Journal of Computational Physics 229 (2010) 7477–7487

Contents lists available at ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

A Rayleigh–Chebyshev procedure for finding the smallest eigenvalues and associated eigenvectors of large sparse Hermitian matrices

Christopher R. Anderson *

UCLA Mathematics Department, P.O. Box 951555, Los Angeles, CA 90095-1555, USA

ARTICLE INFO

Article history:
Received 1 February 2010
Received in revised form 20 May 2010
Accepted 16 June 2010
Available online 23 June 2010

Keywords:
Hermitian matrices
Eigenvalues
Eigenvectors
Inverse iteration
Chebyshev polynomials

ABSTRACT

A procedure is presented for finding a number of the smallest eigenvalues and their associated eigenvectors of large sparse Hermitian matrices. The procedure, a modification of an inverse subspace iteration procedure, uses adaptively determined Chebyshev polynomials to approximate the required application of the inverse operator on the subspace. The method is robust, converges with acceptable rapidity, and can easily handle operators with eigenvalues of multiplicity greater than one. Numerical results are shown that demonstrate the utility of the procedure.

© 2010 Elsevier Inc. All rights reserved.

1. Introduction

Inverse subspace iteration [14] is one of the commonly used procedures for finding the smallest eigenvalues and associated eigenvectors of large sparse Hermitian matrices. Inverse iteration is an attractive method because for many problems the computational cost is proportional to the cost of solving linear systems of equations times the number of desired eigenvalues and eigenvectors. General dense matrix procedures for finding eigenvalues and eigenvectors of $N \times N$ symmetric matrices have an $O(N^3)$ computational cost and an $O(N^2)$ storage requirement, so that even for modest sized sparse matrices, one expects inverse iteration to be much more efficient if the solution of the linear systems can be accomplished in less than $O(N^3)$ operations. Moreover, the storage requirements of the method are minimal, the procedure converges rapidly (especially when a good shift is chosen), and as the method is relatively easy to understand, non-experts in computational linear algebra are likely to create successful implementations. This latter fact is important, as libraries of reliable eigensystem routines may not be available for those exploring the use of new computational platforms. Also, for those who work on more established computational platforms, pre-existing routines for the core computational tasks—standard operations in computational linear algebra—are likely to be available so that efficient implementation can be accomplished rather quickly.

Unfortunately, there is a fundamental problem with inverse subspace iteration when it is applied to large sparse matrices. When shifts are chosen for optimal convergence rates, the procedure requires the solution of singular or nearly singular systems of equations. If one uses Gaussian elimination to solve these systems, then this singularity actually helps the computational process [14] (the errors introduced in the solution procedure are in the direction of the desired eigenvectors). However, when standard iterative methods are used to solve the requisite system of equations the convergence rates are

* Tel.: +1 310 825 1298; fax: +1 310 206 2679.
E-mail address: anderson@math.ucla.edu

0021-9991/$ - see front matter © 2010 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2010.06.030

typically reduced or the methods may even fail to converge. Additionally, the development of good preconditioners for these iterative methods is complicated by the singular or nearly singular nature of the systems. Starting with work by [18], and continuing with [5,17,20] this problem has been studied and procedures for avoiding the difficulties caused by the near singular behavior have been formulated. However, if one considers an inverse subspace iteration procedure where the requisite solutions of the linear systems are computed with Krylov subspace based iterative methods [10] such as Conjugate-Gradients (CG) or generalized minimum residual (GMRES), then one recognizes that the approximations to the eigenvectors are ultimately just polynomials in the operator applied to a particular collection of vectors. Essentially, when "good" shifts are used, then the implicit determination of these polynomials by the Krylov iterative method is breaking down. In this paper, the procedure that we present can be viewed as an implementation of an inverse subspace iteration procedure in which polynomial transformations are still used, but the manner in which the polynomials are being determined is different from that associated with a Krylov subspace iterative method. The net result is an efficient method that only requires matrix vector products, can readily take advantage of multi-processor machines, and has very good reliability properties. In particular, the method will always converge if the initial vectors have components in the desired eigenvectors and can handle without difficulty systems with eigenvalues of multiplicity greater than one.

In the next section we review inverse subspace iteration and present results demonstrating the typical problems that arise when one solves the requisite linear systems using a standard implementation of a Conjugate-Gradient method. In the third section we describe our procedure in general terms and then provide the details required for its implementation. In the last section we present computational results. In addition to results of a computation of the eigenvalues of a discretization of a 2D elliptic PDE we present results on the computation of the smallest eigenvalues and their associated eigenvectors of a large $(10^6 \times 10^6)$ matrix arising from a Full Configuration Interaction (FCI) treatment of a multi-particle Schroedinger equation. In this discussion we also describe the multi-processor implementation that was required to obtain the computational results.

The procedure presented in this paper can be viewed or understood in many different ways. For example, in addition to seeing it as a variant of inverse subspace iteration, it can also be viewed as a variant of general subspace iteration with Chebyshev acceleration [14,19]. Our presentation was selected in an effort to motivate and describe the procedure in terms that would be more readily understood by anyone familiar with inverse iteration—a topic typically covered in undergraduate numerical analysis classes. The computation of the smallest eigenvalues of large sparse symmetric matrices is of considerable interest in quantum chemistry and quantum physics applications, and there are other types of methods that are commonly used. In particular, there are procedures [6,11,13,21,22,24] based on Davidson's method [9] and procedures [3,8,7,15,16,23] based on the Lanczos method. Of particular interest are the procedures described in [24], where adaptive Chebyshev filtering is used in conjunction with Davidson's method and [3] where Chebyshev filtering is used in conjunction with a Lanczos procedure. Direct comparisons have not been carried out between these other procedures and the one presented here, but we conjecture that the computational efficiency of the Rayleigh–Chebyshev procedure should be close to that of a block Lanczos procedure with re-orthogonalization.

In the following, when the task of finding the eigenvalues and eigenvectors of a matrix A is discussed, we will also refer to A as a linear operator. We do this to emphasize that the method presented here only requires the evaluation of Av for input vectors v—an explicit matrix representation of A is not required for the procedure.

