Methods for Large Sparse Eigenvalue Problems from Waveguide Analysis*

Chang Peng and Daniel Boley
Department of Computer Science
University of Minnesota
Minneapolis, MN 55455

Abstract

We discuss several techniques for finding leading eigenvalues and eigenvectors for large sparse matrices. The techniques are demonstrated on a scalar Helmholtz equation derived from a model semiconductor rib waveguide problem. We compare the simple inverse iteration approach with more sophisticated methods, including minimum degree reordering, Arnoldi and Lanczos methods. We then propose a new Arnoldi method designed particularly for the constrained generalized eigenvalue problem, a formulation arising naturally from the scalar waveguide problem.

1 Problem Formulation

In the waveguide analysis (see e.g. [9]), the analysis of the propagation of the electric and magnetic fields in waveguides, based on the use of Maxwell's equations, often lead to scalar Helmholtz equations of the general form $\nabla_T^2 E_x + k^2 E_x = \beta^2 E_x$, where E_x denotes the x component of the electric field, k is the dielectric constant, β is the unknown propagation constant, and ∇_T^2 denotes the Laplacian operator in the transverse (x, y) coordinates (z being the longitudinal coordinate). When this Helmholtz equation is solved for the x coordinate only, one must impose internal continuity conditions across the boundaries between materials of different dielectric constants. The combined equations are then discretized by finite differences yielding a large sparse ordinary matrix eigenvalue problem $A\mathbf{x} = \lambda\mathbf{x}$ [9]. Discretizing the vector fields equations using finite element formulations leads to a large sparse generalized eigenvalue problem [2]:

$$A\mathbf{x} = \lambda B\mathbf{x} \tag{1}$$

where the matrix $A^{n\times n}$ is real and non-singular, $B^{n\times n}$ is real, symmetric, and positive definite.

Solving problem (1) constitutes the largest part of the computational effort. In this paper we are discussing several iterative algorithms that can efficiently deal with problem (1) of several thousands in size on workstations. When B is banded, as is the case in many problem formulations from waveguide analysis, the Cholesky decomposition $B = LL^T$ problem (1) can be used to reduce (1) to an ordinary eigenvalue problem:

$$L^{-1}AL^{-T}\mathbf{y} = \lambda\mathbf{y} \tag{2}$$

where L is lower triangular and $\mathbf{y} = L^T\mathbf{x}$. In an iterative method, $L^{-1}AL^{-T}$ dose not need to be explicitly formed. This is possible because most iterative methods only need to form matrix-vector products. When B is not banded but its Cholesky decomposition gives a sparse L, the reduction is still useful. But if L is not sparse, then solving problems such as (1) becomes significantly more difficult and expensive. Since waveguide analysis leads to generally banded matrices, in this paper, we will mainly concentrate on the ordinary eigenvalue problem (2), assuming that the multiplication by A^{-1} is feasible, and considering the matrix $L^{-1}AL^{-T}$ as A, if appropriate.

As a model test case, we derived the matrix A from the scalar Helmholtz equation, combined with suitable internal continuity conditions, applied to a typical semiconductor rib waveguide. See [9] for the details. In our examples, we used a matrix of modest size (1800×1800) with a zero structure shown in Figure 1. Computations were carried out in MATLAB.

*This work was funded in part by NSF grant CCR-9405380.

2 Inverse Iteration, Minimum Degree Reordering and Deflation

Inverse iteration is a simple, powerful and effective method to compute the smallest modulus eigenvalue. Since in the waveguide analysis the dominant mode (in the sense of largest real part), which is desired, generally does not correspond to the largest or smallest eigenvalue in modulus, it is often necessary to combine a shift σ into the iteration. Iterating with matrix $(A - \sigma I)^{-1}$ will yield the eigenvalue closest to the given σ. The convergence factor is:

$$\rho = \frac{|\lambda_1 - \sigma|}{|\lambda_2 - \sigma|} \tag{3}$$

where λ_1 is the closet eigenvalue to σ and λ_2 is the second close one. We can see that if σ is much closer to λ_1 than to λ_2, the convergence can be very fast. Each iteration requires that a system of linear equations of the form $(A - \sigma I)\mathbf{x}_{\text{new}} = \mathbf{x}_{\text{old}}$ be solved, often by a variant of LU Factorization [4]. If σ is updated using Rayleigh quotients during the iteration, then the iteration converges quadratically in general, and cubically if the matrix is symmetric [4], but changing σ requires that $(A - \sigma I)$ be re-factored from scratch in each iteration at great expense. So often the shift is fixed, allowing the factoring to be done just once, where the shift is estimated in advance by physical considerations. On the other hand, when only the largest eigenvalue in magnitude is needed, the more straightforward power iteration may be used, but this method may suffer from slow convergence when the first and second eigenvalues in magnitude are close to each other. The Inverse Iteration is summarized in the following.

Algorithm 2.1 Basic Inverse Iteration:
1. Choose a random unit vector $\mathbf{x}_0$ and a shift σ;
2. Compute the LU decomposition of $A - \sigma I$;
3. Iterate: for $i = 1, 2, ...$ until convergence do
4. Solve $Ly = \mathbf{x}_0$ and $U\mathbf{x} = \mathbf{y}$;
5. $norm = \|\mathbf{x}\|_2$, $\mathbf{x} = \mathbf{x}/norm$;
6. Compute $\lambda = \mathbf{x}^T A\mathbf{x} = (\mathbf{x}^T\mathbf{x}_0)/norm$;
7. Compute residual $r = \|Ax - \lambda\mathbf{x}\|_\infty$;
8. If r is satisfied, then stop; Otherwise $\mathbf{x}_0 = \mathbf{x}$;
9. *Optional* If $i = 0$ (mod 8 or 10), then set $\sigma = \lambda$,
   and compute the LU decomposition of $(A - \sigma I)$.
   (optional: for Rayleigh quotient iteration adaptive algorithm)
10. end

Line 9 is an adaptive shift step for faster convergence which can yield quadratic convergence (cubic if A is symmetric). However, it requires that the LU decomposition be repreated, which is very expensive, and especially when A is not symmetric, the converged eigenvalue may not be the one that is closest to the original shift. Thus often line 9 is skipped, or performed only once during several passes through the main loop.

If the matrix A is banded, we can reorder A using the minimum degree ordering before we do the LU decomposition [3]. This is a heuristic algorithm that reorders the equations so that (hopefully) the resulting L and U factors are more sparse, i.e. have fewer non-zero elements. Then the total number of flops needed by the algorithm will be considerably reduced. For the details of minimum degree reordering, please refer to [3]. For these experiments, we have taken advantage of MATLAB's built-in functions for various reordering algorithms including the minimum degree reordering.

Algorithm 2.2 Inverse Iteration with Minimum Degree Reordering:
1. Choose a random unit vector $\mathbf{x}_0$ and a shift σ;
2. Find the minimum degree reordering permutation P, replace A with PAP;
3. Compute the LU decomposition of $A - \sigma I$;
4. The rest is the same as the Basic Inverse Iteration (Algorithm 2.1, steps 3–8).

With MATLAB we carried out a series of numerical experiments on a 1800×1800 matrix described in Section 1. Algorithm 2.1 and 2.2 were used respectively to find the dominant mode which corresponds to the eigenvalue

Inverse Iteration	*# Eigenvalues Sought*	*Iters*	*Flop Count*	*Residual Norm*
Basic	1	4	8,255,268	6.4E-5
Reordering	1	4	5,225,559	7.4E-5
Reordering	2	18	8.485,461	3.9E-5

Table 1: Comparison of Basic and Reordering Inverse Iteration

$\lambda = 3.4404$, fixing the shift at $\sigma = 3.4$. From Table 1 we can see that reordered inverse iteration can save over $1/3$ of computation needed by the basic inverse iteration. We should point out that in MATLAB the LU decomposition process may involve pivoting and this pivoting may reduce some of the advantage gained from reordering. To see the potential advantage that could be gained just from minimum degree ordering if pivoting is turned off, we computed the flop count for the Cholesky factorization on a shifted matrix A (like LU Decomposition without pivoting for symmetric positive definite matrices [4]). The resulting Flop Counts with and without minimum degree reordering are $2,013,624$ and $3,685,377$, respectively, representing a savings of 45% over the unreordered version!

