A fast parallel solver for the biharmonic problem

Michael K. Bane[*] *and Milan D. Mihajlović*[†]

Abstract. We examine the convergence characteristics of a preconditioned Krylov subspace solver applied to discrete systems arising from low-order mixed finite element approximation of the classical biharmonic problem. We describe a preconditioning operator which leads to preconditioned systems having an eigenvalue distribution consisting of a tightly clustered set together with a small number of outliers. We present numerical results that show that the performance of our methodology is competitive with other fast iteration methods proposed in this context. The parallel implementation is done on an SGI Origin 2000 architecture, using the HSL 2000 software library.

Key words. Biharmonic equation, mixed methods, finite elements, preconditioning, multifrontal methods.

1 Introduction

In this paper we are studying the performance of the new preconditioning methodology for discrete systems arising from the mixed approximation of the biharmonic problem. The mixed form of the biharmonic problem on some convex polygonal domain $\Omega \in \mathbb{R}^2$ with a sufficiently smooth boundary $\partial\Omega$, and load function $f \in H^{-1}(\Omega)$ can be represented as

$$\left.\begin{array}{rcl} -\Delta\phi & = & \omega \\ -\Delta\omega & = & f \end{array}\right\} \quad \text{in } \Omega, \qquad \phi = \frac{\partial\phi}{\partial n} = 0 \quad \text{on } \partial\Omega. \tag{1}$$

The equation (1) is a model in fluid mechanics for a steady laminar flow, where ϕ and ω represent the stream function and vorticity, respectively. It is also a

[*]Centre for Novel Computing, Department of Computer Science, The University of Manchester, Manchester M13 9PL, England. `bane@cs.man.ac.uk`. The author acknowledges the EPSRC financial support under grant GR/M70155.

[†]Centre for Novel Computing, Department of Computer Science, The University of Manchester, Manchester M13 9PL, England. `milan@cs.man.ac.uk`. The author acknowledges the financial support of The University of Manchester.

model for plate bending (see [22]), in which case ϕ is the deflection and ω is the bending moment. An interesting feature of the solution ϕ of the problem (1) (or of the corresponding eigenproblem, where $f = \lambda u$) near sufficiently acute corners of the domain Ω is a sequence of so-called Moffat eddies [15]. Namely, the solution exhibits a self-similar exponentially decaying oscillatory pattern (which corresponds, in fluid mechanics terms, to a sequence of eddies, whose intensity decays following the exponential pattern). This phenomenon was studied theoretically in [7], [8], and [13], and computationally in [2] and [5].

In the standard mixed approximation of (1) we seek a solution pair $(\omega, \phi) \in H^1(\Omega) \times H_0^1(\Omega)$ which satisfies

$$\begin{array}{rcll} (\omega, \upsilon) - (\nabla\upsilon, \nabla\phi) &=& 0 & \forall \upsilon \in H^1(\Omega) \\ -(\nabla\omega, \nabla\varphi) &=& -(f, \varphi) & \forall \varphi \in H_0^1(\Omega). \end{array} \tag{2}$$

In (2), $(\cdot, \cdot)$ denotes the standard inner product in $L^2(\Omega)$ (vector or scalar). We can get a discrete linear system from (2) by taking finite dimensional subspaces $X_h \subset H^1(\Omega)$ and $M_h = X_h \cap H_0^1(\Omega)$ (with h being a representative mesh parameter), and enforcing (2) over the subspaces X_h and M_h. This reduces to computing $(\omega_h, \phi_h) \in X_h \times M_h$ which satisfies

$$\begin{array}{rcll} (\omega_h, \upsilon_h) - (\nabla\upsilon_h, \nabla\phi_h) &=& 0 & \forall \upsilon_h \in X_h \\ -(\nabla\omega_h, \nabla\varphi_h) &=& -(f, \varphi_h) & \forall \varphi_h \in M_h. \end{array} \tag{3}$$

The stability issues and well-posedness of (2) and (3) are discussed in [1] and [4].

