The parallel computation of the smallest eigenpair of an acoustic problem with damping.

Martin B. van Gijzen* and Femke A. Raeven*

Abstract

Acoustic problems with damping may give rise to large quadratic eigenproblems. Efficient and parallelizable algorithms are required for solving these problems. The recently proposed Jacobi-Davidson method is well suited for parallel computing: no matrix decomposition and no back or forward substitutions are needed. This paper describes the parallel solution of the smallest eigenpair of a realistic and very large quadratic eigenproblem with the Jacobi-Davidson method.

Keywords: Quadratic eigenproblem, Jacobi-Davidson method, acoustics, parallel computers

1 Introduction.

The level of sound is an important quantity in many applications. For example in the design of cars and planes it is of great importance to know in advance that the level of sound will remain below a prescribed value. In the context of acoustics the problem of sound propagation is formulated in mathematical terms, for an example see [6].

In real live applications it is often impossible to obtain an analytical solution of the mathematical model of an acoustic problem. In that case the problem is usually solved numerically. First the mathematical model of the acoustic problem is discretized by, e.g., the finite element method [2, 3]. The second step is to determine the eigenfrequencies of the discretized problem, which involves the solution of a generalized eigenproblem. If the acoustic problem involves damping we even have to solve a quadratic eigenproblem.

This paper describes how a realistic acoustic problem with damping can be modeled and discretized. The resulting quadratic eigenproblem is of very large size and for its solution with reasonable computer resources an efficient and parallelizable algorithm is required.

Recently, Sleijpen and Van der Vorst [11] have proposed the Jacobi-Davidson method for solving the ordinary eigenproblem. The algorithm can be extended for the generalized eigenproblem and polynomial eigenproblems [10], in particular for the quadratic problem [7]. We have implemented the Jacobi-Davidson method on the massively parallel Cray T3D. We have used this implementation to compute the smallest eigenpair of the quadratic problem in parallel, with the number of processors used ranging from 16 to 64.

The outline of this paper is as follows. In section 2 we describe the physical problem: the propagation of sound in a room, and its mathematical formulation. This section also gives a

*Mathematical Institute, University of Utrecht, P.O. Box 80.010. NL-3508 TA Utrecht, The Netherlands, telephone: E-mail: vangijzen@math.ruu.nl, raeven@math.ruu.nl.

brief description of the finite element discretization of the problem and some characteristics of the resulting quadratic eigenproblem that follow from this discretization. Section 3 describes the standard approach to solve the quadratic eigenproblem and it explains the severe drawbacks of this approach. In section 4 the solution algorithm we have used for solving the quadratic eigenproblem can be found. We describe the parallelization of the complete analysis, of both the discretization and the solution in section 5. The analysis has been performed on the Cray T3D. Section 6 gives timings for computing the smallest eigenpair on different numbers of processors. Some further improvements of the algorithm are described in the final section.

2 The model problem and its discretization.

As a model problem we investigate the propagation of sound in a room of $8 \times 4 \times 4$ meter. Four walls of the room are reflecting, one is absorbing, and one side of the room is open.

The propagation of sound can be described by the wave equation

$$\frac{1}{c^2}\frac{\partial^2 p}{\partial t^2} = \Delta p, \tag{1}$$

in the domain $\Omega = [0,8]\times[0,4]\times[0,4]$. In this equation p represents the pressure perturbation, which is a function of place and time, and c is the speed of sound ($c = 340m/s$). The Laplace operator $\triangle$ is defined by $\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2}$. The boundary condition for a reflecting wall is

$$\frac{\partial p}{\partial n} = 0. \tag{2}$$

We impose this boundary condition at $y = 0$, $y = 4$, $z = 0$ and $z = 4$. For the open side at $x = 0$ the boundary condition is

$$p = 0 \tag{3}$$

and for the absorbing wall at $x = 8$

$$\frac{\partial p}{\partial n} = -\frac{1}{cZ_n}\frac{\partial p}{\partial t}, \tag{4}$$

with Z_n the normal impedance. In this example we choose $Z_n = 0.2 - 1.5i$. We assume that the solution is of the form

$$p = \overline{p}e^{\omega t} \tag{5}$$

where $\overline{p}$ is a function of the position only. Substituting this in (1), we obtain

$$\frac{\omega^2}{c^2}\overline{p} = \Delta\overline{p}. \tag{6}$$

The parameter ω is called an eigenvalue and the function $\overline{p}$ is called an eigenfunction of the problem. We subsitute (5) also in the boundary conditions. For the conditions (2) and (3) this is trivial, because they are time independent. Condition (4) changes into

$$\frac{\partial\overline{p}}{\partial n} = -\frac{\omega}{cZ_n}\overline{p}. \tag{7}$$

