Preconditioned Projection Methods for Sparse Unsymmetric Eigenproblems

Gautam M. Schroff

CRPC-TR91116
January, 1991

Center for Research on Parallel Computation
Rice University
P.O. Box 1892
Houston, TX 77251-1892

Preconditioned Projection Methods for Sparse Unsymmetric Eigenproblems

Gautam M. Shroff*

January 1991

Abstract

Projection methods for sparse eigenvalue problems, (like the Lanczos and Arnoldi methods) are useful for computing a few eigenvalues and eigenvectors of a large sparse matrix. These methods compute approximations lying in some small dimensional subspace such as a Krylov subspace. Often one is interested in computing certain eigenvalues and vectors which are not well approximated in a Krylov space and so one uses some form of preconditioning (e.g. polynomial preconditioners). In this paper some other methods for preconditioning like relaxation are considered and it is shown how they may be useful in applications like detecting Hopf bifurcations and computing small singular vectors of sparse matrices.

1 Projection Methods

In many applications we are faced with the problem of computing eigenvalues and eigenvectors of a large, sparse, and in general non-symmetric matrix $A \in \mathcal{R}^{N \times N}$. Usually, only certain eigenvalues and corresponding eigenvectors are required, being specified as either the ones closest to some complex number μ or the ones in some part of the complex plane, (e.g. the right-half plane), etc. A related problem is the computation of specific singular values and corresponding singular vectors of a sparse matrix.

By a sparse matrix we shall take to mean one that is prohibitively expensive to factor (by either a QR or an LU decomposition). We must therefore rule out traditional algorithms for eigenproblems like the QR algorithm. Projection methods are popular for such sparse problems because they can avoid factoring the matrix. See Saad [10] for a survey of projection methods for sparse eigenproblems.

A projection method for computing a specified eigenpair of A computes a sequence of approximate eigenpairs $x^{(n)}, \lambda^{(n)}$ with $x^{(n)} \in K_n$, where K_n is some small dimensional (i.e. much less than N) subspace of $\mathcal{R}^N$. If $\|Ax^{(n)} - \lambda^{(n)}x^{(n)}\| \to 0$ as $n \to \infty$, then the projection method is said to converge. Note that with this definition, we sidestep the problem of condition that naturally arises in the non-symmetric case, where the actual error may be much larger than the above residual. However, this is the best we can expect for case of general non-symmetric matrices.

Let u be the desired eigenvector and π the orthogonal projection onto the subspace K_n. Clearly, the best approximation to u in K_n is πu. However, this is in general impossible to obtain since u is unknown, so a projection method will compute some approximation to πu as $x^{(n)}$. The situation is illustrated in Fig. 1.

*Applied Mathematics 217-50, California Institute of Technology, Pasadena CA 91125; gms@ama.caltech.edu. This work was supported in part by the NSF Cooperative Agreement No. CCR-8809615. The government has certain rights on this material.

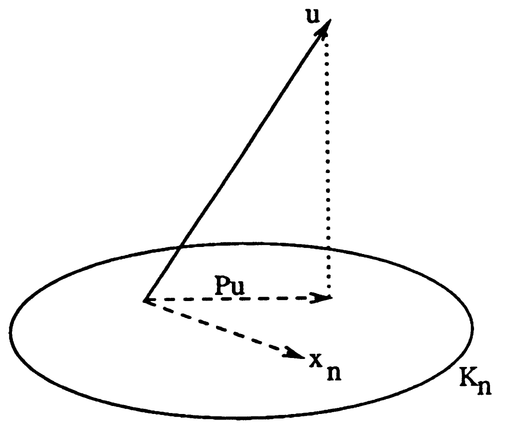

Figure 1: Projection Method

To summarize, a projection method may be described by (i) the choice of the subspaces K_n and (ii) the manner in which the eigenpair approximation $x^{(n)}, \lambda^{(n)}$ is chosen within the subspace K_n. In §2 we describe Krylov subspace methods which are the most common type of projection method, following which in §3 we will introduce the concept of a preconditioned projection method, which will not, in general, be a Krylov space method. In §4 we consider computing certain singular vectors and singular values of a matrix and give a preconditioned projection method for this problem. In §5 we turn our attention to question (ii) above, and suggest using a minimum residual approach to compute the eigenpair approximants. Finally we present numerical experiments in §6 and our conclusions in §7.

2 Krylov Subspace Methods

Common projection methods like the Arnoldi and Lanczos algorithms [12, 11] are examples of Krylov subspace methods. The subspace K_n is chosen as the Krylov subspace

$$K_n = \text{span}[x^{(0)}, Ax^{(0)}, \ldots, A^{n-1}x^{(0)}], \tag{1}$$

where $x^{(0)}$ is some starting vector chosen randomly. So in particular $K_i \subseteq K_{i+1} \subseteq \ldots$, $dim(K_n) = n$, and a vector $v \in K_n$ is of the form $v = p(A)x^{(0)}$ where p is a degree $n-1$ polynomial. To illustrate the dependence of the behavior of a Krylov space method on the spectrum of A, we briefly review some known results [11]. We assume that A has a complete set of linearly independent eigenvectors $\{u_i\}$ and distinct eigenvalues (i.e. the generic case), and let the starting vector be expanded in this basis,

$$x^{(0)} = \sum_{i=1}^{N} \alpha_i u_i, \tag{2}$$

so

$$p(A)x^{(0)} = \sum_{i=1}^{N} p(\lambda_i) u_i. \tag{3}$$

Let us focus on one particular eigenvector, say u_k. Assume that p is normalized so that $p(\lambda_k) = 1$, and also that $\alpha_k \neq 0$. Then

$$||\frac{p(A)x^{(0)}}{\alpha_k} - u_k|| = ||\sum_{i \neq k} \frac{\alpha_i}{\alpha_k} p(\lambda_i)||. \tag{4}$$

