaotong-Liverpool University, Suzhou 215123, China
Email: kangniu@gmail.com

Abstract

In this paper, a relaxed Hermitian and skew-Hermitian splitting (RHSS) preconditioner is proposed for saddle point problems from the element-free Galerkin (EFG) discretization method. The EFG method is one of the most widely used meshfree methods for solving partial differential equations. The RHSS preconditioner is constructed much closer to the coefficient matrix than the well-known HSS preconditioner, resulting in a RHSS fixed-point iteration. Convergence of the RHSS iteration is analyzed and an optimal parameter, which minimizes the spectral radius of the iteration matrix is described. Using the RHSS preconditioner to accelerate the convergence of some Krylov subspace methods (like GMRES) is also studied. Theoretical analyses show that the eigenvalues of the RHSS preconditioned matrix are real and located in a positive interval. Eigenvector distribution and an upper bound of the degree of the minimal polynomial of the preconditioned matrix are obtained. A practical parameter is suggested in implementing the RHSS preconditioner. Finally, some numerical experiments are illustrated to show the effectiveness of the new preconditioner.

Mathematics subject classification: 65F10.
Key words: Meshfree method, Element-free Galerkin method, Saddle point problems, Preconditioning, HSS preconditioner, Krylov subspace method.

1. Introduction

In recent years, meshfree (or meshless) methods have been developed rapidly as a class of potential computational techniques for solving partial differential equations. In the meshfree method, it does not require a mesh to discretize the problem domain, and the approximate solution is constructed entirely on a set of scattered nodes. A lot of meshfree methods have been proposed; see [27] for a general discussion. In this paper, we mainly consider the element-free Galerkin method [11], which is one of the most widely used meshfree methods.

* Received July 5, 2012 / Revised version received February 25, 2013 / Accepted April 16, 2013 / Published online July 9, 2013 /

The EFG method is almost identical to the conventional finite element method (FEM), as both of them are based on the Galerkin formulation, and employ local interpolation/approximation to approximate the trial function. The EFG method requires only a set of distributed nodes on the problem domain, while elements are used in the finite element method. Other key differences between the EFG method and the FEM method lie in the interpolation methods, integration schemes and in the enforcement of essential boundary conditions. The EFG method employs the moving least squares (MLS) approximation method to approximate the trial functions. One disadvantage of MLS approximation is that the shape functions obtained are lack of Kronecker delta function property, unless the weight functions used in the MLS approximation are singular at nodal points. Therefore, the essential boundary conditions in the EFG method can not be easily and directly enforced. Several approaches have been proposed for imposing the essential boundary conditions in the EFG method, such as Lagrange multiplier (LM) method [11, 28], penalty method [33], augmented Lagrangian (AL) method [31], coupled method [22] and so on. Using independent Lagrange multipliers to enforce essential boundary conditions is common in structural analysis when boundary conditions can not be directly applied. However, this method leads to a linear system of saddle point type and increases the number of unknowns. The penalty method is very simple to be implemented and yields a symmetric positive definite stiffness matrix, but the penalty parameter must be chosen appropriately. Moreover, the accuracy of the penalty method is less than that of the Lagrange multiplier method in general. The augmented Lagrangian method uses a generalized total potential energy function to impose essential boundary conditions. The augmented Lagrangian regularization used in the AL method is composed of the sum of the pure Lagrangian term and the penalty term. In fact, the AL method combines the LM method and the penalty method, but it leads to a better matrix structure than the LM method; see discussion in Section 2 and [31]. For other methods studied for imposing essential boundary conditions in the EFG method; see [22, 27] and references therein.

In this paper, we study the iterative solutions of large and sparse linear systems of equations arising from the EFG method with the AL method imposing essential boundary conditions. The discrete linear system of equations has the following block 2×2 form

$$\mathcal{A}x \equiv \begin{bmatrix} A & B^T \\ -B & 0 \end{bmatrix} \begin{bmatrix} u \\ \lambda \end{bmatrix} = \begin{bmatrix} f \\ g \end{bmatrix} \equiv b, \tag{1.1}$$

where $A = K + G \in \mathbb{R}^{n\times n}$ (K is the stiffness matrix and G is obtained from penalty term), $B \in \mathbb{R}^{m\times n}$ is obtained from Lagrangian term, $u \in \mathbb{R}^n$ is the approximation solution at nodes, $\lambda \in \mathbb{R}^m$ is the Lagrange multiplier, $f = f_1 + f_2 \in \mathbb{R}^n$ (f_1 is the stiffness vector and f_2 is obtained from penalty term), $g \in \mathbb{R}^m$ and $m \leq n$. n and m are related to the number of nodes in global problem domain and the number of nodes in essential boundary, respectively. In general, A is symmetric and positive definite and B has full rank. Under those conditions, we know that the solution of (1.1) exists and is unique.

The linear system (1.1) can be regarded as the saddle point problem. It frequently arises from computational fluid dynamics, mixed finite element of elliptic PDEs, constrained optimization, constrained least-squares problem and so on. In recent years, saddle point problems have been received considerable attention. A large amount of work has been devoted to developing efficient algorithms for solving saddle point problems. When the matrix blocks A and B are large and sparse, iterative methods become more attractive than direct methods for solving the saddle point problems (1.1), but direct methods play an important role in the form of preconditioners embedded in an iterative frame work. The best known and the oldest methods are the Uzawa algorithm [14], the inexact Uzawa algorithms [9, 10, 18], the HSS (Hermitian and skew-Hermitian splitting) algorithms [1–3, 6, 13] and so on. These methods are stationary iterative methods. They require much less computer memory than the Krylov subspace methods in actual implementations. But they are less efficient than the Krylov subspace methods in general. Unfortunately, Krylov subspace methods also tend to converge slowly when applied to saddle point problems (1.1), and good preconditioners are needed to achieve rapid convergence. Preconditioners can be constructed from matrix splitting iterative methods or matrix factorization. It can also be constructed by special structure of the coefficient matrix. But the preconditioner should be chosen as close as possible to the coefficient matrix, and the inverse of preconditioner should be computed easily. In the past few years, much work also has been devoted to developing efficient preconditioners for saddle point problems. A variety of preconditioners have been proposed and studied in many papers, such as block diagonal and block triangular preconditioners [7, 19, 20, 24], constraint preconditioners [8, 25], HSS-based preconditioners [3, 6, 12, 13, 30], dimensional split preconditioners [15, 16], $H-$matrix preconditioners [17] and so on. In [14], Benzi, Golub and Liesen discussed a selection of numerical methods and useful preconditioners for saddle point problems. But most existing efficient iterative methods and useful preconditioner are studied for solving saddle point problems arising from constrained optimization, computational dynamics and so on, there is little discussion on saddle point problems from meshfree discretization. When n and m are not large, Zheng et al. proposed a direct method based on Sherman-Morrison formula for solving saddle point problems from the EFG method [32]. When n and m are large, for solving saddle point problems from meshfree discretization [32], Leem et al. studied an algebraic multigrid method [26] and Borne et al. proposed a class of $H-$matrix preconditioners [17]. For saddle point problems arising from the EFG method, the class of preconditioners we considered in this paper is a class of HSS preconditioners, which was proposed by Benzi and Golub in [13]. The HSS preconditioner is induced by the HSS iterative method, which was first proposed by Bai, Golub and Ng in [4] for the solution of a broad class of non-Hermitian positive definite linear systems. It was demonstrated in [4] that the HSS iteration method converges unconditionally to the unique solution of the non-Hermitian system of linear equations. Then, Benzi and Golub applied the HSS iteration method for solving saddle point problems [13]. They also proved that the HSS iteration method converges unconditionally to the unique solution of saddle point problems. Due to its promising performance and elegant mathematical properties, the HSS iteration method has attracted many researchers' attention. Algorithmic variants and theoretical analyses of these HSS iteration methods for saddle point problems have been extensively and deeply discussed in [1, 3, 6, 12, 13, 21, 23, 30].

In this paper, based on the HSS preconditioner, a relaxed HSS (RHSS) preconditioner is proposed for saddle point problems (1.1) arising from the EFG discretization method. The RHSS preconditioner is constructed much closer to the coefficient matrix $\mathcal{A}$ than the HSS preconditioner, resulting in a RHSS fixed-point iteration. Convergence of the RHSS iteration is analyzed and an optimal parameter, which minimizes the spectral radius of the iteration matrix is described. Using the RHSS preconditioner to accelerate the convergence of some Krylov subspace methods (like GMRES) is also studied. Theoretical analyses show that the eigenvalues of the RHSS preconditioned matrix are real and located in a positive interval. Eigenvector distribution and an upper bound of the degree of the minimal polynomial of the preconditioned matrix are obtained. A practical parameter is suggested in implementing the

RHSS preconditioner. The remainder of the paper is organized as follows. In Section 2, a model problem and the element free Galerkin method are introduced. Then, in Section 3.1, the HSS preconditioner is reviewed briefly and a relaxed HSS preconditioner is constructed. The new preconditioner results in a RHSS fixed-point iteration. In Section 3.2, convergence of the RHSS iteration is analyzed and an optimal parameter is described. In Section 3.3, some properties of the RHSS preconditioned matrix are analyzed and a practical parameter is suggested. Implementation of the preconditioning steps of both HSS preconditioners and RHSS preconditioners are given in Section 3.4. In Section 4, some numerical experiments are presented to show the effectiveness of the new preconditioner. Finally, we end this paper with some conclusions in Section 5.

2. A Model Problem and the Element-free Galerkin Method

The model problem considered is a second-order partial differential equations defined on a domain $\Omega \subset \mathbb{R}^2$:

$$-\Delta u(\mathrm{x}) = -\Big(\frac{\partial^2 u(\mathrm{x})}{\partial x^2} + \frac{\partial^2 u(\mathrm{x})}{\partial y^2}\Big) = f(\mathrm{x}), \quad \text{in } \Omega, \tag{2.1}$$

where Δ is the Laplace operator, $u(\mathrm{x})$ is a unknown function, $f(\mathrm{x})$ is a given function of x and y, and Ω is a domain enclosed by $\Gamma = \partial\Omega = \Gamma_u \bigcup \Gamma_q$ with boundary conditions

$$u = \bar{u}, \qquad\qquad \text{on } \Gamma_u, \tag{2.2}$$

$$q(\mathrm{x}) = \frac{\partial u(\mathrm{x})}{\partial n} = \bar{q}(\mathrm{x}), \quad \text{on } \Gamma_q, \tag{2.3}$$

where n is the unit outward normal to the boundary Γ, $\bar{u}$ and $\bar{q}$ are the prescribed values of the function $u(\mathrm{x})$ and its normal derivative over the boundary Γ, respectively.

We use the element free Galerkin method to generate discretizations of the model problem (2.1)-(2.3). As discussed in Section 1, the EFG method is based, as the finite element method, on an integral formulation. This method, requiring only a set of nodes distributed on the problem domain, employs the moving least-squares (MLS) approximation for the construction of the shape functions. In the following, we first give a brief summary of the MLS approximation scheme, and then present the EFG method for the problem described in (2.1)-(2.3). For details of the EFG method, see [11,28].

