IFISS: A Computational Laboratory for Investigating Incompressible Flow Problems[*]

Howard C. Elman[†]
Alison Ramage[‡]
David J. Silvester[§]

Abstract. The *Incompressible Flow & Iterative Solver Software* (IFISS) package contains software which can be run with MATLAB or Octave to create a computational laboratory for the interactive numerical study of incompressible flow problems. It includes algorithms for discretization by mixed finite element methods and a posteriori error estimation of the computed solutions, together with state-of-the-art preconditioned iterative solvers for the resulting discrete linear equation systems. In this paper we give a flavor of the code's main features and illustrate its applicability using several case studies. We aim to show that IFISS can be a valuable tool in both teaching and research.

Key words. finite elements, incompressible flow, iterative solvers, mathematical software

AMS subject classifications. 97N80, 76D99, 65M60, 65N22

DOI. 10.1137/120891393

1. Introduction. Research in computational mathematics is often motivated by the results of numerical experiments. Frequently, a result is conjectured based on behavior observed using a software environment such as MATLAB long before it is supported by any formal analytic results. In this sense, the "computational laboratory" plays just as important a role in modern mathematics as physical laboratories do in physics, chemistry, biology, and engineering. A similar observation applies in the context of teaching computational mathematics: carrying out investigative numerical experiments on particular topics helps students learn how to formulate hypotheses, design simple experiments to test them, and interpret the resulting data. As well as developing important deduction and interpretation skills, this hands-on approach is often more useful in helping students remember critical ideas over a significant period of time than a traditional textbook-only method.

With this in mind, the *Incompressible Flow & Iterative Solver Software* (IFISS) toolbox [36] has been developed as a computational laboratory for the interactive

[†]Department of Computer Science, University of Maryland, College Park, MD 20742 (elman@cs.umd.edu). The work of this author was supported in part by the U.S. National Science Foundation under grant DMS1115317 and the U.S. Department of Energy under grant DE-SC0009301.

[‡]Department of Mathematics and Statistics, University of Strathclyde, Glasgow G1 1XH, UK (A.Ramage@strath.ac.uk).

[§]School of Mathematics, The University of Manchester, Manchester M13 9PL, UK (d.silvester@manchester.ac.uk).

numerical study of incompressible flow problems.[1] The software package includes algorithms for discretization by mixed finite element methods and a posteriori error estimation of the computed solutions, together with state-of-the-art preconditioned iterative solvers for the discrete linear systems that arise. The current release is IFISS 3.2. Version 2.0 was released in conjunction with the monograph *Finite Elements and Fast Iterative Solvers* [13] and was used to perform the computational experiments described therein. An overview of this earlier version of IFISS can be found in [12]. The aim of this review article is therefore not to give a technical description of the software, but, instead, to give a flavor of its usefulness in driving our research and in enhancing our teaching. In what follows we focus on four specific examples of how we have used IFISS in this way, both as a teaching and research tool and as a starting point for developing code for more specialized research applications.

2. Overview. The IFISS toolbox is structured as a standalone package for studying discretization algorithms for partial differential equations (PDEs) and for exploring and developing algorithms in numerical linear and nonlinear algebra for solving the associated discrete systems. There are currently twenty eight built-in test problems, all based on common PDEs. Seventeen of these are elliptic PDE problems and are associated with the Poisson equation or steady-state versions of the convection-diffusion equation, the Stokes equations, or the Navier–Stokes equations. The remaining test problems are parabolic and involve the heat equation, advection-diffusion equations, or the Navier–Stokes equations modeling unsteady incompressible flow.

The first main feature of the package concerns problem specification and finite element discretization. For each PDE, IFISS offers a choice of two-dimensional domains on which the problem can be posed, along with boundary conditions and other aspects of the problem, and a choice of finite element discretizations on uniform or stretched quadrilateral element meshes. The package therefore makes it easy to study the accuracy of finite element solutions, different choices of elements, and a posteriori error estimates (see section 3.1). In addition, special features associated with individual problems can be explored. These include the effects of boundary layers on solution quality for the convection-diffusion equation, and the effects of discrete inf-sup stability and adaptive time stepping on accuracy for the Stokes and Navier–Stokes equations.

Another important aspect of IFISS concerns iterative solution of the discrete algebraic systems, with emphasis on preconditioned Krylov subspace methods. The key to fast solution lies in the choice of effective preconditioning strategies. The IFISS package offers a range of options, including algebraic methods such as incomplete LU factorizations and algebraic multigrid, as well as more sophisticated state-of-the-art methods designed to take advantage of the specific structure of discrete flow equations. In addition, there is a choice of iterative strategies, Picard iteration or Newton's method, for solving the nonlinear algebraic systems arising from Navier–Stokes problems. Again, IFISS offers scope for investigating issues like the choice of finite element approximation and the influence of the time step on the convergence of iterative solvers in a straightforward way (see section 3.2).

There are of course many other finite element packages that could be used to investigate any of the above topics. As well as commercial codes (for example, ABAQUS, COMSOL, and LS–DYNA), there are a number of excellent free software packages: in

---

[1]The software package is freely available (see http://www.manchester.ac.uk/ifiss or http://www.cs.umd.edu/∼elman/ifiss) and can be run using MATLAB or Octave. The source may be redistributed and/or modified under the terms of the GNU Lesser General Public License.

particular, ALBERTA [32], DUNE [6], and FEniCS [23]. A distinctive feature of IFISS is that it runs within a MATLAB environment, which means that the programming level is relatively high, so the source code is readable, portable, and easily adaptable. In this sense, MATLAB is an ideal computational laboratory. Our finite element implementation is also very efficient. Throughout IFISS all local calculations (for example, quadrature in generating element matrices, application of essential boundary conditions, and a posteriori error estimation) are vectorized over the elements—leading to efficient computation on the current generation of multicore processor architectures. We speculate that the majority of the IFISS code runs as fast as if it were coded in C or FORTRAN.