The discrete problem (3) can be expressed as a system of linear algebraic equations by introducing operators $\mathcal{M} : X_h \mapsto X_h$ and $\mathcal{K} : X_h \mapsto M_h$ defined by

$$\begin{array}{rcll} (\mathcal{M}\omega_h, \upsilon_h) &=& (\omega_h, \upsilon_h) & \forall \omega_h, \upsilon_h \in X_h, \\ (\mathcal{K}\omega_h, \varphi_h) &=& (\omega_h, \mathcal{K}^T\varphi_h) = -(\nabla\omega_h, \nabla\varphi_h) & \forall \omega_h \in X_h, \forall \varphi_h \in M_h. \end{array} \tag{4}$$

In (4) $\mathcal{K}^T$ represents an adjoint operator of $\mathcal{K}$. Substituting the definitions (4) into (3), we get a matrix system

$$\begin{pmatrix} \mathcal{M} & \mathcal{K}^T \\ \mathcal{K} & 0 \end{pmatrix} \begin{pmatrix} \omega_h \\ \phi_h \end{pmatrix} = \begin{pmatrix} 0 \\ f_h \end{pmatrix}. \tag{5}$$

Note that in (5) f_h represents the $L^2(\Omega)$ orthogonal projection of $-f$ into M_h.

In order to design efficient multigrid preconditioners and solvers for the system (5), several authors use the Schur complement operator $\mathcal{S} = \mathcal{K}\mathcal{M}^{-1}\mathcal{K}^T$ (see, for example [3], [10], [18]). It is known that $\mathcal{S}$ has a condition number that behaves like $O(h^{-4})$ as the mesh is refined, and previously mentioned algorithms therefore require careful tuning of parameters (including the number of pre and post smoothing steps) in order to achieve performance which is independent of h.

To derive our preconditioning strategy, which involves "black-box" algorithms for the solution of Dirichlet Poisson problems, we shall first express (5) in terms of finite element matrices. We introduce the finite element basis

$$X_h = \operatorname{span}\{\psi_i\}_{i=1}^{n_I+n_B}, \qquad M_h = \operatorname{span}\{\psi_j\}_{j=1}^{n_I}. \tag{6}$$

In (6) we assume n_I interior and n_B boundary degrees of freedom, respectively. Furthermore, we define the vectors $\omega \in \mathbb{R}^{n_I+n_B}$, $\phi \in \mathbb{R}^{n_I}$, and $f \in \mathbb{R}^{n_I}$ that correspond to the finite-dimensional functions ω_h, ϕ_h, and f_h. If we define a "mass" matrix $M_{ij} = (\psi_i, \psi_j)$, $i,j = 1,\ldots,n_I+n_B$ and the "constraint" matrix $K_{ij} = -(\nabla\psi_i, \nabla\psi_j)$, $i = 1,\ldots,n_I$, $j = 1,\ldots,n_I+n_B$, the finite element formulation of (5) is

$$\begin{pmatrix} M & K^t \\ K & 0 \end{pmatrix} \begin{pmatrix} \omega \\ \phi \end{pmatrix} = \begin{pmatrix} 0 \\ f \end{pmatrix}. \tag{7}$$

In order to explore the use of Poisson solvers as a tool for an efficient preconditioning of (7), we decompose ω_h into the sum of interior and boundary contributions. Assuming we enumerate the interior nodes first from 1 to n_I, we obtain

$$\underbrace{\sum_{i=1}^{n_I+n_B} \omega_i \psi_i}_{\omega_h \in X_h} = \underbrace{\sum_{j=1}^{n_I} v_j \psi_j}_{v_h \in M_h} + \underbrace{\sum_{k=1}^{n_B} \lambda_k \psi_{n_I+k}}_{\lambda_h \in T_h}, \tag{8}$$

where $T_h \approx L^2(\partial\Omega)$. This decomposition leads to a matrix partitioning of (7). In particular, $K = (K_I \;\; K_B)$, where K_I denotes the finite element stiffness matrix of dimension $n_I \times n_I$ which arises in discretization of a Dirichlet Laplacian problem using M_h.

