Structural Engineering and Mechanics, Vol. 5, No. 5 (1997) 635-644
DOI: http://dx.doi.org/10.12989/sem.1997.5.5.635

Parallel computation for transcendental structural eigenproblems

D. Kennedy† and F.W. Williams‡

Cardiff School of Engineering, University of Wales Cardiff,
P.O. Box 686, The Parade, Cardiff CF2 3TB, U.K.

Abstract. The paper reviews the implementation and evaluation of exact methods for the computation of transcendental structural eigenvalues, i.e., critical buckling loads and natural frequencies of undamped vibration, on multiple instruction, multiple data parallel computers with distributed memory. Coarse, medium and fine grain parallel methods are described with illustrative examples. The methods are compared and combined into hybrid methods whose performance can be predicted from that of the component methods individually. An indication is given of how performance indicators can be presented in a generic form rather than being specific to one particular parallel computer. Current extensions to permit parallel optimum design of structures are outlined.

Key words: parallel computers; structural analysis; buckling; vibration; optimum design.

1. Introduction

Critical buckling and undamped free vibration problems of structural analysis may be solved by the application of exact analytical solutions of the member stiffness equations. Solutions to the resulting transcendental eigenproblems are guaranteed by the Wittrick-Williams algorithm (Wittrick and Williams 1971) which guarantees convergence on all required critical buckling loads or natural frequencies. This exact method locates the eigenvalues individually using an iterative scheme in which, at successive trial values of the eigenparameter λ (i.e., the load factor or frequency), the real symmetric (or complex Hermitian) dynamic stiffness matrix $\boldsymbol{K}$ of the structure, whose elements vary transcendentally with the eigenparameter, is assembled and reduced to upper triangular form $\boldsymbol{K}^{\Delta}$ in order to calculate J, the number of eigenvalues exceeded by the eigenparameter. The solution time is generally dominated by the triangulation of $\boldsymbol{K}$ at each iteration, but may be reduced by several refinements, including exact multi-level substructuring (Williams 1973), efficient triangulation of $\boldsymbol{K}$ by a hybrid Gauss-Doolittle approach (Williams and Kennedy 1988a) and convergence on eigenvalues by qualified parabolic interpolation (Williams and Kennedy 1988b).

Such exact analysis of three-dimensional frames (Anderson and Williams 1987) and prismatic plate assemblies (Williams, *et al.* 1991) is often much faster (sometimes more than 1000 times faster) than analysis by the traditional finite element method. However, some applications, particu-

† Lecturer
‡ Professor, Head of Structural Division

larly those involving optimum design, still incur substantial solution times. The authors are therefore working intensively on the application of parallel computation to the solution of transcendental structural eigenproblems. This paper reviews their recent parallel methods for analysis (i.e., eigenvalue extraction), then outlines current extensions to permit parallel optimum design.

The methods have been developed for multiple instruction, multiple data (MIMD) parallel computers with distributed memory which synchronise tasks by the passing of messages among the processors. They have been evaluated on two such computers, a 32-processor nCUBE2 and a 48-processor Transtech Paramid, by measuring elapsed solution times t_n on different numbers of processors n, from which are derived values of speedup

$$S_n = \frac{t_1}{t_n}, \tag{1}$$

the ideal case of linear speedup being given by $S_n = n$. The methods are readily adaptable to shared memory MIMD computers where application software can allocate tasks to specific processors.

2. Review of parallel methods for eigenvalue extraction

Fig. 1 lists the principal computational tasks associated with eigenvalue extraction by the Wittrick-Williams algorithm. Parallel processing has been introduced in a number of ways, as described in the following subsections.