2. Difficulties with inverse subspace iteration with shift

Inverse subspace iteration with a shift is a power method using the operator $(A-\sigma)^{-1}$ applied to a collection of vectors S_v that are orthogonalized at each step. The value σ is the "shift". Approximations to the eigenvalues and eigenvectors closest to σ are obtained from the projection of the operator on S_v, e.g. by a Rayleigh–Ritz procedure. The method in its simplest form is

Inverse subspace iteration with shift

Given an initial collection of M vectors, S_0,

(a) Compute $V_v = (A-\sigma I)^{-1} S_{v-1}$ by solving $(A-\sigma I)\ v_v^i = s_{v-1}^i$. (1)
(b) Orthonormalize $V_v = Q_v R_v$ by modified Gram–Schmidt.
(c) Form $H_v = Q_v^T A Q_v$.
(d) Diagonalize $H_v = G_v^T \Theta_v G_v$.
(e) Form $S_v = Q_v G_v$ the Rayleigh–Ritz approximations to the eigenvectors and test for the convergence of approximate eigenvalues $\theta_i^v,\ i = 1 \ldots M$. If convergence has not been obtained, repeat iteration starting with (a) and $v = v + 1$.

When the solution of the linear systems $(A-\sigma)$ required to apply $(A-\sigma)^{-1}$ can be successfully computed, then the diagonal elements $\theta_i^{(v)}$ of Θ_v converge to the eigenvalues of A closest to σ. If one seeks the smallest eigenvalues of A and desires that $(A-\sigma)$ be positive definite to facilitate the iterative solution of the requisite linear equations, then σ should be chosen to be less than the smallest eigenvalue of A. (For a more efficient implementation of the algorithm that avoids the cost associated with step (*c*) see [14].)

If λ_i are the eigenvalues of A, then for a fixed value of σ the standard convergence results [14], show that the approximate eigenvalues $\theta_i^{(v)}$ satisfy

$$\left|\theta_i^{(v)} - \lambda_i\right| = O\left(\left|\frac{\lambda_i - \sigma}{\lambda_{M+1} - \sigma}\right|^v\right) \quad i = 1 \ldots M \tag{2}$$

Thus, for the convergence factors, $\left|\frac{\lambda_i - \sigma}{\lambda_{M+1} - \sigma}\right|$, to have their smallest values, the best choice of σ that preserves positive definiteness of the operator will be when $\sigma = \lambda_{\min} - \epsilon = \lambda_1 - \epsilon$ where ϵ is a small positive value. Unfortunately, this choice of σ results in the operator $(A - \sigma)$ being nearly singular. As mentioned in the introduction, when solving the linear systems with Gaussian elimination this singularity helps [14], rather than hurts, the computational procedure. However, when iterative methods are used, this is not the case.

To illustrate the typical problems encountered if one uses a standard Krylov subspace method to solve the linear systems required to apply $(A - \sigma)^{-1}$, we considered the application of this procedure to finding the smallest nine eigenvalues of a numerical discretization of the problem $(-\Delta - \cos(2\pi x))u(x,y) = \lambda u(x,y)$ where Δ is the two-dimensional Laplace operator, $(x,y) \in [0,1] \times [0,1]$ and $u(x,y)$ is periodic in both coordinate directions. An eighth order finite difference approximation to the second derivatives with a mesh size of $h = .01$ in each coordinate direction was used. The resulting discrete linear operator is represented by a sparse $10^4 \times 10^4$ matrix. This problem was selected as a test case because it possess both multiple eigenvalues and eigenvalues that are very close together, in fact, the relative gap size, $\left|\frac{\lambda_{i+1} - \lambda_i}{\lambda_{\max} - \lambda_{\min}}\right|$, between distinct eigenvalues of interest was $O(10^{-7})$.

In Table 1 we present the results of numerical experiments in which standard Conjugate-Gradients (CG) and preconditioned Conjugate-Gradients (PCCG) were used to solve the requisite linear systems of equations with a stopping tolerance of 1×10^{-7}. In this table are listed the number of inverse subspace iterations required to obtain approximations to the lowest nine eigenvalues with a relative error less than 2.5×10^{-6}, the maximal number of CG or PCCG iterations required to solve any one of the requisite linear systems of equations and the total number of applications of the operator A and the preconditioner $\widetilde{A}^{-1}$. The preconditioner, $\widetilde{A}^{-1}$, used was a symmetrized version of one sweep of Gauss–Seidel relaxation [10]. Each application of the preconditioner was counted as equivalent to one application of A so the values listed in the table are the sum of the total number of applications of A and $\widetilde{A}^{-1}$.

As expected, as the shift is chosen for optimal convergence of the inverse subspace iteration procedure, the number of inverse iterations required decreases and the number of iterations required to solve the requisite linear system increases. Interestingly, when CG is used, these two opposing effects appear to cancel each other out, so that for a reasonably large range of shifts the total number of applications of the operator is approximately constant. The preconditioned iterative method was more sensitive to the use of shifts near $\lambda_{\min}$, and in fact, as the maximal number of any single PCCG computation was limited to 1000 iterations, these results indicate that the PCCG method did not always converge. This latter result is an indicator of the extra difficulties involved when trying to develop effective preconditioners for the iterative solution of the linear systems of inverse subspace iteration.

Although a large number of Conjugate-Gradient iterations were required for the solution of the linear systems of this test problem, it is worth noting that the overall inverse subspace iteration procedure was still much more efficient than one utilizing a dense matrix eigensystem procedure. Specifically, on a computer with an Intel Xeon CPU running at 3.40 GHz the time for computing the lowest nine eigenvalues and associated eigenvectors using the LAPACK routine DSYEVX [2] was 23.0 min, while inverse subspace iteration with standard Conjugate-Gradients required only 0.2 min.

3. Rayleigh–Chebyshev procedure

While the above results indicate the problems that arise when using iterative methods to solve the linear systems of inverse subspace iteration, they also show that the procedure is ultimately successful. Even if one terminates a linear system solver before convergence, one often finds that the overall inverse subspace iteration procedure still converges. At each iteration, the Conjugate-Gradient method approximates the action of $(A - \sigma)^{-1}$ on a vector in S_{v-1} by an element of a Krylov subspace—a polynomial in the operator times a given vector. Thus, we infer that it is a good idea from the viewpoint of

Table 1
Inverse iteration results with varying shift factor $\sigma = \lambda_{\min} - \alpha(\lambda_{\max} - \lambda_{\min})$. The maximal number of CG or PCCG iterations listed is the maximal number of iterations required for any single system solution occurring during the inverse iteration procedure.

