Robust Preconditioning of Large, Sparse, Symmetric Eigenvalue Problems

Andreas Stathopoulos [*], Yousef Saad [†], Charlotte. F. Fischer [*]

September 5, 1994

Abstract

Iterative methods for solving large, sparse, symmetric eigenvalue problems often encounter convergence difficulties because of ill-conditioning. The Generalized Davidson method is a well known technique which uses eigenvalue preconditioning to surmount these difficulties. Preconditioning the eigenvalue problem entails more subtleties than for linear systems. In addition, the use of an accurate conventional preconditioner (i.e., as used in linear systems) may cause deterioration of convergence or convergence to the wrong eigenvalue. The purpose of this paper is to assess the quality of eigenvalue preconditioning and to propose strategies to improve robustness. Numerical experiments for some ill-conditioned cases confirm the robustness of the approach.

Keywords: Symmetric, sparse matrix, eigenvalue, eigenvector, ill-conditioned eigenvectors, iterative methods, preconditioning, generalized Davidson method, spectrum compression, inverse iteration.

1 Introduction

The solution of the eigenvalue problem, $Ax = \lambda x$, is central to many scientific applications. In these applications it is common for A to be real, symmetric, and frequently very large and sparse [17, 7]. Advances in technology allow scientists to continuously tackle larger problems for which only a few lowest or highest eigenpairs are required.

Standard eigenvalue methods that transform the matrix and find the whole spectrum [9, 6], are inadequate for these applications because of the size of the problem. As a result, numerous iterative methods have been developed that are based only on small, low level kernels such as matrix-vector multiplication, inner products and vector updates, that do not modify the matrix, and that find only the required extreme eigenpairs. The simplicity and efficiency of these methods accounts for their extensive use.

*Computer Science Department, Vanderbilt University, Nashville, TN.

†Computer Science Department, University of Minnesota

Preconditioning has been recognized as a powerful technique for improving the convergence of iterative methods for solving linear systems of equations [11, 22]. The original matrix A is multiplied by the inverse of a preconditioning matrix M which is close to A in some sense. This has the effect of bringing the condition number of the preconditioned matrix closer to 1, thereby increasing the convergence rate [9, 14]. Applying preconditioning to the eigenvalue problem is not as obvious for two reasons: the separation gap of a specific eigenvalue is important rather than the condition number, and when the equation $Ax = \lambda x$ is multiplied with a matrix M, it becomes a generalized eigenvalue problem. Preconditioning can be applied indirectly to the eigenvalue problem by using the Preconditioned Conjugate Gradient(PCG) or similar iterative methods to solve the linear systems appearing in each step of inverse iteration, Rayleigh quotient iteration [18, 27] or shift-and-invert Lanczos [24].

The Davidson and Generalized Davidson (GD) methods [2, 12], provide a more direct approach to eigenvalue preconditioning. The original method is a subcase of GD and was introduced for electronic structure computations. Lately, GD has gained popularity as a general iterative eigenvalue solver [1, 25]. The method is similar to the Lanczos method in that it builds the basis of an orthogonal subspace from which the required eigenvectors are approximated through a Rayleigh-Ritz procedure. However, the GD method solves the Rayleigh-Ritz procedure in each step and the residual of the current approximation is preconditioned ($(M-\lambda I)^{-1}(A-\lambda I)x$) before it enters the basis. Therefore, the subspace built deviates from the Krylov subspace, $\mathcal{K}(A,g,m) = \text{span}\{g, Ag, \ldots, A^m g\}$, which is obtained from the traditional Lanczos iteration. There has been some effort in the literature to take advantage of the fact that λ (the Rayleigh Quotient) is nearly constant in later iterations [18, 19], by using the Lanczos procedure to build the Krylov space of the the matrix $(M-\lambda I)^{-1}(A-\lambda I)$ [13]. This approach uses an inner-outer iteration and reduces the higher computational costs of the GD step. However, the number of matrix-vector multiplications is not reduced in general.

The use of an accurate preconditioner for the eigenvalue problem does not necessarily ensure fast and accurate convergence. In the following, the quality of eigenvalue preconditioning is assessed and a more robust approach is proposed. In section 2, a general implementation of the GD method is outlined, which allows for user-provided matrix vector multiplication and for flexible use of preconditioners between iterations. In section 3, the effectiveness of various preconditioning approaches to the eigenvalue problem is discussed. Convergence problems specific to eigenvalue calculations are identified, and a modification to GD that improves robustness for preconditioning is proposed. In section 4, results are presented from several preconditioners applied on matrices from Harwell-Boeing collection [5], and from atomic physics calculations and comparisons of GD with the modified version are performed. The paper concludes with some final remarks.

2 The Generalized Davidson Method

Davidson proposed a way of using the diagonal of A to precondition the Lanczos process [2]. In electronic structure calculations where the eigenvectors have only a few dominant components this choice is sufficient to provide extremely rapid convergence. In the general case, clustering of eigenvalues may cause the eigenvector problem to be ill-conditioned [29], requiring the use of very good preconditioners to even achieve convergence. The GD method extends the above choice to a matrix M which can approximate A better. In the iterative procedure, the next basis vector is chosen as the correction vector δ to the current approximate eigenvector x, as given by $\delta = (M - \lambda I)^{-1} Res(x)$, where λ is very close to the eigenvalue and $Res(x)$ is the residual of x. The computational costs of the GD step are much larger than with the Lanczos algorithm but the iterations can be dramatically reduced [1, 12, 10, 25].

Let the K lowest eigenpairs $(\lambda_i, x_i), i = 1, \ldots, K$, be required. If $(\tilde{\lambda}_i, \tilde{x}_i)$ denote the current approximate eigenpairs, a brief description of the algorithm follows:

The Algorithm

Step 0: Set $m = K$. Compute Basis $B = \{b_1, \ldots, b_m\} \in \Re^{N \times m}$ from initial guesses, also $D = AB = \{d_1, \ldots, d_m\}$, and the projection of size $m \times m$, $S = B^T AB = B^T D$.

Repeat until converged steps 1 through 8:

1. Solve $SC = C\Lambda$, with $C^T C = I$, and $\Lambda = \mathrm{diag}(\tilde{\lambda}_i)$.
2. Target one of the K sought eigenpairs, say $(\tilde{\lambda}, c)$.
3. If the basis size is maximum truncate:
   $D \leftarrow DC,\ B \leftarrow BC,\ C = I_K,\ S = \Lambda,\ m = K.$
4. Compute $\delta = (M - \tilde{\lambda} I)^{-1}(Dc - \tilde{\lambda} Bc)$.
5. Orthogonalize: $b_{new} = \delta - \sum b_i b_i^T \delta$, normalize: $b_{new} \leftarrow b_{new}/\|b_{new}\|$.
6. Matrix-vector multiply: $d_{new} = Ab_{new}$
7. Compute the new column of S: $S_{i,m+1} = b_i^T d_{new},\ i = 1, \ldots, m+1$.
8. Increase m.