2.1. Coarse grain parallelism

A large number of natural frequencies of a structure may be found on a parallel computer by the simple expedient of allocating an (approximately) equal number of frequencies to each processor (Watkins, *et al.* 1996), i.e., by parallelising the outer loop of Fig. 1. Variability in convergence rates often leads to large variations in the times taken by each processor to find its frequencies, especially when the structure has coincident or close natural frequencies. In order to overcome this difficulty, the optional passing of a few short messages among the processors allows frequencies to be re-distributed from busy to idle processors and often enables all the processors to complete their work almost simultaneously. Table 1 lists computational requirements (Watkins,

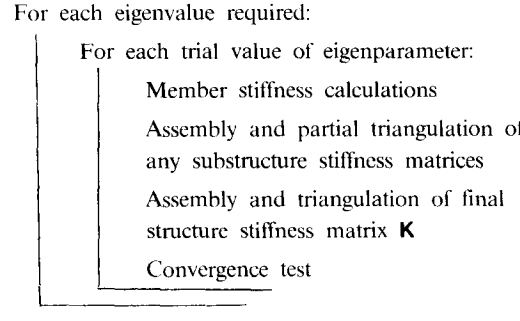

Fig. 1 Tasks involved in eigenvalue extraction.

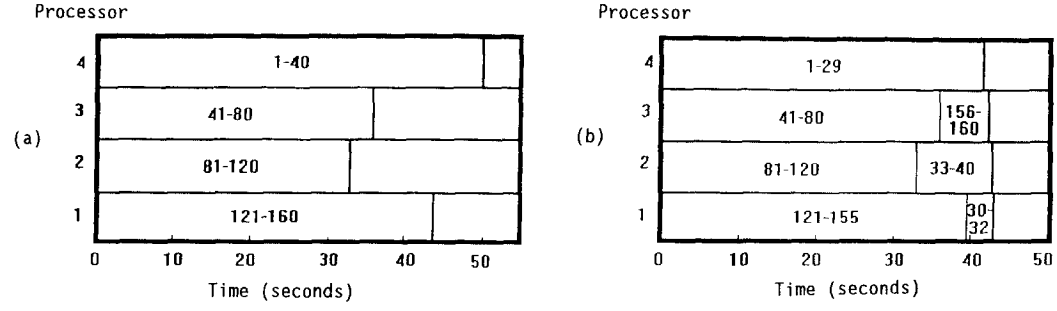

Fig. 2 Sequence of frequencies found by each processor for the solution of the problem of Table 1 using 4 processors, (a) without and (b) with re-distribution of frequencies to idle processors.

Table 1 Computational requirements for finding the first 160 natural frequencies of a 6 bay, 7 storey frame on an nCUBE2 parallel computer. The improved values given in italics were obtained by permitting re-distribution of eigenvalues to idle processors

	Number of processors					
	1	2	4	8	16	32
Solution time (seconds)	157.0	84.3	50.0	25.5	14.8	8.5
	157.0	*82.1*	*42.8*	*23.9*	*14.0*	*8.5*
Speedup	1.00	1.86	3.14	6.16	10.61	18.45
	1.00	*1.91*	*3.67*	*6.57*	*11.21*	*18.45*
Maximum idle time (seconds)	0.0	8.5	16.0	15.2	9.1	4.2
	0.0	*0.6*	*1.5*	*4.3*	*3.8*	*4.2*

et al. 1996) for computing, to an accuracy of 1 in 10^4, the first 160 natural frequencies of a 6 bay, 7 storey rectangular frame with 91 identical members, using different numbers of processors on an nCUBE2 parallel computer. The Table lists overall solution times (i.e., the elapsed time when the last processor completes its work), speedup and the maximum time that any processor was idle having completed its work, for parallel analysis both without and with the optional re-distribution of frequencies to idle processors. This re-distribution substantially reduced the time that processors lay idle and gave savings in solution time of up to 14.4% on 4 processors; Fig. 2 illustrates the sequence in which frequencies were found by each of the 4 processors for these cases.