α	#Inverse iterations	max CG iterations	A applies (CG)	max PCCG iterations	$A, \widetilde{A}^{-1}$ applies (PCCG)
10^{-1}	1082	27	55579	11	204592
10^{-2}	117	84	25237	31	60099
10^{-3}	21	219	19686	88	28575
10^{-4}	11	335	18125	195	29797
10^{-5}	10	373	17622	310	37980
10^{-6}	10	419	17616	1000	57954
10^{-7}	10	517	18257	1000	61188

inverse iteration to transform S_{v-1} by a polynomial in the operator, but it appears to be bad idea to construct that polynomial from the Conjugate-Gradient solution process. Thus, to avoid the difficulties associated with using iterative procedures to apply $(A-\sigma)^{-1}$ to S_{v-1} we utilize polynomial transformations of S_{v-1}, but determine the polynomials explicitly rather than implicitly through an iterative solution procedure.

The motivation for the procedure for selecting the required polynomial transformations can be obtained by understanding the derivation of the error bound for subspace iteration (see [14] for a particularly good treatment), however, rather than repeat this material, we motivate the procedure from an intuitive perspective. Assuming that $\sigma < \lambda_{\min}$ then the convergence factors for inverse subspace iteration, $\left|\frac{\lambda_i-\sigma}{\lambda_{M+1}-\sigma}\right|$ can be understood to arise because the application of $(A-\sigma)^{-1}$ to a vector in S_{v-1} amplifies its ith eigenvector coefficient by $\frac{1}{\lambda_i-\sigma}$ while the orthogonalization step causes this amplification to be relative to the (larger) factors $\frac{1}{\lambda_j-\sigma}, j \leqslant M$. Hence one expects an effective damping of the ith coefficient for $i > M$ of the vectors in S_{v-1} to be determined by the ratio of these amplification factors, specifically

$$\left(\frac{\left(\frac{1}{\lambda_i-\sigma}\right)}{\left(\frac{1}{\lambda_j-\sigma}\right)}\right) = \left(\frac{\lambda_j-\sigma}{\lambda_i-\sigma}\right) \quad j = 1\ldots M \quad i > M$$

The largest of these damping factors, and that which dominates the convergence behavior, occurs when $i = M+1$ and hence the factors (2) are obtained.

A fundamental observation is that the convergence factors are determined by the ratios of the values $\frac{f(\lambda_i)}{f(\lambda_j)}$ for $j = 1\ldots M$, $i > M$ and where $f(x) = \frac{1}{x-\sigma}$. As noticed in early work by Lanczos in [12], and subsequently by many others, instead of using the operator $f(A) = (A-\sigma)^{-1}$, one can use a polynomial operator $p(A)$ where $p(x)$ is chosen to have qualitative features similar to those of $\frac{1}{x-\sigma}$ over the range $[\lambda_{\min}, \lambda_{\max}]$. If such polynomial operators are used, then the convergence factors will be determined by ratios $\frac{p(\lambda_i)}{p(\lambda_j)}$ for $j = 1\ldots M$ and $i > M$. If the polynomial can be chosen so that all these ratios are less than one, then the iteration will be convergent.

In Fig. 1 we display the function $\frac{f(x)}{f(\lambda_{\min})}$ over the range $[\lambda_{\min}, \lambda_{\max}]$ with σ chosen to be slightly less that $\lambda_{\min}$. Also plotted is a polynomial $\frac{p(x)}{p(\lambda_{\min})}$ where $p(x)$ has the same qualitative features as $\frac{1}{x-\sigma}$. Specifically, $p(x)$ has the property that it attains its largest values at the smallest eigenvalues λ_j for $j = 1\ldots 3$ and the $\frac{|p(\lambda_i)|}{p(\lambda_j)|} < 1$ for $j \leqslant 3$ and $i > 3$. The plots of $f(x)$ and $p(x)$ are both normalized by their maximal values to facilitate a visual comparison of the convergence factors that would result through their use. In particular, for the indicated eigenvalues, one would expect the convergence factors obtained with $p(x)$ to be almost the same as the convergence factors obtained with $f(x)$.

The utility of using a polynomial function of A instead of a rational function of A is immediately apparent—no inverses are required. The difficulty is in determining the polynomial operators, a difficulty that arises because the best choices depend upon the knowledge of the eigenvalues of A, and these are the values that one is trying to compute. In the Rayleigh–Chebyshev procedure we present here, the polynomials are from a family of second kind Chebyshev polynomials $p_v(x)$ where specific parameters associated with these polynomials are chosen using eigenvalue estimates from the Rayliegh–Ritz approximation using the vectors in the subspace S_v. The use of Chebyshev polynomials is not new, and in fact these are the polynomials that Lanczos originally suggested in [12]. What is new is adaptively determining the polynomials at each step using information from the Rayleigh–Ritz procedure in such a way that one has guaranteed convergence.

Since the determination of the polynomials is integrated into an overall iterative procedure, before describing the details of selecting these polynomials we outline the simplest form of the Rayleigh–Chebyshev procedure.

Rayleigh–Chebyshev Iteration

Given an initial collection of $M+P$ vectors ($P \geqslant 1$), S_0, and an estimate of $\lambda_{\min}$ and $\lambda_{\max}$ so that $p_0(x)$ can be determined, then the Rayleigh–Chebyshev iteration consists of a repetition of the following steps.

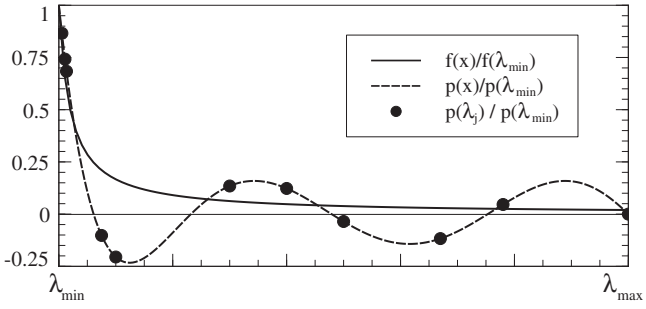

Fig. 1. Comparison of functions used for subspace transformation. The application of the transformation to a vector in the subspace amplifies its ith eigenvector component by $f(\lambda_i)$ or $p(\lambda_i)$. Convergence factors are determined by the ratio of the amplification of the components associated with the larger eigenvalues to the amplification of components associated with the smaller eigenvalues.

(a) Compute $V_v = p_{v-1}(A)S_{v-1}$
(b) Orthonormalize $V_v = Q_v R_v$ by modified Gram–Schmidt.
(c) Form $H_v = Q_v^T A Q_v$.
(d) Diagonalize $H_v = G_v^T \Theta_v G_v$.
(e) Determine $p_v(x)$ from θ_{M+P}, $\lambda_{\min}$ and $\lambda_{\max}$.
(f) Form $S_v = Q_v G_v$ the Rayleigh–Ritz approximations to the eigenvectors and test for the convergence of approximate eigenvalues θ_i^v, $i = 1 \ldots M$. If convergence has not been obtained, repeat iteration starting with (a) and $v = v + 1$. (3)

In this procedure we have included the use of $P \geqslant 1$ "buffer" vectors. This is done to ensure that the particular method for choosing the polynomials $p_v(x)$ leads to guaranteed convergence of the iterative process.

