Quadratic Eigenproblems Are No Problem

Gerard Sleijpen
Henk van der Vorst
Martin van Gijzen

High-dimensional eigenproblems often arise in the solution of scientific problems involving stability or wave modeling. In this article we present results for a quadratic eigenproblem that we encountered in solving an acoustics problem, specifically in modeling the propagation of waves in a room in which one wall was constructed of sound-absorbing material.

Efficient algorithms are known for the standard linear eigenproblem,

$$\mathbf{A}\mathbf{x} = \lambda\mathbf{x}$$

where $\mathbf{A}$ is a real or complex-valued square matrix of order n. Generalized eigenproblems of the form $\mathbf{A}\mathbf{x} = \lambda\mathbf{B}\mathbf{x}$, which occur in finite element formulations, are usually reduced to the standard problem, in a form such as $\mathbf{B}^{-1}\mathbf{A}\mathbf{x} = \lambda\mathbf{x}$. The reduction requires an expensive inversion operation for one of the matrices involved. Higher-order polynomial eigenproblems are also usually transformed into standard eigenproblems. We discuss here the second-degree (i.e., quadratic) eigenproblem

$$\left(\lambda^2\mathbf{C}_2 + \lambda\,\mathbf{C}_1 + \mathbf{C}_0\right)\mathbf{x} = 0$$

in which the matrices $\mathbf{C}_i$ are square matrices.

In a discussion of the quadratic eigenvalue problem, Saad [3] refers to a lack of solution methods: "There seems to be a dichotomy between the need of users, mostly in finite elements modeling, and the numerical methods that numerical analysts develop." Commenting on general, higher-order eigenvalue problems, Bai [1] says, "Besides transforming such an eigenvalue problem to the standard eigenvalue problem, not much progress has been made concerning how to solve such λ-matrix eigenvalue problems directly and efficiently."

We show here that the recently proposed Jacobi-Davidson method [5, 6] can be applied directly to polynomial eigenproblems,

$$\left(\lambda^k \mathbf{C}_k + \lambda^{k-1}\mathbf{C}_{k-1} + \cdots + \mathbf{C}_0\right)\mathbf{x} = 0$$

without inversion of matrices or transformation to the standard case. The method uses projections on low-dimensional subspaces in order to reduce the given polynomial eigenproblem, with matrix coefficients of high order, to a polynomial problem with matrix coefficients of low order. The reduced problem can then be solved by standard techniques. Much of the method is easily parallelizable. The computational results presented here were obtained on 64 processors of a Cray T3D.

The Jacobi-Davidson method

The Jacobi-Davidson method [6] was proposed initially as a solution method for the standard eigenvalue problem

$$\mathbf{A}\mathbf{x} = \lambda\,\mathbf{x}$$

In essence, the method is based on two known ideas. The first is that of computing approximate eigensolutions by projecting the eigenproblem onto a low-dimensional search subspace, spanned by the columns of the matrix $\mathbf{V}$,

$$(\mathbf{V}^*\mathbf{A}\mathbf{V} - \theta\,\mathbf{V}^*\mathbf{V})\,y = 0 \tag{1}$$

where $\mathbf{V}^*$ denotes the conjugate of $\mathbf{V}$. This can be interpreted as the *Davidson* part of the algorithm. The projected system (1) is again a standard eigenproblem, but of much lower dimension than the original problem. The approximate eigensolutions of this system are the Ritz vectors $\mathbf{u} = \mathbf{V}y$, with associated Ritz values θ.

The Jacobi-Davidson method is efficient if the dimension of the search subspace can be kept small. In other words, the (basis for the) search subspace must be constructed in such a way that it contains significant information on the desired eigenpair. In the Jacobi-Davidson method this basis is expanded in each iteration by the approximate solution, $\mathbf{t} \perp \mathbf{u}$, of the correction equation

$$(\mathbf{I} - \mathbf{u}\mathbf{u}^*)\,(\mathbf{A} - \theta\mathbf{I})\,(\mathbf{I} - \mathbf{u}\mathbf{u}^*)\,\mathbf{t} = -\,\mathbf{r} \tag{2}$$

where $\mathbf{r} = \mathbf{A}\mathbf{u} - \theta\,\mathbf{u}$ is the residual of the approximate eigensolution. This idea can be traced back to an 1846 paper of Jacobi. For stability reasons, the basis of the search space, the column vectors of $\mathbf{V}$, is constructed to be orthonormal. The new basis vector is the orthogonal complement of $\mathbf{t}$ with respect to the previous basis vectors. It can be shown that if the correction equation (2) of the Jacobi-Davidson method is solved to a sufficient degree of accuracy, the asymptotic rate of convergence to an eigenpair is at least quadratic.