The Wittrick-Williams method is particularly efficient if only the lowest buckling load is required over a number of loading or response cases, because for the second and subsequent cases calculation of J at the lowest buckling load previously found establishes whether the current case will yield a lower one. The number of buckling loads found explicitly therefore depends upon the order in which the cases are examined. Work recently completed (An, *et al.* 1997a) enables cases to be examined in parallel, messages being passed among the processors whenever the closest upper bound on the critical buckling load is reduced.

2.2. Medium grain parallelism

If only one eigenvalue is required, the coarse grain methods of the previous subsection cannot

be used. In the Wittrick-Williams method new trial values of the eigenparameter are normally selected by bisection or parabolic interpolation. Parallelisation of the inner loop of Fig. 1 is therefore expected to be less efficient than sequential methods in which each trial value is selected after J has been found for all previous trial values. However if $\boldsymbol{K}$ is approximated (Hopper, *et al.* 1980) by a linear matrix pencil between two trial values λ_a and λ_b ($\lambda_a<\lambda_b$) of the eigenparameter λ, so that

$$\boldsymbol{K}(\lambda)\cong\boldsymbol{A}-\lambda\boldsymbol{B} \tag{2}$$

where

$$\left.\begin{aligned}\boldsymbol{A}&=\frac{\lambda_b\boldsymbol{K}(\lambda_a)-\lambda_a\boldsymbol{K}(\lambda_b)}{\lambda_b-\lambda_a}\\ \boldsymbol{B}&=\frac{\boldsymbol{K}(\lambda_a)-\boldsymbol{K}(\lambda_b)}{\lambda_b-\lambda_a}\end{aligned}\right\} \tag{3}$$

and $\boldsymbol{K}(\lambda_a)$ and $\boldsymbol{K}(\lambda_b)$ are the values of $\boldsymbol{K}$ at λ_a and λ_b respectively, then the original transcendental eigenvalue problem

$$\boldsymbol{K}(\lambda)\ \boldsymbol{D}=\boldsymbol{0} \tag{4}$$

becomes the generalised linear eigenvalue problem

$$\boldsymbol{AD}=\lambda\boldsymbol{BD}. \tag{5}$$

The lowest eigenvalue of this linear eigenproblem is a lower bound on λ_1, the lowest eigenvalue of $\boldsymbol{K}$, if $\lambda_a<\lambda_1<\lambda_b$, and an upper bound on λ_1 otherwise, provided that λ_a and λ_b are both below the lowest pole of $\boldsymbol{K}$. Two processors have therefore been used to advantage (Chan, *et al.* 1997a) in simultaneously improving the lower and upper bounds on λ_1. This method gave a speedup of 1.67 for finding the lowest critical buckling load of a 33 bay, 100 storey rectangular plane frame to an accuracy of 1 in 10^8 on a Transtech Paramid parallel computer.

2.3. Fine grain parallelism

As an alternative to the coarse and medium grain methods already described, tasks within the inner loop of Fig. 1 may be performed in parallel. Although work in progress (An, *et al.* 1997b) will permit parallel assembly and partial triangulation of substructure stiffness matrices, investigations into such fine grain methods have concentrated on the triangulation of the final structure stiffness matrix $\boldsymbol{K}$.

$\boldsymbol{K}$ is normally triangulated by Gauss elimination without pivoting or scaling, or by a hybrid Gauss-Doolittle method (Williams and Kennedy 1988a) which performs the same computations but requires fewer memory accesses. The Gauss-Doolittle method has been parallelised (An, *et al.* 1997b) by partitioning $\boldsymbol{K}$ into blocks of v rows and assembling on the ith processor ($i=1, \ldots, n$) a matrix $\boldsymbol{K}_i$ having the same order as $\boldsymbol{K}$ and whose ith, $(n+i)$th, $(2n+i)$th,... blocks are those of $\boldsymbol{K}$ and whose remaining blocks are null, as illustrated in Fig. 3 for the case of $\boldsymbol{K}$ having order $8v$ and $n=4$. Each block is triangulated by the processor on which it was initially assembled. During triangulation, this processor calculates contributions to the next $(n-1)$ blocks and passes them in a message to the next processor. It then continues to calculate contributions to the remaining blocks, while the next processor commences triangulation of the next block.

