Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2015, Article ID 548609, 20 pages
http://dx.doi.org/10.1155/2015/548609

Research Article

Preconditioning Complex Symmetric Linear Systems

Enrico Bertolazzi and Marco Frego

Department of Industrial Engineering, University of Trento, Via Sommarive 9, Povo, 38123 Trento, Italy

Correspondence should be addressed to Enrico Bertolazzi; enrico.bertolazzi@unitn.it

Received 11 July 2014; Revised 4 November 2014; Accepted 27 November 2014

Academic Editor: Luca Bergamaschi

A new preconditioner for symmetric complex linear systems based on Hermitian and skew-Hermitian splitting (HSS) for complex symmetric linear systems is herein presented. It applies to conjugate orthogonal conjugate gradient (COCG) or conjugate orthogonal conjugate residual (COCR) iterative solvers and does not require any estimation of the spectrum of the coefficient matrix. An upper bound of the condition number of the preconditioned linear system is provided. To reduce the computational cost the preconditioner is approximated with an inexact variant based on incomplete Cholesky decomposition or on orthogonal polynomials. Numerical results show that the present preconditioner and its inexact variant are efficient and robust solvers for this class of linear systems. A stability analysis of the inexact polynomial version completes the description of the preconditioner.

1. Introduction

Focus of this paper is the solution of the complex linear system given by $\mathbf{A}\mathbf{x} = \mathbf{b}$, where the symmetric complex matrix $\mathbf{A}$ has the property that can be written as $\mathbf{A} = \mathbf{B} + \imath\mathbf{C}$ with $\mathbf{B}$, $\mathbf{C}$ two real symmetric positive semidefinite matrices (semi-SPD) and $\mathbf{B} + \mathbf{C}$ a symmetric positive definite (SPD) matrix. This kind of linear system arises, for example, in the discretization of problems in computational electrodynamics [1] or time-dependent Schrödinger equations, or in conductivity problems [2, 3].

If $\mathbf{A}$ is Hermitian, a straightforward extension of the conjugate gradients (CG) algorithm can be used [4]. Unfortunately, the CG method cannot be directly employed when $\mathbf{A}$ is only complex symmetric; thus, some specialized iterative methods must be adopted. An effective one is the HSS with its variants (MHSS), which need, at each iteration, the solution of two real linear systems. Other standard procedures to solve this problem are given by numerical iterative methods based on Krylov spaces and designed for complex symmetric linear systems: COCG [1], COCR [5], CSYM [6], and CMRH [7]. Some iterative methods for non-SPD linear systems like BiCGSTAB [8], BiCGSTAB(ℓ) [9, 10], GMRES [11], and QMR [12] can be adapted for complex symmetric matrices [4, 13, 14].

In this work a preconditioned version of COCG and COCR is proposed: the aim is the solution of complex linear systems, where the preconditioner is a single preconditioned MHSS iteration (PMHSS). The PMHSS step may be too costly or impracticable, thus cheaper approximations must be used, in this work some approximations are considered, and a polynomial preconditioner as a possible approximation of the PMHSS step is presented.

Methods based on Hermitian and skew-Hermitian splitting (HSS) [15–18] can be used as standalone solvers or combined (as preconditioner) together with CG like algorithms. The speed of convergence of CG like iterative schemes depends on the condition number of the matrix $\mathbf{A}$; thus, preconditioning is a standard way to improve convergence [19, 20]. Incomplete LU is a standard and accepted way to precondition linear systems. Despite its popularity, incomplete LU is potentially unstable, is difficult to parallelize, and lacks algorithmic scalability. Nevertheless, when incomplete LU is feasible and the preconditioned linear system is well conditioned, the resulting algorithm is generally the best performing.

In this work we focus on large problems, where incomplete LU preconditioning is too costly or not feasible. In this case, iterative methods like SSOR are used as preconditioners, but a better performance is obtained using

HSS iterative methods, which allow reducing the condition number effectively. However, HSS iterative methods need the solution of two SPD real systems at each step. Standard preconditioned CG methods can be used at each iteration [18] which can again be preconditioned with an incomplete Cholesky factorization, although for very large problems, the incomplete Cholesky factorization may be not convenient or not feasible. As an alternative, in the present paper a polynomial preconditioner is proposed, which allows solving the linear system for large matrices. Polynomial preconditioners do not have a good reputation as preconditioners [19, 20] and research on this subject was dropped in the late 80s. In fact, polynomial preconditioners based on Chebyshev polynomials need accurate estimates of minimum and maximum eigenvalues, while least squares polynomials were not computed using stable recurrences, limiting the degree of available stable polynomials [21–25]. However, in the last years, polynomial preconditioner received attention again after the works of Lu et al. [26]. Notwithstanding anything contained above, here we propose as a preconditioner the use of a polynomial approximation of a modified HSS step. A specialization for Chebyshev and Jacobi orthogonal polynomials is discussed and the corresponding polynomial preconditioner is evaluated using a stable recurrence which permits using very high degree polynomials.

The paper has this structure. Section 1.1 describes the problem and gives a brief summary of existing methods for the resolution with the fundamental results and variants that lead to the present method; in particular, the HSS is considered. Section 2 shows how to use one step of the MHSS method as a preconditioner and gives a bound on the conditioning number of the MHSS iteration matrix. Section 3 explains the iterative solution method with the strategy to adopt when Cholesky factorization is possible or not. Section 4 presents a scale transformation of the system in order to move the eigenvalues to the range (0, 1]. Section 5 describes the polynomial preconditioner based first on least squares and then in terms of orthogonal polynomials and furnishes a stable recurrence for the computation of polynomial preconditioners for high degrees. The specialization for Chebyshev and Jacobi orthogonal polynomials is presented. Section 6 studies the numerical stability of this process and Section 7 shows some numerical results; Section 8 concludes the paper.

1.1. The MHSS Iterative Solver.

The complex $N \times N$ linear system $\mathbf{A}\mathbf{x} = \mathbf{b}$, where $\mathbf{A}$ is complex symmetric, is solved via an iterative method based on a splitting algorithm (HSS). The preconditioner requires to solve (each time it is applied) two real symmetric and positive definite (SPD) linear systems.

The HSS scheme can be summarized in this manner: rewrite the vector $\mathbf{x}$ of the unknowns as the sum of real and imaginary part, $\mathbf{x} = \mathbf{y} + \imath\mathbf{z}$, and accordingly, the right hand side $\mathbf{b} = \mathbf{c} + \imath\mathbf{d}$, then the system $\mathbf{A}\mathbf{x} = \mathbf{b}$ is rewritten as

$$(\mathbf{B} + \imath\mathbf{C})(\mathbf{y} + \imath\mathbf{z}) = \mathbf{c} + \imath\mathbf{d}, \tag{1}$$

so that the two steps of the modified HSS method proposed in [15] result in

$$\begin{aligned}(\mathbf{V} + \mathbf{B})\mathbf{x}^{(k+1/2)} &= (\mathbf{V} - \imath\mathbf{C})\mathbf{x}^{(k)} + \mathbf{b},\\ (\mathbf{W} + \mathbf{C})\mathbf{x}^{(k+1)} &= (\mathbf{W} + \imath\mathbf{B})\mathbf{x}^{(k+1/2)} - \imath\mathbf{b},\end{aligned} \tag{2}$$

for suitable matrices $\mathbf{V}$ and $\mathbf{W}$. The previous procedure can be rewritten as a single step of a splitting based scheme $\mathbf{P}\mathbf{x}^{k+1} = \mathbf{Q}\mathbf{x}^{k} + \mathbf{b}$ by posing

$$\begin{aligned}\mathbf{P} &= (\mathbf{V} + \mathbf{B})[\mathbf{W} - \imath\mathbf{V}]^{-1}(\mathbf{W} + \mathbf{C}),\\ \mathbf{Q} &= (\mathbf{V} + \mathbf{B})[\mathbf{W} - \imath\mathbf{V}]^{-1}(\mathbf{W} + \imath\mathbf{B})(\mathbf{V} + \mathbf{B})^{-1}(\mathbf{V} - \imath\mathbf{C}).\end{aligned} \tag{3}$$

This iterative method converges if the iteration matrix $\mathbf{P}^{-1}\mathbf{Q}$, for example,

$$\mathbf{P}^{-1}\mathbf{Q} = (\mathbf{W} + \mathbf{C})^{-1}(\mathbf{W} + \imath\mathbf{B})(\mathbf{V} + \mathbf{B})^{-1}(\mathbf{V} - \imath\mathbf{C}), \tag{4}$$

has spectral radius strictly less than one. It is well known that the choice $\mathbf{V} = \mathbf{W} = \alpha\mathbf{I}$, for a given positive constant α, yields the standard HSS method [15–17] for which an estimate of the spectral radius is given by

$$\varrho\left(\mathbf{P}^{-1}\mathbf{Q}\right) \le \max_{j=1,2,\ldots,N}\left\{\frac{\sqrt{\alpha^2 + \lambda_j(\mathbf{B})^2}}{\left(\alpha + \lambda_j(\mathbf{B})\right)}\right\}, \tag{5}$$

where $\lambda_j(\mathbf{B})$ are the eigenvalues of the SPD matrix $\mathbf{B}$. The optimal value for α can also be computed [15–17] and is

$$\begin{aligned}\alpha_{\text{opt}} &= \arg\min_{\alpha}\max_{j=1,2,\ldots,n}\left\{\frac{\sqrt{\alpha^2 + \lambda_j(\mathbf{B})^2}}{\left(\alpha + \lambda_j(\mathbf{B})\right)}\right\}\\ &= \sqrt{\lambda_{\min}(\mathbf{B})\lambda_{\max}(\mathbf{B})}.\end{aligned} \tag{6}$$

From the previous formulas, it is clear that those computations rely on the knowledge (or estimate) of the minimum and maximum eigenvalue of the matrix $\mathbf{B}$, which is, in general, a hard problem.

Another possible choice for $\mathbf{V}$ and $\mathbf{W}$ is $\mathbf{V} = \alpha\mathbf{B}$ and $\mathbf{W} = \beta\mathbf{B}$ which yields, when $\alpha = \beta$, a variant of the MHSS method by [15–17]. In the next lemma, an upper bound of the spectral radius of the iteration matrix is given.

Lemma 1. *Let* $\mathbf{V} = \alpha\mathbf{B}$ *and* $\mathbf{W} = \beta\mathbf{B}$ *in (4) with* $\mathbf{B}$ *a SPD matrix and* $\mathbf{C}$ *a semi-SPD matrix,* $\alpha, \beta > 0$*; then the spectral radius of* $\mathbf{P}^{-1}\mathbf{Q}$ *satisfies the upper bound*

$$\varrho\left(\mathbf{P}^{-1}\mathbf{Q}\right) \le U(\alpha,\beta), \qquad U(\alpha,\beta) = \frac{\sqrt{1+\beta^2}}{1+\alpha}\max\left\{1, \frac{\alpha}{\beta}\right\} \tag{7}$$

and the minimum value of the upper bound $U(\alpha,\beta)$ *is attained when* $\alpha = \beta = 1$ *where* $U(1,1) = \sqrt{2}/2 \approx 0.707$.

Proof. Substituting $\mathbf{V} = \alpha\mathbf{B}$ and $\mathbf{W} = \beta\mathbf{B}$ in (4) yields

$$\begin{aligned}\mathbf{P}^{-1}\mathbf{Q} &= (\beta\mathbf{B} + \mathbf{C})^{-1}(\beta\mathbf{B} + \imath\mathbf{B})(\alpha\mathbf{B} + \mathbf{B})^{-1}(\alpha\mathbf{B} - \imath\mathbf{C}) \\ &= \frac{\beta + \imath}{1 + \alpha}(\beta\mathbf{B} + \mathbf{C})^{-1}(\alpha\mathbf{B} - \imath\mathbf{C}).\end{aligned} \tag{8}$$

If λ is an eigenvalue of matrix $\mathbf{P}^{-1}\mathbf{Q}$, it satisfies

$$\begin{gathered}0 = \det\left(\mathbf{P}^{-1}\mathbf{Q} - \lambda\mathbf{I}\right) \\ \Downarrow \\ 0 = \det\left(\frac{\beta + \imath}{1 + \alpha}(\alpha\mathbf{B} - \imath\mathbf{C}) - \lambda(\beta\mathbf{B} + \mathbf{C})\right) \\ \Downarrow \\ 0 = \det\left(\left(\alpha\frac{\beta + \imath}{1 + \alpha} - \lambda\beta\right)\mathbf{B} - \left(\imath\frac{\beta + \imath}{1 + \alpha} + \lambda\right)\mathbf{C}\right) \\ \Downarrow \\ 0 = \det\left(\frac{\alpha(\imath - \lambda\beta) + \beta(\alpha - \lambda)}{\lambda(\alpha + 1) - 1 + \beta\imath}\mathbf{B} - \mathbf{C}\right).\end{gathered} \tag{9}$$

Thus, μ defined as

$$\mu = \frac{\alpha(\imath - \lambda\beta) + \beta(\alpha - \lambda)}{\lambda(\alpha + 1) - 1 + \beta\imath}, \tag{10}$$

is a generalized eigenvalue; that is, it satisfies $\det(\mu\mathbf{B} - \mathbf{C}) = 0$ and it is well known that μ must be nonnegative. Computing λ from (10), the function $\lambda(\mu)$ is found to be

$$\lambda(\mu) := \frac{(\alpha + \imath\mu)(\beta + \imath)}{(1 + \alpha)(\beta + \mu)}, \tag{11}$$

which allows evaluating an upper bound $U(\alpha, \beta)$ of the spectral radius of $\mathbf{P}^{-1}\mathbf{Q}$:

$$\varrho\left(\mathbf{P}^{-1}\mathbf{Q}\right) \le \sup_{\mu\ge 0}|\lambda(\mu)| = \sup_{\mu\ge 0}\frac{\sqrt{\alpha^2 + \mu^2}\sqrt{1 + \beta^2}}{(1 + \alpha)(\beta + \mu)} \le U(\alpha, \beta). \tag{12}$$

There are optimal values of α and β that minimize $U(\alpha, \beta)$ for $\alpha \ge 0$ and $\beta \ge 0$. To find them, set $\alpha = \ell\sin\theta$ and $\beta = \ell\cos\theta$ with $\theta \in [0, \pi/2]$; then

$$U(\alpha, \beta) = \frac{\sqrt{1 + \ell^2(\cos\theta)^2}}{1 + \ell\sin\theta}\max\{1, \tan\theta\}. \tag{13}$$

If θ is fixed, the minimum of this last expression is for $\ell = \sin\theta/(\cos\theta)^2$ corresponding to

$$U(\alpha, \beta) = \cos\theta\max\{1, \tan\theta\} = \max\{\cos\theta, \sin\theta\} \quad \text{for } \theta \in \left[0, \frac{\pi}{2}\right]. \tag{14}$$

The minimum of $U(\alpha, \beta)$ is attained for $\theta = \pi/4$, which corresponds to $\alpha = \beta = 1$. The computation of $U(1, 1)$ is then straightforward. □

```
r ← b;    x ← 0;
while $\|\mathbf{r}\| > \varepsilon\|\mathbf{b}\|$ do
    Solve $(\mathbf{B} + \mathbf{C})\mathbf{h} = ((1 - \imath)/2)\mathbf{r}$;
    x ← x + h;
    r ← b − Ax;
end while
```

Algorithm 1: MHSS iterative solver for $\mathbf{A}\mathbf{x} = (\mathbf{B} + \imath\mathbf{C})\mathbf{x} = \mathbf{b}$.