If we have already obtained the leading eigenvalue and need to find the next one closest to the known eigenvalue, we can use the so called *deflation* technique, described as follows [8]. Suppose λ and $\mathbf{u}$ are the known eigenpair of A, and $\mathbf{v}$ is a vector such that $\mathbf{v}^T\mathbf{u} = 1$. Then it is easy to see that the matrix $A' = A - \gamma uv^T$ has the same spectrum as A except that the one eigenvalue λ has been shifted to $\lambda - \gamma$. For the power iteration A' can be used directly in the iteration. But we cannot do so in the case of the inverse iteration, because forming A' explicitly is too expensive and will destroy the sparsity. We here propose an efficient process that avoids forming A'. Since what the inverse iteration needs is the matrix-vector product $\mathbf{x} = A'^{-1}\mathbf{x}_0$. We express A'^{-1} in terms of A^{-1}, whose LU decomposition is already available. By the Sherman-Morrison formula [4] and using $A^{-1}\mathbf{u} = \lambda^{-1}\mathbf{u}$, we have

$$\begin{aligned} A'^{-1} &= A^{-1} + \gamma A^{-1}\mathbf{u}(1 - \gamma\mathbf{v}^T A^{-1}\mathbf{u})^{-1}\mathbf{v}^T A^{-1} \\ &= A^{-1} + \frac{\gamma}{\lambda - \gamma} uv^T A^{-1} \\ &= (I + \beta uv^T)A^{-1} \end{aligned} \tag{4}$$

where $\beta = \frac{\gamma}{\lambda - \gamma}$. Since we already have LU decomposition of A, we can first compute $\mathbf{y} = A^{-1}\mathbf{x}_0$, and then compute the product $\mathbf{x} = A'^{-1}\mathbf{x}_0 = \mathbf{y} + \beta(\mathbf{v}^T\mathbf{y})\mathbf{u}$. Combining this into Algorithm 2.1 and 2.2 is the inverse iteration with deflation. We would like to point out here that this deflation technique can be cascaded multiple time, or generalized to the block version, e.g. deflation of multiple eigenvalues at the same time, by using the Sherman-Morrison-Woodbury formula [4].

There are various choices for the vector $\mathbf{v}$. The Hotelling's deflation chooses $\mathbf{v}$ as the left eigenvector $\mathbf{w}$, e.g. $\mathbf{w}^T A = \lambda\mathbf{w}^T$ or $A^T\mathbf{w} = \lambda\mathbf{w}$. The advantage of this choice is that A' will have the same eigenvectors as A. So when A is symmetric, the right and left eigenvectors are the same, we can more easily compute the product:

$$\begin{aligned} \mathbf{x} &= A'^{-1}\mathbf{x}_0 \\ &= (I + \beta uv^T)A^{-1}\mathbf{x}_0 \\ &= \mathbf{y} + (\lambda - \gamma)^{-1}(\mathbf{u}^T\mathbf{x}_0)\mathbf{u} \end{aligned} \tag{5}$$

where $\mathbf{y} = A^{-1}\mathbf{x}_0$. The other common choice just takes $\mathbf{v} = \mathbf{u}$. But this choice will not preserve the eigenvectors. So after the eigenvalue is found, we need another iteration to determine the eigenvector.

The other strategy to compute more than one eigenpair is block inverse iteration (also known as inverse simultaneous iteration). In this iteration, we choose a shift σ and several random vectors at the start, say $X_0 = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_m]$, and then for $i = 1, 2, ...$ until convergence, iteratively compute:

$$\begin{aligned} X_i &= (A - \sigma I)^{-1} X_{i-1} \\ X_i &= Q_i R_i \\ X_{i+1} &= Q_i \end{aligned} \tag{6}$$

Equation (6) is the result of the QR decomposition in the previous line which orthogonalizes the columns of X_i. If the iteration converges, then the upper triangular matrix R_k will tend to be the leading $k \times k$ part of the Schur canonical form of $(A - \sigma I)^{-1}$ [8], that is the diagonal elements of R_k will be close to the first k eigenvalues of $(A - \sigma I)^{-1}$ in order of magnitude. So the eigenvalues nearest to σ are obtained. The matrix Q_k will be close to the corresponding Schur vectors. If $\mathbf{y}$ is a eigenvector of R_k, then the Ritz vector $\mathbf{x} = Q_k\mathbf{y}$ is an approximate eigenvector of A.

