A Fully Parallel Condensation Method for Generalized Eigenvalue Problems on Distributed Memory Computers

Kai Rothe and Heinrich Voss

University of Hamburg, Institute of Applied Mathematics, Bundesstraße 55, 20146 Hamburg, Federal Republic of Germany
e-mail: rothe@math.uni-hamburg.de, voss@math.uni-hamburg.de

Abstract

For large-scale eigenvalue problems the authors introduced in [16] a coarse grained parallel algorithm for distributed memory computers based on substructuring and improved static condensation. It was observed that the load balancing may become unsatisfactory if the master problem and the problems corresponding to the substructures are of significantly different size. In this paper we improve the load balancing of the processors considerably by incorporating a parallel eigensolver for dense problems.

Keywords. eigenvalue problem, parallel algorithm, distributed memory MIMD computers, condensation, substructuring

1 Introduction

In the analysis of the dynamic response of a linear structure using finite element methods usually a large number of degrees of freedom is needed to model the behaviour of the system accurately enough. In this situation the dimension of the problem often is reduced to manageable size before proceeding to the calculation of the required natural frequencies and modes.

A common method to accomplish this reduction (called static condensation today) was introduced by Irons [9] and Guyan [7]. A relatively small number of variables (called masters) is chosen which seem to be representative for the dynamic behaviour of the structure and the static relationship between these masters and the remaining degrees of freedom (termed slaves) is employed to eliminate the slave variables.

It has frequently been noted that the quality of the eigenvalue and eigenvector approximations produced by static condensation is satisfying only for the lower part of the spectrum. Several attempts have been made to improve the quality (cf. [1], [2], [3], [10], [11], [12], [18], e.g.) most of them being very time consuming since an iterative process is involved. In [15] the authors proposed an enhancement of the eigenvalue and eigenvector approximations exploiting properties of the exactly condensed problem which is a nonlinear eigenvalue problem of the same dimension as the statically condensed problem and which has the same eigenvalues as the uncondensed problem.

In [16] the improved condensation was combined with substructuring. Choosing the master variables as degrees of freedom on the interfaces of the substructures and the slave variables as interior ones the condensation process splits into independent subproblems which can be solved fully in parallel. Moreover, most of the computation in the enhancement of the eigenvalues and eigenvectors can be done in parallel, too. Thus we obtained a parallel solver of very coarse granularity , where one process was allocated to the master problem and one to each substructure.

It was already observed in [16] that the efficiency of this coarse grained parallel eigensolver is influenced crucially by the following fact. If the dimension of the master problem is significantly larger than the dimensions of the problems corresponding to the substructures the solution of the condensed eigenvalue problem dominates the computing time of one phase of the algorithm. Hence in this phase which consumes nearly half of the total computing time the processors corresponding to slave processes are idle for up to 75% which reduces the overall efficiency of the eigensolver drastically.

It is the purpose of this paper to present a fully parallel condensation method on distributed memory MIMD computers. A parallel eigensolver for dense generalized eigenvalue problems described in [13] and [14] is incorporated which is of finer granularity than the overall concept and which leads to good load balancing of the processors.

2 Condensation of eigenvalue problems

The dynamic analysis of structures by finite element methods often leads to large-scale eigenvalue problems

$$\boldsymbol{K}\boldsymbol{x} = \lambda \boldsymbol{M}\boldsymbol{x}, \tag{1}$$

where the stiffness matrix $\boldsymbol{K} \in \mathbb{R}^{(n,n)}$ and the mass matrix $\boldsymbol{M} \in \mathbb{R}^{(n,n)}$ are real symmetric and positive definite, $\boldsymbol{x}$ is the vector of modal displacements, and λ is the square of the angular frequency.

A common technique for reducing the number of degrees of freedom introduced by Irons [9] and Guyan [7] consists in selecting (a fairly large number of) components of the vector $\boldsymbol{x}$ called slaves to be eliminated and to condense problem (1) to a smaller eigenvalue problem with respect to the remaining variables called masters. To this end eqn. (1) is rewritten in block form

$$\begin{bmatrix} \boldsymbol{K}_{mm} & \boldsymbol{K}_{ms} \\ \boldsymbol{K}_{sm} & \boldsymbol{K}_{ss} \end{bmatrix} \begin{Bmatrix} \boldsymbol{x}_m \\ \boldsymbol{x}_s \end{Bmatrix} = \lambda \begin{bmatrix} \boldsymbol{M}_{mm} & \boldsymbol{M}_{ms} \\ \boldsymbol{M}_{sm} & \boldsymbol{M}_{ss} \end{bmatrix} \begin{Bmatrix} \boldsymbol{x}_m \\ \boldsymbol{x}_s \end{Bmatrix} \tag{2}$$

where $\boldsymbol{x}_m \in \mathbb{R}^m$ and $\boldsymbol{x}_s \in \mathbb{R}^s$ denotes the vector of the master variables and of slave variables, respectively.

From the second row of eqn. (2) one obtains the exact transformation relation

$$\boldsymbol{x}_s = -(\boldsymbol{K}_{ss} - \lambda \boldsymbol{M}_{ss})^{-1}(\boldsymbol{K}_{sm} - \lambda \boldsymbol{M}_{sm})\boldsymbol{x}_m \tag{3}$$

between the slave part $\boldsymbol{x}_s$ and the master portion $\boldsymbol{x}_m$ of $\boldsymbol{x}$. Substituting (3) in the first row of eqn. (2) yields the exactly condensed eigenvalue problem

$$\boldsymbol{T}(\lambda)\boldsymbol{x}_m = \mathbf{o}, \tag{4}$$

where

$$\boldsymbol{T}(\lambda) := -\boldsymbol{K}_{mm} + \lambda \boldsymbol{M}_{mm} + (\boldsymbol{K}_{ms} - \lambda \boldsymbol{M}_{ms})(\boldsymbol{K}_{ss} - \lambda \boldsymbol{M}_{ss})^{-1}(\boldsymbol{K}_{sm} - \lambda \boldsymbol{M}_{sm}). \tag{5}$$

Observe that the nonlinear eigenvalue problem (4) of (small) dimension m is equivalent to the original problem (1), if the matrix $\boldsymbol{K}_{ss} - \lambda_j \boldsymbol{M}_{ss}$ is regular for every eigenvalue λ_j of eqn. (1), i.e. if (1) and the slave eigenvalue problem

$$\boldsymbol{K}_{ss}\boldsymbol{\phi} = \omega \boldsymbol{M}_{ss}\boldsymbol{\phi} \tag{6}$$

do not have eigenvalues in common. The prize to be paid for this nice property is the nonlinearity of the exactly condensed problem.

