INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids 2001; **36**: 309–327 (DOI: 10.1002/fld.135)

Large-scale eigenvalue calculations for stability analysis of steady flows on massively parallel computers

Richard B. Lehoucq[a] and Andrew G. Salinger[b,*]

[a] *Applied and Numerical Mathematics Department, Sandia National Laboratories, Albuquerque, NM, U.S.A.*
[b] *Parallel Computational Sciences Department, Sandia National Labotatories, Albuquerque, NM, U.S.A.*

SUMMARY

This paper presents an approach for determining the linear stability of steady states of partial differential equations (PDEs) on massively parallel computers. Linearizing the transient behavior around a steady state solution leads to an eigenvalue problem. The eigenvalues with the largest real part are calculated using Arnoldi's iteration driven by a novel implementation of the Cayley transformation. The Cayley transformation requires the solution of a linear system at each Arnoldi iteration. This is done iteratively so that the algorithm scales with problem size. A representative model problem of three-dimensional incompressible flow and heat transfer in a rotating disk reactor is used to analyze the effect of algorithmic parameters on the performance of the eigenvalue algorithm. Successful calculations of leading eigenvalues for matrix systems of order up to 4 million were performed, identifying the critical Grashof number for a Hopf bifurcation. Copyright © 2001 John Wiley & Sons, Ltd.

1. INTRODUCTION

Computing a numerical solution to the discretized Navier–Stokes equations for system sizes of $\mathcal{O}(10^4–10^5)$ is commonplace. Massively parallel computers have demonstrated the ability to simulate three-dimensional fluid flow, where the system's size is $\mathcal{O}(10^6–10^7)$. However, the linear hydrodynamic stability of flows for these latter system sizes is typically not computed. A linear stability analysis capability is an important tool for performing engineering design using computations.

A standard approach used to determine the dynamical behavior of the solution is *via* a numerical time integration of the discretized equations. Another approach is to compute steady state solutions and then to determine their stability by computing selected eigenvalues of a large-scale generalized eigenvalue problem. This latter approach is the subject of the current study. This approach has the advantage that a steady state solution can be efficiently located with robust solver and then can be tracked using continuation techniques. Each

* Correspondence to: Applied and Numerical Mathematics Department, Sandia National Laboratories, MS 1110, PO Box 5800, Albuquerque, NM 87185-1110, U.S.A.

Received 14 June 2000
Revised 25 April 2000

resulting solution can be classified as stable or not. A time integration approach only determines stable steady states and is highly dependent on initial conditions. Hence, qualitative information describing the behavior of the Navier–Stokes system upon parameter changes (e.g., Grashof, Rayleigh, and Reynolds numbers) cannot be readily determined. We refer the reader to the recent studies in References [1–4] for further information and citations to the recent literature. We do note that there are techniques for augmenting codes that time integrate to the steady state [5]. We are unaware of any studies where augmenting a transient code with these techniques is used to study the qualitative behavior of complex three-dimensional flow.

Our interest is in characterizing the stability of complex three-dimensional systems with coupled fluid flow, heat transfer, and mass transfer. In particular, we interested in the numerical solution of large-scale generalized eigenvalue problems that arise from finite element methods for the Navier–Stokes equations when the matrix system size is of $n = \mathcal{O}(10^6)$. The solution of a generalized eigenvalue problem with an Arnoldi iteration necessarily involves solving linear systems, but for the targeted applications sparse direct methods are not a viable alternative. They possibly require $\mathcal{O}(n^2)$ operations plus a prohibitive amount of memory and are not scalable to hundreds or thousands of processors. Instead, this paper considers the use of iterative methods for the necessary linear solves on massively parallel machines. Along with a scalable eigensolver, such an approach allows stability analysis to be performed on large systems arising from three-dimensional models.

We present a large-scale eigenvalue algorithm that allows us to determine the linear stability of a representative problem of three-dimensional incompressible flow and heat transfer in a rotating disk reactor. While the steady flow for this application is axisymmetric and can be computed with a two-dimensional model, the stability of the flow to three-dimensional disturbances is needed to confidently use the results to design reactors. Moreover, axisymmetric modes located with the three-dimensional calculation can be verified against the more routine two-dimensional calculations. We carefully discuss the influence of the various algorithmic parameters on the performance of the stability analysis. Successful calculations were performed on this problem where the order of the matrix eigenvalue problem was 4 million. Our algorithm identified a critical Grashof number for a Hopf bifurcation above which the reactor exhibits undesirable flow behavior.

The contribution of our study is the detailed documentation of the overall integration and solution process that is needed when sophisticated large-scale linear algebra techniques are used in conjunction with a fully non-linear steady state Navier–Stokes solver on a massively parallel computer. We are unaware of another study similar in scope to ours. However, more work needs to be accomplished so that large-scale linear stability analysis becomes an everyday tool of the analyst and design engineer. We list several outstanding topics for further research at the end of our paper.

Our paper is organized as follows. Section 2 discusses a representative problem of incompressible flow and heat transfer in a rotating disk reactor and Section 3 reviews the computation of a steady state solution. Section 4 formulates the eigenvalue problem used to compute the stability of the steady state and describes in detail our numerical scheme for solving the large-scale generalized eigenvalue problem using the Cayley transformation. Section 5 discusses in detail some of the issues related to using an iterative linear solver on parallel computers as part of the eigenvalue calculation. Section 6 applies the linear stability

analysis capability to determine the critical Grashof number for a Hopf bifurcation. We summarize our findings along with concluding remarks in Section 7.

2. STEADY FLOW PROBLEM

In this section we will describe our representative three-dimensional flow and heat transfer problem. A common approach to investigate the behavior of such a non-linear model is to track steady state solutions as they evolve with changes in system parameters. The fact that computing a solution to the steady state equations does not indicate whether the solution is stable or unstable motivates the development of a linear stability analysis capability for large-scale flow problems.

The rotating disk reactor (RDR) is a common system for growing high quality thin films via chemical vapor deposition. A top view and cross-sectional view of the reactor configuration are shown in Figure 1. The reactor consists of an outer cylindrical can and a smaller cylinder inside, which is rotating and heated on top. On this heated disk, the deposition occurs via surface reactions. Plug flow enters the top circular area, passes over the heated, rotating disk, and through the annular region before leaving the computational domain. Under certain conditions, the flow in the reactor is well represented by the von Karman similarity solution for flow over an infinite rotating disk [6], leading to desirable growth conditions.

The rotating disk reactor is known from experiments and calculations to exhibit flow instabilities. These include the formation of stable yet undesirable re-circulation cells [6–8] as well as unsteady flows [9]. The steady state behavior of this system can be uncovered by bifurcation analysis of steady state flow [7,10]. While these solution are axisymmetric and require only two-dimensional calculations, the stability analysis must be able to detect instabilities to non-axisymmetric states and so we have calculated the steady flow using a full three-dimensional model.