The design of IFISS not only facilitates the study of alternative discretizations and iterative solution algorithms, it also makes it easy to examine the interaction between the two and hence determine the overall solution cost. This feature is illustrated later using the example of the development of an optimal iterative solver for symmetric indefinite systems (see section 3.3). As well as being a useful source of benchmark problems, the package provides a convenient starting point for research projects that require the construction of new problem classes, discretizations, or solution algorithms. A detailed description of how new problems can be incorporated into IFISS is given in the *Installation and Software Guide* [35]. An example of this type of spin-off development is described in section 3.4.

3. Illustrative Examples. In this section we describe four situations where IFISS has been used as a teaching or research tool. The descriptions given here are, by necessity, somewhat abbreviated. Readers who would like to learn more about the specific details should consult the cited references.

3.1. Investigating Finite Element Convergence. When teaching finite element analysis, it is common to establish a priori and a posteriori error bounds which describe the convergence of the finite element approximation as the underlying grid is refined (see, for example, [3, 13, 20, 41] for an introduction to finite element approximation and [4] for a detailed derivation of classical error bounds). Having done this in class, it is easy to reinforce the practical importance of these ideas using numerical experiments within IFISS. (The following example uses diffusion test problems 3 and 4. Figures 3.1 and 3.3 can be generated using the IFISS function `sirev_ex3_1`.)
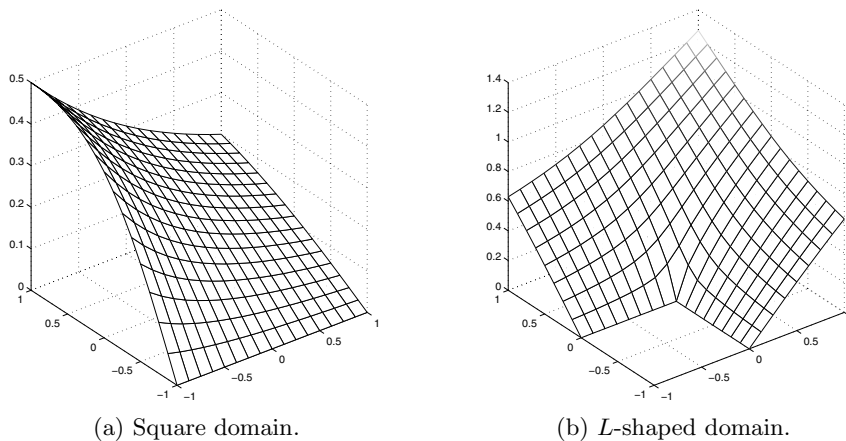

(a) Square domain. (b) *L*-shaped domain.

Fig. 3.1 *Sample finite element approximations of analytic solutions.*

Let us consider the problem of solving Laplace's equation on the domain $[-1, 1] \times [-1, 1]$ for a smooth solution whose analytic form is known (see [13, Example 1.1.3]). A three-dimensional surface plot of a finite element solution computed on a 16×16 square grid is shown in Figure 3.1(a). The IFISS toolbox automatically provides *a posteriori energy estimates* of the approximation error for both bilinear Q_1 and biquadratic Q_2 approximation. These energy estimates are plotted in Figure 3.2(a) for a sequence of square grids (Q_1: solid line, Q_2: dashed line). It can be readily seen that the Q_1 error is halved with every successive grid refinement, whereas the Q_2 error is reduced by a factor of four. This behavior is consistent with classical energy estimates; see, e.g., (1.80) and (1.81) in [13].[2]

Next, let us repeat the experiment for a more interesting test problem with a singular solution. This time we cut out the region $[-1, 0] \times [-1, 0]$ and solve Laplace's equation on the resulting L-shaped domain. The problem now has a singularity at the reentrant corner (see Figure 3.1(b)). The errors for this problem are plotted in Figure 3.2(b).
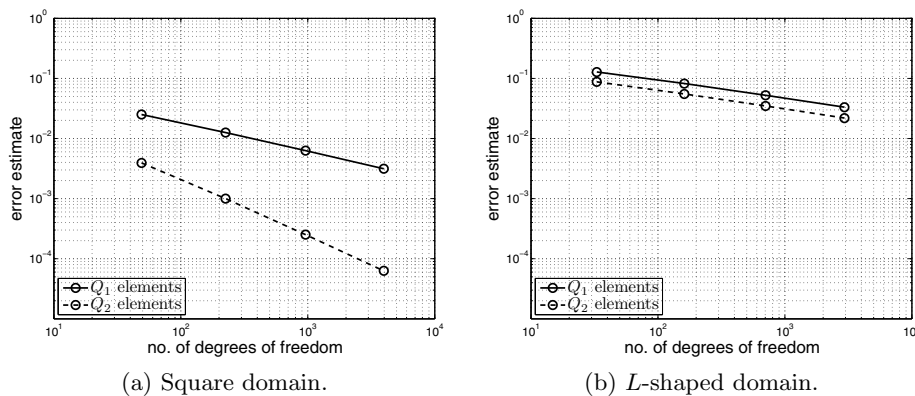

(a) Square domain.	(b) L-shaped domain.