Notice that the nonlinear eigenvalue problem can be interpreted as the result of a projection of eqn. (1) to the subspace

$$\left\{ \begin{pmatrix} \boldsymbol{I} \\ -(\boldsymbol{K}_{ss} - \lambda \boldsymbol{M}_{ss})^{-1}(\boldsymbol{K}_{sm} - \lambda \boldsymbol{M}_{sm}) \end{pmatrix} \boldsymbol{x}_m \; : \; \boldsymbol{x}_m \in \mathbb{R}^m \right\}.$$

where the subspace is not fixed but it depends on the eigenparameter λ.

Usually in the literature it is argued that for low frequencies the effects of inertia forces on the slave displacements is small compared to the effects of static forces, and therefore the inertia terms in the second equation of (2) can be ignored. Hence, the exact transformation relation (3) is replaced by

$$\boldsymbol{x}_s \approx -\boldsymbol{K}_{ss}^{-1}\boldsymbol{K}_{sm}\boldsymbol{x}_m,$$

and the large-scale eigenvalue problem (1) is projected to the subspace

$$\left\{ \begin{pmatrix} \boldsymbol{I} \\ -\boldsymbol{K}_{ss}^{-1}\boldsymbol{K}_{sm} \end{pmatrix} \boldsymbol{x}_m \; : \; \boldsymbol{x}_m \in \mathbb{R}^m \right\}.$$

This leads to the so called statically condensed problem

$$\boldsymbol{K}_0 \boldsymbol{x}_m = \lambda \boldsymbol{M}_0 \boldsymbol{x}_m \tag{7}$$

where the reduced stiffness matrix $\boldsymbol{K}_0$ and the reduced mass matrix $\boldsymbol{M}_0$ are given by

$$\begin{array}{rcl} \boldsymbol{K}_0 & := & \boldsymbol{K}_{mm} - \boldsymbol{K}_{ms}\boldsymbol{K}_{ss}^{-1}\boldsymbol{K}_{sm}, \\ \boldsymbol{M}_0 & := & \boldsymbol{M}_{mm} - \boldsymbol{K}_{ms}\boldsymbol{K}_{ss}^{-1}\boldsymbol{M}_{sm} - \boldsymbol{M}_{ms}\boldsymbol{K}_{ss}^{-1}\boldsymbol{K}_{sm} \\ & & +\boldsymbol{K}_{ms}\boldsymbol{K}_{ss}^{-1}\boldsymbol{M}_{ss}\boldsymbol{K}_{ss}^{-1}\boldsymbol{K}_{sm}. \end{array} \tag{8}$$

It is well known that only a small number of eigenvalues of problem (1) in the lower part of the spectrum can be approximated sufficiently well by static condensation (cf. the a priori bounds in [17]). Several attempts have been made to improve its approximation properties. In [1], [2], [11], [12] and [18], among others, inverse iteration is applied to the exactly condensed problem, in [3] and [10] polynomial approximations to $\boldsymbol{T}(\lambda)$ are considered and are solved iteratively. All of these methods are quite costly since the approximations to different eigenvalues of (1) have to be improved individually by an iterative process where in every step a reduced matrix $\boldsymbol{T}(\bar{\lambda})$ according to (5) for some parameter $\bar{\lambda}$ or a polynomial approximation of $\boldsymbol{T}(\bar{\lambda})$ has to be calculated.

In [15] the authors introduced the following method to enhance the eigenvalue approximations obtained from static condensation which avoids the solution of the exactly condensed problem. We took advantage of the fact that there exists a Rayleigh functional of the nonlinear problem (4) which has similar approximation properties as the Rayleigh quotient for linear eigenvalue problems.

To define the Rayleigh functional we rewrite $\boldsymbol{T}(\lambda)$ in more convenient form. Let $\boldsymbol{\Phi} \in \mathbb{R}^{(n,n)}$ and $\boldsymbol{\Omega} := \mathrm{diag}\,[\omega_i] \in \mathbb{R}^{(n,n)}$ be the modal and the spectral matrix of the slave eigenvalue problem (6), respectively, where $\boldsymbol{\Phi}$ is normalized by

$$\boldsymbol{\Phi}^t \boldsymbol{M}_{ss} \boldsymbol{\Phi} = \boldsymbol{I} \quad \text{and} \quad \boldsymbol{\Phi}^t \boldsymbol{K}_{ss} \boldsymbol{\Phi} = \boldsymbol{\Omega}. \tag{9}$$

Then (cf. Leung [11], e.g.)

$$\boldsymbol{T}(\lambda) = -\boldsymbol{K}_0 + \lambda \boldsymbol{M}_0 + \boldsymbol{S}\boldsymbol{D}(\lambda)\boldsymbol{S}^t, \tag{10}$$

where the reduced matrices $\boldsymbol{K}_0$ and $\boldsymbol{M}_0$ are given in eqn. (8) and

$$\begin{array}{rcl} \boldsymbol{S} & := & \boldsymbol{M}_{ms}\boldsymbol{\Phi} - \boldsymbol{K}_{ms}\boldsymbol{\Phi}\boldsymbol{\Omega}^{-1}, \\ \boldsymbol{D}(\lambda) & := & \mathrm{diag}\,[\lambda^2/(\omega_i - \lambda)]\,. \end{array} \tag{11}$$

Notice that the nonlinearity of $\boldsymbol{T}(\lambda)$ is concentrated in the diagonal matrix $\boldsymbol{D}(\lambda)$ and that the statically condensed problem is the linearization of the exactly condensed problem at $\hat{\lambda} = 0$.

The Rayleigh quotient of a linear eigenvalue problem

$$\boldsymbol{V}(\lambda)\boldsymbol{x} := (\boldsymbol{K} - \lambda\boldsymbol{M})\boldsymbol{x} = \mathbf{o}$$

at some vector $\bar{\boldsymbol{x}} \neq \mathbf{o}$ is the unique solution of the real equation $\bar{\boldsymbol{x}}^t\boldsymbol{V}(\lambda)\bar{\boldsymbol{x}} = 0$. The Rayleigh functional of a nonlinear eigenvalue problem is defined correspondingly.

