A Comparison of Factorization-Free Eigensolvers with Application to Cavity Resonators

Peter Arbenz

Institute of Scientific Computing, ETH Zentrum, CH-8092 Zurich, Switzerland
arbenz@inf.ethz.ch

Abstract. We investigate eigensolvers for the generalized eigenvalue problem $A\mathbf{x} = \lambda M\mathbf{x}$ with symmetric A and symmetric positive definite M that do not require matrix factorizations. We compare various variants of Rayleigh quotient minimization and the Jacobi-Davidson algorithm by means large-scale finite element problems originating from the design of resonant cavities of particle accelerators.

1 Introduction

In this paper we consider the problem of computing a few of the smallest eigenvalues and corresponding eigenvectors of the generalized eigenvalue problem

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

without factorization of neither A nor M. Here, the real n-by-n matrices A and M are symmetric and symmetric positive definite, respectively.

It if is feasible the Lanczos algorithm combined with a spectral transformation [1] is the method of choice for solving (1). The spectral transformation requires the solution of a linear system $(A - \sigma M)\mathbf{x} = \mathbf{y}$, $\sigma \in \mathbb{R}$, which may be solved by a direct or an iterative system solver. The factorization of $A - \sigma M$ may be impossible due to memory constraints. The system of equations can be solved iteratively. However, the solution must be computed very accurately, in order that the Lanczos three-term recurrence remains correct.

In earlier studies [2, 3] we found that for large eigenvalue problems the Jacobi-Davidson algorithm [4] was superior to the Lanczos algorithm or the restarted Lanczos algorithm as implemented in ARPACK [5]. While the Jacobi-Davidson algorithm retains the high rate of convergence it only poses small accuracy requirements on the solution of the so-called correction equation, at least in the initial steps of iterations.

In this paper we continue our investigations on eigensolvers and their preconditioning. We include block Rayleigh quotient minimization algorithms [6] and the locally optimal block preconditioned conjugate gradient (LOBPCG) algorithm by Knyazev [7] in our comparison.

We conduct our experiments on an eigenvalue problem originating in the design of the new RF cavity of the 590 MeV ring cyclotron installed at the Paul Scherrer Institute (PSI) in Villigen, Switzerland.

P.M.A. Sloot et al. (Eds.): ICCS 2002, LNCS 2331, pp. 295–304, 2002.

2 The application: the cavity eigenvalue problem

After separation of time/space variables and after elimination of the magnetic field intensity the Maxwell equations become the eigenvalue problem

$$\mathbf{curl}\,\mathbf{curl}\,\mathbf{e}(\mathbf{x}) = \lambda\mathbf{e}(\mathbf{x}), \quad \operatorname{div}\mathbf{e} = 0, \quad \mathbf{x}\in\Omega, \qquad \mathbf{n}\times\mathbf{e} = \mathbf{0}, \quad \mathbf{x}\in\partial\Omega. \tag{2}$$

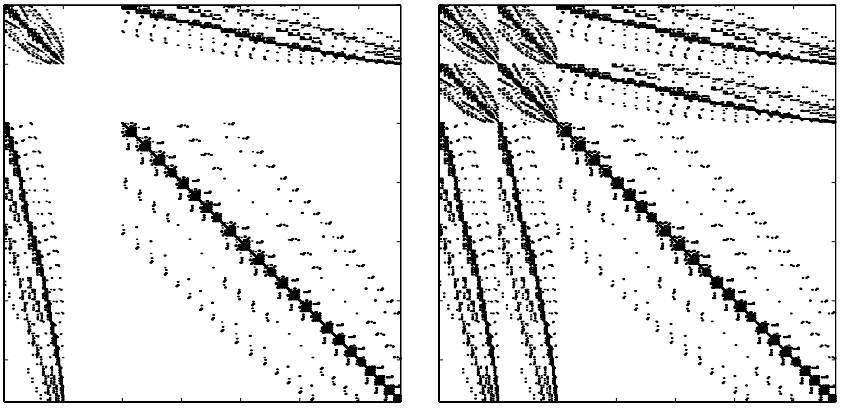
where $\mathbf{e}$ is the electric field intensity. We assume that Ω, the cavity, is a simply connected, bounded domain in $\mathbb{R}^3$ with a polyhedral boundary $\partial\Omega$. Its inside is all in vacuum and its metallic surfaces are perfectly conducting. Following Kikuchi [8] we discretize (2) as