There are many extensions that improve the functionality and the run-time behavior of the above algorithm, most of which are implemented in the routine DVDSON described in [25]. The matrix vector multiplication is provided by the user as an external routine. However, DVDSON in [25] has the following "limitations": It requires at least K initial estimates, the diagonal of the matrix has to be given, the matrix needs to be in a COMMON block to be accessed by the matrix-vector multiplication, and the preconditioner chosen is simply the diagonal of the matrix. While these limitations are transparent to electronic structure calculations applied on serial computers, they can be a drawback in more difficult problems where better preconditioners are needed that may vary between successive steps and where complicated data structures may be used on parallel computers. To improve the flexibility of the DVDSON code the following changes have been made:

1. If only $(\tilde{x}_1, \ldots, \tilde{x}_{K_1})$, $K_1 < K$, initial estimates are available, the algorithm builds the starting basis by computing the first $(K - K_1)$ orthogonal Krylov subspace vectors of the $\tilde{x}_i$, $i = 1, \ldots, K_1$. This is especially useful when no good initial estimates are known.

2. In electronic structure calculations initial estimates can be obtained from the diagonal of the matrix [3]. An optional preprocessing routine is provided for this reason. In this way functionality and generality of the code are both maintained.

3. To avoid the use of COMMON blocks and the adherence on one preconditioner the *reverse communication* mechanism is adopted [22]. Whenever a matrix-vector multiplication or a preconditioning operation is required, the DVDSON routine is exited and the calling routine performs the appropriate operation. After the operation has concluded the result is placed in one of the DVDSON's arguments and DVDSON is called again. Some information parameter is used for the calling routine to be able to distinguish between the different required operations. This information parameter may be an index array carrying additional information about the procedure's status. A typical use of this mechanism with the DVDSON routine is as follows:

```
      irev(1) = 0
 100  continue
      call dvdson(irev,n,work,...)
      inp  = irev(2)
      outp = irev(3)
      ieval= irev(4)
      ires = irev(5)
      if (irev(1).eq.1) then
          eigval = work( ieval )
          residual = work( ires )       <-- To determine preconditioner
          call precond(n,eigval,residual,
                       work( inp ),work( outp ),...) <-- User provided
          goto 100
      else if (irev(1).eq.2) then
          call matvec(n,work( inp ),work( outp ),...) <-- User provided
          goto 100
      endif
```

   In the current implementation, the array IREV holds information about how many vectors are to be multiplied or preconditioned in the block version of DVDSON, the location of the input and output vectors in the work array, the corresponding eigenvalue approximations to be used by the preconditioning and the residual norm estimates for choosing an appropriate preconditioner. More information can also be provided in IREV to meet the users' needs.

The above modifications have significantly enhanced the flexibility of the procedure. The resulting code can be used unchanged with any matrix format, multiplication and preconditioning routine or in a parallel environment. Moreover, the user is able to control the initial estimates and embed intelligence in the choice of preconditioner.

3 Preconditioning the Eigenvalue Problem

As mentioned earlier the obvious way to precondition the eigenvalue problem is to use PCG to solve the linear systems arising in inverse or Rayleigh quotient iteration. This scheme is less likely to be as efficient and robust as the GD method mainly because of the subspace character of the latter one [13]. Sophisticated PCG implementations however may bridge that gap. When the matrix A is multiplied by a matrix M the problem becomes a generalized one: $MAx = \lambda Mx$. However, when M has the same eigenvectors as A and eigenvalues μ_i, the matrix MA has also the same eigenvectors and eigenvalues $\lambda_i \mu_i$. Theoretically one could pick a M that yields a desired arrangement of eigenvalues but this assumes the knowledge of the eigenvectors. As in linear systems the search for M is restricted to approximate inverses of A.

The main goal of preconditioning in eigenvalue problems is the compression of the spectrum away from the required eigenvalue rather than away from the origin as in linear systems. The convergence of eigenvalue iterative methods depends on the gap ratios, i.e., relative ratios of eigenvalues that indicate how well separated the required eigenvalue is from the rest of the spectrum [18, 29]. If for example $M \approx A^{-1}$ is used all the eigenvalues tend to be compressed around 1 and the convergence of iterative methods deteriorates. In the extreme case where $M = A^{-1}$, and A^{-1} exists, the system becomes $Ix = x$ and all information about A is lost.

To isolate a specific eigenvalue of A, e.g., λ_k, the operator

$$(M - \tilde{\lambda} I)^{-1}(A - \tilde{\lambda} I) \tag{1}$$

may be used, where $\tilde{\lambda}$ is any close approximation to λ_k. The advantage of this over MA is that $(A - \tilde{\lambda} I)$ approaches singularity as $\tilde{\lambda} \to \lambda_k$, and if $M \approx A$ the rest of the eigenvalues are compressed much more than λ_k is. Also the eigenvectors approximate those of A [12]. There are a few points that need closer attention.

i. In the extreme case where $M = A$, the operator (1) becomes the identity operator and the information is lost. However, the preconditioner $(M - \tilde{\lambda} I)^{-1}$ depends on two variables, M and $\tilde{\lambda}$. To avoid the above cancellation, the $\tilde{\lambda}$ may be fixed to some value $s < \lambda_k$ which is close to the desired eigenvalue but farther from the current approximation. The new operator $(A - sI)^{-1}(A - \tilde{\lambda} I)$ still carries the eigensystem information and has eigenvalues $(\lambda_i - \tilde{\lambda})/(\lambda_i - s)$ which are very close to one except $(\lambda_k - \tilde{\lambda})/(\lambda_k - s)$ which is close to zero.

ii. When the current approximation $\tilde{\lambda} \approx \lambda_i, i \neq k$, the preconditioner is likely to give a good separation to the wrong eigenpair (similar to inverse iteration). The convergence of the new operator to λ_k may be slow or even not guaranteed.

iii. The matrix (1) is nonsymmetric in general, and it may have complex eigenvalues if $(M - \tilde{\lambda} I)^{-1}$ is not positive definite [14]. This disables the direct use of the Lanczos method on matrix (1). It also impairs the selective convergence to interior eigenpairs inherited by the inverse-iteration-like behavior, when $\tilde{\lambda}$ is chosen below the spectrum.

As was already mentioned, some inner-outer iterative schemes have been developed for the lowest/highest eigenpair. The matrix (1) is updated after a few Lanczos steps with the latest Rayleigh quotient as $\tilde{\lambda}$. In early iterations $\tilde{\lambda}$ may be far from λ_k, and the total number of matrix-vector multiplications is usually increased with this scheme.

3.1 The Generalized Davidson approach

The GD method does not use the fixed operator (1), but it varies it between iterations as the approximation $\tilde{\lambda}$ varies. The resulting algorithm can be derived either from perturbation theory or the Newton's method [3, 4, 15]. If E is a matrix such that $A = M + E$, and δ and ϵ are the corrections to the approximate eigenvector $\tilde{x}$ and eigenvalue $\tilde{\lambda}$ correspondingly, then:

$$(M + E)(\tilde{x} + \delta) = (\tilde{\lambda} + \epsilon)(\tilde{x} + \delta) \tag{2}$$
$$\Leftrightarrow \quad (M - \tilde{\lambda})\delta - \epsilon\delta + E\delta = \epsilon\tilde{x} - (A - \tilde{\lambda} I)\tilde{x}. \tag{3}$$

If the quadratic correction terms are omitted,

$$(M - \tilde{\lambda} I)\delta = \epsilon\tilde{x} - (A - \tilde{\lambda} I)\tilde{x}. \tag{4}$$

Since $|\epsilon| = O(\|Res(x)\|^2)$ near convergence [18], the following familiar equation is used to derive δ:

$$(M - \tilde{\lambda})\delta = Res(\tilde{x}) = (A - \tilde{\lambda} I)\tilde{x}. \tag{5}$$

The sign is dropped because only the direction is of interest.