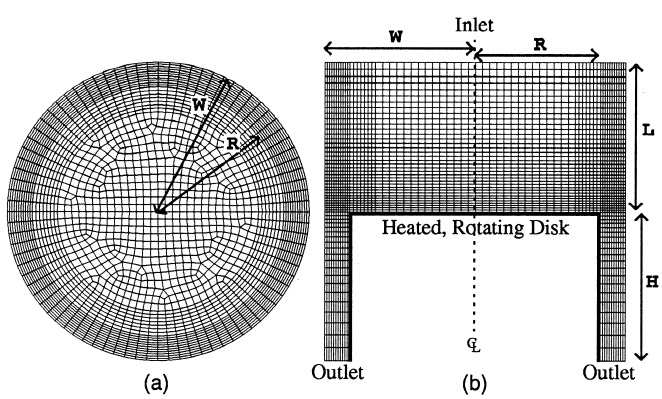

Figure 1. Top view (a) and cross-section (b) of rotating disk reactor for chemical vapor deposition reactions. The surface elements shown correspond to a 94656-element mesh of hexahedrons and 500215 unknowns.

In order to provide the most general results, we study only the fluid flow and heat transfer model (no mass transfer or reactions) and look at dimensionless numbers that are based on the assumptions of constant properties and the Boussinesq approximation for buoyancy. For the calculations in this paper, we have fixed the design parameters as shown in the figure, with $L/R = 1.0$, $W/R = 1.2$, and $H/R = 1.0$. The operating parameters in the model that are also fixed for these calculations include the rotational Reynolds number, $Re_{\text{rot}} = (\Omega R^2)/\nu = 83.77$; the Prandtl number $Pr = \nu/\alpha = 0.1$, where Ω is the rotation rate; ν is the kinematic viscosity; and α is the thermal diffusivity. The Reynolds number at the inlet is fixed at the matching condition, which is the flow rate that would be drawn by an infinite disk rotating at Re_{rot}. The asymptotic value for the inlet velocity [9] of $V = 0.884\sqrt{\Omega\nu}$ leads to an inlet Reynolds number of $Re = (2RV)/\nu = 16.18$. The final parameter is the Grashof number, measuring the relative strength of buoyancy forces to viscous forces. This parameter is varied at the end of the paper but for most calculations is held constant at $Gr = (g\beta TR^3)/\nu^2 = 15000$, where g is the magnitude of gravity, β is the thermal expansion coefficient, and T is the temperature difference between the heated disk and the inlet (with the outer walls also being held at the inlet temperature).

The steady state Navier–Stokes equations with the Boussinesq approximation are solved along with the continuity equation for incompressible flows. In addition, a heat equation with convection and conduction terms is solved. The equations are shown in Table I and include the time-dependent terms that are important for the formulation of the stability (eigenvalue) calculation. The discretized system can be expressed in the form

$$\mathbf{f}(\mathbf{y}, \dot{\mathbf{y}}, \tau) = \mathbf{0} \tag{1}$$

where $\mathbf{y}$ represents the vector of nodal unknowns, $\dot{\mathbf{y}}$ represents the time-dependent terms, and τ denotes the system parameter of interest, which for the current study is the Grashof number. The next section describes the computational procedure.

3. COMPUTATIONAL PROCEDURE

A non-trivial amount of computing infrastructure is needed before a linear stability analysis on a massively parallel machine can be undertaken. This section serves as an introduction/review

Table I. The governing PDEs for incompressible flow with heat transfer are shown in dimensionless form, including the Navier–Stokes equations with the Boussinesq approximation, the continuity equation, and a heat balance.

Momentum	$\dfrac{dv}{dt} + v \cdot \nabla v = -\nabla P + \nabla^2 v + GrTe_x$
Total mass	$\nabla \cdot v = 0$
Thermal energy	$\dfrac{dT}{dt} + v \cdot \nabla T = \dfrac{1}{Pr} \nabla^2 T$

of this infrastructure along with a brief discussion on the software tools used. This procedure can be summarized in these three steps, with details to follow

1. The finite element mesh is generated and a graph partitioning tools is employed to distribute the mesh among the processors.
2. A parallel finite element computational fluid dynamic (CFD) simulator is used to compute a steady state solution.
3. A parallel eigensolver determines the linear stability from a linearization of a steady state solution.

The CUBIT [11] mesh generation environment produces a three-dimensional unstructured finite element mesh of the domain into hexahedral elements. The mesh is partitioned among the processors with the Chaco graph partitioning tool [12], using a multi-level method and Kernihan–Lin refinement. Chaco partitions the mesh into sub-domains of equal numbers of mesh nodes and determines their assignments to the processors.

The parallel finite element CFD simulator we used is MPSalsa. We refer the reader to Reference [13] for a recent paper describing MPSalsa. Using the distributed unstructured mesh produced by CUBIT and Chaco, MPSalsa computes the steady state solution to the Navier–Stokes equations. The discretization used by MPSalsa is a variant of the Galerkin/least-squares (GLS) method [14]. This formulation includes a pressure stabilization term so that the velocity components, temperature, and pressure fields can all be represented with the same trilinear basis functions. The non-linear set of equations are solved using a fully coupled Newton's method [15], with an analytically calculated Jacobian matrix. This solution procedure, while memory intensive, leads to robust convergence to steady state solutions. MPSalsa has been successfully used to analyze flows and deposition profiles in chemical vapor deposition reactors [16,17].

The parallel implementation [18] of the ARPACK [19] eigensolver library numerically solves the sparse generalized non-symmetric eigenvalue problem that results from a linearization of a steady state solution computed by MPSalsa. Section 4 presents further details.

The Aztec [20] distributed memory parallel iterative library is used by both MPSalsa during the non-linear solve and parallel ARPACK. Aztec implements the standard Krylov techniques (GMRES, TFQMR, and BICSTAB) and preconditioners, including additive Schwarz domain decomposition schemes. A key aspect of Aztec is that given the partition of the domain among the processors produced by Chaco, the necessary parallel matrix–vector products are generated, thus relieving the user of these tedious computations.

We now discuss details associated with the representative steady flow problem previously discussed.

The mesh was partitioned into the same number of sub-domains as the number of processors for the run, which was 250 for most of the calculations described below. The calculations were performed on the Sandia-Intel Tflops Computer [21].

The computational domain is discretized using a mesh of 94656 hexahedral elements, which corresponds to 100043 nodes. The circular area is paved with an unstructured mesh, as can be seen from a top view in Figure 1(a), while the axial direction is structured, as seen in the cross-sectional view in Figure 1(b).

This discretization leads to a system of 500215 unknowns. Within each iteration of Newton's method, the finite element residuals and Jacobian matrix are assembled in 2.0 s when run on 250 processors. The linear solve is performed with the Aztec package [20] using a GMRES iteration without restarts. The matrix is first scaled to unit row sum and then an additive Schwartz domain decomposition preconditioner is computed, where on each subdomain (with one subdomain per processor), ILU is used as the solver with a fill-in factor of 7. The fill-in factor is a parameter that allows the preconditioner to retain more non-zeros than the sparse Jacobian; in this case the ILU factorization process can create up to seven times as many. An average GMRES solve required 80 iterations and 30 s (including the time to construct the preconditioner) to reach a drop in the scaled residual to 10^{-3}. The steady state solution at $Gr = 15000$ was reached from a trivial initial guess in 7 min using two consecutive steady state solves for increasing Gr.