$$\begin{aligned} &\textit{Find } (\lambda_h, \mathbf{e}_h, p_h)\in\mathbb{R}\times N_h\times L_h \textit{ such that } \mathbf{e}_h\neq\mathbf{0} \textit{ and}\\ &\text{(a) } (\mathbf{curl}\,\mathbf{e}_h, \mathbf{curl}\,\boldsymbol{\Psi}_h) + (\mathbf{grad}\,p_h, \boldsymbol{\Psi}_h) = \lambda_h(\mathbf{e}_h, \boldsymbol{\Psi}_h), \qquad \forall\boldsymbol{\Psi}_h\in N_h\\ &\text{(b) } (\mathbf{e}_h, \mathbf{grad}\,q_h) = 0, \qquad \forall q_h\in L_h \end{aligned} \tag{3}$$

where $N_h\subset H_0(\mathbf{curl};\Omega) = \{\mathbf{v}\in L^2(\Omega)^3 \mid \mathbf{curl}\,\mathbf{v}\in L^2(\Omega)^3, \mathbf{v}\times\mathbf{n} = \mathbf{0} \text{ on } \partial\Omega\}$ and $L_h\subset H_0^1(\Omega)$. The domain Ω is triangulated by tetrahedrons. In order to avoid spurious modes we choose the subspaces N_h and L_h, respectively, to be the Nédélec (or edge) elements $N_h^{(k)}$ of degree k [9–11] and the well-known Lagrange (or node-based) finite elements consisting of piecewise polynomials of degree $\leq k$. In this paper we exclusively deal with $k = 2$. In order to employ a multilevel preconditioner we use hierarchical bases and write [11]

$$N_h^{(2)} = N_h^{(1)}\oplus\bar{N}_h^{(2)}, \qquad L_h^{(2)} = L_h^{(1)}\oplus\bar{L}_h^{(2)}. \tag{4}$$

Let $\{\boldsymbol{\Phi}_i\}_{i=1}^n$ be a basis of $N_h^{(2)}$ and $\{\varphi_l\}_{l=1}^m$ be a basis of $L_h^{(2)}$. Then (3) defines



(a) (b)

Fig. 1. Example of matrices A (a) and M (b) for quadratic edge elements.

the matrix eigenvalue problem

$$\begin{bmatrix} A & C \\ C^T & O \end{bmatrix} \begin{pmatrix} \mathbf{x} \\ \mathbf{y} \end{pmatrix} = \lambda \begin{bmatrix} M & O \\ O & O \end{bmatrix} \begin{pmatrix} \mathbf{x} \\ \mathbf{y} \end{pmatrix}, \tag{5}$$

respectively, where A and M are n-by-n and C is n-by-m with elements

$$a_{i,j} = (\mathbf{curl}\, \mathbf{\Phi}_i, \mathbf{curl}\, \mathbf{\Phi}_j), \quad m_{i,j} = (\mathbf{\Phi}_i, \mathbf{\Phi}_j), \quad c_{i,l} = (\mathbf{\Phi}_i, \mathbf{grad}\, \varphi_l).$$

The non-zero structures of A and M are depicted in Fig. 1. The block of zero columns / columns in A corresponds to those curl-free basis functions of $\bar{N}_h^{(2)}$ that are the gradient of some $\varphi \in \bar{L}_h^{(2)}$. The non-zero block in the upper-left corner of A corresponding to the basis functions of $N_h^{(1)}$ is rank-deficient. The deficiency equals $\dim L_h^{(1)}$. The reason for approximating the electric field $\mathbf{e}$ by Nédélec elements and the Lagrange multipliers by Lagrange finite elements is that [9, §5.3]

$$\mathbf{grad}\, L_h^{(k)} = \left\{ \mathbf{v}_h \in N_h^{(k)} \mid \mathbf{curl}\, \mathbf{v}_h = \mathbf{0} \right\}. \tag{6}$$