With the choice of Chebyshev polynomials described in the next section, the application of the polynomial $p_v(A)$ to vectors in S_v can be implemented using a two-term vector recurrence with one matrix–vector product required per recurrence step. Thus, if the number of non-zero elements in each row of the matrix is bounded independently of the system size N, then the computational cost of the Rayleigh–Chebyshev method is $O(N)$ per subspace iteration step, with a constant depending upon the degree of the polynomial and the number of vectors in the subspace.

Both this procedure and an inverse subspace iteration procedure in which the linear systems are solved via Krylov subspace methods transform the subspace S_{v-1} using a polynomial in the operator, a polynomial that can change at each step. However, in the Rayleigh–Chebyshev procedure the same polynomial is applied to all vectors in S_{v-1}, and the polynomials are determined using eigenvalue estimates obtained from the Rayleigh–Ritz procedure rather than implicitly as a consequence of the iterative solution procedure used to apply $(A - \sigma I)^{-1}$. This aspect of the method is significant for those cases when the fraction of the non-zero elements in each row of the matrix is not particularly small. In such cases one often seeks to improve computational efficiency by utilizing multiple processors, and their use is facilitated when the polynomial operator is the same for all vectors in the subspace. Also, as can be inferred from the structure of (3), the procedure can be readily implemented as a variant of a standard inverse subspace iteration procedure.

3.1. Polynomial selection

The selection of the polynomials p_v used at step (e) of the Rayleigh–Chebyshev iteration (3) require estimates of $\lambda_{\min}$ and $\lambda_{\max}$. There are many ways in which one can obtain such estimates, however, we have found that using the largest and smallest eigenvalues of the tridiagonal matrix obtained from a classical Lanczos procedure to be an effective method for computing these values.

In order to insure that the polynomials used in (3) lead to a convergent process, we require that every polynomial p_v used in the iteration satisfy the following two properties

$$\begin{aligned} &\text{(a)} \quad |p_v(x)| > |p_v(y)| \quad \text{for all } x \in [\lambda_{min}, \lambda^*] \quad \text{and } y \in (\lambda^*, \lambda_{\max}] \\ &\text{(b)} \quad p_v(x) \quad \text{is monotone over } [\lambda_{min}, \lambda^*] \end{aligned} \tag{4}$$

where λ^* is a value (which may depend upon v), such that $\lambda_j < \lambda^*$ for $j = 1 \ldots M$.

As suggested by Lanczos in [12], a particularly useful family of polynomials that can be used to satisfy these conditions are the Chebyshev polynomials. Specifically, let $C_n(x)$ denote the nth Chebyshev polynomial of the second kind that has been shifted and scaled to the interval $[0,1]$. These polynomials are defined by the recurrence

$$\begin{aligned} C_0(x) &= 1 \\ C_1(x) &= 1 - 2x \\ C_2(x) &= \left(\frac{2}{3}\right)(2 - 4x)C_1(x) - \left(\frac{1}{3}\right)C_0(x) \\ &\vdots \\ C_n(x) &= \left(\frac{n}{n+1}\right)(2 - 4x)\ C_{n-1}(x) - \left(\frac{n-1}{n+1}\right)C_{n-2}(x) \end{aligned} \tag{5}$$

The first step in creating the required polynomials from this set consists of determining, for each degree $n \geqslant 2$, a linear transformation, $l_n(x)$ so that the polynomial $q^n(x) = C_n(l_n(x))$ satisfies the conditions (4), for some value λ_n^*. Let γ_n be the second largest root of $C_n(x)$, and define $l_n(x)$ by

$$l_n(x) = \left(\frac{\gamma_n}{\lambda_{\max} - \lambda_{\min}}\right)[x - \lambda_{\min}] \tag{6}$$

The polynomial $q^n(x) = C_n(l_n(x))$ then satisfies (4) with value λ_n^* given by

$$\lambda_n^* = \left[\frac{\alpha_n^*[\lambda_{\max} - \lambda_{\min}] + \lambda_{\min}}{\gamma_n}\right] \tag{7}$$

where

$$\alpha_n^* = C_n^{-1}(|C_n(r_n)|) \tag{8}$$

with r_n the first root of $C_n'(x)$, and α_n^* the value of C_n^{-1} closest to 0. In Fig. 2a, the values of γ_n, r_n and α_n^* are shown for $C_6(x)$ and in Fig. 2b, λ_n^* is shown for the polynomial $q^6(x) = C_6(l_6(x))$.

A consequence of this construction is that if for some degree n the lowest M' eigenvalues of A are contained in $[\lambda_{\min}, \lambda_n^*)$ with λ_n^* defined by (7) then the iteration (3) applied with a transformation defined by $p_v(x) = q^n(x)$ on a subspace of size M' will converge to a subspace consisting of the M' eigenvectors corresponding to those lowest M' eigenvalues. Thus, the degree $n \geqslant 2$ should be chosen so that $\lambda_j < \lambda_n^*$ for $j \leqslant M$. There will always be an n for which this inequality is true, since $q^2(x)$ is monotonically decreasing over the interval $[\lambda_{\min}, \lambda_{\max}]$. In addition, to obtain a rapidly convergent procedure, one should also choose, if possible, n sufficiently large so that $\lambda_i > \lambda_n^*$ for $i > M$. Fig. 2 illustrates the desired distribution of eigenvalues with respect to the parameters λ_n^*.

Unfortunately, the optimal choice of polynomial degree depends upon the values one is trying to compute, in particular, the values of $\lambda_j, j \leqslant M$. One has to be somewhat careful in choosing the degree, as too small a value of n results in only a modest acceleration of convergence, and too large an n leads to an iteration that can converge to eigenpairs for eigenvalues that are not the desired ones. To overcome this difficulty we adapt the polynomial degree based upon estimates of the eigenvalues that are extracted from the approximations at each step of the iterative process (3).

Specifically, at each step of the iteration (3) approximate eigenvalues θ_i^v for $i = 1 \ldots M + P$ are available. The extremal properties of the Rayleigh–Ritz approximations imply that $\lambda_i \leqslant \theta_{M+P}^v$ for $i = 1 \ldots M + P$, and thus the previous considerations lead one to the following procedure for selecting $p_v(x)$ in step *(e)*.