A visualization of the steady state flow is shown in Figure 2. Several streamlines are shown entering the top of the reactor, spiraling over the disk, and exiting through the annular region. This calculation does not give any information on the stability of the steady state solution to small perturbations.

For the remainder the article, excluding the mesh convergence study in Section 5.3, all numerical experiments on linear stability analysis algorithms are about the steady state calculations described in this section.

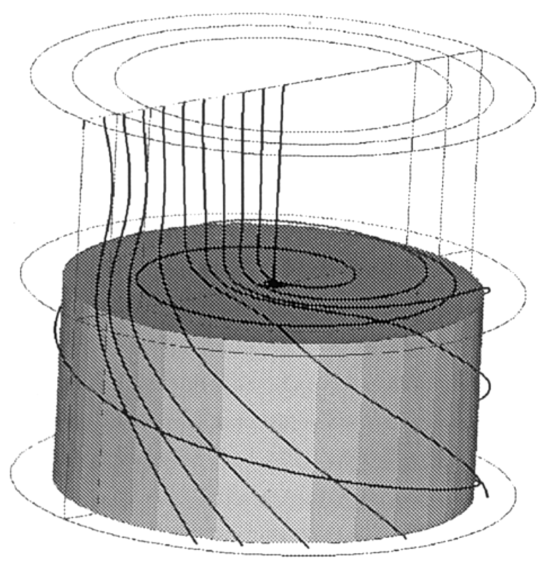

Figure 2. Visualization of the three-dimensional steady state flow solution. Streamlines enter the top, pass over the heated disk, and leave through the annular region.

4. STABILITY ANALYSIS CALCULATIONS

If we linearize Equation (1) about the steady state $(\mathbf{y}_0, \tau_0)$ to small perturbations $e^{\lambda t}\mathbf{z}$, we obtain the generalized eigenvalue problem

$$\mathbf{J}\mathbf{z} = \lambda \mathbf{B}\mathbf{z} \tag{2}$$

where $\mathbf{J} = \mathbf{f}_{\mathbf{y}}(\mathbf{y}_0, \mathbf{0}, \tau_0)$ and $\mathbf{B} = -\mathbf{f}_{\dot{\mathbf{y}}}(\mathbf{y}_0, \mathbf{0}, \tau_0)$ are the Jacobian and mass matrices, respectively. We denote the order of the matrices $\mathbf{J}$ and $\mathbf{B}$ by n. Because we use a Galerkin least-squares (GLS) discretization scheme, the generalized eigenvalue problem can be written as

$$\begin{pmatrix} \mathbf{L} & -\mathrm{C} \\ \mathbf{C}^T + \mathbf{G} & \mathbf{K} \end{pmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{p} \end{bmatrix} = \begin{pmatrix} \mathbf{M} & \mathbf{0} \\ \mathbf{N} & \mathbf{0} \end{pmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{p} \end{bmatrix} \lambda \tag{3}$$

where $\mathbf{u}$ is the vector of fluid velocity components and temperature unknowns, $\mathbf{p}$ is the pressure, $\mathbf{M}$ is the symmetric positive definite matrix of the overlaps of the finite element basis functions, $\mathbf{N}$ is an upwinded mass matrix, $\mathbf{L}$ is the sum of discretized diffusion, non-linear convection and any possible reaction operators, $\mathbf{C}$ is the discrete gradient, $\mathbf{C}^T$ is the discrete divergence operator, and $\mathbf{G}$ and $\mathbf{K}$ (pressure Laplacian) are stabilization terms arising from the GLS.

The steady state is stable if $\mathrm{Real}(\lambda) < 0$ for all the eigenvalues of Equation (3). Hence, computing approximations to the right-most eigenvalue determines the stability of the steady state.

4.1. Formulation of the eigenvalue problem

To compute the right-most eigenvalues, a shift-invert spectral transformation [22] is typically used to transform Equation (3) into the standard eigenvalue problem

$$\mathbf{T}_s\mathbf{z} = (\mathbf{J} - \sigma\mathbf{B})^{-1}\mathbf{B}\mathbf{z} = \gamma\mathbf{z}, \quad \gamma = \frac{1}{\lambda - \sigma} \tag{4}$$

The above formulation maps the infinite eigenvalue of Equation (3) (arising from singular $\mathbf{B}$) to zero. By selecting the pole σ near the imaginary axis, the right-most eigenvalues are mapped by $\mathbf{T}_s$ onto those of largest magnitude. However, because $\mathbf{J}$ and $\mathbf{B}$ are real matrices we only allow a real σ to keep the computation in real arithmetic.

The computational burden is in solving the linear set of equations with coefficient matrix $(\mathbf{J} - \sigma\mathbf{B})^{-1}\mathbf{B}$. Although this transformation maps the eigenvalues near the pole to those of largest magnitude, the transformation also maps the eigenvalues far from the pole to zero. Hence, the spectral condition number (the ratio of the largest-to-smallest, in magnitude, eigenvalues) of $\mathbf{T}_s$ can be quite large. The resulting linear systems will be difficult to solve because the rate of convergence of a Krylov-based iterative method [23,24] depends strongly upon the spectral condition number.

A better conditioned linear set of equations is achieved when using a generalized Cayley [22] transformation

$$\mathbf{T}_c\mathbf{z} = (\mathbf{J} - \sigma\mathbf{B})^{-1}(\mathbf{J} - \mu\mathbf{B})\mathbf{z} = \gamma\mathbf{z}, \quad \gamma = \frac{\lambda - \mu}{\lambda - \sigma} \tag{5}$$

We call μ the zero of the Cayley transform. In contrast to the shift-invert transform, the Cayley transform maps eigenvalues of Equation (3) far from the pole close to one. If we are able to select a pole σ that is to the right of all the eigenvalues (3) and choose $\mu > \sigma\lambda$, then the smallest eigenvalue of $\mathbf{T}_c$ is no smaller than one (in magnitude). Moreover, by judiciously choosing the pole, we can approximately bound the largest eigenvalue of $\mathbf{T}_c$ (in magnitude) resulting in a small (say order 10) spectral condition number.

The last two paragraphs describe a delicate balancing act. On the one hand, the ability to compute the right-most eigenvalue (λ) requires that the Cayley transformation maps these values to γ that are the largest (in magnitude). Such a situation allows the eigensolver to perform well. On the other hand, the iterative solver used to solve the linear systems arising from the Cayley transformation deteriorates if the ratio of $\max(|\gamma|)$ to $\min(|\gamma|)$ (the spectral condition number of $\mathbf{T}_c$) is large.