By (3)(b), $\mathbf{e}_h$ is in the orthogonal complement of $\mathbf{grad}\, L_h^{(k)}$. On this subspace A in (5) is positive definite. Notice that $\mathbf{e}_h$ is divergence-free only in a discrete sense. Because of (6) we can write $\mathbf{grad}\, \varphi_l = \sum_{j=1}^{n} \eta_{jl} \mathbf{\Phi}_j$, whence

$$(\mathbf{\Phi}_i, \mathbf{grad}\, \varphi_l) = \sum_{j=1}^{n} (\mathbf{\Phi}_i, \mathbf{\Phi}_j) \eta_{jk} \qquad \text{or} \qquad C = MY, \tag{7}$$

where $Y = ((\eta_{jk})) \in \mathbb{R}^{n \times m}$. In a similar way one obtains

$$H := C^T Y = Y^T M Y, \qquad h_{kl} = (\mathbf{grad}\, \varphi_k, \mathbf{grad}\, \varphi_l). \tag{8}$$

H is the system matrix that is obtained when solving the Poisson equation with the Lagrange finite elements $L_h^{(k)}$. Notice that Y is very sparse. We have already mentioned that the gradient of a basis function $\varphi_k \in \bar{L}_h^{(2)}$ is an element of the first set of basis functions of $\bar{N}_h^{(2)}$. So, m_1 rows of Y have a single entry 1. The gradient of the piecewise linear basis function corresponding to vertex k, say, is a linear combination (with coefficients ± 1) of the basis functions of $N_h^{(1)}$ whose corresponding edge has vertex k as one of its endpoints.

Equation (7) means that $C^T \mathbf{x} = \mathbf{0}$ is equivalent to requiring $\mathbf{x}$ to be M-orthogonal to the eigenspace $\mathcal{N}(A) = \mathcal{R}(Y)$ corresponding to the eigenvalue 0. Thus, the solutions of (5) are precisely the eigenpairs of

$$A\mathbf{x} = \lambda M \mathbf{x} \tag{9}$$

corresponding to the *positive* eigenvalues. We could therefore compute the desired eigenpairs $(\lambda_j, \mathbf{x}_j)$ of (5) by means of (9) alone. The computed eigenvectors corresponding to positive eigenvalues would automatically satisfy the constraint $C^T \mathbf{x}_j = \mathbf{0}$. This is actually done if the linear systems of the form $(A - \sigma M)\mathbf{x} = \mathbf{y}$ that appear in the eigensolver can be solved by direct methods. Numerical experiments showed however that the high-dimensional zero eigenspace has a negative effect on the convergence rates if *preconditioned iterative* solvers have to be used.

3 Positive definite formulations of the eigenvalue problem

In this section we present two alternative formulations for the matrix eigenvalue problem (5) that should make its solution easier. The goal is to have generalized eigenvalue problems with both matrices positive definite.

3.1 The nullspace method

We can achieve this goal in a straightforward way by performing the computations in the space that is spanned by the eigenvectors corresponding to the positive eigenvalues. This space is $\mathcal{R}(Y)^{\perp_M} = \mathcal{R}(C)^{\perp} = \mathcal{N}(C^T)$, i.e. the nullspace of C^T whence its name. Formally, we solve

$$A|_{\mathcal{N}(C^T)}\mathbf{x} = \lambda M|_{\mathcal{N}(C^T)}\mathbf{x}. \tag{10}$$

In the iterative solvers, we apply the M-orthogonal projector $P_{\mathcal{N}(C^T)} = I - YH^{-1}C^T$ onto $\mathcal{N}(C^T)$ whenever a vector is not in this space, i.e. for generating starting vectors and after solving with the preconditioner. The eigenvalue problem (10) has dimension $n - m$. However, the vector $\mathbf{x} \in \mathcal{N}(C^T)$ has n components.

3.2 The approach of Arbenz-Drmač [12]

Let P be a permutation matrix and

$$\widehat{Y} = PY = \begin{bmatrix} \widehat{Y}_1 \\ \widehat{Y}_2 \end{bmatrix}, \qquad \widehat{Y}_2 \in \mathbb{R}^{m\times m} \text{ nonsingular.} \tag{11}$$

Let

$$\widehat{A} = P^TAP = \begin{bmatrix} \widehat{A}_{11} & \widehat{A}_{12} \\ \widehat{A}_{21} & \widehat{A}_{22} \end{bmatrix}, \quad \widehat{M} = P^TMP = \begin{bmatrix} \widehat{M}_{11} & \widehat{M}_{12} \\ \widehat{M}_{21} & \widehat{M}_{22} \end{bmatrix}, \quad \widehat{C} = P^TC = \begin{bmatrix} \widehat{C}_1 \\ \widehat{C}_2 \end{bmatrix}.$$

Then the $n - m$ eigenvalues of the eigenvalue problem

$$\widehat{A}_{11}\hat{\mathbf{x}} = \lambda(\widehat{M}_{11} - \widehat{C}_1H^{-1}\widehat{C}_1^T)\hat{\mathbf{x}} \tag{12}$$

are precisely the $n-m$ positive eigenvalues of (5). The matrix on the right-hand side of (12) must not be formed explicitely as it is full.