Let J be the open interval $J := (0, \underline{\omega})$ where $\underline{\omega}$ denotes the smallest eigenvalue of the slave eigenvalue problem (6). Then, for any fixed vector $\boldsymbol{u} \in \mathbb{R}^m \setminus \{\mathbf{o}\}$ the real valued function

$$f(\cdot, \boldsymbol{u}) \,:\, J \to \mathbb{R} \quad , \quad \lambda \mapsto f(\lambda, \boldsymbol{u}) := \boldsymbol{u}^t\boldsymbol{T}(\lambda)\boldsymbol{u},$$

i.e.

$$f(\lambda, \boldsymbol{u}) = -\boldsymbol{u}^t\boldsymbol{K}_0\boldsymbol{u} + \lambda\boldsymbol{u}^t\boldsymbol{M}_0\boldsymbol{u} + \sum_{i=1}^{s}\frac{\lambda^2}{\omega_i - \lambda}\Big(\boldsymbol{\phi}_i^t\boldsymbol{M}_{sm}\boldsymbol{u} - \frac{1}{\omega_i}\boldsymbol{\phi}_i^t\boldsymbol{K}_{sm}\boldsymbol{u}\Big)^2 \qquad (12)$$

is strictly monotonically increasing (Observe that $\boldsymbol{M}_0$ is a positive definite matrix). Hence, the nonlinear equation $f(\lambda, \boldsymbol{u}) = 0$ has at most one solution in J, and therefore, it implicitly defines a functional

$$p \,:\, \mathbb{R}^m \supset D(p) \to J \quad , \quad f(p(\boldsymbol{u}), \boldsymbol{u}) = 0.$$

p is called the Rayleigh functional of the nonlinear eigenvalue problem (4).

p has similar approximation properties as the Rayleigh quotient in the linear case. Since the eigenvectors of $\boldsymbol{T}$ are stationary points of p, a first order error of an eigenvector approximation $\bar{\boldsymbol{u}}$ yields a second order approximation for the corresponding eigenvalue by the value of the Rayleigh functional $p(\bar{\boldsymbol{u}})$ (if $\bar{\boldsymbol{u}} \in D(p)$).

Let $\bar{\lambda}_1 \leq \bar{\lambda}_2 \leq \ldots \leq \bar{\lambda}_k \leq \underline{\omega}$ be the eigenvalues of the statically condensed eigenvalue problem (7) which are contained in J and denote by $\bar{\boldsymbol{u}}_j$ an eigenvector corresponding to $\bar{\lambda}_j$. Then it follows from

$$f(0, \bar{\boldsymbol{u}}_j) = -\bar{\boldsymbol{u}}_j^t\boldsymbol{K}_0\bar{\boldsymbol{u}}_j < 0 \leq \bar{\boldsymbol{u}}_j^t\boldsymbol{S}\boldsymbol{D}(\bar{\lambda}_j)\boldsymbol{S}^t\bar{\boldsymbol{u}}_j = f(\bar{\lambda}_j, \bar{\boldsymbol{u}}_j) \qquad (13)$$

that $p(\bar{\boldsymbol{u}}_j)$, $j = 1, \ldots, k$, is defined and it can be expected to be a much better approximation to the j-th eigenvalue λ_j of the original eigenvalue problem (1) than $\bar{\lambda}_j$ (cf. the error estimation in [15]).

It is our experience that for those eigenpairs $(\bar{\lambda}_j, \bar{\boldsymbol{u}}_j)$ of the statically condensed problem with $\bar{\lambda}_j \leq 0.5 \cdot \underline{\omega}$ the enhanced values $p(\bar{\boldsymbol{u}}_j)$ have relative errors less than 1% (cf. [8], [15], [16] and the example in Section 6). For $\bar{\lambda}_j > 0.5\underline{\omega}$ the approximation of the master part of an eigenvector of eqn. (1) by $\bar{\boldsymbol{u}}_j$ usually is not good enough to reduce the relative error to less than 1%. In [15] we discussed a shift strategy to cope with eigenvalues in the interval $(0.5 \cdot \underline{\omega}\,,\, \underline{\omega})$.

3 Substructuring

In the free vibration analysis of structures all steps of the condensation process and the enhancement of the eigenvalue approximations, namely the reduction of the eigenvalue problem (1), the solution of the slave eigenproblem, the evaluation of the Rayleigh functional and the computation of the slave portions of the eigenvector approximations can be done in parallel if substructuring is used. Although this technique is well known we shall briefly comment on its computational merits, since we shall need the notation.

Assume that the entire structure is divided into r substructures (cf. Figure 2) and that the vibration problem is discretized (by finite elements or finite differences) such that $k_{ij} = 0$ and $m_{ij} = 0$ whenever i and j are indices of interior nodes of different substructures. Suppose that all degrees of freedom on the boundaries of the substructures (and to improve the approximation properties additionally some of the interior degrees of freedom, cf. [8]) are chosen as masters and the (remaining) interior degrees of freedom are chosen as slaves. Then if the variables are numbered in an appropriate way the stiffness matrix is given by

$$\boldsymbol{K} = \begin{bmatrix} \boldsymbol{K}_{mm} & \boldsymbol{K}_{ms1} & \boldsymbol{K}_{ms2} & \dots & \boldsymbol{K}_{msr} \\ \boldsymbol{K}_{sm1} & \boldsymbol{K}_{ss1} & \boldsymbol{O} & \dots & \boldsymbol{O} \\ \boldsymbol{K}_{sm2} & \boldsymbol{O} & \boldsymbol{K}_{ss2} & \dots & \boldsymbol{O} \\ \dots & \dots & \dots & \dots & \dots \\ \boldsymbol{K}_{smr} & \boldsymbol{O} & \boldsymbol{O} & \dots & \boldsymbol{K}_{ssr} \end{bmatrix},$$

and the mass matrix $\boldsymbol{M}$ has the same block structure.

