NUMERICAL LINEAR ALGEBRA WITH APPLICATIONS
Numer. Linear Algebra Appl. 2000; **7**:197–218

Real valued iterative methods for solving complex symmetric linear systems

Owe Axelsson[1,*,†] and Andrey Kucherov[2,‡]

[1]*Department of Mathematics, University of Nijmegen, Toernooiveld 1, 6525 ED Nijmegen, The Netherlands.*
[2]*Department of Computational Mathematics and Cybernetics, Moscow State University, 119899, Moscow, Russia*

SUMMARY

Complex valued systems of equations with a matrix $R + \imath S$ where R and S are real valued arise in many applications. A preconditioned iterative solution method is presented when R and S are symmetric positive semi-definite and at least one of R, S is positive definite. The condition number of the preconditioned matrix is bounded above by 2, so only very few iterations are required. Applications when solving matrix polynomial equation systems, linear systems of ordinary differential equations, and using time-stepping integration schemes based on Padé approximation for parabolic and hyperbolic problems are also discussed. Numerical comparisons show that the proposed real valued method is much faster than the iterative complex symmetric QMR method.

KEY WORDS: complex symmetric matrices; matrix polynomial equations; preconditioned iterative methods

1. INTRODUCTION

Complex valued problems arise in many applications. A common case is the solution of a matrix polynomial equation

$$Q_m(A)x = b \tag{1}$$

where A is a real square matrix and Q_m is a polynomial of degree m that has no zeroes at the eigenvalues of A. With no limitation we may assume that the leading coefficient of Q_m is $(-1)^m$. Letting z_k be the zeroes of Q_m we find

$$\prod_{k=1}^{m}(A - z_k I)x = b$$

* Correspondence to: Owe Axelsson, Department of Mathematics, University of Nijmegen, Toernooiveld 1, 6525 ED Nijmegen, The Netherlands.
† E-mail: axelsson@sci.kun.nl
‡ E-mail: kucherov@cmc.msk.su

Received 31 December 1998
Revised 15 January 2000

or assuming that $z_k \neq z_l$, $k \neq l$,

$$x = \sum_{k=1}^{m} a_k (A - z_k I)^{-1} b$$

where $a_k = [Q'_m(z_k)]^{-1}$. Hence the solution of Equation (1) can be computed by solving m systems

$$(A - z_k I) y^{(k)} = b, \quad k = 1, 2, \ldots, m \tag{2}$$

in parallel and then forming $x = \sum a_k y^{(k)}$. In general, z_k are complex numbers. The systems in Equation (2) have a special form. Consider more generally

$$Au = b$$

or

$$(R + \imath S)(x + \imath y) = \phi + \imath\psi \tag{3}$$

where R, S are real matrices of order N, x, y, ϕ, $\psi \in \mathbb{R}^N$, and $\imath$ is the imaginary unit. Normally R and S are large but sparse matrices, which is why such systems are traditionally solved by iterations.

There are two general approaches to solving complex systems. The first is to solve the system in a complex arithmetic. However, it turns out to be more difficult to precondition complex valued systems as the eigenvalues can be spread in the whole complex plane and the iterative solution method used may then converge too slowly. One can also apply a preconditioned conjugate gradient (CG) method to the Hermitian positive definite normal system $A^{\mathrm{H}}Au = A^{\mathrm{H}}b$ for which the eigenvalues are real. In any case these involves complex arithmetic that costs typically three to four times as much as the corresponding real arithmetic.

It is possible to avoid complex arithmetic by rewriting Equation (3) in real valued form. This can be done in several ways, but the following two are basic:

$$\left.\begin{aligned} A_* \begin{pmatrix} x \\ y \end{pmatrix} &\equiv \begin{pmatrix} R & -S \\ S & R \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} \phi \\ \psi \end{pmatrix} \\ \text{and} \quad A_{**} \begin{pmatrix} x \\ -y \end{pmatrix} &\equiv \begin{pmatrix} R & S \\ S & -R \end{pmatrix} \begin{pmatrix} x \\ -y \end{pmatrix} = \begin{pmatrix} \phi \\ \psi \end{pmatrix} \end{aligned}\right\} \tag{4}$$

The second approach is then to solve these real and, in general, non-symmetric and/or indefinite systems by one of the generalized conjugate gradient methods such as GMRES [1] or GCG [2–4].

As is well-known for any real matrix, such as A_* it holds that complex eigenvalues λ appears in complex conjugate pairs $\lambda, \bar{\lambda}$. Further, it is readily seen that for any eigenvalue $\lambda \neq 0$ of the real symmetric matrix A_{**}, $-\lambda$ is also an eigenvalue. Thus the spectrum $\sigma(A_*)$ is symmetric with respect to the real axes and the spectrum $\sigma(A_{**})$ is symmetric with respect to the imaginary axes, i.e. in both cases the spectrum embraces the origin. From best polynomial approximation properties it is known that this situation leads to the polynomials of essentially a square degree for the same approximation accuracy compared to the case with a one-sided spectrum.

We refer the reader to Reference [5] and the references cited therein for more general results and theoretical arguments (see especially Section 5 and Theorem 5.4 in Reference [5]) and numerical results,

explaining why Krylov subspace methods are inefficient for solving complex systems, represented in the basic real forms.

In this paper we shall consider a class of linear systems with complex symmetric matrix $A = A^{\mathrm{T}}$, or equivalently, real symmetric matrices $R = R^{\mathrm{T}},\ S = S^{\mathrm{T}}$, and a special subclass of such systems, so-called shifted complex symmetric systems, whose matrices have the form

$$A = R + \imath\omega I \tag{5}$$

where the 'shift' parameter ω is a real scalar. The example of solving matrix polynomial equations, mentioned above, generates a set of shifted linear systems with different shifts ω_k but a single right hand side b.

On the other hand some iterative algorithms, such as CSYM [6], the complex symmetric QMR method [5], and shifting techniques have been proposed (for some other methods, see References [7–9]. These algorithms take into account the structure of complex symmetry and posses some remarkable properties. The method CSYM is based on a unitary transformation of A to complex symmetric tridiagonal form and it is proven that CSYM has both the minimal residual property and constant costs (i.e. a fixed number of recursion vectors) per iteration step [6]. The QMR method is based on a variant of the non-symmetric Lanczos algorithm, and generates, with constant costs per step, iterates satisfying a quasi-minimal residual property.

We refer the reader to Reference [6] for numerical comparisons of CSYM, QMR and CGNE, which indicate that QMR wins on complex tests like Helmholtz problems, while CSYM is better on some band matrices, and both methods always outperform the CGNE method.

A technique of shifted Krylov subspaces utilizes the fact that Krylov subspace $K_j(R + \imath\omega_k I, r_0)$ induced by the shifted matrix $R + \imath\omega_k I$ and the same initial residual r_0 do not depend on the shifts ω_k. Thus one can take only one system from the shifted family, run the Lanczos process for this system, and on each jth Lanczos step obtain the iterates for all other systems, using relatively simple and cheap modifying formulae. For details see References [10,5], also References [11,12] in the case of symmetric positive definite systems, and References [13,14] for the application of GMRES to the solution of shifted systems with general matrices. Here, we just note that QMR can be implemented in such a way that the number of matrix–vector and inner products per iteration do not depend on the number of shifted systems, i.e. are the same as for a single system. Besides, if some starting data of the Lanczos process are real then all other quantities involved in the Lanczos iteration part, including Lanczos vectors and scalar products, remain real. Therefore, the most expensive part of the algorithm can be implemented in real arithmetic, and only for the cheaper modifying formulae is complex arithmetic required.