A highly accurate solution of (2) is not required, however. In practice, it is often more efficient to construct an approximate solution by means of a small number of GMRES [4] steps. Of course, this may lead to an increase in the number of Jacobi-Davidson iterations, but each iteration will be considerably less expensive then if it were solved to higher accuracy. In this way, the Jacobi-Davidson algorithm becomes explicit; that is, no matrix decomposition or backward or forward substitutions are required in the algorithm. As a result, the method is well suited for parallel computing. If the desired eigenvalue is well isolated from the other eigenvalues, then a few steps of GMRES suffice for fast, almost quadratic, convergence. In other cases convergence is linear, with the rate depending on the relative separation of the eigenvalue to be determined.

The Jacobi-Davidson approach can also be used for generalized eigenproblems and higher-order polynomial eigenproblems [5]:

$$\mathbf{P}_\ell(\lambda)\mathbf{x} = 0 \tag{3}$$

where $\mathbf{P}_\ell(\lambda) = \lambda^\ell \mathbf{C}_\ell + \lambda^{\ell-1}\mathbf{C}_{\ell-1} + \ldots + \mathbf{C}_0$. The projected problem has the same structure as the original problem, but it is of much lower dimension.

The algorithm for the higher-order problem (3) is outlined in Figure 1. Again, the asymptotic rate of convergence is quadratic when the correction equation in step #b.5 is solved accurately. We suggest that the modified Gram-Schmidt method (ModGS in Figure 1) be used in step #a and step #b.6 for the orthonormalization.

a. **Start:**

Choose an initial subspace $\mathbf{V}$

Orthonormalize $\mathbf{V}$

b. **Repeat:**

(1) Compute $\mathbf{W}_i \leftarrow \mathbf{C}_i \mathbf{V}$

and $H_i \leftarrow \mathbf{V}^* \mathbf{W}_i, \quad (i = 0, \ldots, \ell)$.

(2) Compute the desired eigenpair (θ, y) of

$$\theta^\ell H_\ell y + \theta^{\ell-1} H_{\ell-1} y + \ldots + H_0 y = 0,$$

with $\| y \| = 1$.

(3) Compute $\begin{array}{l} \mathbf{u} \leftarrow \mathbf{V} y, \quad \mathbf{w} \leftarrow \mathbf{P}'_\ell(\theta)\mathbf{u}, \\ \mathbf{r} \leftarrow \mathbf{P}_\ell(\theta)\mathbf{u}. \end{array}$

(4) **Stop** if satisfied.

(5) Solve (approximately)

$$(\mathbf{I} - \frac{\mathbf{w}\,\mathbf{u}^*}{\mathbf{u}^*\mathbf{w}})\, \mathbf{P}_\ell(\theta)\, (\mathbf{I} - \mathbf{u}\,\mathbf{u}^*)\, \mathbf{t} = -\mathbf{r}$$

(6) Expand $\mathbf{V}$: $\mathbf{V} \leftarrow \text{ModGS}([\,\mathbf{V}\,|\,\mathbf{t}\,])$.

Figure 1: Jacobi-Davidson method for the higher-order eigenproblem (3). In step #b.3 $\mathbf{P}'_\ell(\theta)\mathbf{u} = \ell\, \theta^{\ell-1}\, \mathbf{C}_\ell \mathbf{u} + \ldots + \mathbf{C}_1 \mathbf{u}$.

Solution of the projected eigenproblem leads to a number of approximate eigenpairs, one of which is selected for the construction of a new correction equation. We can direct the convergence of the Jacobi-Davidson method toward a chosen target value by selecting the approximate eigenvalue closest to the target. A target value can also be chosen in the interior of the spectrum, making it possible to compute interior eigenvalues. The usual approach for computing interior eigenvalues is to use a shift-and-invert strategy. Expensive inversion operations, as mentioned earlier, are avoided in the Jacobi-Davidson approach.

Parallelization of the algorithm

Because of its explicit nature, the Jacobi-Davidson method is well suited to parallel computing. In a parallel setting, the computational steps of the algorithm presented in Figure 1 are carried out as follows:

In step #a a subspace is chosen and an orthonormal basis for this subspace is constructed. This involves the computation of inner products and vector updates.