From the block form of the matrices $\boldsymbol{K}$ and $\boldsymbol{M}$ we get the reduced matrices

$$\boldsymbol{K}_0 = \boldsymbol{K}_{mm} - \sum_{j=1}^{r} \boldsymbol{K}_{mmj}\ , \quad \boldsymbol{M}_0 = \boldsymbol{M}_{mm} - \sum_{j=1}^{r} \boldsymbol{M}_{mmj},$$

where

$$\boldsymbol{K}_{mmj} := \boldsymbol{K}_{msj}\boldsymbol{K}_{ssj}^{-1}\boldsymbol{K}_{smj}$$

and

$$\boldsymbol{M}_{mmj} := \boldsymbol{K}_{msj}\boldsymbol{K}_{ssj}^{-1}\boldsymbol{M}_{smj} + \boldsymbol{M}_{msj}\boldsymbol{K}_{ssj}^{-1}\boldsymbol{K}_{smj} - \boldsymbol{K}_{msj}\boldsymbol{K}_{ssj}^{-1}\boldsymbol{M}_{ssj}\boldsymbol{K}_{ssj}^{-1}\boldsymbol{K}_{smj}.$$

The evaluation of the matrices $\boldsymbol{K}_{mmj}$ and $\boldsymbol{M}_{mmj}$, $j = 1, \dots, r$, only requires data from the j-th substructure and therefore can be done completely in parallel in the following way. Solve simultaneously the linear systems of equations

$$\boldsymbol{K}_{ssj}\boldsymbol{X}_j := \boldsymbol{K}_{smj} \tag{14}$$

and determine

$$\begin{aligned} \boldsymbol{K}_{mmj} &= \boldsymbol{K}_{msj}\boldsymbol{X}_j, \\ \boldsymbol{M}_{mmj} &= \boldsymbol{Y}_j + \boldsymbol{Y}_j^t - \boldsymbol{X}_j^t\boldsymbol{M}_{ssj}\boldsymbol{X}_j, \quad \boldsymbol{Y}_j := \boldsymbol{M}_{msj}\boldsymbol{X}_j. \end{aligned}$$

Observe that only those columns of $\boldsymbol{K}_{smj}$ are different from the nullvector which correspond to master degrees of freedom in the j-th substructure and therefore only those right hand sides have to be considered in (14).

The matrices $\boldsymbol{K}_{ss}$ and $\boldsymbol{M}_{ss}$ are block diagonal. Hence the slave eigenvalue problem splits into r independent eigenvalue problems

$$\boldsymbol{K}_{ssj}\boldsymbol{\phi}_j = \omega \boldsymbol{M}_{ssj}\boldsymbol{\phi}_j, \quad j = 1, \ldots, r. \tag{15}$$

which can be solved independently.

Moreover, the modal matrix $\boldsymbol{\Phi} = \text{diag}\,[\boldsymbol{\Phi}_j]$ and the spectral matrix $\boldsymbol{\Omega} = \text{diag}\,[\boldsymbol{\Omega}_j]$ of the slave problem are block diagonal. Thus for $\bar{\boldsymbol{u}} \in \mathbb{R}^m \setminus \{\mathbf{o}\}$

$$\boldsymbol{S}^t\bar{\boldsymbol{u}} = \left\{ \begin{array}{c} \vdots \\ \boldsymbol{\Phi}_j^t \boldsymbol{M}_{smj}\bar{\boldsymbol{u}} - \boldsymbol{\Omega}_j^{-1}\boldsymbol{\Phi}_j^t \boldsymbol{K}_{smj}\bar{\boldsymbol{u}} \\ \vdots \end{array} \right\}, \tag{16}$$

and those components of the vector $\boldsymbol{S}^t\bar{\boldsymbol{u}}$ which belong to different substructures again can be evaluated in parallel.

Finally the computation of the slave portion $\boldsymbol{x}_s$ corresponding to an eigenvector $\bar{\boldsymbol{u}}$ of the slave problem can be done in parallel as well

$$(\boldsymbol{K}_{ssj} - p(\bar{\boldsymbol{u}})\boldsymbol{M}_{ssj})(\boldsymbol{x}_s)_j = -(\boldsymbol{K}_{smj} - p(\bar{\boldsymbol{u}})\boldsymbol{M}_{smj})\bar{\boldsymbol{u}}, \quad j = 1, \cdots, r\,.$$

4 Substructuring and parallel processes

To each substructure we attach one process named 'S_j' and with the master information we associate one further process called 'Ma'. These processes work in parallel as shown in Table 1. For the necessary communication each 'S_j' is connected to 'Ma' directly or indirectly. In our example we have chosen a tree topology which is shown in Figure 3.

In step 2 of the parallel concept the condensed problem and the slave eigenproblems may be solved (in parallel but each of them sequentially) by appropriate library routines from EISPACK [5], e.g. It turns out however, that these eigenproblems in general are of extremely different size and sparseness (cf. [16] and the example in section 6). A dominating condensed problem (7) which usually is dense leads to an unbalanced load of the processors in this phase and consequently the performance of the entire algorithm becomes unsatisfactory. To achieve an improved balance we incorporated a parallel algorithm for generalized dense symmetric eigenvalue problems which is described briefly in the next section. A detailed description is contained in [14].

Table 1 shows how this parallel eigensolver which consists of the processes 'Ma' and 'R_1',..., 'R_t' is integrated into the whole concept. A possible connection topology for all processes is shown in Figure 3. We anticipate here that the idle times of some processes do not have significant influence on the load balancing because of the multiprocessing capability of the processors in use.

process/ step	Ma	$R_i \quad i=1,\cdots,t$ (optional)	$S_j \qquad j=1,\cdots,r$
initialization			
1	compute $\boldsymbol{K}_{mm}, \boldsymbol{M}_{mm}$	idle	compute $\boldsymbol{K}_{ssj}, \boldsymbol{K}_{smj}, \boldsymbol{M}_{ssj}, \boldsymbol{M}_{smj}$ solve $\boldsymbol{K}_{ssj}\boldsymbol{X}_j = \boldsymbol{K}_{smj}$ for $\boldsymbol{X}_j$; $\boldsymbol{K}_{mmj} := \boldsymbol{K}_{msj}\boldsymbol{X}_j$; $\boldsymbol{Y}_j := \boldsymbol{M}_{msj}\boldsymbol{X}_j$; $\boldsymbol{M}_{mmj} := \boldsymbol{Y}_j + \boldsymbol{Y}_j{}^t - \boldsymbol{X}_j{}^t\boldsymbol{M}_{ssj}\boldsymbol{X}_j$
communication			
2	solve $\boldsymbol{K}_0\bar{\boldsymbol{u}}_\mu = \bar{\lambda}_\mu \boldsymbol{M}_0\bar{\boldsymbol{u}}_\mu$ $\mu=1,\cdots,l$		solve $\boldsymbol{K}_{ssj}\boldsymbol{\Phi}_j = \boldsymbol{M}_{ssj}\boldsymbol{\Phi}_j\boldsymbol{\Omega}_j$
communication			
3	compute $\bar{\boldsymbol{u}}_\mu^t\boldsymbol{K}_0\bar{\boldsymbol{u}}_\mu, \bar{\boldsymbol{u}}_\mu^t\boldsymbol{M}_0\bar{\boldsymbol{u}}_\mu$ $\mu=1,\cdots,l$	idle	compute $(\boldsymbol{S}^t\bar{\boldsymbol{u}}_\mu)_j = (\boldsymbol{\Phi}_j^t\boldsymbol{M}_{smj} - \boldsymbol{\Omega}_j^{-1}\boldsymbol{\Phi}_j^t\boldsymbol{K}_{smj})\bar{\boldsymbol{u}}_\mu$ $\mu=1,\cdots,l$
communication			
4	compute $p(\bar{\boldsymbol{u}}_\mu)$ from $f(\lambda,\bar{\boldsymbol{u}}_\mu)=0$ $\mu=1,\cdots,l$	idle	idle
communication			
5	idle	idle	compute $(\boldsymbol{K}_{ssj} - p(\bar{\boldsymbol{u}}_\mu)\boldsymbol{M}_{ssj})(\boldsymbol{x}_{s,\mu})_j = -(\boldsymbol{K}_{smj} - p(\bar{\boldsymbol{u}}_\mu)\boldsymbol{M}_{smj})\bar{\boldsymbol{u}}_\mu$ $\mu=1,\cdots,l$