We remark that although the Cayley and the shift-invert spectral transformations both involve $(\mathbf{J} - \sigma\mathbf{B})^{-1}$, the system of linear equations solved by each transformation is distinct. Given a vector $\mathbf{x}$, the Cayley system requires the solution of

$$(\mathbf{J} - \sigma\mathbf{B})\mathbf{v} = (\mathbf{J} - \mu\mathbf{B})\mathbf{x} \tag{6}$$

so that $\mathbf{v} = \mathbf{T}_c\mathbf{x}$. Instead, the shift-invert system solves $(\mathbf{J} - \sigma\mathbf{B})\mathbf{v} = \mathbf{B}\mathbf{x}$. That the spectral condition number of the Cayley system can be tightly bounded (via a careful choice of σ and μ) implies that the Cayley system results in a better conditioned set of linear equations.

Before leaving our discussion on shift-invert and Cayley transformations, we are compelled to point out a significant drawback of these transformations. Unfortunately, the largest in magnitude eigenvalue of Equation (4) or (5) does not necessarily coincide with right-most eigenvalue of Equation (2). We emphasize that this is a direct result of using a rational transformation $\nu \equiv \nu(\lambda)$ necessary because $\mathbf{B}$ is singular. Hence, the solution of the resulting eigenvalue problem must be carefully undertaken. We caution the reader that there is no available theory to verify whether the right-most eigenvalue has been calculated. This is in contrast to the large-scale symmetric eigenvalue problem [25], where at the cost of computing a sparse direct factorization, reliability of the eigensolver can be determined.

4.2. Solution of the eigenvalue problem

We employ an implicitly restarted Arnoldi method (IRAM) as implemented in the parallel implementation [18] of the ARPACK [19] to compute eigenvalues and eigenvectors of the generalized eigenvalue problem. We have slightly modified the P_ARPACK subroutines `pdnaupd` and `pdeupd` to implement the Cayley transformation. We refer the reader to Reference [19] for full details about the software and underlying algorithm.

Figure 3 lists the scheme used for computing several (say k) right-most eigenvalues (3). A few remarks are in order. The starting vector is chosen so that it does not contain any components [26] in the null-space of $\mathbf{B}$. For all the eigenvalue problems solved, the value of $m = 24$ was used. At step 2, the eigenvalue (θ approximating γ) of the $m \times m$ Hessenberg matrix are mapped back to the system defined by Equation (3) via the inverse Cayley transformation resulting in approximations of $\hat{\lambda}$. The eigensolver is terminated when these k right-most approximate eigenvalues satisfy the user-specified tolerance. The check that must be satisfied is $\|\mathbf{J}\hat{\mathbf{z}} - \hat{\lambda}\mathbf{B}\hat{\mathbf{z}}\|/\|\mathbf{B}\hat{\mathbf{z}}\|$, where $\hat{\mathbf{z}}$ is the associated approximate eigenvector. By implicitly restarting the Arnoldi iteration, we compute a new starting vector for the subsequent run (step 1). Implicitly restarting is an efficient and stable manner to restart Arnolid's method so that storage requirements remain fixed for the computation. Finally, at step 5, the new pole and zero for the next Cayley transformation are updated so that the spectral condition number of $\mathbf{T}_c$ is of order 10.

We remark that there are two iterations—an outer and inner iteration. The outer iteration is Step 1 of the algorithm listed in Figure 3. During each of these outer iterations, there is an inner iteration used to solve the linear set of Equations (6) arising from applying $\mathbf{T}_c$. We use a preconditioned GMRES iteration for solving this linear set of equations. The next section discusses details associated with the inner iteration.

The two parameters σ and μ in the Cayley transformation give a considerable amount of flexibility over what eigenvalues will be located by Arnoldi's method, how accurately they will be calculated, and how expensive the calculation will be. This major consideration is the size $|\gamma|$ for the eigenvalues λ of interest. Eigenvalues λ that are mapped to large $|\gamma|$ will emerge and quickly be approximated by Arnoldi's method. We present results that quantify the various trade-offs in picking these parameters while preserving the spectral condition number of $\mathbf{T}_c$.

A good choice for these parameters is for the right-most eigenvalues of interest, λ_i for $i = 1{:}k$, to have real parts that satisfy $2\sigma - \mu < \text{Real}(\lambda_i) < \sigma < \mu$. This implies that these λ_i are mapped so that $|\gamma(\lambda_i)| \geq 2$ as long as $|\text{Imag}(\lambda_i)|$ is not large compared with $\sigma - \text{Real}(\lambda_i)$.

To illustrate how the Cayley transformation maps eigenvalues of the system, we plot the magnitude of Cayley transformation in Figure 4. This figure shows how λ is mapped to $|\gamma|$ for

Start P_ARPACK with the vector $\mathbf{v} = \mathbf{J}^{-1}\mathbf{B}\mathbf{x}$ where $\mathbf{x}$ is random vector.
Select a pole σ and zero μ for $\mathbf{T}_c$.

1. Compute m iterations of Arnoldi's method with $\mathbf{T}_c$ using the starting vector $\mathbf{v}$. Compute m eigenvalues (the θ's approximating the γ's) of the order m upper Hessenberg matrix constructed by P_ARPACK.
2. Map the θ's to $\hat{\lambda}$'s (approximations to the λ's) via the inverse Cayley transformation.
3. Exit if the k rightmost $\hat{\lambda}$'s satisfy the user specified tolerance.
4. Implicitly restart Arnoldi's method resulting in an updated starting vector $\mathbf{v}$.
5. Update σ and μ using the current approximate eigenvalues.

Figure 3. Computing the leading eigenvalues of $\mathbf{J}\mathbf{z} = \mathbf{M}\mathbf{z}\lambda$ using the Cayley transformation and IRAM.

fixed values of $\sigma = 20$ and $\mu = 0$ and four imaginary portions of λ. Note that as the real part of λ decreases, $|\gamma|$ approaches unity. The σ and μ values in this plot map the real eigenvalues in the range of $-50 < \mathrm{Real}(\lambda) < 20$ to magnitudes in the Cayley transformed system of $|\gamma| > 2$, which is sufficiently well separated from the many eigenvalues near $|\gamma| = 1$ for the eigensolver. For any real eigenvalues satisfying $\mathrm{Real}(\lambda) < -40$, the Cayley transformation maps these eigenvalues so that $1 < |\gamma(\lambda)| < 2$. Hence, Arnoldi's method will provide the best approximations to the eigenvalues satisfying $2\sigma - \mu < \mathrm{Real}(\lambda_i) < \sigma$.

Figure 4 also indicates that eigenvalues with large imaginary parts are mapped to small $|\gamma|$. Therefore it is difficult to compute approximations to eigenvalues with large imaginary parts—this is an example of the drawback of using a rational transformation discussed at the end of Section 4.1. For instance, an eigenvalue at $0 \pm 50\imath$ might not be located if there are many eigenvalues with large $|\gamma|$, such as near $-5 \pm 0\imath$. This problem is resolved by moving the σ parameter to the right and increasing the Arnoldi space m needed by P_ARPACK.

An appropriate choice of σ, μ, and the size of the Arnoldi space m is therefore a trade-off between two factors: selecting m large enough so that the right-most eigenvalues λ are reliably computed by the eigensolver and avoiding large values of $|\gamma|$, so that the resulting linear systems can be efficiently solved with preconditioned iterative methods.