The remainder of the paper is organised as follows. In Section 2 we present a novel approach that is based on an exact indefinite constraint preconditioner. Inexact versions of this methodology have been suggested in the context of mixed finite element formulations of magnetostatics problems (see, for example, [19]), but have not been considered in the context of the biharmonic problem until now. Section 3 gives some "black-box" implementation details of the exact version of a preconditioner using parallel routines for sparse matrix computations from the HSL 2000 library [11], and Krylov iterative solvers from NAG library [17]. We report our numerical results in terms of iteration counts of the preconditioned BICGSTAB(2) method and the execution time on the SGI Origin 2000 architecture. The comparison of our methodology with some other solvers used for the biharmonic problem is given in [20].

2 An indefinite constraint preconditioner

In this section we outline the main idea of an indefinite constraint preconditioner that we developed in [20] in the context of the biharmonic problem. The general idea of the constraint preconditioning approach is described in [12]. This strategy has been used in the context of mixed approximations [19]. In our strategy we tailor the preconditioner to the special structure of the coefficient matrix in (7).

Let us consider again a coefficient matrix in (7), together with the splitting (8)

$$\mathcal{A} = \begin{pmatrix} M & K^t \\ K & 0 \end{pmatrix} = \begin{pmatrix} M_I & M_c^t & K_I \\ M_c & M_B & K_B^t \\ K_I & K_B & 0 \end{pmatrix} \tag{9}$$

Table 1. *Extremal eigenvalues of the eigenproblem* (11) *computed on a set of uniformly refined triangular grids using piecewise linear approximation.*

mesh	h	n	$\mu_{\min}$	$\mu_{\max}$	$\kappa(\mathcal{P}^{-1}\mathcal{A})$
12×12	0.083333	290	0.685100749	14.854800875	21.68
24×24	0.041667	1154	0.693016600	28.670967418	41.37
48×48	0.020833	4610	0.695583923	56.292502654	80.93

and the following version of the preconditioner

$$\mathcal{P} = \begin{pmatrix} 0 & 0 & K_I \\ 0 & M_B & K_B^t \\ K_I & K_B & 0 \end{pmatrix} \equiv \begin{pmatrix} G & K^t \\ K & 0 \end{pmatrix}. \tag{10}$$

The preconditioner $\mathcal{P}$ in (10) is symmetric and indefinite. The action of its inverse can be achieved by a simple backsubstitution (note, however, that the block K_I has to be previously factorized or approximated in a suitable manner). It can be proved that the matrix $\mathcal{P}$ is non-singular (see [20]). Therefore, the eigenvalues $\{\mu_i\}_{i=1}^{2n_I+n_B}$ of the preconditioned operator

$$\begin{pmatrix} M_I & M_c^t & K_I \\ M_c & M_B & K_B^t \\ K_I & K_B & 0 \end{pmatrix} \begin{pmatrix} v \\ \lambda \\ \phi \end{pmatrix} = \mu \begin{pmatrix} 0 & 0 & K_I \\ 0 & M_B & K_B^t \\ K_I & K_B & 0 \end{pmatrix} \begin{pmatrix} v \\ \lambda \\ \phi \end{pmatrix} \tag{11}$$

are real and non-zero. The eigenvectors that correspond to the multiple eigenvalue $\mu = 1$ are analysed in [12]. In particular, if the preconditioner (10) is invertible, then the preconditioned operator (11) has an eigenvalue $\mu = 1$ with multiplicity $\geq 2n_I$. The analysis that covers the case $\mu \neq 1$ is covered in [20]. We outline here only that the preconditioned matrix operator (11) has a multiple eigenvalue at unity, together with a small cluster of eigenvalues ($\leq n_B$) that lie in the interval $(1, Ch^{-1}]$ (see also Fig. 2.1 in [20]). This leads us to believe that a Krylov subspace method applied to the preconditioned system should converge very rapidly. In particular, for a given distribution of the eigenvalues in [12] it has been demonstrated that GMRES method will converge in at most n_B+2 iterations (providing that exact arithmetic is used). In our experiments we adopt BICGSTAB(2) [21], which showed robustness in application. We also make one additional simplification in the preconditioner $\mathcal{P}$: we approximate the "mass" boundary block M_B by its diagonal D_{M_B}. This changes the lower bound of a spectrum $\{\mu_i\}$ from 1 to $1/\Theta^2$ (see [20, Proposition 2.2]), where Θ is independent of h and $\Theta > 1$. To illustrate the previous points, we present in Table 1 the extremal eigenvalues of the preconditioned operator (11) for the case of piecewise linear approximation, computed on a set of uniformly refined triangular grids. In the next section we examine the performance characteristics of the preconditioned BICGSTAB(2) method and consider one particular parallel implementation of the preconditioner given by (10).