Fig. 3.2 *Behavior of the a posteriori error estimates.*

This time, using higher-order elements does not improve the asymptotic convergence rate, and both discretization methods have *exactly the same* rate of convergence. To emphasize the importance of the smoothness in the solution, we re-solve the second problem using a stretched grid, whose successive element edges are a factor of 1.5 times longer then the adjacent edge; see Figure 3.3(a). The surface plot of the estimated error that IFISS provides, shown in Figure 3.3(b), clearly shows the difficulty encountered at the interior corner. By comparing the estimated errors, it can be seen that for Q_2 approximation, the overall accuracy obtained with 161 degrees of freedom is comparable with that obtained using a uniform grid with 2945 degrees of freedom. Of course, the challenge here is to determine the optimal stretching a priori!

3.2. Comparing Iterative Solvers for Parabolic PDE Problems. The IFISS toolbox is an excellent tool for studying unsteady flow problems. To show this, consider solving a Navier–Stokes flow problem with a view to exploring the effect of the size of the time step on the convergence of the linear solvers used at each time level. Details

---

[2]The a posteriori estimates are remarkably accurate—the effectivity index (the ratio of the estimated and the exact energy errors) is $0 \cdot 9992$ for the 16×16 grid shown in Figure 3.1(a).

(a) $\boldsymbol{Q}_1$ finite element solution.   (b) $\boldsymbol{Q}_1$ estimated error.
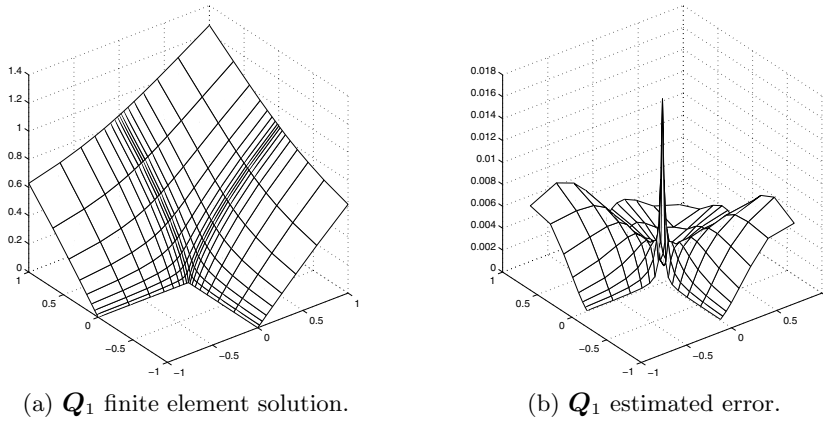
Fig. 3.3 *L-shaped domain with an optimally-stretched rectangular grid.*

of the mixed finite element methodology required for such situations can be found in [5, 13, 14, 18]. As a representative problem, we take the example of spinning up from a quiescent state to a steady lid-driven cavity flow in a square-shaped domain. The viscosity parameter ν is set to $1/200$ and the flow Reynolds number is 400. The spatial discretization is a mixed finite element method with a stable $\boldsymbol{Q}_2$–$\boldsymbol{Q}_1$ mixed approximation of the velocity and pressure, respectively. (See [13, Example 7.1.3] for details of the steady-state version of this problem). Snapshots of the computed flow (generated by the IFISS function `sirev_ex3_2`) are shown in Figure 3.4.
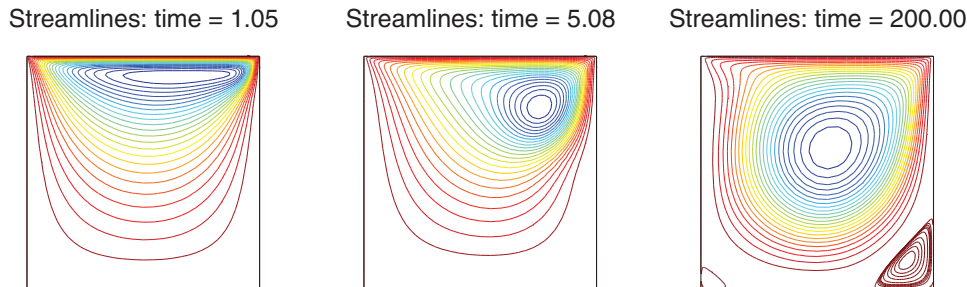

Streamlines: time = 1.05   Streamlines: time = 5.08   Streamlines: time = 200.00

Fig. 3.4 *Representative snapshots of the lid-driven cavity flow.*

The self-adaptive time-stepping algorithm used in IFISS involves an implicit time integrator using a stabilized trapezoidal rule with an explicit Adams–Bashforth method for error control (see [21] for details). The evolution of the time step with the accuracy error tolerance ε_t set to 10^{-4} is shown in Figure 3.5(a). At each individual time step, a linear algebra system with a distinctive saddle-point structure (see [13, section 7.3]),

$$\left(\begin{array}{cc} F & B^T \\ B & 0 \end{array} \right) \left(\begin{array}{c} \mathbf{u} \\ \mathbf{p} \end{array} \right) = \left(\begin{array}{c} \mathbf{f} \\ \mathbf{g} \end{array} \right),$$

must be solved. We present results here using a preconditioned GMRES linear solver [31], comparing the performance of three alternative preconditioners (additional meth-

ods are available in IFISS). The choices are

- ILU Incomplete LU factorization with zero fill-in (using the MATLAB function `ilu` with the 'nofill' option); see, e.g., [29, Chapter 10].
- PCD Pressure convection-diffusion preconditioning, described in [13, 22, 34].
- LSC Least-squares commutator preconditioning, described in [9, 10, 13].