2.1. The MLS approximation

Consider a sub-domain Ω_{x}, the neighborhood of a point x and denoted as the domain of definition of the MLS approximation for the trial function at x, which is located in the problem domain Ω. To approximate the distribution of function u in Ω_{x}, over a number of randomly located nodes $\{\mathrm{x}_i\}, i = 1, \cdots, n$, the moving least-squares (MLS) approximation $u^h(\mathrm{x})$ of u, $\forall \mathrm{x} \in \Omega_{\mathrm{x}}$, can be defined by

$$u^h(\mathrm{x}) = \sum_{j=1}^{l} p_j(\mathrm{x})\mathrm{a}_j(\mathrm{x}) = p^T(\mathrm{x})a(\mathrm{x}), \tag{2.4}$$

where $p^T(\mathrm{x}) = [p_1(\mathrm{x}), \cdots, p_l(\mathrm{x})]$ is a complete monomial basis of order l, and $\mathrm{a}_j(\mathrm{x})$ $(j = 1, \cdots, l)$ are coefficient of the basis functions.

The coefficient vector $a(\mathrm{x})$ can be obtained at any point x by minimizing a weighted discrete L_2 norm, which can be defined as

$$J(\mathrm{x}) = \sum_{i=1}^{n} w_i(\mathrm{x})[p^T(\mathrm{x}_i)a(\mathrm{x}) - \hat{u}_i]^2 = [Pa(\mathrm{x}) - \hat{u}]^T W[Pa(\mathrm{x}) - \hat{u}], \tag{2.5}$$

where $w_i(\mathrm{x}) = w(\mathrm{x} - \mathrm{x}_i)$ is the weight function associated with the node i, with $w_i(\mathrm{x}) > 0$ for all x in the support domain of $w_i(\mathrm{x})$, x_i denotes the value of x at node i, n is the number of nodes in Ω_{x}.

The matrices P and W in (2.5) are defined as

$$P = \begin{bmatrix} p_1(\mathrm{x}_1) & p_1(\mathrm{x}_2) & \cdots & p_1(\mathrm{x}_n) \\ p_2(\mathrm{x}_1) & p_2(\mathrm{x}_2) & \cdots & p_2(\mathrm{x}_n) \\ \vdots & \vdots & \ddots & \vdots \\ p_l(\mathrm{x}_1) & p_l(\mathrm{x}_2) & \cdots & p_l(\mathrm{x}_n) \end{bmatrix}, \quad W = \begin{bmatrix} w_1(\mathrm{x}) & 0 & \cdots & 0 \\ 0 & w_2(\mathrm{x}) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & w_n(\mathrm{x}) \end{bmatrix},$$

and $\hat{u} = [\hat{u}_1, \hat{u}_2, \cdots, \hat{u}_n]$. Here it should be noted that $\hat{u}_i (i = 1, 2, \cdots, n)$ are the fictitious nodal values, not the nodal values in general (see Fig. 1 for a simple one-dimensional case for the distinction between u_i and $\hat{u}_i$).

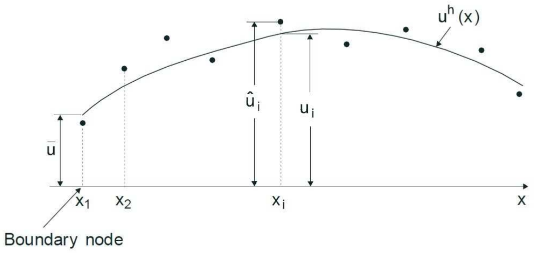

Fig. 2.1. The distinction between the nodal values u_i of the trial function $u^h(\mathrm{x})$ and the undetermined fictitious nodal values $\hat{u}_i$ in the MLS approximated trial function $u^h(\mathrm{x})$.

To find $a(\mathrm{x})$, we obtain the extremum of J in (2.5) by

$$\frac{\partial J}{\partial a} = D(\mathrm{x})a(\mathrm{x}) - E(\mathrm{x})\hat{u} = 0, \tag{2.6}$$

where $D(\mathrm{x}) = PWP^T$ and $E(\mathrm{x}) = PW$. If $D(\mathrm{x})$ is nonsingular, then from (2.6) we obtain

$$a(\mathrm{x}) = D^{-1}(\mathrm{x})E(\mathrm{x})\hat{u}.$$

Substituting $a(\mathrm{x})$ into (2.4), the expression of the local approximation $u^h(\mathrm{x})$ is thus

$$u^h(\mathrm{x}) = \Phi(\mathrm{x})\hat{u} = \sum_{i=1}^{n} \phi_i(\mathrm{x})\hat{u}_i, \tag{2.7}$$

where $\Phi(\mathrm{x}) = \{\phi_1(\mathrm{x}) \ \cdots \phi_n(\mathrm{x})\}$ is the shape function and

$$\phi_i(\mathrm{x}) = \sum_{j=1}^{l} p_j(\mathrm{x})[D^{-1}(\mathrm{x})E(\mathrm{x})]_{ji}.$$

The derivative of these shape functions can also be obtained

$$\phi_{i,k}(\mathrm{x}) = \sum_{j=1}^{l} [p_{j,k}(D^{-1}E)_{ji} + p_j(D_{,k}^{-1}E + D^{-1}E_{,k})_{ji}], \quad i = 1, \cdots, n,$$

where

$$D_{,k}^{-1} = -D^{-1}D_{,k}D^{-1},$$

and the index following a comma is a spatial derivative.

2.2. The EFG formulation

It can be found from above discussion that MLS approximation does not pass through the data used to fit the curve in general, i.e., it does not have the property of nodal interpolants as in the FEM, i.e. $\phi_i(\mathrm{x}_j) = \delta_{ij}$, where $\phi_i(\mathrm{x}_j)$ is the shape function corresponding to the node at x_i, evaluated at a nodal point x_j, and δ_{ij} is the Kronecker delta. Thus the essential boundary conditions can not be imposed directly in the EFG method. In this paper, the augmented Lagrangian method [31] is used to enforce the essential boundary conditions.

In Cartesian coordinate system, the functional corresponding to equation (2.1) associated to the boundary conditions (2.2) and (2.3) is given by

$$\begin{aligned}\Pi(u) = \frac{1}{2}\int_{\Omega} \Big((\frac{\partial u}{\partial x})^2 + (\frac{\partial u}{\partial y})^2 - 2uf\Big)d\Omega - \int_{\Gamma_q} \bar{q}ud\Gamma \\ + \int_{\Gamma_u} \lambda(u-\bar{u})d\Gamma + \frac{\beta}{2}\int_{\Gamma_u}(u-\bar{u})^T(u-\bar{u})d\Gamma,\end{aligned} \tag{2.8}$$

where λ is the Lagrange multiplier and β is the penalty parameter, they are used to impose the essential boundary conditions. The Lagrange multiplier λ can be expressed by

$$\lambda(s) = \sum_i N_i(s)\lambda_i, \tag{2.9}$$

where $N_i(s)$ is a Lagrange interpolation, s is the arc length along the boundary Γ_u, λ_i is the Lagrange multiplier at the ith node located at the essential boundary.

The necessary condition for (2.8) to reach its minimum yields

$$\begin{aligned}\delta_u\Pi(u,\lambda) = \int_{\Omega}[(\delta u_x)^T u_x + (\delta u_y)^T u_y - f^T\delta u]d\Omega - \int_{\Gamma_q}\bar{q}\delta u d\Gamma \\ + \textstyle\int_{\Gamma_u}\lambda^T\delta u d\Gamma + \beta\int_{\Gamma_u}[(\delta u)^T u - (\delta u)^T\bar{u}]d\Gamma = 0,\end{aligned} \tag{2.10}$$

$$\delta_\lambda\Pi(u,\lambda) = \int_{\Gamma_u}\delta\lambda(u-\bar{u})d\Gamma = 0, \tag{2.11}$$

where δu and $\delta\lambda$ are the test functions, $(\delta u)_x$ and $(\delta u)_y$ (u_x and u_y) are the derivatives of $\delta u(u)$ in terms of x and y, respectively.

Substituting (2.7) and (2.9) into (2.11) and (2.11), then integrating, we obtain the following discrete linear system of equations

$$\mathcal{A}x \equiv \begin{bmatrix} A & B^T \\ -B & 0 \end{bmatrix}\begin{bmatrix} u \\ \lambda \end{bmatrix} = \begin{bmatrix} f \\ g \end{bmatrix} \equiv b, \tag{2.12}$$

where

$$A=\int_\Omega G^TGd\Omega+\beta\int_\Omega\Phi^T\Phi d\Gamma,\quad B=\int_{\Gamma_u}N^T\Phi d\Gamma, \tag{2.13}$$

$$f=\int_{\Gamma_q}\Phi^T\bar{q}d\Gamma+\beta\int_{\Gamma_u}\Phi^T\bar{u}d\Gamma,\quad g=-\int_{\Gamma_u}N^T\bar{u}d\Gamma, \tag{2.14}$$

and

$$G=\left[\begin{array}{ccc}\frac{\partial\phi_1}{\partial x} & \cdots & \frac{\partial\phi_n}{\partial x}\\ \frac{\partial\phi_1}{\partial y} & \cdots & \frac{\partial\phi_n}{\partial y}\end{array}\right],\quad \Phi=\left[\begin{array}{ccc}\phi_1 & \cdots & \phi_n\end{array}\right],\quad N=\left[\begin{array}{ccc}N_1 & \cdots & N_m\end{array}\right].$$

It should be noted that u in (2.12) is $\hat{u}$ in fact. It is also the fictitious nodal value, not the nodal value in general. Once (2.12) has been solved, we can use (2.7) to get approximate values for every point of interest. Using the augmented Lagrangian method to enforce essential boundary conditions in the EFG method has some advantages [31], such as high precision, good stability and so on. Moreover, the blocks of discrete linear system (2.12) have some nice properties, such as A is symmetric and positive definite, B has full row rank. If there is no penalty used, i.e. using the Lagrange multipliers only, then A is symmetric and positive semi-definite in general. In the following, we will study the iterative solution of linear system (2.12). However, the coefficient matrix $\mathcal{A}$ is ill-conditioned in general for large n and m. Many iterative methods (for example splitting iterative methods, Krylov subspace methods) tend to converge very slowly when applied to saddle point system (2.12). In the next section, a relaxed HSS preconditioner will be proposed to achieve rapid convergence when some Krylov subspace methods (such as GMRES) are used. The corresponding iterative method is also studied.

3. The Relaxed HSS (RHSS) Preconditioner

In this section, we first review briefly the HSS iterative method and introduce the HSS preconditioner, for details; see [4,13,30]. Then a relaxed HSS preconditioner will be constructed. Some properties of the RHSS preconditioned matrix are studied.