Using the above and the fact that πu_k is the closest vector in K_n to u_k, we easily get the bound

$$||u_k - \pi u_k|| \leq \left(\sum_{i \neq k} |\frac{\alpha_i}{\alpha_k}| \right) \varepsilon_k^{n-1}, \tag{5}$$

where

$$\varepsilon_k^{n-1} = \min_{\substack{p \in \mathcal{P}_{n-1} \\ p(\lambda_k) = 1}} \max_{i \neq k} |p(\lambda_i)| \tag{6}$$

is the "degree of approximation" of the null function by degree $n-1$ polynomials satisfying $p(\lambda_k) = 1$ on the set $\{\lambda_i, i \neq k\}$.

The influence of the spectrum of A on the Krylov method is now clear. The isolated eigenvalues are associated with small values of $\varepsilon^{(n-1)}$ and therefore the corresponding eigenvectors are likely to be much better approximated in a given Krylov space K_n than those corresponding to interior or clustered eigenvalues, for which $\varepsilon^{(n-1)}$ is much larger. This is a basic weakness in the Krylov space technique - convergence is slow for interior eigenvalues.

Of course, πu_k is not in general available, rather $x^{(n)}$ is computed as some approximation to it. A commonly used technique is the **Galerkin approximation**, in which the approximate eigenpair $x^{(n)}, \lambda^{(n)}$ is chosen with $x^{(n)} \in K_n$ such that

$$(Ax^{(n)} - \lambda^{(n)} x^{(n)}) \perp K_n. \tag{7}$$

If V is an orthonormal basis for K_n, then $x^{(n)} = Vy^{(n)}$ for some $y^{(n)} \in \mathcal{R}^n$ and it follows from the above that

$$V^T A V = \lambda^{(n)} y^{(n)}, \tag{8}$$

i.e., the approximate eigenpair $x^{(n)}, \lambda^{(n)}$ corresponds to an exact eigenpair of $H = V^T A V$, the restriction of A to K_n.

The Krylov space method as described above involves successive approximation in a sequence of nested subspaces $K_1 \subseteq K_2 \ldots K_N$, and therefore the process must terminate in at most N steps since $K_N = \mathcal{R}^N$. This is however computationally infeasible in general since the amount of work required per step grows as the dimension of the subspace increases. Therefore a **restarted** method is used in practice, i.e. the computation is restarted after a fixed number (say s) steps with the new starting vector chosen as $x^{(s)}$. In such a restarted method the computational cost per step is bounded, but the finite termination property is naturally lost, and in fact, the process may not even converge. The **Restarted Arnoldi Iteration**, see [11], is a restarted Krylov space method as described above.

The Lanczos [12] algorithm is also a Krylov space method, and in the symmetric case is identical with Arnoldi's method. In the unsymmetric case, though the subspaces are still Krylov spaces, the eigenpair approximation is computed differently from that described above (see [11]). Further, a three term relationship between the vectors keeps the computational cost per step constant, so restarting is not generally necessary. However, in the unsymmetric case multiplications of vectors by A^T as well as A are required, which firstly may not be practical in some applications, and further may cause difficulty in parallel implementations. Also, recovering the eigenvectors becomes much more expensive in the Lanczos approach and of course there are stability problems in the unsymmetric case. We do not consider the Lanczos approach in this paper; for some recent work on stable versions of the unsymmetric Lanczos algorithm see [3].

3 Preconditioning:

As we have seen in the previous section, the convergence of a Krylov space method to a particular eigenvector u_k is dependent on the "degree of approximation", $\varepsilon_k^{(n-1)}$, which implies in general slower convergence for the interior or clustered eigenvalues. In certain situations however, it is precisely these eigenvalues and eigenvectors which are of interest. Preconditioning techniques are therefore desirable to accelerate convergence in such situations.

In the context of iterative methods for linear systems preconditioning means replacing the problem $Ax = b$ with an equivalent problem $\hat{A}\hat{x} = \hat{b}$, and indeed, a variety of such preconditioning techniques are available. For the eigenproblem the corresponding preconditioning method is to use a **spectral transformation**, i.e., instead of determining the eigenvalues of A, we work with $f(A)$ instead. If f is analytic on the spectrum of A, the eigenvalues of $f(A)$ are $f(\lambda_i)$ and the eigenvectors are still u_i. One example of this type of preconditioning is "shift and invert", i.e. if the eigenvalue closest to μ is desired, we work with the matrix

$$(A - \mu)^{-1}.$$

Typically the shift μ is also refined as better approximations to the eigenvalue are obtained. However, this involves solving linear systems at each step of the process and may not be feasible in situations where the matrix is too large and sparse to be factored often. Another popular spectral transformation is to use some polynomial for f, such as a Chebyshev polynomial which is small on the unwanted parts of the spectrum and large near the desired eigenvalues (i.e., some approximation to the optimal polynomial $\varepsilon_k^{(n-1)}$). One disadvantage then becomes the spurious eigenvalues obtained when $f(\lambda) = \nu$ is solved to recover the eigenvalues of the original matrix A.