However such impressive results as cited above do not really mean that the QMR or CSYM method provide a reliable and fast solution tool for any complex symmetric system. It should be noted that the complex symmetric QMR and CSYM were originally proposed without preconditioning. In practice unpreconditioned methods converge very slowly. How to choose preconditioners for a complex system that are as efficient and robust as preconditioners for positive definite symmetric systems remains an open problem. Besides, the intention to keep for the QMR or shifted Krylov approach a part of computation in real arithmetic prevents the use of any incomplete factorization preconditioner, since the system, preconditioned in such a way, will no longer be a shifted one [15]. Furthermore, many problems can involve other, non-diagonal, shifts.

In this paper we reduce a complex system $(R + \imath S)(x + \imath y) = \phi + \imath\psi$ to the following real valued form

$$\begin{pmatrix} R - \alpha S & \sqrt{1+\alpha^2}S \\ \sqrt{1+\alpha^2}S & -R - \alpha S \end{pmatrix} \begin{pmatrix} x \\ z \end{pmatrix} = \begin{pmatrix} \phi \\ \chi \end{pmatrix} \tag{6}$$

where α is a real parameter ($\alpha \neq 0$), $z, \chi \in \mathbb{R}^N$ are real vectors connected with the solution and the right-hand side of a given system via simple formulae, and we always suppose that $R + \alpha S$ is non-singular ($det(R + \alpha S) \neq 0$).

Instead of solving a full block system (6), we apply a Schur complement approach by the elimination of one component, which results in the following reduced system:

$$\left.\begin{aligned} C_\alpha x &= f \\ C_\alpha &= R - \alpha S + (1+\alpha^2)S(R+\alpha S)^{-1}S \end{aligned}\right\} \tag{7}$$

with corresponding right-hand side f.

We shall assume that one of the matrices R and S (in our notation it will be R) is symmetric and positive definite (s.p.d.), while the other, S, is symmetric and positive semi-definite (s.p.s-d.):

$$R = R^{\mathrm{T}} > 0, \qquad S = S^{\mathrm{T}} \geq 0 \tag{8}$$

We prove under the stated assumptions that the matrix C_α of system (7) is symmetric and positive definite.

Then we show that the matrix

$$B_\alpha = R + \alpha S$$

serves as an excellent preconditioner for the reduced system (7). In particular we shall obtain upper and lower bounds on eigenvalues of the preconditioned matrix $M_\alpha = B_\alpha^{-1} C_\alpha$ and choose an optimal value $\alpha_{\mathrm{opt}} = \arg\min \mathrm{Cond}(M_\alpha)$ of the parameter α. We prove that

$$\alpha_{\mathrm{opt}} = \hat{\alpha} = \frac{\hat{\lambda}}{1 + \sqrt{1 + \hat{\lambda}^2}}, \quad \mathrm{Cond}(M_\alpha) = 1 + \hat{\alpha}^2$$

where $\hat{\lambda}$ is an upper bound on eigenvalues of $R^{-1}S$. If the value of $\hat{\lambda}$ is not known, one can take $\alpha_{\mathrm{opt}} = \hat{\alpha} = 1$ and get $\mathrm{Cond}(M_\alpha) = 2$. Hence, in any case

$$\mathrm{Cond}(M_\alpha) \leq 2$$

and the reduced real system (7) can be solved efficiently by the conjugate gradient method or the Chebyshev iteration method, preconditioned by matrix B_α. The number of iterations n_{iters}, required to reduce the initial error by $1/\epsilon$ times, $0 < \epsilon < 1$, do not exceed the value

$$n_{\mathrm{iters}} \leq \left\lceil \frac{1}{2\ln(1+\sqrt{2})} \ln \frac{2}{\epsilon} \right\rceil \simeq 0.5673 \ln \frac{2}{\epsilon}$$

i.e. only very few iterations are required.

We note that the real valued block form (6) plays only an intermediate role in our approach, and the

unsatisfactory spectral properties of the real block matrices A_* or A_{**} do not occur in our method. The first works in this direction appeared in References [16,17]; the system in the form (6) was considered first in Reference [18].

We describe in detail and prove the mentioned results in the first part of Section 2. The remainder of the paper is organized as follows. In the second part of Section 2 we describe a generalization of the proposed transformation, in which two parameters are involved; and describe in detail an algorithm for solving matrix polynomial equations. This algorithm will be used in Section 3, devoted to applications of our approach for differential equations. We consider the solution of a system of ODEs with periodic source function, which leads to a family of shifted complex systems and the solution of multidimensional parabolic and hyperbolic problems with general source functions, by using Padé approximation type time-stepping schemes.

A key point in the applicability of our RV method is the question of how to solve the equations with the matrix $R + \alpha S$ and how costly these solutions are. Of course, this is a problem dependent issue. These inner linear systems can be solved by iterations; since they are real-valued, several known preconditioners may be used in the inner iterations. In the final section, Section 4, we shall discuss some algorithmic details of our method and give the results of a numerical comparison with the complex symmetric QMR method on 2-D mesh problems with 100×100 to 500×500 grid-points. Numerical comparisons show that the proposed method, utilizing a sparse direct solver with minimal-degree ordering, is 6–150 or more times faster than the complex symmetric QMR method. One of our methods has been used to solve electrodynamic problems; see Reference [19].

2. EFFICIENT PRECONDITIONING METHODS FOR COMPLEX SYSTEMS REDUCED TO REAL VALUED FORM

At first, we shall derive the real valued system of the forms (6), (7). To this end we introduce a real parameter α and the following matrix decompositions

$$\begin{pmatrix} R & S \\ S & -R \end{pmatrix} = \begin{pmatrix} I & 0 \\ \alpha I & \sqrt{1+\alpha^2}I \end{pmatrix} \begin{pmatrix} R-\alpha S & \sqrt{1+\alpha^2}S \\ \sqrt{1+\alpha^2}S & -R-\alpha S \end{pmatrix} \begin{pmatrix} I & 0 \\ \frac{\alpha}{\sqrt{\alpha^2+1}}I & \frac{1}{\sqrt{\alpha^2+1}}I \end{pmatrix} \tag{9}$$

and

$$\begin{pmatrix} R-\alpha S & \sqrt{\alpha^2+1}S \\ \sqrt{\alpha^2+1}S & -R-\alpha S \end{pmatrix} = \begin{pmatrix} I & -\sqrt{1+\alpha^2}S(R+\alpha S)^{-1} \\ 0 & I \end{pmatrix} \begin{pmatrix} C_\alpha & 0 \\ \sqrt{1+\alpha^2}S & -R-\alpha S \end{pmatrix} \tag{10}$$

where

$$C_\alpha = R - \alpha S + \sqrt{\alpha^2+1}S(R+\alpha S)^{-1}S \tag{11}$$

is a Schur complement matrix.