- For $v = 0$, choose $p_v(x) = q^2(x) = C_2(l_2(x))$
- For $v > 0$, choose $p_v(x) = q^n(x) = C_n(l_n(x))$ where the degree n is selected so that $\lambda_{n+1}^* < \theta_{M+P}^v \leqslant \lambda_n^*$ where the values of λ_n^* and λ_{n+1}^* are determined by (7).

If A has distinct eigenvalues and the initial vectors contain components in the directions of the desired eigenvectors, then as $v \to \infty$, S_v will converge to a subspace containing the eigenvectors corresponding to the lowest M eigenvectors. The need for the use of at least one buffer vector, e.g. $P \geqslant 1$, is to insure that there is a strict inequality $\lambda_M < \lambda_n^*$, so that the convergence

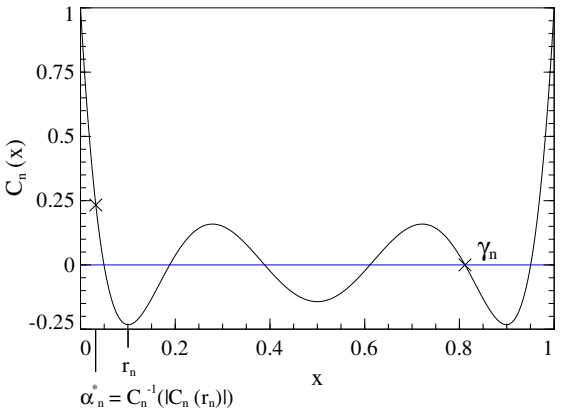

Fig. 2a. $C_n(x)$ with $n = 6$, a Chebyshev polynomial of the second kind scaled and shifted to $[0, 1]$.

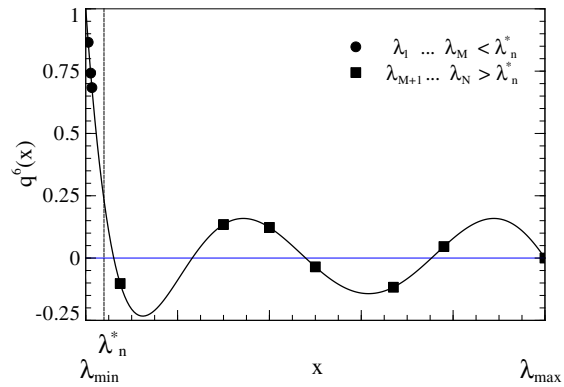

Fig. 2b. $q^6(x) = C_6(l_6(x))$ with the linear transformation $l_6(x)$ selected so $\left|\frac{q^6(\lambda_i)}{q^6(\lambda_j)}\right| < 1$ for $j \leqslant M$ and $i > M$.

factors $\left|\frac{p_v(\lambda_i)}{p_v(\lambda_j)}\right|$ for $j \leqslant M$ and $i > M$ at each step are strictly less than one. As the iteration (3) progresses, the degree of p_v increases as the accuracy with which θ^v_{M+P} approximates of λ_{M+P}. Except for the case to be noted below, this degree will be bounded by a value $n_{\max}$ for which $\lambda^*_{n_{\max}+1} < \lambda_{M+P} \leqslant \lambda^*_{n_{\max}}$. Thus, the asymptotic convergence behavior will be determined by the convergence factors $\left|\frac{q^{n_{\max}}(\lambda_i)}{q^{n_{\max}}(\lambda_j)}\right|$ for $j \leqslant M$ and $i > M$. Since the polynomial $q^{n_{\max}}(x)$ is not monotone for $\lambda_i > \lambda_{n_{\max}}$ it is difficult to determine a general bound for the convergence factors other than that they are all strictly less than one. However, as will be seen in the computational experiments, once the maximal degree is reached the convergence is very rapid, and observed convergence factors are often close to .22 (a value expected from the properties of the Chebyshev polynomials used in the procedure).

When A has eigenvalues of multiplicity greater than one, then the iteration will converge for the eigenpairs associated with all eigenvalues less than $\tilde{\lambda}_{M+P}$ where $\tilde{\lambda}_{M+P}$ is the value of the $(M+P)$th eigenvalue when they are counted with multiplicity. The most serious consequence associated with the convergence in the case of eigenvalues of multiplicity greater than one is that one may not identify a full set of eigenvectors associated with the largest eigenvalue. This problem can be detected and then overcome by either increasing the dimension of the approximating subspace or by utilizing a deflation procedure.

There is one problem that can arise in the determination of the polynomials, and that occurs when the dimension of the subspace used, $M+P$, is less than the multiplicity of the smallest eigenvalue. When this occurs, the polynomial degree that is determined will increase without bound and the convergence factors will tend to zero (e.g. the Rayleigh–Ritz approximation of the eigenvalues will converge with ever increasing speed). This problem can therefore be detected by estimating the convergence factors of the approximate eigenvalues, and if they are uniformly less than a specified factor (we use .2), then one stops increasing the degree of the polynomial used in the procedure.

3.2. Deflation and buffer dimension selection

When the dimension of the subspaces S_v is insufficient to compute all the desired eigenvector pairs at once, then one can add a deflation step to the procedure to compute additional eigenvectors. Specifically, if one uses a subspace of size $\widetilde{M}+P$ where $\widetilde{M} < M$ and $P \geqslant 1$ "buffer" vectors are used, then the iteration (3) converges to eigenpairs associated with the smallest $\widetilde{M}$ eigenvalues. After convergence, the procedure is restarted with a new value of $\lambda_{\min}$ being set equal to $\lambda_{\widetilde{M}}$ and the first P vectors of the approximating subspace being taken as the eigenvectors associated with the largest P eigenvalues of the previous iterates (e.g. the buffer vectors are used as initial guesses). At each step in the restarted application, the vectors in the subspace are orthogonalized with respect to the collection of known eigenvectors. This process computes the next $\widetilde{M}$ eigenpairs. If more eigenpairs are desired, the process is repeated as many times as necessary, each time computing the next $\widetilde{M}$ eigenpairs.