The frequency f, that is of interest in physics, can be computed from the eigenvalues ω by

$$f = \frac{Im(\omega)}{2\pi} \quad . \tag{8}$$

We have discretized the model problem with the finite element method. We used $80\times40\times40\times5$ tetrahedral elements with linear interpolation functions. The grid points are equidistantly distributed. The discretization yields the following damped quadratic eigenproblem

$$\lambda^2 Mx + \lambda Cx + Kx = 0. \tag{9}$$

In this equation M, C, and K are square matrices of dimension 136161. The eigenvalues λ are numerical approximations for ω, and the corresponding eigenvectors x are approximations for the eigenfunctions $\overline{p}$. The stiffness matrix K results from the discretization of $-\triangle$, and is symmetric positive definite. The damping matrix C stems from the discretization of (7). We used a lumping technique in order to make this matrix diagonal. The matrix C is complex, since the normal impedance Z_n is complex. The mass matrix M represents the discretization of the unit operator times the factor $\frac{1}{c^2}$. The matrix M has the same sparsity structure as K, and is also symmetric positive definite. We could have used a lumping technique to make M diagonal too, but we did not do this in order to illustrate that systems with a nondiagonal matrix M can also be solved efficiently in parallel with the new Jacobi-Davidson method. The efficient parallel solution of systems with a nondiagonal matrix M is not so simple in the standard approach for quadratic eigenproblems, as will be explained in the next section. In our model problem lumping of the mass matrix would have had no effect on the the order of the accuracy of the discretization or on the positive definiteness of M, but this is in general not the case. For example, if the problem is discretized with higher order elements, lumping may lead to a singular or indefinite mass matrix.

3 Standard approach for solving the quadratic eigenvalue problem.

A quadratic eigenvalue problem is usually solved by first linearizing it into a twice as large generalized eigenproblem. We can linearize in many different ways. An obvious method is to rewrite the n-dimensional quadratic problem (9) into the $2n$-dimensional generalized eigenproblem

$$\begin{pmatrix} -C & -K \\ I & 0 \end{pmatrix} \begin{pmatrix} \lambda x \\ x \end{pmatrix} = \lambda \begin{pmatrix} M & 0 \\ 0 & I \end{pmatrix} \begin{pmatrix} \lambda x \\ x \end{pmatrix}. \tag{10}$$

The advantage of this linearization is that the matrix on the right-hand side is symmetric and positive definite. This generalized eigenproblem can be reduced to a standard eigenproblem:

$$\begin{pmatrix} M^{-1} & 0 \\ 0 & I \end{pmatrix} \begin{pmatrix} -C & -K \\ I & 0 \end{pmatrix} \begin{pmatrix} \lambda x \\ x \end{pmatrix} = \lambda \begin{pmatrix} \lambda x \\ x \end{pmatrix}. \tag{11}$$

This problem can be solved by a standard method like Arnoldi's method [1]. In order to understand the improvements that can be obtained though the Jacobi-Davidson method, we will first describe the usage of the Arnoldi method in some detail. In this approach one calculates eigenvalues λ and $2n$-dimensional eigenvectors y of the product of these two block-matrices in (11). The fact that an eigenvector y consists of the two parts λx and x is enforced by (11). Hence the eigenvector of (11) leads immediately to the eigenvector x corresponding to λ of the quadratic problem.

Arnoldi's method constructs an orthogonal basis for the $K^i(A; v_1)$. This subspace is defined by

$$K^i(A; v_1) = \text{span}\{v_1, Av_1, \ldots, A^{i-1}v_1\}, \tag{12}$$

where v_1 is the starting vector. For our problem the matrix A in (12) is defined by

$$A = \begin{pmatrix} M^{-1} & 0 \\ 0 & I \end{pmatrix} \begin{pmatrix} -C & -K \\ I & 0 \end{pmatrix} \tag{13}$$

The Krylov subspace is expanded by one new basis vector in every iteration. To compute a new basis vector a multiplication with the system matrix A is performed. The resulting vector is orthogonalized against all previously computed basis vectors.