In this setting, since $\alpha = \beta = 1$ are optimal, from now on it is assumed $\alpha = \beta = 1$ and therefore $\mathbf{V} = \mathbf{W} = \mathbf{B}$. Using these values, the two-step method (2) is recast as the one step method:

$$\mathbf{x}^{(k+1)} = \mathbf{P}^{-1}\left(\mathbf{Q}\mathbf{x}^{(k)} + \mathbf{b}\right) = \mathbf{x}^{(k)} + \mathbf{P}^{-1}\left(\mathbf{b} - \mathbf{A}\mathbf{x}^{(k)}\right), \tag{15}$$

where the simplified expressions for $\mathbf{P}$ and $\mathbf{Q}$ are

$$\mathbf{P} = (1 + \imath)(\mathbf{B} + \mathbf{C}), \qquad \mathbf{Q} = \mathbf{C} + \imath\mathbf{B}. \tag{16}$$

Notice that $\mathbf{P}$ is well defined and nonsingular provided that $\mathbf{B} + \mathbf{C}$ is not singular. Thus the assumptions of Lemma 1 are weakened when $\alpha = \beta = 1$, resulting in the next corollary.

Corollary 2. *Let $\mathbf{B}$ and $\mathbf{C}$ be semi-SPD with $\mathbf{B}+\mathbf{C}$ not singular and $\mathbf{P}$ and $\mathbf{Q}$ as defined in (16); then the spectral radius of $\mathbf{P}^{-1}\mathbf{Q}$ satisfies the upper bound $\varrho(\mathbf{P}^{-1}\mathbf{Q}) \le \sqrt{2}/2$.*

Remark 3. The spectral radius of the iteration matrix is bounded independently of its size; thus, once the tolerance is fixed, the maximum number of iterations is independent of the size of the problem.

The iterative method (15) can be reorganized in Algorithm 1.

Remark 4. The MHSS iterative solver of Algorithm 1 needs at each iteration the resolution of two real linear systems, respectively, for the real and imaginary part, whose coefficient matrices are SPD, namely, $\mathbf{B} + \mathbf{C}$. For small matrices this can be efficiently done by using Cholesky decomposition. For large and sparse matrices a preconditioned conjugate gradient method is mandatory.

Although Algorithm 1 can be used to solve the linear system (1), better performance is obtained using one or more steps of Algorithm 1 not for solving the linear system (1) but as a preconditioner for a faster conjugate gradient like iterative solver such as COCG or COCR. The convergence rate estimation for these iterative schemes for a linear system $\mathbf{M}\mathbf{x} = (\mathbf{B} + \mathbf{C})\mathbf{x} = \mathbf{b}$ depends on the condition number $\kappa_2 = \|\mathbf{M}\|_2\|\mathbf{M}^{-1}\|_2$, where $\|\mathbf{M}\|_2 = \sqrt{\varrho(\mathbf{M}^T\mathbf{M})}$ is the classic spectral norm of a matrix. The energy norm $\|\cdot\|_{\mathbf{M}}$ induced by the (real) SPD matrix $\mathbf{M}$ is used to obtain the well known estimate

$$\frac{\left\|\mathbf{x}^{(k)} - \mathbf{x}^{\star}\right\|_{\mathbf{M}}}{\left\|\mathbf{x}^{(0)} - \mathbf{x}^{\star}\right\|_{\mathbf{M}}} \le 2\left(\frac{\sqrt{\kappa_2} - 1}{\sqrt{\kappa_2} + 1}\right)^k, \qquad \|\mathbf{v}\|_{\mathbf{M}} = \sqrt{\mathbf{v}^T\mathbf{M}\mathbf{v}}, \tag{17}$$

where $\mathbf{x}^\star$ is the solution of the linear system. In general, conjugate gradient-like iterative schemes perform efficiently if a good preconditioner makes the system well conditioned.

2. Use of MHSS as Preconditioner

In this section the effect of a fixed number of steps of Algorithm 1, used as a preconditioner for the linear system (1), is analyzed in terms of the reduction of the condition number. Performing n steps of Algorithm 1 with $\mathbf{x}^{(0)} = \mathbf{0}$ is equivalent to compute $\mathbf{x}^{(n)} = \mathscr{P}_n^{-1}\mathbf{b}$, where

$$\mathscr{P}_n^{-1} = \left(\mathbf{I} + \mathbf{P}^{-1}\mathbf{Q} + \left(\mathbf{P}^{-1}\mathbf{Q}\right)^2 + \cdots + \left(\mathbf{P}^{-1}\mathbf{Q}\right)^{n-1}\right)\mathbf{P}^{-1}. \quad (18)$$

Matrix $\mathscr{P}_n$ can be interpreted as an approximation of matrix $\mathbf{A} = \mathbf{B} + \imath\mathbf{C}$.

Thus, it is interesting to obtain an estimate of the condition number of the preconditioned matrix $\mathscr{P}_n^{-1}\mathbf{A}$ in order to check the effect of MHSS when used as preconditioner.

For the estimation, we need to recall some classical results about spectral radii and norms. For any matrix $\mathbf{M}$ and any matrix norm, Gelfand's Formula connects norm and spectral radius [27, 28]:

$$\varrho(\mathbf{M}) = \limsup_{k\to\infty} \left\|\mathbf{M}^k\right\|^{1/k}. \quad (19)$$

Notice that when $\varrho(\mathbf{M}) < 1$, for k large enough, $\|\mathbf{M}^k\| < 1$.

Lemma 5. *Let* $\mathbf{A} = \mathbf{P} - \mathbf{Q}$ *so that* $\mathbf{P}^{-1}\mathbf{Q}$ *is such that* $\varrho(\mathbf{P}^{-1}\mathbf{Q}) < 1$*; then for any* $\varepsilon > 0$ *satisfying* $\varrho(\mathbf{P}^{-1}\mathbf{Q}) + \varepsilon < 1$ *there is an integer* $n_\varepsilon > 0$ *such that*

$$\kappa\left(\mathscr{P}_n^{-1}\mathbf{A}\right) \le \frac{1 + \left(\varrho\left(\mathbf{P}^{-1}\mathbf{Q}\right) + \varepsilon\right)^n}{1 - \left(\varrho\left(\mathbf{P}^{-1}\mathbf{Q}\right) + \varepsilon\right)^n}, \quad n \ge n_\varepsilon, \quad (20)$$

where $\kappa(\mathbf{M}) = \|\mathbf{M}\|\|\mathbf{M}^{-1}\|$ *is the condition number with respect to the norm* $\|\cdot\|$ *and* $\mathscr{P}_n$ *is defined by (18).*

Proof. Observe that $\mathbf{P}^{-1}\mathbf{A} = \mathbf{P}^{-1}(\mathbf{P} - \mathbf{Q}) = \mathbf{I} - \mathbf{P}^{-1}\mathbf{Q}$; hence using (18) the preconditioned matrix becomes $\mathscr{P}_n^{-1}\mathbf{A} = \mathbf{I} - (\mathbf{P}^{-1}\mathbf{Q})^n$. From Gelfand's Formula (19) there exists n_ε such that

$$\left\|\left(\mathbf{P}^{-1}\mathbf{Q}\right)^n\right\| \le \left(\varrho\left(\mathbf{P}^{-1}\mathbf{Q}\right) + \varepsilon\right)^n < 1 \quad \forall n \ge n_\varepsilon \quad (21)$$

and from (21), by setting $\mathbf{M} = (\mathbf{P}^{-1}\mathbf{Q})^n$, the convergent series (see [29, 30]) gives a bound for the norm of the inverse

$$(\mathbf{I} - \mathbf{M})^{-1} = \sum_{k=0}^{\infty}\mathbf{M}^k, \qquad \left\|(\mathbf{I} - \mathbf{M})^{-1}\right\| \le \sum_{k=0}^{\infty}\|\mathbf{M}\|^k = \frac{1}{1 - \|\mathbf{M}\|},$$
$$\kappa\left(\mathscr{P}_n^{-1}\mathbf{A}\right) = \kappa(\mathbf{I} - \mathbf{M}) = \|\mathbf{I} - \mathbf{M}\|\left\|(\mathbf{I} - \mathbf{M})^{-1}\right\| \le \frac{1 + \|\mathbf{M}\|}{1 - \|\mathbf{M}\|}. \quad (22)$$

The thesis follows trivially from (21). □

Corollary 6. *Let* $\mathbf{A} = \mathbf{P} - \mathbf{Q}$ *so that* $\mathbf{P}^{-1}\mathbf{Q}$ *has the property that* $\varrho(\mathbf{P}^{-1}\mathbf{Q}) < 1$*; then there exists a matrix norm* $|||\cdot|||$ *such that the conditioning number of the matrix* $\mathscr{P}_n^{-1}\mathbf{A}$ *with respect to this norm satisfies*

$$\kappa\left(\mathscr{P}_n^{-1}\mathbf{A}\right) \le \frac{1 + 0.8^n}{1 - 0.8^n} \le 9, \quad (23)$$

where $\kappa(\mathbf{M}) = |||\mathbf{M}|||\,|||\mathbf{M}^{-1}|||$.

Proof. Recall that for any matrix $\mathbf{M}$ and $\epsilon > 0$ there exists a matrix norm $|||\cdot|||$ such that $|||\mathbf{M}||| \le \varrho(\mathbf{M}) + \epsilon$. This is a classical result of linear algebra; see, for example, Section 2.3 of [29] or Section 6.9 of [30]. Thus, given $\mathbf{P}^{-1}\mathbf{Q}$ and choosing $0 < \varepsilon \le 0.8 - \sqrt{2}/2$ there exists a matrix norm $|||\cdot|||$ such that $|||\mathbf{P}^{-1}\mathbf{Q}||| \le 0.8$. The proof follows from Lemma 5. □

From Corollary 2 using the Euclidean norm and choosing ε such that $\varrho(\mathbf{P}^{-1}\mathbf{Q}) + \varepsilon = \sqrt{2}/2 + \varepsilon = 0.8$ for $n \ge n_\varepsilon$, the condition number of the preconditioned matrix satisfies

$$\kappa_2\left(\mathscr{P}_n^{-1}\mathbf{A}\right) \le \frac{1 + 0.8^n}{1 - 0.8^n}, \qquad \kappa_2(\mathbf{M}) = \|\mathbf{M}\|_2\left\|\mathbf{M}^{-1}\right\|_2. \quad (24)$$

This estimate shows that using n steps of MHSS, with n large enough, the condition number of the preconditioned system can be bounded independently of the size of the linear system. In practice, when $n = 1$ the reduction of the condition number is enough; in fact, Corollary 6 shows that, using the appropriate norm, the condition number of the preconditioned linear system is less than 9, independently of its size.

Remark 7. From [31], when the condition number κ is large, the estimate of m conjugate gradient (CG) iterations satisfies $m \propto \sqrt{\kappa}$. The cost of computation is proportional to the number of iterations, whereas the cost of each iteration is proportional to $1 + Cn$, where C is the cost of an iteration of MHSS used as preconditioner, relative to the cost of a CG step. Thus, using κ_2 from (24), when n is large enough, a rough estimate of the computational cost is

$$\text{cost} \propto \sqrt{\kappa_2\left(\mathscr{P}_n^{-1}\mathbf{A}\right)}(1 + Cn) \propto \sqrt{\frac{1 + 0.8^n}{1 - 0.8^n}}(1 + Cn). \quad (25)$$

Fixing C, the cost (25), as a function of n alone, is convex and has a minimum for $n = n_{\min}(C)$ which satisfies

$$C = \frac{-a^n\ln(a)}{a^n\ln(a) - a^{2n} + 1}, \quad a = 0.8. \quad (26)$$

The inverse function $C(n)$ which satisfies $n_{\min}(C(n)) = n$ is the right hand side of (26); moreover, $C(n)$ is a monotone decreasing function so that also $n_{\min}(C)$ is monotone decreasing.

Table in equation (27) shows the constant C as a function of n giving the critical values of C such that n steps of MHSS are better than $n - 1$ steps,

$n_{\min}$	1	2	3	4	5	6	7
$C(n_{\min})$	0.98	0.47	0.29	0.2	0.14	0.1	0.07

(27)

```
$\mathbf{r} \leftarrow \mathbf{b} - \mathbf{A}\mathbf{x}$;
$\tilde{\mathbf{r}} \leftarrow \mathbf{P}^{-1}\mathbf{r}$;
$\mathbf{p} \leftarrow \tilde{\mathbf{r}}$;
$\rho \leftarrow [\tilde{\mathbf{r}}, \mathbf{r}]$;
while $\|\mathbf{r}\| > \varepsilon\|\mathbf{b}\|$ do
    $\mathbf{q} \leftarrow \mathbf{A}\mathbf{p}$;
    $\mu \leftarrow [\mathbf{q}, \mathbf{p}]$;
    $\alpha \leftarrow \rho/\mu$;
    $\mathbf{x} \leftarrow \mathbf{x} + \alpha\mathbf{p}$;
    $\mathbf{r} \leftarrow \mathbf{r} - \alpha\mathbf{q}$;
    $\tilde{\mathbf{r}} \leftarrow \mathbf{P}^{-1}\mathbf{r}$;
    $\beta \leftarrow \rho$;
    $\rho \leftarrow [\tilde{\mathbf{r}}, \mathbf{r}]$;
    $\beta \leftarrow \rho/\beta$;
    $\mathbf{p} \leftarrow \tilde{\mathbf{r}} + \beta\mathbf{p}$;
end while
```

Algorithm 2: COCG.

```
$\mathbf{r} \leftarrow \mathbf{b} - \mathbf{A}\mathbf{x}$;
$\tilde{\mathbf{r}} \leftarrow \mathbf{P}^{-1}\mathbf{r}$;
$\mathbf{p} \leftarrow \tilde{\mathbf{r}}$;
$\mathbf{q} \leftarrow \mathbf{A}\mathbf{p}$;
$\rho \leftarrow [\tilde{\mathbf{r}}, \mathbf{q}]$;
while $\|\mathbf{r}\| > \varepsilon\|\mathbf{b}\|$ do
    $\tilde{\mathbf{q}} \leftarrow \mathbf{P}^{-1}\mathbf{q}$;
    $\alpha \leftarrow \rho/[\tilde{\mathbf{q}}, \mathbf{q}]$;
    $\mathbf{x} \leftarrow \mathbf{x} + \alpha\mathbf{p}$;
    $\mathbf{r} \leftarrow \mathbf{r} - \alpha\mathbf{q}$;
    $\tilde{\mathbf{r}} \leftarrow \tilde{\mathbf{r}} - \alpha\tilde{\mathbf{q}}$;
    $\mathbf{t} \leftarrow \mathbf{A}\tilde{\mathbf{r}}$;
    $\beta \leftarrow \rho$;
    $\rho \leftarrow [\tilde{\mathbf{r}}, \mathbf{t}]$;
    $\beta \leftarrow \rho/\beta$;
    $\mathbf{p} \leftarrow \tilde{\mathbf{r}} + \beta\mathbf{p}$;
    $\mathbf{q} \leftarrow \mathbf{t} + \beta\mathbf{q}$
end while
```

Algorithm 3: COCR (as in [32]).