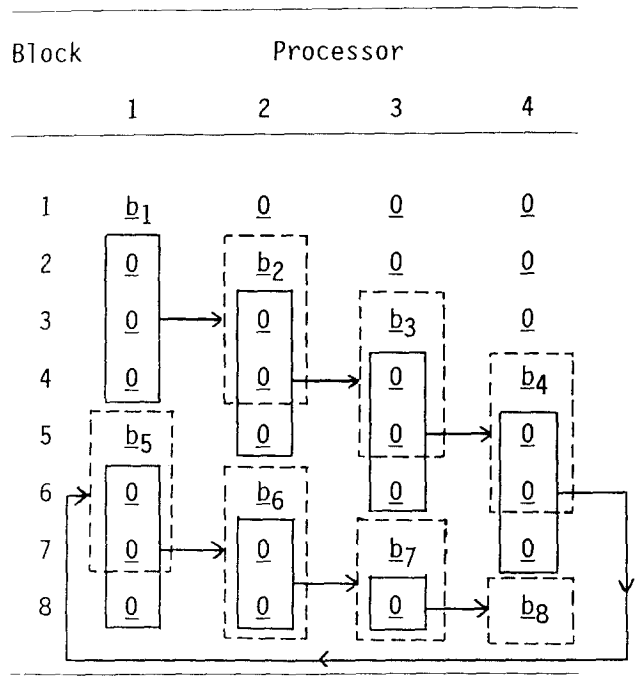

Fig. 3 Initial contents of $\boldsymbol{K}_i$ in processor i ($i=1, \ldots, n$) for parallel Gauss-Doolittle triangulation, with $\boldsymbol{K}$ having order 8ν and using $n=4$ processors. The blocks in the solid boxes are initially null, but are updated during triangulation and passed in messages to adjacent processors as indicated by arrows, where they are added to the contents of the dashed boxes.

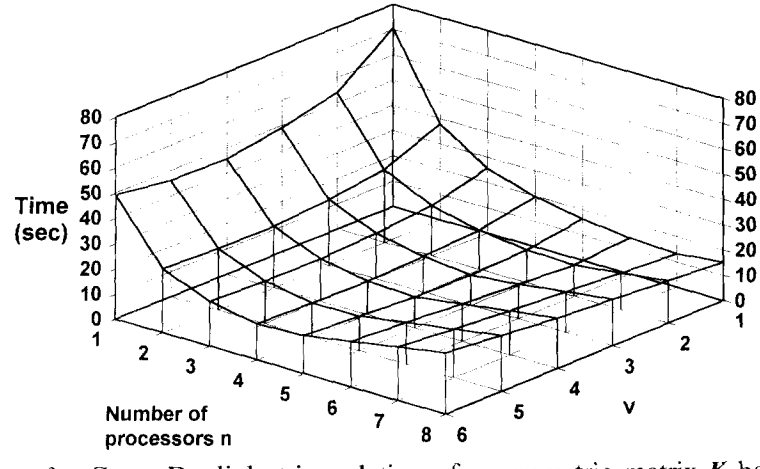

Fig. 4 Solution times for Gauss-Doolittle triangulation of a symmetric matrix $\boldsymbol{K}$ having order 840 and bandwidth 350 on an nCUBE2 parallel computer.