In acoustics one is usually interested in a few of the smallest eigenpairs. A well known method to speed up convergence towards the smallest eigenpairs is to solve the inverse problem [8]

$$Mx + \mu Cx + \mu^2 Kx = 0, \quad \lambda = \frac{1}{\mu}. \tag{14}$$

Consequently, the matrix system matrix A now becomes

$$A = \begin{pmatrix} K^{-1} & 0 \\ 0 & I \end{pmatrix} \begin{pmatrix} -C & -M \\ I & 0 \end{pmatrix} \tag{15}$$

If the matrices M and K have the same sparsity pattern, multiplying with the system matrix for the inverse problem (15) is as expensive as multiplying with the system matrix (13). However, if M is diagonal, then multiplying with (13) is much less expensive than multiplying with (15). Since the number of iterations will increase there will be a tradeoff, and even in that case it may be faster to solve the inverse problem.

The multiplication with A requires the solution of a systems with the mass matrix (or with the stiffness matrix if the inverse problem is solved). The same problem arises when solving any generalized linear eigenproblem of the form $Kx = \lambda Mx$. The usual approach is to make a Choleski decomposition of M before the iterative process is started, and to perform a back- and forward substitution with the Choleski factors in every iteration. The back- and forward substitution are expensive operations compared with the multiplication with a sparse matrix, like the matrices K, C, and M in our model problem. The multiplication of a vector with a sparse matrix requires $O(n)$ floating point operations. A back- or forward substitution requires $4n \times mb + O(n)$ floating point operations, where mb is the bandwidth of the matrix, M in our case. This is twice the number of floating point operations that is usually given for the back- or forward substitution, see e.g. [5]. The reason is that, although M is real, the right-hand side vector is complex. The bandwidth of a matrix is usually $O(n^{2/3})$ in 3D problems. For the model problem in section 2, we have $n = 136161$ and $mb = 1681$. Computation of the the Choleski decomposition of M is even more expensive, requiring $n \times mb^2 + O(n \times mb)$ floating point operations. However, the decomposition has to be computed only once, whereas the back- and forward substitutions must be performed in every iteration.

The back and forward substitutions are sequential processes by nature. The coefficients of the solution vector of a triangular solve must be computed one by one. Hence, back- and forward substitutions are ill-suited for parallel processing.

Reducing the quadratic eigenproblem (9) to the standard eigenproblem (11) implies that we have to perform expensive back- and forward substitutions. Moreover, because the system (11) is of order $2n$, we have to compute basis vectors for the Krylov subspace of size $2n$. This seems rather unnatural since the system (9) is of order n. Can these drawbacks be avoided by applying a solution method directly to the quadratic problem (9)? The answer to this question is given in the next section.

4 The solution with the Jacobi-Davidson method.

We propose to solve (9) without reducing it to a standard eigenproblem. For our algorithm we use a generalization of the Jacobi-Davidson algorithm [11]. Algorithmic variants of this algorithm for generalized linear eigenproblems and of polynomial eigenproblems are described in [10]. A detailed description of the Jacobi-Davidson method for second order polynomial or quadratic eigenproblems can be found in [7]. In this section we will give a brief outline of the ideas that underly this algorithm.

We investigate an Newton process applied to the equation

$$r(x) = \lambda(x)^2 Mx + \lambda(x)Cx + Kx. \tag{16}$$

Here λ is a function of x and is a root of the quadratic form

$$\lambda^2 x^* Mx + \lambda x^* Cx + x^* Kx = 0. \tag{17}$$

We try to find an approximation for an x for which

$$r(x) = 0. \tag{18}$$

It follows that if x is a solution, then so is every scalar multiple of x. For this reason we focus only on the solutions with $\| x \| = 1$, because this reduces the solution to a finite number, namely 2. We start with a vector x_0 with $\| x_0 \| = 1$ and in each step of the Newton method we calculate an approximate solution $\overline{z}$ of the Jacobian system in x_i

$$J(x_i)z = -r(x_i). \tag{19}$$

Here J is the Jacobian of (16). If we evaluate this Jacobian we find for the Jacobian system (19):

$$(I - \frac{(2\theta Mx_i + Cx_i)x_i^*}{x_i^*(2\theta Mx_i + Cx_i)})(\theta^2 M + \theta C + K)z = -r(x_i). \tag{20}$$