3.1. The HSS preconditioner and the RHSS preconditioner

From the structure of the saddle point system (1.1), we know that the coefficient matrix $\mathcal{A}$ naturally possesses the Hermitian and skew-Hermitian splitting

$$\mathcal{A}=\mathcal{H}+\mathcal{S},$$

where

$$\mathcal{H}=\frac{1}{2}(\mathcal{A}+\mathcal{A}^T)=\left[\begin{array}{cc}A & 0\\ 0 & 0\end{array}\right],\quad \mathcal{S}=\frac{1}{2}(\mathcal{A}-\mathcal{A}^T)=\left[\begin{array}{cc}0 & B^T\\ -B & 0\end{array}\right].$$

Let $\alpha>0$ be a given parameter, I be the (appropriately dimensioned) identity matrix. It is easy to see that $\alpha I+\mathcal{H}$ and $\alpha I+\mathcal{S}$ are both nonsingular matrices. Consider two splittings of $\mathcal{A}$

$$\mathcal{A}=(\alpha I+\mathcal{H})-(\alpha I-\mathcal{S})\quad\text{and}\quad\mathcal{A}=(\alpha I+\mathcal{S})-(\alpha I-\mathcal{H}).$$

Then we can obtain the following HSS iterative method for solving saddle point problem (2.12).

Algorithm 3.1. *(The HSS iteration method)*
Given an initial guess x^0, for $k = 0, 1, 2, \cdots$, until $\{x^k\}$ converges, compute

$$\begin{cases} (\alpha I + \mathcal{H})x^{k+\frac{1}{2}} = (\alpha I - \mathcal{S})x^k + b, \\ (\alpha I + \mathcal{S})x^{k+1} = (\alpha I - \mathcal{H})x^{k+\frac{1}{2}} + b, \end{cases}$$

where α is a given positive constant.

It has been studied in [13] that the HSS iteration method is convergent unconditionally to the unique solution of the saddle point problem (2.12). In matrix-vector form, the above HSS iteration can be equivalently rewritten as

$$x^{k+1} = \tilde{\Gamma}_\alpha x^k + \mathcal{N}_\alpha b, \tag{3.1}$$

where

$$\begin{aligned} \tilde{\Gamma}_\alpha &= (\alpha I + \mathcal{S})^{-1}(\alpha I - \mathcal{H})(\alpha I + \mathcal{H})^{-1}(\alpha I - \mathcal{S}), \\ \mathcal{N}_\alpha &= 2\alpha(\alpha I + \mathcal{S})^{-1}(\alpha I + \mathcal{H})^{-1}. \end{aligned}$$

Here, $\tilde{\Gamma}(\alpha)$ is the iteration matrix of the HSS iteration. In fact, (3.1) also results from the splitting

$$\mathcal{A} = \tilde{\mathcal{P}}_{HSS} - \mathcal{N}_{HSS}$$

of the coefficient matrix $\mathcal{A}$, with

$$\tilde{\mathcal{P}}_{HSS} = \frac{1}{2\alpha}(\alpha I + \mathcal{H})(\alpha I + \mathcal{S}), \text{ and } \mathcal{N}_{HSS} = \frac{1}{2\alpha}(\alpha I - \mathcal{H})(\alpha I - \mathcal{S}).$$

We note that $\Gamma_\alpha = \tilde{\mathcal{P}}_{HSS}^{-1}\mathcal{N}_{HSS}$, $\mathcal{N}(\alpha) = \tilde{\mathcal{P}}_{HSS}^{-1}$, and $\tilde{\mathcal{P}}_{HSS}$ can be served as a preconditioner, called the HSS preconditioner, to the system of linear equations (1.1). It should be noted that the pre-factor $\frac{1}{2\alpha}$ in the HSS preconditioner $\tilde{\mathcal{P}}_{HSS}$ has no effect on the preconditioned system. For the purpose of analysis, we can take the HSS preconditioner as

$$\mathcal{P}_{HSS} = \frac{1}{\alpha}(\alpha I + \mathcal{H})(\alpha I + \mathcal{S}). \tag{3.2}$$

By performing the matrix multiplication on the right-hand side of (3.2), it follows that $\mathcal{P}_{HSS}$ has the following structure

$$\begin{aligned} \mathcal{P}_{HSS} &= \frac{1}{\alpha}\begin{bmatrix} A + \alpha I & 0 \\ 0 & \alpha I \end{bmatrix}\begin{bmatrix} \alpha I & B^T \\ -B & \alpha I \end{bmatrix} \\ &= \begin{bmatrix} A + \alpha I & B^T + \frac{1}{\alpha}AB^T \\ -B & \alpha I \end{bmatrix}. \end{aligned} \tag{3.3}$$

It follows from (2.12) and (3.3) that the difference between the preconditioner $\mathcal{P}_{HSS}$ and the coefficient matrix $\mathcal{A}$ is given by

$$\mathcal{R}_{HSS} = \mathcal{P}_{HSS} - \mathcal{A} = \begin{bmatrix} \alpha I & \frac{1}{\alpha}AB^T \\ 0 & \alpha I \end{bmatrix}. \tag{3.4}$$

From (3.4), we can see that when α tends to zero, the same as the diagonal blocks. However, the off-diagonal block becomes unbound. Hence, we should choose an ideal α to balance the weight of both parts (see [30]).

To obtain an improved invariant of the HSS preconditioner, we hope that the preconditioner $\mathcal{P}_{HSS}$ is as close as possible to the coefficient matrix $\mathcal{A}$. Based on this idea, we propose a relaxed HSS preconditioner as follows

$$\mathcal{P}_{RHSS} = \frac{1}{\alpha}\begin{bmatrix} A & 0 \\ 0 & \alpha I \end{bmatrix}\begin{bmatrix} \alpha I & B^T \\ -B & 0 \end{bmatrix} = \begin{bmatrix} A & \frac{1}{\alpha}AB^T \\ -B & 0 \end{bmatrix}. \tag{3.5}$$

We can easily see that the difference between $\mathcal{P}_{RHSS}$ and $\mathcal{A}$ is given by

$$\mathcal{R}_{RHSS} = \mathcal{P}_{RHSS} - \mathcal{A} = \begin{bmatrix} 0 & (\frac{1}{\alpha}A - I)B^T \\ 0 & 0 \end{bmatrix}. \tag{3.6}$$

Compared with the matrix $\mathcal{R}_{HSS}$, diagonal blocks of the matrix $\mathcal{R}_{RHSS}$ become zero matrices, while the nonzero off-diagonal becomes $(\frac{1}{\alpha}A - I)B^T$. This means that the relaxed HSS preconditioner $\mathcal{P}_{RHSS}$ should be closer to the coefficient matrix $\mathcal{A}$ than the original HSS preconditioner $\mathcal{P}_{HSS}$. This observation suggests that $\mathcal{P}_{RHSS}$ will lead to a better preconditioner than $\mathcal{P}_{HSS}$, since it gives a better approximation of the coefficient matrix $\mathcal{A}$. It should be noted that the RHSS preconditioner $\mathcal{P}_{RHSS}$ no longer relates to an alternating direction iteration method, but this fact is of no consequence when $\mathcal{P}_{RHSS}$ is used as a precodntioner for Krylov subspace method like GMRES. In fact, the RHSS preconditioner $\mathcal{P}_{RHSS}$ can be obtained by the following splitting of the coefficient matrix $\mathcal{A}$

$$\mathcal{A} = \begin{bmatrix} A & \frac{1}{\alpha}AB^T \\ -B & 0 \end{bmatrix} - \begin{bmatrix} 0 & (\frac{1}{\alpha}A - I)B^T \\ 0 & 0 \end{bmatrix} \equiv \mathcal{P}_{RHSS} - \mathcal{R}_{RHSS}, \tag{3.7}$$

which results in the following RHSS iteration method.

Algorithm 3.2. *(The RHSS iteration method)*
Let α be a given positive constant. Let $[u^0, \lambda^0]$ be an initial guess vector. For $k = 0, 1, 2, \cdots$, until certain stopping criteria satisfied, compute

$$\begin{bmatrix} A & \frac{1}{\alpha}AB^T \\ -B & 0 \end{bmatrix}\begin{bmatrix} u^{k+1} \\ \lambda^{k+1} \end{bmatrix} = \begin{bmatrix} 0 & (\frac{1}{\alpha}A - I)B^T \\ 0 & 0 \end{bmatrix}\begin{bmatrix} u^k \\ \lambda^k \end{bmatrix} + \begin{bmatrix} f \\ g \end{bmatrix}.$$

From (3.5), the RHSS iteration method can be solved by two steps. We may first solve the system of linear equations with the first coefficient matrix in (3.5), then solve the system of linear equations with the second coefficient matrix in (3.5). Details of implementation steps are studied in Section 3.4.

3.2. Analysis of the RHSS iteration

In this section, we deduce the convergence property of the RHSS iteration method and the optimal parameter α. Note that the iteration matrix of the RHSS iteration method is

$$\Gamma = \mathcal{P}_{RHSS}^{-1}\mathcal{R}_{RHSS} = \begin{bmatrix} A & \frac{1}{\alpha}AB^T \\ -B & 0 \end{bmatrix}^{-1}\begin{bmatrix} 0 & (\frac{1}{\alpha}A - I)B^T \\ 0 & 0 \end{bmatrix}. \tag{3.8}$$

Let $\rho(\Gamma)$ denote the spectral radius of Γ. Then the RHSS iteration method converges if and only if $\rho(\Gamma) < 1$. Let

$$P_1 = \begin{bmatrix} A & 0 \\ 0 & \alpha I \end{bmatrix}, \qquad P_2 = \begin{bmatrix} \alpha I & B^T \\ -B & 0 \end{bmatrix}.$$

It is easy to check that

$$P_1^{-1} = \begin{bmatrix} A^{-1} & 0 \\ 0 & \frac{1}{\alpha} I \end{bmatrix}, \quad P_2^{-1} = \begin{bmatrix} \frac{1}{\alpha} I - \frac{1}{\alpha} B^T(BB^T)^{-1}B & -B^T(BB^T)^{-1} \\ (BB^T)^{-1}B & \alpha(BB^T)^{-1} \end{bmatrix}. \tag{3.9}$$

Then the iteration matrix Γ can be rewritten by

$$\Gamma = \alpha P_2^{-1} P_1^{-1} \mathcal{R}_{RHSS} = \begin{bmatrix} 0 & (B^T(BB^T)^{-1}B - I)(A^{-1} - \frac{1}{\alpha} I)B^T \\ 0 & I - \alpha(BB^T)^{-1}BA^{-1}B^T \end{bmatrix}.$$

From the above expression, we know that the spectral radius of the RHSS iteration matrix Γ is

$$\rho(\Gamma) = \max_{1 \le i \le m} |1 - \alpha \mu_i|, \tag{3.10}$$

where μ_i $(1 \le i \le m)$ is the ith eigenvalue of $(BB^T)^{-1}(BA^{-1}B^T)$. In fact, the spectral radius (3.10) is similar to a spectral radius when a stationary Richardson iteration applied to the following linear system

$$(BB^T)^{-\frac{1}{2}}(BA^{-1}B^T)(BB^T)^{-\frac{1}{2}}x = b.$$

Let μ_1 and μ_m be the largest and smallest eigenvalues of $(BB^T)^{-1}(BA^{-1}B^T)$, respectively. If A is symmetric and positive definite, so is $(BB^T)^{-\frac{1}{2}}(BA^{-1}B^T)(BB^T)^{-\frac{1}{2}}$. Since $(BB^T)^{-1}(BA^{-1}B^T)$ is similar to $(BB^T)^{-\frac{1}{2}}(BA^{-1}B^T)(BB^T)^{-\frac{1}{2}}$, the smallest and largest eigenvalues of $(BB^T)^{-\frac{1}{2}}(BA^{-1}B^T)(BB^T)^{-\frac{1}{2}}$ are μ_m and μ_1, respectively. It is well known that Richardson's iteration converges for all α such that

$$0 < \alpha < \frac{2}{\mu_1}.$$

Furthermore, the spectral radius $\rho(\Gamma)$ is minimized by taking

$$\alpha_{opt} = \frac{2}{\mu_1 + \mu_m}. \tag{3.11}$$

Same results are true for the RHSS iteration method. We summarize them in the following theorem.