This means that it is convenient to use $n > 1$ steps in the preconditioner MHSS only if the cost of one step of MHSS is less than 0.47, that is, half of one step of the conjugate gradient method, a situation that never happens in practice.

According to Remark 7, only $\mathscr{P}_n$ with $n = 1$ is considered; that is, $\mathscr{P}_1 = \mathbf{P}$ as preconditioner for the linear system (1).

3. Iterative Solution of Complex Linear System

From the previous section, it is clear that the use of one iteration step of MHSS is a good choice that lowers the condition number of the original complex linear system (1). The resulting preconditioner matrix is $\mathbf{P} = (1 + \imath)(\mathbf{B} + \mathbf{C})$ as defined in (16). Preconditioner $\mathbf{P}$ will be used together with semi-iterative methods specialized in the solution of complex problems. Examples of those methods include COCG [4, 32] or COCR [5, 32–34]. They are briefly exposed in Algorithms 2 and 3.

There are also other methods available for performing this task, for example, CSYM [6] or QMR [35].

The application of the preconditioner $\mathbf{P}$ for those methods is equivalent to the solution of two real SPD systems depending on $\mathbf{B} + \mathbf{C}$, as in Remark 4. Of course, one can use a direct method [20, 36–38], or a conjugate gradient method [18] with incomplete Cholesky factorizations, or approximate inverse as a preconditioner [20, 39–43].

For very large linear systems $\mathbf{B} + \mathbf{C}$ may be expensive or impractical to be formed; moreover, well suited variants of the incomplete Cholesky factorization or of the sparse inverse must be considered [44–46]. In the next section, a polynomial preconditioner based on *orthogonal polynomials* is presented; it will allow solving very large complex linear systems with a simple algorithm that can be implemented with few lines of code.

The suggested strategy to get the solution of the complex linear system of the form $\mathbf{A}\mathbf{x} = (\mathbf{B} + \imath\mathbf{C})\mathbf{x} = \mathbf{b}$ is to use an iterative method like COCG (Algorithm 2) or COCR (Algorithm 3) preconditioned with $\mathbf{P} = (1 + \imath)(\mathbf{B} + \mathbf{C})$. The fully populated factorization of $\mathbf{P}^{-1}$ for large or huge linear systems is not practically computable so that some approximation must be used. The strategy can be split in two.

(1) Compute $(\mathbf{B} + \mathbf{C})\mathbf{z} = \mathbf{v}/(1 + \imath)$ by using iterative methods. This can be organized as the solution of two SPD real system, one for the real and one for the imaginary part. This systems can be efficiently solved by using preconditioned conjugate gradient.

(2) Approximate $(\mathbf{B} + \mathbf{C})^{-1} = \mathbf{Q} + \mathbf{E}$ with $\|\mathbf{E}\|$ "small" and use $(1 + \imath)^{-1}\mathbf{Q}^{-1}$ as *approximate* preconditioner.

Approximations of strategy 2 can be used as preconditioners for strategy 1. In strategy 2 the approximation must not be too far from the inverse of $(1 + \imath)(\mathbf{B} + \mathbf{C})$; otherwise the convergence will fail. In general for strategy 2 the choice of the preconditioner can be done as follows.

(i) If the complete Cholesky of $\mathbf{B} + \mathbf{C} = \mathbf{L}\mathbf{L}^T$ is available use $\mathbf{P} = (1 + \imath)\mathbf{L}\mathbf{L}^T$ as preconditioner.

(ii) If the incomplete Cholesky of $\mathbf{B} + \mathbf{C} = \tilde{\mathbf{L}}\tilde{\mathbf{L}}^T + \mathbf{E}$ is computationally not too expensive and $\|\mathbf{E}\|$ is "small" then use $\mathbf{P} = (1 + \imath)\tilde{\mathbf{L}}\tilde{\mathbf{L}}^T$ as preconditioner.

(iii) If $\|\mathbf{E}\|$ is "too large" or the solution of $(\mathbf{B}+\mathbf{C})\mathbf{z} = \mathbf{v}/(1+\imath)$ by means of PCG with $\mathbf{Q} = \tilde{\mathbf{L}}\tilde{\mathbf{L}}^T$ as preconditioner is too costly, try some alternative. In particular try:

  (a) Approximate $(\mathbf{B} + \mathbf{C})^{-1}$ using sparse inverse [39, 47–50] or algebraic multigrid [51–53] or any other good approximation.

  (b) Use the polynomial preconditioner proposed in Section 5.

Incomplete LU decomposition is easy to set up and exhibits good performances, while better performance may be obtained by using algebraic multigrid method (AMG); however, comparisons with AMG are difficult because of the large number of parameters to set up, like the depth and the shape of the W-cycle and the smoother (or relaxation method) and the coarse grid correction step.

If incomplete Cholesky fails or it gives poor preconditioning, we propose a simple but effective polynomial preconditioner for which we give details such as stability and convergence properties. Notice that the optimal preconditioner is problem dependent and a good preconditioner for a problem may be a bad preconditioner for another one. The polynomial preconditioner proposed in Section 5, without being optimal, is extremely simple to implement and easily parallelizable and ensures convergence also for very large linear systems. Finally, polynomial preconditioners have the following benefits with respect to incomplete Cholesky and algebraic multigrid.

(i) If matrix-vector multiplication $(\mathbf{B}+i\mathbf{C})\mathbf{z}$ can be computed but the matrices $\mathbf{B}$ and $\mathbf{C}$ cannot be explicitly formed, the polynomial preconditioner is computable while incomplete LU and algebraic multigrid cannot be built.

(ii) Polynomial preconditioners are easily parallelizable.

(iii) The polynomial preconditioner never breaks down as it can happen to incomplete Cholesky or algebraic multigrid.

(iv) The polynomial preconditioner can be extended also for problems with a singular matrix.

Strategy number 1, that is, computing $(\mathbf{B}+\mathbf{C})\mathbf{z} = \mathbf{v}/(1+i)$ by using iterative methods (e.g., PCG), is in general not competitive. In fact, to be competitive, the system should be solved with very few iterations, but this means that the preconditioner for $\mathbf{B}+\mathbf{C}$ is very good; therefore it can be used directly and successfully in strategy number 2.

4. Scaling the Complex Linear System

The polynomial preconditioner presented in the next section depends on the knowledge of an interval containing eigenvalues. Scaling is a cheap procedure to recast the problem into one with eigenvalues in the interval $(0,1]$. Using the diagonal matrix $\mathbf{S}$, the linear system (1) becomes

$$(\mathbf{SAS})\mathbf{w} = (\mathbf{SBS} + i\mathbf{SCS})\mathbf{w} = \mathbf{Sb}, \quad \text{with } \mathbf{x} = \mathbf{Sw}, \tag{28}$$

where $\mathbf{S}$ is a real diagonal matrix with positive entries on the diagonal. The scaled system inherits the properties of the original and still has the matrices $\mathbf{SBS}$ and $\mathbf{SCS}$ semi-SPD with $\mathbf{SBS}+\mathbf{SCS}$ SPD. The next lemma shows how to choose a good scaling factor $\mathbf{S}$ used forward.

Lemma 8. *Let* $\mathbf{M}$ *be a SPD matrix and* $\mathbf{S}$ *a diagonal matrix with* $S_{ii} = (\sum_{j=1}^{n} |M_{ij}|)^{-1/2}$*; then the scaled matrix* $\mathbf{SMS}$ *has the eigenvalues in the range* $(0,1]$.

Proof. Notice that $\mathbf{SMS}$ is symmetric and positively defined and is similar to $\mathbf{S}^2\mathbf{M}$. Moreover, the estimate $\lambda_{\max}(\mathbf{S}^2\mathbf{M}) \le \|\mathbf{S}^2\mathbf{M}\|_\infty = 1$ follows trivially. □

Assumption 9. From Lemma 8, the linear system (1) is scaled to satisfy the following:

(i) matrices $\mathbf{B}$ and $\mathbf{C}$ are semi-SPD;

(ii) matrix $\mathbf{B}+\mathbf{C}$ is SPD with eigenvalues in $(0,1]$.

5. Preconditioning with Polynomials

On the basis of the results of the previous sections with Assumption 9, the linear system to be preconditioned has the form $\mathbf{Ax} = \mathbf{b}$ with $\mathbf{A} = \mathbf{B} + i\mathbf{C}$ with $\mathbf{B}$ and $\mathbf{C}$ semi-SPD and $\mathbf{M} = \mathbf{B}+\mathbf{C}$ SPD with eigenvalues distributed in the interval $(0,1]$.

A good preconditioner for this linear system is one step of MHSS in Algorithm 1, which results in a multiplication by $\mathbf{P}^{-1}$ where $\mathbf{P} = (1+i)(\mathbf{B}+\mathbf{C}) = (1+i)\mathbf{M}$. Here the following polynomial approximation of $\mathbf{P}^{-1}$ is proposed:

$$\mathbf{P}^{-1} = \frac{1-i}{2}\mathbf{M}^{-1} \approx \frac{1-i}{2} s_m(\mathbf{M}). \tag{29}$$

The matrix polynomial $s_m(\mathbf{M})$ must be an approximation of the inverse of $\mathbf{M}$, that is, $s_m(\mathbf{M})\mathbf{M} \approx \mathbf{I}$, where $s_m(x)$ is a polynomial with degree m. A measure of the quality of the preconditioned matrix for a generic polynomial $s(x)$ is the distance from the identity matrix:

$$\mathcal{Q}_\sigma(s) = \|s(\mathbf{M})\mathbf{M} - \mathbf{I}\|_2 = \max_{\lambda\in\sigma(\mathbf{M})} |1 - \lambda s(\lambda)|, \tag{30}$$

where $\sigma(\mathbf{M}) = \{\lambda_1,\ldots,\lambda_n\}$ is the spectrum of $\mathbf{M}$. If, in particular, the preconditioned matrix $s_m(\mathbf{M})\mathbf{M}$ is the identity matrix then $\mathcal{Q}_\sigma(s_m) = 0$. Thus, the polynomial preconditioner s_m should concentrate the eigenvalues of $s_m(\mathbf{M})\mathbf{M}$ around 1 in order to be effective.

A preconditioner polynomial can be constructed by minimizing $\mathcal{Q}_\sigma(s)$ of (30) within the space Π_m of polynomials of degree at most m. This implies the knowledge of the spectrum of the matrix $\mathbf{M}$ which is in general not available making problem (30) unfeasible. The following approximation of quality measure (30) is feasible:

$$\mathcal{Q}_{[\epsilon,1]}(s) = \max_{\lambda\in[\epsilon,1]} |1 - \lambda s(\lambda)|, \qquad \sigma(\mathbf{M}) \subset [\epsilon, 1] \tag{31}$$

and it needs the knowledge of $[\epsilon,1]$ and an interval for $\epsilon > 0$, containing the spectrum of $\mathbf{M}$. The polynomial which minimizes $\mathcal{Q}_{[\epsilon,1]}(s)$ for $s \in \Pi_m$ is well known and is connected to an appropriately scaled and shifted Chebyshev polynomial. The construction of such solution is described in Section 5.1 and was previously considered by Ashby et al. [21, 22], Johnson et al. [23], Freund [54], Saad [24], and Axelsson [19]. The computation of $\mathcal{Q}_{[\epsilon,1]}(s)$ needs the estimation of a positive lower bound of the minimum eigenvalue of $\mathbf{M}$, which is, in general, expensive or infeasible. The estimate $\epsilon = 0$ cannot be used because $\mathcal{Q}_{[0,1]}(s) \ge 1$ for any polynomial s. A different way to choose ϵ is analysed later in this section. Saad observed that the use of Chebyshev polynomials with the conjugate gradient method, that is, the polynomial which minimizes the condition number of the preconditioned system, is in general far from being the best polynomial preconditioner, that is, the one that minimizes the CG iterations [24]. Practice shows that although nonoptimal, Chebyshev preconditioners perform well in many situations. The following integral average quality measure proposed in [22–24, 55] is a feasible alternative to (31):

$$\mathcal{Q}(s) = \int_0^1 |1-\lambda s(\lambda)|^2 \, d\lambda, \qquad \sigma(\mathbf{M}) \subset [0,1]. \tag{32}$$

The preconditioner polynomial s_m proposed here is the solution of minimization of quality measure (31) or (32):

$$s_m = \operatorname*{argmin}_{s\in\Pi_m} \mathcal{Q}_{[\epsilon,1]}(s), \quad \text{or} \quad s_m = \operatorname*{argmin}_{s\in\Pi_m} \mathcal{Q}(s). \tag{33}$$

Solution of problem (33) is detailed in the next sections. The proposed solution to the first problem is by means of the Chebyshev polynomials, while the solution of the second problem is done with the Jacobi weight.

5.1. Chebyshev Polynomial Preconditioner. The solution of minimization problem (33) with quality measure $\mathcal{Q}_{[\epsilon,1]}(s)$ is well known and can be written in terms of Chebyshev polynomials [21, 22]:

$$1-\lambda s_m(\lambda) = \frac{T^{\epsilon}_{m+1}(\lambda)}{T^{\epsilon}_{m+1}(0)}, \tag{34}$$

where $T^{\epsilon}_{m+1}(\lambda)$ is the $(m+1)$th Chebyshev polynomial scaled in the interval $[\epsilon,1]$. Polynomials $T^{\epsilon}_k(\lambda)$ satisfy the recurrence

$$\begin{gathered} T^{\epsilon}_0(x) = 1, \qquad T^{\epsilon}_1(x) = ax+b, \\ T^{\epsilon}_{n+1}(x) = 2(ax+b)T^{\epsilon}_n(x) - T^{\epsilon}_{n-1}(x), \end{gathered} \tag{35}$$

where

$$a = \frac{2}{1-\epsilon}, \qquad b = -\frac{1+\epsilon}{1-\epsilon}. \tag{36}$$

From (34), the preconditioner polynomial $s_m(\lambda)$ becomes

$$s_m(\lambda) = \frac{1}{\lambda}\left(1 - \frac{T^{\epsilon}_{m+1}(\lambda)}{T^{\epsilon}_{m+1}(0)}\right) \tag{37}$$

and from (35) it is possible to give a recursive definition for $s_m(\lambda)$ too.