Table 1. Steps of parallel computation

5 Parallel solution of dense eigenproblems

In this section we briefly sketch the concept of parallelization for solving the dense generalized eigenvalue problem (7). It mainly consists in the parallelization of the linear algebra commonly used for solving generalized eigenvalue problems, namely Cholesky decomposition of the mass matrix, transformation to a special eigenvalue problem, further transformation to tridiagonal form, determination of eigenvalues and eigenvectors and two steps of back transformation of the eigenvectors.

The main criterion for the choice of the procedures was to avoid unnecessary communication between the steps caused by data restructuring, i.e. it was essential that the output of one step is identical to the input of the following one. To keep this principle additional arithmetic effort was accepted in step 2 of the algorithm in Table 2.

The source code of the parallel algorithm with a detailed description is contained in [14], a brief synopsis in [13].

step	description	formula	subroutine
	generalized problem	$\boldsymbol{K}_0\boldsymbol{u} = \lambda\boldsymbol{M}_0\boldsymbol{u}$	
1.	Cholesky decomposition	$\boldsymbol{L}\boldsymbol{L}^T := \boldsymbol{M}_0$	CHOLESKY
2.	reduction	$\boldsymbol{L}\boldsymbol{C} := \boldsymbol{K}_0$	FORWARD
3.		$\boldsymbol{A}\boldsymbol{L}^T := \boldsymbol{C}$	ALTEQC
	standard problem	$\boldsymbol{A}\boldsymbol{y} = \lambda\boldsymbol{y}$	
4.	Householder reduction	$\boldsymbol{T} := \boldsymbol{H}_{n-2}\cdots\boldsymbol{H}_1$ $\cdot\boldsymbol{A}\boldsymbol{H}_1\cdots\boldsymbol{H}_{n-2}$	HOUSEHOLDER
	tridiagonal problem	$\boldsymbol{T}\boldsymbol{z} = \lambda\boldsymbol{z}$	
5.	eigenvalue computation		EIGENVALUES
6.	eigenvector computation		EIGENVECTORS
7.	back transformation	$\boldsymbol{y} = \boldsymbol{H}_{n-2}\cdots\boldsymbol{H}_1\boldsymbol{z}$	TRANSBAK1
8.	back transformation	$\boldsymbol{y} = \boldsymbol{L}^T\boldsymbol{u}$	TRANSBAK2
9.			COLLECT

Table 2. Standard algorithm for $\boldsymbol{K}_0\boldsymbol{u} = \lambda\boldsymbol{M}_0\boldsymbol{u}$

For the parallel version of the standard algorithm the processes named 'Ma', 'R_1',..., 'R_t' are created. At the beginning 'Ma' holds the data of the eigenvalue problem, it initializes the ring topology in use and at the end it collects the eigenvalues and eigenvectors. During the computing phase the processes 'Ma', 'R_1',..., 'R_t' run identical codes.

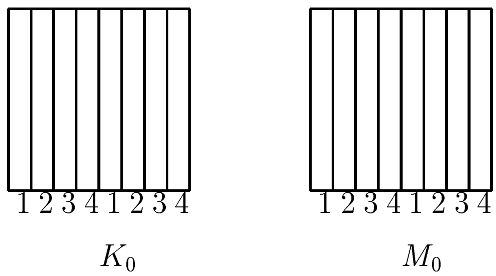

Figure 1. Columnwise wrapped mapping of $\boldsymbol{K}_0$ and $\boldsymbol{M}_0$ to four processes

We parallelize the nine computational steps of the standard algorithm by nine corresponding subroutines (cf. Table 2) which are called simultaneously by each process.

The main feature of an efficient parallel algorithm is the data handling. In our case we start with the wrapped mapping of the columns of $\boldsymbol{K}_0$ and $\boldsymbol{M}_0$ to the processes (cf. Figure 1) which is established in the initializing step.

The algorithm then continues with the wrapped mapping of the columns of each matrix involved.

The algorithm, in particular the mode of working of the subroutines given in Table 2, is briefly described in the following.

CHOLESKY: For the Cholesky decomposition we choose an algorithm that is recommended by Golub and Van Loan [6]. It starts with a columnwise wrapped mapping of $\boldsymbol{M}_0$ to the ring. Process 1 (i.e. 'Ma') computes the first column $\boldsymbol{\ell}^1$ of the Cholesky factor $\boldsymbol{L}$ (notice that the only information needed is the first column of $\boldsymbol{M}_0$ which is contained in the memory of 'Ma'). Then it sends $\boldsymbol{\ell}^1$ to its right neighbour process 2 ('R_1') and dates up the remaining columns of $\boldsymbol{M}_0$ which are contained in its local memory. 'R_1' receives $\boldsymbol{\ell}^1$ and sends it to its right neighbour (if this is not 'Ma'). It dates up its columns of $\boldsymbol{M}_0$ and computes the second column $\boldsymbol{\ell}^2$ of $\boldsymbol{L}$. Afterwards it sends $\boldsymbol{\ell}^2$ to its right neighbour and dates up the rest of its columns of $\boldsymbol{M}_0$ using $\boldsymbol{\ell}^2$. The processes 'R_2' to 'R_t' proceed in the same way. The algorithm closes with a columnwise distribution of the columns of $\boldsymbol{L}$ which have overwritten the columns of $\boldsymbol{M}_0$.