A popular alternative to using a spectral transformation is to precondition the **vectors** in the projection method. For example, a **polynomial preconditioned** restarted Arnoldi method [11] would perform s steps of the Arnoldi process, and then restart using $q(A)x^{(n)}$ rather than $x^{(n)}$, where q is some polynomial that approximates the desired optimal approximating polynomial $\varepsilon_k^{(n-1)}$, like a Chebyshev polynomial. Note that the resulting projection method is a restarted Krylov method with a different restarting technique. The preconditioned projection method we will propose will also precondition the vectors, but rather than apply the preconditioning only at the restarting step, we will precondition each new vector before adding it to the subspace. The resulting subspaces will therefore no longer be Krylov subspaces. Also, we will not use polynomials (in which case we would merely recover the spectral transformation method), but "relaxation", as is described below.

Relaxation for eigenproblems

Ruhe [9] proposed the following relaxation method for sparse symmetric eigenvalue problems. If the exact eigenvalue μ is known, then it may be possible to use an iterative method such as SOR to solve the singular system

$$(A - \mu I)y = 0$$

for the eigenvector y. Ruhe uses such an iteration with approximate eigenvalues μ_k which are updated at each step of the iteration using the Rayleigh quotient of the current vector iterate y_k,

$$\mu_k = \frac{y_k^T A y_k}{y_k^T y_k}.$$

The behavior of such a scheme for computing a specified eigenvalue and the corresponding eigenvector is dependent on the behavior of the iterative method used on singular systems. Iterative methods for singular systems have not received a lot of attention, some important references are

[5, 7] and a recent survey can be found in [2]. If A is singular, and $A = M - N$ is a splitting with M nonsingular, then the iterative method

$$x_{k+1} \longleftarrow M^{-1}Nx_k$$

is said to converge if $\lim_{k\to\infty} x_k$ lies in the null space of A. Note that since A is singular, $Az = 0$ for some z, i.e. $Mz = Nz$, so the iteration matrix has one as an eigenvalue. In [5] it is shown that the convergence depends on the largest of the remaining eigenvalues of $M^{-1}N$.

Davidson's method

Davidson [1] proposed a projection method for the symmetric eigenvalue problem that uses a particular kind of relaxation as preconditioning. In Davidson's method, the subspace K_n is built up as follows. Let $V = \{v_1, \dots v_{n-1}\}$ be an orthonormal basis for K_{n-1} and let $x^{(n-1)}, \lambda^{(n-1)}$ be an approximate eigenpair for A computed by a Galerkin process using K_{n-1}. The new vector

$$y_n = (D - \lambda^{(n-1)})^{-1}Ax^{(n-1)} \tag{9}$$

is formed where D is the diagonal of A. This vector is orthogonalized with respect to the vectors V to obtain v_n, thus forming the basis for K_n. Morgan and Scott [8] noted that the new vector y_n in Davidson's method is nothing but the correction produced by a Jacobi iteration for the nearly singular system

$$(A - \lambda_{n-1})y = 0 \tag{10}$$

starting with $x^{(n-1)}$ as initial guess.

General Preconditioned Projection Method

The relaxation scheme of Ruhe converges only under restrictive assumptions and Davidson's method uses only the simple Jacobi relaxation scheme. As a natural combination of the above ideas, we consider projection methods using a general relaxation preconditioning. We will also consider general unsymmetric matrices unlike the above two methods. Of course, in practice a restarted method must be used for the same reason as in the Krylov space methods. We summarize this **Relaxation Preconditioned Projection** algorithm in Fig. 2. The algorithm attempts to find the eigenvalue closest to $\mu \in C$ and the corresponding eigenvector, therefore the eigenvalue estimate is kept constant at μ for the first p steps. The restart parameter s and a starting vector x also need to be given. It should be noted that certain implementation details have been omitted in the algorithm as presented in Fig. 2 for clarity. For example it is clearly redundant to have to compute $H = V^TAV$ from scratch at each step as indicated (i.e. $O(nk^2)$ work), rather, if the vectors AV are maintained along with V, the new H can be constructed from the previous step with $O(nk)$ work (such details will also be omitted in the presentation of other algorithms).

Note that the applying one step of the iterative method based on the splitting $A = M - N$ for the equation $(A - \lambda^{(k)}I)x^{(k)} = 0$ yields that vector $(I - M^{-1}(A - \lambda^{(k)}I))x^{(k)}$. Since $x^{(k)}$ already lies in the subspace V, only the correction $M^{-1}(A - \lambda^{(k)}I)x^{(k)}$ is added to the subspace.

The convergence of the algorithm can be understood by noting that once the eigenvalue estimate $\lambda^{(k)}$ has almost converged, the algorithm is nothing but a projection method using the Krylov space of the matrix $B = M^{-1}(A - \lambda I)$. The convergence is therefore governed by the separation of the zero eigenvalue of B from the remaining eigenvalues. The ideal choice for the splitting matrix M^{-1} is some sort of pseudoinverse, for if