GD circumvents the requirement for positive definiteness of $(M - \tilde{\lambda} I)^{-1}$ since it does not iterate with matrix (1) but with A. Preconditioning for interior eigenvalues is thus possible, and selective convergence can be achieved without any compromise on how close $\tilde{\lambda}$ can be to λ_k. Moreover, the minimization of the Rayleigh quotient and the approximate eigenvectors are computed based on the exact eigensystem of A, rather than an approximate one, and the total number of matrix-vector multiplications is usually reduced. The problems in point iii are thus removed but problems can still arise from points i and ii. Expressed for the GD method, the problems may stem from the following:

P1. $(M - \tilde{\lambda} I)$ is very close to $(A - \tilde{\lambda} I)$ and the resulting δ is almost identical to $\tilde{x}$. If $\tilde{\lambda}$ is a good approximation to λ_k, the inverse iteration behavior may provide δ with some new information but this is usually limited.

P2. $(M - \tilde{\lambda} I)$ is very close to $(A - \tilde{\lambda} I)$, and $\tilde{\lambda} \approx \lambda_i, i \neq k$. In this situation convergence can be extremely slow or erroneous, i.e., towards an undesired eigenvalue.

Both of these problems are related to preconditioners that approximate $(A - \tilde{\lambda} I)^{-1}$ accurately. This is in contrast to the linear systems experience, where accurate preconditioners yield good convergence. Moreover, in case of **P2** the inverse iteration amplification of the components of the wrong eigenvector may cause erroneous convergence. These problems are clearly demonstrated in the results appearing in section 4. A poor preconditioner as the diagonal must be applied for several iterations before the change to an accurate one occurs. This is necessary until the approximation $\tilde{\lambda}$ escapes from all $\lambda_i, i \neq k$. However, the decision on the quality of the poor preconditioner is arbitrary, and an optimum choice may require sophisticated heuristics. Evidently, problems **P1** and **P2** must also be solved along with finding accurate conventional preconditioners.

3.2 Modifying GD for Robustness

Problem **P1** can be handled by solving equation (5) with a right hand side different from $Res(\tilde{x})$. Olsen at al. [15] proposed the use of equation (4) instead of (5), where the right hand side is augmented by the term $\epsilon\tilde{x}$. Recall that ϵ is the correction to the eigenvalue. If the eigenvector correction δ is forced to be orthogonal to $\tilde{x}$, ϵ can determined as (denoted ϵ_o):

$$\epsilon_o = \frac{\tilde{x}^T(M-\tilde{\lambda}I)^{-1}(A-\tilde{\lambda}I)\tilde{x}}{\tilde{x}^T(M-\tilde{\lambda}I)^{-1}\tilde{x}}. \quad (6)$$

In the extreme case when $(M-\tilde{\lambda}I) = (A-\tilde{\lambda}I)$, equation (4) yields $\delta = \epsilon_o(A-\tilde{\lambda}I)^{-1}\tilde{x} - \tilde{x}$. Since δ is made orthogonal to $\tilde{x}$, this method performs one step of inverse iteration, having the potential to provide new information. More precisely, $\tilde{\lambda}$ is the current Rayleigh quotient in GD, and the above iteration becomes equivalent to the Rayleigh quotient iteration with the favorable asymptotical cubic convergence rate [18].

Problem **P2** focuses precisely on what constitutes a "good" eigenvalue preconditioner. Unlike in linear systems, a "good" preconditioner should not yield $(M-\tilde{\lambda}I)$ very close to $(A-\tilde{\lambda}I)$, but to $(A-\lambda_k I)$ instead. In this way, when $\tilde{\lambda}$ is accurate the desired spectrum compression is achieved and when it is far from λ_k, the scheme acts as inverse iteration. In the numerical experiments of the first part of section 4, the preconditioners try to approximate $(A-\tilde{\lambda}I)$ causing the expected convergence problems.

Solving problem **P2** is more difficult since the best known approximation to λ_k is $\tilde{\lambda}$. To achieve convergence in the test cases of section 4, flexible preconditioning is employed. There are two disadvantages with this approach: it relaxes the power of the preconditioner rather than adjusting it, and it requires external information for tuning the parameters for the proper timing of preconditioner switch.

Adjusting the preconditioner involves finding ϵ, the correction to $\tilde{\lambda}$, and using it in the shift: $(M-(\tilde{\lambda}+\epsilon)I)$. The method becomes obvious if instead of solving equation (4) as in Olsen's method, equation (3) is solved, by letting the term $E\delta = 0$. The latter is further justified by the assumption of an accurate preconditioner. Thus, the preconditioning step in GD consists of solving approximately the equation:

$$(M-(\tilde{\lambda}+\epsilon)I)\delta = \epsilon\tilde{x} - (A-\tilde{\lambda}I)\tilde{x}. \quad (7)$$

If ϵ can be estimated accurately and a good conventional preconditioner is used, the above modification solves both problems **P1** and **P2**, providing a "good" eigenvalue preconditioner.

Another explanation of this result is as follows. From the minimization of the Rayleigh quotient in the Rayleigh Ritz procedure, and from the Courant-Fischer theorem [29, 18], the approximations $\tilde{\lambda}$ decrease monotonically in each step. When an accurate preconditioner is used near the wrong eigenvalue λ_i, the decrease in $\tilde{\lambda}$ is diminished because successive subspaces improve slightly and only in the direction of x_i. What equation (7) attempts to do is shift the eigenvalue to a lower value to prevent this halt. Notice that in this sense it is not vital that the exact correction is known. A large enough ϵ is needed to "pull" the eigenvalue below the halting level. When $\tilde{\lambda}$ is close to λ_k, it has been shown that the algorithm does not require a particularly accurate ϵ [12].

The explanation suggests that it is not necessary to know λ_k to obtain a "good" preconditioner. A robust preconditioner that approaches the convergence properties of the exact $(M - \lambda_k I)$ can be derived by using either of the following estimations of ϵ:

E1. ϵ_o as obtained from equation (6).

E2. $\Delta\lambda \equiv \Delta\lambda^{(j)} = \tilde{\lambda}^{(j)} - \tilde{\lambda}^{(j-1)}$, where j is the current iteration. Because of the monotonic eigenvalue convergence, $\Delta\lambda < \epsilon^{(j-1)}$, so $\Delta\lambda$ is an underestimation of the exact correction of the previous iteration. Experiments have shown that this is very similar to ϵ_o, but it is cheaper to compute.