Theorem 3.1. *Let $A \in \mathbb{R}^{n \times n}$ be a symmetric and positive definite matrix, $B \in \mathbb{R}^{m \times n}$ have full row rank, and let α be a positive constant. Let $\rho(\Gamma)$ be the spectral radius of the RHSS iteration matrix. Then*

$$\rho(\Gamma) = \max_{1 \le i \le m} |1 - \alpha \mu_i|,$$

where μ_i is the ith eigenvalue of $(BB^T)^{-1}(BA^{-1}B^T)$. Let μ_1 and μ_m be the largest and smallest eigenvalues of $(BB^T)^{-1}(BA^{-1}B^T)$, respectively. If α satisfies

$$0 < \alpha < \frac{2}{\mu_1},$$

then the RHSS iteration method is convergent. The optimal α, which minimizes the spectral radius $\rho(\Gamma)$, is given by

$$\alpha_{opt} = \frac{2}{\mu_1 + \mu_m}.$$

The corresponding optimal spectral radius is

$$\rho_{opt}(\Gamma) = \frac{\mu_1 - \mu_m}{\mu_1 + \mu_m}.$$

In general, the asymptotic rate of convergence of the stationary iteration is governed by the spectral radius of the iteration matrix Γ, so it makes sense to try to choose the parameter α so as to make $\rho(\Gamma)$ as small as possible. In Theorem 3.1, the optimal parameter and its corresponding optimal spectral radius are presented. From the expression of the optimal spectral radius, we can see that if μ_1 is very close to μ_m, then $\rho_{opt}(\Gamma)$ is very close to zero and fast convergence will be obtained. But if $\mu_1 \gg \mu_m$, then $\rho_{opt}(\Gamma)$ is very close to 1 and the RHSS iteration will converge very slowly. Fortunately, the rate of convergence can be greatly improved by Krylov subspace acceleration. In other words, using $\mathcal{P}_{RHSS}$ as a preconditioner for some Krylov subspace methods (such as GMRES) may be a good choice. In fact, we can rewrite the RHSS iteration in correction form:

$$x^{k+1} = x^k + \mathcal{P}_{RHSS}^{-1} r_k, \quad r_k = b - \mathcal{A}x_k. \tag{3.12}$$

This will be useful when we consider Krylov subspace acceleration. Moreover, knowing convergent condition of the RHSS iteration method is very important, since it implies that the spectrum of the preconditioned matrix lies entirely in a circle centered at $(1, 0)$ with unity radius which is a desirable property for Krylov subspace acceleration. Using the optimal parameter α_{opt}, eigenvalues of preconditioned matrix may be more cluster than that of other cases. To obtain the optimal parameter α_{opt}, we need to solve the extreme eigenvalues of $(BB^T)^{-1}(BA^{-1}B^T)$. This is a difficult task. In the next section, some properties of the preconditioned matrix are studied and a practical parameter is suggested in implementing the RHSS preconditioner.

3.3. Analysis of the preconditioned matrix $\mathcal{P}_{RHSS}^{-1}\mathcal{A}$

The RHSS iteration method is a stationary iteration. It is very simple and very easy to implement. But if $\mu_1 \gg \mu_m$, the convergence of the RHSS iteration is typically too slow for the method to be competitive even with the optimal choice of the parameter α. In this section, we propose using the Krylov subspace method like GMRES to accelerate the convergence of the iteration.

It follows from (3.12) that the linear system $\mathcal{A}x = b$ is equivalent to the linear system

$$(I - \Gamma)x = \mathcal{P}_{RHSS}^{-1}\mathcal{A}x = c,$$

where $c = \mathcal{P}_{RHSS}^{-1}b$. This equivalent (left-preconditioned) system can be solved with GMRES. Hence, the matrix $\mathcal{P}_{RHSS}$ can be seen as a preconditioner for GMRES. Equivalently, we can say that GMRES is used to accelerate the convergence of the splitting iteration applied to $\mathcal{A}x = b$ [29]. In general, a clustered spectrum of the preconditioned matrix $\mathcal{P}_{RHSS}^{-1}\mathcal{A}$ often translates in rapid convergence of GMRES. In the following, we will deduce the eigenvalue distribution of the preconditioned matrix $\mathcal{P}_{RHSS}^{-1}\mathcal{A}$. Besides, the eigenvector distribution and an upper bound of the minimal polynomial of the preconditioned matrix are also discussed.

Theorem 3.2. *The eigenvalues of the preconditioned matrix $\mathcal{P}_{RHSS}^{-1}\mathcal{A}$ are given by 1 with multiplicity at least n. The remaining eigenvalues are positive real and located in*

$$[\alpha\mu_m,\ \alpha\mu_1],$$

where μ_1 and μ_m are the maximum and the minimum eigenvalues of matrix $(BB^T)^{-1}BA^{-1}B^T$.

Proof. From (3.9), we have

$$\begin{aligned}
&\mathcal{P}_{RHSS}^{-1}\mathcal{A} = \mathcal{P}_{RHSS}^{-1}(\mathcal{P}_{RHSS} - \mathcal{R}_{RHSS})\\
&= I - \mathcal{P}_{RHSS}^{-1}\mathcal{R}_{RHSS} = I - \alpha P_2^{-1}P_1^{-1}\mathcal{R}_{RHSS}\\
&= I - \alpha \begin{bmatrix} \frac{1}{\alpha}I - \frac{1}{\alpha}B^T(BB^T)^{-1}B & -B^T(BB^T)^{-1} \\ (BB^T)^{-1}B & \alpha(BB^T)^{-1} \end{bmatrix} \begin{bmatrix} A^{-1} & 0 \\ 0 & \frac{1}{\alpha}I \end{bmatrix} \begin{bmatrix} 0 & (\frac{1}{\alpha}A - I)B^T \\ 0 & 0 \end{bmatrix}
\end{aligned}$$

$$\begin{aligned}
&= I - \begin{bmatrix} I - B^T(BB^T)^{-1}B & -\alpha B^T(BB^T)^{-1} \\ \alpha(BB^T)^{-1}B & \alpha^2(BB^T)^{-1} \end{bmatrix} \begin{bmatrix} 0 & (\frac{1}{\alpha}I - A^{-1})B^T \\ 0 & 0 \end{bmatrix}\\
&= I - \begin{bmatrix} 0 & (B^T(BB^T)^{-1}B - I)(A^{-1} - \frac{1}{\alpha}I)B^T \\ 0 & I - \alpha(BB^T)^{-1}BA^{-1}B^T \end{bmatrix}\\
&= \begin{bmatrix} I & (I - B^T(BB^T)^{-1}B)(A^{-1} - \frac{1}{\alpha}I)B^T \\ 0 & \alpha(BB^T)^{-1}BA^{-1}B^T \end{bmatrix}. \qquad (3.13)
\end{aligned}$$

Eq. (3.13) implies that the eigenvalues of the preconditioned matrix $\mathcal{P}_{RHSS}^{-1}\mathcal{A}$ are given by 1 with multiplicity at least n, the remaining nonunit eigenvalues are the nonunit eigenvalues of $\alpha(BB^T)^{-1}BA^{-1}B^T$. Since BB^T and $BA^{-1}B^T$ are symmetric and positive definite, and α is a positive parameter, the eigenvalues of $\alpha(BB^T)^{-1}BA^{-1}B^T$ are positive real. Denote μ_1 and μ_m be the maximum and the minimum eigenvalues of matrix $(BB^T)^{-1}BA^{-1}B^T$, then the remaining nonunit eigenvalues of the preconditioned matrix $\mathcal{P}_{RHSS}^{-1}\mathcal{A}$ are located in $[\alpha\mu_m,\ \alpha\mu_1]$.

This completes the proof. □

From Theorem 3.1, we know that eigenvalues of the preconditioned matrix are 1 with multiplicity at least n, and the remaining eigenvalues are real and located in a positive interval, which is related to the parameter α. Like all parameter based iterative methods, it is very difficult to obtain the optimal choice of the parameter. The optimal parameter α_{opt} (3.11), which minimizes the spectral radius of the RHSS iteration matrix Γ, can be used. In this case, all the eigenvalues of the preconditioned matrix $\mathcal{P}_{RHSS}^{-1}\mathcal{A}$ lie entirely in a circle centered at $(1, 0)$ with unity radius which is a desirable property for Krylov subspace acceleration. To obtain the optimal parameter α_{opt}, we need to solve the extreme eigenvalues of $(BB^T)^{-1}(BA^{-1}B^T)$. This should be a difficult task. Fortunately, from Theorem 3.2, we know that the parameter has no effect on the ratio of the interval $[\alpha\mu_m,\ \alpha\mu_1]$. This means that $\alpha = 1$ may be a good choice for the RHSS preconditioner. In the next section, some numerical experiments will show this observation. It should be noted that $\alpha = 1$ may be not a good choice for the RHSS iteration method, since the RHSS iteration method may be divergence in this case.

In the following, we turn to study the eigenvector distribution and an upper bound of the degree of the minimal polynomial of the preconditioned matrix $\mathcal{P}_{RHSS}^{-1}\mathcal{A}$.