$$A - \lambda I = P^{-1}\begin{pmatrix} \lambda_1 - \lambda & & & \\ & \ddots & & \\ & & \lambda_{N-1} - \lambda & \\ & & & 0 \end{pmatrix} P \tag{11}$$

```
Algorithm RPP($A, x, \mu, p, s$)
do
  $v_0 = \frac{x}{\|x\|_2}$; $\lambda^{(0)} = \mu$; count = 0;
  for $k = 1$ to $s$
    1. Galerkin Step:
       $V = [v_0, \ldots, v_{(k-1)}]$; $H = V^T A V$;
       $(z^{(k)}, \hat{\lambda}^{(k)})$ = eigenpair of $H$ with $\hat{\lambda}^{(k)}$ closest to $\lambda^{(k-1)}$;
       $x^{(k)} = V z^{(k)}$; count = count + 1;
       if (count $\geq p$) $\lambda^{(k)} = \hat{\lambda}^{(k)}$ else $\lambda^{(k)} = \lambda^{(k-1)}$; endif
    2. Relaxation Step:
       Let $(A - \lambda^{(k)} I) = M - N$ be a splitting, $M$ nonsingular;
       $y^{(k)} = M^{-1}(A - \lambda^{(k)} I) x^{(k)}$;
       $v_k = y^{(k)} - \sum_{i=0}^{k-1} h_{ik} v_i$, where $h_{ik}$
       are chosen so that $v_k \perp v_i$ for $i = 0, \ldots, k-1$;
       $v_k \leftarrow \frac{v_k}{\|v_k\|_2}$;
    3. Convergence check:
       if ($\|(A - \lambda^{(k)} I) x^{(k)}\| < tol$) End;
  endfor
  $v_0 = x^{(k)}$;
enddo
```

Figure 2: Relaxation Preconditioned Projection Method

and

$$M^{-1} = P^{-1} \begin{pmatrix} \frac{1}{\lambda_1 - \lambda} & & & \\ & \ddots & & \\ & & \frac{1}{\lambda_{N-1} - \lambda} & \\ & & & 1 \end{pmatrix} P \tag{12}$$

then

$$M^{-1}(A - \lambda I) = P^{-1} \begin{pmatrix} 1 & & & \\ & \ddots & & \\ & & 1 & \\ & & & 0 \end{pmatrix} P \tag{13}$$

yielding the perfect separation, so that $\varepsilon^{(k-1)} = 0$ and convergence is immediate. In practice however, we do not know such a preconditioner, and the standard iterative methods based on splittings, like Gauss Seidel, SOR, etc. may be used instead. In the section on experiments we will present results using the SOR relaxation for an application involving detecting Hopf bifurcations.

In the context of symmetric matrices, Davidson's method is considerably more expensive per step than the conventional Lanczos algorithm, because the Galerkin process is carried out using an arbitrary basis rather than a Krylov basis for which a natural three term recurrence exists. Therefore, even though the convergence is markedly faster for certain eigenvalues, the overall behavior is not drastically improved over Lanczos. In the unsymmetric case however, when we compare the preconditioned method to the Arnoldi process, the computational requirements per step are comparable, and therefore it is hoped that the expected faster convergence of the preconditioned method will translate into an overall faster algorithm.

```
for $i = 1$ to $N$
    $\delta_i \leftarrow (\theta x_2(i) - b_i^T x_1)/a_{ii}$
    $x_1 \leftarrow x_1 + \delta_i e_i$
endfor
for $i = 1$ to $N$
    $\delta_i \leftarrow (\theta x_1(i) - a_i^T x_2)/a_{ii}$
    $x_2 = x_2 + \delta_i e_i$
endfor
```

Figure 3: SVD Relaxation Step

4 Singular Values and Vectors

Related to the eigenvalue problem is the important problem of computing singular values and singular vectors of a general matrix $A \in \mathcal{R}^{N\times N}$. Recall that the singular vectors u and v associated with the singular value σ of A satisfy

$$\begin{aligned} Au &= \sigma v \qquad (14)\\ A^T v &= \sigma u. \qquad (15)\end{aligned}$$

The singular values of A, $\{\sigma_1 \geq \sigma_2, \ldots, \geq \sigma_N \geq 0\}$ are related to the eigenvalues of the symmetric matrix

$$M = \begin{pmatrix} 0 & A \\ A^T & 0 \end{pmatrix}, \qquad (16)$$

which are $\{\sigma_1, \ldots, \sigma_N, -\sigma_N, \ldots, -\sigma_1\}$.

A popular algorithm for computing a few singular values and vectors of a large sparse matrix is the bidiagonalization algorithm which implicitly performs the symmetric Lanczos algorithm on M [4]. Therefore, the larger singular values of A, which are the extreme eigenvalues of M, are computed rapidly, whereas the smaller singular values may take much longer to converge. This problem is heightened if A is nearly singular, in which case the eigenvalues $\sigma_N, -\sigma_N$ of M are clustered relative to its other eigenvalues, and the convergence is extremely slow for these eigenvectors. Further, it is often the case that the small singular values and vectors are in fact of great interest, for example in continuation/path following applications near turning points [6].

We consider therefore the case when A is nearly singular and its small singular value and associated singular vectors are to be computed. We will use a projection method with a form of relaxation preconditioning on the matrix M.