The use of at least one buffer vector, $P \geqslant 1$, is required to guarantee convergence of the iterative process, but one has the option of specifying a greater number of "buffer" vectors. Using a greater number of buffer vectors can improve the convergence behavior, and can also help avoid problems in the case of operators with eigenvalues of multiplicity greater than one. The improvement in convergence behavior follows because the convergence factors for first M eigenpairs is a function of the gap between the first M eigenvalues and those greater than $M+P$—increasing P can increase the gap size. The amount of improvement in convergence factors depends upon the specific eigenvalue distribution of the operator A and the properties of $p_v(\lambda_i)$, and as such, this buffer size is typically an experimentally determined parameter. The cost of using more "buffer" vectors is an increase in work associated with a larger dimension to the subspace S_v, and this must be weighed against any reduction in the number of iterations required. However, if one is computing more than just a few eigenpairs and deflation is used, the negative consequences of using a few buffer vectors is negligible as they become initial guess vectors when the deflation steps are incorporated.

With a deflation step included the Rayleigh–Chebyshev iteration has the following form:

Rayleigh–Chebyshev iteration with deflation

Given an initial collection, S_0, of $\widetilde{M}+P$ vectors ($P \geqslant 1$), an estimate of $\lambda_{\min}$ and $\lambda_{\max}$ so that $p_0(x) = q^2(x) = C_2(l_2(x))$ can be determined, and an initially empty collection of eigenvectors W, then the Rayleigh–Chebyshev procedure with deflation has the general structure

(a) Compute $V'_v = p_{v-1}(A)S_{v-1}$. (9)
(b) Compute $V_v =$ components of V'_v orthogonal to W.
(c) Orthonormalize $V_v = Q_v R_v$ by modified Gram–Schmidt.
(d) Form $H_v = Q_v^T A Q_v$.
(e) Diagonalize $H_v = G_v^T \Theta_v G_v$.
(f) Determine $p_v(x)$ from $\theta_{\widetilde{M}+P}$, $\lambda_{\min}$ and $\lambda_{\max}$. For $v = 0$, choose $p_v(x) = q_2(x) = C_2(l_2(x))$ and for $v > 0$, choose $p_v(x) = q_n(x) = C_n(l_n(x))$ where the degree n is selected so that $\lambda^*_{n+1} < \theta^v_{\widetilde{M}+P} \leqslant \lambda^*_n$ where the values of λ^*_n and λ^*_{n+1} are determined by (7).
(g) Form $S_v = Q_v G_v$ the Rayleigh–Ritz approximations to the eigenvectors and test for the convergence of approximate eigenvalues θ^v_i, $i = 1 \ldots \widetilde{M}$. If convergence has not been obtained, repeat iteration starting with (a) and $v = v+1$.
(h) Update the collection of computed eigenpairs. If the smallest $\widetilde{M}$ eigenvalues have converged, add the corresponding eigenvectors to W, set $\lambda_{\min} = \lambda_{\widetilde{M}+1}$, and reinitialize S_0 by setting the first P vectors of S_0 to correspond to the

largest P eigenvectors computed in (g) and fill out the remaining dimensions of $S_0$with random initial vectors orthogonal to those in W. Repeat iteration starting with (a), $v = 1$, and $p_0(x)$ determined from the new value of $\lambda_{\min}$.

This iteration is repeated until W contains the desired number of eigenvectors corresponding to the smallest eigenvalues.

4. Computational results

The first computational result concerns the test problem of Section 2, the computation of the lowest nine eigenvalues of the operator $(-\Delta - \cos(2\pi x))\ u(x,y) = \lambda u(x,y)$ where Δ is the two-dimensional Laplace operator, $(x,y) \in [0,1] \times [0,1]$ and $u(x,y)$ is periodic in both coordinate directions. Other than a stopping condition, the only free parameters of the procedure that must be selected are the size of the subspace to capture the eigenvectors and the number of buffer vectors to use. We used a single subspace of dimension 9 to capture the eigenvalues ($M = 9$) and chose the smallest number $P = 1$ of buffer vectors. The required eigenvalue estimates $\lambda_{\min}$ and $\lambda_{\max}$ for starting the computational process were calculated as the extremal eigenvalues of the tridiagonal matrix obtained from a classical Lanczos procedure. The Rayleigh–Chebyshev iteration was stopped when the relative difference between eigenvalue estimates was uniformly less than 1×10^{-5}. This value was selected because it resulted in eigenvalues with relative errors that were less than 2.5×10^{-6}—the size of the errors in the eigenvalues obtained with the inverse subspace iteration procedure described in Section 2. Each iteration of the Rayleigh–Chebyshev procedure requires the application of $p_v(A)$ to a collection of vectors. The application of this polynomial was implemented using a two-term recurrence based on the recurrence that the Chebyshev polynomials satisfy (5).

16 iterations were required obtain a relative error of the eigenvalues to be less than 2.5×10^{-6}. This compares quite favorably with the 10 iterations that was the smallest number required by inverse subspace iteration with shift. The total number of applications of the operator A (including the cost of the extremal eigenvalue estimation) was 4115, significantly less than 17616, the best results presented in Table 1. The number of applications of the operator A required to obtain the initial extremal eigenvalue estimates was 165—only 4% of the total computational cost of the procedure.

The behavior of the iterative procedure is revealed by considering the degree of the polynomial p_v used at a given step, and the reduction in the relative difference between eigenvalue estimates between steps, a value that was monitored to assess convergence. These quantities are plotted in Figs. 3a and 3b. During the early stages of the iteration, the eigenvalue estimates do not improve very much, but work is being accomplished as the polynomial degree is rapidly increasing to its optimal value. Once the polynomial degree reaches its optimal value then convergence of the eigenvalue estimates occurs quickly – in about as many iterations as are required by an inverse subspace iteration with optimal shift. An estimate of the dominant convergence factor based upon the data at later steps in Fig. 3b is 0.16—thus the procedure converges with an error that is approximately proportional to $(.16)^v$.

One of the advantages of the Rayliegh–Chebyshev method is that creating a distributed (parallel) implementation is rather straight forward. The core computational tasks are those associated with applying a polynomial in a linear operator to a collection of vectors, orthogonalizing a set of vectors using a modified Gram–Schmidt process and accumulating a collection of vectors to form the desired eigenvectors. These are standard computational linear algebra tasks and one can make use of libraries that provide such functionality and are optimized for distributed processor machines. Additionally, the problem of load balancing that arises when iterative methods are used with inverse iteration does not arise here. The polynomial in the operator that is applied to the approximating subspace is known in advance of its application and is the same for all vectors in the subspace.