Theorem 3.3. *Let the RHSS preconditioner be defined in* (3.5)*, then the preconditioned matrix $\mathcal{P}_{RHSS}^{-1}\mathcal{A}$ has $n + i + j$ linearly independent eigenvectors. There are*

- *n eigenvectors of the form $\begin{bmatrix} u^T & 0^T \end{bmatrix}^T$ that correspond to the eigenvalue 1.*
- *i ($1 \le i \le m$) eigenvectors of the form $\begin{bmatrix} u^T & v^T \end{bmatrix}^T$ arising from $(\frac{1}{\alpha}A - I)B^Tv = 0$ for which i vectors v are linearly independent and the eigenvalue is 1.*
- *j ($1 \le j \le m$) eigenvectors of the form $\begin{bmatrix} u^T & v^T \end{bmatrix}^T$ that correspond to nonunit eigenvalues.*

Proof. The form of the eigenvectors of the preconditioned matrix $\mathcal{P}_{RHSS}^{-1}\mathcal{A}$ can be derived by considering the following generalized eigenvalue problem

$$\begin{bmatrix} A & B^T \\ -B & 0 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \theta \begin{bmatrix} A & \frac{1}{\alpha}AB^T \\ -B & 0 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix}, \tag{3.14}$$

where θ is an eigenvalue of the preconditioned matrix $\mathcal{P}_{RHSS}^{-1}\mathcal{A}$ and $\begin{bmatrix} u^T & v^T \end{bmatrix}^T$ is the corresponding eigenvector. Expanding (3.14) out, we obtain

$$Au + B^Tv = \theta Au + \frac{\theta}{\alpha}AB^Tv, \tag{3.15}$$

$$(1-\theta)Bu = 0. \tag{3.16}$$

Eq. (3.16) implies that either $\theta = 1$ or $Bu = 0$. If $\theta = 1$, then (3.15) can be rewritten as

$$\Big(\frac{1}{\alpha}A - I\Big)B^Tv = 0. \tag{3.17}$$

Eq. (3.17) is trivially satisfied by $v = 0$, and hence there are n linearly independent eigenvectors of the form $\begin{bmatrix} u^T & 0^T \end{bmatrix}^T$ associated with the eigenvalue 1. If there exists any $v \neq 0$ which satisfies (3.17), then there will be i ($1 \le i \le m$) linearly independent eigenvectors of the form $\begin{bmatrix} u^T & v^T \end{bmatrix}^T$, where the components v arise from (3.17).

If $\theta \neq 1$, then from (3.15), we have

$$u = \frac{1}{1-\theta}\Big(\frac{\theta}{\alpha}I - A^{-1}\Big)B^Tv. \tag{3.18}$$

Substituting (3.18) into (3.16), we get

$$\alpha BA^{-1}B^Tv = \theta BB^Tv. \tag{3.19}$$

For this case, it must be $v \neq 0$. Otherwise, from (3.18) we get $u = 0$, a contradiction. If there exists any $v \neq 0$ which satisfies (3.19), then there will be j ($1 \le j \le m$) linearly independent eigenvectors of the form $\begin{bmatrix} u^T & v^T \end{bmatrix}^T$ that correspond to nonunit eigenvalues. Thus, the form of the eigenvectors are obtained. Let

$$a^{(1)} = \begin{bmatrix} a_1^{(1)} & \cdots & a_n^{(1)} \end{bmatrix}^T, \quad a^{(2)}= \begin{bmatrix} a_1^{(2)} & \cdots & a_i^{(2)} \end{bmatrix}^T, \quad a^{(3)}= \begin{bmatrix} a_1^{(3)} & \cdots & a_j^{(3)} \end{bmatrix}^T,$$

be three vectors. To show that the $n+i+j$ eigenvectors of the preconditioned matrix $\mathcal{P}_{RHSS}^{-1}\mathcal{A}$

are linearly independent, we need to show that

$$\begin{bmatrix} u_1^{(1)} & \cdots & u_n^{(1)} \\ 0 & \cdots & 0 \end{bmatrix} \begin{bmatrix} a_1^{(1)} \\ \vdots \\ a_n^{(1)} \end{bmatrix} + \begin{bmatrix} u_1^{(2)} & \cdots & u_i^{(2)} \\ v_1^{(2)} & \cdots & v_i^{(2)} \end{bmatrix} \begin{bmatrix} a_1^{(2)} \\ \vdots \\ a_i^{(2)} \end{bmatrix}$$

$$+ \begin{bmatrix} u_1^{(3)} & \cdots & u_j^{(3)} \\ v_1^{(3)} & \cdots & v_j^{(3)} \end{bmatrix} \begin{bmatrix} a_1^{(3)} \\ \vdots \\ a_j^{(3)} \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}, \tag{3.20}$$

implies that the vectors $a^{(k)}$ $(k = 1, 2, 3)$ are zero vectors. Recalling that in (3.20) the first matrix arises from the case $\theta_k = 1$ $(k = 1, \cdots, n)$, the second matrix from the case $\theta_k = 1$ $(k = 1, \cdots, i)$, and the last matrix from the case $\theta_k \neq 1$ $(k = 1, \cdots, j)$. Multiplying (3.20) by $\mathcal{P}_{RHSS}^{-1}\mathcal{A}$, gives

$$\begin{bmatrix} u_1^{(1)} & \cdots & u_n^{(1)} \\ 0 & \cdots & 0 \end{bmatrix} \begin{bmatrix} a_1^{(1)} \\ \vdots \\ a_n^{(1)} \end{bmatrix} + \begin{bmatrix} u_1^{(2)} & \cdots & u_i^{(2)} \\ v_1^{(2)} & \cdots & v_i^{(2)} \end{bmatrix} \begin{bmatrix} a_1^{(2)} \\ \vdots \\ a_i^{(2)} \end{bmatrix}$$

$$+ \begin{bmatrix} u_1^{(3)} & \cdots & u_j^{(3)} \\ v_1^{(3)} & \cdots & v_j^{(3)} \end{bmatrix} \begin{bmatrix} \theta_1 a_1^{(3)} \\ \vdots \\ \theta_j a_j^{(3)} \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}. \tag{3.21}$$

Subtracting (3.20) from (3.21), we obtain

$$\begin{bmatrix} u_1^{(3)} & \cdots & u_j^{(3)} \\ v_1^{(3)} & \cdots & v_j^{(3)} \end{bmatrix} \begin{bmatrix} (\theta_1 - 1) a_1^{(3)} \\ \vdots \\ (\theta_j - 1) a_j^{(3)} \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}.$$

Since the fires matrix above are j linearly independent vectors and the eigenvalues $\theta_k \neq 1$ $(k = 1, \cdots, j)$, we have $a_k^{(3)} = 0$ $(k = 1, \cdots, j)$. We also have linear independence of $v_k^{(2)}$ $(k = 1, \cdots, i)$, and thus $a_k^{(2)} = 0$ $(k = 1, \cdots, i)$. Thus, (3.20) can be simplified to

$$\begin{bmatrix} u_1^{(1)} & \cdots & u_n^{(1)} \\ 0 & \cdots & 0 \end{bmatrix} \begin{bmatrix} a_1^{(1)} \\ \vdots \\ a_n^{(1)} \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}.$$

Because of the linear independence of $u_k^{(1)}$ $(k = 1, \cdots, n)$, we have $a_k^{(1)} = 0$ $(k = 1, \cdots, n)$. Thus, the proof is complete. □

Theorem 3.4. *Let the RHSS preconditioner be defined in* (3.5)*, then the degree of the minimal polynomial of the preconditioned matrix* $\mathcal{P}_{RHSS}^{-1}\mathcal{A}$ *is at most* $m+1$. *Thus, the dimension of the Krylov subspace* $\mathcal{K}(\mathcal{P}_{RHSS}^{-1}\mathcal{A}, b)$ *is at most* $m + 1$.

Proof. By (3.13), we know that the preconditioned matrix $\mathcal{P}_{RHSS}^{-1}\mathcal{A}$ takes the form

$$\mathcal{P}_{RHSS}^{-1}\mathcal{A} = \begin{bmatrix} I & \Theta_2 \\ 0 & \Theta_1 \end{bmatrix}, \tag{3.22}$$

where $\Theta_1 = \alpha(BB^T)^{-1}BA^{-1}B^T \in \mathbb{R}^{m\times m}$ and $\Theta_2 = (I - B^T(BB^T)^{-1}B)(A^{-1} - \frac{1}{\alpha}I)B^T \in \mathbb{R}^{n\times m}$.

From the eigenvalue distribution studied in Theorem 3.2, it is evident that the characteristic polynomial of the preconditioned matrix $\mathcal{P}_{RHSS}^{-1}\mathcal{A}$ is

$$(\mathcal{P}_{RHSS}^{-1}\mathcal{A} - I)^n \prod_{i=1}^{m}(\mathcal{P}_{RHSS}^{-1}\mathcal{A} - \mu_i I).$$

Expanding the polynomial $(\mathcal{P}_{RHSS}^{-1}\mathcal{A} - I)\prod_{i=1}^{m}(\mathcal{P}_{RHSS}^{-1}\mathcal{A} - \mu_i I)$ of degree $m+1$, we obtain

$$(\mathcal{P}_{RHSS}^{-1}\mathcal{A} - I)\prod_{i=1}^{m}(\mathcal{P}_{RHSS}^{-1}\mathcal{A} - \mu_i I) = \begin{bmatrix} 0 & \Theta_2 \prod_{i=1}^{m}(\Theta_1 - \mu_i I) \\ 0 & (\Theta_1 - I)\prod_{i=1}^{m}(\Theta_1 - \mu_i I) \end{bmatrix}.$$

Since μ_i $(i = 1, \cdots, m)$ are also the eigenvalues of $\Theta_1 \in \mathbb{R}^{m\times m}$, thus we have

$$\prod_{i=1}^{m}(\Theta_1 - \mu_i I) = 0.$$

Therefore, we obtain the degree of the minimal polynomial of the preconditioned matrix $\mathcal{P}_{RHSS}^{-1}\mathcal{A}$ is at most $m+1$. From [29, Proposition 6.1], we know that the degree of the minimal polynomial is equal to the dimension of the corresponding Krylov subspace $\mathcal{K}(\mathcal{P}_{RHSS}^{-1}\mathcal{A}, b)$ (for general b). So, the dimension of the Krylov subspace $\mathcal{K}(\mathcal{P}_{RHSS}^{-1}\mathcal{A}, b)$ is also at most $m+1$. Thus, we complete the proof. □

It directly follows from Theorem 3.4 that any Krylov subspace iterative method with an optimality or Galerkin property, for example GMRES [29], will terminate in at most $m+1$ iterations with the solution to a linear system of the form (2.12) if the RHSS preconditioner $\mathcal{P}_{RHSS}$ is used. It should be noted that for the linear system of equations (2.12), the dimension m is related to the number of nodes located at the essential boundary and n is related to the number of nodes located in global problem domain. In general, m is much less than n. It seems that the RHSS preconditioner $\mathcal{P}_{RHSS}$ should be very effective for the saddle point problems arising from the element free Galerkin discretization method.

3.4. Implementation

In this section, for saddle point problems (2.12) we use the relaxed HSS preconditioner $\mathcal{P}_{RHSS}$ to derive a preconditioning implementation in a Krylov subspace iterative method, such as GMRES. To compare with the RHSS preconditioner, the preconditioning steps of the HSS preconditioner $\mathcal{P}_{HSS}$ are also given.