Relaxation

We would like to perform relaxation steps on nearly singular systems of the form $(M - \theta I)x = 0$, where θ is an approximate singular value of A. Because of the structure of M, the diagonal elements are θ, which, as we have assumed, is likely to be small. So the standard relaxations like SOR or Gauss-Seidel will be unstable. We propose a modified relaxation for this problem as follows. The Gauss Seidel relaxation proceeds by updating the ith coordinate of the vector, $x(i)$, in order to make the residual orthogonal to the principle vector e_i, cycling through all i in this fashion. More generally, the ith coordinate can be updated to make the residual orthogonal to e_j, where $j \neq i$ in general; and as we cycle through i, we make sure that all e_j are covered. So let $A = [a_1^T, \ldots, a_N^T]^T$ and $A^T = [b_1^T, \ldots, b_N^T]^T$. Also partition the vector x as $x = [x_1^T, x_2^T]^T$. Then one **SVD Relaxation** step on $(M - \theta I)x = 0$ is shown in Fig. 3.

```
Algorithmn SVD-RPP($A, x, p, s$)
do
  $x \leftarrow x/\|x\|_2$; Partition $x = [x_1^T, x_2^T]^T$;
  $v_0 = x$; $v_1 = [-x_1^T, x_2^T]^T$;
  $\sigma^{(0)} = 0$; count = 0;
  for $k = 1$ to $s$
          1. Galerkin Step:
             $V = [v_0, v_1 \dots, v_{(2k-2)}]$; $H = V^T M V$;
             $(z^{(k)}, \hat{\sigma}^{(k)})$ = eigenpair of $H$ with $\hat{\sigma}^{(k)}$ closest to $\sigma^{(k-1)}$;
             $x^{(k)} = Vz^{(k)}$; count = count + 1;
             if (count $\geq p$) $\sigma^{(k)} = \hat{\sigma}^{(k)}$ else $\sigma^{(k)} = \sigma^{(k-1)}$;
          2. SVD Relaxation Step:
             $y^{(k)}$ =SVD-Relax($x^{(k)}$)
             Partition $y^{(k)} = [y_1^{(k)^T}, y_2^{(k)^T}]^T$;
             $\hat{y}^{(k)} = [-y_1^{(k)^T}, y_2^{(k)^T}]^T$;
             $v_{2k} = y^{(k)} - \sum_{i=0}^{2k-1} h_{ik} v_i$;
             $v_{2k+1} = \hat{y}^{(k)} - \sum_{i=0}^{2k-1} g_{ik} v_i$;
             where $h_{ik}$ are chosen so that $v_{2k}, v_{2k+1} \perp v_i$ for $i = 0, \dots, 2k-1$;
             $v_{2k} \leftarrow \frac{v_{2k}}{\|v_{2k}\|_2}$;
             $v_{2k+1} \leftarrow \frac{v_{2k+1}}{\|v_{2k+1}\|_2}$;
          3. Convergence check:
             if ($\|(M - \sigma^{(k)} I)x^{(k)}\| < tol$) End;
  endfor
  $v_0 = x^{(k)}$;
enddo
```

Figure 4: SVD Relaxation Preconditioned Projection Method

Now we can think of using a projection method preconditioned with the above relaxation to compute the eigenvector of M corresponding to one of its small eigenvalues, say σ_N. The singular vectors of A can then be read off from the components of this eigenvector, which is just $[u^T, v^T]^T$. In numerical experiments this approach was found to converge faster than the bidiagonalization algorithm; but it was noticed that the presence of the eigenvalue $-\sigma_N$ close to σ_N caused oscillations which were slowing the convergence. However if we attempt to capture the invariant subspace associated with σ_N and $-\sigma_N$ instead, the convergence is much faster. We use the fact that if $x = [x_1^T, x_2^T]^T$ is an eigenvector of M corresponding to σ_N, then $\hat{x} = [-x_1^T, x_2^T]^T$ is an eigenvector corresponding to $-\sigma_N$. Therefore, once we have an approximate eigenvector x corresponding to σ_N which we add to the subspace, we also form the vector $\hat{x}$ and add that to the subspace as well. In Fig. 4 we summarize the **SVD Relaxation Preconditioned Projection** algorithm for computing singular vectors associated with the small singular values of A. As before, a restart parameter s, as well as a starting vector $x \in \mathcal{R}^{2N}$ need to be provided. The initial guess for σ_N is kept at zero for p steps in order to ensure convergence to the smallest singular value. The procedure *SVD Relax* incorporating the relaxation from Fig. 3 is used.

Note that each step of algorithm SVD-RPP requires twice the number of matrix vector multiplies as RPP, also since V consists of $2k$ columns at step k, the Galerkin approximation calls for solving a $2k \times 2k$ problem rather than a $k \times k$ problem at each step. In Section §6 we will present numerical

experiments using the above algorithm and also compare it to the bidiagonalization algorithm.

5 Minimum Residual Algorithm

In the previous sections we have discussed projection methods in which the subspaces K_n are preconditioned and are therefore no longer the traditional Krylov subspaces. We now turn our attention to the eigenpair approximation within each subspace. The question of the best eigenpair approximation in K_n may be formulated as

$$\min_{\substack{x \in K_n \\ \theta \approx \text{ close to} \\ \text{desired eigenvalue}}} \|Ax - \theta x\|. \tag{17}$$

For the moment assume that θ is fixed, use the $\|.\|_2$ norm, and let V be an orthonormal basis for K_n with y such that $x = Vy$. Then we have a least squares problem which is solved by choosing y as the eigenvector corresponding to the minimum eigenvalue of

$$V^T(A - \theta I)^T(A - \theta I)V. \tag{18}$$

Now, in case A is symmetric, the above is nothing but $V^T(A-\theta I)^2V$, and y can therefore be chosen as the eigenvector corresponding to the eigenvalue of smallest magnitude of

$$V^T(A - \theta I)V = V^TAV - \theta I.$$

Allowing θ to vary, we see that this is the same as the Galerkin approximation. So in the symmetric case, the Galerkin approximation also minimizes the residual over K_n.

Of course, this is no longer true in the general unsymmetric case. We have so far been using the Galerkin approximation to compute approximate eigenpairs $x^{(n)}, \lambda^{(n)}$ with $x^{(n)} \in K_n$. In general this strategy works well even in the non-symmetric case when a number of eigenvalues and eigenvectors are to be determined [11]. However, in case we are interested in computing only a few specified eigenvectors we should think of minimizing the residual rather than doing a Galerkin approximation.