An approximate solution $\overline{z}$ of this system of linear equations is obtained by performing a limited number of iterations of the GMRES method [9]. Because we restricted ourselves to $\| x_i \| = 1$, we have to add a projection $(I - \frac{x_i x_i^*}{x_i^* x_i})$ to the Jacobian and use $\Delta x = (I - \frac{x_i x_i^*}{x_i^* x_i})\overline{z}$ in stead of $\overline{z}$. We obtain the new approximation x_{i+1} for the next iteration by $x_{i+1} = x_i + \Delta x$.

To speed up the convergence of the inexact Newton method, we use an accelerated version. The algorithm is changed such that we do not only use the $\overline{z}$ from this iteration for updating x_i, but we also use all the previous directions to find the best approximation for the eigenvector x corresponding to the eigenvalue closest to a target value. We collect all directions $\overline{z}$ in a matrix V. The best vector in the space spanned by the columns of V, is selected by calculating the solutions of the projected quadratic eigenproblem

$$(\theta^{(i)})^2 V^* MVy + \theta^{(i)} V^* CVy + V^* KVy = 0. \tag{21}$$

Note that in step i, the matrices V^*MV, V^*CV, and V^*KV are of size $i \times i$, and therefore small compared with the original matrices M, C, and K. We select the solution (θ, y) for which θ is closest to a target value. We choose the target value in the area where we want to find an eigenvalue. Then $x^{(i)}$ is calculated by $x^{(i)} = Vy$. This comes down to calculating all solutions for which

$$(\theta^{(i)})^2 Mx^{(i)} + \theta^{(i)} Cx^{(i)} + Kx^{(i)} \perp span(V). \tag{22}$$

The $\theta^{(i)}$'s are called Ritz values and the vectors $x^{(i)}$ Ritz vectors. Because we use an accelerated version of this algorithm, we find more than one θ in each step of the algorithm. In iteration i, V is of dimension i as well and therefore we find $2i$ solutions of (21). To select one, we can choose a target value in advance, and select the Ritz value that is closest to the target value.

We recollect these steps in the following algorithm:

Jacobi-Davidson Algorithm
Select starting vector x. $V = [x]$. $i = 1$
Repeat:

1. Choose θ and y from solutions of $\theta^2 V^*MVy + \theta V^*CVy + V^*KVy = 0$.
2. $x = Vy$.
3. Compute an approximate solution $\overline{z}$ of
$$(I - \frac{(2\theta Mx + Cx)x^*}{x^*(2\theta Mx + Cx)})(\theta^2 M + \theta C + K)(I - \frac{xx^*}{x^*x})z = -r.$$
4. Orthonormalize $\overline{z}$ against V.
5. Extend V with $\overline{z}$: $V = [V, \overline{z}]$.
6. $i = i + 1$.

In exact arithmetic, the columns of V do not have to be orthogonalized, any other set of basis vectors for the column space of V will do as well. We orthogonalize the column vectors of V for reasons of numerical stability. The algorithm requires the storage of 4 sets of basis vectors: V, MV, CV, and KV. These bases are extended with one new basis vector in each iteration. Storage of these basis vectors may become a problem if many iterations have to be performed. To overcome this problem one may have to restart the algorithm.

5 Parallelization.

We have incorporated the Jacobi-Davidson algorithm in a finite element code [12] for the Cray T3D. The discretization is parallelized by a domain decomposition strategy. The computation of element matrices and vectors can be performed completely in parallel, no communication is needed. Solving the eigenproblem is a global operation and a certain amount of communication is unavoidable. The basic operations of this algorithm are matrix-vector multiplications, vector updates, inner product operations, and the computation of the Ritz values and -vectors. Almost all computations for the inner product can be performed in parallel, but (relatively expensive) communication between all processors is needed for assembling the partial results to yield the global inner product. In the matrix-vector multiplication we make use of the same domain decomposition as for the discretization. The matrix-vector multiplication requires communication, but only to exchange data between neighboring domains. For a systematic treatment of the influence of communication, see [4]. The communication operations are implemented with the Cray SHMEM_GET and SHMEM_PUT routines. A detailed description of the implementation of these operations can be found in [12]. The vector update operation can be performed completely in parallel without any communication. The computation of the Ritz values is restricted to the projected systems. These are small sized, the size of the projected system does not depend on the size of the large matrices. Computations for the projected system are therefore relatively inexpensive for large problems, and this operation has not been parallelized. The computations for the projected problem are performed on all processors to avoid unnecessary communication and synchronization.