We end this section with some thoughts regarding our use of an Arnoldi-based eigensolver. There is a popular belief that Arnoldi's method is not as reliable as the power method (or sub-space iteration). However, as long the starting vector contains some component of the right-most eigenvector, both Arnoldi's method and the power method will find the right-most eigenvalue. What is the basis of the aforementioned popular belief? First, Arnoldi's method typically produces good approximations to eigenvalues much faster than the power method. And herein lies the problem: users of Arnoldi's method prematurely halt its progress when several of the dominant (those largest in magnitude) eigenvalue of Equation (4) and (5) are computed—even though the dominant eigenvalues may not correspond to the right-most

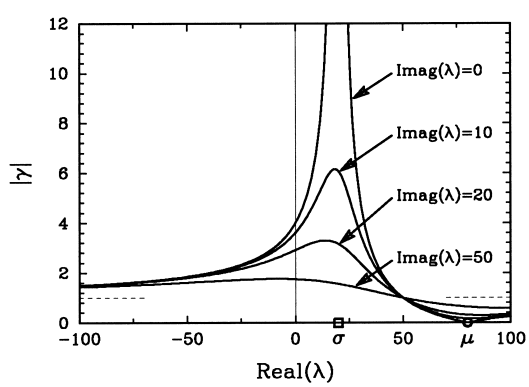

Figure 4. Plot of the transformation of the eigenvalue in the physical system to those in the Cayley transformed system. The magnitude of the transformed eigenvalue is plotted against the real part of λ for three different imaginary contributions to λ.

eigenvalues of Equation (2). We emphasize that this will cause difficulty whether the power or Arnoldi's method is employed. Arnoldi's method only allows the false conclusions to be made faster.

5. PRECONDITIONED ITERATIVE LINEAR SOLVES

The computationally intensive part of the eigenvalue calculation is the linear solve with $\mathbf{T}_c$ (inner iteration) that occurs during each outer iteration of Arnoldi's method. Since we are targeting large-scale problems and algorithms that scale to thousands of processors, we are limited to preconditioned iterative linear solves of distributed matrices. In this section we first discuss the tolerances used for the linear solver and eigensolver, the details associated with our use of Aztec [20] and the outcome of a mesh resolution study.

5.1. Error tolerances

Figure 5 plots the residuals associated with the three right-most eigenvalues and eigenvectors versus the convergence tolerance used for the linear solver. The eigensolver residual error is defined as

$$\frac{\|\mathbf{J}\hat{\mathbf{z}} - \hat{\lambda}\mathbf{B}\hat{\mathbf{z}}\|}{\|\mathbf{B}\hat{\mathbf{z}}\|} \tag{7}$$

where $\hat{\lambda}$ and $\hat{\mathbf{z}}$ are the computed eigenvalue and eigenvector approximations. The residual contains the normalization with $\mathbf{B}\hat{\mathbf{z}}$ because P_ARPACK normalizes $\|\hat{\mathbf{z}}\| = 1$ and so Equation (7) is independent of the scaling of the data. The linear solver uses the criterion

$$\frac{\|\mathbf{b} - \mathbf{A}\mathbf{x}_j\|}{\|\mathbf{B}\mathbf{v}\|} < \eta \tag{8}$$

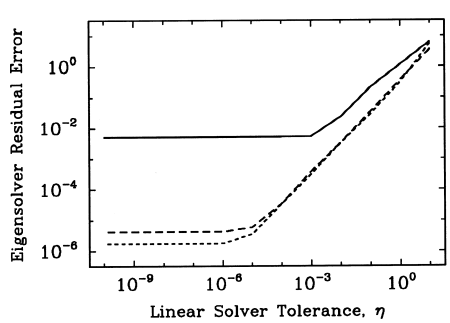

Figure 5. The error in the eigenvalue calculation for the three rightmost eigenvalues is shown to be a function of the acceptance criterion of the iterative linear solver. The eigensolver residual error and linear solver tolerance is given in Equations (7) and (8).

where $\mathbf{A} = \mathbf{J} - \sigma\mathbf{B}$ and $\mathbf{b} = (\mathbf{J} - \mu\mathbf{B})\mathbf{v}$ from Equation (6), and η is a tolerance parameter that must be chosen. Here, $\mathbf{v}$ is the distributed unit vector provided by P_ARPACK that is to be transformed via $\mathbf{T}_c$ during the ith ($1 \leq i \leq m$) outer iteration and $\mathbf{x}_j$ is the approximate solution after j GMRES iterations (the inner iteration).

In the experiment shown in Figure 5 we show the influence of η on the eigensolver residual error (7). The residual error of the right-most eigenvalue pair (denoted by the solid line) stops decreasing at $\eta \approx 10^{-3}$. The residual error of the next two eigenvalue stops decreasing when $\eta \approx 10^{-6}$. Driving the residual errors lower would require a larger Arnoldi space m or a different choices of σ and μ. For the rest of the calculations in this section the linear solver tolerance was fixed at $\eta = 10^{-3}$.

A series of calculations are presented in Figure 6 to illustrate the tradeoffs in choosing σ. The right-most eigenvalue of the steady state calculation has real part equal to 0.3 and we set $\mu = 80$. As σ is increased from 1 to 70, the maximum $|\gamma|$ decreases (recall the $\max(|\gamma|)$ is approximately equal to the spectral condition number of $\mathbf{T}_c$). This decrease is seen to correspond directly to the decrease in the CPU time and memory requirements for the linear solve, as measured by solution time and the average number of GMRES iterations needed for a solve. However, as σ is increased so too do the residuals (7). For this problem, a choice of $\sigma = 20$ provides a balance between efficiency and accuracy. The trends seen as a function of σ

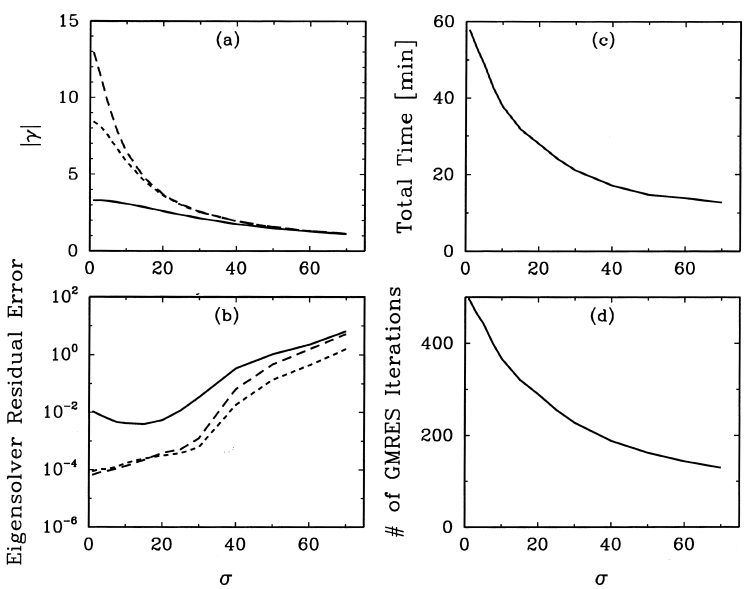

Figure 6. The model eigenvalue calculation is repeated for several values of the σ parameter in the Cayley transformation. The effect of σ on (a) the magnitude of the three largest complex γ pairs, (b) the residual errors (7) associated with the three largest complex γ pairs in the transformed system, (c) the time for the calculation, and (d) the average number of GMRES iterations needed for each of the 24 linear solvers are shown.

point to a remedy for systems where the preconditioned linear solver is unable to reach the specified tolerance: increase σ and μ until the linear problem can be solved and then increase the number of outer iterations needed by the eigensolver (and therefore the number of linear solves required) until Equation (7) is sufficiently small for the right-most eigenvalue.