The problem (17) is in general quite difficult, however, if we keep θ fixed then y can be chosen as the eigenvector of (18). One way to view this procedure is the following. If θ were an exact eigenvalue, then by minimizing the residual we are finding $x \in K_n$ such that

$$(Ax - \theta x) \perp (A - \theta I)K_n, \text{ (rather than } K_n). \tag{19}$$

Using the terminology of Saad '89 [11], this is an "oblique" projection method as opposed to the Galerkin approximation which is an orthogonal projection method. Once we have computed the eigenvector estimate x, the eigenvalue estimate remains to be updated. For this the Galerkin approach is used. However, if we perform both the minimum residual as well as Galerkin approximations at each step, the computational cost essentially doubles. Therefore in practice we update the eigenvalue estimate only once every r steps, where r is usually 3 or 4, and use the Galerkin approximation for both the eigenvalue and eigenvector in this case. In the remaining steps the eigenvalue estimate is kept constant while using the minimum residual approximation for the eigenvector. This approach is combined with the preconditioning discussed in the previous sections, obtaining the **Preconditioned Projection - Minimum Residual** algorithm shown in Fig. 5. The quantities x,s,p and μ are assumed given as in Algorithm RPP of Fig. 2, as well as r - the number of minimum residual steps between Galerkin steps.

```
Algorithm PP-MR($A, x, \mu, p, s, r$)
do
  $v_0 = \frac{x}{\|x\|_2}$; $\lambda^{(0)} = \mu$;
  count = 0; rcount = 0;
  for $k = 1$ to $s$
     1. Eigenvalue approximation:
        $V = [v_0, \ldots, v_{(k-1)}]$;
        if (rcount = $r$) {Galerkin Step}
          $(z^{(k)}, \hat{\lambda}^{(k)})$ = eigenpair of $H = V^T A V$ with $\hat{\lambda}^{(k)}$ closest to $\lambda^{(k-1)}$;
          if (count $\geq p$) $\lambda^{(k)} = \hat{\lambda}^{(k)}$ else $\lambda^{(k)} = \lambda^{(k-1)}$; endif
        else {Min. Res. Step}
          $\lambda^{(k)} = \lambda^{(k-1)}$;
          $z^{(k)}$ = smallest eigenvector of $V^T(A - \lambda^{(k)}I)^T(A - \lambda^{(k)}I)V$;
        endif
        $x^{(k)} = V z^{(k)}$; count = count + 1;
     2. Relaxation Step:
        Let $(A - \lambda^{(k)}I) = M - N$ be a splitting, $M$ nonsingular;
        $y^{(k)} = M^{-1}(A - \lambda^{(k)}I)x^{(k)}$;
        $v_k = y^{(k)} - \sum_{i=0}^{k-1} h_{ik} v_i$, where $h_{ik}$
        are chosen so that $v_k \perp v_i$ for $i = 0, \ldots, k-1$;
        $v_k \leftarrow \frac{v_k}{\|v_k\|_2}$;
     3. Convergence check:
        if ($\|(A - \lambda^{(k)}I)x^{(k)}\| < tol$) End;
  endfor
  $v_0 = x^{(k)}$;
enddo
```

Figure 5: Preconditioned Projection - Minimum Residual Algorithm

6 Numerical Experiments

We present numerical results for each of the algorithms discussed in the previous sections, i.e. the two methods for eigenvalues and eigenvectors - RPP and PP-MR and the small singular vector algorithm SVD-RPP. Our first test example, taken from [11], models the concentration waves in reaction-diffusion interaction of chemical solutions in a tubular reactor. The concentrations $x(t, z)$ and $y(t, z)$ of two chemical components, where $0 \leq z \leq 1$ is the space coordinate and t time, are modeled by the system

$$\frac{\partial x}{\partial t} = \frac{D_x}{L^2}\frac{\partial^2 x}{\partial t^2} + f(x, y) \quad (20)$$

$$\frac{\partial y}{\partial t} = \frac{D_y}{L^2}\frac{\partial^2 y}{\partial t^2} + g(x, y) \quad (21)$$

with Dirichlet boundary conditions. In the so called Brusselator wave model we have

$$f(x, y) = A - (B + 1)x + x^2 y \quad (22)$$

$$g(x, y) = Bx - x^2 y \quad (23)$$

and there exists the trivial stationary solution $\hat{x} = A, \hat{y} = B/A$. The linear stability of this steady state is examined by checking whether the eigenvalues of the Jacobian at $\hat{x}, \hat{y}$ have a positive real part. For this example the problem is anlaytically solvable. In [11] the parameters are taken to be $D_x = .008, D_y = .004, A = 4, B = 5.45$. It is known that at $L \approx .51302$ a pair of purely imaginary eigenvalues appear, causing a Hopf bifurcation to a periodic solution; we take L to be this value.

An eigenvalue of the discretized Jacobian closest to zero was computed using the restarted Arnoldi, RPP and PP-MR algorithms. The SOR iteration was used as a relaxation preconditioning, i.e. the splitting $A = M - N$ in algorithms RPP and PP-MR was

$$M = (D + \omega L) \text{ and } N = (1 - \omega)D - U \quad (24)$$

where $A = L + D + U$ with D diagonal, L strictly lower and U strictly upper triangular. A 100 point discretization was used, resulting in the Jacobian A being a 200×200 matrix. Fig. 6 compares the performance of the three algorithms. A restart parameter of $s = 20$ and relaxation parameter $\omega = .95$ was used. The eigenvalue approximation was updated every 3 steps in algorithm MM-PP, i.e. the parameter $r = 3$.