E3. $\begin{cases} \|Res(\tilde{x})\|, & \text{if } \|Res(\tilde{x})\| \geq \gamma \\ \|Res(\tilde{x})\|^2/\gamma, & \text{if } \|Res(\tilde{x})\| < \gamma \end{cases}$, where γ is the gap ratio of the eigenvalue λ_k. This is suggested by the a posteriori bounds for eigenvalues [18]: $|\epsilon| < \|Res(\tilde{x})\|$, $\forall \tilde{x}$, and near convergence $|\epsilon| < \|Res(\tilde{x})\|^2/\gamma$. This is an overestimation of the exact correction of the current iteration. γ is easily approximated from the Ritz values.

Combinations of the above choices may also be considered. For example, when $\tilde{\lambda}$ is far from λ_k **E3** may be chosen for the left hand side ϵ of equation (7), while either of **E1** or **E2** for the right hand side ϵ. Experiments in section 4, illustrate the robustness of the modified approach, for some difficult problems. It is interesting to note that when a subspace method is adopted in RQI (to ensure monotonic eigenvalue convergence), the above results are also applicable to the choice of PCG preconditioner by choosing a better shift than the Rayleigh quotient.

Several options exist for the conventional preconditioner. Taking $M = Diag(A)$ has been extensively used in the literature. In [12, 1] the three main diagonals of the matrix are used as M. In [13] Incomplete LU factorization is used while other possibilities are mentioned. The study of linear systems in the last two decades has made available a big variety of preconditioners. Jacobi, SOR, SSOR and their block counterparts, ILU, ILUT, multigrid are only a few examples [22]. Next, the performance of Jacobi, SOR, band diagonal LU, and ILUT is examined as well as in combination with some accelerator [21]. Through these preconditioners the two problems are identified and the improvements from the above modification are demonstrated.

4 Numerical Experiments

4.1 Test cases

The diagonal scaling (Jacobi preconditioning) is the simplest way to precondition a matrix. It is also the least effective one, unless the matrix has large diagonal-dominance ratio — the term is used to refer to the ratio $d = \min_{i,j} |(A_{ii} - A_{jj})/A_{ij}|$ — i.e., very small off-diagonal elements compared with the changes in magnitude between diagonal elements [10, 12, 26]. Matrices for ground state systems in electronic structure calculations share this property[7, 3].

A natural extension to the Jacobi preconditioner is the LU decomposition of a few main diagonals of the matrix. This is expected to be effective when the important elements of the matrix are clustered around the main diagonal. Such matrices are encountered when an operator is approximated by minimal support basis functions (such as B-Splines [8]). The cost of band LU factorization is considerably higher than diagonal scaling but it still increases linearly with the size of the matrix.

Successive Overrelaxation (SOR) is the most popular relaxation scheme and it is extensively used as a preconditioner in solving linear systems. Because of its use as a preconditioner the choice of an optimal parameter ω is not as important and it is common to let $\omega = 1$ (Gauss Seidel scheme). Also the requirement that the matrix be positive definite is not as restrictive since relaxation is used only to improve the residual towards the required eigenvector. With SOR(k), k SOR iterations may be performed in each preconditioning step. Each iteration consists of the solution of two triangular systems of equations, which is as expensive as a matrix-vector multiplication in serial computers.

ILUT(p, τ) is an extension of the Incomplete LU factorization [11] that uses a two parameter strategy for dropping elements [20]. The first parameter controls the allowable fill-in in the sparse matrix. The second parameter controls the magnitude of the remaining elements in the factorization. When constructing the current row of L and U, elements that are smaller than some relative tolerance are dropped. In this way, the algorithm allows for selection of the important matrix elements in the factorization and it usually improves the efficiency of ILU.

An extension to the above preconditioners is to combine them with some iterative accelerator, as PCG, GMRES, etc [16, 21]. Since the matrices are symmetric either of the above is expected to work equally well. PCG iterations are much faster than those of GMRES because of the simple three-term recurrence of PCG. Using GMRES, however, may provide better robustness since it is not affected by near indefiniteness when A is shifted for interior eigenvalues [23]. Some early experiments have also verified this assumption. Since robustness is the subject of this study, the GMRES accelerator is adopted in the tests. It should be mentioned that since the preconditioning step involves the matrix $(M - \tilde{\lambda} I)$, band LU and ILUT factorization must be performed in every step. When the number of nonzero elements in the matrix is large, factorization especially in ILUT becomes very expensive and it may constitute the bottleneck of the iteration. In these cases, $(M - \tilde{\lambda} I)$ should be factored only when the change in $\tilde{\lambda}$ is significant, usually in early iterations. This methodology is not adopted in the following experiments.

Three test cases are used; two from the Harwell-Boeing collection and one from an application in atomic structure calculations. The first case, BCSSTK07, describes a stiffness matrix as a medium test problem. It is a relatively small matrix of dimension 420 and it has 7860 non zero elements. The three lowest eigenvalues are 460.65466, 1349.9746, 1594.5963, and the maximum is 1.915E+9. The separation gap for the lowest eigenvalue is 4.6E-07 and poor convergence characteristics are expected with simple preconditioners. The second case, SHERMAN1 from oil reservoir simulation is of dimension 1000 and it is very sparse with 3750 nonzero elements. The four smallest eigenvalues are 0.32348E-3, 0.10178E-2, 0.11131E-2, 0.15108E-2 and the maximum is 5.0448. The relative separation gap of the first eigenvalue is 1.4E-4 and convergence problems are still expected. The third case, called LITHIUM in this paper, is derived from the MCHF program for Li, 2S [28]. It is of dimension 862 and it is a dense problem with 240608 nonzero elements. This matrix has a fairly good eigenvalue separation with a gap of 6.2E-3 for the lowest eigenvalue, and diagonal preconditioning is expected to perform well. These three cases are representative of a variety of matrix spectrums. In the first convergence is slow even after the required eigenvalue has been closely approached. The second case converges faster but there are many close eigenvalues that can trap convergence. The third is an easy case for verification of robustness.

The experiments are performed on a SUN Sparcstation 2. The lowest eigenpair is sought in all test cases. In the first part, results from the GD method are given and cases where GD does not perform well are identified. In the second part, the robustness of the proposed modification is illustrated through the convergence improvement of the above cases.

4.2 Results from GD

Tables 1, 2 and 3 show the results for the three respective cases. Each table has several subtables where results from each preconditioner are given. The bottom subtable gives a summary of the best performing choices. Time and number of matrix-vector multiplications (Matvec) are reported for various parameter settings. SOR iterations are counted as matrix-vector multiplications. The parameter TRH appearing in the tables denotes the number of diagonal preconditioning iterations that are performed before the switch to the current preconditioner occurs. If less than TRH iterations are performed the method does not converge or if it does it is either slow or to the wrong eigenpair.

The results show that preconditioning can significantly reduce both the number of iterations and the total execution time. Like in linear systems, ILUT yields the best results followed by the cheaper but slower SOR. The straightforward band and diagonal preconditioners are the slowest except for cases predicted in the previous section. On the other hand the results verify that the choice of a preconditioner is not as obvious as in solving linear systems. More accurate preconditioners may perform much more poorly than simpler ones because of problem **P2**.