Step #b.1 of the iteration loop involves only the multiplication of the matrices $\mathbf{C}_i$ with the last added basis vector of $\mathbf{V}$. The inner products of the columns of the matrices $\mathbf{V}$ and $\mathbf{W}_i$ are then used to compute the projected matrices H_i. Only one new row and column of each projected matrix H_i must be computed in each iteration.

In step #b.2 the eigenpairs of the projected (low-dimension) system are computed. Because the order of the projected matrices is typically under 30, the number of computations in this step is limited and often negligible in comparison with that in the other steps.

In step #b.3 the selected Ritz vector $\mathbf{u}$ is computed by taking a linear combination of the columns of $\mathbf{V}$. The vectors $\mathbf{w}$ and $\mathbf{r}$ can be computed by multiplying $\mathbf{u}$ with the operators $\mathbf{P}_\ell'(\theta)$ and $\mathbf{P}_\ell(\theta)$, respectively. Since $\mathbf{u} = \mathbf{V}y$ and $\mathbf{W}_i = \mathbf{C}_i\mathbf{V}$, however, the vectors $\mathbf{w}$ and $\mathbf{r}$ can also be computed by taking a linear combination of the columns of the matrices $\mathbf{W}_i$. If cleverly implemented, then, this step requires no multiplications with the matrices $\mathbf{C}_i$—vector updates alone will suffice.

In step #b.4 the convergence criterion is checked. This typically involves computation of the norm of the residual $\mathbf{r}$.

The computation of the approximate solution of the correction equation in step #b.5 is usually the most expensive step in the algorithm. If a method like GMRES is used, the only time-consuming operations are matrix–vector multiplications, vector updates, and inner-product computations. Preconditioning for GMRES could have been used but was not necessary for our problem. The matrix $(\mathbf{I} - \frac{\mathbf{w}\,\mathbf{u}^*}{\mathbf{u}^*\mathbf{w}})\,\mathbf{P}_\ell(\theta)\,(\mathbf{I} - \mathbf{u}\,\mathbf{u}^*)$ does not have to be constructed explicitly. Multiplication with this matrix can be carried out in three steps: multiplication with the projector on the right; multiplication by the matrices $\mathbf{C}_i$, followed by a linear combination of the results; and multiplication with the projector on the left. For a multiplication with a projector, only one inner product and one vector update are required.

In #b.6, the final step of the algorithm, the subspace $\mathbf{V}$ is expanded with the solution of the correction equation and this new basis vector is orthogonalized against the others. This process requires inner products and vector updates.

This global analysis reveals that multiplications with the matrices $\mathbf{C}_i$, vector updates, and inner-product computations are the operations that consume the most CPU time; only these operations need to be parallelized. We have incorporated the method into a finite element program, described in [8], with the parallelization based on a simple domain decomposition strategy. Each element is uniquely assigned to a subdomain, but neighboring subdomains share common nodal points along the edges. Each subdomain is mapped onto a processor. The matrix–vector multiplications can be carried out with the submatrices that correspond to a subdomain. Results that correspond to overlapping nodal points have to be communicated to any other processors that contain the nodal points in their subdomains. Since each subdomain is adjacent to only a few others, this communication involves only a few processors.

For the computation of inner products, partial inner products are computed within each subdomain and the results are sent to one master processor. The partial inner products are combined by the master processor, and the result is broadcast to all other processors. Since this type of communication involves all processors, it needs global communication and may be expensive for large numbers of processors.

To avoid unnecessary communication, computation of the eigenvalues and eigenvectors of the projected system, a relatively inexpensive operation, is performed on all processors.

We have implemented the algorithm on the Cray T3D, a distributed-memory computer. For the communication steps, we used the fast `SHMEM_GET` and `SHMEM_PUT` routines. The observed speedups were very satisfactory; the example discussed in [7] has linear speedup for 16 to 64 processors.

The Problem

We encountered our problem in modeling the propagation of sound waves in a room in which one wall was made of a sound-absorbing material. The propagation of sound can be described by the wave equation:

$$\frac{1}{c^2}\frac{\partial^2 p}{\partial t^2} = \triangle p \quad \text{in } \Omega \tag{4}$$

where p is the pressure perturbation, $\triangle$ is the Laplace operator, $\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2}$, and c is the speed of sound (340 meter/second). The three-dimensional domain Ω is $4 \times 4 \times 4$ meter^3.

Substitution of an expression of the form $p = \bar{p}e^{\lambda t}$ for the solution leads to the eigenproblem,

$$\frac{\lambda^2}{c^2}\bar{p} = \triangle \bar{p} \quad \text{in } \Omega \tag{5}$$

The parameter λ is an eigenvalue, and $\bar{p}$ is an eigenfunction.