Using these decompositions we transform the basic system (4) and consequently obtain

$$\begin{pmatrix} R-\alpha S & \sqrt{1+\alpha^2}S \\ \sqrt{1+\alpha^2}S & -R-\alpha S \end{pmatrix} \begin{pmatrix} x \\ \frac{\alpha x - y}{\sqrt{1+\alpha^2}} \end{pmatrix} = \begin{pmatrix} \phi \\ \frac{\psi - \alpha\phi}{\sqrt{1+\alpha^2}} \end{pmatrix} \tag{12}$$

and

$$\begin{pmatrix} C_\alpha & 0 \\ \sqrt{1+\alpha^2}S & -R-\alpha S \end{pmatrix} \begin{pmatrix} x \\ \dfrac{\alpha x - y}{\sqrt{1+\alpha^2}} \end{pmatrix} = \begin{pmatrix} f \\ \dfrac{\psi - \alpha\phi}{\sqrt{1+\alpha^2}} \end{pmatrix} \tag{13}$$

where

$$f = \phi + S(R+\alpha S)^{-1}(\psi - \alpha\phi)$$

Summarizing, we obtain the real valued solution algorithm.

Algorithm 2.1. Real Valued algorithm for solving complex symmetric system

Step 1. **Compute** $f = \phi + S(R+\alpha S)^{-1}(\psi - \alpha\phi)$
Step 2. **Solve** $C_\alpha x = f$
Step 3. **Solve** $(R+\alpha S)z = \alpha\phi - \psi + (1+\alpha^2)Sx$
Step 4. **Compute** $y = \alpha x - z$

The solution of equation $C_\alpha x = f$ is the most costly step of the algorithm. Evidently, under assumptions (8) matrix C_α is symmetric. Let $B_\alpha = R + \alpha S$ be a preconditioner to C_α. To analyse the condition number of the corresponding preconditioned matrix

$$M_\alpha = B_\alpha^{-1} C_\alpha \tag{14}$$

we let $H = R^{-1/2} S R^{-1/2}$ and consider the generalized eigenvalue problem $\mu B_\alpha z = C_\alpha z$ or,

$$\mu(R+\alpha S)z = (R - \alpha S + \sqrt{\alpha^2+1}S(R+\alpha S)^{-1}S)z$$

where μ is the eigenvalue corresponding to an eigenvector z. Equivalently,

$$\mu(I+\alpha H)y = (I - \alpha H + (\alpha^2+1)H(I+\alpha H)^{-1}H)y \tag{15}$$

where $y = R^{1/2}z$.

If $\lambda R z = S z$, $z \neq 0$, or, equivalently, $Hy = \lambda y$, $y \neq 0$, then (2.8) shows that

$$\mu = \frac{1 - \alpha\lambda + (1+\alpha^2)\lambda^2/(1+\alpha\lambda)}{1+\alpha\lambda}$$

or

$$\mu = \frac{1+\lambda^2}{(1+\alpha\lambda)^2} \tag{16}$$

It has thus been shown that matrix C_α is symmetric and positive definite. Hence, to solve an equation $C_\alpha x = f$ we can apply a preconditioned conjugate gradient method. The choice of the optimal value of the parameter α is considered in the following subsection.

Remark 1
There exists a slightly more efficient form of the iteration matrix that also shows that we can weaken the assumptions made on R and S. Hence, consider

$$\left.\begin{aligned} Rx - Sy &= \phi \\ Sx + Ry &= \psi \end{aligned}\right\} \tag{17}$$

and assume that α is a real parameter such that $R + \alpha S$ is non-singular. Such a parameter exists if $\ker(R) \cap \ker(S) = \emptyset$.

Multiplying the first equation by $-\alpha(R + \alpha S)^{-1}$, the second by $(R + \alpha S)^{-1}$ and adding yields

$$(R + \alpha S)^{-1}(S - \alpha R)x + y = (R + \alpha S)^{-1}(\psi - \alpha\phi) \tag{18}$$

Now multiplying this equation by S, using $Sy = Rx - \phi$, and rewriting the equation properly we find

$$r \equiv Rx - \phi + S(R + \alpha S)^{-1}((S - \alpha R)x + \alpha\phi - \psi) = 0 \tag{19}$$

When solving the system by iteration, such as by Chebyshev iterations r will be the residual and we observe that r can be written in the form (see Equation (18)

$$\begin{aligned} y &= (R + \alpha S)^{-1}((S - \alpha R)x + \alpha\phi - \psi) \\ r &= Rx - Sy - \phi \end{aligned}$$

In this form there is no need to compute the right hand side vector f initially as is done in algorithm (9) and the vector y is found during the iteration process. This saves two solutions with the matrix $R + \alpha S$. Since we need few iterations, such a saving can be important to decrease the total expense of the method.

To solve Equation (19) we use $R + \alpha S$ as a preconditioner. The resulting preconditioned matrix takes the form

$$\begin{aligned} E_\alpha &= (R + \alpha S)^{-1}[R + S(R + \alpha S)^{-1}(S - \alpha R)] \\ &= (R + \alpha S)^{-1}[(R + \alpha S) - \alpha S](R + \alpha S)^{-1}R + (R + \alpha S)^{-1}S(R + \alpha S)^{-1}S \\ &= ((R + \alpha S)^{-1}R)^2 + ((R + \alpha S)^{-1}S)^2 \end{aligned}$$

This form can also be used to derive eigenvalue estimates in more general cases than was done above. If R is non-singular we find

$$E_\alpha = (I + \alpha R^{-1}S)^{-2}(I + (R^{-1}S)^2)$$

Therefore, the preconditioned matrix is a rational function in the matrix $R^{-1}S$. It follows that the eigenvalues μ of E_α satisfy

$$\mu = \frac{1 + \lambda^2}{(1 + \alpha\lambda)^2}$$

where λ is an eigenvalue of $R^{-1}S$. This generalizes the relation (16) to arbitrary matrices where R is non-singular.

A similar method was considered in References [16,17]. There, the Schur complement system with

$R + SR^{-1}S$ to Equation (17) was preconditioned by $(R + \alpha S)R^{-1}(R + \alpha S)$.

2.1. Condition number estimate

We want to choose α to minimize the spectral condition number

$$\mathrm{Cond}(M_\alpha) = \mu_{\max}/\mu_{\min}$$

.

Theorem 2.1. *Assume that R is s.p.d and S is s.p.s-d. Then the extreme eigenvalues of the preconditioned matrix M_α, defined in (14), satisfy*

$$\mu_{\min} = \begin{cases} \dfrac{1}{1+\alpha^2}, & \text{if} \quad 0 \le \alpha \le \hat{\lambda} \\ \dfrac{1+\hat{\lambda}^2}{(1+\alpha\hat{\lambda})^2}, & \text{if} \quad \hat{\lambda} \le \alpha \end{cases}$$

$$\mu_{\max} = \begin{cases} 1, & \text{if} \quad \hat{\alpha} \le \alpha \\ \dfrac{1+\hat{\lambda}^2}{(1+\alpha\hat{\lambda})^2}, & \text{if} \quad 0 \le \alpha \le \hat{\alpha} \end{cases}$$

where $\hat{\lambda}$ is the maximal eigenvalue of $R^{-1}S$,

$$R^{-1}S \le \hat{\lambda} I$$

and

$$\hat{\alpha} = \frac{\hat{\lambda}}{1+\sqrt{1+\hat{\lambda}^2}}$$

The spectral condition number is minimized when $\alpha = \hat{\alpha}$, in which case

$$\mu_{\min} = \frac{1}{1+\hat{\alpha}^2}, \qquad \mu_{\max} = 1$$

$$\mathrm{Cond}(M_\alpha) = 1 + \hat{\alpha}^2 = 2\frac{\sqrt{1+\hat{\lambda}^2}}{1+\sqrt{1+\hat{\lambda}^2}}$$

Proof
The bounds of the extreme eigenvalues follow by elementary computations of $\mu = (1+\lambda^2)/(1+\alpha\lambda)^2, \quad 0 \le \lambda \le \hat{\lambda}$. Similarly, it is readily seen that $\mu_{\max}/\mu_{\min}$ is minimized for some α in the interval $\hat{\alpha} \le \alpha \le \hat{\lambda}$, where $\mu_{\max} = 1$. Hence, it is minimized for $\alpha = \mathrm{argmax}_{\hat{\alpha}\le\alpha}(1+\alpha^2)^{-1}$, i.e. for $\alpha = \hat{\alpha}$. ■

Note that condition number is bounded above by 2 for any choice of α, $\hat{\alpha} \leq \alpha \leq 1$. In practice $\hat{\lambda}$ is often large, so $\hat{\alpha} = 1 - 1/\hat{\lambda} + O(1/\hat{\lambda}^2), \quad \hat{\lambda} \to \infty$. Therefore , if $\hat{\lambda}$ is not known we let $\alpha = 1$, in which case the smallest eigenvalue is $\frac{1}{2}$, the largest is 1 and the condition number takes the value 2.