FORWARD: Input data are $\boldsymbol{K}_0$ and $\boldsymbol{L}$ distributed columnwise in a wrapped fashion in the ring. The processes rotate the columns $\boldsymbol{\ell}^1, \ldots, \boldsymbol{\ell}^{t+1}$ counter-clockwise through the ring. At the end each process holds all columns $\boldsymbol{\ell}^1, \ldots, \boldsymbol{\ell}^{t+1}$ with which the first $t+1$ rows of $\boldsymbol{C}$ are computed fully in parallel. Then the columns $\boldsymbol{\ell}^{t+2}, \ldots \boldsymbol{\ell}^{2t+2}$ are communicated throughout the ring and the next $t+1$ forward substitution steps are performed, etc. At the end of FORWARD the matrix $\boldsymbol{K}_0$ is overwritten by $\boldsymbol{C}$. Although in step 3 only the lower triangle of the matrix $\boldsymbol{C}$ would be needed to compute the lower triangle of $\boldsymbol{A}$ we provide the whole matrix $\boldsymbol{C}$ in this step to preserve the column oriented data distribution.

ALTEQC: This algorithm works exactly in the same way as CHOLESKY with the slight difference that the currently involved columns of $\boldsymbol{A}$ and $\boldsymbol{L}$ have to be rotated through the ring which causes little more communication.

HOUSEHOLDER: $\boldsymbol{A}$ is transformed to tridiagonal form via Householder matrices using an adaptation of an algorithm from Dongarra and van de Geijn [4] to the ring topology. The algorithm starts with the columnwise wrapped mapping of $\boldsymbol{A}$ and closes with $\boldsymbol{T}$ and the Householder information distributed in the same way.

EIGENVALUES: Initially the columnwise distributed information of $\boldsymbol{T}$ is communicated to all processes. Then each process computes one segment of the spectrum of $\boldsymbol{T}$, e.g. by bisection. Again this can be done fully in parallel.

EIGENVECTORS: Each process holds the necessary information (without communication) to compute the eigenvectors of its segment which can be done by inverse iteration, e.g.

TRANSBAK1: The computed eigenvectors of $\boldsymbol{T}$ are transformed back to those of the matrix $\boldsymbol{A}$ by rotating the columnwise distributed information of the Householder matrices (cf. HOUSEHOLDER) through the ring.

TRANSBAK2: The second back transformation involves the Cholesky factor again. The eigenvectors are updated by the columns of $\boldsymbol{L}$ which have to be rotated through the ring.

COLLECT: In a final step the eigenpairs of problem (1) are collected in the process 'Ma' for further treatment.

$n\,/\,p$	1	2	3	4
100	0.658	1.168	1.461	1.601
200	0.646	1.237	1.731	2.078
300	0.643	1.255	1.805	2.283
400	0.642	1.262	1.844	2.369

Table 3. Speedup Parallel Solver compared with EISPACK

Table 3 contains the speedup of the presented algorithm as compared to the corresponding sequence of EISPACK routines where the first ten eigenpairs are computed. n is the dimension of the problem and p is the number of processors. Notice that in step 2 additional arithmetic work is done in our algorithm to preserve the data structure. If the parallel algorithm is compared to its sequential version then a speedup of 3.69 results for dimension $n = 400$ and $p = 4$ processors.

In section 6 we consider a condensed problem of dimension $n = 164$. According to the results above the computing time for $p = 4$ processes is roughly halved (cf. Figure 4 and 5, especially step 2).

6 Numerical results

The parallel concept was tested on a distributed memory PARSYTEC transputer system equipped with T800 INMOS transputers (25MHz, 4 MB RAM) under the distributed operating system 'helios'. Since each processor has a multiprocessing capability we were able to execute more than one process from Table 1 on every processor which turned out to be extremly important for a good load balancing of the system.

We do not discuss the mapping of the process topology to the processor network. The operating system 'helios' has the option to do this mapping automatically, although this is not always the fastest solution. In our example with eight substructures we tested different mappings, and we report on the results of the best (fastest) one. However, for problems with a larger number of processes this point requires further investigations.

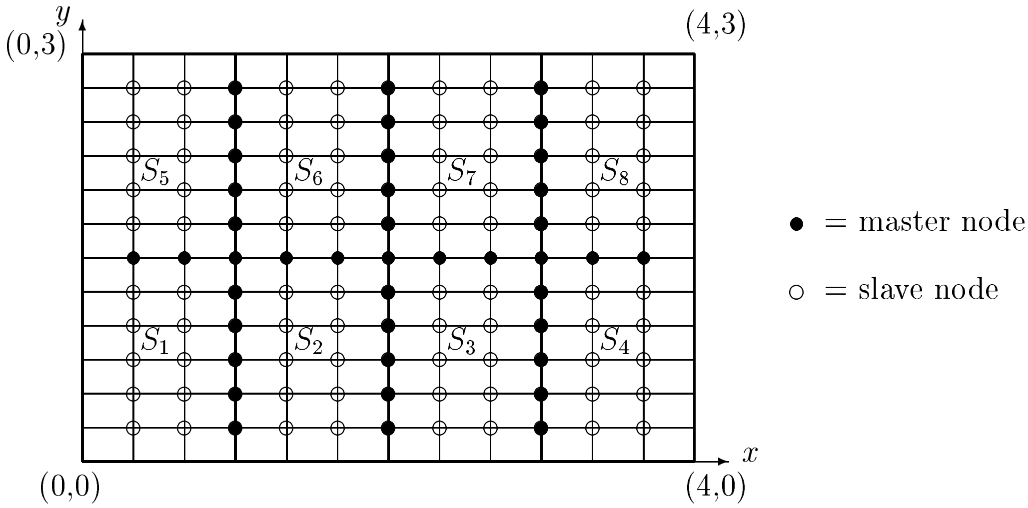

Figure 2. Discretized clamped plate (8 substructures)

Consider the free vibrations of a uniform thin clamped plate covering the rectangular region $\Omega := [0,4] \times [0,3]$ which is governed by the eigenvalue problem

$$\begin{aligned} \Delta^2 u &= \lambda u \quad \text{in } \Omega \\ u = \frac{\partial u}{\partial n} &= 0 \quad \text{on } \partial\Omega. \end{aligned} \tag{17}$$

The problem is discretized by 144 rectangular Bogner-Fox-Schmidt elements (node variables u, u_x, u_y, u_{xy}) yielding an uncondensed problem (1) of dimension $n = 484$ (cf. Figure 2).