3 Implementation details and numerical results

We now assess the performance of the exact version of the preconditioner (10) when applied in conjuction with the BICGSTAB(2) method to the solution of the discrete system (7). At the end of this section we shall briefly outline some alternatives where the action of the discrete operator K_I within the preconditioner $\mathcal{P}$ is replaced by some suitable approximation K_I^*, thus yielding an inexact version of the constraint preconditioner. We report our results in terms of the iteration counts of the preconditioned Krylov subspace solver, and the execution time needed for computing the Cholesky factorization of the block K_I and the associated backsubstitution phase (since these are the most time consuming steps in the BICGSTAB(2) algorithm). We constrain ourselves to one particular implementation, using the HSL 2000 subroutine library [11]. We perform all our experiments on an Origin 2000 architecture.

Firstly, we discuss some implementation-related details. As a representative of Krylov subspace iterative solver that can be applied in the case when both the coefficient matrix and the preconditioner are indefinite we take the BICGSTAB(2) method. In our application BICGSTAB(2) proved to be a reliable choice: it has never broken down or failed to converge. One goal in this work is to develop an implementation of the iterative solver and preconditioner $\mathcal{P}$ using as much publicly available code as possible. Following this goal, we use the BICGSTAB(2) algorithm supplied as a part of the routine F11BEF from the NAG library Mark 19 (see [17]). Since this routine is sequential, none of the operations that represent the core of BICGSTAB(2) method can be parallelized, except computing of the preconditioner matrix and its inverse. These operations are made accessable to the user by extracting them from the library code. Having in mind that the asymptotic complexity of these operations is significantly larger than those that belong to the remainder of the BICGSTAB(2) algorithm (including the sparse matrix-vector multiply), we conclude that we need to parallelize only this external part of the algorithm to achieve high performance. To illustrate this, we shall briefly study the asymptotic behaviour of the various parts in the BICGSTAB(2) algorithm. We have already mentioned in Section 2 that the action of $\mathcal{P}^{-1}$ can be achieved by backsubstitution. However, in the exact version, the discrete Dirichlet Laplacian block K_I must be exactly factorized. This is the most time consuming part of the BICGSTAB(2) algorithm, and should be done only once at the start. In our experience [6], this operation has an asymptotic complexity $O(n^f)$, where $f \approx 5/3$ using the SuperLU package [9]. In addition, one forward and backsubstitution operation using previously obtained factors takes $O(n^b)$, where $b \approx 5/4$ ([6]). The remaining operations in the BICGSTAB(2) algorithm are either BLAS 1, or sparse matrix-vector multiplications (where the matrix $\mathcal{A}$ in (9) has $O(1)$ elements per row), and therefore have complexity $O(n)$.

There is a host of parallel library codes for performing the Cholesky factorization of a SPD matrix (see [14]). We chose the HSL 2000 library subroutine HSL_MP62 [11] to factorize the block K_I in parallel. This routine is designed for factorization of an SPD matrix which originates from a finite element discretization. The routine uses a multifrontal approach – parallel frontal factorization on subdomains and the sequential factorization of an interface problem (the variables that are shared between the subdomains). MPI [16] is used as a communicator. We performed an extensive testing of the routine for various types of Lagrangian elements (linear, quadratics, cubics) and different choices of h. However, in our experiments we constrained ourselves to the unit square domain $\Omega = [0,1]^2$, discretized by the uniform mesh (congruent right-angled triangles). Nonetheless, the method is fully applicable to any domain and any type of the triangulation.