Algorithm 2.3 Block Inverse Iteration:
1. Choose a set of random vectors $X_0 = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_m]$ and a shift σ;
2. Compute the LU decomposition of $A - \sigma I$;
3. Iterate: for $i = 1, 2, ...$ until convergent do
4. Solve $LY = X_0$ and $UX = X_0$;
5. Compute QR decomposition $X = QR$;
6. If $i = 0$ mod k,
7. Compute the residual norm $\|AQ - QR\| = \|AQ - X\|$.
8. To compute AQ, solve $LZ = Q$ and $UY = Z$.
9. If satisfied, then stop. Otherwise, $X_0 = Q$ and go on iteration;
7. end
8. Compute the eigenvector $\mathbf{y}$ of R, and the eigenvector of A is $\mathbf{x} = Q\mathbf{y}$.

Note that the number m of iterated vectors should be larger than the number k of eigenvalues needed. Depending on the spectrum distribution, normally we could take $m = k + 2$ if k is two or three.

3 Arnoldi and Lanczos Method

Arnoldi method for the algebraic eigenvalue problem is very much like Galerkin method for the approximation of eigenfunctions. Here we just give a brief derivation of the method. The Galerkin method uses a subspace to form the approximate eigenfunctions. In a similar way, the Arnoldi method takes a subspace $\mathcal{K} \subseteq R^n$ to extract approximate eigenvectors. The subspace is chosen to be the Krylov space of the form $\mathcal{K} = \{\mathbf{v}_0, A\mathbf{v}_0, ..., A^{m-1}\mathbf{v}_0\}$, where $\mathbf{v}_0$ is an arbitrary vector and m is typically much smaller than n. Suppose we have a set of orthonormal vectors $\mathbf{v}_i$ that form the basis of $\mathcal{K}$, and denote matrix $V_m = [\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_m]$. Take $\mathbf{x} = V_m\mathbf{y} \in \mathcal{K}$, and impose the Galerkin condition to the residual $r(\lambda, \mathbf{x})$:

$$r(\lambda, \mathbf{x}) = A\mathbf{x} - \lambda\mathbf{x} - \mathcal{K} \tag{7}$$

which is equivalent to:

$$(AV_m\mathbf{y} - \lambda V_m\mathbf{y}, \mathbf{v}_i) = 0, \text{i} = 1,2, ..., \text{m} \tag{8}$$

Write (8) in the matrix form and denote $H_m = V^T AV$, we then obtain a reduced approximation problem of smaller size:

$$H_m\mathbf{y} = \lambda\mathbf{y} \tag{9}$$

Now the question is how to form the orthonormal basis $\mathbf{v}_i$ of $\mathcal{K}$. Here, the modified Gram-Schmidt orthogonalization process can be used. Each $A\mathbf{v}_j$ is generated, it is orthogonalized against all $\mathbf{v}_i$ with $i < j$. It is now known that this reduced problem can indeed give good approximation to large modulus eigenvalues [7]. Below is the algorithm.

Algorithm 3.1 Arnoldi Algorithm:
1. Choose an random unit vector $\mathbf{v}_1$;
2. Iterate: for $j = 1, 2, ..., m$ do
3. Compute $\mathbf{w}_j = A\mathbf{v}_j$;
4. For $i = 1, 2, ..., j$ do
5. $h_{ij} = (\mathbf{v}_i, \mathbf{w}_j)$;