2.2. A two-parametric real-valued method

As has been shown in Reference [20], the RV method can be extended using two parameters. Let then $0 < \beta < \alpha$ and consider

$$(R + \frac{1}{\alpha}S)^{-1}C_\alpha x = (R + \frac{1}{\alpha}S)^{-1} f_\alpha \tag{20}$$

where C_α is as defined in Equation (11) and $f_\alpha = \phi + S(R + \alpha S)^{-1}(\psi - \alpha\phi)$. Similarly

$$(R + \frac{1}{\beta}S)^{-1}C_\beta x = (R + \frac{1}{\beta}S)^{-1} f_\beta. \tag{21}$$

Then, subtracting Equation (20), multiplied by $(\alpha + 1)^2/2(\alpha^2 + 1)$, from Equation (21), multiplied by $(\beta + 1)^2/2(\beta^2 + 1)$, we obtain

$$M_{\alpha,\beta}x = g_{\alpha,\beta}$$

where

$$M_{\alpha,\beta} = \frac{(\beta+1)^2}{2(\beta^2+1)}(R + \frac{1}{\beta}S)^{-1}C_\beta - \frac{(\alpha+1)^2}{2(\alpha^2+1)}(R + \frac{1}{\alpha}S)^{-1}C_\alpha$$

and $g_{\alpha,\beta}$ is defined similarly. If $H = R^{-1/2}SR^{-1/2}$ has eigenvalue λ, a computation shows that $M_{\alpha,\beta}$ has eigenvalues

$$\mu = \frac{1}{2}(1 + \frac{1}{b})\frac{1}{1 + b\eta} - \frac{1}{2}(1 + \frac{1}{a})\frac{1}{1 + a\eta} \tag{22}$$

where

$$a = \frac{1}{2}(\alpha + \frac{1}{\alpha}) > 1, \qquad b = \frac{1}{2}(\beta + \frac{1}{\beta}) \geq 1, \qquad \eta = \frac{2\lambda}{1 + \lambda^2}$$

and $0 \leq \eta \leq 1$, since $0 \leq \lambda < \infty$.

Using Equation (22) it can be shown that

$$\mathrm{Cond}(M_{\alpha,\beta}) = \frac{\mu_{\max}}{\mu_{\min}} = \left(\frac{2a + 1}{a\sqrt{2} + \sqrt{a + 1}}\right)^2 \tag{23}$$

where we can take $\beta = 1$, which is the optimal value of β to minimize the condition number. Since the function in the right hand side of Equation (23) is monotonically increasing in a, we let a be a number close to 1 (but $a > 1$, because $\alpha > \beta$ should hold). The lower bound of the condition number, which corresponds to a value $a = 1$ is $(3/2\sqrt{2})^2 = 9/8 = 1.125$.

Each iteration of the above two-parameter RV method involves four solutions of systems with $R + \theta S$, where $\theta = \alpha, 1/\alpha$ and 1. Alternatively, we can use two steps of the CG iteration method with one parameter, the cost of which will be about the same as one step of the two-parameter method. Since both methods involve rational functions in H of the same order, and because of the optimality property of the CG (or Chebyshev) iteration method, it can be seen that the latter never converges more slowly

than the first method. Therefore, since both methods involve about the same amount of computation, there is actually no advantage in general in using the two-parameter preconditioning method compared with a one parameter when the CG (or Chebyshev) acceleration scheme is used.

However, a two-parameter RV method is better tailored for parallel computers, because twice the number of parallel processors can be utilized as in the one-parameter method.

2.3. Solving matrix polynomial equations

Consider now the matrix polynomial equation (1). We assume that Q_m has real coefficients. We want to reduce the solution of Equation (1) to a number of first order equations. To this end, we note that Q_m can be factored in first and second order factors with real coefficients, where each second factor corresponds to a pair of complex conjugate roots of Q_m. Therefore, it suffices to consider a second order polynomial

$$P_2(A) = I + 2\mathrm{Re}(a)A + a\bar{a}A^2 \tag{24}$$

where $-a^{-1}$, $-\bar{a}^{-1}$ are the roots of P_2. An elementary computation shows that

$$P_2^{-1}(A) = b(I + aA)^{-1} + \bar{b}(I + \bar{a}A)^{-1}$$

where

$$b = \frac{a}{a - \bar{a}} = \frac{1}{2} - \imath\frac{1}{2}\frac{\mathrm{Re}(a)}{\mathrm{Im}(a)}$$

Now there are two ways how to solve an equation $P_2(A) = f$. The first approach is to implement the operation $y = P_2^{-1}(A)f$ by solving a family of two shifted linear systems with one right-hand side f:

$$(I + aA)v_1 = f$$
$$(I + \bar{a}A)v_2 = f$$

and then compute $y = bv_1 + \bar{b}v_2$.

On the other hand, for $w_2 = \bar{v_2}$ we have $(I + aA)w_2 = \bar{f}$, since matrix A is real. Thus, we can find the solution vector y by solving a family of two systems with the same matrix, but different right-hand sides,

$$(I + aA)w_1 = f$$
$$(I + aA)w_2 = \bar{f}$$
$$y = bw_1 + \bar{b}w_2$$

If f is real, then $w_2 = \bar{w_1}$ and $v_2 = \bar{v_1}$ and a second system in each family is no longer necessary. In this case both algorithms can be simplified. In particular, we obtain the following algorithm:

Algorithm 2.2. To solve matrix polynomial equation $P_2(A)y = f$ with real A, f, and complex a

Step 1. **Solve** $(I + aA)w = f$
Step 2. **Compute** $y = 2Re(bw)$

3. APPLICATIONS FOR DIFFERENTIAL EQUATIONS

Let u and f be N-dimensional vector functions and consider the system of ordinary differential equations

$$\frac{\mathrm{d}u(t)}{\mathrm{d}t} + Lu(t) = f(t), \qquad t > 0, \qquad u(0) = u_0 \tag{25}$$

where is L is an $N \times N$ real matrix that does not depend on t and with eigenvalues with positive real parts.

3.1. Periodic source function

Assume $f(t)$ to be periodic in t, i.e.

$$f(t) = f_0 e^{\imath\omega t}$$

where f_0 is a real constant vector. As well known, the solution of Equation (25) takes the form

$$u(t) = \exp(-Lt)u_0 + \int_0^t \exp\left(L(s-t)\right) f(s)\,\mathrm{d}s$$

and an elementary computation shows that for the periodic function f

$$u(t) = \exp(-Lt)(u_0 - v_0) + e^{\imath\omega t} v_0 \tag{26}$$

where v_0 is the solution of

$$(L + \imath\omega I)v_0 = f_0 \tag{27}$$

By assumption, $\exp(-Lt) \to 0$, so the stationary solution of Equation (25) is periodic, $e^{\imath\omega t}$.