5.2. Using Aztec

Another issue associated with preconditioned Krylov-based iterative solvers is the robustness of the algorithm in reaching a specified tolerance. This includes the access to, and selection of, an appropriate preconditioner and solution algorithm. For the calculations in this paper, the Aztec linear solver library was used. An additive Schwarz domain decomposition preconditioner with ILUT as a solver on each subdomain was used. Fill-in was limited so that the resulting preconditioner used no more than four times more memory than the matrix itself. The preconditioner is computed once and reused for each iteration of Arnoldi's method needed by the eigenvalue.

Since Figure 6(d) shows that a few hundred GMRES iterations are possibly needed during each outer iteration, the numerical stability of the GMRES implementation becomes critical. Originally, a classical Gram–Schmidt scheme was used for the orthogonalization, but the lack of numerical stability prevented the GMRES algorithm from reaching the required tolerance. Two alternative orthogonalization schemes were used successfully: two-step classical Gram–Schmidt (CGS) and a modified Gram–Schmidt (MGS). The CGS method uses two steps of orthogonalization (the second step is the correction for the possible loss of orthogonality of the Arnoldi basis vectors) but the number of global communication points remains fixed (at two) independent of the GMRES iteration. On the other hand, the MGS scheme requires i communications to orthogonalize i vectors (at GMRES iteration i) but no additional floating point operations (flops). Both schemes reached the specified tolerance and provided identical results in terms of the number of GMRES iterations needed.

There was a significant difference in the scalability of the two algorithms as the number of processors was changed. The time required to perform a single linear solve (using a precalculated preconditioner) was recorded with the number of processors being varied from 100 to 1000. The message of this calculation is clear when presenting the total CPU time (calculated as the wall clock time multiplied by the number of processors) as shown in Figure 7. When the problem was run on 100 processors, the extra communications required by MGS were slightly less expensive than the extra flops required by the CGS algorithm. However, as the number of processors is increased, it is seen that the communication time in MGS starts to dominate, while the total CGS time remains relatively flat. At 1000 processors, the CGS routine requires only about a quarter of the time of the MGS method.

The results in Figure 7 show that the two-step CGS scheme scales much better than the MGS scheme. It should be pointed out that the inter-processor communication rate of the Sandia-Intel Tflop computer is very fast compared with more loosely coupled parallel machines, where we would expect the difference to be more dramatic and the cross-over point (at around 175 processors for this case) to occur at fewer processors.

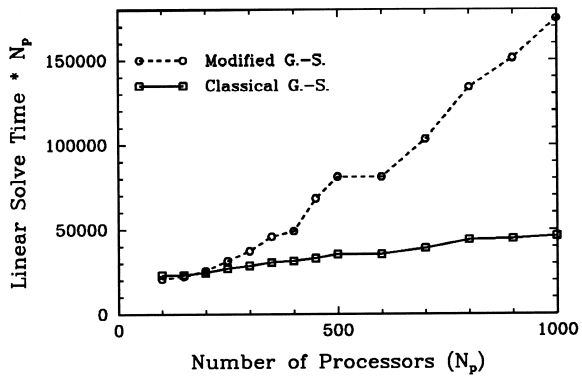

Figure 7. The parallel efficiency of two stable orthogonalization schemes in the GMRES algorithm is compared. The extra communications of the MGS approach scale poorly with number of processors, while the extra operations of the two-step CGS approach scale well.

5.3. Mesh resolution

All the calculations previously discussed were carried out for a single finite element mesh corresponding to just over a half million unknowns (see Section 2). While we have shown above that the eigenvalues are accurate for this given discretized system, a mesh resolution study verifies that these eigenvalues are good approximations to those of the continuous partial differentiation equation (PDE) model. The results of such a study are shown in Table II. The six eigenvalues with largest real parts at $Gr = 15000$ are shown for five successively finer meshes, each approximately doubling the number of unknowns of the previous. They range from 250000 to 4 million unknowns. Since the first eigenmode was determined through visualization to be axisymmetric, a final calculation on a very fine two-dimensional axisymmetric mesh of 2 million unknowns was used to verify this calculation.

The parameter value was chosen to be near a Hopf bifurcation. What we find is that while the coarsest mesh indicates a stable steady state, the second coarsest mesh (that was used in all

Table II. Mesh resolution studies at $Gr = 15\,000$ on the six eigenvalues with largest real parts.

n	$\lambda_{1,2}$	$\lambda_{3,4}$	$\lambda_{5,6}$
0.25 million	$-0.08 \pm 25.33\imath$	$-0.49 \pm 9.63\imath$	$-1.44 \pm 5.96\imath$
0.50 million	$0.35 \pm 25.16\imath$	$-0.05 \pm 9.50\imath$	-1.13 ± 5.91
1.0 million	$0.57 \pm 25.06\imath$	$0.21 \pm 9.36\imath$	$-0.98 \pm 6.01\imath$
2.0 million	$0.73 \pm 25.02\imath$	$0.39 \pm 9.31\imath$	$-0.85 \pm 6.03\imath$
4.0 million	$0.84 \pm 24.94\imath$	$0.50 \pm 9.22\imath$	$-0.78 \pm 6.08\imath$
2D mesh; 2.0 million	$1.06 \pm 24.69\imath$		

The coarse mesh results would indicate a stable solution, but the finest mesh shows that two pairs of eigenvalues have positive real parts.

other computations in this paper) shows one unstable eigenpair, and the three finest meshes predict that two eigenvalues are unstable. Only a narrow range of parameter values would see this behavior, where a different mesh gives different stability predictions. While the change in the eigenvalues with successive refinement is slowing, the values are still changing even when the number of unknowns increases from 2 to 4 million unknowns. Apparently we are not in the asymptotic regime.

There was no attempt to calculate a convergence rate because the data does not fall on a smooth curve. This is explained by several reasons. First, the mesh refinement of the unstructured mesh was not precisely uniform, but was done 'by hand' in an attempt to refine in all directions equally. The accuracy of the non-linear steady state calculation and of the linear solves within the eigensolver, both of which are iterative procedures subject to a stopping tolerance, are additional sources of error that influence that calculated eigenvalues. These results imply that extremely fine meshes and accurate linear and non-linear solves may be needed to pinpoint the exact parameter value of a Hopf bifurcation in three-dimensional problems. However, the fact that the coarsest mesh is converging to the same physical modes as the finest meshes implies that the system's behavior can be quickly explored with a relatively coarse mesh, and the finest meshes are only needed to locate the parameter values to a higher degree of accuracy.