Lemma 10 (recurrence formula for preconditioner). *Given the polynomials q_n defined by the recurrence*

$$\begin{gathered} q_0(x) = 1, \qquad q_1(x) = a_0x + b_0, \\ q_{n+1}(x) = (a_nx+b_n)q_n(x) + c_nq_{n-1}(x), \quad n = 1,2,3,\ldots \end{gathered} \tag{38}$$

then the polynomials $r_n(x) = q_n(x)/q_n(0)$ and $s_n(x) = (1 - r_{n+1}(x))/x$ satisfy the recurrences

$$\begin{gathered} r_0(x) = 1, \qquad r_1(x) = 1 - a_0'x, \\ r_{n+1}(x) = (a_n'x + b_n')r_n(x) + c_n'r_{n-1}(x), \\ s_0(x) = a_0', \qquad s_1(x) = a_1'x + b_1', \\ s_n(x) = (a_n'x+b_n')s_{n-1}(x) + c_n's_{n-2}(x) - a_n', \end{gathered} \tag{39}$$

where

$$\begin{gathered} a_0' = -\frac{a_0}{b_0}, \qquad a_1' = -\frac{a_0a_1}{b_0b_1+c_1}, \qquad b_1' = -\frac{a_0b_1+a_1b_0}{b_0b_1+c_1}, \\ a_n' = a_n\gamma_n, \qquad b_n' = b_n\gamma_n, \qquad c_n' = c_n\gamma_{n-1}\gamma_n \end{gathered} \tag{40}$$

and $\gamma_n = q_n(0)/q_{n+1}(0)$ satisfies the recurrence

$$\gamma_1 = \frac{b_0}{b_0b_1+c_1}, \qquad \gamma_n = \frac{1}{b_n + c_n\gamma_{n-1}}. \tag{41}$$

Proof. Take the ratio

$$\begin{aligned} \frac{q_{n+1}(x)}{q_{n+1}(0)} &= (a_nx+b_n)\frac{q_n(x)}{q_{n+1}(0)} + c_n\frac{q_{n-1}(x)}{q_{n+1}(0)}, \\ r_{n+1}(x) &= (a_nx+b_n)r_n(x)\frac{q_n(0)}{q_{n+1}(0)} \\ &\quad + c_nr_{n-1}(x)\frac{q_{n-1}(0)}{q_n(0)}\frac{q_n(0)}{q_{n+1}(0)}, \\ r_{n+1}(x) &= (a_nx+b_n)r_n(x)\gamma_n + c_nr_{n-1}(x)\gamma_n\gamma_{n-1}, \end{aligned} \tag{42}$$

and notice that $r_n(0) = 1$ for all n. Recurrence for γ_n is trivially deduced. From $r_{n+1}(x) = 1 - xs_n(x)$ and by using (41),

$$\begin{aligned} r_{n+1}(x) &= (a_nx+b_n)r_n(x)\gamma_n + c_nr_{n-1}(x)\gamma_n\gamma_{n-1}, \\ 1 - xs_n(x) &= (a_nx+b_n)(1-xs_{n-1}(x))\gamma_n \\ &\quad + c_n(1-xs_{n-2}(x))\gamma_n\gamma_{n-1}, \\ 1 - xs_n(x) &= a_nx(1-xs_{n-1}(x))\gamma_n + b_n\gamma_n \\ &\quad - b_nxs_{n-1}(x)\gamma_n + c_n\gamma_n\gamma_{n-1} \\ &\quad - c_nxs_{n-2}(x)\gamma_n\gamma_{n-1}, \\ -xs_n(x) &= a_nx\gamma_n - a_nx^2s_{n-1}(x)\gamma_n - b_nxs_{n-1}(x)\gamma_n \\ &\quad - c_nxs_{n-2}(x)\gamma_n\gamma_{n-1}, \\ -xs_n(x) &= -x\gamma_n\left[(a_nx+b_n)s_{n-1}(x) + c_ns_{n-2}(x)\gamma_{n-1} - a_n\right], \end{aligned} \tag{43}$$

dividing by $-x$ recurrence (39) is retrieved. Polynomials s_0 and s_1 are trivially computed. □

Using Lemma 10 the polynomial preconditioner (34) satisfies the recurrence (39) with

$$\begin{gathered} a_0' = \frac{2}{1+\epsilon}, \qquad a_1' = \frac{-8}{\epsilon^2+6\epsilon+1}, \qquad b_1' = 8\frac{1+\epsilon}{\epsilon^2+6\epsilon+1}, \\ a_n' = \frac{4\gamma_n}{1-\epsilon}, \qquad b_n' = -2\gamma_n\frac{1+\epsilon}{1-\epsilon}, \qquad c_n' = -\gamma_n\gamma_{n-1}, \end{gathered} \tag{44}$$

where $c_n = -1$ is used and $\gamma_n = T_n^\epsilon(0)/T_{n+1}^\epsilon(0)$ is computed by solving recurrence (35) for $x = 0$; that is,

$$T_n^\epsilon(0) = \frac{1}{2}\left(c^n + c^{-n}\right), \quad c = \frac{\sqrt{\epsilon}-1}{\sqrt{\epsilon}+1}, \\ \Longrightarrow \gamma_n = \frac{T_n^\epsilon(0)}{T_{n+1}^\epsilon(0)} = \frac{c^n + c^{-n}}{c^{n+1} + c^{-(n+1)}}. \tag{45}$$

Numerical stability of recurrence (44) is discussed in Section 6. The estimation of ϵ is the complex task and some authors perform it dynamically. As an alternative, the present approach is to move the eigenvalues of the coefficient matrix from the interval $[\epsilon, 1]$ to a stripe $[1-\delta, 1+\delta]$, so that the condition number remains bounded. The value of ϵ is not determined from the estimate of the eigenvalues but from the degree of the preconditioner polynomial and from the amplitude of the stripe δ. Once δ is fixed, the higher the degree of the preconditioner, the lower the value of ϵ, which decreases to zero. Thus, if the degree of the preconditioner is high enough, the eigenvalues are moved in the interval $[1-\delta, 1+\delta]$. The important fact is that even if the degree is not high enough to move the complete spectrum, the majority of the eigenvalues are moved in the desired stripe, improving the performance of the conjugate gradient method. An idea of this behaviour is showed in Figure 1 on the right. The end of this section is devoted to the explicit expression of the value of ϵ computed backwards from the value of δ_n: once the maximum condition number is fixed, it is possible to increase the degree of the polynomial preconditioner so that ϵ decreases until the whole (or at least the most) spectrum of the matrix is contained in the specified range.

In the interval $[\epsilon, 1]$, Chebyshev polynomial $T_n^\epsilon(x)/T_n^\epsilon(0)$ is bounded in the range $[-\delta, \delta]$ where $\delta = T_n^\epsilon(0)^{-1} = 2/(c^{n+1} + c^{-(n+1)})$ and solving for ϵ gives

$$\epsilon = \left(\frac{|c|-1}{|c|+1}\right)^2, \quad |c| = \left(\frac{1+\sqrt{1-\delta^2}}{\delta}\right)^{1/(n+1)}. \tag{46}$$

5.2. Jacobi Polynomial Preconditioner. The solution of minimization problem (33) with quality measure $\mathcal{Q}(s)$ is well known and can be written in terms of Jacobi orthogonal polynomials [24].

Definition 11. Given a nonnegative weight function $w(\lambda) : [0,1] \mapsto \mathbb{R}^+$ (see [23, 24, 56]) the scalar product $\langle\cdot,\cdot\rangle_w$ and the relative induced norm $\|\cdot\|_w$ are defined as

$$\langle p, q\rangle_w = \int_0^1 p(\lambda)\, q(\lambda)\, w(\lambda)\, \mathrm{d}\lambda, \\ \|p\|_w = \sqrt{\langle p, p\rangle_w} = \sqrt{\int_0^1 p(\lambda)^2\, w(\lambda)\, \mathrm{d}\lambda}, \tag{47}$$

where p and q are continuous functions. The orthogonal polynomials with respect to the scalar product $\langle\cdot,\cdot\rangle_w$ are the polynomials $p_k(\lambda)$ which satisfy $\langle p_k, p_j\rangle_w = 0$ if $k \neq j$.

The orthogonal polynomials with respect to the weight

$$w^{\alpha,\beta}(\lambda) = (1-\lambda)^\alpha \lambda^\beta, \quad \text{for } \alpha, \beta > -1, \tag{48}$$

defined in the interval $[0,1]$, are the Jacobi polynomials and they satisfy the recurrence (see [57]):

$$p_0^{\alpha,\beta}(x) = 1, \qquad p_1^{\alpha,\beta}(x) = a_0^{\alpha,\beta}x + b_0^{\alpha,\beta}, \\ p_{n+1}^{\alpha,\beta}(x) = \left(a_n^{\alpha,\beta}x + b_n^{\alpha,\beta}\right)p_n^{\alpha,\beta}(x) + c_n^{\alpha,\beta}p_{n-1}^{\alpha,\beta}(x), \tag{49}$$

for

$$a_n^{\alpha,\beta} = 1, \\ b_n^{\alpha,\beta} = -\frac{1}{2}\left(1 + \frac{\beta^2-\alpha^2}{(2n+\alpha+\beta)(2(n+1)+\alpha+\beta)}\right), \\ c_n^{\alpha,\beta} = -\frac{n(n+\alpha)(n+\beta)(n+\alpha+\beta)}{(2n-1+\alpha+\beta)(2n+1+\alpha+\beta)(2n+\alpha+\beta)^2}. \tag{50}$$

The class of polynomials of the form $1-\lambda s(\lambda)$ with $s \in \Pi_{m-1}$ can be thought as polynomials $r \in \Pi_{m+1}$ with $r(0) = 1$; thus, the minimization problem (33) for $\mathcal{Q}(s)$ can be recast to the following constrained minimization for $w(\lambda) \equiv 1$:

$$r_{m+1} = \operatorname*{argmin}_{r\in\Pi_{m+1}, r(0)=1} \|r\|_w. \tag{51}$$

The preconditioner polynomial is $s_m(\lambda) = \lambda^{-1}(1 - r_{m+1}(\lambda))$. Polynomial $r_{m+1}(\lambda)$ is expanded by means of Jacobi orthogonal polynomials with $\alpha = \beta = 0$:

$$r_{m+1}(\lambda) = \sum_{k=0}^{m+1}\alpha_k p_k^{0,0}(\lambda). \tag{52}$$

Making use of the property of orthogonality with respect to the scalar product, one has

$$\|r_{m+1}\|_w^2 = \sum_{k=0}^{m+1}\alpha_k^2\left\|p_k^{0,0}\right\|_w^2, \\ 1 = r_{m+1}(0) = \sum_{k=0}^{m+1}\alpha_k p_k^{0,0}(0), \tag{53}$$

and thus the constrained minimum problem (51) is recast as

$$\text{minimize} \quad \sum_{k=0}^{m+1}\alpha_k^2\left\|p_k^{0,0}\right\|_w^2, \\ \text{subject to} \quad \sum_{k=0}^{m+1}\alpha_k p_k^{0,0}(0) = 1. \tag{54}$$

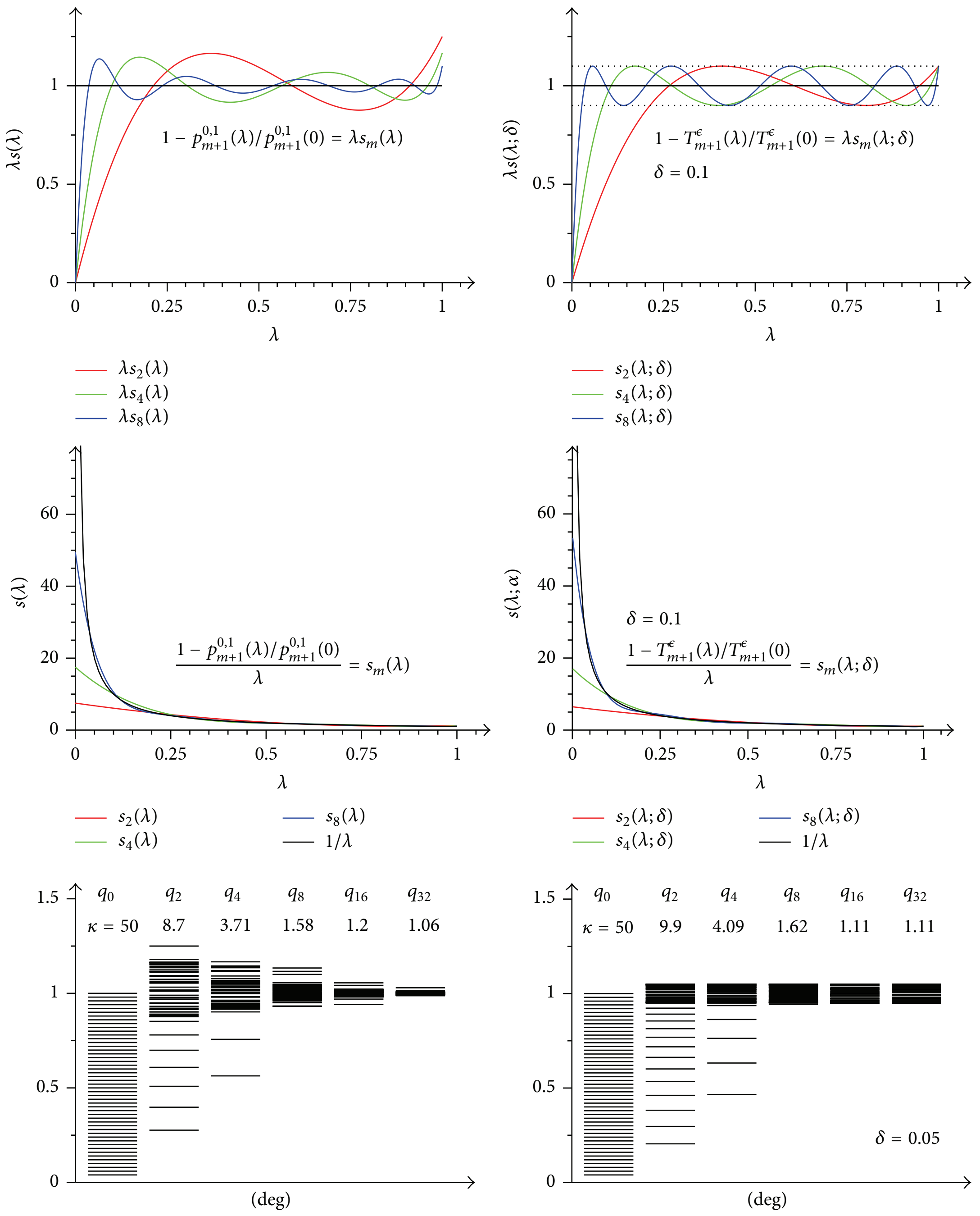

FIGURE 1: The polynomials described in Lemma 10: in the first row the product $\lambda s_m(\lambda)$ showing the approximation of the identity; in the second row the explicit graph of the polynomial preconditioner compared with the function $1/\lambda$; in the third row the performance of the preconditioner in terms of the degree, condition number, and concentration of the eigenvalues of the coefficient matrix around 1. The left column represents the Jacobi weight and the right column the Chebyshev polynomials.

Problem (54) is solved by using Lagrange multiplier with first order conditions resulting in

$$\alpha_k = \frac{p_k(0)/\left\|p_k^{0,0}\right\|_w^2}{\sum_{k=0}^{m+1}\left(p_k^{0,0}(0)^2/\left\|p_k^{0,0}\right\|_w^2\right)},$$
$$\Longrightarrow r_{m+1}(\lambda) = \frac{\sum_{k=0}^{m+1}\left(p_k^{0,0}(0)\,p_k^{0,0}(\lambda)/\left\|p_k^{0,0}\right\|_w^2\right)}{\sum_{k=0}^{m+1}\left(p_k^{0,0}(0)^2/\left\|p_k^{0,0}\right\|_w^2\right)}. \tag{55}$$

Remark 12. The solution (55) is only formal but barely useful from a computational point of view, because it requires computing explicitly the least squares polynomial. In fact, it is well known that the evaluation of a polynomial of high degree is a very unstable process. To make solution (55) practical, it is mandatory to obtain a stable recurrence formula that allows evaluating polynomials even of very high degree, for example, 1000 or more.