The architectural and support specification of the system are as follows. The results are obtained by running the code on a 16 processor SGI Origin 2000 using the "cpuset" facility to give exclusive access to processors and their local memory. The MIPSpro 7.3.1.1m Fortran 90 compiler is used in 64 bit mode with optimization flags `-O3 -OPT:Olimit=0`. The latter flag is required to ensure all routines are optimized irrespective of source size. The object files are linked with SGI's message passing toolkit mpt 1.4.0.0 for MPI 1.2, Mark 19 of the NAG numerical Fortran library [17] and version 1.3.0.0 of SGI's Scientific Software Library (for the machine tuned BLAS).

Here we report a small representative of the performance results that we have obtained. Table 2 contains two particular cases of equally partitioned domain Ω onto 4 and 9 subdomains, respectively. In Table 2 we use the following notation; `mesh` denotes the number of grid cells in x and y direction, and n is the total dimension of a discrete system ($n = 2n_I + n_B$). N_S is the number of equal subdomains to which the domain Ω is subdivided, and N_P is the number of processes (note that for good load balance we consider only the cases $N_P \mid N_S$). T_f is the factorization time, and T_b is the backsubstitution time (see [11]). The last component T_b determines the time of one BICGSTAB(2) iteration T_B (and we have $T_B = 2T_b + o(T_b)$). We also report the efficiency of the factorization phase E_f^p when executed on more than one processor ($E_f^p = (T_f)_p/(T_f)_1$). All the times given represent wall clock times in seconds. We do not report the time spent in the analysis phase, which is of the same order as T_b on a single processor.

From Table 2 we can conclude that the asymptotic behaviour of the HSL_MP62 routine in our experiments is similar to that of the SuperLU in the factorization phase, and slightly worse in the backsubstitution phase. The code exibits a good load balance and efficiency when executed on multiple processors. The code performance and efficiency is improving as the large-scale problems are solved. Therefore, in our opinion it is a valuable alternative which can be employed in this context.

We further report the convergence results of the preconditioned BICGSTAB(2) algorithm with the preconditioner $\mathcal{P}$ that is implemented with the Poisson solver HSL_MP62. The right hand side vector f in (7) is chosen randomly, although the elements of f behave like $O(h^2)$, which gives a realistic situation of a randomly distributed load. The stopping criterion for BICGSTAB(2) is given by

$$\|r_i\|_\infty \leq \varepsilon\,(\|b\|_\infty + \|\mathcal{A}\|_\infty \|x_i\|_\infty), \tag{12}$$

as provided by the routine F11BEF. In (12), r_i represents the residual vector, b is the right hand side vector, $\mathcal{A}$ the coefficient matrix, and x_i the current approximation of the solution. In our experiments we adopt 3 different levels of accuracy, governed by the values of ε: $\varepsilon = 10^{-6}$, 10^{-9}, and 10^{-12}. By comparing our results with that obtained by a direct solver, we expect at least 3, 6, and 9 correct digits in the

Table 2. *The execution time of the different phases of the parallel Poisson solver* HSL_MP62 *for various types of finite element approximation.*

a) linear elements

mesh		72 × 72			144 × 144			288 × 288		
n		10370			41474			165890		
N_S	N_P	T_f	T_b	E_f^p	T_f	T_b	E_f^p	T_f	T_b	E_f^p
4	1	0.77	0.02	1.00	10.00	0.19	1.00	143.04	1.30	1.00
	2	0.40	0.01	0.96	5.08	0.12	0.98	72.05	0.84	0.99
	4	0.72	0.01	0.88	2.61	0.07	0.96	36.58	0.46	0.98
9	1	0.49	0.03	1.00	5.57	0.19	1.00	75.00	1.25	1.00
	3	0.20	0.01	0.82	2.02	0.09	0.92	26.07	0.60	0.96
	9	0.11	0.01	0.49	0.94	0.06	0.66	11.23	0.34	0.74