6. $\mathbf{w}_j = \mathbf{w}_j - h_{ij}\mathbf{v}_i$;
7. end
8. $h_{j+1,j} = \|\mathbf{w}_j\|$. If $h_{j+1,j} = 0$, then go to line 11;
9. $\mathbf{v}_{j+1} = \mathbf{w}_j / h_{j+1,j}$;
10. end
11. Compute all the eigenpairs $\lambda, \mathbf{y}$ of H_m;
12. Select the first k λ's in order of magnitude, and compute the eigenvectors $\mathbf{x}_i = V\mathbf{y}_i (i = 1, 2, ...k)$;
13. Compute the residual $\|A\mathbf{x}_i - \lambda_i \mathbf{x}_i\|_\infty = (\mathbf{e}_m^T \mathbf{y}_i) h_{m+1,m}$. If satisfied, then stop;
Otherwise increase m by 5 or 10, set $\mathbf{v}_0 = \mathbf{x}_1$ and restart iteration from line 2 again.

In line 13 the residual expression is obtained by multiplying $\mathbf{y}_i$, the eigenvector of H_m, to the following equation, which comes straight from line 2 to 9.

$$AV_m = V_m H_m + h_{m+1,m}\mathbf{v}_{m+1}\mathbf{e}_m^T \tag{10}$$

In line 8, if $h_{j+1,j} = 0$, then equation (10) will becomes

$$AV_j = V_j H_j \tag{11}$$

The above equation indicates the exact invariant subspace V_j of A is obtained, and therefore the eigenvalues from line 11 will be the exact eigenvalues of A. In practice, this is a rare case. The choice of m depends on the problem size, spectrum distribution and the number of eigenvalues wanted. For a matrix of several thousands in size, choosing $m = 20 \sim 200$ may be good enough if about first $4 \sim 10$ top modulus eigenvalues are wanted.

The shift and inverse strategy as previously described for the inverse iteration can also be naturally combine into Algorithm 2.1. In case the eigenvalues near a given point are wanted, this can be the winning strategy, as the numerical experiment shows.

Algorithm 3.2 Shift and Inverse Arnoldi Algorithm:
1. Choose an random unit vector $\mathbf{v}_1$ and shift σ;
2. Compute the LU decomposition of $(A - \sigma I)$;
3. Iterate: for $j = 1, 2, ..., m$ do
4. Solve $L\mathbf{y} = \mathbf{v}_j$ and $U\mathbf{w} = \mathbf{y}$, $\mathbf{w}_j = \mathbf{w}$;
5. Continue as in Algorithm 3.1, steps 4–9.
6. Compute eigenvalues and vectors of $(A - \sigma I)^{-1}$ as in Algorithm 3.1, steps 11–12,
mapping the eigenvalues back to those of A.

While the Arnoldi method reduces the large matrix A into a small upper Hessenberg matrix H_m, the Lanczos method, however, reduces the matrix into a small tridiagonal matrix, and solve the tridiagonal matrix for eigenvalues as the approximation [1]. In the Lanczos method the matrix V_m will not be orthonormal. Instead, another $n \times m$ matrix W_m will be formed such that $W_m^T V_m = I_m$ and $W_m^T A V_m = T_m$, where T_m is the tridiagonal matrix. The shift and inverse strategy can be incorporated into the Lanczos as Algorithm 3.2.

Algorithm 3.3 Lanczos Algorithm
1. Choose two vectors $\mathbf{v}_1$ and $\mathbf{w}_1$ such that $(\mathbf{v}_1, \mathbf{w}_1) = 1$ Set $\beta_1 = 0, \mathbf{v}_0 = \mathbf{w}_0 = 0$
2. Iterate: for $i = 1, 2, ..., m$ do
3. $\alpha_i = (A\mathbf{v}_i, \mathbf{w}_i)$;
4. $\mathbf{v} = A\mathbf{v}_i - \alpha_i \mathbf{v}_i - \beta_i \mathbf{v}_{i-1}$;
5. $\mathbf{w} = A^T \mathbf{w}_i - \alpha_i \mathbf{w}_i - \delta_i \mathbf{w}_{i-1}$;
6. $\delta_{i+1} = |(\mathbf{v}, \mathbf{w})|^{1/2}$;
7. $\beta_{i+1} = (\mathbf{v}, \mathbf{w}) / \delta_{i+1}$;
8. $\mathbf{v}_{i+1} = v / \beta_{i+1}$;
9. $\mathbf{w}_{i+1} = w / \delta_{i+1}$;
10. end
11. Compute the eigen pairs of $T_m = [\delta_i, \alpha_i, \beta_i]$;
12. Select the first k eigenvalues λ_i in order of their magnitude;

Iterative Method	k	m	*Flop Count*	*Residual Norm*
Arnoldi	4	15	9,308,634	4.1E-4
Lanczos	4	17	12,365,568	1.5E-5
Block Inverse itr.	3	*n.a.*	53,560,729	4.4E-5

Table 2: Comparison of FLOPS by Arnoldi and Lanczos. k is the number of eigenvalues sought, and m is the number of vectors generated.