In BCSSTK07, diagonal and band preconditioning is extremely slow since a few diagonals do not capture the characteristics of this matrix. SOR(k) is more global and it drastically reduces time and iterations. After some k value (5-7) the increase of the matrix-vector multiplications outweighes the reduction in the number of iterations and time increases. This behavior is typical in linear systems as well. The reduction in time and iterations is more evident with ILUT. In the best measured case, ILUT($6,10^{-2}$) improves the time of

DIAG	Time	Matvec
	362.62	4484

Band LU(diags)			
# diags	TRH	Time	Matvec
3-diag	0	339.11	3970
5-diag	0	304.02	3334
7-diag	20	538.87	5304
9-diag	20	$\gg 748$	$\gg 7000$

SOR(k)			
k	TRH	Time	Matvec
1	10	94.70	2068
2	10	50.15	1378
3	50	37.01	1146
4	50	33.69	1140
5	50	27.63	974
6	50	26.86	1002
7	50	28.20	1106
8	50	25.21	995
9	50	25.88	1060
12	50	27.49	1207
15	50	29.11	1330

ILUT(fill,tol)				
fill	tol	TRH	Time	Matvec
0	0.	80	36.99	114
1	0.	80	39.10	114
2	0.	80	42.50	115
0	10^{-3}	50	44.94	155
1	10^{-3}	50	32.75	116
4	10^{-3}	50	29.52	98
6	10^{-3}	50	28.83	91
8	10^{-3}	50	34.86	96
0	10^{-2}	50	41.20	182
1	10^{-2}	50	27.41	129

Cont... ILUT(fill,tol)				
fill	tol	TRH	Time	Matvec
3	10^{-2}	50	22.60	106
5	10^{-2}	50	18.90	91
6	10^{-2}	50	17.43	85
7	10^{-2}	50	23.00	100
8	10^{-2}	50	21.40	93
0	10^{-1}	50	47.34	343
1	10^{-1}	50	40.8	298
2	10^{-1}	50	35.79	260
3	10^{-1}	50	37.10	267
0	10.	10	504.0	4484

SUMMARY		
Method	Time	Matvec
ILUT(6,10^{-2})	17.43	85
ILUT(5,10^{-2})	18.90	91
ILUT(3,10^{-2})	22.60	106
ILUT(1,10^{-2})	27.41	129
SOR(5)	27.63	974
SOR(6)	26.86	1002

Table 1: BCSSTK07. Results from DIAG, SOR, Band LU and ILUT preconditioning for various parameter settings. TRH diagonal preconditioning iterations are performed before the switch to the preconditioner occurs.

DIAG	Time	Matvec
	102.04	799

Band LU(diags)			
# diags	TRH	Time	Matvec
3-diag	200	45.52	327
5-diag	200	46.84	327
7-diag	200	48.11	327
21-diag	200	46.14	272

SOR(k)			
k	TRH	Time	Matvec
1	0	51.34	737
2	0	32.10	637
3	15	26.05	583
4	15	21.55	550
5	15	19.07	555
6	20	19.27	594
7	20	18.18	580
8	20	18.43	605
10	20	19.26	691
12	20	19.67	761

ILUT(fill,tol)				
fill	tol	TRH	Time	Matvec
0	0.	200	38.76	233
0	10^{-4}	200	35.84	236
0	10^{-3}	200	35.63	236
0	10^{-2}	0	17.67	84
0	10^{-2}	16	15.64	80
1	10^{-2}	0	17.52	73
1	10^{-2}	16	9.82	51
2	10^{-2}	16	8.26	42
3	10^{-2}	16	8.11	40

Cont... ILUT(fill,tol)				
fill	tol	TRH	Time	Matvec
0	10^{-1}	0	25.00	120
0	10^{-1}	16	19.69	112
1	10^{-1}	0	26.72	142
1	10^{-1}	16	18.23	104
2	10^{-1}	16	18.04	103
3	10^{-1}	16	18.04	104

SUMMARY		
Method	Time	Matvec
ILUT(3,10^{-2})	8.11	40
ILUT(2,10^{-2})	8.26	42
ILUT(1,10^{-2})	9.82	51
ILUT(0,10^{-2})	15.64	80
SOR(5)	19.07	555
SOR(7)	18.18	580

Table 2: SHERMAN1. Results from DIAG, SOR, Band LU and ILUT preconditioning for various parameter settings. TRH diagonal preconditioning iterations are performed before the switch to the preconditioner occurs.

DIAG	Time	Matvec
	23.98	47

Band LU(diags)			
# diags	TRH	Time	Matvec
3-diag	0	24.19	45
5-diag	0	21.38	39
9-diag	0	20.24	36
13-diag	0	20.69	35

SOR(k)			
k	TRH	Time	Matvec
1	0	23.98	49
2	0	22.28	49
3	2	21.76	50
4	3	22.75	53
5	2	24.21	56

ILUT(fill,tol)				
fill	tol	TRH	Time	Matvec
0	0.	0	-	-
0	100.	0	62.89	47
0	10.	0	41.88	31
1	10.	0	42.38	31
0	1.	0	26.93	16
1	1.	0	25.83	15
2	1.	0	24.95	14
3	1.	0	24.02	13
0	10^{-1}	5	29.89	15
2	10^{-1}	5	29.28	14

SUMMARY		
Method	Time	Matvec
Band LU(9-diag)	20.24	36
Band LU(13-diag)	20.69	35
Band LU(5-diag)	21.38	39
DIAG	23.98	47
SOR(3)	21.76	50
SOR(2)	22.28	49

Table 3: LITHIUM. Results from DIAG, SOR, Band LU and ILUT preconditioning for various parameter settings. TRH diagonal preconditioning iterations are performed before the switch to the preconditioner occurs.

the simple diagonal scaling case 20 times. However, the better preconditioner ILUT(2,0.) does not perform better in reducing the number of iterations. Notice, that TRH is between 50 and 80 even for simple preconditioners. Evidently, the ill-conditioning of the matrix makes it difficult for preconditioners to surpass higher eigenpairs in early iterations.

In SHERMAN1, band LU performs better than in the previous example because of the diagonal structure of the matrix. However, large bandwidth is required and the method is not competitive. Similar results with BCSSTK07 hold for the SOR(k). The turning point for the value k is a little higher because of the sparsity of the matrix and the low cost of matrix-vector multiplications. ILUT also yields similar results, outperforming all other methods. TRH is not necessarily large for all preconditioners and sometimes it can be zero. However, in cases as ILUT($1,10^{-2}$) and ILUT($1,10^{-1}$) increasing TRH speeds the method up. Again, these preconditioners would spend many early iterations trying to converge to a higher eigenpair. An extreme demonstration of this behavior is the ILUT(0,0.) which does not converge for any TRH<200.

In LITHIUM, the diagonal dominance of the matrix accounts for the very good performance of the band and diagonal preconditioners. The density of the matrix prevents SOR(k) and ILUT from reducing the total execution time although the number of iterations is reduced. Apart from a few cases TRH is always zero. The good separation of the required eigenvalue λ_k allows the preconditioners to "view" clearly λ_k from the approximation $\tilde{\lambda}$ even in the early steps.