The first of these is a standard "black-box" preconditioner, while the latter two methods are specifically designed for saddle-point systems coming from discretization of the linearized Navier–Stokes equations. Both of these involve the use of an upper block-triangular preconditioner,

$$\mathcal{M} = \begin{pmatrix} F & B^T \\ 0 & -M_S \end{pmatrix},$$

where M_S is an approximation to the Schur complement operator $S = BF^{-1}B^T$. The two alternatives differ in the way in which this approximation is made.
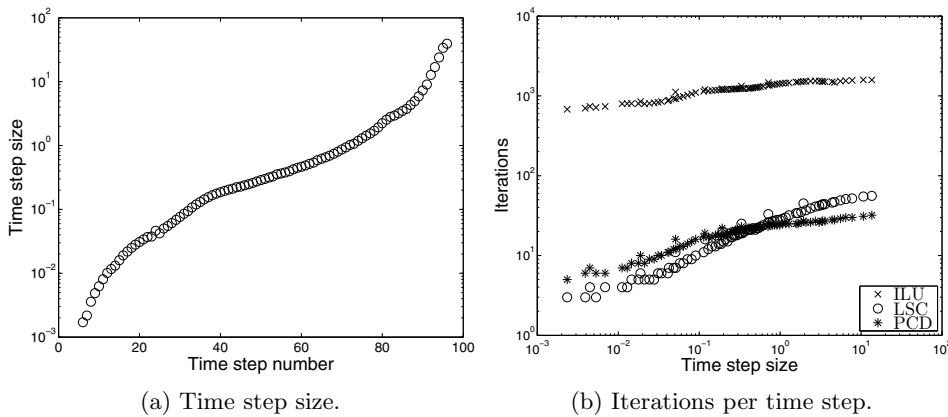

(a) Time step size.   (b) Iterations per time step.

Fig. 3.5 *Time-stepping data for the representative lid-driven cavity flow problem.*

Figure 3.5(b) shows the number of preconditioned GMRES iterations taken for the linear solve at each time step for the lid-driven cavity test problem with 8450 degrees of freedom (these individual systems are generated using the IFISS function `snapshot_solveflow`). The horizontal axis shows the time step size, and the vertical axis shows the number of GMRES iterations required for convergence with a tolerance of 10^{-6} at that time step. From this plot it is evident that the performance of all the preconditioned linear solvers tested deteriorates as the time step size increases— this is a known issue with unsteady simulations. The interesting thing, however, is the degree of deterioration, which is clearly different for each preconditioner. Such considerations can be important in the design of an efficient adaptive solver, suggesting the possibility of automatically switching the type of preconditioner used as the time step size changes.

3.3. Linking Discretizations and Solvers. One of the more unusual features that IFISS offers is easy access to the interplay between finite element discretizations and iterative solvers. Details of how this can be achieved are given in the *Installation and Software Guide* [35]. As an example of how the package has been used effectively in this way, we now briefly summarize the work of Silvester and Simoncini [38] on the development of an optimal solver for symmetric linear systems associated with mixed

approximation of PDEs. Specifically, we consider this work in the context of solving a discrete Stokes problem, namely, the slow-flow version of the regularized lid-driven cavity problem described in the previous section (see, for example, [5, 13, 14, 18] for background material on Stokes problems).

The new optimal solver described in [38] has three key ingredients. First, it uses a block preconditioner which gives convergence at a rate independent of the problem parameters. Second, it uses an adaptation of the usual MINRES algorithm of Paige and Saunders [26], which takes advantage of certain aspects of the finite element discretization in the design of the iterative solution method. The resulting EST_MINRES algorithm (see [38, Fig. 10]), instead of minimizing in the standard Euclidean norm of the residual vector, minimizes in a computable monotonically decreasing norm that is equivalent to the natural norm for error estimation of the discrete solution. The final new feature is the incorporation of a posteriori error estimation functionality so that a precise stopping criterion can be identified which balances the algebraic error with the PDE approximation error. The combination of these ideas clearly shows why considering the PDE origin of a linear system is essential in the design of efficient algebraic solvers.

To fix notation, we write the linear system arising from mixed finite element approximation of our test Stokes cavity flow problem as

$$(3.1) \qquad \begin{bmatrix} A & B^T \\ B & 0 \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \mathbf{f} \\ \mathbf{g} \end{bmatrix} \qquad \text{or} \qquad K\mathbf{x} = \mathbf{b},$$

with residual and algebraic error at the kth step given by $\mathbf{r}^{(k)} = \mathbf{b} - K\mathbf{x}^{(k)}$ and $\mathbf{e}^{(k)} = \mathbf{x} - \mathbf{x}^{(k)}$, respectively (where $\mathbf{x}$ represents the exact solution to (3.1)). Now suppose system (3.1) is preconditioned with a symmetric and positive definite preconditioner, say M. That is, conceptually we think of solving $MK\mathbf{x} = M\mathbf{b}$ instead, choosing M so that MINRES convergence is faster than for (3.1). One natural choice for a Stokes problem is the *canonical* preconditioner

$$(3.2) \qquad M = \begin{bmatrix} A^{-1} & 0 \\ 0 & Q^{-1} \end{bmatrix}$$

(where Q represents the pressure mass matrix) that maps the dual finite element space into the original (see [24, 28] for details). Note that the quantity minimized at each step by standard preconditioned MINRES iteration, $\|\mathbf{r}^{(k)}\|_M$, is measured in a *preconditioner-dependent* norm. In terms of the underlying PDE, as the discretization error for the Stokes problem is measured in the L_2 norm for the gradient of the velocity and in the L_2 norm for pressure (see [13, section 5.4]), the natural matrix norm here is $\|\mathbf{e}^{(k)}\|_E$, where