6 Results.

We solved the problem with 16, 32, and 64 processors. Table 1 gives the elapsed times (in seconds) for the discretization of the problem.

Number of processors	Elapsed time
16	20.8
32	10.4
64	5.2

Table 1: Timings for the discretization phase.

The discretization phase does not require communication, and the timings confirm that the speed-up for this phase is linear.

We have used the Jacobi-Davidson method to compute the smallest eigenpair of the resulting quadratic problem. The vector with all coefficients equal to 1 is used to start the process. In each step an approximate solution of the Jacobian system is computed with 20 GMRES-iterations. The Jacobi-Davidson algorithm converges rapidly for our problem. Only 17 iterations are needed to reduce the norm of the residual corresponding to the smallest eigenvalue $-2.596 + 42.00i$ to 10^{-8}. Table 2 gives the timings for the solution phase.

Number of processors	Elapsed time
16	206.4
32	101.3
64	52.1

Table 2: Timings for the solution phase.

We observe a super linear speed-up from 16 to 32 processors, but this is probably due to a cache effect. The speed-up from 32 to 64 processors is almost linear. This indicates that the time for the global communication for the inner products is small relative to the computation time per processor.

The performance of the code is approximately 0.6 Gflop per second on 64 processors. The single processor performance is about 10 Mflop per second. A major reason for of this modest single processor performance (the peak performance is 150 Mflops) is the implementation of the matrix-vector multiplication. The regular structure of our model problem has not been exploited in this operation, so that our experiment mimics well the simulation for more irregular finite element computations. The matrix-vector multiplication is performed element-by-element. This technique avoids the assembly of the element matrices to a global matrix and is well suited for irregular grids. The drawback of the method is that it requires indirect addressing. The single processor performance of the matrix-vector multiplication is approximately 6 Mflops and 8 MIPS (million integer operations per second). A detailed description of the element-by-element matrix-vector multiplication can be found in [13].

We can make an estimate of the time that would have been needed to solve the model problem with the technique described in Section 3, i.e., when solving the standard eigenproblem (11) with Arnoldi's method. Let us assume that the performance on 64 processors for the back and forward substitution is 0.6 Gflops as well. This is rather optimistic, since the back and forward substitution are not very well parallelizable. Assuming a performance of 0.6 Gflops would mean that one back or forward substitution takes $4 \times mb \times n / 0.6 \cdot 10^9 \approx 1.5s$. To be faster than the Jacobi-Davidson method convergence should take place within $52s$. In this time only 17 back and forward substitutions can be performed, and hence at most 17 Arnoldi-iterations, which is unrealistically fast for a problem of this size.

With inverse iteration, the time for Choleski decomposition cannot be neglected, despite the fact that this operation has to be performed only once. The Choleski decomposition would take an additional $641s$., again assuming a performance of 0.6 Gflops. Even if we would make the unrealistic assumption that the Choleski decomposition for these sparse matrices can be performed at close to the peak-performance of 9.6 Gflops on 64 processors, the time for the decomposition would still be considerable: approximately $40s$, leaving little room for improvement by Arnoldi's method (only 4 steps in order to beat the Jacobi-Davidson method).

7 Concluding remarks.

The algorithm of section 4 can be extended in a number of ways. In our example the number of Jacobi-Davidson iterations to compute the smallest eigenvalue is modest. If many iterations are needed it may be necessary to restart the algorithm. E.g., the memory of the computer may be insufficient to store all basis vectors. It may also be necessary to restart the algorithm to limit the cost of the orthogonalization of the basis vectors and of the computation of the Ritz values and Ritz vectors. It is possible to restart the algorithm after a cycle of iterations with the Ritz vector belonging to the Ritz value closest to the target value. This Ritz vector is likely to have a reasonable component in the direction of the eigenvector we want to approximate (in any case it is the best approximation in the current subspace).