		Only Prc		Prc-GMRES(5)	
Matrix	Precond Prc	Time	Matvec	Time	Matvec
BCSSTK07	ILUT($6,10^{-2}$)	17.43	85	15.8	204
	SOR(5)	27.63	974	39.63	2061
SHERMAN1	ILUT($1,10^{-2}$)	9.82	51	8.78	145
	SOR(5)	19.07	555	15.83	706
LITHIUM	Band LU(9-diag)	20.69	36	86.78	179
	SOR(2)	22.28	49	98.2	229

Table 4: Results from using (Prc) as a preconditioner to DVDSON, versus using GMRES(5) with preconditioner (Prc)

In Table 4 results from combining some of the previously tested preconditioners with GMRES(5) are presented. GMRES is allowed to run for 5 iterations. The total number of iterations decreases in general and the method demonstrates the robustness predicted earlier. Note especially that for the difficult cases even the time is reduced. However, the matrix-vector multiplications increase and for the easier LITHIUM case this method is much slower. Therefore, this method cannot be beneficial to all cases, because of the potential cost penalty.

4.3 Results from the modified GD

The improvement in the robustness with the modified GD is verified in both difficult and easier test cases. The two modifications on equation (5) are also tested separately. First, only the right hand side of equation (5) is changed and **E1** is used as a choice for ϵ (Olsen's method). Second, only the left hand side of equation (5) is changed and **E2** is used as a choice for shifting the preconditioning matrix $(M - (\tilde{\lambda} + \epsilon)I)$. Third, both of the above changes are combined yielding the modified method of equation (7). Finally, the exact correction to the eigenvalue is given as ϵ to both sides of equation (7), which provides the best eigenvalue preconditioner for a specified conventional preconditioner.

Figures 1, 2 and 3 depict the results from the above comparisons. The first two figures illustrate the effectiveness of the modification for ill-conditioned matrices, where a very accurate preconditioner is supplied. Figure 3 shows that the effectiveness of the modification is not relaxed when applied to a well-conditioned matrix or with a less accurate preconditioner.

Results from BCSSTK07, using ILUT(6,0.), appear in Figure 1. GD and Olsen's method converge extremely slowly. After 230 iterations they both terminate but GD terminates with the wrong eigenpair. On the contrary, shifting alone solves the problems encountered by the two methods and gives a convergence in 37 steps. When Olsen's method is used in addition, the modified method ameliorates the convergence and brings it much closer to the best possible convergence by ILUT(6,0.) than any other method. The difference of 10 iterations between the best possible and the modified GD is due to the ill-conditioning of the matrix, which makes it hard for the algorithm to pick an appropriate shift.

Results from SHERMAN1, using ILUT(0,0.), are similar and appear in Figure 2. The figure gives a clear pictorial explanation of the failure of GD. After some steps GD locks on some eigenvalue and reduces the residual. Only after the residual is below 10^{-8} does the method realize that it has the wrong eigenvalue and continues iterating. Soon, it locks on a new value but this time GD does not recognize the wrong eigenvalue. Olsen's method overcomes the first problem but it gets trapped in the second as well. The shifted and modified versions have no problems converging to the required eigenpair with convergence very close to the best possible obtainable by ILUT(0,0.).

Figure 3 shows that in well-conditioned cases or when a less accurate preconditioner is used, the differences between the methods diminish. However, in figure 3 the modified GD is still the best performing method. This attests the robustness of the modified GD, which can be effective in both easy and difficult problems.

It should be mentioned that in all methods the asymptotic convergence rate is the same. However, in GD and Olsen's methods the assumption of this rate is deferred until all the higher eigenvalues are "cleared". The modified method tries to expedite the "clearance", by shifting the preconditioning matrix.

5 Conclusions

The Generalized Davidson method is a well known variant of the Lanczos algorithm which exploits the important ideas of preconditioning. Some modifications to a previously developed code are proposed so that it can handle arbitrary matrix-vector multiplication routines and flexible preconditioning, thus improving ease of implementation and experimentation.

Preconditioning the eigenvalue problem is intrinsically more difficult than preconditioning linear systems. The spectrum needs to be compressed away from the required

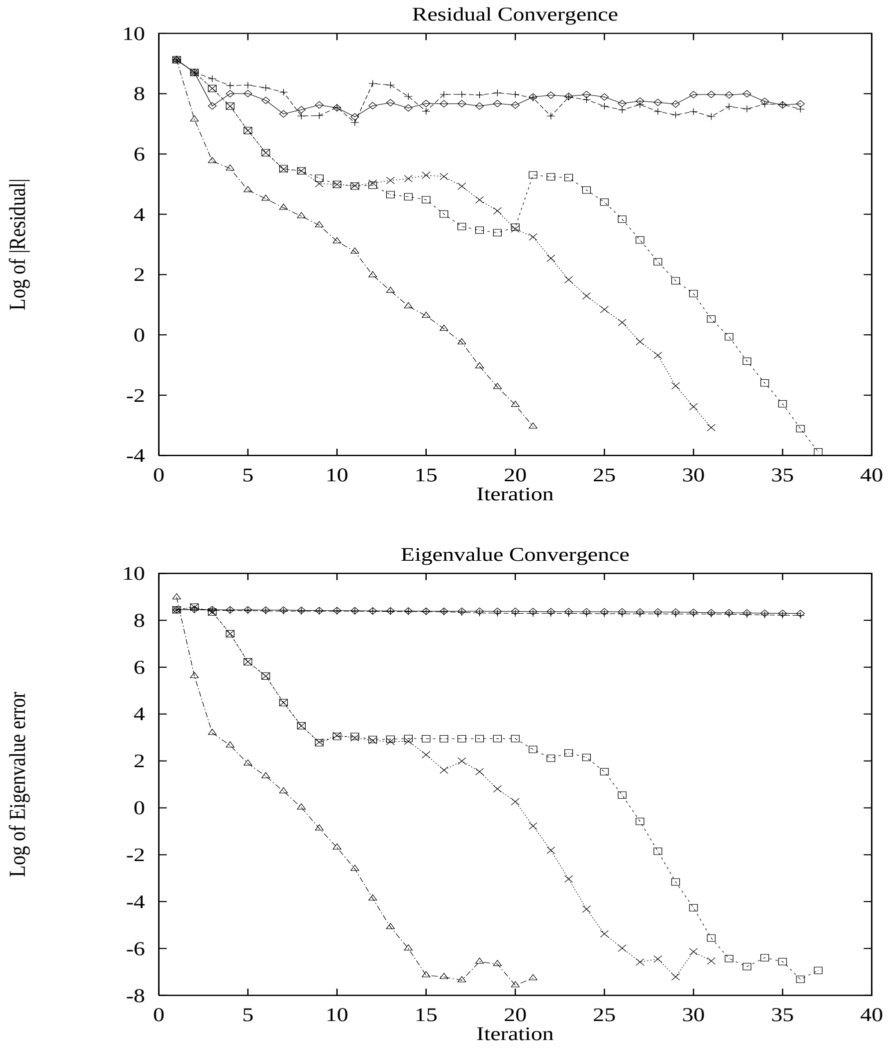

Figure 1: BCSSTK07 with ILUT(6,0.) preconditioner. Residual and eigenvalue convergence comparisons for GD and modified methods. ($\diamond$): Original GD, ($+$): Olsen's modification only, ($\square$): Choice **E2** as a shift only, ($\times$): Olsen's method with choice **E1** and choice **E2** as a shift, ($\triangle$): Exact ϵ for both choices.