Fig. 4 shows solution times (Watkins, *et al.* 1997a) for the Gauss-Doolittle triangulation of a symmetric matrix $\boldsymbol{K}$ having order 840 and bandwidth 350, for $\nu=1, \ldots, 6$ and $n=1, \ldots, 8$ processors on an nCUBE2 parallel computer. For this problem, the highest possible speedup for $\nu=1$ was 5.36 on 7 processors; for $\nu>1$ this limit varied between 3.41 on 4 processors and 4.36 on 6 processors. In each case, if more processors were used the overall solution time increased

Table 2 Speedup for extraction of the first 16 natural frequencies of a 23 bay, 30 storey rectangular plane frame on an nCUBE2 parallel computer, using coarse grain and fine grain parallelism and a hybrid method with p clusters of q processors

		Number of processors: $n=pq$				
		1	2	4	8	16
Fine grain parallelism:	$q=1$	1.00	1.90	3.57	5.90	10.71
	$q=2$	–	1.94	3.72	6.88	11.34
	$q=4$	–	–	2.96	5.67	10.47

because processors routinely had to lie idle waiting for a message to arrive before proceeding to the triangulation of the next block of rows. This potentially serious loss of efficiency has been shown (Kennedy 1994) to depend crucially on the number of processors used, the bandwidth of $\boldsymbol{K}$, the block size v and the ratio of calculation speed to communication speed for the parallel computer being used.

2.4. Hybrid coarse and fine grain parallelism

Each of the parallel methods described above performs well on sufficiently large problems of a particular class, but suffers a loss of efficiency if too many processors are employed on smaller problems. Therefore optimum performance can be obtained by means of hybrid methods employing two or more forms of parallelism simultaneously. As an illustration (Kennedy, *et al.* 1995), the first 16 natural frequencies of a 23 bay, 30 storey rectangular frame were found on an nCUBE2 parallel computer, using $n=pq$ processors divided into p clusters of q processors, each cluster being allocated a set of eigenvalues to find by the coarse grain method and the processors within the cluster co-operating in fine grain parallel Gauss-Doolittle triangulation to find them. The results, listed in Table 2, show that, for each value of $n>2$, the best speedup is obtained with $p>1$ and $q>1$, i.e., using the hybrid method rather than simply the coarse grain method ($p=n$, $q=1$) or the fine grain method ($p=1$, $q=n$). It has also been shown (Kennedy, *et al.* 1995) that the speedup of the hybrid method can be predicted to good accuracy from those of the component methods by

$$S_{Hpq}=S_{Cp}S_{Fq} \tag{6}$$

where subscripts H, C and F refer to the hybrid, coarse grain and fine grain methods respectively.

2.5. Estimating machine dependency

The coarse and medium grain methods described above require relatively little message passing between processors. In contrast, the fine grain methods require much inter-processor communication and their performance depends crucially on the ratio of calculation speed to communication speed on the parallel computer being used. The solution time T_1 for a problem using any chosen number of processors on a particular parallel computer can be written (Watkins, *et al.* 1997b) as

$$T_1=T_0(1+\mu) \tag{7}$$

where T_0 is a measure of the time spent performing calculations and the problem-dependent

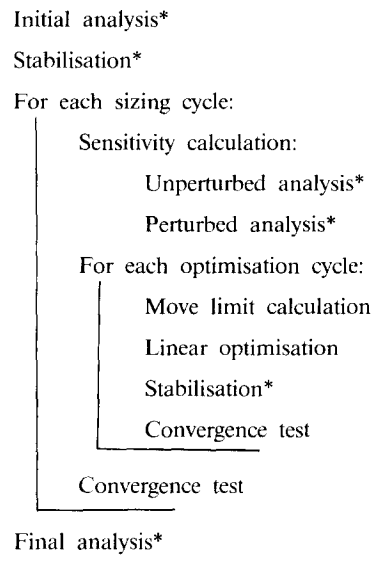

Fig. 5 Tasks involved in optimum design.
*denotes tasks which include some or all of the eigenvalue extraction tasks of Fig. 1.

parameter μ is a non-dimensional measure of the time spent performing other activities, including (primarily) sending and receiving messages. If the same problem is run, using the same number of processors, on a parallel computer whose calculation speed is ρ_c times slower and whose communication speed is ρ_m times slower, the solution time T_2 can be predicted by

$$T_2 = \rho_c T_0(1+\mu r) = \rho_m T_0(s+\mu) \tag{8}$$

where

$$r = \frac{\rho_m}{\rho_c}, \quad s = \frac{1}{r}. \tag{9}$$

Accurate predictions for T_2 can therefore be obtained (Watkins, *et al.* 1997b) on the first computer by artificially sending and receiving each message r times (for integers $r>1$), or by artificially performing the principal calculations s times (for integers s if $r<1$). Thus a parallel algorithm's performance indicators such as speedup can be presented in a generic form rather than being specific to the particular computer used to obtain them.