3.3 Discussion

These approaches have in common that they require the solution of a system of equations involving the matrix H. In the present application this is the discretized Laplace operator in $W_h^{(2)}$. The order of H is about the same as the dimension of the 'coarse' space $V_h^{(1)}$. Therefore, we solve systems with H by a direct method. The method of Arbenz-Drmač has the advantage that the order of the eigenvalue problem is smaller by at least 1/8. Thus, less memory space is required.

4 Solving the matrix eigenvalue problem

Factorization-free methods are the most promising for effectively solving very large eigenvalue problems

$$A\mathbf{x} = \lambda M\mathbf{x}. \tag{13}$$

The factorization of A or M or a linear combination of them which is needed if a spectral transformation like shift-and-invert is applied requires way too much memory. If the shift-and-invert spectral transformation in a Lanczos type method is solved iteratively then high accuracy is required to establish the three-term recurrence. This is very time consuming even if the conjugate gradient method is applied with a good preconditioner. In most of our experiments the Jacobi-Davidson algorithm was much more effective than the implicitly restarted Lanczos algorithm as implemented in ARPACK [5] for solving the cavity and other eigenvalue problems [2, 3].

In this study we investigate three factorization-free algorithms for solving (13)

– **The symmetric Jacobi-Davidson algorithm (JDSYM)**
The Jacobi-Davidson algorithm has found considerable attention in recent years. It has been introduced by Sleijpen and van der Vorst [4]. There are variants for all types of eigenvalue problems [13]. Here, we use a modification of the algorithm JDQZ adapted to the generalized symmetric eigenvalue problem (13) as described in [2, 14].
In every step of JD, the search space is expanded by the solution of the so-called correction equation. This solution is closely related to the Newton correction of $\mathbf{grad}\,\rho(\mathbf{x}) = \mathbf{0}$ where $\rho(\mathbf{x}) = \mathbf{x}^T A\mathbf{x}/\mathbf{x}^T M\mathbf{x}$ is the Rayleigh quotient. The correction equation need not be solved to high accuracy. A few steps of MINRES is adequate except close to convergence (of the eigensolver) where high accuracy is required to get a decent convergence rate. By introducing a restart procedure the memory requirements can be bounded. Typically, besides the storage for the matrices A and M and the preconditioner, $(p + 3j_{\max})n$ memory locations are needed where $j_{\max}$ is the maximal dimension of the search space.
– **Block Rayleigh Quotient minimization (BRQMIN)**
BRQMIN is a proper subspace method. In the k-th iteration step the actual approximations of the eigenvectors corresponding to the p smallest eigenvalues of (13) are stored in X_k. X_{k+1} is obtained through the Rayleigh-Ritz procedure in the $2p$ dimensional space spanned by the columns $\mathbf{x}_j^{(k)}$ of X_k and P_k, the so-called search directions. The latter are obtained as $P_k = K^{-1}G_k + P_{k-1}B_k$. B_k is determined such that $P_{k-1}^T AP_k = O$, K is a preconditioner, and $G_k\mathbf{e}_j = \mathbf{grad}\,\rho(\mathbf{x}_j^{(k)})$. This is a cg-type algorithms for minimizing the Rayleigh quotient as proposed (but not implemented) 20 years ago by Longsine and McCormick [6]
– **The locally optimal block PCG method (LOBPCG)**
The locally optimal block preconditioned conjugate gradient algorithm has been introduced by Knyazev [7] as an improvement over BRQMIN. Local optimality is obtained by executing the Rayleigh-Ritz procedure in the $3p$ dimensional subspace $\mathcal{R}(X_k, K^{-1}G_k, P_{k-1}) \supset \mathcal{R}(X_k, P_k)$. For large eigenvalue problems, the

overhead introduced by the larger subproblem is negligible. The memory requirement is $10pn$ floating point numbers which is bearable as long as p is small.

5 The preconditioners

In the eigensolvers that we have discussed in the previous section a preconditioner, say K, is required to get a good convergence behavior. K should be a good approximation of A or $A - \theta M$ for some fixed or variable θ. For ease of presentation we set $\theta = 0$.