$$E = \begin{bmatrix} A & 0 \\ 0 & Q \end{bmatrix}.$$

Ideally, we would like to relate the quantity computed by MINRES to this natural norm which arises naturally from the PDE. For the Stokes problem, we assume that the finite element discretization used is *inf-sup stable* (see, e.g., [5, equation (2.29)]). With this assumption, we can use the heuristic

$$(3.3) \qquad \|\mathbf{e}^{(k)}\|_E \leq \frac{\sqrt{2}}{\gamma^2} \|\mathbf{r}^{(k)}\|_M,$$

where γ is the so-called *inf-sup constant* (see [38]).

In theory, inequality (3.3) could be used as the basis for a stopping criterion for MINRES iteration (by stopping when the quantity on the right-hand side of (3.3) is less than some fixed tolerance). However, there are still two issues to be addressed before this can be considered useful in practice. The first of these is that, for realistic problems, an "ideal" preconditioner of the form (3.2) would usually have to be replaced by a (cheaper) spectrally equivalent preconditioner, say M_*: the example given in [38] replaces the action of applying A^{-1} by a fixed number of algebraic multigrid (AMG) V-cycles and the action of applying Q^{-1} by a fixed number of Jacobi iterations with Chebyshev acceleration. The authors show there that if the approximation of M by M_* is sufficiently accurate, then the impact on convergence of replacing the M-norm in (3.3) by the M_*-norm will be minimal.

The second practical issue with (3.3) is the need to explicitly compute the inf-sup constant γ. It turns out, however, that an accurate estimate of γ^2 can be automatically generated at every step of EST_MINRES by exploiting the connection between the iteration and the Lanczos estimates of the eigenvalues of the preconditioned matrix. In addition, the algorithm gives a bonus in the form of a free estimate of the accuracy of M_* as an approximation to M: see [38, section 4.1] for details.

The final step in producing an optimal parameter-free iterative method is to connect the actual stopping tolerance with the PDE discretization error. With established a posteriori error estimation technology, an estimate for the PDE discretization error at each step can be computed and the iteration terminated when the appropriate measure of the absolute residual is less than this value. Denoting such an error estimate at EST_MINRES step k by $\eta^{(k)}$, the final practical stopping criterion suggested is

$$(3.4) \qquad\qquad \frac{\sqrt{2}}{\gamma^2}\|\mathbf{r}^{(k)}\|_{M_*} \leq \eta^{(k)},$$

which, in light of (3.3), forces the algebraic error to be comparable with the estimation of the discretization error.

Figure 3.6 shows some convergence plots for EST_MINRES applied to the regularized lid-driven cavity test problem with $\boldsymbol{Q}_2$–$\boldsymbol{P}_1$ approximation on a 64×64 stretched grid of square elements. In each case, the evolution of three quantities is shown: the upper bound estimate for $\|\mathbf{e}^{(k)}\|_E$ from (3.3) (dashed line), the approximation error
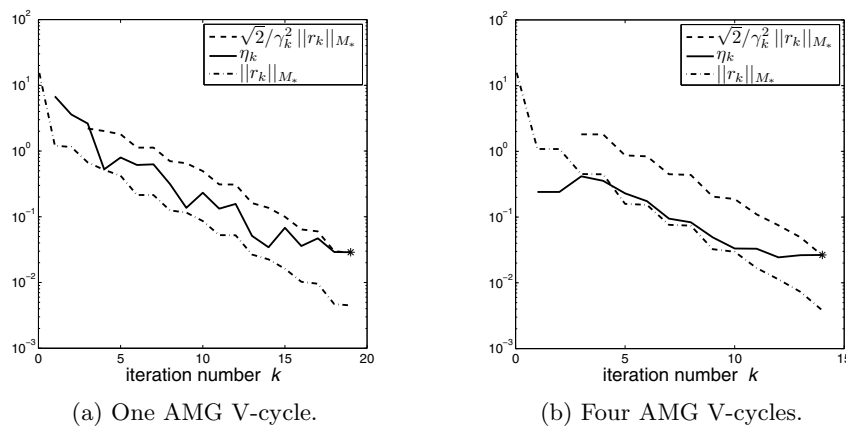

(a) One AMG V-cycle.   (b) Four AMG V-cycles.

Fig. 3.6 *Convergence curves.*

estimate $\eta^{(k)}$ (solid line), and the preconditioned residual error (dash-dot line). The (automatically computed) termination point associated with the stopping test (3.4) is marked with $*$. It can be seen from Figure 3.6(a) that with one AMG V-cycle in the preconditioner, the iteration converges in 19 iterations, whereas with four AMG V-cycles (see Figure 3.6(b)), only 14 iterations are required. Other aspects of the performance of this optimal solver can be seen using the IFISS routine `stokesflowdemo`.

3.4. Applications and Extensions. One immediate application of IFISS is as a convenient source of sample matrices for testing new ideas. This is not just limited to the development of algorithms specifically for problems in incompressible fluid flow [1, 2]. IFISS has also been used as a source of test problems in a much more general way, for example, in assessing the performance of new Krylov subspace solvers and preconditioners for a wider class of saddle-point problems [42, 45, 46], and for specific applications in optimization [7], signal processing [15], and eigenvalue computations [19, 27, 43]. Details of how the code can be used in this way are given in the *Installation and Software Guide* [35]. Since its creation in 1997 (see [8]) the IFISS package has evolved to include many additional features such as new equation models, discretization methods, preconditioning methods, and solver algorithms. Because IFISS has been deliberately designed in a modular way, it is relatively easy for the user to incorporate new developments and additional code, or to extract parts of IFISS to be used in combination with other software. Examples where IFISS has been used successfully in this way include the enhanced package PIFISS [37] for solving potential flow problems, stochastic finite element software used to study uncertainty quantification [33, 44], and a coupled flow-orientation solver in liquid crystal modeling [39].