One can apply the RV method to solving (27) in several ways. The first approach is to choose

$$R = L, \qquad S = \omega I \tag{28}$$

and then obtain

$$\hat{\lambda} = \frac{\omega}{\lambda_{\min}(L)}, \quad \alpha_{\mathrm{opt}} = \hat{\alpha} = \frac{1}{\sqrt{1 + (\frac{\lambda_{\min}(L)}{\omega})^2} + \frac{\lambda_{\min}}{\omega}} \tag{29}$$

On the other hand we can multiply Equation (27) by $\imath$ and get a system

$$(\omega I + \imath L)\bar{v}_0 = \imath\bar{f}_0 \tag{30}$$

instead. This gives us

$$\hat{\lambda} = \frac{\lambda_{\max}(L)}{\omega}, \quad \alpha_{\mathrm{opt}} = \hat{\alpha} = \frac{1}{\sqrt{1 + (\frac{\omega}{\lambda_{\max}(L)})^2} + \frac{\omega}{\lambda_{\max}}} \tag{31}$$

Of those methods, the one is better for which $\hat{\alpha}$, or equivalently, $\hat{\lambda}$ is smaller. So, a value

$$\omega_0 = \sqrt{\lambda_{\min}(L)\lambda_{\max}(L)}$$

is a point of choice between two above approaches. In other words, for smaller frequencies $\omega < \omega_0$ the choice of Equations (28), (29) is better, while for larger frequencies $\omega \geq \omega_0$ it is better to transform Equation (27) in Equation (30) and take $R = \omega I$, $S = L$. If $\lambda_{\min}(L)$ or $\lambda_{\max}(L)$ are not known, we let $\alpha = 1$. In any case, the spectral condition number of the preconditioned matrix is bounded above by 2 for all ω, $0 \leq \omega < \infty$. Hence we observe and can explain the effect of the superconvergence of iterations for the regions of low and high frequencies ω. A similar effect is typical for small-signal ac analysis in electronic device modelling (see Reference [21]), but it is known also that most iterative methods used for such problems, lose convergence at high frequencies and another method, claimed to be robust and efficient, shows a significant increase of computational costs for an intermediate interval of frequencies.

Clearly, the proposed methods are applicable also when f is a sum of periodic functions, $f = \sum f_k e^{i\omega_k t}$, in which case we seek u in the form $u = exp(-Lt)(u_0 - \sum v_k) + \sum v_k e^{i\omega_k t}$, where each v_k satisfies $(L + i\omega_k I)v_k = f_k$. Note that the vectors v_k can be computed in parallel.

3.2. Parabolic problems, general source function

Consider now Equation (25) for a general function $f(t)$. The exponential function in Equation (26) can be approximated using a Padé approximation

$$R_{km}(z) = \sum_{j=0}^{k} \frac{k!}{(k-j)!}\frac{(k+m-j)!}{(k+m)!}\frac{z^j}{j!} \Big/ \sum_{j=0}^{m} \frac{m!}{(m-j)!}\frac{(k+m-j)!}{(k+m)!}\frac{(-z)^j}{j!}$$

which has error (see, for instance Reference [22]),

$$e^z - R_{km}(z) = (-1)^m \frac{k!m!}{(k+m)!(k+m+1)!} z^{k+m+1} + O(z^{k+m+2}).$$

It is known that if $m \geq 2$, then the denominator polynomial has complex zeroes, except when m is odd in which case one zero is real. Hence the arising matrix polynomial equation can be solved using the methods in Sections 1 and 2. This particular application has been discussed previously in References [16,17].

Consider now the following step by step approximation of type R_{22} of Equation (25)

$$\left(I + \frac{1}{2}\tau L + \frac{1}{12}\tau^2 L^2\right) u_{n+1} = \left(I - \frac{1}{2}\tau L + \frac{1}{12}\tau^2 L^2\right) u_n + \tau\phi_n \tag{32}$$

where

$$\phi_n = \frac{1}{2}(f_{n+1} + f_n) + \frac{1}{12}\tau L(f_{n+1} - f_n) - \frac{1}{12}\tau(f'_{n+1} - f'_n), \quad n = 0, \ldots$$

and τ is the step-size. This approximation has an optimal fourth order of accuracy. It is computationally

efficient to rewrite Equation (32) in the form

$$\left(I + \frac{1}{2}\tau L + \frac{1}{12}\tau^2 L^2\right)(u_{n+1} - u_n) = \tau(-Lu_n + \phi_n), \quad n = 0, 1, \ldots$$

Applying Algorithm (2.1) we obtain the following algorithm:

$$\left.\begin{array}{l} \textbf{For} \quad n = 0, 1, \ldots, \quad \textbf{do} \\ \quad g_n = -Lu_n + \phi_n, \\ \quad \textbf{Solve} \quad (I + (1 + \frac{1}{\sqrt{3}}\imath)\frac{\tau}{4}L)v = \tau g_n \\ \quad u_{n+1} = u_n + v_{\mathrm{Re}} + \sqrt{3}v_{\mathrm{Im}} \\ \textbf{enddo} \end{array}\right\} \tag{33}$$

where v_{Re} and v_{Im} denote the real and imaginary parts of v, respectively.

Assuming that L is positive semi-definite, which is typical for parabolic second order problems where the differential operator in space has been discretized using finite difference or finite element methods, and using the previously discussed method to solve the complex system with the preconditioner $(I + \frac{\tau}{4}L + \alpha\frac{\tau}{4\sqrt{3}}L)$ where in Theorem 2.1, $R = I + \frac{\tau}{4}L$, $S = \frac{\tau}{4\sqrt{3}}L$

$$\hat{\lambda} = \max \lambda((I + \frac{\tau}{4}L)^{-1}\frac{\tau}{4\sqrt{3}}L) \leq \frac{1}{\sqrt{3}}$$

we find

$$\hat{\alpha} = \frac{1/\sqrt{3}}{1 + \sqrt{1 + \frac{1}{3}}} = \frac{1}{2 + \sqrt{3}}$$

The condition number κ of the corresponding preconditioned matrix is (see Theorem 2.1)

$$\kappa = 1 + \hat{\alpha}^2 = \frac{4}{2 + \sqrt{3}} \simeq 1.072$$

This corresponds to a reduction factor

$$\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \simeq 0.018$$

and with such a small reduction factor it suffices in practice with few iterations to solve Equation (33).

3.3. Hyperbolic problems

Consider now the problem (25) in the form

$$\frac{\mathrm{d}U}{\mathrm{d}t} + \tilde{L}U = F(t), \quad t > 0, \quad U(0) = U_0 \tag{34}$$

where

$$U = \begin{bmatrix} U^{(1)} \\ U^{(2)} \end{bmatrix}, \quad \tilde{L} = \begin{bmatrix} 0 & -I \\ L & 0 \end{bmatrix}, \quad F = \begin{bmatrix} 0 \\ f(t) \end{bmatrix}$$

and L is symmetric positive semi-definite, typically a discrete analogue of the negative Laplacian operator. Systems of the form of Equation (34) arise when solving second order wave equations

$$\frac{\partial^2 u}{\partial t^2} = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + f(x, y), \quad t > 0$$

for instance.