a) quadratic elements

mesh		36 × 36			72 × 72			144 × 144		
n		10370			41474			165890		
N_S	N_P	T_f	T_b	E_f^p	T_f	T_b	E_f^p	T_f	T_b	E_f^p
4	1	0.26	0.02	1.00	2.47	0.19	1.00	29.48	1.38	1.00
	2	0.15	0.01	0.87	1.32	0.12	0.94	15.26	0.91	0.97
	4	0.09	0.01	0.72	0.73	0.07	0.85	7.90	0.46	0.93
9	1	0.24	0.03	1.00	2.05	0.18	1.00	23.23	1.26	1.00
	3	0.12	0.01	0.67	0.85	0.09	0.80	8.78	0.56	0.88
	9	0.08	0.01	0.33	0.55	0.05	0.41	5.45	0.34	0.47

a) cubic elements

mesh		24 × 24			48 × 48			96 × 96		
n		10370			41474			165890		
N_S	N_P	T_f	T_b	E_f^p	T_f	T_b	E_f^p	T_f	T_b	E_f^p
4	1	0.21	0.02	1.00	1.71	0.17	1.00	18.23	1.09	1.00
	2	0.13	0.01	0.81	0.93	0.10	0.92	9.57	0.70	0.95
	4	0.08	0.01	0.66	0.54	0.06	0.79	5.12	0.39	0.89
9	1	0.20	0.02	1.00	1.53	0.17	1.00	15.91	1.04	1.00
	3	0.12	0.01	0.56	0.75	0.08	0.68	7.00	0.48	0.76
	9	0.08	0.01	0.28	0.48	0.05	0.35	4.64	0.31	0.38

iterative solution, respectively. Note that for the majority of practical applications the level of accuracy $\varepsilon = 10^{-6}$ is sufficient. From Table 3 we observe that the number of iteration steps for $\varepsilon = 10^{-6}$ is independent both of h and the type of elements. For the tighter tolerances $\varepsilon = 10^{-9}$ and $\varepsilon = 10^{-12}$ the iteration counts grow slowly as h is reduced, but again are not significantly dependent upon the type of approximation. The total execution time is, as expected, governed by $O(n^f)$. The memory requirements are quite moderate (bearing in mind that we employ the

Table 3. *Iteration count of the preconditioned* BICGSTAB(2) *method with the exact preconditioner* $\mathcal{P}$.

a) linear elements

mesh	72×72	144×144	288×288
n	10370	41474	165890
$\varepsilon = 10^{-6}$	5	5	5
$\varepsilon = 10^{-9}$	13	16	21
$\varepsilon = 10^{-12}$	26	31	39

a) quadratic elements

mesh	36×36	72×72	144×144
n	10370	41474	165890
$\varepsilon = 10^{-6}$	5	5	5
$\varepsilon = 10^{-9}$	11	15	19
$\varepsilon = 10^{-12}$	22	29	38

a) cubic elements

mesh	24×24	48×48	96×96
n	10370	41474	165890
$\varepsilon = 10^{-6}$	5	5	5
$\varepsilon = 10^{-9}$	11	16	20
$\varepsilon = 10^{-12}$	21	30	38

direct method) and have never exceeded 200Mb (for the largest example that we have considered $n = 165890$).

Finally, we briefly point to some ways of substituting the exact version of the preconditioner $\mathcal{P}$ by an approximation $\mathcal{P}^*$. One of the most attractive ways is to replace the action of the discrete operator K_I by a multigrid approximation K_I^*, where, typically, a fixed number of V-cycles is performed. This strategy is studied in more detail in [20]. For an efficient parallel implementation of this approach one would need multigrid or BPX solvers based on a domain decomposition methodology.

Acknowledgement. The authors would like to acknowledge the use of HSL 2000 and NAG numerical software libraries as a part of their work. We are grateful to Dr David J. Silvester (UMIST, Manchester, UK) for useful discussions related to this work. We are also indebted to Prof. Iain S. Duff, and especially Dr Jennifer Scott (Rutherford Appleton Laboratory, Didcot, UK) for their patience, help and support in our mastering the use of HSL libraries.