Solid walls are modeled by a homogeneous Neumann boundary condition, $\frac{\partial p}{\partial n} = 0$, which we imposed on five sides of the domain. The sixth side of the domain was an absorbing wall. The boundary condition for such a wall is

$$\frac{\partial p}{\partial n} = -\frac{1}{cZ_n}\frac{\partial p}{\partial t} \tag{6}$$

where Z_n is the normal impedance. Substitution of $p = \bar{p}e^{\lambda t}$ in this equation leads to

$$\frac{\partial \bar{p}}{\partial n} = -\frac{\lambda}{cZ_n}\bar{p} \tag{7}$$

Beltman [2] explains how a few eigenvalues can be computed analytically for a given (complex) impedance Z_n. In our model problem, for which we selected an impedance $Z_n of 0.2 - 1.5i$, $\lambda = -5.19 + 217.5i$ is an analytical eigenvalue. In more realistic circumstances, the eigenvalues would not be known, but we chose this particular example in order to build confidence in our algorithm and in order to show that it can be used to achieve accurate results.

We used the finite element method to discretize the problem. This was done with $64 \times 64 \times 64 \times 5$ tetrahedral elements, with linear interpolation functions, that were equidistantly distributed throughout the solution domain. This leads to the quadratic eigenproblem

$$\hat{\lambda}^2 Mx + \hat{\lambda}Cx + Kx = 0 \tag{8}$$

in which the matrices M, C, and K are of order $274,625$.

The eigenvalues associated with the discretized model, $\hat{\lambda}$, are numerical approximations of the analytical eigenvalues, λ. The eigenvectors x are approximations of the eigenfunctions $\bar{p}$. As a consequence, we expected (8) to have an eigenvalue close to the analytical eigenvalue $-5.19 + 217.5i$, which we selected as our target. The stiffness matrix K, the discretization of $-\triangle$, is a sparse matrix that has 19 diagonals with nonzero coefficients. In our case it is singular, implying that zero is an eigenvalue of the quadratic eigenproblem, which makes the inversion of K impossible. The damping matrix, C, which stems from the discretization of (7), is complex because of the complex impedance. The mass matrix, M, the discretization of the unit operator multiplied by the factor $\frac{1}{c^2}$, has the same sparsity structure as K.

In order to get an impression of the spectrum near $-5.19 + 217.5i$, we also computed a number of eigenvalues of a smaller eigenproblem, discretized with $10 \times 10 \times 10 \times 5$ elements. Figure 2 shows the eigenvalues of the discrete system, along with the analytical eigenvalue ($+$) that we were seeking to approximate.

We also wanted to demonstrate that the Jacobi-Davidson method can be used for interior eigenvalues. The analytical eigenvalue is in the interior of the spectrum, and at each step we selected the eigenvalue approximation closest to this value for the correction equation (step #b.5 of the algorithm in Figure 1). The Jacobi-Davidson algorithm was restarted after every 20th iteration, with the eigenpair of the eigenvalue closest to the target. This is done in order to limit the dimension of the search subspace $\mathbf{V}$. The correction equations were solved approximately with 30 steps of GMRES. We carried out the numerical experiments on 64 processors of a Cray T3D in single-precision arithmetic, for a working precision of approximately 16 digits (an artifact of the Cray; single precision is normally 8 decimal digits).

The algorithm produced the eigenvalue $-5.20 + 217.5i$ in 33 Jacobi-Davidson iterations. Our stopping criterion was that the norm of the residual of the computed eigenpair be reduced by a factor of 10^6 relative to the starting residual. The computing time was 93.4 seconds (elapsed time), a computational speed of slightly more than 1 Gflops.

To put this result in perspective, we point out that *one* shift-and-invert operation would have taken more than 10 minutes if carried out at peak performance (9.6 Gflops). Peak performance never being attained in practice, an estimate of more than an hour of computing time per shift-and-invert