Our interest in a distributed implementation of the Rayleigh–Chebyshev method arose out of a need to determine the lowest eigenvalues and eigenvectors of the linear operator arising from a Full Configuration Interaction treatment of the

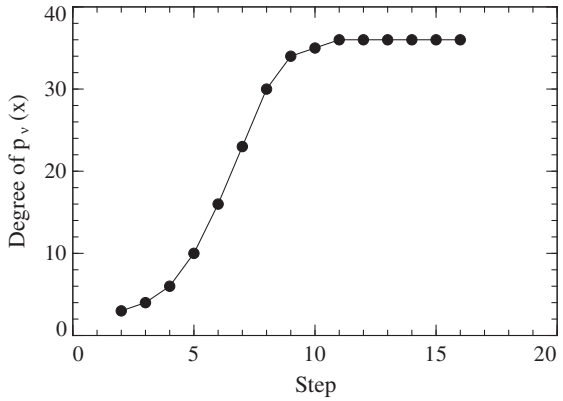

Fig. 3a. Chebyshev polynomial degree selected at each subspace iteration step for the test problem of Section 2.

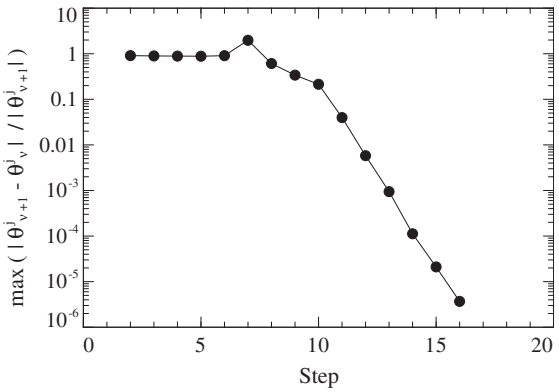

Fig. 3b. Maximal relative difference between computed eigenvalues at each subspace iteration step for the test problem of Section 2.

multi-particle Schroedinger operator. The dimension of the associated matrix was approximately $10^6 \times 10^6$ and contained 3.7×10^9 non-zero elements. The large memory requirements associated with computing and storing the matrix entries dictated that a distributed (parallel) implementation of the procedure be used.

In our implementation we took advantage of the PETSc library [4] which contains support for MPI based distributed vectors and sparse distributed matrices. The linear operator was represented as a PETSc sparse distributed matrix with an equal number of contiguous rows of the matrix assigned to each processor. For our particular operator, this distribution resulted in a nearly equal number of non-zero elements on each processor. The PETSc routines for distributed vector operations and for applying a distributed sparse matrix to a distributed vector were used and so "low level" MPI programming was kept to a minimum. As with the previous example, the estimates of $\lambda_{\min}$ and $\lambda_{\max}$ were computed as the extremal eigenvalues of the tridiagonal matrix arising from a classical Lanczos procedure. The distributed implementation of the latter was also easily accomplished using the PETSc library.

The smallest five eigenvalues and eigenvectors were computed and the iteration was stopped when the relative difference between successive eigenvalue estimates was less than 0.5×10^{-6}. The degree of the polynomial used at each step and the maximal relative difference between eigenvalues at each step are presented in Figs. 4a and 4b. These results show the same qualitative behavior of the iteration as in the previous example—in the early stages there is not much improvement, but the degree of p_v increases. Once this polynomial has reached it's maximal degree, then the iteration converges quite rapidly. Unlike the first test problem, where the relative gaps between distinct eigenvalues was on the order of 10^{-7} the relative gaps between eigenvalues for this problem were on the order of 10^{-2}. This more favorable eigenvalue distribution explains the lower degree of the polynomial p_v determined by the iteration. An estimate of the dominant convergence factor based upon the data in the later steps in Fig. 4b is 0.27, which explains why more Rayleigh–Chebyshev steps were taken, 18, than in the previous example. However, the total number of operator applications, 788, was much lower (with 50 of those applications being required by the Lanczos estimation procedure).

While our principle interest in the use of a distributed implementation of the Rayleigh–Chebyshev procedure was to enable computations that had large memory requirements, we were also interested in the impact that using multiple CPU's would have on the total computation time. These computations were carried out on a cluster consisting of nine dual

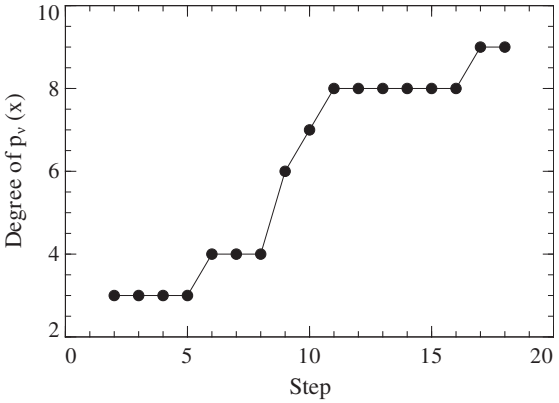

Fig. 4a. Chebyshev polynomial degree selected at each subspace iteration step for a Full Configuration Interaction problem.

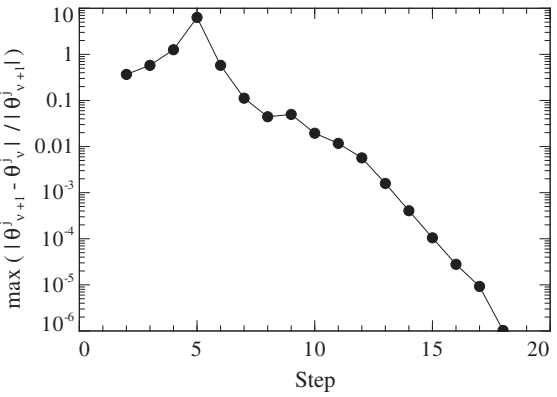

Fig. 4b. Maximal relative difference between computed eigenvalues at each subspace iteration step for a Full Configuration Interaction problem.

processor nodes with Intel Xeon CPUs running at 3.40 GHz and 8 GB of memory per node. We were thus able to run the same computation (one whose total memory requirements just fit within the 72 GB limit) using either 9 or 18 processors. The computation time when using 9 processors was 40.01 min and 27.7 min when using 18 processors. Since the parallel implementation was just a serial implementation modified to use PETSc operations and no specific algorithmic changes were made to enhance parallel performance these results, we find these results quite encouraging. We expect that with an implementation that specifically targets distributed architecture machines, one would be able to improve the parallel efficiency.

5. Conclusion

The paper is concerned with the task of computing a modest number of the smallest eigenvalues and associated eigenvectors of Hermitian operators. It is particularly concerned with this task for operators that have high dimension, are sparse, and may possess eigenvalues with multiplicity greater than one. A procedure commonly used to accomplish this task is inverse subspace iteration with shift, and results have been shown that indicate the typical problems that occur when a standard Krylov subspace method is employed to solve the requisite systems of linear equations at each step. Since such a procedure is ultimately just an iterative process where an approximating subspace is transformed by a polynomial in the operator, one concludes that the problem is occurring because the implicit determination of the polynomial transformation by the Krylov subspace solution procedure is yielding ineffective transformations.