Some more details on the 4 million unknown calculations follow. For this finest mesh, the steady state solution was reached from a trivial guess in three continuation steps in the *Gr* number, using 2.5 h of CPU time on 1024 processors. The leading eigenvalues of the sparse matrix with over 500 million non-zero elements were calculated in under 4 h, where each linear solve required about 12 min. The linear solves used row-sum scaling, an additive Schwartz domain decomposition preconditioner with ILUT used on each sub-domain (with a full-in factor of 8), and required an average of just over 700 iterations of (unrestarted) GMRES to converge.

In an effort to determine an asymptotic range rate of convergence, we exploited the axisymmetric nature of the problem to reduce the problem to two dimensions. For the two-dimensional problem, it was possible to isolate the effects of mesh resolution by using a uniform mesh and by ensuring the other sources of error were not a factor. That is, the convergence tolerance for the non-linear solver, the convergence tolerance for the iterative linear solver within the eigensolver, and the size of the Arnoldi space were all set so that the eigenvalues were computed to six digits.

Six meshes were used, ranging from 2225 to 2035205 unknowns, with each mesh being formed by bisecting the elements of the previous mesh in each dimension. The finest mesh found the right-most eigenvalue to be at $1.057 \pm 24.86\imath$. In Figure 8 the error in the value of the real and imaginary parts of the eigenvalue for the five coarser meshes compared with this value is plotted versus the relative mesh spacing h. The eigenvalue is seen to converge with order h^2. Since linear basis functions are used in the finite element discretization, this result was expected.

We end this section by cautioning the reader that although we have gone to many lengths to determine the right-most eigenvalue, we cannot guarantee that this eigenvalue has been calculated. Currently, there is no theory available that enables a check as to whether the right-most eigenvalue has truly been calculated. This is in contrast to the large-scale symmetric

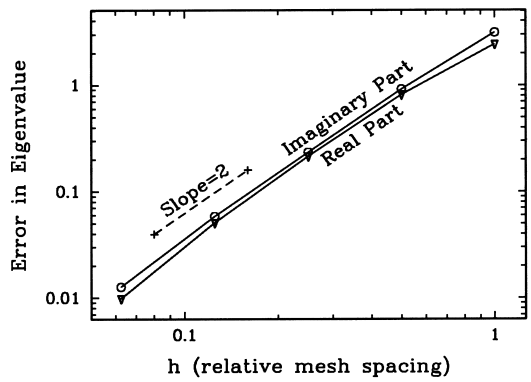

Figure 8. Rate of convergence of the rightmost eigenvalue of the two-dimensional axisymmetric problem.

eigenvalue problem [25], where at the cost of computing a sparse direct factorization, reliablility of the eigensolver can be determined.

6. REACTOR ANALYSIS

In this section we apply the linear stability analysis capability that has been represented above. Experiments have shown that the desirable non-recirculating flow in the rotating disk reactor can go unstable to periodic oscillations [9]. It is important during reactor design to be able to locate this instability. With that goal in mind, the steady state solution branch was tracked using first-order continuation and the leading eigenvalues were calculated at each step. The calculations were performed on the standard mesh corresponding to half a million unknowns.

Figure 9 shows how the six eigenvalues that have largest real part at $Gr = 15000$ evolve from $Gr = 10000$ to 16000. By interpolating between the symbols to where the curves cross the imaginary axis, the first Hopf bifurcation is seen to occur near 14800, the second just above 15000 and a third near 15500. By including the trends seen in the mesh resolution study in Table II, where the systems became less stable with more refined meshes, we can extrapolate that with a finer mesh the first Hopf bifurcation would fall in the range $Gr = 14000-14500$.

The eigenvectors associated with these largest eigenvalues are the perturbations that will not die out if the parameter value puts the system past the Hopf bifurcation. Visualization of the eigenvectors gives information that can be used to suggest modifications to the design or operation of the reactor to delay the onset of these unwanted instabilities. Since the instabilities involve oscillations between real and imaginary parts of the eigenvector, each of which corresponds to a three-dimensional flow field, it was not possible to produce satisfactory still pictures for this publication. What the visualization found was that the first Hopf bifurcation is an axisymmetric state with a toroidal roll cell. The oscillation is the roll cell being forced out by a counter-rotating roll cell. The second Hopf bifurcation breaks symmetry with a mode 1 instability, with a single large roll cell over the disk that rotates in time. The third Hopf

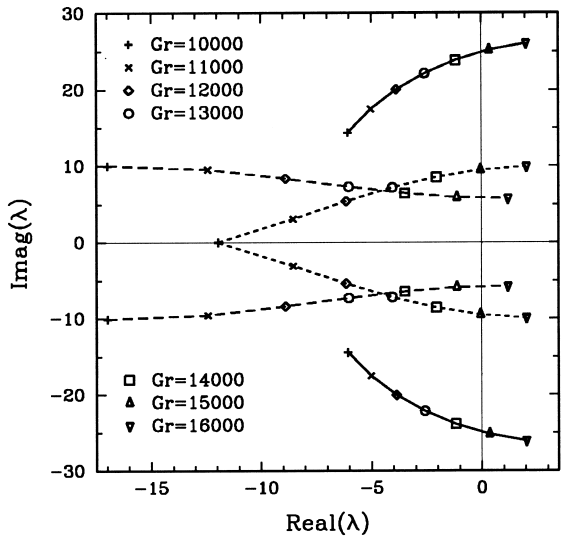

Figure 9. The tracking of the six largest eigenvalues as a function of parameter indicate a Hopf bifurcation near 14800.

bifurcation is a mode 2 symmetry breaking one, where there is up-flow in two quadrants of the disk and down-flow in the two others. Again, this flow structure precesses around the reactor in time. It is interesting that modes 0, 1, and 2 symmetry breakings occur under nearly the same conditions. While the problem could have been solved using a two-dimensional model with an axisymmetric formulation and complex arithmetic for the non-axisymmetric modes, the methods were developed as a general three-dimensional capability. And since Cartesian co-ordinates were used to model the system, the fact that the solution was axisymmetric did not simplify the calculations in any way.

7. SUMMARY AND CONCLUSIONS

A massively parallel code for calculating steady state of incompressible and reacting flows (MPSalsa) has been linked to P_ARPACK for calculating selected eigenvalues for the purpose of linear stability analysis. A novel implementation of the Cayley transform has been presented and analyzed for an example of three-dimensional flow and heat transfer in a rotating disk CVD reactor. This implementation allows control over the spectral condition number of the linear system that must be solved during each step Arnoldi's iteration used by P_ARPACK making it particularly well suited for use with scalable iterative linear solvers.

By using preconditioned Krylov based algorithms and software for iterative solutions of large sparse, distributed matrices, we were able to calculate several right-most eigenvalues for linearized systems corresponding to 4 million unknowns and 530 million non-zero matrix entries on 1024 parallel processors.