In this section we highlight one particular application [11] where IFISS has been used to explore a fundamental question in mathematical modeling concerning the stability of dynamical systems. When they arise from discretization of parameter-dependent PDEs, these systems have the form

$$(3.5) \qquad\qquad Mu_t = f(u, \nu),$$

where $f : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}^n$ is a nonlinear mapping, $u = u^{(\nu)}(t) \in \mathbb{R}^n$ is a discrete state variable that depends on a scalar parameter ν, and (for finite elements) M is a mass matrix. The question of interest is: given a steady solution $u^{(\nu)}$, i.e., $u_t^{(\nu)} = 0$, and a small perturbation $\delta(0)$ at time $t = 0$, does $\delta(t)$ decay with time, in which case the steady solution is *stable*, or does it grow? A necessary condition for stability in the study of *linear* stability analysis is that all eigenvalues of the generalized eigenvalue problem

$$(3.6) \qquad\qquad J\mathbf{x} = \mu M \mathbf{x}$$

have negative real parts, where $J = \frac{\partial f}{\partial u}(u, \nu)$ is the Jacobian matrix of f evaluated at the specific parameter value ν.

In cases of *Hopf bifurcation* [17], the eigenvalues of interest consist of a complex conjugate pair that, as ν varies from a stable regime to a critical value ν^* at which stability is lost, traces out paths moving from left to right in the complex plane until they cross the imaginary axis. The identification of ν^* is a difficult computational task because of complexities associated with the eigenvalue problem (3.6). For large-scale discretizations, J will be too large to use direct eigenvalue methods such as the QR algorithm [16, 40]. The alternative choice, iterative methods such as Arnoldi-based methods or subspace iteration [30, 40], can compute only a few selected eigenvalues.
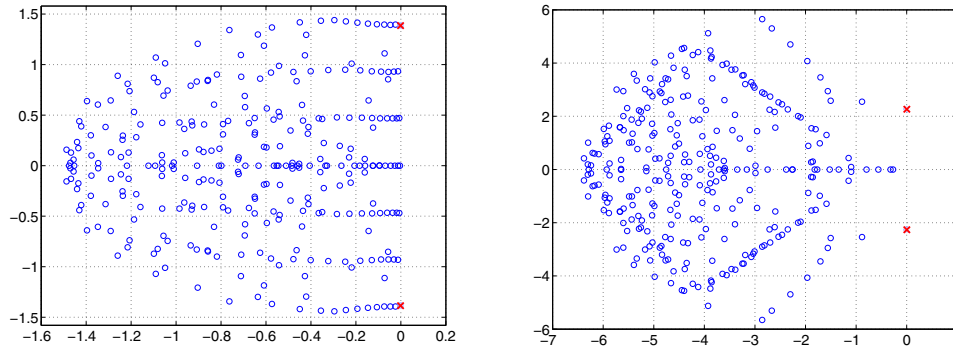
Fig. 3.7 *Eigenvalues of the Jacobian system for driven cavity flow (left) and flow over an obstacle (right).*

Without any information about the location of the important eigenvalues, iterative methods may have difficulty finding them.

IFISS provides an easy way to generate interesting benchmark problems and to incorporate them into the study of new algorithms. In particular, it is known that two classic problems, driven cavity flow and flow over an obstacle, exhibit Hopf bifurcations. Examples of the distributions of 300 eigenvalues closest to the origin for these two problems are shown in Figure 3.7. The rightmost eigenvalues are highlighted with "×". Note that for the cavity problem (shown on the left), there are many poorly separated eigenvalues, and an iterative method applied to $M^{-1}J$ will have great difficulty finding them. The alternative is to find the largest inverse eigenvalue of (3.6) using $J^{-1}M$. If there are many eigenvalues smaller in modulus than the critical ones, as in the case of the cavity problem, this also causes problems for (inverse) iterative methods. In [11], Elman et al. have used IFISS to explore and develop new algorithms to address these issues.

For models of incompressible flow, the Jacobian matrix has the structure

$$J(\nu) = \begin{pmatrix} \nu L + N(\nu) & B^T \\ B & 0 \end{pmatrix},$$

where ν is the kinematic viscosity. If we have an estimate ν_0 for the critical value ν^*, then

$$J(\nu^*) \approx \begin{pmatrix} \nu_0 L + N(\nu_0) & B^T \\ B & 0 \end{pmatrix} + (\nu^* - \nu_0) \begin{pmatrix} L + N'(\nu_0) & 0 \\ 0 & 0 \end{pmatrix} \equiv A + \lambda C,$$

where L is a discrete Laplacian and $N(\nu)$ is a parameter-dependent matrix that represents the effect of nonlinearity in the Navier–Stokes equations, including convection.[3] When the Jacobian is of this form, (3.6) is a parameterized eigenvalue problem $(A + \lambda C)\mathbf{x} = \mu M \mathbf{x}$. It is shown in [25] that the solution to this problem can be obtained by solving a related eigenvalue problem with a special *Lyapunov structure*