The method presented to overcome such problems retains the idea of using polynomial transformations of an approximating subspace but adaptively determines the polynomial transformation using information from the Rayleigh–Ritz approximation of the eigenvalues. The adaptation is done in such a way that, under mild conditions on the properties of the distribution of eigenvalues of A and the choice of initial subspace, the overall iterative procedure will be guaranteed to converge to a subspace containing the desired eigenvectors.

The resulting method has several desirable properties. First, it can be implemented as a minor modification of inverse subspace iteration. One need only replace a call to a routine that solves the requisite linear systems with one that applies a polynomial in the operator times a collection of vectors. The method is robust, converges with acceptably rapidity, and can easily handle operators with eigenvalues of multiplicity greater than one. The method is well suited for implementation on multi-processor machines since the polynomial transformation of the subspace required at each step of the iteration is known in advance of the application of the transformation, and the transformation is identical for all vectors in the subspace.

There is an initial cost associated with the procedure, that of obtaining estimates of the minimal and maximal eigenvalues. However, as the computational results show, using the classical Lanczos procedure to obtain these is quite effective, and does not contribute significantly to the overall cost of the procedure. The numerical results presented here, as well as results presented in [1] demonstrate that the procedure scales well to very large eigenvalue problems, and good success has been obtained with its use on eigenproblems related to the single particle Schroedinger operators in three dimensions as well as the very large eigensystem problems occurring in an FCI treatment of the multi-particle Schroedinger operator.

Acknowledgments

Research sponsored by DARPA through the Quantum Information Science and Technology (QuIST) program under Army Research Contract number DAAD-19–01-C-0077 and by United States Department of Defense. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressly or implied, of the U.S. Government.

References

[1] C. Anderson, Efficient solution of the Schroedinger–Poisson equations in layered semiconductor devices, Journal of Computational Physics 228 (2009) 4745–4756.
[2] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J.D. Croz, A. Greenbaum, S. Hammarling, A. McKenney, D. Sorensen, LAPACK Users' Guide, third ed., Society for Industrial and Applied Mathematics, Philadelphia, PA, 1999.
[3] J. Baglama, D. Calvetti, L. Reichel, Iterative methods for the computation of a few eigenvalues of a large symmetric matrix, BIT Numerical Mathematics 36 (3) (1996) 400–421.
[4] S. Balay, K. Buschelman, V. Eijkhout, W. Gropp, D. Kaushik, M. Knepley, L. McInnes, B. Smith, H. Zhang, Petsc users manual, Argonne National Laboratory, Technical Report ANL-95/11-Revision, 2(5), 2004.
[5] J. Berns-Muller, I. Graham, A. Spence, Inexact inverse iteration for symmetric matrices, Linear Algebra and its Applications 416 (2–3) (2006) 389–413.
[6] L. Borges, S. Oliveira, A parallel Davidson-type algorithm for several eigenvalues, Journal of Computational Physics 144 (2) (1998) 727–748.
[7] J. Cullum, R. Willoughby, Lanczos Algorithms for Large Symmetric Eigenvalue Computations: Programs, Birkhauser, Boston, 1985. Available online at <http://www.netlib.org/lanczos>. URL: <http://myfiles>.
[8] J. Cullum, R. Willoughby, Lanczos Algorithms for Large Symmetric Eigenvalue Computations: Theory, Society for Industrial Mathematics, 2002.
[9] E. Davidson, Note the iterative calculation of a few of the lowest eigenvalues and corresponding eigenvectors of large real-symmetric matrices, Journal of Computational Physics 17 (1975) 87–94.
[10] G. Golub, C. Van Loan, Matrix Computations, Johns Hopkins University Press, 1996.
[11] N. Kosugi, Modification of the Liu–Davidson method for obtaining one or simultaneously several eigensolutions of a large real-symmetric matrix, Journal of Computational Physics 55 (1984) 426.
[12] C. Lanczos, Applied Analysis, Dover Publications, 1988.
[13] R. Morgan, D. Scott, Generalizations of Davidson's method for computing eigenvalues of sparse symmetric matrices, SIAM Journal on Scientific and Statistical Computing 7 (3) (1986) 817–825.
[14] B. Parlett, The symmetric eigenvalue problem, Classics in Applied Mathematics, vol. 20, SIAM, Philadelphia, 1998.
[15] B. Parlett, D. Scott, The Lanczos algorithm with selective orthogonalization, Mathematics of Computation 33 (145) (1979) 217–238.
[16] B. Parlett, D. Taylor, Z. Liu, A look-ahead Lanczos algorithm for unsymmetric matrices, Mathematics of Computation 44 (169) (1985) 105–124.
[17] M. Robbé, M. Sadkane, A. Spence, Inexact inverse subspace iteration with preconditioning applied to non-Hermitian eigenvalue problems, SIAM Journal on Matrix Analysis and Applications 31 (2009) 92.
[18] A. Ruhe, T. Wiberg, The method of conjugate gradients used in inverse iteration, BIT Numerical Mathematics 12 (4) (1972) 543–554.
[19] Y. Saad, Numerical Methods for Large Eigenvalue Problems, Manchester University Press, 1992. Available at <http://www.cs.umn.edu/~saad/books.html>
[20] V. Simoncini, L. Eldén, Inexact Rayleigh quotient-type methods for eigenvalue computations, BIT Numerical Mathematics 42 (1) (2002) 159–182.
[21] G. Sleijpen, H. Van der Vorst, A Jacobi–Davidson iteration for linear eigenvalue problems, SIAM Journal on Matrix Analysis and Applications 17 (1996) 401–425.
[22] G. Sleijpen, H. van der Vorst, E. Meijerink, Efficient expansion of subspaces in the Jacobi–Davidson method for standard and generalized eigenproblems, Electronic Transactions on Numerical Analysis 7 (1998) 75–89.
[23] K. Wu, H. Simon, Thick-restart Lanczos method for large symmetric eigenvalue problems, SIAM Journal on Matrix Analysis and Applications 22 (2) (2000) 602–616. <http://link.aip.org/link/?SML/22/602/1>.
[24] Y. Zhou, Y. Saad, A Chebyshev–Davidson algorithm for large symmetric eigenproblems, SIAM Journal on Matrix Analysis and Applications 29 (3) (2008) 954–971.