Since the pre-factor $\frac{1}{\alpha}$ in the RHSS preconditioner $\mathcal{P}_{RHSS}$ has no effect on the preconditioned system, application of the RHSS preconditioner with GMRES, we need to solve the linear system of the form

$$\begin{bmatrix} A & 0 \\ 0 & \alpha I \end{bmatrix}\begin{bmatrix} \alpha I & B^T \\ -B & 0 \end{bmatrix} z = r,$$

for a given r at each iteration. Let $r = [r_1^T,\ r_2^T]^T$ and $z = [z_1^T,\ z_2^T]^T$, where $r_1,\ z_1 \in \mathbb{R}^n$ and $r_2,\ z_2 \in \mathbb{R}^m$, then by (3.9), we have

$$\begin{aligned}\begin{bmatrix} z_1 \\ z_2 \end{bmatrix} &= \begin{bmatrix} \alpha I & B^T \\ -B & 0 \end{bmatrix}^{-1} \begin{bmatrix} A & 0 \\ 0 & \alpha I \end{bmatrix}^{-1} \begin{bmatrix} r_1 \\ r_2 \end{bmatrix} \\ &= \begin{bmatrix} I & -\frac{1}{\alpha}B^T \\ 0 & I \end{bmatrix} \begin{bmatrix} \frac{1}{\alpha}I & 0 \\ 0 & \alpha(BB^T)^{-1} \end{bmatrix} \begin{bmatrix} I & 0 \\ \frac{1}{\alpha}B & I \end{bmatrix} \begin{bmatrix} A^{-1} & 0 \\ 0 & \frac{1}{\alpha}I \end{bmatrix} \begin{bmatrix} r_1 \\ r_2 \end{bmatrix}.\end{aligned} \tag{3.23}$$

Hence, we can derive the following algorithm for implementing RHSS preconditioner.

Algorithm 3.3. *For a given vector $r = (r_1^T, r_2^T)^T$, we can compute the vector $z = (z_1^T,\ z_2^T)^T$ by* (3.23) *from the following steps:*

```
(1)solve $Au_1 = r_1$;
(2) $u_2 = \frac{1}{\alpha}(Bu_1 + r_2)$;
(3) solve $BB^T z_2 = \alpha u_2$;
(4) $z_1 = \frac{1}{\alpha}(u_1 - B^T z_2)$.
```

In Algorithm 3.3, it is required to solve two sub-linear systems with coefficient matrices A and BB^T exactly. However, this may be very costly and impractical in actual implementation. In fact, it is important to stress that these two sub-linear systems need not be solved exactly. The upshot is that inexact solvers can be used to greatly reduce the cost of each iteration, at the expense of somewhat slower convergence. Typically, in practical implementations, inexact solvers result in a much more competitive algorithm. Fortunately, under the assumptions given in Section 2, we know that both A and BB^T are symmetric and positive definite matrices. Therefore, we can employ the conjugate gradient (CG) or the preconditioned conjugate gradient (PCG) method to solve two sub-linear systems with the coefficient matrices A and BB^T to a prescribed accuracy at each step of the preconditioning steps.

Similarity, for implementing the HSS preconditioner with GMRES, we need to solve the linear system of the form

$$(\alpha I + \mathcal{H})(\alpha I + \mathcal{S})z = \begin{bmatrix} A + \alpha I & 0 \\ 0 & \alpha I \end{bmatrix} \begin{bmatrix} \alpha I & B^T \\ -B & \alpha I \end{bmatrix} z = r,$$

for a given r at each iteration. We can also do the matrix factorization of $\alpha I + \mathcal{S}$ and derive the following algorithm.

Algorithm 3.4. *For a given vector $r = (r_1^T, r_2^T)^T$, we can compute the vector $z = (z_1^T,\ z_2^T)^T = (\alpha I + \mathcal{S})^{-1}(\alpha I + \mathcal{H})^{-1} r$ from the following steps:*

```
(1) solve $(\alpha I + A)u_1 = r_1$;
(2) $u_2 = \frac{1}{\alpha}(Bu_1 + r_2)$;
(3) solve $(\alpha I + \frac{1}{\alpha}BB^T)z_2 = u_2$;
(4) $z_1 = \frac{1}{\alpha}(u_1 - B^T z_2)$.
```

Since the matrix $\alpha I + A$ and $\alpha I + \frac{1}{\alpha}BB^T$ are symmetric and positive definite matrices, we could also use a subroutine of the CG method to obtain u_1 and z_2.

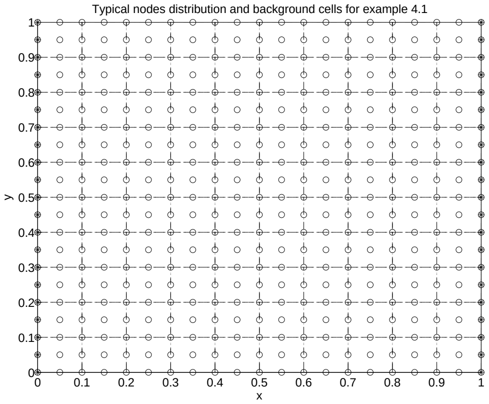

Fig. 4.1. Example 4.1: 21 × 21 regular nodes distribution and 10 × 10 background cells.

From Algorithm 3.3 and Algorithm 3.4, we can see that two steps are different. In the first step and the third step of Algorithm 3.3, we need to solve linear systems with coefficient matrix A and BB^T, respectively, while in the first and the third step of Algorithm 3.4, we need to solve linear systems with coefficient matrix $\alpha I + A$ and $\alpha I + \frac{1}{\alpha}(BB^T)$, respectively. This is a simple modification, but it may result in good convergence. In the next section, some numerical experiments are illustrated to show this observation.

4. Numerical Experiments

In this section, two numerical examples are illuatrated to show the effectiveness of the RHSS iteration method (Algorithm 3.2) and the RHSS preconditioner (3.5) for saddle point problem (2.12), from the point of view of both number of total iteration steps (denoted by 'IT') and elapsed CPU time in seconds (denoted by 'CPU'). In actual computations, all runs are started from the initial vector $(u_0^T, \lambda_0^T) = [0, \cdots, 0]^T$, and terminated if the current iterations satisfy ERR= $||r_k||_2/||r_0||_2 \leq 10^{-8}$ (where $r_k = b - \mathcal{A}x_k$ is the residual at the kth iteration) or if the number of prescribed iteration $k_{max} = 1000$ is exceeded. All runs are performed in MATLAB 7.0 on an Intel core i5 (2G RAM) Windows 7 system.

Model problems are discretized by the EFG method. Using the augmented Lagrangian method to impose essential boundary conditions leads to saddle point problems (2.12). The linear basis $p^T = [1,\ x,\ y]$ and the cubic spline weight function [28] are used to construct MLS shape functions. To obtain integrals in (2.13) and (2.14), background cells will be used in this paper. It should be noted that these background cells are independent of nodes distribution. For each background cell, 4 × 4 Gauss points are used. For each computational point (node or Gauss point), rectangular influence domain is adopted. The radius of each influence is taken as $\gamma \times dc$, where γ is the scale and dc is the average node distance. In this paper, we take $\gamma = 2.5$. Taking the penalty $\beta = 10$ in (2.13) and (2.14) to ensure that (1,1) block A of saddle point matrix is positive definite.

Example 4.1. The first numerical example is the Poisson equation

$$-\Delta u = 2\pi^2 sin\pi x cos\pi y \quad \text{in} \quad \Omega = [0,1] \times [0,1]$$

Table 4.1: Values of m, n and the order of $\mathcal{A}$ for Example 4.1.

Nodes	21×21	41×41	61×61	81×81
n	441	1681	3721	6561
m	42	82	122	162
$n+m$	483	1763	3843	6723

Table 4.2: Parameters for Example 4.1.

Nodes	21×21	41×41	61×61	81×81
$2/\mu_1$	0.67203	0.65661	0.65368	0.65264
α_{opt}	0.54657	0.53623	0.53425	0.53355
$[\alpha\mu_m,\ \alpha\mu_1]$	[0.37338, 1.62662]	[0.36665,1.63335]	[0.36541,1.63457]	[0.36494,1.63506]

with boundary conditions

$$u = 0 \quad \text{on} \quad x = 0 \text{ and } x = 1,$$
$$\frac{\partial u}{\partial n} = 0 \quad \text{on} \quad y = 0 \text{ and } y = 1.$$

For this example, we have two essential boundaries, i.e., $x = 0$ and $x = 1$. Other two boundaries, i.e., $y = 0$ and $y = 1$, are natural boundaries. To discritize the problem domain $\Omega = [0,1] \times [0,1]$, we take four regular nodes distribution, i.e., 21×21, 41×41, 61×61, 81×81. For each node distribution, 10×10, 20×20, 30×30 and 40×40 background cells are used to obtain integrals, respectively. A typical 21×21 nodes distribution and the corresponding 10×10 background cells are plotted in Fig. 4.1. The orders of saddle point matrices $\mathcal{A}$ are listed in Table 4.1.

In Table 4.2, we list the upper bound of the convergent interval and the optimal parameters for the RHSS iteration method, and the bound of eigenvalues of the RHSS preconditioned matrix $\mathcal{P}_{RHSS}^{-1}\mathcal{A}$ with optimal α_{opt}. From Table 4.2, we can see that all eigenvalues of the preconditioned matrix $\mathcal{P}_{RHSS}^{-1}\mathcal{A}$ are located in $(0,2)$ if optimal parameters were used. In Table 4.3, we list the numbers of iteration steps and the computing CPU times (in seconds) with respect to RHSS iteration method with optimal parameter (Algorithm 3.2), GMRES (no preconditioning, denote by $\mathcal{I}$), HSS preconditioned GMRES (with $\alpha = 1, 0.1, 0.01$) and RHSS preconditioned GMRES (with $\alpha = 1, \alpha_{opt}$). From Table 4.3, we can see that

- RHSS iteration converges very fast for solving saddle point problems (2.13) when optimal parameter is used. Without preconditioning, GMRES converges very slowly. If the HSS preconditioner or the RHSS preconditioner were used, the preconditioned GMRES method converges very fast.

- For the RHSS preconditioned GMRES, the numbers of iteration steps and the elapsed CPU times are almost the same when $\alpha = 1$ and $\alpha = \alpha_{opt}$ are used. But for the HSS preconditioned GMRES, the numbers of iteration steps and the elapsed CPU times vary greatly when different α are used.

- The optimal parameter α_{opt}, which minimizes the spectral radius of corresponding iteration matrix, is not the optimal parameter for the RHSS preconditioner. But this optimal parameter can get very good numerical results.