3. Parallel methods for optimum design

Fig. 5 outlines the sequential optimum (minimum mass) design procedure used in the authors' panel program VICONOPT (Williams, *et al.* 1991) and in a simple plane frame design program currently being used to develop and evaluate parallel design methods. The six tasks marked with an asterisk each include some or all of the Wittrick-Williams eigenvalue extraction analysis tasks of Fig. 1 and comprise the most computationally intensive parts of the procedure. Design moves are performed by a linear optimiser using sensitivity information calculated by a finite difference method. Key features of the procedure are the stabilisation steps following the initial

Table 3 Speedup for sensitivity calculations for the fundamental natural frequency of a 10 bay, 12 storey rectangular plane frame with 24 design variables, on a Transtech Paramid parallel computer

Number of processors	2	3	4	6	12	24
Speedup	1.92	2.78	3.54	4.94	8.02	11.49

analysis and each design move, which adjust the design to a 'just stable' configuration, and the re-use of sensitivity information to perform alternative optimisation cycles with different move limits starting from each intermediate design.

The initial and final analyses may be performed in parallel using any of the analysis methods of the previous section. The sensitivity calculations require calculation of the partial derivatives of the buckling, strength, stiffness and geometric constraints g_i with respect to the design variables x_j by in turn making a small perturbation of each x_j to x_j', calculating the resulting perturbed constraints g_{ij}' and using the approximation

$$\frac{\partial g_i}{\partial x_j} = \frac{(g_{ij}' - g_i)}{(x_j' - x_j)}. \tag{10}$$

The g_{ij}' rarely need to be found explicitly, since they can be estimated (Williams, *et al.* 1991) by triangulating $\boldsymbol{K}$ at just one trial value of the eigenparameter and using information obtained during convergence on g_i. In a very recent parallel implementation of these calculations, the unperturbed constraints g_i are found sequentially and then each processor perturbs its share of the design variables x_j to find the perturbed constraints g_{ij}'. Preliminary results, involving explicit calculation of the g_{ij}', are given in Table 3. A forthcoming publication (Chan, *et al.* 1997b) gives further results including the effects of re-distribution of the workload between processors when some of the g_{ij}' must be found explicitly while others are estimated.

The stabilisation steps of Fig. 5 involve, for each loading and response case, a single iteration check for stability, i.e., an evaluation of J at the required load level. If necessary, convergence on a just stable design is achieved by a procedure analogous to that of Fig. 1 which iterates on trial member sizes instead of on trial values of the eigenparameter and which it is proposed to parallelise in analogous ways. Advantage has also been taken (Chan, *et al.* 1997b) of performing entire optimisation cycles in parallel within each sizing cycle.

The authors are also currently working on sequential methods to avoid convergence on local optima. These methods include the use of alternative starting designs and new random search algorithms based on the Improving Hit-and-Run Method (Zabinsky, *et al.* 1993) which seek distant lower mass designs following convergence on an optimum. It is planned subsequently to exploit the natural parallelism of such approaches.

A principal application is the detailed minimum mass design of aerospace wing panels. This is commonly performed as part of a much larger, computationally intensive, multi-level, multi-disciplinary process requiring the repeated simultaneous design of all the panels on a wing and for which parallel processing offers enormous potential benefits.