(3.7) $$AZM^T + MZA^T + \lambda (CZM^T + MZC^T) = 0.$$

The quantity sought is the smallest real parameter λ for which there is a solution Z in (3.7). (The matrix solution Z, which is also referred to as an eigenvector, is of rank 2

---

[3] In [11], the matrix $C = \frac{dJ}{d\nu}(\nu_0)$ was approximated using a forward difference approximation.

and has the form $Z = VDV^T$; see [11, 25] for details.) This has the desirable feature of requiring a small real eigenvalue, and a variant of inverse iteration can be used to find it. Once λ is obtained, the critical parameter is estimated as $\nu^* = \nu_0 + \lambda$. It is also shown in [25] that the rightmost eigenvalues of the parameterized eigenvalue problem can also be found easily by solving the 2×2 problem

$$V^T(A + \lambda C)\mathbf{y} = \mu V^T M V \mathbf{y}.$$

In [11], this idea was explored in detail for the benchmark problems above.

In general, it can be difficult to identify the desired (rightmost) eigenvalue using a method such as implicitly restarted Arnoldi, especially when, as in Figure 3.7, there are many poorly separated eigenvalues. Using the alternative approach of trying to find the largest eigenvalue of the inverse eigenvalue problem is also difficult when the imaginary part of the eigenvalue sought is large. The method discussed above is potentially more robust than these conventional methods for identifying Hopf bifurcations. Having access to IFISS is what made this study possible.

4. Summary and Future Developments. The IFISS package is designed to be accessible—anyone with a basic knowledge of MATLAB can use it to gain understanding of the key concepts underlying the numerical modeling of PDEs. The case studies described in this review article highlight the potential of using the IFISS package to explore the numerical analysis of the PDEs underlying incompressible flow. The package is currently used in academic institutions around the world to support research and teaching. Examples of extensions to IFISS that are currently being developed include the incorporation of unstructured triangular meshes and the extension to PDE problems with random data. In the future, we also envisage extending the capabilities of IFISS to three dimensions, making use of parallel MATLAB constructs in order to take advantage of multicore architectures.

REFERENCES

[1] M. Benzi and X. P. Guo, *A dimensional split preconditioner for Stokes and linearized Navier-Stokes equations*, Appl. Numer. Math., 61 (2011), pp. 66–76.

[2] M. Benzi, M. A. Olshanskii, and Z. Wang, *Modified augmented Lagrangian preconditioners for the incompressible Navier-Stokes equations*, Internat. J. Numer. Methods Fluids, 66 (2011), pp. 486–508.

[3] D. Braess, *Finite Elements*, 3rd ed., Cambridge University Press, London, 2007.

[4] S. C. Brenner and L. R. Scott, *The Mathematical Theory of Finite Element Methods*, 3rd ed., Springer-Verlag, New York, 2010.

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

[6] A. Dedner, R. Klöfkorn, M. Nolte, and M. Ohlberger, *DUNE–FEM: The FEM Module of the Distributed and Unified Numerics Environment*, http://dune.mathematik.uni-freiburg.de, accessed January 1, 2014.

[7] H. S. Dollar, N. I. M. Gould, M. Stoll, and A. J. Wathen, *Preconditioning saddle-point systems with applications in optimization*, SIAM J. Sci. Comput., 32 (2010), pp. 249–270.

[8] H. Elman, D. Silvester, and A. Wathen, *Iterative methods for problems in computational fluid dynamics*, in Iterative Methods in Scientific Computing, R. Chan, T. Chan, and G. Golub, eds., Springer, Singapore, 1997, pp. 271–327.

[9] H. C. Elman, V. E. Howle, J. Shadid, R. Shuttleworth, and R. Tuminaro, *Block preconditioners based on approximate commutators*, SIAM J. Sci. Comput., 27 (2006), pp. 1651–1668.

[10] H. C. Elman, V. E. Howle, J. Shadid, D. J. Silvester, and R. Tuminaro, *Least squares preconditioners for stabilized discretizations of the Navier–Stokes equations*, SIAM J. Sci. Comput., 30 (2007), pp. 290–311.

[11] H. C. Elman, K. Meerbergen, A. Spence, and M. Wu, *Lyapunov inverse iteration for identifying Hopf bifurcations in models of incompressible flow*, SIAM J. Sci. Comput., 34 (2012), pp. A1584–A1606.

[12] H. C. Elman, A. Ramage, and D. J. Silvester, *Algorithm 886: IFISS, a Matlab toolbox for modelling incompressible flow*, ACM Trans. Math. Software, 33 (2007), 14.

[13] H. C. Elman, D. J. Silvester, and A. J. Wathen, *Finite Elements and Fast Iterative Solvers with Applications in Incompressible Fluid Dynamics*, Oxford University Press, Oxford, UK, 2005.

[14] V. Girault and P.-A. Raviart, *Finite Element Methods for Navier-Stokes Equations*, Springer-Verlag, Berlin, 1986.

[15] G. H. Golub, M. Stoll, and A. J. Wathen, *Approximation of the scattering amplitude and linear systems*, Electron. Trans. Numer. Anal., 31 (2008), pp. 178–203.

[16] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 3rd ed., The Johns Hopkins University Press, Baltimore, MD, 1996.

[17] W. J. F. Govaerts, *Numerical Methods for Bifurcations of Dynamical Equilibria*, SIAM, Philadelphia, 2000.

[18] M. D. Gunzburger, *Finite Element Methods for Viscous Incompressible Flows*, Academic Press, San Diego, CA, 1989.