Let A in (5) be partitioned according to (4),

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}, \qquad A_{ij} \in \mathbb{R}^{n_i \times n_j}, \tag{14}$$

where the $(1,1)$-blocks correspond to the bilinear forms involving basis functions in $N_h^{(1)}$. Proceeding as Bank [15] we approximate the inverse of A in (14) by

$$K^{-1} = \begin{bmatrix} A_{11}^{-1} & O \\ O & \widetilde{A}_{22}^{-1} \end{bmatrix}, \tag{15}$$

where $\widetilde{A}_{22}^{-1}$ corresponds to a few (usually one) step of a stationary iteration method. K has similar properties as A. In particular, $KY = O$. This is important as it allows us to get a preconditioner $\widehat{K}_{11}$ for $\widehat{A}_{11}$ in (12) by choosing the (1,1)-block of $P^T K P$, where P is the permutation in (11).

In the nullspace method the inverse of the preconditioner has the form

$$K^{-1} = (I - Y H^{-1} C^T)(A_1 - \theta M_1)^{-1} \tag{16}$$

where K is given in (15).

6 Numerical experiments

6.1 MATLAB experiments with a rectangular cavity

We first report on experiments that were executed in the MATLAB 6 (R12) environment on a Intel Pentium III (500 MHz, 512kB Cache, 512 MB RAM) running the Linux 2.2 operating system.

The test problem, a rectangular cavity, had the size $n = \dim(N_h^{(2)}) = 6292$. The number of constraints was $m = \dim(L_h^{(2)}) = 1155$. We used both the nullspace method and the Arbenz-Drmač approach. In the latter the problem size was $n - m = 5137$. The timings for computing $p = 8$ and $p = 16$ eigenpairs are listed in Table 1. The four columns give the execution times for the nullspace and Arbenz-Drmač (AD) approaches with the two-level preconditioner discussed in section 5 where $\widetilde{A}_{22}^{-1}$ in (15) was chosen to be one step of the Jacobi or symmetric Gauss-Seidel iteration (which is SSOR with relaxation parameter $\omega = 1$). Approximate eigenvectors $\mathbf{x}$ are considered converged if $\|A\mathbf{x} - \rho(\mathbf{x})M\mathbf{x}\| \leq 10^{-8}$.

solver	nullspace Jacobi	nullspace SSOR(1)	Arbenz-Drmač Jacobi	Arbenz-Drmač SSOR(1)
$p = 8$				
LOBPCG	78.3	49.6	63.4	37.3
JDSYM (MINRES)	373.8	287.4	193.0	129.0
BRQMIN	78.5	51.7	63.4	40.0
$p = 16$				
LOBPCG	184.9	115.1	170.9	83.8
JDSYM (MINRES)	859.0	612.6	440.0	308.3
BRQMIN	172.2	119.3	144.2	100.2

Table 1. MATLAB timings for computing 8 and 16, respectively, eigenvalues of a problem of size $n = 6292$ with $m = 1155$.

The AD approach proved to be faster than the nullspace approach. LOBPCG is slightly faster than block Rayleigh quotient minimization in most cases. LOBPCG needs fewer iteration steps than BRQMIN until convergence, however a single step is more expensive as the dimension of the trial space has 1.5 times the dimension. Jacobi-Davidson does not perform so well. It is about three times slower than the other two solvers. JDSYM performs slightly better for $p = 16$. This conforms with the observation that it takes JDSYM a large number of iterations steps until it finds the first eigenpair. In these computations the dimension of trial spaces in JDSYM varied between p and $2.5p$. The correction equation was solved very inaccurately initially. Close to convergence the accuracy requirements are increased to obtain the high convergence rate [2]. The subspace dimension in LOBPCG and BRQMIN were p. There were no significant gains or losses if the subspace dimensions were chosen slightly bigger than p. It is however crucial for the performance that converged eigenvectors are locked.