To find a recurrence for (55), it must be rewritten as a ratio of orthogonal polynomials as for the Chebyshev preconditioner (34). To this scope, some classical theorems and definitions on orthogonal polynomials are here recalled for convenience. Christoffel-Darboux formulas and Kernel Polynomials, here recalled without proofs (see [55, 57]), are used to build the recurrence.

Theorem 13 (Christoffel-Darboux formulas). *Orthogonal polynomials with respect to the scalar product $\langle\cdot,\cdot\rangle_w$ share the following identities:*

$$\sum_{j=0}^{k}\frac{p_j(x)\,p_j(y)}{\|p_j\|_w^2} = \frac{1}{\|p_k\|_w^2}\,\frac{p_{k+1}(x)\,p_k(y)-p_{k+1}(y)\,p_k(x)}{x-y}. \tag{56}$$

Theorem 14 (Kernel Polynomials). *Given orthogonal polynomials $p_k(x)$ with respect to the scalar product $\langle\cdot,\cdot\rangle_w$, that is, with respect to weight function $w(x)$, then the polynomials*

$$q_k(x) = \frac{\left(p_{k+1}(x)\,p_k(0)-p_{k+1}(0)\,p_k(x)\right)}{x} \tag{57}$$

are orthogonal polynomials with respect to the scalar product $[\cdot,\cdot]_w$ defined as $[p,q]_w = \int_0^1 p(x)q(x)xw(x)\mathrm{d}x$, that is, with respect to the weight function $xw(x)$. Moreover $q_0(x)=1$.

With the formulas of Christoffel-Darboux (56) and $x=\lambda$, $y=0$, it is possible to rewrite (55) as

$$r_{m+1}(\lambda) = \frac{1}{C}\,\frac{p_{m+2}^{0,0}(\lambda)\,p_{m+1}^{0,0}(0)-p_{m+2}^{0,0}(0)\,p_{m+1}^{0,0}(\lambda)}{\lambda}, \qquad C = \left\|p_{m+1}^{0,0}\right\|_w^2\sum_{k=0}^{m+1}\frac{p_k^{0,0}(0)^2}{\left\|p_k^{0,0}\right\|_w^2}. \tag{58}$$

Using the Kernel Polynomials of this last theorem, expression (58) becomes

$$r_{m+1}(\lambda) = \frac{p_{m+1}^{0,1}(\lambda)}{p_{m+1}^{0,1}(0)}, \tag{59}$$

where $p_k^{0,1}(x)$ are orthogonal polynomials with respect to the weight $w(\lambda)=\lambda$. In fact, the Kernel Polynomials with respect to $\lambda w^{\alpha,\beta}(\lambda) = w^{\alpha,\beta+1}(\lambda)$ satisfy $q^{\alpha,\beta}(x) = p^{\alpha,\beta+1}(x)$.

Preconditioner polynomial can be computed recursively using Lemma 10, where coefficients a_n', b_n', c_n', and γ_n are computed from $a_n^{0,1}$, $b_n^{0,1}$, and $c_n^{0,1}$. Given

$$a_n^{0,1} = 1, \qquad b_n^{0,1} = -\frac{1}{2}\left(1+\frac{1}{(2n+1)(2n+3)}\right), \qquad c_n^{0,1} = -\frac{n(n+1)}{4(2n+1)^2}, \tag{60}$$

```
t_1 ← $a_0'$v;
y ← $a_1'$Mv + $b_1'$v;
for $n = 2, 3, \ldots, m$ do
    t_0 ← t_1;
    t_1 ← y;
    y ← $a_n'(\mathbf{M}\mathbf{t}_1 - \mathbf{v}) + b_n'\mathbf{t}_1 + c_n'\mathbf{t}_0$;
end for
return y;
```

Algorithm 4: Application of preconditioner $s_m(\mathbf{M})$ to a vector $\mathbf{v}$.

the values of a_n', b_n', c_n', and γ_n from Lemma 10 become

$$a_0' = \frac{3}{2}, \qquad a_1' = -\frac{10}{3}, \qquad b_1' = 4,$$
$$\gamma_n = a_n' = -4 + \frac{2(3n+5)}{(n+2)^2}, \qquad b_n' = 2-\Delta,$$
$$c_n' = -1+\Delta, \qquad \Delta = \frac{2\left(3n^2+6n+2\right)}{(2n+1)(n+2)^2}. \tag{61}$$

The only difficulty of the previous coefficients computation lies in the recursive solution of γ_n, which is here omitted for conciseness but that is a linear three-term recurrence with polynomial coefficients. Notice that $\Delta \to 0$; thus the limit value of the above coefficients is evident.

5.3. Recurrence Formula for the Preconditioner. Looking at Algorithms 2 and 3, the polynomial preconditioner $s_m(\mathbf{M})$ is applied to a vector, that is, $s_m(\mathbf{M})\mathbf{v}$. Thus, to avoid matrix-matrix multiplication, by defining $\mathbf{v}^{(k)} = s_k(\mathbf{M})\mathbf{v}$ and using Lemma 10 with (61), the following recurrence is obtained for $s_m(\mathbf{M})\mathbf{v} = \mathbf{v}^{(m)}$:

$$\mathbf{v}^{(0)} = a_0'\mathbf{v}, \qquad \mathbf{v}^{(1)} = a_1'\mathbf{M}\mathbf{v} + b_1'\mathbf{v},$$
$$\mathbf{v}^{(n)} = a_n'\left(\mathbf{M}\mathbf{v}^{(n-1)} - \mathbf{v}\right) + b_n'\mathbf{v}^{(n-1)} + c_n'\mathbf{v}^{(n-2)}, \tag{62}$$

where $n = 2, 3, \ldots, m$ and recurrence (62) is the proposed preconditioner with coefficients given by (44) for Chebyshev and (61) for Jacobi the polynomials. Equation (62) yields Algorithm 4.

6. Numerical Stability

Algorithm 4, that is, the application of preconditioner $s_m(\mathbf{M})$ given by (62) to a vector $\mathbf{v}$, also taking into account rounding errors, results in

$$\mathbf{w}^{(0)} = a_0'\mathbf{v} + \boldsymbol{\varrho}^{(0)}, \qquad \mathbf{w}^{(1)} = a_1'\mathbf{M}\mathbf{v} + b_1'\mathbf{v} + \boldsymbol{\varrho}^{(1)},$$
$$\mathbf{w}^{(n)} = a_n'\left(\mathbf{M}\mathbf{w}^{(n-1)} - \mathbf{v}\right) + b_n'\mathbf{w}^{(n-1)} + c_n'\mathbf{w}^{(n-2)} + \boldsymbol{\varrho}^{(n)}, \tag{63}$$

where $\|\boldsymbol{\varrho}^{(k)}\|_\infty \leq \delta$ are the errors due to floating point operations with δ as an upper bound of such errors. The cumulative error $\mathbf{e}^{(k)} = \mathbf{w}^{(k)} - \mathbf{v}^{(k)}$ satisfies the linear recurrence

$$\begin{gathered} \mathbf{e}^{(0)} = \boldsymbol{\varrho}^{(0)}, \qquad \mathbf{e}^{(1)} = \boldsymbol{\varrho}^{(1)}, \\ \mathbf{e}^{(n)} = a_n' \mathbf{M}\mathbf{e}^{(n-1)} + b_n' \mathbf{e}^{(n-1)} + c_n' \mathbf{e}^{(n-2)} + \boldsymbol{\varrho}^{(n)}. \end{gathered} \tag{64}$$

The next definitions introduce the concept of generalized and joint spectral radius needed for the proof of the theorem of the matrix bound; they can be found in [58].

Definition 15. A matrix set $\Sigma = \{\mathbf{A}_k \in \mathbb{R}^{n\times n} \mid k \in \mathbb{N}\}$ is bounded if there is a constant C such that $\|\mathbf{A}\| \leq C$ for all $\mathbf{A} \in \Sigma$. An invariant subspace V for Σ is a vector space such that $\mathbf{A}V \subseteq V$ for all $\mathbf{A} \in \Sigma$. The set Σ is irreducible if the only invariant subspace is $\{\mathbf{0}\}$ or $\mathbb{R}^n$.

Definition 16. The generalized spectral radius $\varrho(\Sigma)$ and the joint spectral radius $\widehat{\varrho}(\Sigma, \|\cdot\|)$ of any set of matrices Σ are defined as

$$\begin{gathered} \varrho(\Sigma) = \limsup_{k\to\infty} \left(\varrho_k(\Sigma)\right)^{1/k}, \\ \varrho_k(\Sigma) = \sup\left\{\varrho\left(\prod_{i=1}^k \mathbf{A}_i\right) \mid \mathbf{A}_i \in \Sigma\right\}, \\ \widehat{\varrho}(\Sigma, \|\cdot\|) = \limsup_{k\to\infty} \left(\widehat{\varrho}_k(\Sigma, \|\cdot\|)\right)^{1/k}, \\ \widehat{\varrho}_k(\Sigma, \|\cdot\|) = \sup\left\{\left\|\prod_{i=1}^k \mathbf{A}_i\right\| \mid \mathbf{A}_i \in \Sigma\right\}. \end{gathered} \tag{65}$$

Theorem 17. *Let Σ be a bounded and irreducible set of matrices with $\varrho(\Sigma) > 0$; then there is a constant C such that*

$$\|\mathbf{A}_1\mathbf{A}_2\cdots\mathbf{A}_k\| \leq C\varrho(\Sigma)^k, \quad \forall \mathbf{A}_j \in \Sigma \tag{66}$$

for all $k > 0$.

Proof. It is Theorem 2.1 by [58] with a slight modification to match the present case. □

Theorem 18. *Recurrence (64) satisfies*

$$\left\|\mathbf{e}^{(n)}\right\|_\infty \leq (C\delta N)\, n, \tag{67}$$

where N is the dimension of the linear system, δ is the amplitude of the stripe for the eigenvalues, and C is an unknown constant coming from the norm inequalities which is found experimentally to be small.

Proof. The matrix $\mathbf{M} = \mathbf{B} + \mathbf{C}$ in (64), by Assumption 9, is SPD with eigenvalues in $(0, 1]$. Thus $\mathbf{M} = \mathbf{T}^T\mathbf{\Lambda}\mathbf{T}$, with $\mathbf{T}$ orthogonal matrix, that is, $\mathbf{T}^T\mathbf{T} = \mathbf{I}$, and $\mathbf{\Lambda}$ diagonal. Multiplying on the left the recurrence (64) by $\mathbf{T}$, the following error estimate is obtained:

$$\begin{gathered} \mathbf{T}\mathbf{e}^{(0)} = \mathbf{T}\boldsymbol{\varrho}^{(0)}, \qquad \mathbf{T}\mathbf{e}^{(1)} = \mathbf{T}\boldsymbol{\varrho}^{(1)}, \\ \mathbf{T}\mathbf{e}^{(n)} = a_n'\mathbf{\Lambda}\mathbf{T}\mathbf{e}^{(n-1)} + b_n'\mathbf{T}\mathbf{e}^{(n-1)} + c_n'\gamma_{n-1}\mathbf{T}\mathbf{e}^{(n-2)} + \mathbf{T}\boldsymbol{\varrho}^{(n)}. \end{gathered} \tag{68}$$

Focusing on jth component of the transformed error, $f^{(n)} = (\mathbf{T}\mathbf{e}^{(n)})_j$ and $\eta^{(n)} = (\mathbf{T}\boldsymbol{\varrho}^{(n)})_j$, a scalar recurrence is obtained:

$$\begin{gathered} f^{(0)} = \eta^{(0)}, \qquad f^{(1)} = \eta^{(1)}, \\ f^{(n)} = \left(a_n'\lambda_j + b_n'\right) f^{(n-1)} + c_n' f^{(n-2)} + \eta^{(n)}. \end{gathered} \tag{69}$$

Recurrence (69) is restated in matrix form as

$$\begin{gathered} \mathbf{f}_n = \mathbf{A}_n\mathbf{f}_{n-1} + \mathbf{b}_n, \qquad \mathbf{A}_n = \begin{pmatrix} a_n'\lambda_j + b_n' & c_n' \\ 1 & 0 \end{pmatrix}, \\ \mathbf{b}_n = \begin{pmatrix} \eta^{(n)} \\ 0 \end{pmatrix}, \qquad \mathbf{f}_n = \begin{pmatrix} f^{(n)} \\ f^{(n-1)} \end{pmatrix}, \end{gathered} \tag{70}$$

with initial data $\mathbf{f}_1^T = (\eta^{(1)}, \eta^{(0)})$. Notice that $\eta^{(n)}$ is bounded by

$$\begin{aligned} \left|\eta^{(n)}\right| &\leq \left\|\mathbf{T}\boldsymbol{\varrho}^{(n)}\right\|_\infty \leq \left\|\mathbf{T}\boldsymbol{\varrho}^{(n)}\right\|_2 \\ &= \left\|\boldsymbol{\varrho}^{(n)}\right\|_2 \leq \sqrt{N}\left\|\boldsymbol{\varrho}^{(n)}\right\|_\infty \leq \delta\sqrt{N} \end{aligned} \tag{71}$$

and thus, $\|\mathbf{f}_1\|_\infty \leq \delta\sqrt{N}$ and $\|\mathbf{b}_n\|_\infty \leq \delta\sqrt{N}$. From (70) it follows that

$$\mathbf{f}_n = \mathbf{A}_n\mathbf{A}_{n-1}\cdots\mathbf{A}_2\mathbf{f}_1 + \sum_{k=2}^n \mathbf{A}_n\mathbf{A}_{n-1}\cdots\mathbf{A}_{n-k+1}\mathbf{b}_k. \tag{72}$$

The set $\Sigma = \{\mathbf{A}_i \mid i = 1, \ldots, \infty\}$ is bounded and irreducible; each matrix has spectral radius strictly less than 1 (see Lemma 19 for a proof); therefore the joint spectral radius is less than 1. From (72), with Theorem 17 using the infinity norm,

$$\begin{aligned} \left|f^{(n)}\right| &\leq \left\|\mathbf{f}_n\right\|_\infty \leq \left\|\mathbf{A}_n\mathbf{A}_{n-1}\cdots\mathbf{A}_2\right\|_\infty \left\|\mathbf{f}_1\right\|_\infty \\ &\quad + \sum_{k=2}^n \left\|\mathbf{A}_n\mathbf{A}_{n-1}\cdots\mathbf{A}_{n-k+1}\right\|_\infty \left\|\mathbf{b}_k\right\|_\infty, \\ &\leq C\varrho(\Sigma)^{n-2}\delta\sqrt{N} + C\sum_{k=2}^n \varrho(\Sigma)^k\delta\sqrt{N} \\ &\leq C\delta\sqrt{N}n, \end{aligned} \tag{73}$$

and, because of $f^{(n)} = (\mathbf{T}\mathbf{e}^{(n)})_j$, it follows that $\|\mathbf{T}\mathbf{e}^{(n)}\|_\infty \leq C\delta\sqrt{N}n$. A bound of the term $\mathbf{e}^{(n)}$ is done as

$$\left\|\mathbf{e}^{(n)}\right\|_\infty \leq \left\|\mathbf{e}^{(n)}\right\|_2 = \left\|\mathbf{T}\mathbf{e}^{(n)}\right\|_2 \leq \sqrt{N}\left\|\mathbf{T}\mathbf{e}^{(n)}\right\|_\infty \leq C\delta N n. \tag{74}$$

This shows that the error grows at most linearly. □

The above relation shows that the recurrence is at worst linearly unstable; that is, the error grows at most linearly. The existence is proved in the works of Rota and Strang [59] where the concept of joint spectral radius is introduced. The determination of C is not possible but practice reveals that it is small. In conclusion it is possible to employ even a very high degree polynomial preconditioner with a stable computation.