The stability of the flow in the rotating disk reactor was analyzed as a function of the Grashof number, Gr. The desirable flow field was found to go unstable in the range of $Gr = 14000–14500$ after extrapolating the results to finer meshes. While for this reactor configuration the flow goes unstable to an axisymmetric mode, there are mode 1 and mode 2 instabilities that go unstable at slightly higher values of Gr.

We have shown that determining the linear stability of steady state solutions arising from the discretization of three-dimensional incompressible flow PDEs is possible. We have also demonstrated the potential impact by locating a flow instability in an engineering system that can be used to interpret certain experimental results and guide the design of next generation reactors.

As mentioned at the end of Section 1, several outstanding issues need to be addressed so that large-scale linear stability analysis is employed on a regular basis by the analyst and design engineer. These include an improved understanding of the role of the error made in approximating the steady state upon the linear stability analysis; improved preconditioners to reduce the cost of the inner iteration during the eigensolver; and the ability to rigorously verify that the right-most eigenvalue has been computed.

ACKNOWLEDGMENTS

We would like to thank Beth Burroughs, David Day, Louis Romero, John Shadid and Ray Tuminaro for helpful discussions. This work was in part supported by the United States Department of Energy Applied Mathematics Program.

REFERENCES

1. Fortin A, Jardak M, Gervais JJ, Pierre R. Localization of Hopf bifurcations in fluid flow problems. *International Journal for Numerical Methods in Fluids* 1997; **24**(11): 1185–1210.
2. Gervais JJ, Lemelin D, Pierre R. Some experiments with stability analysis of discrete incompressible flows in the lid-driven cavity. *International Journal for Numerical Methods in Fluids* 1997; **24**(11): 1185–1210.
3. Morzynski M, Afanasiev K, Thiele F. Solution of the eigenvalue problems resulting from global non-parallel flow stability analysis. *Computing Methods in Applied Mechanics and Engineering* 1999; **169**: 161–176.
4. Lust K. Numerical bifurcation analysis of periodic solutions of partial differential equations. PhD thesis, Katholieke Universitet, Leuven, Belgium, 1997.
5. Shroff GM, Keller HB. Stabilization of unstable projections: the recursive projection method. *SIAM Journal of Numerical Analysis* 1993; **30**(4): 1099–1120.
6. Evans G, Greif R. A numerical model of the flow and heat transfer in a rotating disk chemical vapor deposition reactor. *Journal of Heat Transfer, ASME* 1987; **109**: 928–935.
7. Kieda S, Fotiadis DI, Jensen KF. Transport phenomena in vertical reactors for metalorganic vapor phase epitaxy. *Journal of Crystal Growth* 1990; **102**: 441–470.
8. Weber C, van Opdorp C, de Keijser M. Modeling of gas-flow patterns in a symmetric vertical vapor-phase-epitaxy reactor allowing asymmetric solutions. *Journal of Applied Physics* 1990; **67**: 2109–2118.
9. Breiland WG, Evans GH. Design and verification of nearly ideal flow and heat transfer in a rotating disk chemical vapor deposition reactor. *Journal of the Electrochemical Society* 1991; **138**: 1806–1816.
10. Salinger AG, Lehoucq R, Romero L. Stability analysis of large-scale incompressible flow calculations on massively parallel computers. In *Proceedings of ISCFD'99*. Volume 9 of *Computational Fluid Dynamics Journal*, Rath H, Yabe T (eds), 2000.
11. Blacker TD, Benzley S, Jankovich S, Kerr R, Kraftcheck J, Kerr R, Knupp P, Leland R, Melander D, Meyers R, Mitchell S, Shepard J, Tautges T, White D. *CUBIT Mesh Generation Environment Users Manual Volume 1*. Sandia National Laboratories: Albuquerque, NM, 1999. Revised.
12. Hendrickson B, Leland R. The Chaco User's Guide: Version 2.0. Technical Report SAND94-2692, Sandia National Laboratories, Albuquerque, NM, 1995.

13. Shadid JN. A fully-coupled Newton–Krylov solution method for parallel unstructured finite element fluid flow, heat and mass transport. *Internatioanl Journal of Computational Fluid Dynamics* 1999; **12**: 199–211.
14. Hughes JR, Franca L P, Hulbert GM. A new finite element formulation for computational fluid dynamics: VIII. The Galerkin/least-squares method for advective–diffusive equations. *Computer Methods in Applied Mechanics and Engineering* 1989; **73**: 173–189.
15. Shadid JN, Tuminaro RS, Walker HF. An inexact Newton method for fully coupled solution of the Navier–Stokes equations with heat and mass transport. *Journal of Computational Physics* 1997; **137**: 155–185.
16. Salinger AG, Shadid JN, Hutchinson SA, Hennigan GL, Devine KD, Moffat HK. Massively parallel computation of 3D flow and reactions in chemical vapor deposition reactors. Technical Report SAND97-3092, Sandia National Laboratories, Albuquerque, NM, 1997.
17. Salinger AG, Shadid J N, Hutchinson SA, Hennigan GL, Devine KD, Moffat HK. Analysis of gallium arsenide deposition in a horizontal chemical vapor deposition reactor using massively parallel computations. *Journal of Crystal Growth* 1999; **203**: 516–533.
18. Maschhoff KJ, Sorensen DC. P_ARPACK: an efficient portable large scale eigenvalue package of distributed memory parallel architectures. In *Applied Parallel Computing in Industrial Problems and Optimization*. Volume 1184 of *Lecture Notes in Computer Science*, Wasniewski J, Dongarra J, Madsen K, Olesen D (eds). Springer: Berlin, 1996.
19. Lehoucq RB, Sorensen DC, Yang C. *ARPACK User's Guide: Solution of Large Scale Eigenvalue Problems with Implicitly Restarted Arnoldi Methods*. SIAM: Phildelphia, PA, 1998.
20. Hutchinson SA, Prevost LV, Tuminaro RS, Shadid JN. *Aztec Users Guide: Version 2.0 Technical Report*. Sandia National Laboratories: Albuquerque, NM, 1998.
21. Mattson TG, Henry G. An overview of the Intel TFLOPS super-computer. *Intel Technology Journal* 1998; **1**: 0.
22. Meerbergen K, Spence A, Roose D. Shift-invert and Cayley transforms for the detection of rightmost eigenvalue of nonsysmmetric matrices. *BIT* 1994; **34**: 409–423.
23. Saad Y. *Iterative Methods for Sparse Linear Systems*. PWS: Boston, MA, 1996.
24. Greenbaum A. *Iterative Methods for Solving Linear Systems*. SIAM: Philadelphia, PA, 1997.
25. Grimes RG, Lewis JG, Simon HD. A shifted block Lanczos algorithm for solving sparse symmetric generalized eigenproblems. *SIAM Journal of Matrix Analysis and Applications* 1994; **15**(1): 228–272.
26. Meerbergen K, Spence A. Implicitly restarted Arnoldi with purification for the shift–invert transformation. *Mathematics of Computation* 1997; **218**: 667–689.