In our example we have computed only one eigenvalue. It may often be of interest to know more than one eigenvalue. There are several strategies possible to compute a number of eigenvalues. The most simple one is to change the target value once an eigenvalue has converged. A more subtle approach is to use deflation. The idea of this technique is to keep a basis for the converged eigenvectors in the search space, and to expand the search space V in a direction perpendicular to the converged eigenvectors. By this strategy we will refind the converged eigenvalues in every iteration and we can take a nonconverged eigenvalue as our target. Applying either of the two strategies for computing more than one eigenvalue has no consequences for the parallelization of the algorithm. No new operations are introduced in the algorithm.

The approximate solution of the Jacobian system can be improved by applying GMRES with a suitable preconditioning technique. This does introduce a new operation in the algorithm. Finding an effective and parallelizable preconditioner is still a topic of research.

A detailed description with illustrative examples of the improvements to the algorithm, in particular of restarting and deflation strategies, can be found in [7].

We conclude with a brief summary. Acoustic problems may lead to very large quadratic eigenproblems. We have discussed a generalization of the Jacobi-Davidson method for computing solutions of these problems. The algorithm does not require back- and forward substitutions, which are very expensive operations and difficult to parallelize. We have applied the Jacobi-Davidson algorithm to a "real live" problem, a finite element discretization of the wave equation plus boundary condition for a wall with damping. The parallel efficiency of the method was quite good. On the Cray T3D we observed linear speed-up going from 16 to 64 processors. Moreover, only a few iterations were sufficient to compute the smallest eigenpair.

Acknowledgements. We like to thank Marco Beltman for his extensive help with the acoustic problem. We thank Henk van der Vorst and Gerard Sleijpen for many helpful discussions on numerical details.

This work was sponsored by the National Computing Facilities Foundation (NCF) enabling the use of computer facilities, with financial support from the Netherlands Organization for Scientific Research (NWO), through its program for HPCN projects.

References

[1] W.E. Arnoldi. The principle of minimized iteration in the solution of the matrix eigenvalue problem. *Quart. Appl. Math.*, 9:17–29, 1951.

[2] W.M. Beltman, P.J.M van der Hoogt, R.M.E.J. Spiering, and H. Tijdeman. Application and experimental verification of DIANA for acousto-elastic problems. in *DIANA Computational Mechanics*, G.M.A Kusters and M.A.N Hendriks (eds), pp 215–224, Kluwer, Dordrecht, the Netherlands, 1994.

[3] A. Craggs. A finite element model for acoustically lined small rooms. *SIAM J. of Sound and Vibration*, 108(2):327–337, 1986.

[4] E. de Sturler and H.A. van de Vorst. Reducing the effect of global communication in GMRES(m) and CG on parallel distributed memory computers. Preprint 832, Utrecht University, Department of Mathematics, the Netherlands. To appear in *J. Appl. Num. Math.*.

[5] G.H. Golub and C. van Loan. *Matrix Computations.* North Oxford Academic, Oxford, 1984.

[6] P.M. Morse and K.U. Ingard. *Theoretical Acoustics.* McGraw Hill, New York, 1968.

[7] F.A. Raeven. *Quadratic eigenproblems.* Masters thesis, Utrecht University, June 1995.

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

[9] Y. Saad and M.H. Schultz. GMRES: A generalized minimum residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Statist. Comput.*, 7:856–869, 1986.

[10] G.L.G. Sleijpen, G.L. Booten, D.R. Fokkema, and H.A. van der Vorst. Jacobi-Davidson type methods for generalized eigenproblems and polynomial eigenproblems: Part I. Preprint 923, Utrecht University, Department of Mathematics, the Netherlands.

[11] G.L.G. Sleijpen and H.A. van der Vorst. A Jacobi-Davidson iteration method for linear eigenvalue problems. Preprint nr. 856, June 1994 (revised version January 1995), Utrecht University, Department of Mathematics, the Netherlands. To appear in *SIAM J. on Mat. Anal. Appl.*.

[12] 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*, Vol. 919, 1995, Springer Verlag.

[13] M.B. van Gijzen. Large scale finite element computations with GMRES-like methods on a Cray Y-MP. Preprint nr. 872, September 1994, Utrecht University, Department of Mathematics, the Netherlands. To appear in: *J. of Comput. and Appl. Math.*.