[19] C.-H. Guo, N. J. Higham, and F. Tisseur, *An improved arc algorithm for detecting definite Hermitian pairs*, SIAM J. Matrix Anal. Appl., 31 (2009), pp. 1131–1151.

[20] C. Johnson, *Numerical Solution of Partial Differential Equations by the Finite Element Method*, Dover, Mineola, NY, 2009.

[21] D. A. Kay, P. M. Gresho, D. F. Griffiths, and D. J. Silvester, *Adaptive time-stepping for incompressible flow. Part II: Navier–Stokes equations*, SIAM J. Sci. Comput., 32 (2010), pp. 111–128.

[22] D. Kay, D. Loghin, and A. J. Wathen, *A preconditioner for the steady-state Navier–Stokes equations*, SIAM J. Sci. Comput., 24 (2002), pp. 237–256.

[23] A. Logg, K.-A. Mardal, and G. Wells, eds., *Automated Solution of Differential Equations by the Finite Element Method*, Lect. Notes Comput. Sci. Eng. 84, Springer, Berlin, 2012, http://fenicsproject.org.

[24] K.-A. Mardal and R. Winther, *Preconditioning discretizations of systems of partial differential equations*, Numer. Linear Algebra Appl., 18 (2010), pp. 1–40.

[25] K. Meerbergen and A. Spence, *Inverse iteration for purely imaginary eigenvalues with application to the detection of Hopf bifurcations in large-scale problems*, SIAM J. Matrix Anal. Appl., 31 (2010), pp. 1982–1999.

[26] C. C. Paige and M. A. Saunders, *Solution of sparse indefinite systems of linear equations*, SIAM J. Numer. Anal., 12 (1975), pp. 617–629.

[27] J. Rommes, *Arnoldi and Jacobi-Davidson methods for generalized eigenvalue problems $Ax = \lambda Bx$ with singular B*, Math. Comp., 77 (2008), pp. 995–1015.

[28] T. Rusten and R. Winther, *A preconditioned iterative method for saddlepoint problems*, SIAM J. Matrix Anal. Appl., 13 (1992), pp. 887–904.

[29] Y. Saad, *Iterative Methods for Sparse Linear Systems*, 2nd ed., SIAM, Philadelphia, 2003.

[30] Y. Saad, *Numerical Methods for Large Eigenvalue Problems*, revised ed., SIAM, Philadelphia, 2011.

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

[32] A. Schmidt and K. G. Siebert, *ALBERTA: An Adaptive Hierarchical Finite Element Toolbox*, http://www.alberta-fem.de, accessed January 1, 2014.

[33] D. J. Silvester, A. Bespalov, and C. E. Powell, *A framework for the development of implicit solvers for incompressible flow problems*, Discrete Contin. Dyn. Syst. Ser. S, 5 (2012), pp. 1195–1221.

[34] D. J. Silvester, H. C. Elman, D. Kay, and A. J. Wathen, *Efficient preconditioning of the linearized Navier-Stokes equations for incompressible flow*, J. Comput. Appl. Math., 128 (2001), pp. 261–279.

[35] D. J. Silvester, H. C. Elman, and A. Ramage, *Incompressible Flow Iterative Solution Software (IFISS) Installation and Software Guide*, Version 3.3, http://www.maths.manchester.ac.uk/ifiss/ifiss_guide_3.3.pdf, 2013.

[36] D. J. Silvester, H. C. Elman, and A. Ramage, *Incompressible Flow and Iterative Solver Software (IFISS)*, Version 3.3, http://www.manchester.ac.uk/ifiss/, 2013.

[37] D. J. Silvester and C. E. Powell, *Potential (Incompressible) Flow and Iterative Solver Software (PIFISS)*, http://www.manchester.ac.uk/ifiss/pifiss.html, 2007.

[38] D. J. Silvester and V. Simoncini, *An optimal iterative solver for symmetric indefinite systems stemming from mixed approximation*, ACM Trans. Math. Software, 37 (2010), 42.

[39] A. M. SONNET AND A. RAMAGE, *Computational Fluid Dynamics for Nematic Liquid Crystals*, Research Report 28-07, Department of Mathematics and Statistics, University of Strathclyde, 2007.

[40] G. W. STEWART, *Matrix Algorithms. Volume* II: *Eigensystems*, SIAM, Philadelphia, 2001.

[41] G. STRANG AND G. J. FIX, *An Analysis of the Finite Element Method*, 2nd ed., Wellesley–Cambridge Press, 2008.

[42] L.-Y. SUN AND J. LIU, *Constraint preconditioning for nonsymmetric indefinite linear systems*, Numer. Linear Algebra, 17 (2010), pp. 677–689.

[43] D. B. SZYLD AND F. XUE, *Efficient preconditioned inner solves for inexact Rayleigh quotient iteration and their connections to the single-vector Jacobi–Davidson method*, SIAM J. Matrix Anal. Appl., 32 (2011), pp. 993–1018.

[44] E. ULLMANN, H. C. ELMAN, AND O. G. ERNST, *Efficient iterative solvers for stochastic Galerkin discretizations of log-transformed random diffusion problems*, SIAM J. Sci. Comput., 34 (2012), pp. A659–A682.

[45] M. B. VAN GIJZEN AND P. SONNEVELD, *Algorithm* 913: *An elegant IDR(s) variant that efficiently exploits biorthogonality properties*, ACM Trans. Math. Software, 38 (2011), 5.

[46] J. T. ZHOU AND Q. A. NIU, *Substructure preconditioners for a class of structured linear systems of equations*, Math. Comput. Model., 52 (2010), pp. 1547–1553.