To solve Equation (34) we apply the previously presented R_{22}-Padé type approximation and the algorithm (2.2) now takes the form

$$\left.\begin{array}{l} \textbf{For} \quad n = 0, \ldots, \quad \textbf{do} \\ \quad g_n = \begin{bmatrix} U_n^{(2)} - \frac{1}{12}\tau(f_{n+1} - f_n) \\ -LU_n^{(1)} + \frac{1}{2}(f_{n+1} + f_n) - \frac{1}{12}\tau(f'_{n+1} - f'_n) \end{bmatrix} \\ \quad \textbf{Solve} \quad \begin{bmatrix} I & -(1 + \frac{1}{\sqrt{3}}\imath)\frac{\tau}{4}I \\ (1 + \frac{1}{\sqrt{3}}\imath)\frac{\tau}{4}L & I \end{bmatrix} V = \tau g_n, \\ \quad U_{n+1} = U_n + V_{\mathrm{Re}} + \sqrt{3}V_{\mathrm{Im}}, \quad n = 0, 1, \ldots \\ \textbf{enddo} \end{array}\right\} \tag{35}$$

To find components $V^{(1)}$, $V^{(2)}$ of V from Equation (35) we must solve the equation

$$[I + (1 + \sqrt{3}\imath)\frac{\tau^2}{24}L]w = \psi \tag{36}$$

where $\psi = g_n^{(1)} + (1 + \frac{1}{\sqrt{3}}\imath)\frac{\tau}{4}g_n^{(2)}$, and then compute

$$\begin{aligned} V^{(1)} &= \tau w, \\ V^{(2)} &= \tau(g^{(2)} - \tfrac{1}{4}(1 + \tfrac{\imath}{\sqrt{3}})LV^{(1)}) \end{aligned}$$

The preconditioning matrix for Equation (36) is

$$I + \frac{1 + \alpha\sqrt{3}}{24}\tau^2 L \tag{37}$$

and we find

$$\hat{\lambda} = \sqrt{3}/(1 + \frac{24}{\tau^2\lambda_{\max}(L)}) \leq \sqrt{3}$$

If L is the standard difference approximation of the Laplacian operator on a unit square and $\tau = h$,

where h is the space discretization parameter, then

$$\hat{\lambda} = \frac{\sqrt{3}}{4}, \quad \alpha = \hat{\alpha} = \frac{\sqrt{3}}{4 + \sqrt{19}}$$

and the condition number κ of the preconditioned matrix (cf. Theorem 2.1) becomes

$$\kappa \leq 1 + \frac{3}{(4 + \sqrt{19})^2} \simeq 1.04$$

which results in a reduction factor

$$\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \simeq 0.01$$

Note that in this case, when $\tau = O(h)$, the preconditioning matrix (37) has a condition number $O(1)$. Therefore, each such system can be solved with a computational cost $O(N)$, where N is the number of degrees of freedom. Hence, the above algorithm works essentially as an explicit time-stepping method, but is still unconditionally stable.

4. NUMERICAL COMPARISONS

This section is devoted to the numerical comparison of the proposed RV method with the well-known and popular complex symmetric QMR method. We shall compare these methods both in the number of iterations and in a solution time on the following two types of problems—shifted 'omega' systems (see Subsection 3.1)

$$(L + \imath\omega I)u = f \tag{38}$$

and R_{22}-Padé parabolic complex systems

$$\left(I + (1 + \frac{1}{\sqrt{3}}\imath)\frac{\tau}{4}L\right)u = f \tag{39}$$

arising in Padé approximation type integration schemes for parabolic problems (see subsection 3.2). Here L is the matrix of a standard five point discrete operator, approximating the negative Laplacian operator

$$Ly(s) = -\frac{\partial^2 y}{\partial s_1^2} - \frac{\partial^2 y}{\partial s_2^2}, \qquad s = (s_1, s_2) \in \Omega$$

with homogeneous Dirichlet boundary conditions on an uniform mesh $\bar{\Omega}_h$ in the unit square. Parameter h is a step-size, $h = 1/(\ell + 1)$, where ℓ is a number of inner grid-points (e.g. unknowns) in one direction.

We run all experiments on large 2-D meshes with $\ell = 100, 200, \ldots, 500$, so the arising systems contain from 10 000 to 250 000 unknowns. The memory capacity of modern computers allow us to handle such problems with no difficulty. On the other hand there is an actual need in the fine meshes, especially, when solving shifted 'omega' systems. In fact, consider for simplicity the 1-D case, e.g. the

homogeneous equation

$$L_1 u(s) = -\frac{d^2 u(s)}{ds^2} + \imath\omega u(s) = 0 \tag{40}$$

the solution of which, in general, is an oscillating function

$$u(s) = c_1 \exp(\sqrt{\omega}\frac{1+\imath}{2}s) + c_2 \exp(-\sqrt{\omega}\frac{1+\imath}{2}s)$$

where $c_1,\ c_2$ are some constants. From Equation (40) we obtain

$$u^{(4)} = -\omega^2 u$$

and the truncation error of a usual three-point finite difference scheme, approximating Equation (40), taken on the solution u is

$$L_1 u(s)_h - L_1^h u^h = -\frac{\omega^2 h^2}{12} u + O(\omega^4 h^4)$$

Thus, to provide convergence we have to take a small mesh that depends on the parameter ω, $\omega h < 1$.

Now we discuss the issue of how to solve an equation (subscript α is omitted)

$$By \equiv (R + \alpha S)y = g \tag{41}$$

constituting the major computational task in the RV algorithm. The first and natural suggestion is to use some preconditioned iterative solver. It is especially viable since the particular problem (40) under consideration posses many features favorable for iterative methods.

However we have used a sparse direct solver with traditional minimum degree ordering available from the Yale Sparse Matrix Package—YSMP [23] (see also Reference [24] about the minimum degree and other ordering algorithms). Since the RV method generate two systems (41) on each iteration step, and some number of iterations, typically three to eight, must be done, the cost of factoring matrix B in triangular factors is not detrimental, if only the resulting triangular systems can be solved efficiently. We shall see later that even on a grid with 500×500 nodes the expense of solving both triangular systems does not exceed about 80 multiplications per grid-point. Thus the choice of the sparse direct method is optimal for our problem.

To increase the efficiency of a sparse direct solver we apply an initial 2×2 decomposition of matrix B, associated with so-called red–black colouring for five-point grid problems (see, for instance, Reference [25]). After the red-black ordering the matrix B takes the form

$$\begin{pmatrix} D_1 & B_{21}^{\mathrm{T}} \\ B_{21} & D_2 \end{pmatrix}$$

where $D_1,\ D_2$ are diagonal matrices of order $N_1 = \lceil N/2 \rceil$, and $N_2 = n - N_1$, respectively. Then, we perform a 2×2 block decomposition

$$\begin{pmatrix} D_1 & B_{21}^{\mathrm{T}} \\ B_{21} & D_2 \end{pmatrix} = \begin{pmatrix} I & 0 \\ B_{21} D_1^{-1} & I \end{pmatrix} \begin{pmatrix} D_1 & 0 \\ 0 & \tilde{B}_{22} \end{pmatrix} \begin{pmatrix} I & D_1^{-1} B_{21}^{\mathrm{T}} \\ 0 & I \end{pmatrix}$$

Table I. Sparse direct method for the original and reduced systems.