13. Compute the corresponding eigenvectors $\mathbf{x}_i = V_m \mathbf{y}_i$;
14. Compute the residual norm $\|A\mathbf{x}_i - \lambda_i \mathbf{x}_i\|_\infty$;
15. If satisfied, then stop. Otherwise, increase m and start over again.

We used Arnoldi, Lanczos and block inverse iteration to solve the same problem as in Section 2. Shift and inverse scheme and the minimum degree ordering were all combined in each algorithm. And five eigenvalues closest to the shift $\sigma = 3.4$ were computed. The results are $\lambda : 3.4408, 3.5665, 3.1464, 3.7539, 2.8379$. Table 2 gives the comparison of flops. It is apparent that the Arnoldi and Lanczos methods are capable of computing more eigenvalues with less cost than block inverse iteration, but inverse iteration may suffice if just one eigenvalue is desired, depending on the distribution of the spectrum. When the matrix is symmetric, the the Arnoldi and Lanczos methods reduce to the same algorithm, and the resulting algorithm is known by the Kaniel-Paige theory to have very nice convergence properties, at least for the extreme eigenvalues [4]. When the matrix is nonsymmetric, the Arnoldi method is more straightforward to implement and has somewhat better convergence properties, but requires more space and cost per iteration than the Lanczos method. In addition, it is not entirely a simple matter to compute the eigenvalues of the nonsymmetric tridiagonal matrix resulting from the Lanczos method. Deflation can be incorporated into both Arnoldi and Lanczos methods.

4 Constrained Raleigh Quotient Maximization Problems

In the transverse magnetic field formulation using finite element method [5], the following problem is encountered:

$$\text{Find } (\lambda, \mathbf{x}) \text{ s.t.} \quad \lambda = \max_{\substack{C^T\mathbf{x}=0, \\ \mathbf{x}\neq 0}} \frac{(A\mathbf{x}, \mathbf{x})}{(B\mathbf{x}, \mathbf{x})} \tag{12}$$

where A is symmetric, B is symmetric positive definite, and C has full column rank. This problem would be the usual generalized eigenvalue problem if there were no constraint $C^T x = 0$. We call (12) Constrained Raleigh Quotient Maximization Problem(CRQM). Mathematically, (12) is equivalent to the following regular eigenvalue problem:

$$V^T A V \mathbf{y} = V^T B V \mathbf{y} \tag{13}$$

where the matrix V consists of an orthonormal basis spanning the constraint space $\{\mathbf{x} : C^T\mathbf{x} = 0\}$. In case that the matrix size is small, we can determine the matrix V by the Singular Value Decomposition and solve the dense problem (13). However, if the matrix size is as large as several thousands, this approach becomes computationally intractable on workstations. Here, we propose an algorithm that can efficiently deal with large sparse case of CRQM.

The algorithm is based on the Arnoldi. The regular Arnoldi forms an orthonormal basis of the Krylov space $\mathcal{K}$. As each new basis vector $A\mathbf{v}_j$ is generated, it is orthogonalized against all $\mathbf{v}_i, 1 \leq i \leq j-1$. Now we modify this process by making $A\mathbf{v}_j$ orthogonal not only to all previous $\mathbf{v}_i$'s but also orthogonal to all the column vectors of C. The column vectors of the resulting matrix V_m will be the basis of a "projected Krylov space" and will be very good components to form approximate solution vectors for CRQM.