We used a convergence threshold of 10^{-5}, for the residual norm $||Ax - \lambda x||_2$. Looking at Fig. 6, we first notice that the Arnoldi algorithm failed to converge even after 600 iterations. Algorithm RPP performed better and converged in ≈ 490 iterations, and PP-MR was the fastest, converging in only 215 iterations. We also observe that in all three algorithms the residual norm oscillates. This is because we are restarting after every 20 iterations which reduces the dimension of the subspace and consequently the residual norm initially increases immediately after the restart step before decreasing again. We also notice that PP-MR, which incorporates the minimum residual steps, exhibits a much larger amplitude of oscillation. We do not as yet have an explanation for this, but it appears that the minimum residual steps cause a much more rapid decrease in the residual than the Galerkin steps alone, thus the convergence threshold is reached much faster.

Experiments with algorithm PP-MR using different values of the relaxation parameter ω were also carried out and the results are displayed in Fig. 7. Each line corresponds to experiments with a different starting vector, and two sets of experiments were carried out, with $N = 100$ and $N = 200$ respectively. In the $N = 200$ case the restart parameter was $s = 20$, whereas for $N = 100$ we used $s = 15$. The effect of the value of ω is not quite clear from the experiments, since the starting vector can also make a difference in the convergence. We also mention that in all the experiments, the

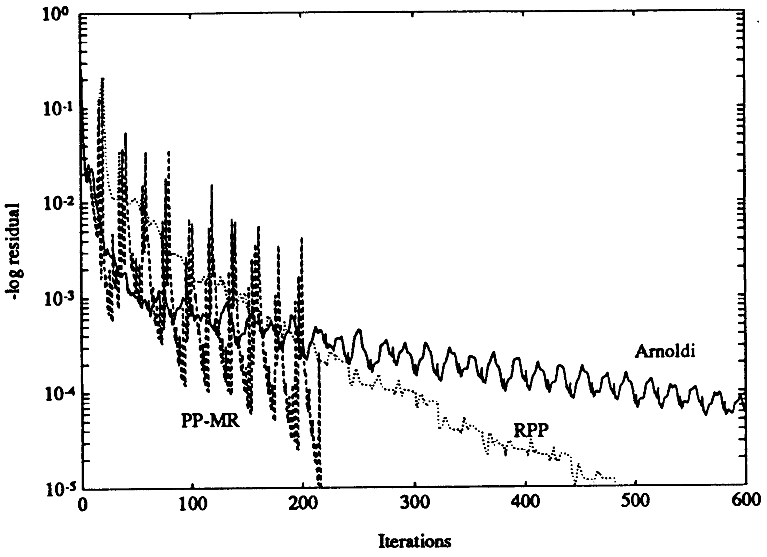

Figure 6: Performance for Brusselator problem, $N = 200, s = 20, w = .95$

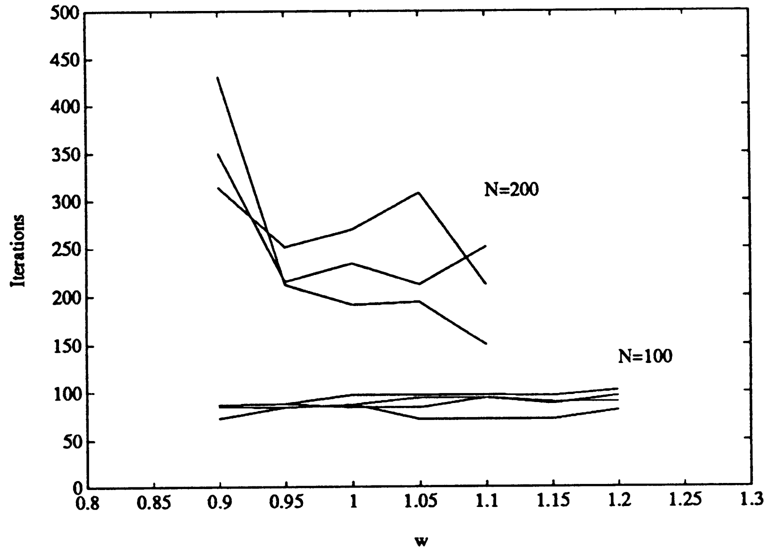

Figure 7: More experiments with Brusselator problem

Iterations of SVD-RPP with different starting vectors				
112	72	83	102	99

Table 1: Performance of SVD-RPP, $N = 100$

Arnoldi algorithm failed to converge, even after 600 iterations in the $N = 200$ case and 300 iterations in the $N = 100$ case.

For the algorithm SVD-RPP, we constructed the following test problem. First we consider the matrix $\hat{A}$

$$\hat{A} = \begin{pmatrix} B & -I & & \\ -I & B & \ddots & \\ & \ddots & \ddots & -I \\ & & -I & B \end{pmatrix} \quad \text{where } B = \begin{pmatrix} 4 & a & \dots & 0 \\ b & 4 & \ddots & \\ & \ddots & \ddots & a \\ 0 & \dots & b & 4 \end{pmatrix} \tag{25}$$

with $a = -1 + \delta$, $b = -1 - \delta$. We take δ, the "asymmetry", to be 0.5. Let λ be an eigenvalue of $\hat{A}$. We create an almost singular matrix A from $\hat{A}$ as

$$A = \hat{A} - \lambda I + \Delta I \tag{26}$$

where Δ is set to a small value (we used $\Delta = 10^{-3}$). The matrix A will have a small singular value of $O(\Delta)$. Algorithm SVD-RPP was used to compute the smallest singular value σ and associated singular vectors u and v of a matrix of size 100 of the form A. A restart parameter $r = 6$ was used and the iteration was carried out till the residual satisfied

$$\|Au - \sigma v\|_2 + \|A^T v - \sigma u\|_2 < 10^{-6}. \tag{27}$$

(Note that using a restart parameter $r = 6$ means that subspaces of dimension at most 12 are searched, since the SVD algorithm uses $2k$ vectors at each step.) The number of iterations required for convergence with five different starting vectors is shown in Table 1.