Lemma 19. *Given a_n', b_n', c_n' from (44) or (61), respectively, for the Chebyshev and the Jacobi preconditioner, then the roots z_1 and z_2 of the characteristic polynomial of homogeneous recurrence (69), that is,*

$$z^2 - \left(a_n'\lambda + b_n'\right) z - c_n', \tag{75}$$

satisfy $|z_1| < 1$ and $|z_2| < 1$ for all $n > 0$ and $0 < \lambda \leq 1$.

Proof. Consider first the coefficients for the Jacobi polynomials defined in (61). If the roots are complex, then they must be conjugate; thus $z_1 = z$ and $z_2 = \bar{z}$, because the coefficients of the polynomial are real. In that case, the constant term of the polynomial is equal to the square of the modulus of the roots, $z\bar{z} = |z|^2 = -c_n'$; thus it is easy to see that $|z| < 1$ for all $n > 0$. Suppose now that the two roots z_1 and z_2 are real; multiplying the characteristic polynomial by $(n+2)^2(2n+1)$, yields, after some manipulation,

$$\begin{aligned} z_1 &= \frac{A - B\lambda + \sqrt{B^2\lambda^2 - 2AB\lambda + C}}{D}, \\ z_2 &= \frac{A - B\lambda - \sqrt{B^2\lambda^2 - 2AB\lambda + C}}{D} \end{aligned} \tag{76}$$

for

$$\begin{aligned} A &= 2n^3 + 6n^2 + 6n + 2, \qquad B = 4n^3 + 12n^2 + 11n + 3, \\ C &= \left(3n^2 + 6n + 2\right)^2, \qquad D = 2n^3 + 9n^2 + 12n + 4, \end{aligned} \tag{77}$$

with A, B, C, and D strictly positive for all $n \geq 0$. The discriminant $\Delta(\lambda)$ of the equation is $\Delta(\lambda) = B^2\lambda^2 - 2AB\lambda + C$ and represents a convex parabola because $B^2 > 0$. Its minimum is obtained for $\lambda = A/B \in (0,1)$, which gives $\Delta(A/B) < 0$ and so complex roots, but this case was already considered. Hence we can set $\Delta_{\min} = 0$. The maximum of $\Delta(\lambda)$ is achieved at one of the extrema of the interval of definition of λ. A quick calculation shows that $\Delta(\lambda)$ is maximum for $\lambda = 0$, yielding a value of $\Delta_{\max} = C$. Using $\Delta_{\max}$ and $\Delta_{\min}$ it is possible to bound the roots z_1 and z_2:

$$\begin{aligned} z_1 &< \frac{A + \sqrt{\Delta_{\max}}}{D} \\ &= \frac{2n^3 + 6n^2 + 6n + 2 + \left(3n^2 + 6n + 2\right)}{2n^3 + 9n^2 + 12n + 4} = 1, \\ z_1 &\geq \frac{A - B + \sqrt{\Delta_{\min}}}{D} = -\frac{2n^3 + 6n^2 + 5n + 1}{2n^3 + 9n^2 + 12n + 4} > -1, \\ z_2 &< \frac{A - \sqrt{\Delta_{\min}}}{D} = \frac{2n^3 + 6n^2 + 6n + 2}{2n^3 + 9n^2 + 12n + 4} < 1, \\ z_2 &\geq \frac{A - B - \sqrt{\Delta_{\max}}}{D} = -\frac{2n^3 + 9n^2 + 11n + 3}{2n^3 + 9n^2 + 12n + 4} > -1. \end{aligned} \tag{78}$$

The previous inequalities prove the lemma for the Jacobi preconditioner. Now consider the case of the coefficients of the Chebyshev polynomials defined in (44). Recall the expression for $c = (\sqrt{\epsilon} - 1)/(\sqrt{\epsilon} + 1)$, and notice that, for $\epsilon \in (0,1)$, c is bounded in $-1 < c < 0$, so that

$$\omega_n := c^n + c^{-n} \tag{79}$$

is positive for even n and negative for odd n; moreover, $|\omega_n| = |c^n + c^{-n}| \geq 2$ is monotone increasing for $n = 1, 2, 3, \ldots$. In the case of complex roots, it was already shown that $z\bar{z} = |z|^2 = -c_n'$, with

$$0 \leq -c_n' = \frac{\omega_{n-1}}{\omega_{n+1}} < 1, \quad \epsilon \in (0,1). \tag{80}$$

In the rest of the proof it is useful to consider also the ratio $-1 < \omega_n/\omega_{n+1} < 0$, for $c \in (-1, 0)$, that corresponds to $\epsilon \in (0,1)$. The coefficients of (44), observing that $\epsilon = (\omega_1 + 2)/(\omega_1 - 2)$, are simplified in

$$\begin{aligned} a_n' &= \frac{4\omega_n}{(1-\epsilon)\,\omega_{n+1}} = -\frac{\omega_n}{\omega_{n+1}}\left(\omega_1 - 2\right), \\ b_n' &= \frac{-2(1+\epsilon)\,\omega_n}{(1-\epsilon)\,\omega_{n+1}} = \frac{\omega_n\omega_1}{\omega_{n+1}}, \qquad c_n' = -\frac{\omega_{n-1}}{\omega_{n+1}}. \end{aligned} \tag{81}$$

Polynomial (75) is rewritten as

$$z^2 + \frac{\omega_n}{\omega_{n+1}}\left(\lambda\left(\omega_1 - 2\right) - \omega_1\right) z + \frac{\omega_{n-1}}{\omega_{n+1}} \tag{82}$$

and its roots are (using $\omega_1\omega_n = \omega_{n-1} + \omega_{n+1}$)

$$\begin{aligned} z_{1,2}(\lambda) &= \frac{\omega_1\omega_n}{2\omega_{n+1}} - \frac{\omega_n}{\omega_{n+1}}\frac{\omega_1 - 2}{2}\left[\lambda \pm \sqrt{\Delta(\lambda)}\right], \\ \Delta(\lambda) &= \lambda^2 - \frac{2\omega_1}{\omega_1 - 2}\lambda + \frac{\left(\omega_{n-1} - \omega_{n+1}\right)^2}{\omega_n^2\left(\omega_1 - 2\right)^2}. \end{aligned} \tag{83}$$

Looking at the discriminant $\Delta(\lambda)$, the minimum of the associated convex parabola is for $\lambda = \omega_1/(\omega_1 - 2) \in (1/2, 1)$. The corresponding value is

$$\Delta\left(\frac{\omega_1}{\omega_1 - 2}\right) = -4\frac{\omega_{n-1}\omega_{n+1}}{\omega_n^2\left(\omega_1 - 2\right)^2} < 0. \tag{84}$$

The value at the right extremum is also negative:

$$\Delta(1) = 4\frac{\omega_n^2 - \omega_{n-1}\omega_{n+1}}{\omega_n^2\left(\omega_1 - 2\right)^2} < 0, \tag{85}$$

and in fact

$$\omega_n^2 - \omega_{n-1}\omega_{n+1} = 2 - c^{-2} - c^2 = -\frac{(c-1)^2(c+1)^2}{c^2} < 0. \tag{86}$$

For $\lambda = 0$, with some manipulations, the roots of (82) are

$$z_1(0) = 1, \qquad z_2(0) = \frac{\omega_{n-1}}{\omega_{n+1}} < 1. \tag{87}$$

Table 1: Numerical results for COCG, COCR, and QMR preconditioned on the basis of the complex matrix **A** with LU decomposition, with incomplete LU (ILU) and with incomplete LU with threshold 10^{-5} (ILU0). The reported numbers represent the iterations of the corresponding solver with the specified preconditioner. The dash indicates that it was not feasible to compute a particular test. The column "preconditioner" shows the time required to assemble the preconditioner. The stopping tolerance was 10^{-8}. The time elapsed in the computation is expressed in seconds.

Test case	Preconditioner			COCG		COCR		QMR		
	LU	ILU0	ILU	ILU0	ILU	ILU0	ILU	No Preco	ILU0	ILU
T5k										
Iter.				86	9	86	9		85	8
Time	0.59	0.027	1.28	**0.21**	1.34	0.22	1.34	—	0.65	1.73
T16k										
Iter.				65	6	65	6	399	64	5
Time	13.9	0.08	315	**0.44**	317	**0.44**	317	1.75	1.35	337
T80k										
Iter.					204		205			200
Time	21.8	0.26	49.6	—	**73**	—	74	—	—	266
T150k										
Iter.				661	24	634	24		642	23
Time	13.0	0.058	152	**12.7**	156	13.1	156	—	40.4	174
T500k										
Iter.										
Time	—	0.79	—	—	—	—	—	—	—	—
S13k										
Iter.					1253		1258		2152	1307
Time	—	0.032	6.15	—	36.5	—	36.7	—	**29.4**	333
S32k										
Iter.										
Time	—	0.082	28	—	—	—	—	—	—	—
S500k										
Iter.										
Time	—	—	—	—	—	—	—	—	—	—

Moreover, $\Delta(\epsilon) = \Delta(1)$; thus there exists $0 < \lambda^\star < \epsilon$ such that $\Delta(\lambda^\star) = 0$. Thus for $\lambda \in [\lambda^\star, 1]$ the roots are complex conjugate and with modulus less than 1. For $\lambda \in [0, \lambda^\star]$, where the roots $z_{1,2}(\lambda)$ are real, $z_1(\lambda)$ and its derivative satisfy

$$z_1(\lambda) = \frac{\omega_1\omega_n}{2\omega_{n+1}} + \frac{\omega_n}{\omega_{n+1}}\frac{\omega_1 - 2}{2}\left[\sqrt{\Delta(\lambda)} - \lambda\right],$$
$$z_1'(\lambda) = \frac{\omega_n}{\omega_{n+1}}\frac{\omega_1 - 2}{2}\left[\frac{\Delta'(\lambda)}{\sqrt{\Delta(\lambda)}} - 1\right], \tag{88}$$
$$\Delta'(\lambda) = 2\lambda - \frac{2\omega_1}{\omega_1 - 2},$$

and thus for $\lambda = 0$ we have $z_1'(0) < 0$. Hence in a neighbourhood of $\lambda = 0, -1 < z_{1,2}(\lambda) < 1$, and for $\lambda \in (0, \lambda^\star]$ there are no roots equal to 1 or 0; thus the roots of (82) are bounded in $(0,1)$. In fact, by contradiction, let $z = 1$ be a root; then by (82)

$$1 + \frac{\omega_n}{\omega_{n+1}}\left(\lambda\left(\omega_1 - 2\right) - \omega_1\right) + \frac{\omega_{n-1}}{\omega_{n+1}} = 0, \qquad \Longrightarrow \lambda = 0. \tag{89}$$

Moreover $z = 0$ is never a root of (82).

Thus the roots are bounded in the interval $(-1, 1)$ for $n \geq 0$ and $\lambda \in (0, 1]$. □

7. Numerical Tests

In this section a group of tests is proposed for the solution of a complex linear system of the form (1); that is, $(\mathbf{B} + i\mathbf{C})(\mathbf{y} + i\mathbf{z}) = \mathbf{c} + i\mathbf{d}$, where **B** and **C** are semi-SPD with **B**+**C** SPD. The solvers used are COCG (Algorithm 2), COCR (Algorithm 3), and the standard Matlab's QMR. The first two are tested together with different preconditioners such as ILU/ILU0 and with the present polynomial preconditioners. As a comparison QMR is also added, with and without preconditioning. When feasible, the direct LU factorization

Table 2: Numerical results for COCG and COCR preconditioned on the basis of the matrix ($\mathbf{B} + \mathbf{C}$) with complete Cholesky decomposition ($\mathbf{LL}^T$), with incomplete Cholesky (IC), and with incomplete Cholesky with threshold 10^{-5} (IC0). The reported numbers represent the iterations of the corresponding solver with the specified preconditioner. The dash indicates that it was not feasible to compute a particular test. The stopping tolerance was 10^{-8}. The column "preconditioner" shows the time required to assemble the preconditioner. The time elapsed in the computation is expressed in seconds.

Test case	Preconditioner			COCG-MHSS			COCR-MHSS		
	$\mathbf{LL}^T$	IC0	IC	$\mathbf{LL}^T$	IC0	IC	$\mathbf{LL}^T$	IC0	IC
T5k									
Iter.				25	148	25	25	148	25
Time	0.21	0.01	0.25	0.51	**0.3**	0.48	0.52	0.32	0.42
T16k									
Iter.				24	78	24	24	75	24
Time	25	0.031	22	34.7	**0.48**	23.4	36	0.64	23.9
T80k									
Iter.				22			22		
Time	5.3	—	—	**11**	—	—	11.2	—	—
T150k									
Iter.					638	32		634	32
Time	—	0.03	3.0	—	11	**5.6**	—	11.2	5.8
T500k									
Iter.									
Time	—	—	—	—	—	—	—	—	—
S13k									
Iter.				5			5		
Time	0.66	—	—	0.88	—	—	**0.85**	—	—
S32k									
Iter.									
Time	—	—	—	—	—	—	—	—	—
S500k									
Iter.									
Time	—	—	—	—	—	—	—	—	—

is used; then two variants are proposed, an incomplete LU and an ILU with threshold. More in detail, Table 1 presents the preconditioners:

(i) LU, the complete LU decomposition, for the matrix $\mathbf{A} = \mathbf{B} + \imath\mathbf{C}$,

(ii) ILU0, the incomplete ILU(0), for the matrix $\mathbf{A} = \mathbf{B} + \imath\mathbf{C}$, the threshold used is 10^{-5},

(iii) ILU, the incomplete ILU(0), for the matrix $\mathbf{A} = \mathbf{B}+\imath\mathbf{C}$,

and the use of the previous preconditioners with COCG, COCR, and QMR:

(i) COCG-ILU0 and COCG-ILU, COCG solver preconditioned with ILU0 and ILU,

(ii) COCR-ILU0 and COCR-ILU, COCR solver preconditioned with ILU0 and ILU,

(iii) QMR, standard QMR implementation of Matlab, with no preconditioning,

(iv) QMR-ILU0 and QMR-ILU, standard QMR implementation of Matlab, with ILU0 and ILU preconditioning.