		Five-point system		Reduced system	
ℓ	N	nzU	nzU/N	nzU	nzU/N
100	10 000	195 647	19.56	167 698	16.77
200	40 000	1 052 542	26.31	892 619	22.32
300	90 000	2 824 060	31.38	2 347 897	26.09
400	160 000	5 707 412	35.67	4 646 083	29.04
500	250 000	9 649 230	38.60	7 819 036	31.28

Table II. Time for sparse direct solution of the original and reduced systems.

		Five-point system		Reduced system	
ℓ	N	TimeFact	TimeSolv	TimeFact	TimeSolv
100	10 000	0.728	0.044	0.555	0.035
200	40 000	6.605	0.240	5.324	0.195
300	90 000	26.40	0.636	21.04	0.506
400	160 000	73.32	1.284	55.80	1.010
500	250 000	154.08	2.159	123.38	1.660

where

$$\tilde{B}_{22} = D_2 - B_{21} D_1^{-1} B_{21}^{\mathrm{T}} \tag{42}$$

Since matrix D_1 is diagonal then the Schur complement matrix can be computed easily. In our case matrix $\tilde{B}_{22}$ contains no more than nine non-zero entries per row.

Using this decomposition we obtain the following algorithm for solving the system $By = g$:

$$\begin{aligned} &\textbf{Compute} \quad z_1 = D_1^{-1} g_1 \\ &\textbf{Solve} \quad \tilde{B}_{22} y_2 = g_2 - B_{21} z_1 \\ &\textbf{Compute} \quad y_1 = z_1 - D_1^{-1} B_{21}^{T} y_2 \end{aligned}$$

The described initial red–black decomposition allowed to reduce the computational expense required on solving the systems 1.2–1.3 times. The corresponding results are shown in Tables I and II, in which for both approaches the sparse direct method with minimum degree ordering, $PBP^{\mathrm{T}} = U^{\mathrm{T}}U$, P a permutation matrix, is applied to the original system (40) or to the reduced system with matrix $\tilde{B}_{22}$ from Equation (42).

The following data are displayed:

nzU—the number of non-zero entries in the triangular factor U, where $PBP^{\mathrm{T}} = U^{\mathrm{T}}U$ or, in the case of the reduced system, $P\tilde{B}_{22}P^{\mathrm{T}} = U^{\mathrm{T}}U$,

Table III. Results for the shifted 'omega' system, $\omega = 0.1$.

	RV method				CS QMR method	
ℓ	Iters	Time		RV-RS time	Iters	Time
100	3	0.55	(1.31)	0.58 (1.22)	383	34.78
200	3	2.91	(9.80)	3.18 (8.94)	746	283.45
300	3	7.43	(34.42)	8.05 (30.06)	1103	942.62
400	3	14.84	(89.80)	14.97 (72.07)	1458	2218.00
500	3	24.30	(178.00)	25.34 (152.70)	1811	4304.10

Table IV. Results for the shifted 'omega' system, $\omega = 1$.

	RV method				CS QMR method	
ℓ	Iters	Time		RV-RS time	Iters	Time
100	4	0.65	(1.41)	0.69 (1.33)	383	35.20
200	4	3.43	(10.26)	3.71 (9.39)	746	283.26
300	4	8.99	(35.84)	9.55 (31.42)	1103	945.72
400	4	17.75	(91.99)	18.51 (75.70)	1458	2222.40
500	4	29.31	(183.09)	30.92 (159.30)	1811	4318.80

nzU/N—the average number of non-zeroes in a triangular factor U per unknown (scaling to the original number of unknowns N is always used),

$TimeFact$ and $TimeSolv$—time in seconds for the matrix factorization and the solution of a factorized system, respectively.

All numerical experiments were performed on a workstation Sun Enterprise 10000 (170 MHz, 1 Gb of RAM).

Numerical results for shifted 'omega' systems are presented in Tables III–V and Figure 1. To compare

Table V. Results for the shifted 'omega' system, $\omega = 10$.

	RV method				CS QMR method	
ℓ	Iters	Time		RV-RS time	Iters	Time
100	7	0.97	(1.73)	1.06 (1.72)	382	35.26
200	7	5.10	(11.90)	5.60 (11.34)	743	281.14
300	7	13.32	(39.97)	14.27 (36.05)	1100	939.64
400	7	26.59	(99.92)	27.26 (83.97)	1453	2202.10
500	7	44.85	(200.00)	45.46 (173.41)	1805	4286.50

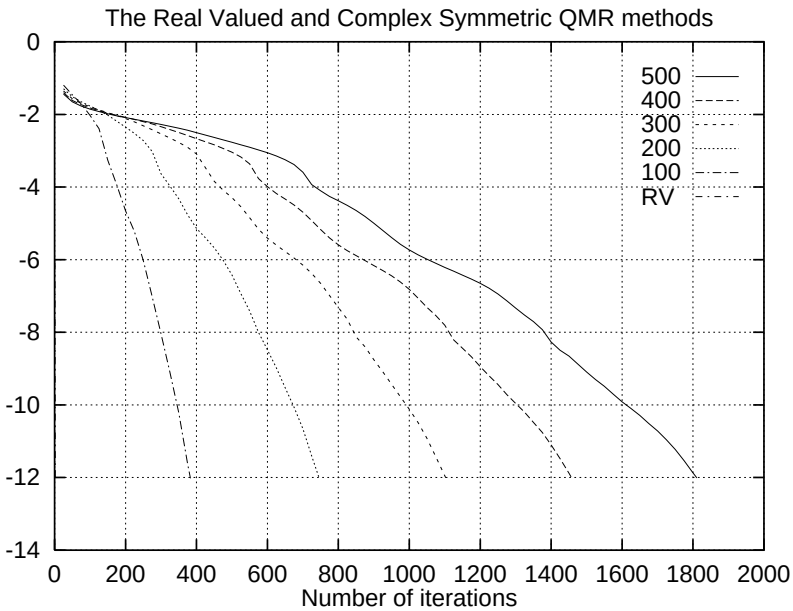

Figure 1. Shifted 'omega' problem, $\omega = 0.1$.

the real valued (RV) method with the Complex Symmetric QMR method (CS QMR) we display the number of iterations (iters) and the solution time (time). As was noted in the Introduction the task of how to precondition the complex-valued system robustly and effectively has received little attention in the literature and still remains open. Therefore we put aside the issue of comparison with some complex-valued iterations in preconditioned form. For the variant of the RV method, in which a sparse direct solver is applied to the red–black reduced systems, we use the notation RV-RS. For both these methods we display the time to solve the given complex system and in brackets the total time of the RV method, which includes a time required to factorize the preconditioning matrix $R + \alpha S$. For all tests the initial approximation was taken as the zero vector, a right hand side b of the complex system $Au = b$ was taken as a vector with components

$$b_j = \frac{j}{j+1}(1 - \frac{j}{j+1})(1 - \imath), \quad j = 1, 2, \ldots, N$$

and the required relative accuracy was $\epsilon = 10^{-12}$. The convergence history for these methods are plotted in Figure 1.

The results for the R_{22}-Padé parabolic problem with $\tau = h$ are presented in Table VI and Figure 2. Similar conditions were used for the experiments, as used previously.

Table VI. Results for R_{22}-Padé parabolic problem.