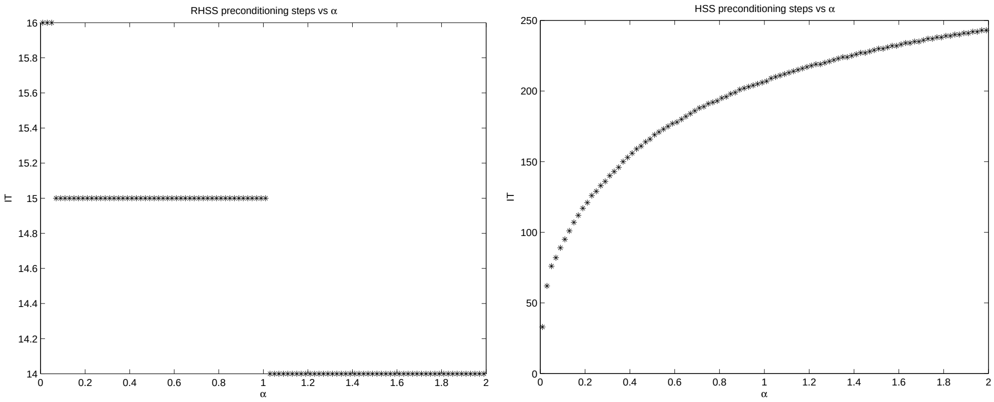

Fig. 4.2. Example 4.1: the effect of α: RHSS preconditioning steps vs α (left); HSS preconditioning steps vs α (right).

In order to study the effect of preconditioners with respect to parameters, Fig. 4.2 plots RHSS preconditioning steps and HSS preconditioning steps (varying α from 0.01 to 2) when 41×41 nodes are used. From Fig. 4.2, we can see that the parameter α does not influence the RHSS preconditioning steps greatly, while it influences the HSS preconditioning steps greatly. It seems that in the RHSS preconditioner $\alpha = 1$ may be a good choice. But for the HSS preconditioner, we should compute optimal parameters. If inappropriate parameter were used, the preconditioned GMRES may converge slowly. In general, finding optimal parameter α in the HSS preconditioner is a very difficult task.

Fig. 4.3 plots eigenvalue distribution of the saddle point matrix $\mathcal{A}$ (without preconditioning), the HSS preconditioned matrix $\mathcal{P}_{HSS}^{-1}\mathcal{A}$ (with $\alpha = 0.01$), the RHSS preconditioned matrix $\mathcal{P}_{RHSS}^{-1}\mathcal{A}$ (with $\alpha = 1$) and the RHSS preconditioned matrix $\mathcal{P}_{RHSS}^{-1}\mathcal{A}$ (with $\alpha = \alpha_{opt}$) when $n + m = 1763$. From Fig. 4.3, we can see that the eigenvalues of preconditioned matrices are more cluster than those of original matrix and good convergence may be obtained.

Table 4.3: Numerical results of GMRES and preconditioned GMRES for Example 4.1.

$Nodes$		21×21	41×41	61×61	81×81
RHSS iteration	IT	28	28	25	24
	CPU	0.204	0.332	0.9222	1.891
$\mathcal{I}$	IT	153	304	403	479
	CPU	0.496	1.456	6.465	12.770
$\mathcal{P}_{HSS}(\alpha{=}1)$	IT	106	207	271	316
	CPU	0.114	0.953	3.221	7.201
$\mathcal{P}_{HSS}(\alpha{=}0.1)$	IT	52	92	115	131
	CPU	0.064	0.313	0.968	2.057
$\mathcal{P}_{HSS}(\alpha{=}0.01)$	IT	26	33	44	56
	CPU	0.041	0.144	0.375	0.835
$\mathcal{P}_{RHSS}(\alpha_{opt})$	IT	11	14	14	14
	CPU	0.018	0.073	0.204	0.367
$\mathcal{P}_{RHSS}(\alpha{=}1)$	IT	11	15	15	16
	CPU	0.039	0.098	0.199	0.383

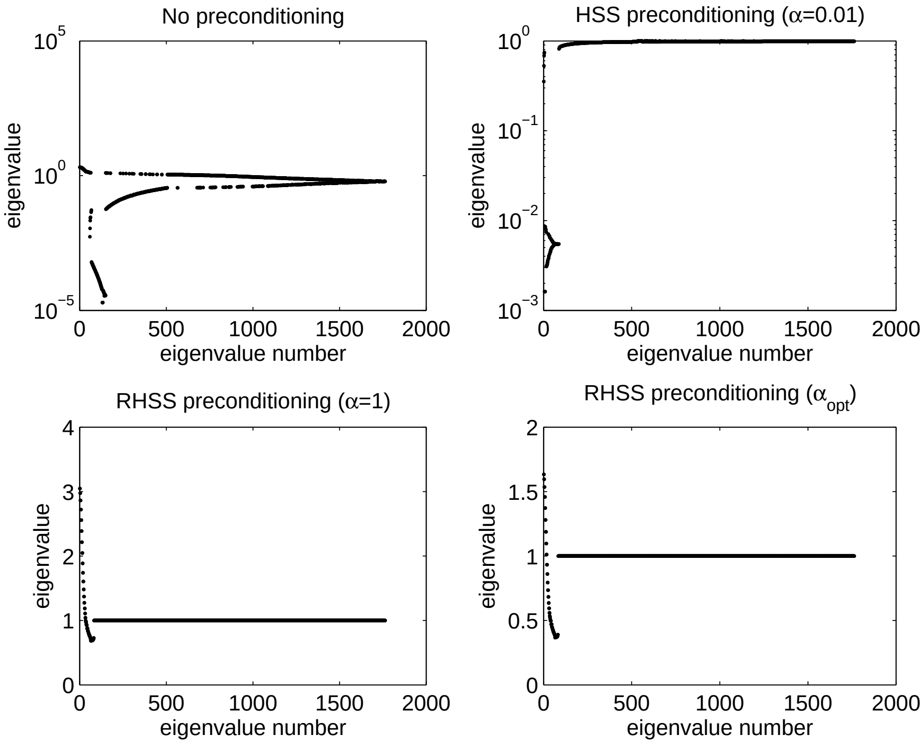

Fig. 4.3. Eigenvalue distribution for Example 4.1.

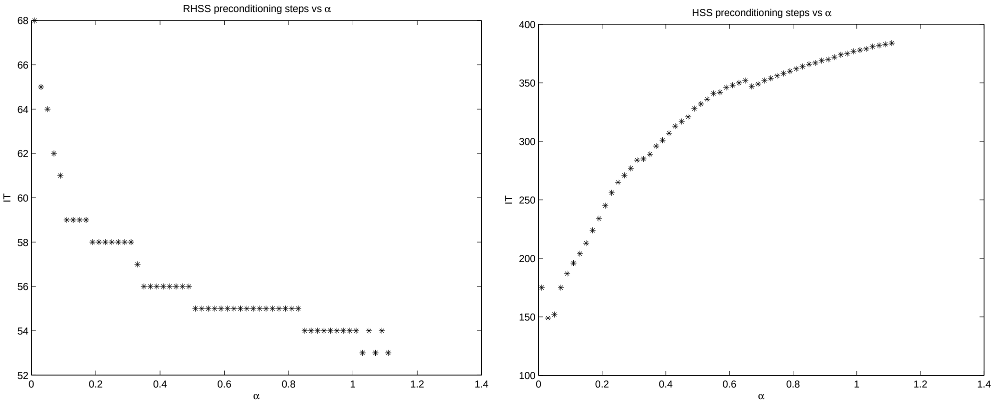

Fig. 4.4. Example 4.2: the effect of α: RHSS preconditioning steps vs α (left); HSS preconditioning steps vs α (right).

Example 4.2. The second numerical example is the Stokes problem: find $\mathbf{u}$ and p such that

$$\begin{cases} -\nu\Delta\mathbf{u} + \bigtriangledown p = \tilde{f}, & in\ \Omega, \\ \bigtriangledown \cdot \mathbf{u} = \tilde{g}, & in\ \Omega, \\ \mathbf{u} = 0, & on\ \partial\Omega, \\ \int_\Omega p(x)dx = 0. \end{cases} \tag{4.1}$$

where $\Omega = [0,1] \times [0,1] \subset \mathbb{R}^2, \partial\Omega$ is the boundary of Ω, ν stands for the viscosity scalar, Δ is the componentwise Laplace operator, $\mathbf{u} = (u^T, v^T)^T$ is a vector-valued function representing the velocity, and p is a scalar function representing the pressure.

For this example, we have four essential boundaries, i.e., $x = 0$, $x = 1$, $y = 0$, $y = 1$. There is no natural boundary. Nodes distribution and background cells are taken the same as Example 4.1. The viscosity scalar ν is taken as 1. The pressure p is taken at the center of each background cell. The orders of saddle point matrices $\mathcal{A}$ are listed in Table 4.4.

Table 4.4: Values of m, n and the order of $\mathcal{A}$ for Example 4.2.

Nodes	21×21	41×41	61×61	81×81
n	882	3362	7442	13122
m	184	564	1144	1924
$n+m$	1066	3926	8586	15046

Table 4.5: Parameters for Example 4.2.

Nodes	21×21	41×41	61×61	81×81
$2/\mu_1$	0.22975	0.06153	0.02776	0.01571
α_{opt}	0.22828	0.06142	0.02773	0.01569
$[\alpha\mu_m,\ \alpha\mu_1]$	[0.01277,1.98723]	[0.00344,1.99656]	[0.00155,1.99845]	[0.00088,1.99912]

Table 4.6: Numerical results of GMRES and preconditioned GMRES for Example 4.2.

Nodes		21×21	41×41	61×61	81×81
$\mathcal{I}$	*IT*	469	835	985	–
	CPU	2.685	27.029	66.226	–
$\mathcal{P}_{HSS}(\alpha{=}1)$	*IT*	377	644	743	955
	CPU	1.653	15.257	41.833	117.693
$\mathcal{P}_{HSS}(\alpha{=}0.1)$	*IT*	191	336	446	479
	CPU	0.502	4.877	17.495	36.358
$\mathcal{P}_{HSS}(\alpha{=}0.01)$	*IT*	175	243	252	302
	CPU	0.442	2.974	7.347	18.084
$\mathcal{P}_{RHSS}(\alpha_{opt})$	*IT*	53	70	85	98
	CPU	0.124	0.616	1.906	4.479
$\mathcal{P}_{RHSS}(\alpha{=}1)$	*IT*	54	68	82	92
	CPU	0.144	0.620	1.866	4.228

In Table 4.5, we list the optimal parameters used in the RHSS preconditioner and the corresponding positive interval for Example 4.2. Table 4.6 lists the numbers of iteration steps and the computing CPU times (in seconds) with respect to GMRES (no preconditioning), HSS preconditioned GMRES (with $\alpha = 1, 0.1, 0.01$) and RHSS preconditioned GMRES (with $\alpha = 1, \alpha_{opt}$). Fig. 4.4 plots RHSS preconditioning steps, and HSS preconditioning steps (varying α from 0.01 to 1.1) when $n{+}m = 1066$. Fig. 4.5 depicts eigenvalue distribution of the saddle point matrix $\mathcal{A}$ (without preconditioning), the HSS preconditioned matrix $\mathcal{P}_{HSS}^{-1}\mathcal{A}$ (with $\alpha = 0.01$), the RHSS preconditioned matrix $\mathcal{P}_{RHSS}^{-1}\mathcal{A}$ (with $\alpha = 1$) and the RHSS preconditioned matrix $\mathcal{P}_{RHSS}^{-1}\mathcal{A}$ (with $\alpha = \alpha_{opt}$) when $n + m = 1066$.