We divided Ω into $r = 8$ identical substructures of edge length 1 in x-direction and 3/2 in y-direction. Thus, the master problem is of dimension $m = 164$, and the slave problem splits into 8 independent eigenproblems of dimension $s/r = 40$. The minimal slave eigenvalue is $\underline{\omega} \approx 734.71$.

j	λ_j	$\bar\lambda_j$	$(\bar\lambda_j - \lambda_j)/\lambda_j$	$p(\bar{\boldsymbol{u}}_j)$	$(p(\bar{\boldsymbol{u}}_j) - \lambda_j)/\lambda_j$
1	10.1601	10.1907	$3.0 \cdot 10^{-3}$	10.1708	$1.1 \cdot 10^{-3}$
2	28.6409	29.0485	$1.4 \cdot 10^{-2}$	28.6404	$-1.7 \cdot 10^{-5}$
3	56.3358	59.2327	$5.1 \cdot 10^{-2}$	56.3300	$-1.0 \cdot 10^{-4}$
4	79.4394	81.5605	$2.7 \cdot 10^{-2}$	79.4599	$2.6 \cdot 10^{-4}$
5	90.2518	97.0869	$7.6 \cdot 10^{-2}$	90.2851	$3.7 \cdot 10^{-4}$
6	165.7031	180.4788	$8.9 \cdot 10^{-2}$	165.9673	$1.6 \cdot 10^{-3}$
7	190.1096	219.1016	$1.5 \cdot 10^{-1}$	190.5553	$2.3 \cdot 10^{-3}$
8	198.1060	220.0651	$1.1 \cdot 10^{-1}$	196.9449	$-5.9 \cdot 10^{-3}$
9	256.1018	287.6977	$1.2 \cdot 10^{-1}$	253.9148	$-8.5 \cdot 10^{-3}$
10	309.8084	400.4072	$2.9 \cdot 10^{-1}$	373.7422	$2.1 \cdot 10^{-1}$

Table 4. Eigenvalues (8 substructures)

Nine eigenvalues $\bar{\lambda}_1, \cdots, \bar{\lambda}_9$ of the statically condensed problem are contained in the interval $(0\,,\,0.5 \cdot \underline{\omega})$ and the relative errors are reduced from up to 15% to less than 1% by the use of the Rayleigh functional p (cf. Table 4).

For the parallel solution of the matrix eigenvalue problem via condensation and improvement using the Rayleigh functional according to Table 1 we proposed in [16] the following proceeding. Nine processes are defined called 'Ma' and 'S_1',...,'S_8' which are connected by the tree topology shown in the lower part of Figure 3.

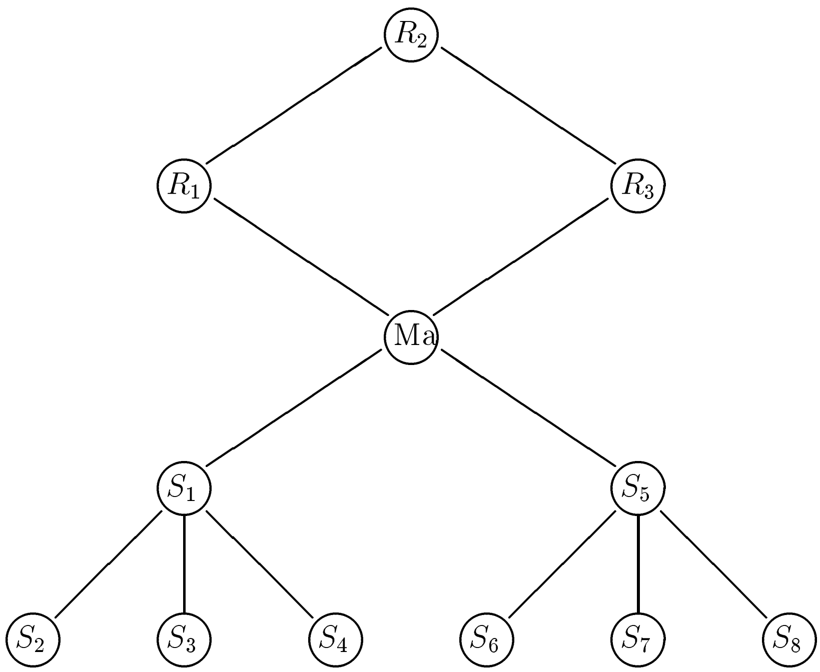

Figure 3. Connection topology (8 substructures)

At the end of step 1 $\boldsymbol{K}_0$ and $\boldsymbol{M}_0$ are updated using an intermediate summation of the submatrices from 'S_2', 'S_3', 'S_4' by 'S_1' and from 'S_6', 'S_7', 'S_8' by 'S_5', respectively, to avoid a bottleneck at 'Ma'.

The efficiency of the total program is influenced crucially by the different sizes of the eigenproblems in step 2 (cf. Figure 4). Since the dimension of the master problem is significantly larger than the dimensions of the slave eigenvalue problems corresponding to the substructures, the solution of the master eigenvalue problem dominates the computing time in this step although each of the processors 2, 3 and 4 run at least two slave processes.

The processors 2, 3 and 4 are idle for at least 85% of step 2. This drawback of the presented algorithm using a sequential solution of the reduced eigenproblem was already observed in [16].

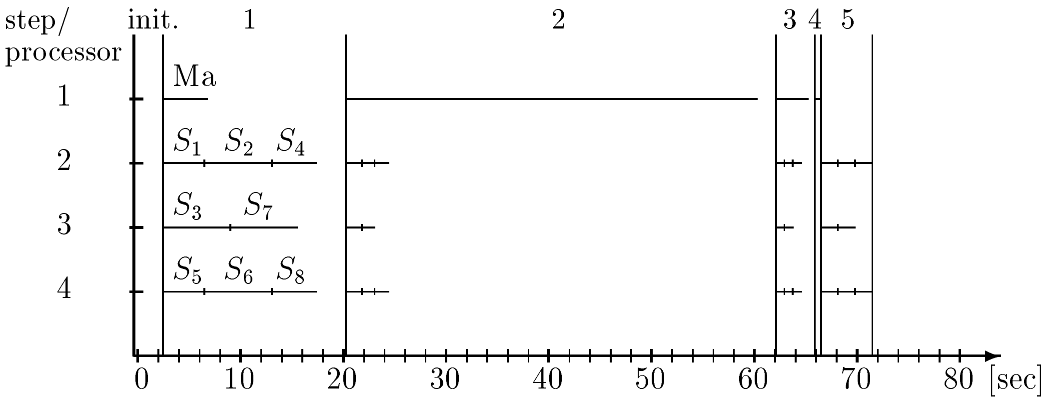

Figure 4. Loading diagram for 4 processors (without parallel Eigensolver)

To overcome this difficulty we incorporated the presented parallel eigensolver with the additional processes 'R_1', 'R_2', 'R_3'. The ring topology for this solver is appended to the tree as shown in Figure 3.