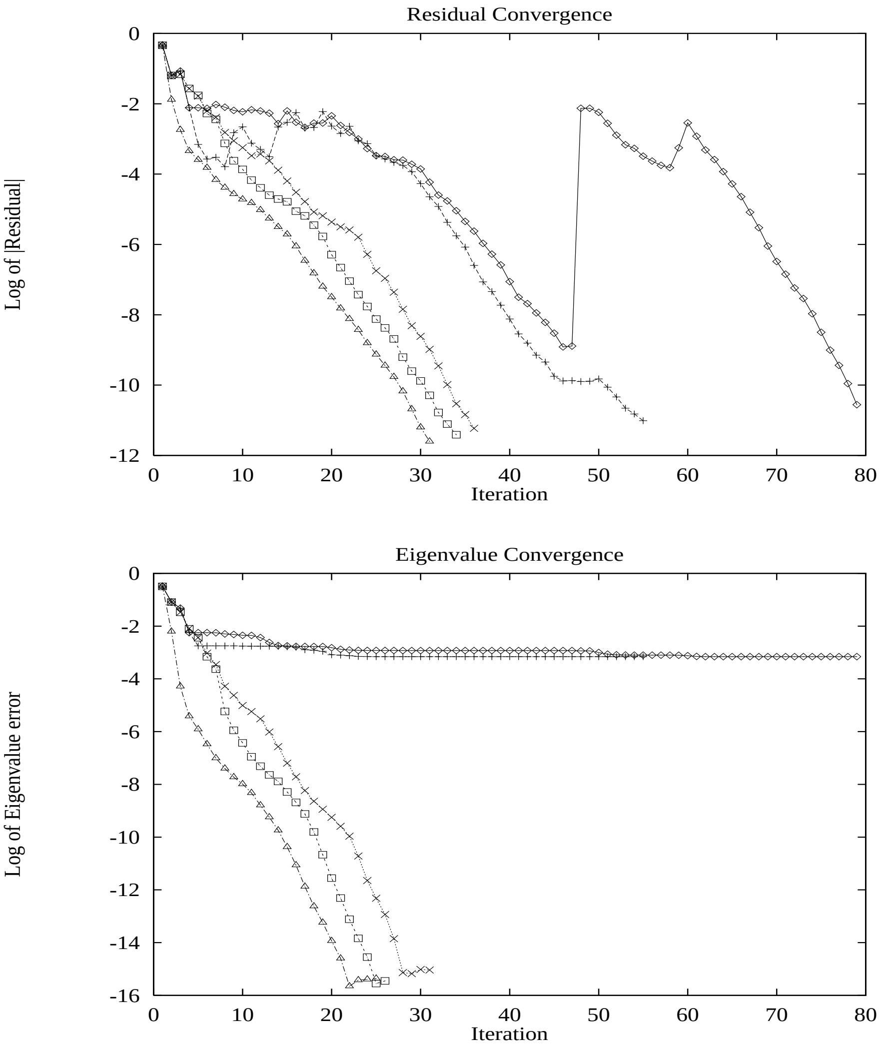

Figure 2: SHERMAN1 with ILUT(0,0.) preconditioner. Residual and eigenvalue convergence comparisons for GD and modified methods. ($\diamond$): Original GD, ($+$): Olsen's modification only, ($\Box$): Choice **E2** as a shift only, ($\times$): Olsen's method with choice **E1** and choice **E2** as a shift, ($\triangle$): Exact ϵ for both choices.

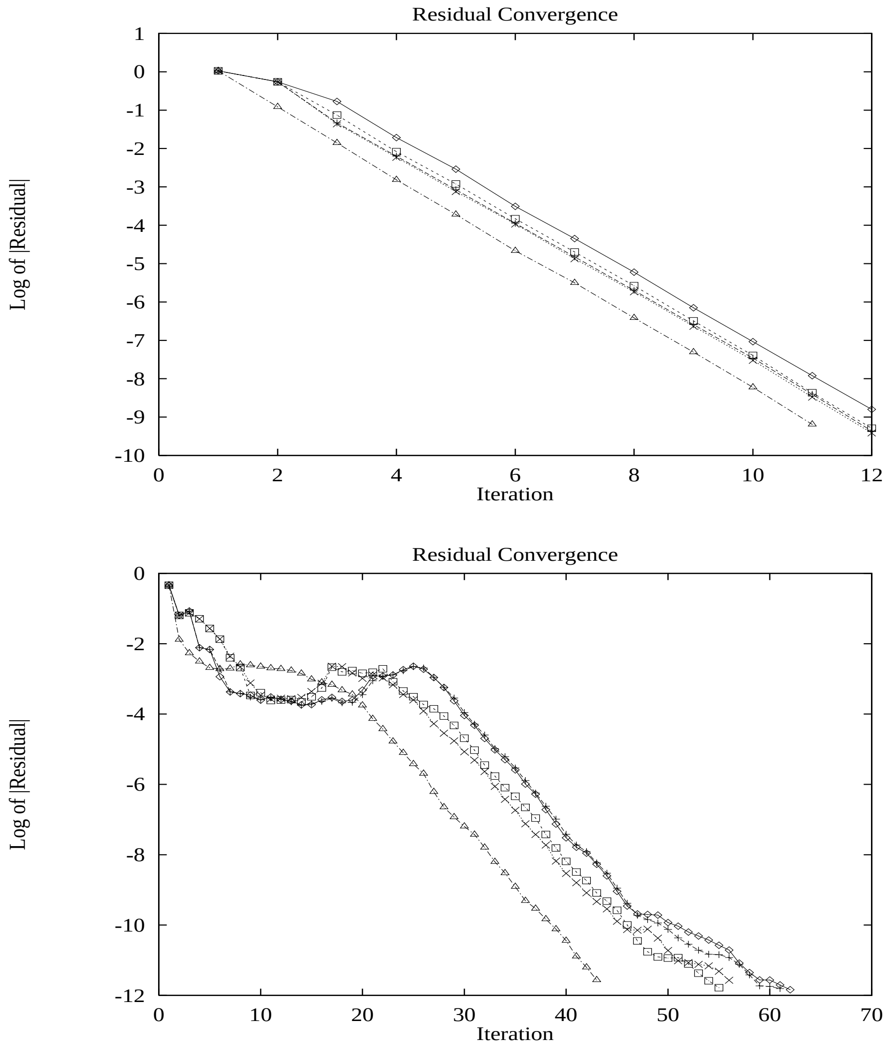

Figure 3: (Top) LITHIUM with ILUT(3,1.), (Bottom) SHERMAN1 with SOR(12). Residual convergence comparisons for GD and modified methods for an easy case (Top) and a less accurate preconditioner (Bottom). ($\diamond$): GD, ($+$): Olsen's only, ($\Box$): Shift only, ($\times$): Modified GD, ($\triangle$): Modified GD with exact ϵ.

eigenvalue λ_k, as opposed to the origin. Therefore, a "good" preconditioner $(M - \tilde{\lambda} I)$ should approximate $(A - \lambda_k I)$, and thus it depends on the required eigenvalue which is unknown.