Table 2 presents the results of COCG and COCR iterative solver with the preconditioner MHSS approximated with

(i) COCG-MHSS-LL^T, COCG-MHSS-IC0, and COCG-MHSS-IC the complete and incomplete Cholesky decompositions for the preconditioner $\mathbf{P}$ defined in (16), threshold used for IC being 10^{-5};

(ii) COCR-MHSS-LL^T, COCR-MHSS-IC0, and COCR-MHSS-IC the complete and incomplete Cholesky decompositions for the preconditioner $\mathbf{P}$ defined in (16), threshold used for IC being 10^{-5}.

Table 3 presents the results of COCG and COCR iterative solver with the preconditioner of Section 5:

(i) COCG-MHSS-J, COCG-MHSS-C the Jacobi and Chebyshev polynomial approximation of MHSS used with COCG iterative method;

(ii) COCR-MHSS-J, COCR-MHSS-C the Jacobi and Chebyshev polynomial approximation of MHSS used with COCR iterative method;

Table 3: Numerical results for COCG and COCR, the reported numbers represents the iterations of the corresponding solver with the specified preconditioner. The dash indicates that it was not feasible to compute a particular test, while the letters "—" mean "not converged" after 10 000 iterations. The value of δ used in the Chebyshev preconditioner was 0.2 while the stopping tolerance was 10^{-8}. The time elapsed in the computation is expressed in seconds.

Test case	Degree of MHSS-J					Degree of MHSS-C				
	10	50	100	500	1000	10	50	100	500	1000
COCG										
T5k										
Iter.	127	32	26	25	25	111	32	32	31	31
Time	0.9	1.0	1.6	7.4	15.0	**0.8**	1.0	1.9	9.2	18.4
T16k										
Iter.	31	24	24	23	23	31	29	28	29	30
Time	**0.6**	1.9	3.6	16.6	32.9	**0.6**	2.3	4.2	21.1	43.2
T80k										
Iter.	4765	657	298	55	31	3904	596	251	52	33
Time	275.6	162.8	145.4	130.4	146.4	225.8	148.0	**121.6**	123.8	156.6
T150k										
Iter.	323	81	42	19	19	277	65	36	23	23
Time	14.2	15.1	15.5	33.9	67.4	12.9	**12.4**	13.3	41.1	81.9
T500k										
Iter.	4928	829	479	91	47	4805	929	461	87	46
Time	1665	**1215**	1374	1378	1393	1763	1333	1343	1246	1380
S13k										
Iter.	4681	1123	591	80	38	4016	1199	613	93	57
Time	29.6	29.1	30.1	**19.8**	23.3	25.6	33.2	32.0	24.1	28.6
S32k										
Iter.	4665	1102	686	106	49	4012	1156	612	102	64
Time	77.1	91.3	96.6	64.8	56.8	**65.1**	77.3	80.6	66.9	81.3
S500k										
Iter.		4328	2165	394	107		4179	2359	390	185
Time	—	5366	5232	4757	2646	—	5250	5715	**4629**	4637
COCR										
T5k										
Iter.	124	32	26	25	25	109	32	32	32	32
Time	0.9	1.0	1.6	7.5	14.8	**0.8**	1.0	1.9	9.5	18.1
T16k										
Iter.	31	24	23	23	23	31	29	28	29	30
Time	**0.6**	1.9	3.6	16.6	32.9	**0.6**	2.2	4.2	21.0	43.2
T80k										
Iter.	3591	533	226	52	32	2947	413	223	49	34
Time	209.5	132.1	109.3	123.4	151.1	172.0	**102.6**	107.7	116.1	160.2
T150k										
Iter.	303	79	42	19	19	267	62	36	23	23
Time	14.2	15.1	15.5	34.0	67.4	13.0	**12.4**	13.3	41.1	81.9
T500k										
Iter.	4066	828	407	80	41	4067	807	397	88	46
Time	1410	1204	1182	1208	1218	1471	**1172**	1174	1330	1357
S13k										
Iter.	3683	896	481	64	38	3153	742	452	93	53
Time	24.7	23.8	24.2	**16.1**	20.5	20.8	20.3	20.6	24.2	27.8
S32k										
Iter.	3904	918	487	76	43	3352	756	408	90	61
Time	64.1	62.6	65.27	**48.1**	58.0	54.6	52.8	56.9	57.2	74.6
S500k										
Iter.	8504	2025	1017	191	75	7203	1634	856	216	102
Time	2567	2525	2473	2386	**1909**	2393	2058	2083	2572	2546

Table 4

Name	Number of rows	NNZ
s1rmq4m1	5 489	262 411
s3rmt3m3	5 357	207 123
Pres_Poisson	14 822	715 804
Dubcova1	16 129	253 009
nasasrb	54 870	2 677 324
apache1	80 800	542 184
denormal	89 400	726 674
G2_circuit	150 102	726 674
pwtk	217 918	11 524 432
parabolic_fem	525 825	3 674 625

Table 5

Test name	Matrix pairing	Number of rows	NNZ
T5k	(s1rmq4m1, s3rmt3m3)	5 489	399 907
T16k	(Pres_Poisson, Dubcova1)	16 129	925 819
T80k	(apache1, nasasrb)	80 800	3 072 500
T150k	(G2_circuit, denormal)	150 102	1 616 970
T500k	(pwtk, parabolic_fem)	525 825	14 810 591

The Incomplete LU decomposition was computed with the standard Matlab command`ilu` with parameters ’`type`’ = ’`crout`’, ’`droptol=1e-5`’; the ILU0 was computed`ilu` with parameters ’`type`’ = ’`nofill`’; for the Cholesky decomposition was used`ichol` with parameters ’`type`’ = ’`ict`’, ’`droptol=1e-5`’; for the IC0 the parameter used was ’`type`’=’`nofill`’. The degrees used for the polynomial preconditioner are 10, 50, 100, 500, and 1000. Due to the lack of complex symmetric matrices with SPD real and imaginary part, it was decided to combine two real SPD matrices of not too far dimension (eventually padding with zeros to match the size of the biggest one). The real SPD matrices used are summarized in Table 4 and can be found on the NIST “Matrix Market” Sparse Matrix Collection [60] or on University of Florida Sparse Matrix Collection [61]. As usual “NNZ” means number of nonzero elements and it is understood that the matrices are square; hence only the number of rows is reported.

These matrices are used paired where the first matrix of the pair corresponds to the real part and the second to the imaginary part. If the dimensions disagree, the smallest matrix is padded with zero rows and columns up to the size of the biggest one. The pairing of the matrices with the name of the corresponding test is resumed in Table 5.

The right hand side used for all tests, unless explicitly written, is assumed to be $(1+\imath)\mathbb{1}$, that is, $1+\imath$ for all components. A test from a real application is found in [2], from which the complex symmetric SPD matrices are provided with a specific right hand side. Size and name of the matrices are resumed in Table 6.

Table 6

Test name	Matrix name	Number of rows	NNZ
S13k	eddy_6k_gauged	13 067	295 571
S32k	eddy_16k_gauged	31 853	720 689
S500k	eddy_265k_gauged	538 709	11 690 125

List of the tests is as follows: for the first group (T tests) the right hand side was $(1+\imath)\mathbb{1}$; for the second group (S tests) the right hand side was the one prescribed in the paper [2].

7.1. Analysis of the Experiments. From the comparison of Tables 1, 2, and 3 it is clear that there is no preconditioner that is always better than the others. The optimal preconditioner is in general problem dependent; therefore, for general preconditioners, a good quality measure is the robustness over different problems. Table 3 shows that the strategy presented in Section 2 is effective. In fact, when the ILU factorization is available and the iterations converge, incomplete factorization preconditioner is faster than polynomial preconditioner, but the proposed polynomial preconditioner is an effective alternative when ILU is not available or when it is not sufficient as preconditioner. This is true also for the relatively small matrices with 5000 rows, where the number of iterations of the ILU-based methods is comparable with the polynomial preconditioners of low degree, for example, 10. Rising the degree of the polynomial corresponds to lowering the number of iterations needed by COCG/COCR. It is also apparent that it is not possible to go below a certain number of iterations even with a very high degree polynomial; this is evident, for example, in test T16k of Table 3 with both COCG and COCR and with both preconditioners. This is explained from the fact that the condition number of the preconditioned system is independent of the system size. Another behaviour that is common to most of the tests is the better performance of the COCR over the COCG: this can be appreciated looking at Figure 2 and Figure 3. They show the history of the residual for each iteration of both methods with the Jacobi and the Chebyshev preconditioners. In both cases the decrease of the residual is always slower in the COCG than in the COCR; also the number of iterations is higher.

8. Conclusions

A polynomial preconditioner was presented for the solution of the linear system $\mathbf{A}\mathbf{x} = \mathbf{b}$, for $\mathbf{A}$ complex symmetric, that is, such that $\mathbf{A} = \mathbf{B} + \imath\mathbf{C}$, where $\mathbf{B}$, $\mathbf{C}$ are real symmetric positive semidefinite matrices (semi-SPD) and $\mathbf{B}+\mathbf{C}$ is symmetric positive definite (SPD). Typical problems of this form come from the field of electrodynamics, where the involved matrices are complex but not Hermitian and standard methods cannot be used directly. This algorithm is suitable for large matrices, where Cholesky decomposition, or its inexact form, are too costly or infeasible. It works as a polynomial approximation of a single step of the MHSS method, but it is successfully applied as preconditioner of

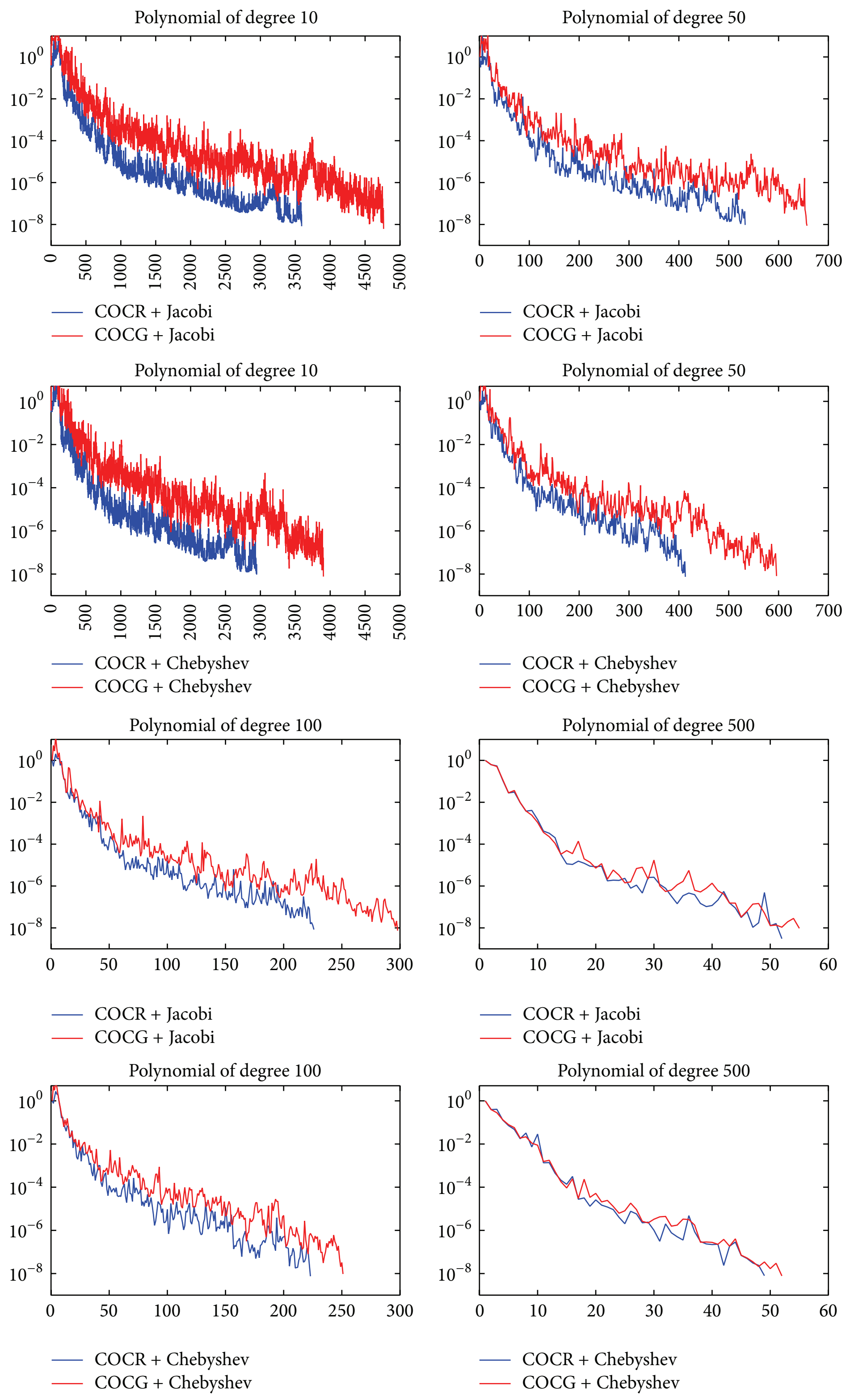

Figure 2: The history of the residual for the test T80k with COCR and COCG preconditioned with Jacobi and Chebyshev polynomial.

conjugate gradient-like methods; in particular it is showed how to use it together with COCG or COCR. Following the trend of the last years, but being aware of the criticism that arose in the 80s, the proposed new preconditioner is computed as a recurrence of orthogonal polynomials and is proved to be stable. This allows employing polynomials of very high degree and numerical tests confirm the expected theoretical good performances.