6.2 The small model of the copper cavity

The next experiment concerned a coarse model of the future design of the copper cavity. Here, we computed $p = 10$ eigenpairs of a problem of size $n = 45'040$ with $m = 7824$ constraints. We used our C programs that make extensive use of the BLAS. The compute platform was a Sun Enterprise E3500 with 336 MHz UltraSPARC-II processors and 3 GB of main memory which was running the Solaris 2.6 operating system. We only used the nullspace method. The purpose of this experiment is to compare the two 2-level preconditioners used above with diagonal and no preconditioning. Timings are given in Table 2

For the three solvers t_{eig} gives the time spent for computing the 10 eigenpairs to the desired accuracy of $\|A\mathbf{x} - \rho(\mathbf{x})M\mathbf{x}\| \leq 10^{-8}$. t_{eig} does not include preparatory work like grid handling, matrix assembly etc. t_{11} gives the time that is spent for system solving by direct solvers. This concerns the $(1, 1)$ block of the preconditioner (16) and the matrix H that appears in the projector in (16) in the nullspace method or in the matrix M in the AD approach, cf. (12). Notice that

Table 2. Timings for computing 10 eigenvalues of the problem of size $n = 45'040$ with $m = 1155$ constraints. Times are in seconds.

Jacobi-Davidson				
preconditioner	it_{int}	it_{out}	t_{eig}	t_{11}
none	55	89	3680	
diagonal	19	86	1289	
hierarchical/Jacobi	8	82	900	237
hierarchical/SSOR(1)	4	76	593	137
LOBPCG				
preconditioner			t_{eig}	t_{11}
none			6135	
diagonal			1707	
hierarchical/Jacobi			644	163
hierarchical/SSOR(1)			413	95
BRQMIN				
preconditioner			t_{eig}	t_{11}
none			5862	
diagonal			1667	
hierarchical/Jacobi			502	129
hierarchical/SSOR(1)			384	89

the critical operation is not the factorization of these matrices but the forward and backward substitution in each iteration step. For JDSYM we also give the quantities it_{int} and it_{out}. The latter is the number of correction equations that had to be solved. The number it_{int} provides the average number of inner iterations, i.e., the steps until MINRES solver the correction equation to the desired accuracy. This number is quite low as the accuracy requirements is initially very loose and becomes more stringent as the outer iteration converges [2]. In this example we let the subspace dimensions in JDSYM vary between p and $2p$.

The Jacobi-Davidson performs best with the simple preconditioners while the simpler eigensolvers profit most from the more sophisticated preconditioners. With the two-level preconditioners the execution times of BRQMIN and LOBPCG are shorter than those of JDSYM by a factor of about 1.5.

With all three eigensolvers higher sophistication in the preconditioner decreases not only the iteration counts but also the execution times. This is in contrast to the experiments that we conducted with node elements in the box shaped cavity [16].

Numbers not given here show that with the hierarchical basis preconditioner it_{int} does not increase with the problem size as the analysis predicts. In fact, here it even decreased substantially.

Notice that t_{11} takes a considerable fraction of the solution of the eigenvalue problem. It is clear, that the present two-level preconditioner will not suffice for very problem sizes. A multi-level method will have to replace the direct solver that is used for solving systems involving the matrices A_{11} and H.

6.3 The big model of the copper cavity

Finally we discuss is a bigger model of the copper cavity with $n = 119'758$ and $m = 23'400$. We computed 10 eigenpairs with the nullspace method as well as with the Arbenz-Drmač approach. The convergence criterion was $\|A\mathbf{x} - \rho(\mathbf{x})M\mathbf{x}\| \leq 10^{-4}$. The best preconditioner of the previous examples was chosen. Timings are given in Table 3.

Table 3. Timings for computing 10 eigenvalues of the problem of size $n = 119'758$ with $m = 23'400$ constraints. Times are in seconds.

method, preconditioner	JDSYM	LOBPCG	BRQMIN
null space, 2-level/SSOR(1)	1803	849	1124
AD, 2-level/SSOR(1)	1024	668	990

LOBPCG turns out to be the fastest solver. In this example it is clearly superior to BRQMIN. The AD approach is much faster that the nullspace method. In the AD approach, JDSYM needs to solve (approximately) $it_{\text{out}} = 80$ correction equations with altogether $it_{\text{inner}} = 326$ iteration steps to extract the ten eigenpairs. This means that in the average four iteration steps are used per system of equations. LOBPCG in turn executes 33 block iteration steps which requires 251 calls of the preconditioner. This ratio explains to a large extent the ratio of the execution times. Notice that there are less than $33p = 330$ invocations of the preconditioner as converged eigenvectors are locked, i.e. kept fixed.