From these tables and figures, we can also see that using preconditioners can accelerate the convergence greatly, especially when suitable parameters are used. Moreover, the parameter α in the RHSS preconditioner is less sensitive than the parameter α in the HSS preconditioner. For the preconditioner $\mathcal{P}_{RHSS}$, $\alpha = 1$ is also a practical choice.

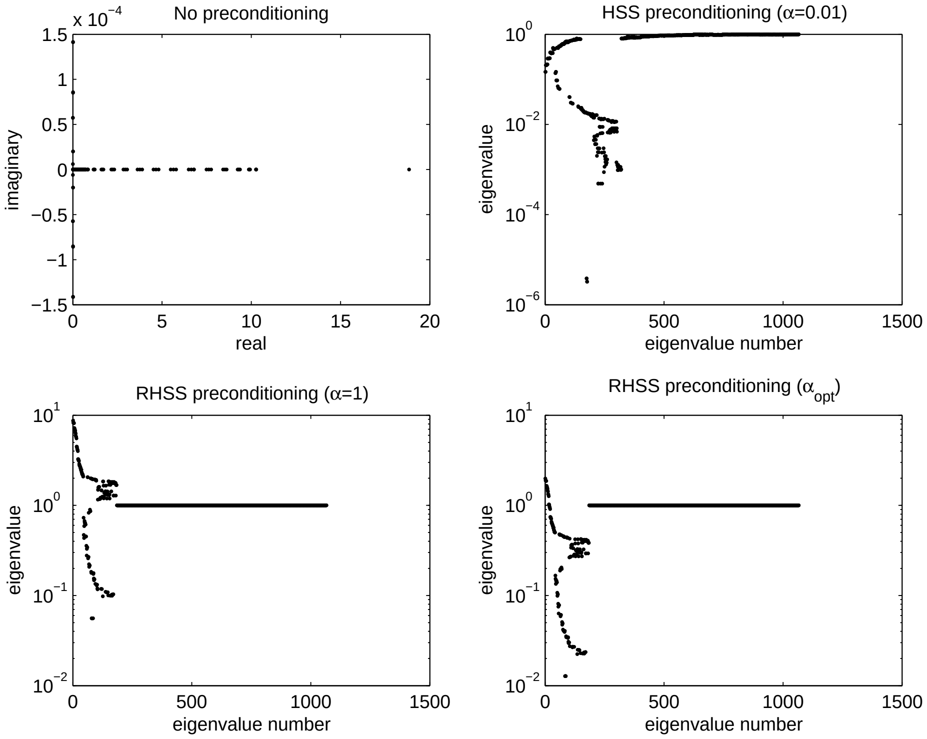

Fig. 4.5. Eigenvalue distribution for Example 4.2.

5. Conclusion

In this paper, we studied a relaxed Hermitian and skew-Hermitian splitting (RHSS) preconditioner for saddle point problems from the element-free Galerkin (EFG) discretization method, which is one of the most widely used meshfree methods. The RHSS preconditioner is constructed much closer to the coefficient matrix than the well-known HSS preconditioner. Although the new preconditioner no longer relates to an alternating direction method, it can be induced by a matrix splitting iterative method, which is called the RHSS iteration method. Convergence of the RHSS iteration method is analyzed and an optimal parameter is studied. Although the RHSS iteration method is very simple and very easy to implement, the convergence is typically too slow even with optimal parameter if bad conditions happen. For this reason, the Krylov subspace method (like GMRES) is used to accelerate the convergence of the RHSS iteration. Properties of the preconditioned matrix are studied. The eigenvalues of the preconditioned matrix are 1 with multiplicity at least n, and the remaining eigenvalues are real and located in a positive interval. Most important, the parameter has no effect on the ratio of the interval. Thus we suggest the parameter to be 1 in implementing the RHSS preconditioner. Besides, theoretical analysis shows that the degree of the minimal polynomial of the preconditioned matrix, which is equal to the dimension of the Krylov subspace, is at most $m+1$. This indicates that the RHSS preconditioner is very efficient for saddle point problems from the EFG discretization method. Numerical experiments arising from two model problems are presented to confirm the analytical results and to show effectiveness of the parameter $\alpha = 1$ used in the RHSS preconditioner.

Acknowledgments. The authors express their thanks to the referees for the comments and constructive suggestions, which were valuable in improving the quality of the manuscript. This work is supported by the National Natural Science Foundation of China(11172192) and the National Natural Science Pre-Research Foundation of Soochow University (SDY2011B01).

References

[1] Z.-Z. Bai, Optimal parameters in the HSS-like methods for saddle point problems, *Numer. Linear Algebra Appl.*, **16** (2009), 447-479.

[2] Z.-Z. Bai and G.H. Golub, Accelerated Hermitian and skew-Hermitian splitting iteration methods for saddle-point problems, *IMA J. Numer. Anal.*, **27** (2007), 1-23.

[3] Z.-Z. Bai, G.H. Golub, C.-K. Li, Convergence properties of preconditioned Hermitian and skew-Hermitian splitting methods for non-Hermitian positive semidefinite matrices, *Math. Comput.*, **76** (2007), 287-298.

[4] Z.-Z. Bai, G.H. Golub and 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.

[5] Z.-Z. Bai, G.H. Golub and M.K. Ng, On successive-overrelaxation acceleration of the Hermitian and skew-Hermitian splitting iterations, *Numer. Linear Algebra Appl.*, **14** (2007), 319-335.

[6] Z.-Z. Bai, G.H. Golub and J.-Y. Pan, Preconditioned Hermitian and skew-Hermitian splitting methods for non-Hermitian positive semidefinite linear systems, *Numer. Math.*, **98** (2004), 1-32.

[7] Z.-Z. Bai and M.K. Ng, On inexact preconditioners for nonsymmetric matrices, *SIAM J. Sci. Comput.*, **26** (2005), 1710-1724.

[8] Z.-Z. Bai, M.K. Ng and Z.-Q. Wang, Constraint preconditioners for symmetric indefinite matrices, *SIAM J. Matrix Anal. Appl.*, **31** (2009), 410-433.

[9] Z.-Z. Bai, B.N. Parlett and Z.-Q. Wang, On generalized successive overrelaxation methods for augmented linear systems, *Numer. Math.*, **102** (2005) 1-38.

[10] Z.-Z. Bai and Z.-Q. Wang, On parameterized inexact Uzawa methods for generalized saddle point problems, *Linear Algebra Appl.*, **428** (2008), 2900-2932.

[11] T. Belytschko, Y.Y. Lu and L. Gu, Element-free Galerkin methods, *Int. J. Numer. Meth. Engrg.*, **37** (1994), 229-256.

[12] M. Benzi, M.J. Gander and G.H. Golub, Optimization of the Hermitian and skew-Hermitian splitting iteration for saddle-point problems, *BIT*, **43** (2003), 881-900.

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

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

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

[16] M. Benzi, M.K. Ng, Q. Niu and Z. Wang, A relaxed dimensional fractorization preconditioner for the incompressible Navier-Stokes equations, *J. Comput. Phys.*, **230** (2011), 6185-6202.

[17] S.L. Borne, S. Oliveira and F. Yang, H-matrix preconditioners for symmetric saddle-point systems from meshfree discretization, *Numer. Linear Algebra Appl.*, **15** (2008), 911-924.

[18] Y. Cao, M.-Q. Jiang and L.-Q. Yao, New choices of preconditioning matrices for generalized inexact parameterized iterative methods, *J. Comput. Appl. Math.*, **235** (2010), 263-269.

[19] Y. Cao, M.-Q. Jiang and Y.-L. Zheng, A splitting preconditioner for saddle point problems, *Numer. Linear Algebra Appl.*, **18** (2011), 875-895.

[20] Y. Cao, M.-Q. Jiang and Y.-L. Zheng, A note on the positive stable block triangular preconditioner for generalized saddle point problems, *Appl. Math. Comput.*, **218** (2012) 11075-11082.

[21] Y. Cao, W.-W. Tan and M.-Q. Jiang, A generalization of the positive-definite and skew-Hermitian splitting iteration, *Numer. Algebra Control Optimization*, **2** (2012), 811-821.

[22] Y. Cao, L.-Q. Yao and Y. Yin, New treatment of essential boundary conditions in EFG method by coupling with RPIM, *Acta Mech. Solida Sinica*, To appear.

[23] M.-Q. Jiang and Y. Cao, On local Hermitian and skew-Hermitian splitting iteration methods for generalized saddle point problems, *J. Comput. Appl. Math.*, **231** (2009), 973-982.

[24] M.-Q. Jiang, Y. Cao and L.-Q. Yao, On parameterized block triangular preconditioners for generalized saddle point problems, *Appl. Math. Comput.*, **216** (2010), 1777-1789.

[25] C. Keller, N.I.M. Gould and A.J. Wathen, Constraint preconditioning for indefinite linear systems, *SIAM J. Matrix Anal. Appl.*, **21** (2000), 1300-1317.

[26] K.H. Leem, S. Oliveira and D.E. Stewart, Algebraic multigrid (AMG) for saddle point systems from meshfree discretizations, *Numer. Linear Algebra Appl.*, **11** (2004) 293-308.

[27] G.-R. Liu and Y.-T. Gu, An introduction to meshfree methods and their programming, Netherland: Springer; 2005.

[28] F.Z. LouaÏ, N. NaÏt-SaÏd and S. Drid, Implementation of an efficient element-free Galerkin method for electromagnetic computation, *Eng. Anal. Boundary Elem.*, **31** (2007), 191-199.

[29] Y. Saad, Iterative Methods for Sparse Linear Systems(2nd edn), SIAM: Philadelphia, 2003.

[30] V. Simoncini and M. Benzi, Spectral properties of the Hermitian and skew-Hermitian splitting preconditioner for saddle point problems, *SIAM J. Numer. Anal.*, **27** (2007), 1-23.

[31] G. Ventura, An augmented Lagrangian approach to essential boundary conditions in meshless methods, *Int. J. Numer. Meth. Engng.*, **53** (2002), 825-842.

[32] H. Zheng and J.-L. Li, A practical solution for KKT systems, *Numer. Algor.*, **46** (2007) 105-119.

[33] T. Zhu and S.N. Atluri, A modified collocation method and a penalty formulation for enforcing the essential boundary conditions in the element free Galerkin method, *Comput. Mech.*, **21** (1998), 211-222.