We also ran the Lanczos bidiagonalization algorithm on the same problem. Without any kind of reorthogonalization, that algorithm failed to converge even after 200 iterations. With full reorthogonalization, (i.e. guaranteeing finite termination in N steps), the residual for the smallest singular value decreased to only .1 after 90 iterations, then decreased sharply because of the finite termination. Of course full reorthogonalization is essentially impractical, since in the final stages of the process we are solving SVD problems of order N at each step! Nevertheless, the extremely slow convergence of Lanczos even with full reorthogonalization for the small singular value is to be noted. The bidiagonalization algorithm can also be used in a restarted manner, i.e. with reorthogonalization with respect to only the previous s vectors. With $s = 12$ and $s = 24$ respectively, the algorithm failed to converge in 200 iterations.

To conclude, we found both algorithms PP-MR for eigenvalues and eigenvectors as well as SVD-RPP for singular values and vectors to perform favorably compared to the traditional methods like Arnoldi and Lanczos bidiagonalization.

7 Conclusions

We have considered some projection methods for computing specific eigenvalues and corresponding eigenvectors of general sparse matrices which are prohibitively expensive to factor.

Traditional Krylov space methods converge slowly to the eigenvalues interior to the spectrum, so we used relaxation preconditioning to accelerate the converence of the projection method in such a situation. The effectiveness of the traditional Galerkin approximation in these cases is also questionable and an alternative, the minimum residual approach, was proposed and shown to be superior. The resulting algorithm was tested on a model problem, the Brusselator wave model, and performed better than the Arnoldi algorithm (which in fact failed to converge). A preconditioned projection algorithm was also proposed for the important problem of computing the small singular vectors and associated singular values of a sparse matrix. Experimental results comparing this algorithm and the Lanczos bidiagonalization algorithm were also presented. The experiments indicate that both these algorithms can be superior to the traditional Arnoldi or Lanczos approaches for certain problems.

An important extension of the preconditioned projection method approach presented here would be to extend the technique to computing invariant subspaces, i.e. instead of computing a specified eigenvalue and eigenvector, one is interested in an invariant subspace $X \in \mathcal{R}^{N \times d}$, such that

$$AX = XB$$

with $B \in \mathcal{R}^{d \times d}$. In this case a number of questions arise. Firstly, it is usually desirable to compute an orthonormal basis for the subspace, which may not neccesarily be a basis of eigenvectors, for example if the eigenvalue is multiple or a real basis is sought for a pair of complex conjugate eigenvalues. The form of relaxation preconditioning to be used is no longer obivious. Another question is an eficient generalization of the minimum residual aproach to this case.

It should also be noted that the relaxation preconditioning presented here may be combined with some form of polynomial preconditioning as well, and this may lead to even faster algorithms. The use of other types of preconditioners like incomplete factorizations or domain decomposition also need to be investigated in the context of eigenproblems. Finally, the algorithms presented here are likely to be well suited for parallel and vector computer architectures because of the reliance on matrix-vector products rather than factoring the matrix; also parallel preconditioners such as relaxation using red-black or multicolored orderings may be used as well.

References

[1] E.R. Davidson. The iterative calculation of a few of the lowest eiegnvalues and the corresponding eigenvectors of large real-symmetric matrices. *J. Comp. Phys.*, 17:87–94, 1975.

[2] A. Dax. The convergence of linear stationary iterative processes for solving singular unstructured systems of linear equations. *SIAM Review*, 32(4):611–635, 1990.

[3] R.W. Freund, M.H. Gutknecht, and N.M. Nachtigal. An implementation of the look-ahead lanczos algorithm for non-hermitian matrices, parts i and ii. Technical Report 90.45 and 90.46, RIACS, Nasa Ames Research Center, 1990.

[4] G. Golub and C. Van Loan. *Matrix Computations*. Johns Hopkins University Press, 1983.

[5] H.B. Keller. On the solution of singular and semidefinite linear systems by iteration. *SIAM J. Num. Anal.*, Ser. B 2:281–290, 1965.

[6] H.B. Keller. Practical procedures in path following near limit points. In R. Glowinski and J.L. Lions, editors, *Computing Methods in Applied Sciences and Engineering*. Academic Press, NY, 1977.

[7] C.D. Meyer and R.J. Plemmons. Convergent powers of a matrix with applications to iterative solution of singular linear systems. *SIAM J. Num. Anal.*, 14:699–705, 1977.

[8] R.B. Morgan and D.S Scott. Generalizations of Davidson's method for computing eigenvalues of sparse symetric matrices. *SIAM J. Sci. Stat. Comp.*, 7(3):817–825, 1986.

[9] A. Ruhe. SOR-methods for the eigenvalue problem with large sparse matrices. *Math. Comp.*, 28(127):695–709, 1974.

[10] Y. Saad. Projection methods for solving large sparse eigenvalue problems. Technical Report 224, Yale University, 1982.

[11] Y. Saad. Numerical solution of large sparse nonsymmetric eigenvalue problems. Technical Report 88.39, RIACS, Nasa Ames Research Center, 1988.

[12] J.H. Wilkinson. *The Algebraic Eigenvalue Problem.* Oxford University Press, London, 1965.