Bibliography

[1] I. Babuška, J. Osborn, and J. Pitkäranta, *Analysis of mixed methods using mesh dependent norms*, Math. Comput., 35 (1980), pp. 1039–1062.

[2] P. E. Bjøstrad and B. P. Tjøstheim, *A note on high precision solutions of two fourth order eigenvalue problems*, Computing, 63(2) (1999), pp. 97–107.

[3] D. Braess and P. Peisker, *On the numerical solution of the biharmonic equation and the role of squaring matrices for preconditioning*, IMA J. Numer. Anal., 6 (1986), pp. 393–404.

[4] F. Brezzi and M. Fortin, *Mixed and Hybrid Finite Element Methods,* Springer–Verlag, New York, 1991.

[5] B. M. Brown, E. B. Davies, P. K. Jimack, and M. D. Mihajlović, *A numerical investigation of the solution of a class of fourth-order eigenvalue problems*, Proc. Roy. Soc. (London), A(456) (2000), pp. 1505–1521.

[6] B. M. Brown, P. K. Jimack, and M. D. Mihajlović, *A new direct solver for a class of mixed finite element problems*, Appl. Numer. Math., to appear.

[7] H. Blum and R. Rannacher, *On the boundary value problem of the biharmonic operator on domains with angular corners*, Math. Meth. in Appl. Sci., 2 (1980), pp. 556–581.

[8] C. V. Coffman, *On the structure of solutions to $\Delta^2 u = \lambda u$ which satisfy the clamped plate boundary conditions on a right angle*, SIAM J. Math. Anal., 13(5) (1982), pp. 746–757.

[9] J. W. Demmel, J. R. Gilbert, and X. S. Li, *SuperLU User's Guide,* Technical Report UCB//CSD 97-944, Computer Science Division, University of California, Berkeley, CA, USA, November 1997.

[10] M. R. Hanisch, *Multigrid preconditioning for the biharmonic Dirichlet problem*, SIAM J. Numer. Anal., 30 (1993), 184–214.

[11] *HSL 2000. A collection of Fortran codes for large scale scientific computation,* `http://www.numerical.rl.ac.uk`

[12] C. Keller, N. J. M. Gould, and A. J. Wathen, *Constraint preconditioning for indefinite linear systems*, SIAM J. Matrix Anal. Appl., 2000, to appear.

[13] V. A. Kozlov, V. A. Kondratev, and V. G. Maz'ya, *On sign variation and the absence of "strong" zeros of solutions of elliptic equations,* Mat. USSR Izv., 34 (1990), pp. 337–353.

[14] X.S. Li, *Direct solvers for sparse matrices*, preprint.

[15] K. Moffat, *Viscous and resistive eddies near a sharp corner*, J. Fluid Mech., 18 (1964), pp. 1–18.

[16] MPI v1.2, `http://www.mpi-forum.org/docs/mpi-20-html/mpi2-report.html`

[17] Numerical Algorithms Group, *NAG Manual, Fortran Library Mark 19*, 1999.

[18] P. Peisker, *A multilevel algorithm for the biharmonic problem*, Numer. Math., 46 (1985), pp. 623–634.

[19] I. Perugia and V. Simoncini, *Block diagonal and indefinite symmetric preconditioners for mixed finite element formulations*, Numer. Lin. Alg. Appl., 2000, to appear.

[20] D. J. Silvester and M. D. Mihajlović, *Efficient preconditioning of the biharmonic equation*, SIAM J. Sci. Comp., submitted.

[21] G. L. G. Sleijpen and D. R. Fokkema, *BiCGSTAB(ℓ) for linear equations involving unsymmetric matrices with complex spectrum*, Electron. Trans. Numer. Anal., 1 (1993), Sept., pp. 11–32.

[22] S. Timoshenko and S. Woinowsky-Krieger, *Theory of plates and shells*, McGraw–Hill, New York, 1959.