Algorithm 4.1 Arnoldi Algorithm for CRQM:

1. Choose a unit vector $\mathbf{v}_1$ such that $C^T\mathbf{v}_1 = 0$;
2. For $j = 1, 2, ..., m$ do
3. Compute $\mathbf{w}_j = A\mathbf{v}_j$;
4. For $i = 1, 2, ..., k$ do
5. $t_{ij} = (\mathbf{c}_i, \mathbf{w}_j)$;
6. $\mathbf{w}_j = \mathbf{w}_j - t_{ij}\mathbf{c}_i$;
7. end
8. For $i = 1, 2, ..., j$ do
9. $h_{ij} = (\mathbf{v}_i, \mathbf{w}_j)$;
10. $\mathbf{w}_j = \mathbf{w}_j - h_{ij}\mathbf{v}_i$;
11. end
12. $h_{j+1,j} = \|\mathbf{w}_j\|_2$. If $h_{j+1,j} = 0$, then go to 14;
13. $\mathbf{v}_{j+1} = \mathbf{w}_j / h_{j+1,j}$.
14. Compute the eigenvalues of H_m and choose the k largest λ_is, where $k \leq m$.
15. Compute eigenvectors $\mathbf{y}_i$ of H_m associated with λ_i, and the CRQM solution vector $\mathbf{x}_i = V_m\mathbf{y}_i$, $i = 1, \ldots, k$.

We have the following relations for the above algorithm.

$$AV_m = V_m H_m + CT_m + h_{m+1,m}\mathbf{v}_{m+1}\mathbf{e}_m^T \quad (14)$$

$$V_m^T A V_m = H_m \quad (15)$$

$$C^T V_m = 0 \quad (16)$$

Notice that equation (15) lays the groundwork for the algorithm that the Ritz vector $\mathbf{x} = V_m\mathbf{y}$ is an approximation solution of CRQM.

It is seen that the Arnoldi method is easily adapted to incorporate the constraints. However, this method is still under investigation, and though we expect the behavior to be very similar to the unconstrained Arnoldi method, further numerical experiments are needed. In addition, the convergence theory has not yet been developed for this specific algorithm. This work is in progress and will be reported soon.

5 Conclusions

Arnoldi and Lanczos methods are promising for the large sparse eigenvalue problems in waveguide analysis. The computation cost of these two methods is far less than that of inverse iteration, especially when several eigenvalues are sought. Strategies such as reordering, or "shift and inverse," can be easily combined in these methods. And either of them can be modified to fit new formulations of finite element method. Reordering technique can significantly save computation, and should be explored whenever possible. We have also found MATLAB to be a convenient and efficient tool for problems with size reaching of several thousands.

References

[1] D. L. Boley. Krylov space methods on state-space control models. *Circ., Syst. & Signal Proc.*, 1992. to appear.

[2] Bernice M. Dillon and Jon P. Webb, "A comparison of Formulations for the Vector Finite Element Analysis of Waveguides", *IEEE Trans. Microwave Theory Tech.*, vol. 42, pp. 308-316, Feb. 1994.

[3] Alan George and Joseph W. Liu, *Computer Solution of Large Sparse Positive Definite Systems.* Prentice Hall, Englewood Cliffs, 1981.

[4] Gene H. Golub and Charles F. Van Loan, *Matrix Computations*, John Hopkins University Press, 1989.

[5] Zine-Eddine Abid, Klein L. Johnson, and Anand Gopinath, "Analysis of Dielectric Guides by Vector Transverse Magnetic Field Finite Elements", *J. of Lightwave Tech.*, vol. 11, pp. 1545-1549.

[6] B. N. Parlett, *The Symmetric Eigenvalue Problem*. Prentice Hall, Englewood Cliffs, 1980.

[7] Y. Saad, "Variations on Arnoldi's method for computing eigenelements of large unsymmetric matrices", *Linear Algebra Appl.*, 34:269-295, 1980.

[8] Y. Saad, *Numerical Methods for Large Eigenvalue Problems*, pp. 153-156. Manchester University Press, 1992.

[9] M.S. Stern, "Semivectorial Polarised finite difference method for Optical Waveguides with Arbitrary Index Profiles", *IEE PROCEEDINGS*, vol .135, Pt.J, No.1, pp. 56-63, Feb.