Two problems are identified with preconditioning in GD. If M is a very good approximation to A, the preconditioner may yield no improvement. In addition, if $\tilde{\lambda}$ is far from λ_k, slow or erroneous convergence may occur. Experiments show that the second problem can plague convergence in ill-conditioned matrices. Flexible preconditioning may alleviate some problems but prior knowledge about the system is usually required. Olsen's method is beneficial for the first problem but it provides no improvement for the ill-conditioned case.

The solution proposed for the second problem is to shift the approximation $\tilde{\lambda}$ by an estimated correction, and thus obtain a "better" preconditioner for λ_k. With $(M - (\tilde{\lambda} + \epsilon) I)$ as a preconditioner, the iteration has the potential of avoiding wrong eigenvalues and leads to a more rapid convergence to the required one. Several easily obtainable choices exist for the estimation of the correction. From perturbation theory, this modification can be naturally combined with Olsen's result. This modified GD method improves the robustness and convergence of the original GD. Experiments verify the improvement in robustness, and show that even for very ill-conditioned cases the modified GD gives results very close to the best possible preconditioner $(M - \lambda_k I)$.

Further research should also focus on the choice of conventional preconditioner. It would be interesting to see if the large variety of preconditioners developed for linear systems can be used as effectively for eigenvalue problems.

Acknowledgements

This work was supported by National Science Foundation under grant numbers ASC-9005687 and DMR-9217287, and by AHPCRC (University of Minnesota) under Army Research Office grant number DAAL03-89-C-0038.

References

[1] M. Crouzeix, B. Philippe and M. Sadkane, The Davidson Method, *SIAM J. Sci. Comput.* 15 (1994) 62.

[2] E.R. Davidson, The Iterative Calculation of a Few of the lowest Eigenvalues and Corresponding Eigenvectors of Large Real-Symmetric Matrices, *J. Comput. Phys.* 17 (1975) 87.

[3] E.R. Davidson, Super-Matrix Methods, *Comput. Phys. Commun.* 53 (1989) 49.

[4] J.J. Dongarra, C.B. Moler and J.H. Wilkinson, Improving the accuracy of computed eigenvalues and eigenvectors, *SIAM J. Numer. Anal.* 20 (1983) 23.

[5] I.S. Duff, R.G. Grimes, and J.G. Lewis, Sparse Matrix Test Problems, *ACM Trans. Math. Software* 15 (1989) 1.

[6] B.S. Garbow et al., *Matrix eigensystem routines : EISPACK guide extension* (Berlin; New York: Springer-Verlag, 1977).

[7] C.F. Fischer, *The Hartree-Fock Method for Atoms: A Numerical approach* (J. Wiley, New York, 1977).

[8] C.F. Fischer and Muhammad Idrees, Spline algorithms for continuum functions, *Computers in Physics* 3 (1989) 53.

[9] G.H. Golub and C.F. Van Loan, *Matrix Computations, 2nd ed.* (Johns Hopkins Univ. Press, Baltimore, 1989).

[10] T.Z. Kalamboukis, Davidson's algorithm with and without perturbation corrections, *J. Phys. A* 13 (1980) 57.

[11] J.A. Meijerink and H.A. Van der Vorst, An iterative solution method for linear systems of which the coefficient matrix is a symmetric M-matrix, *Math. Comp.* 31 (1977) 148.

[12] R.B. Morgan and D.S. Scott, Generalizations of Davidson's Method for Computing Eigenvalues of Sparse Symmetric Matrices, *SIAM J. Sci. Stat. Comput.* 7 (1986) 817.

[13] R.B. Morgan and D.S. Scott, Preconditioning the Lanczos Algorithm for Sparse Symmetric Eigenvalue Problems, *SIAM J. Sci. Comput.* 14 (1993) 585.

[14] J. Ortega, *Numerical Analysis, as Second Course* (SIAM, Philadelphia, series: classics in numerical analysis, 1990).

[15] J. Olsen, P. Jørgensen and J. Simons, Passing the One-Billion Limit in Full Configuration-Interaction (FCI) Calculations, *Chem. Phys. Lett.* 169 (1990) 463.

[16] T.C. Oppe, W.D. Joubert, and D.R. Kincaid, *NSPCG User's Guide* Version 1.0, Numerical Analysis Center, The University of Texas at Austin.

[17] B.N. Parlett, The Software Scene in the Extraction of Eigenvalues from Sparse Matrices, *SIAM J. Sci. Stat. Comput.* 5 (1984) 590.

[18] B.N. Parlett, *The Symmetric Eigenvalue Problem* (Prentice-Hall, Englewood Cliffs, New Jersey, 1980).

[19] J.H. van Lenthe and Peter Pulay, A space-saving modification of Davidson's eigenvector algorithm, *J. Comput. Chem.* 11 (1990) 1164.

[20] Y. Saad, ILUT: a dual threshold incomplete LU factorization, Tech. Rep. 92-38, Minnesota Supercomputer Institute, University of Minnesota, 1992.

[21] Y. Saad, A Flexible Inner-Outer Preconditioned GMRES Algorithm, *SIAM J. Sci. Comput.* 14 (1993) 461.

[22] Y. Saad, Krylov subspace methods in distributed computing environments, Preprint 92-126, Army High Performance Computing Research Center, University of Minnesota, 1992.

[23] Y. Saad and M.H. Schultz, GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems, *SIAM J. Sci. Stat. Comput.* 7 (1986) 856.

[24] D.S. Scott, The advantages of inverted operators in Rayleigh-Ritz approximations, *SIAM J. Sci. Stat. Comput.* 3 (1982) 102.

[25] A. Stathopoulos and C. F. Fischer, A Davidson program for finding a few selected extreme eigenpairs of a large, sparse, real, symmetric matrix, *Comput. Phys. Commun.* 79 (1994) 268.

[26] A. Stathopoulos and C. F. Fischer, Reducing Synchronization on the Parallel Davidson method for the Large, Sparse, Eigenvalue Problem, in: *Proceedings of Supercomputing '93 Conference* (ACM Press, Portland 1993) 172.

[27] D.B. Szyld, Criteria for combining inverse and Rayleigh quotient iteration, *SIAM J. Numer. Anal.* 25 (1988) 1369.

[28] M. Tong, P. Jönsson and C. F. Fischer, Convergence Studies of Atomic Properties from Variational Methods: Total Energy, Ionization Energy, Specific Mass Shift, and Hyperfine Parameters for Li, In Press, *Physica Scripta.*

[29] J.H. Wilkinson, *The Algebraic Eigenvalue Problem* (Oxford University Press, New York, 1965).

Addresses of the Authors

Andreas Stathopoulos
Box 1679-B
Computer Science Department
Vanderbilt University
Nashville, TN 37235
USA

Office phone: (615) 322-3233
Fax Number: (615) 343-8006
email: andreas@vuse.vanderbilt.edu

Yousef Saad
Computer Science Department
University of Minnesota
Minneapolis, MN
USA

Charlotte F. Fischer
Computer Science Department
Vanderbilt University
Nashville, TN 37235
USA