The main portions of the execution time of 1024 seconds of JDSYM are matrix-vector products (152" or 15%), applying the preconditioner (294" or 29%) and applying the projector (465" or 45%). The bulk of the latter is solving system involving H. The corresponding numbers for LOBPCG are 88" (13%) for matrix-vector products, 134" (20%) for the projector and 326" (49%) for matrix-vector product with $\widehat{M}_{11} - \widehat{C}_1 H^{-1} \widehat{C}_1^T$.

7 Conclusions

Jacobi-Davidson is in general a very powerful method that can be applied to every kind of eigenvalue problems. For the special class of symmetric matrices it can be outperformed by Rayleigh quotient minimization-type algorithms. Good preconditioners are needed to get fast eigensolvers. In the cavity eigenvalue problems diagonal preconditioning was not satisfactory. Hierarchical preconditioning makes iteration numbers insensitive to problem size. But our two-level approach turns out to be ineffective with large problem sizes as the 'coarse' systems are getting too big. We intend to replace the direct solvers by a conjugate gradient method with a algebraic multilevel preconditioner.

Acknowledgment

I thank Roman Geus for performing the experiments on the Sun.

References

1. Grimes, R., Lewis, J.G., Simon, H.: A shifted block Lanczos algorithm for solving sparse symmetric generalized eigenproblems. SIAM J. Matrix Anal. Appl. **15** (1994) 228–272
2. Arbenz, P., Geus, R.: A comparison of solvers for large eigenvalue problems originating from Maxwell's equations. Numer. Lin. Alg. Appl. **6** (1999) 3–16
3. Arbenz, P., Geus, R., Adam, S.: Solving Maxwell eigenvalue problems for accelerating cavities. Phys. Rev. ST Accel. Beams **4** (2001) 022001 (Electronic journal available from `http://prst-ab.aps.org/`).
4. Sleijpen, G.L.G., van der Vorst, H.A.: A Jacobi-Davidson iteration method for linear eigenvalue problems. SIAM J. Matrix Anal. Appl. **17** (1995) 401–425
5. Lehoucq, R.B., Sorensen, D.C., Yang, C.: ARPACK Users' Guide: Solution of Large-Scale Eigenvalue Problems by Implicitely Restarted Arnoldi Methods. SIAM, Philadelphia, PA (1998).
6. Longsine, D.E., McCormick, S.F.: Simultaneous Rayleigh–quotient minimization methods for $Ax = \lambda Bx$. Linear Algebra Appl. **34** (1980) 195–234
7. Knyazev, A.V.: Toward the optimal preconditioned eigensolver: Locally optimal block preconditioned conjugate gradient method. SIAM J. Sci. Comput. **23** (2001) 517–541
8. Kikuchi, F.: Mixed and penalty formulations for finite element analysis of an eigenvalue problem in electromagnetism. Comput. Methods Appl. Mech. Eng. **64** (1987) 509–521
9. Girault, V., Raviart, P.A.: Finite Element Methods for the Navier-Stokes Equations. Springer-Verlag, Berlin (1986) (Springer Series in Computational Mathematics, 5).
10. Nédélec, J.C.: Mixed finite elements in $\mathbb{R}^3$. Numer. Math. **35** (1980) 315–341
11. Silvester, P.P., Ferrari, R.L.: Finite Elements for Electrical Engineers. 3rd edn. Cambridge University Press, Cambridge (1996)
12. Arbenz, P., Drmač, Z.: On positive semidefinite matrices with known null space. Tech. Report 352, ETH Zürich, Computer Science Department (2000) (Available at URL `http://www.inf.ethz.ch/publications/`).
13. Fokkema, D.R., Sleijpen, G.L.G., van der Vorst, H.A.: Jacobi-Davidson style QR and QZ algorithms for the partial reduction of matrix pencils. SIAM J. Sci. Comput. **20** (1998) 94–125
14. Geus, R.: The Jacobi-Davidson algorithm for solving large sparse symmetric eigenvalue problems. PhD thesis, Computer Science Department, ETH Zurich (2002)
15. Bank, R.E.: Hierarchical bases and the finite element method. Acta Numerica **5** (1996) 1–43
16. Arbenz, P., Adam, S.: On solvinging Maxwellian eigenvalue problems for accelerating cavities (1998) Paper presented at the International Computational Accelerator Physics Conference (ICAP'98), Monterey CA, September 14–18, 1998. 5 pages. Available from `http://www.inf.ethz.ch/~arbenz/ICAP98.ps.gz`.