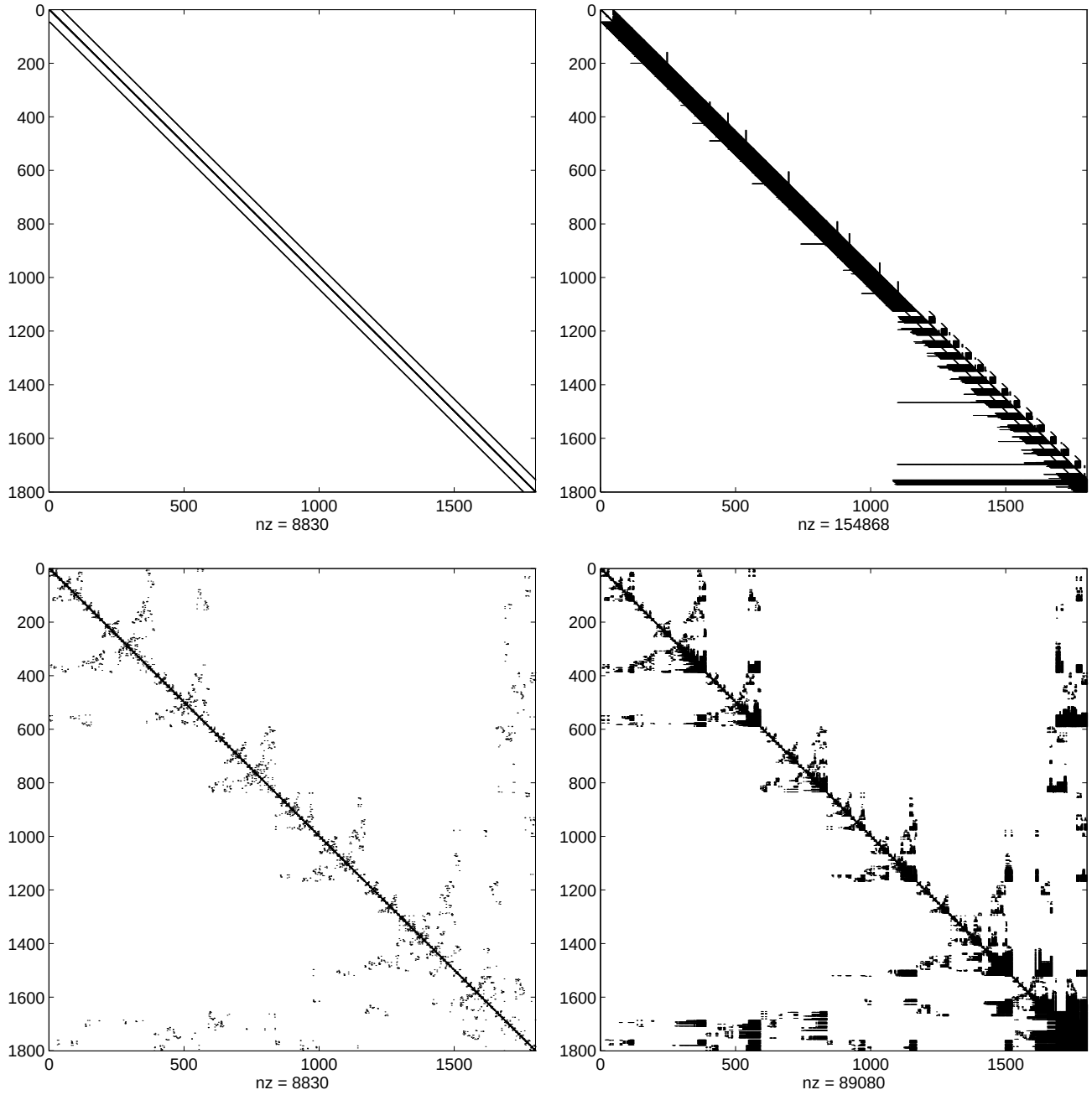

Figure1. Matrix Non-Zeroes showing original matrix and LU factors combined. Upper figures show original order, lower figures show minimum degree ordering. `nz` is the total number of nonzeroes.