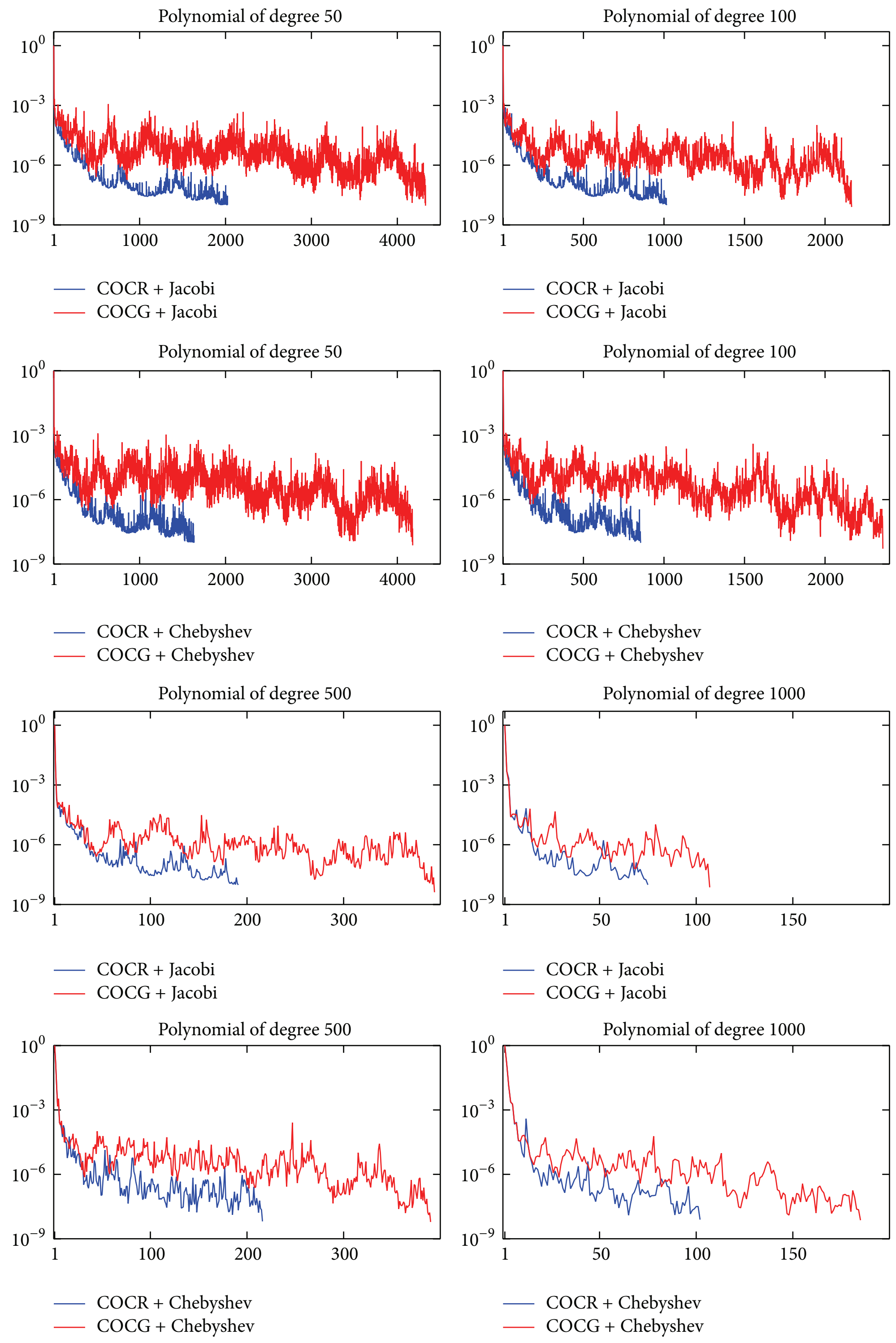

Figure 3: The history of the residual for the test S500k with COCR and COCG preconditioned with Jacobi and Chebyshev polynomial.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

The authors wish to thank Professor R. Specogna for providing the matrices used for the Eddy Current test problems

S13k, S32k, and S500k. They are also grateful for the positive and constructive feedback in testing the presented preconditioners on real problems.

References

[1] U. Van Rienen, *Numerical Methods in Computational Electrodynamics: Linear Systems in Practical Applications*, Lecture Notes in Computational Science and Engineering, Springer, Berlin, Germany, 2001.

[2] L. Codecasa, R. Specogna, and F. Trevisan, "Base functions and discrete constitutive relations for staggered polyhedral grids," *Computer Methods in Applied Mechanics and Engineering*, vol. 198, no. 9–12, pp. 1117–1123, 2009.

[3] A. Pirani, M. Ricci, R. Specogna, A. Tamburrino, and F. Trevisan, "Multi-frequency identification of defects in conducting media," *Inverse Problems*, vol. 24, no. 3, Article ID 035011, 2008.

[4] H. A. van der Vorst and J. B. M. Melissen, "Petrov-Galerkin type method for solving Ax = b, where A is symmetric complex," *IEEE Transactions on Magnetics*, vol. 26, no. 2, pp. 706–708, 1990.

[5] T. Sogabe and S.-L. Zhang, "A COCR method for solving complex symmetric linear systems," *Journal of Computational and Applied Mathematics*, vol. 199, no. 2, pp. 297–303, 2007.

[6] A. Bunse-Gerstner and R. Stöver, "On a conjugate gradient-type method for solving complex symmetric linear systems," *Linear Algebra and Its Applications*, vol. 287, no. 1–3, pp. 105–123, 1999.

[7] H. Sadok, "CMRH: a new method for solving nonsymmetric linear systems based on the Hessenberg reduction algorithm," *Numerical Algorithms*, vol. 20, no. 4, pp. 303–321, 1999.

[8] H. van der Vorst, "Bi-CGSTAB: a fast and smoothly converging variant of Bi-CG f or the solution of nonsymmetric linear systems," *SIAM Journal on Scienti fi c and Statistical Computing*, vol. 13, no. 2, pp. 631–644, 1992.

[9] G. L. G. Sleijpen and H. A. van der Vorst, "Maintaining convergence properties of BiCGstab methods in finite precision arithmetic," *Numerical Algorithms*, vol. 10, no. 2, pp. 203–223, 1995.

[10] G. L. G. Sleijpen and D. R. Fokkema, "BiCGstab(ell) for linear equations involving unsymmetric matrices with complex spectrum," *Electronic Transactions on Numerical Analysis*, vol. 1, pp. 11–32, 1993.

[11] Y. Saad and M. Schultz, "GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems," *SIAM Journal on Scientific and Statistical Computing*, vol. 7, no. 3, pp. 856–869, 1986.

[12] R. W. Freund and N. M. Nachtigal, "QMR: a quasi-minimal residual method for non-Hermitian linear systems," *Numerische Mathematik*, vol. 60, no. 1, pp. 315–339, 1991.

[13] K. Abe and G. L. G. Sleijpen, "BiCR variants of the hybrid BiCG methods for solving linear systems with nonsymmetric matrices," *Journal of Computational and Applied Mathematics*, vol. 234, no. 4, pp. 985–994, 2010.

[14] Y.-F. Jing, T.-Z. Huang, Y. Zhang et al., "Lanczos-type variants of the COCR method for complex nonsymmetric linear systems," *Journal of Computational Physics*, vol. 228, no. 17, pp. 6376–6394, 2009.

[15] Z.-Z. Bai, M. Benzi, and F. Chen, "Modified HSS iteration methods for a class of complex symmetric linear systems," *Computing*, vol. 87, no. 3-4, pp. 93–111, 2010.

[16] Z.-Z. Bai, M. Benzi, and F. Chen, "On preconditioned MHSS iteration methods for complex symmetric linear systems," *Numerical Algorithms*, vol. 56, no. 2, pp. 297–317, 2011.

[17] Z.-Z. Bai, G. H. Golub, and M. K. Ng, "Hermitian and skew-Hermitian splitting methods for non-Hermitian positive definite linear systems," *SIAM Journal on Matrix Analysis and Applications*, vol. 24, no. 3, pp. 603–626, 2003.

[18] Z.-Z. Bai, G. H. Golub, and M. K. Ng, "On inexact hermitian and skew-Hermitian splitting methods for non-Hermitian positive definite linear systems," *Linear Algebra and Its Applications*, vol. 428, no. 2-3, pp. 413–440, 2008.

[19] O. Axelsson, "A survey of preconditioned iterative methods for linear systems of algebraic equations," *BIT Numerical Mathematics*, vol. 25, no. 1, pp. 165–187, 1985.

[20] M. Benzi, "Preconditioning techniques for large linear systems: a survey," *Journal of Computational Physics*, vol. 182, no. 2, pp. 418–477, 2002.

[21] "Minimax polynomial preconditioning for Hermitian linear systems," *SIAM Journal on Matrix Analysis and Applications*, vol. 12, pp. 766–789, 1991.

[22] S. F. Ashby, T. A. Manteuffel, and J. S. Otto, "A comparison of adaptive Chebyshev and least squares polynomial preconditioning for Hermitian positive definite linear systems," *SIAM Journal on Scientific and Statistical Computing*, vol. 13, pp. 1–29, 1992.

[23] O. Johnson, C. Micchelli, and G. Paul, "Polynomial preconditioners for conjugate gradient calculations," *SIAM Journal on Numerical Analysis*, vol. 20, pp. 362–376, 1983.

[24] Y. Saad, "Practical use of polynomial preconditionings for the conjugate gradient method," *SIAM Journal on Scientific and Statistical Computing*, vol. 6, no. 4, pp. 865–881, 1985.

[25] B. Fischer, *Polynomial Based Iteration Methods for Symmetric Linear Systems*, Classics in Applied Mathematics, Society for Industrial and Applied Mathematics, 2011.

[26] L. Lu, J. Lai, and S. Xu, "A polynomial preconditioner for the CMRH algorithm," *Mathematical Problems in Engineering*, vol. 2011, Article ID 545470, 12 pages, 2011.

[27] I. Gelfand, "Normierte Ringe," *Recueil Mathématique. Matematicheskiĭ Sbornik (Nouvelle Serie)*, vol. 9, no. 51, pp. 3–24, 1941.

[28] V. Kozyakin, "On accuracy of approximation of the spectral radius by the Gelfand formula," *Linear Algebra and Its Applications*, vol. 431, no. 11, pp. 2134–2141, 2009.

[29] A. Householder, *The Theory of Matrices in Numerical Analysis*, Blaisdell Book in the Pure and Applied Sciences, Blaisdell Publishing Company, 1965.

[30] J. Stoer and R. Bulirsch, *Introduction to Numerical Analysis*, Texts in Applied Mathematics, Springer, New York, NY, USA, 2002.

[31] O. Axelsson, "Iteration number for the conjugate gradient method," *Mathematics and Computers in Simulation*, vol. 61, no. 3–6, pp. 421–435, 2003.

[32] T. Sogabe, M. Sugihara, and S.-L. Zhang, "An extension of the conjugate residual method to nonsymmetric linear systems," *Journal of Computational and Applied Mathematics*, vol. 226, no. 1, pp. 103–113, 2009.

[33] G. Golub and C. Van Loan, *Matrix Computations*, Johns Hopkins Studies in the Mathematical Sciences, Johns Hopkins University Press, Baltimore, Md, USA, 1996.

[34] Society for Industrial and Applied Mathematics, *Iterative Methods for Sparse Linear Systems*, Society for Industrial and Applied Mathematics, Philadelphia, Pa, USA, 2nd edition, 2003.

[35] "Conjugate gradient-type methods for linear systems with complex symmetric coeficient matrices," *SIAM Journal on Scientific and Statistical Computing*, vol. 13, pp. 425–448, 1992.

[36] I. Duff, A. Erisman, and J. Reid, *Direct Methods for Sparse Matrices*, Monographs on Numerical Analysis, Clarendon Press, 1986.

[37] A. George and J. Liu, *Computer Solution of Large Sparse Positive Definite Systems*, Prentice-Hall Series in Computational Mathematics, Prentice-Hall, 1981.

[38] T. A. Davis, *Direct Methods for Sparse Linear Systems (Fundamentals of Algorithms 2)*, Society for Industrial and Applied Mathematics, Philadelphia, Pa, USA, 2006.

[39] L. Bergamaschi, G. Pini, and F. Sartoretto, "Approximate inverse preconditioning in the parallel solution of sparse eigenproblems," *Numerical Linear Algebra with Applications*, vol. 7, no. 3, pp. 99–116, 2000.

[40] P. F. Dubois, A. Greenbaum, and G. H. Rodrigue, "Approximating the inverse of a matrix for use in iterative algorithms on vector processors," *Computing*, vol. 22, no. 3, pp. 257–268, 1979.

[41] T. Dupont, R. Kendall, and H. Rachford Jr., "An approximate factorization procedure for solving self-adjoint elliptic difierence equations," *SIAM Journal on Numerical Analysis*, vol. 5, pp. 559–573, 1968.

[42] C.-J. Lin and J. J. Moré, "Incomplete Cholesky factorizations with limited memory," *SIAM Journal on Scientific Computing*, vol. 21, no. 1, pp. 24–45, 1999.

[43] Y. Saad, "ILUM: a multi-elimination ILU preconditioner for general sparse matrices," *SIAM Journal on Scientific Computing*, vol. 17, no. 4, pp. 830–847, 1996.

[44] A. Gupta and T. George, "Adaptive techniques for improving the performance of incomplete factorization preconditioning," *SIAM Journal on Scientific Computing*, vol. 32, no. 1, pp. 84–110, 2010.

[45] C. Janna and M. Ferronato, "Adaptive pattern research for block fsai preconditioning," *SIAM Journal on Scientific Computing*, vol. 33, no. 6, pp. 3357–3380, 2011.

[46] J. Scott and M. Tůma, "On positive semidefinite modification schemes for incomplete cholesky factorization," *SIAM Journal on Scientific Computing*, vol. 36, no. 2, pp. A609–A633, 2014.

[47] M. Benzi, C. D. Meyer, and M. Tůma, "A sparse approximate inverse preconditioner for the conjugate gradient method," *SIAM Journal on Scientific Computing*, vol. 17, no. 5, pp. 1135–1149, 1996.

[48] E. Chow, "Parallel implementation and practical use of sparse approximate inverse preconditioners with a priori sparsity patterns," *International Journal of High Performance Computing Applications*, vol. 15, no. 1, pp. 56–74, 2001, cited By (since 1996)36.

[49] A. Rafiei, "Left-looking version of AINV preconditioner with complete pivoting strategy," *Linear Algebra and Its Applications*, vol. 445, pp. 103–126, 2014.

[50] D. K. Salkuyeh and F. Toutounian, "A sparse-sparse iteration for computing a sparse incomplete factorization of the inverse of an SPD matrix," *Applied Numerical Mathematics*, vol. 59, no. 6, pp. 1265–1273, 2009.

[51] J. Boyle, M. Mihajlović, and J. Scott, "HSL_MI20: an efficient AMG preconditioner for finite element problems in 3D," *International Journal for Numerical Methods in Engineering*, vol. 82, no. 1, pp. 64–98, 2010.

[52] A. Napov and Y. Notay, "An algebraic multigrid method with guaranteed convergence rate," *SIAM Journal on Scientific Computing*, vol. 34, no. 2, pp. A1079–A1109, 2012.

[53] K. Stuben, "An introduction to algebraic multigrid," in *Multigrid*, U. Trottenberg, C. Oosterlee, and A. Schuller, Eds., pp. 413–532, Academic Press, New York, NY, USA, 2001.

[54] R. Freund, "On conjugate gradient type methods and polynomial preconditioners for a class of complex non-hermitian matrices," *Numerische Mathematik*, vol. 57, no. 1, pp. 285–312, 1990.

[55] E. Stiefel, *Kernel Polynomials in Linear Algebra and Their Numerical Applications*, ETH, Zürich, Switzerland, 1958.

[56] S. F. Ashby, *Polynomial Preconditioning for Conjugate Gradient Methods*, DOE/ER, Department of Computer Science, University of Illinois at Urbana-Champaign, 1988.

[57] G. Szego, *Orthogonal Polynomials*, American Mathematical Society, 1939.

[58] R. Jungers, *The Joint Spectral Radius: Theory and Applications*, Lecture Notes in Control and Information Sciences, Springer, New York, NY, USA, 2009.

[59] G.-C. Rota and G. Strang, "A note on the joint spectral radius," *Indagationes Mathematicae*, vol. 22, pp. 379–381, 1960.

[60] R. F. Boisvert, R. Pozo, K. Remington, R. F. Barrett, and J. J. Dongarra, "Matrix market: a web resource for test matrix collections," in *Proceedings of the IFIP TC2/WG2.5 Working Conference on Quality of Numerical Software: Assessment and Enhancement*, pp. 125–137, Chapman & Hall, London, UK, 1997.

[61] T. A. Davis and Y. Hu, "The University of Florida sparse matrix collection," *ACM Transactions on Mathematical Software*, vol. 38, no. 1, article 1, 2011.