4. Summary of conclusions

A number of coarse grain, medium grain and fine grain methods for the parallel extraction

of transcendental structural eigenvalues have been presented, with illustrative results. The methods can be combined into hybrid methods employing several kinds of parallelism simultaneously, whose performance can be predicted from those of the component methods individually. Additionally, performance indicators can be presented in a generic form rather than being specific to one particular computer. Current extensions to the parallel optimum design of structures have been outlined.

Acknowledgements

The authors gratefully acknowledge financial support from the Engineering and Physical Sciences Research Council (grant GR/H45490), the Higher Education Funding Council for Wales and the Research Committee of the University of Wales, Cardiff, and the use of computing facilities provided by the Departments of Physics and Computer Science.

References

An, W., Watkins, W.J., Kennedy, D. and Williams, F.W. (1997a), "Parallel critical buckling calculations for prismatic plate assembly" submitted for publication.

An, W., Watkins, W.J., Kennedy, D. and Williams, F.W. (1997b), "Parallel substructuring in exact eigenvalue calculations for prismatic plate assembly", in preparation.

Anderson, M.S. and Williams, F.W. (1987), "BUNVIS-RG: exact frame buckling and vibration program, with repetitive geometry and substructuring", *J. Spacecr. Rockets*, **24**(4), 353-361.

Chan, K.L., Kennedy, D. and Williams, F.W. (1997a), "Efficient parallel bounding solution method for structural transcendental eigenvalue problems", submitted for publication.

Chan, K.L., Kennedy, D. and Williams, F.W. (1997b), "Parallel design methods for optimising lightweight structures", to be presented at the Mouchel Centenary Conference on Innovation in Civil and Structural Engineering, Cambridge, U.K.

Hopper, C.T., Simpson, A. and Williams, F.W. (1980), "A study of the bounds on eigenvalues of a transcendental stiffness matrix provided by a simply derived linear matrix pencil", *J. Struct. Mech.*, **8**(4), 365-422.

Kennedy, D. (1994), "Efficient computation of transcendental structural eigenvalues using exact member stiffnesses", PhD Thesis, University of Wales, U.K.

Kennedy, D., Watkins, W.J. and Williams, F.W. (1995), "Hybrid parallel computation of transcendental structural eigenvalues", *AIAA J.*, **33**(11), 2194-2198.

Watkins, W.J., Kennedy, D. and Williams, F.W. (1996), "Efficient parallel solution of structural eigenvalue problems", *Adv. Eng. Software*, **25**(2/3), 281-289.

Watkins, W.J., Kennedy, D. and Williams, F.W. (1997a), "Efficient parallel Gauss-Doolittle matrix triangulation", *Comput. Struct.*, **62**(1), 185-195.

Watkins, W.J., Kennedy, D. and Williams, F.W. (1997b), "On estimating machine dependency of fine- and coarse-grained parallel structural computations", *Microcomp. in Civil Eng.*, **12**(2), 119-128.

Williams, F.W. (1973), "Comparison between sparse matrix and substructure methods", *Int. J. Num. Methods Eng.*, **5**(3), 383-394.

Williams, F.W. and Kennedy, D. (1988a), "Fast Gauss-Doolittle matrix triangulation", *Comput. Struct.*, **28**(2), 143-148.

Williams, F.W. and Kennedy, D. (1988b), "Reliable use of determinants to solve non-linear structural eigenvalue problems efficiently", *Int. J. Num. Methods Eng.*, **26**(8), 1825-1841.

Williams, F.W., Kennedy, D., Butler, R. and Anderson, M.S. (1991), "VICONOPT: program for exact vibration and buckling analysis or design of prismatic plate assemblies", *AIAA J.*, **29**(11), 1927-1928.

Wittrick, W.H. and Williams, F.W. (1971), "A general algorithm for computing natural frequencies of elastic structures", *Q. J. Mech. Appl. Math.*, **24**(3), 263-284.
Zabinsky, Z.B., Smith, R.L., McDonald, J.F., Romeijn, H.E. and Kaufman, D.E. (1993), "Improving hit-and-run for global optimization", *J. Global Opt.*, **3**(2), 171-192.