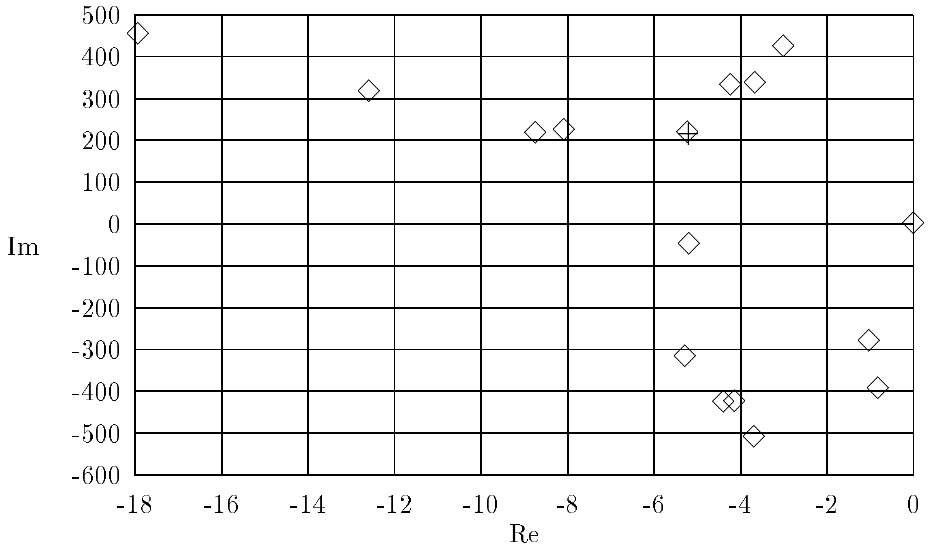

Figure 2: Eigenspectrum of our sample problem near the analytical eigenvalue $-5.19 + 217.5i$.

operation is more realistic. Direct methods, of course, are completely impractical for this problem because of their excessive storage requirements (600 Gbytes). Even if we could neglect the issue of storage, however, Householder reductions would take more than a month of wall-clock time, assuming peak performance.

To demonstrate the scalability of the algorithm, we present results obtained for a smaller problem (of order 136,161) on varying numbers of processors on the Cray T3D. These results are shown in Table 1. In going from 16 to 32 processors, there is a slightly greater then linear speedup. This is a result of the larger aggregate cache size. These results indicate that the relative speedup achieved in moving from 16 to 64 processors is nearly linear.

Processors	Elapsed time, seconds
16	206.4
32	101.3
64	52.1

Table 1: Results for a problem of order 136,161 on a Cray T3D.

Conclusions

The Jacobi-Davidson method is a flexible new method that can be applied to standard eigenproblems, as well as to higher-order polynomial eigenproblems. The method requires no matrix inversion operations and is therefore relatively easy to implement on parallel computers. The rate of convergence is very high for problems of the type described in this article.

Acknowledgments

The authors thank Marco Beltman for his extensive help with the acoustic problem. Access to the Cray T3D was provided by the National Computing Facilities Foundation (NCF). The research of the second and third authors was funded by the Netherlands Organization for Scientific Research (NWO), for project MPR-cluster 95MPR04.

References

[1] Z. Bai, *Progress in the numerical solution of the nonsymmetric eigenvalue problem*, Numerical Linear Algebra with Applications, 2:3(1995), pp: 219–234.

[2] W.M. Beltman, *An impedance problem: An analytical solution for a long tube*, Report No. WB.94/TM-952, University of Twente, The Netherlands, 1994.

[3] Y. Saad, *Numerical methods for large eigenvalue problems*, Manchester University Press, Manchester, 1992.

[4] Y. Saad and M.H. Schultz, *GMRES: A generalized minimum residual algorithm for solving nonsymmetric linear systems*, SIAM Journal on Scientific and Statistical Computing, 7(1986), pp: 856–869.

[5] G.L.G. Sleijpen, J.G.L. Booten, D.R. Fokkema, and H.A. van der Vorst, *Jacobi-Davidson type methods for generalized eigenproblems and polynomial eigenproblems*, accepted for publication, BIT (1996).

[6] G.L.G. Sleijpen and H.A. van der Vorst, *A Jacobi-Davidson iteration method for linear eigenvalue problems*, SIAM Journal on Matrix Analysis and Applications, 17(1996), pp: 401–425.

[7] M.B. van Gijzen, *A parallel eigensolution of an acoustic problem with damping*, submitted for publication.

[8] M.B. van Gijzen, *Parallel iterative solution methods for linear finite element computations on the Cray T3D*, Proceedings of the HPCN Europe 1995 Conference, *Lecture Notes in Computer Science*, Springer Verlag, Vol. 919, 1995.

The authors are all with the Mathematical Institute, Utrecht University, PO Box 80.010, NL-3508 TA Utrecht, The Netherlands. Their e-mail addresses are: `sleijpen@math.ruu.nl, vorst@math.ruu.nl, vangyzen@math.ruu.nl`