	RV method				CS QMR method	
ℓ	Iters	Time		RV-RS time	Iters	Time
100	7	1.01	(1.80)	1.04 (1.70)	197	17.89
200	7	5.18	(12.41)	5.58 (11.39)	284	105.92
300	7	13.59	(40.90)	14.58 (36.63)	352	296.91
400	7	26.56	(101.10)	27.26 (84.59)	405	606.19
500	7	44.08	(197.20)	46.21 (174.30)	450	1052.80

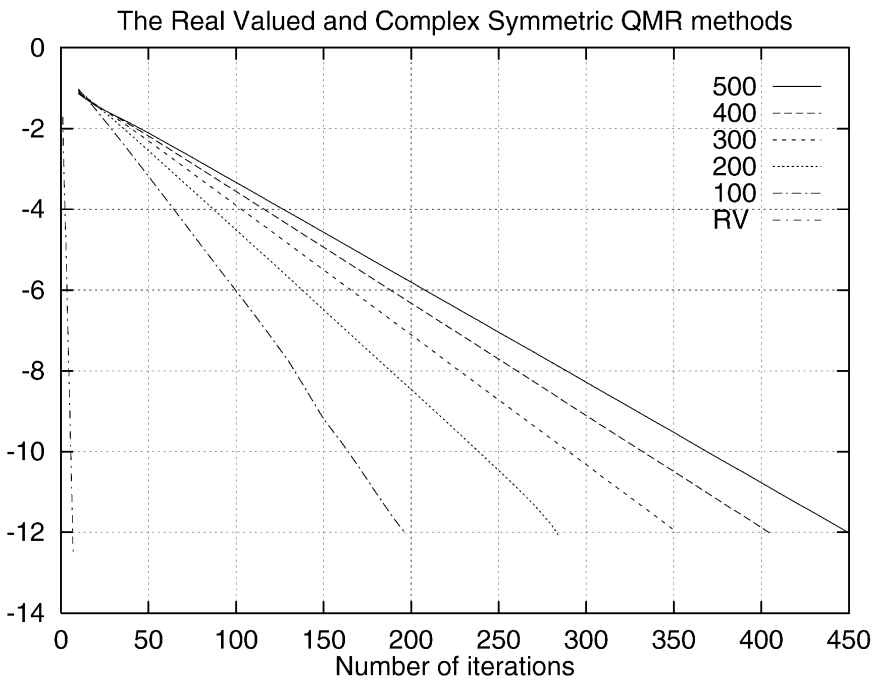

Figure 2. Padé parabolic problem, $\tau = h$.

5. CONCLUSIONS

It has been shown how certain complex symmetric linear systems of equations can be rewritten to involve only real arithmetic. The new reduction yields a real symmetric positive definite system and in this respect favourably differs from well-known basic reductions that produce real matrices possessing unsatisfactory spectral properties.

The resulting system can be solved efficiently by iteration involving a positive definite system as preconditioner and a small resulting condition number that is no greater than 2.

Various applications of the method when solving matrix polynomial equations and evolution equations have also been demonstrated. Numerical comparisons with the QMR method showed the proposed real valued method requires significantly less computer time for solving linear systems with a complex symmetric matrix $R + \imath S$ when R and S are symmetric positive definite and at least one of R and S is positive definite.

REFERENCES

1. Saad Y, Schultz MH. GMRES: A generalized minimum residual algorithm for solving nonsymmetric linear systems. *SIAM Journal on Scientific and Statistical Computing* 1986; **7**:856–869.
2. Axelsson O. *Iterative Solution Methods*. Cambridge University Press: New York, 1994.
3. Axelsson O. A generalized conjugate gradient, least square method. *Numerische Mathematik* 1987; **51**:209–227.
4. Axelsson O, Nikolova M. A generalized conjugate gradient minimum residual method (GCG-MR) with variable preconditioners and relation between residuals of the GCG-MR and GCG-OR methods. *Communications on Applied Analysis* 1997; **1**:371–388.
5. Freund R. Conjugate gradient-type methods for linear systems with complex symmetric matrices. *SIAM Journal on Scientific and Statistical Computing* 1992; **13**:425–448.
6. Bunse-Gerstner A, Stöever R. On a conjugate gradient-type method for solving complex symmetric systems. *Linear Algebra and its Applications* 1999; **287**:105–123.
7. Jacobs DAH. A generalization of the conjugate gradient method to solve complex systems. *IMA Journal of Numerical Analysis* 1986; **6**:447–452.
8. Markham G. Conjugate gradient type methods for indefinite, asymmetric and complex systems. *IMA Journal of Numerical Analysis* 1990; **10**:155–170.
9. Joly P, Meurant G. Complex conjugate gradient methods. *Numerical Algorithms* 1993; **4**:379–406.
10. Freund R. Solution of shifted linear systems by quasi-minimal residual iterations. In *Numerical Linear Algebra* , Reichel L, Ruttan A, Varga RS (eds). W. de Gruyter 1993;101–121.
11. Young D. The search for 'high-level' parallelism for iterative sparse linear system solvers. In *Parallel Supercomputing: Methods, Algorithms and Applications*, Carey GF (ed). Wiley, 1989; 89–105.
12. Young DM, Vona BR. On the use of rational iterative methods for solving large sparse linear systems. *Applied Numerical Mathematics* 1992; **10**:261–278.
13. Datta BN, Saad Y. Arnoldi methods for large Sylvester-like observer matrix equations, and an associated algorithm for partial spectrum assignment. *Linear Algebra and its Applications* 1991; **154/156**:225–244.
14. Frommer A, Glässner U. Restarted GMRES for shifted linear systems. *SIAM Journal on Scientific Computing* 1998; **19**:15–26.
15. Freund R. On conjugate gradient type methods and polynomial preconditioners for a class of complex non-Hermitian matrices. *Numerische Mathematik* 1990; **57**:285–312.
16. Axelsson O. On the computational complexity of some matrix iterative algorithms. Department of Computer Sciences, Chalmers University of Technology, Göteborg, Sweden, 1974.
17. Axelsson O. On the efficiency of a class of A-stable methods. *BIT* 1974; **14**:279–287.
18. Kucherov AB. A fast iterative method for solving in real arithmetic a system of linear equation with a complex symmetric matrix. *Sovietsky Mathematischeskoe Doklady* 1991; **43**:377–379.
19. van Rienen U. *Numerical Methods in Computational Electrodynamics—Linear Systems in Practical Applications*. Springer Lecture Notes in Computational Science and Engineering, vol. 15, 2000, to be published.
20. Kucherov AB, Bastis A. Fast iterative methods for solving in a real arithmetic a discrete Helmholtz equation with a complex coefficient. *Mathematical Modelling* 1991; **3**:115–121.

21. Wu K-C, Yu Z, So L, Dutton RW, Sato-Iwanaga J. Robust and efficient AC analysis of high-speed devices. In *Proceedings of the 1992 Electron Device Meeting*, IEEE; 935–938.
22. Hairer E, Wanner G. *Solving Ordinary Differential Equations—Part II. Stiff and Differential Algebraic Problems*. Springer-Verlag: Berlin, 1996.
23. Eisenstat SC, Gursky MC, Schultz MH, Sherman AH. Yale sparse matrix package—Part I: The symmetric codes. *International Journal for Numerical Methods in Engineering* 1982; **18**:1145–1151.
24. George A, Liu JWH. *Computer Solution of Large Sparse Positive Definite Systems*. Prentice-Hall: Englewood Cliffs, NJ, 1981.
25. Axelsson O, Gustafsson I. On the use of preconditioned conjugate gradient methods for red-black ordered five-point difference schemes. *Journal of Computational Physics* 1980; **35**:284–299.