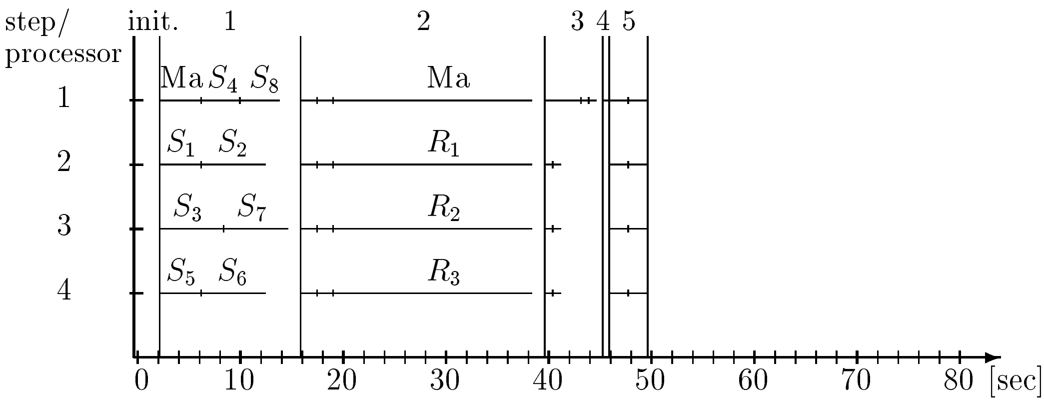

Figure 5. Loading diagram for 4 processors (with parallel eigensolver)

Figure 5 shows the best (fastest) mapping of the process topology of Figure 3 to four processors. We observe, that the inclusion of the processes 'R_1', 'R_2', 'R_3' yields a good load balancing.

To get the ten smallest eigenvalues and eigenvectors using the EISPACK routines corresponding to Table 2 for the uncondensed problem of dimension $n = 484$ sequentially one transputer needed 910.9 sec. To compute these eigenvalues with a sequential version of the improved condensation method took only 124.0 sec (13.6%). The presented parallel concept using EISPACK routines for the solution of the reduced problem and of the slave eigenvalue problems (cf. Figure 4) on four transputers reduced the effort to 71.6 sec (7.9%) and the incorporation of the parallel eigensolver for the condensed eigenvalue problem (cf. Figure 5) further reduced the computing time to 49.5 sec (5.4%).

References

[1] K. Berkkan and M.A. Dokainish, Accuracy of Condensed Eigenvalue Solution. AIAA J. 28 (1990) 1324 — 1325.

[2] N. Bouhaddi and R. Fillod, Substructuring using a linearized dynamic condensation method. Computers & Structures 45 (1992) 679 — 683.

[3] S.-H. Chen and H.H. Pan, Guyan reduction. Comm. Appl. Numer. Meth. 4 (1988) 549 — 556.

[4] J.J. Dongarra and R.A. van de Geijn, Reduction to condensed form for the eigenvalue problem on distributed memory architectures. Parallel Computing 18 (1992) 973 — 982.

[5] B.S. Garbow, J.M. Boyle, J.J Dongarra, and C.B. Moler, Matrix eigensystem routines – EISPACK Guide Extension. Lecture notes in computer science 51 (Springer, New York, 1977).

[6] G.H. Golub and C.F. Van Loan, Matrix Computations. (The John Hopkins University Press, Baltimore and London, 1989).

[7] R.J. Guyan, Reduction of stiffness and mass matrices. AIAA J. 3 (1965) 380.

[8] T. Hitziger, W. Mackens, and H. Voss, A condensation-projection method for generalized eigenvalue problems. To appear in C.A. Brebbia and H. Power (eds.): High Performance Computing in Engineering (Computational Mechanics Publications, Southampton).

[9] B. Irons, Structural eigenvalue problems: Elimination of unwanted variables. AIAA J. 3 (1965) 961 - 962.

[10] C.P. Johnson, R.R. Craig Jr., A. Yargicoglu, and R. Rabatabhothi, Quadratic reduction for the eigenproblem. Internat. J. Numer. Meth. Engrg. 15 (1980) 911 — 923.

[11] Y.T. Leung, An accurate method of dynamic condensation in structural analysis. Internat. J. Numer. Meth. Engrg. 12 (1978) 1705 - 1715.

[12] Y.T. Leung, An accurate method of dynamic substructuring with simplified computation. Internat. J. Numer. Meth. Engrg. 14 (1979) 1241 - 1256.

[13] K. Rothe, A parallel algorithm for the generalized symmetric eigenvalue problem on a transputer network, to appear in ZAMM

[14] K. Rothe, A parallel algorithm for the generalized symmetric eigenvalue problem on a transputer network, Technical Report, Inst. Appl. Math., Univ. Hamburg, 1994.

[15] K. Rothe and H. Voss, Improving condensation methods for eigenvalue problems via Rayleigh functional, Comp. Meth. Appl. Mech. Engrg. 111 (1994) 169 — 183.

[16] K. Rothe and H. Voss, Improved dynamic substructuring on distributed memory MIMD-computers, in: C.A. Brebbia and H. Power, eds., Applications of Supercomputers in Engineering III (Computational Mechanics Publications, Southampton, Boston and Elsevier Science Publishers, London, New York, 1993) 339 — 359.

[17] H. Voss, An error bound for eigenvalue analysis by nodal condensation, in J. Albrecht, L. Collatz and W. Velte, eds., Numerical Treatment of Eigenvalue Problems vol. 3, Internat. Series Numer. Math. 69 (Birkhäuser Verlag, Stuttgart, 1983) 205 — 214.

[18] M. Zehn, Dynamische FEM-Strukturanalyse mit Substrukturtechnik. Technische Mechanik 9 (1988) 245 — 253.