Large-Scale Eigenvalue Calculations for Stability Analysis of Steady Flows on Massively Parallel Computers*

Richard B. Lehoucq and Andrew G. Salinger†

April 25, 2000

Abstract

We present an approach for determining the linear stability of steady-states of PDEs on massively parallel computers. Linearizing the transient behavior around a steady-state solution leads to an eigenvalue problem. The eigenvalues with 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 3D 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.

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

*This work was in part supported by the United States Department of Energy Applied Mathematics Program.

†Massively Parallel Computer Research Center, Sandia National Laboratories, Albuquerque, NM 87185-1110.

are demonstrating the ability to simulate three dimensional fluid flow where the systems size is $\mathcal{O}(10^6 - 10^7)$. However, the linear hydrodynamic stability of flows for these latter systems 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 its 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 a robust solver and then can be tracked using continuation techniques. Each 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 [1, 2, 3, 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. We are 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 3D models.

We present a large scale eigenvalue algorithm that allows us to determine the linear stability of a representative problem of 3D incompressible flow of heat transfer in a rotating disk reactor. While the steady flow for this application is axisymmetric and can be computed with a 2D model, the stability of the flow to 3D disturbances is needed to confidently use the results to design reactors. In addition, axisymmetric modes located with the 3D calculation can be verified against the more routine 2D 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 up to 4 million. Our algorithm identified a critical Grashof number for a Hopf bifurcation, above which the reactor exhibits undesirable flow behavior.

We believe that 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 nonlinear steady-state Naiver-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 report.

Our paper is organized as follows. Section 2 discusses model problem of the 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 3D flow and heat transfer problem. A common approach to investigating the behavior of such a nonlinear 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 moti-

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

vates 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 can 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 very desirable growth conditions.

The rotating disk reactor is known from experiments and calculations to exhibit flow instabilities. This includes the formation of stable yet undesirable re-circulation cells [6, 7, 8] as well as unsteady flows [9]. The steady-state behavior of this system can be uncovered by bifurcation analysis of steady-state flows [7, 10]. While these solutions are axisymmetric and require just 2D 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 3D model.

To provide the most general results, we study just the fluid flow and heat transfer model (no mass transfer or reactions) and look at dimensionless numbers, which 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, $Re_{rot} = (\Omega R^2)/\nu = 83.77$ and the Prandlt number $Pr = \nu/\alpha = 1.0$ 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 a 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 $Gr = (g\beta T R^3)/\nu^2 = 15000$, 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 convective and conduction terms is solved. The equations are shown in Table 1 and include the time dependent terms, which are important for the formulation of the stability (eigenvalue) calculation. The discretized system can be expressed in the form $\mathbf{f}(\mathbf{y}, \dot{\mathbf{y}}, \tau) = \mathbf{0}$ where, $\mathbf{y}$ represents the vector of nodal unknows, $\dot{\mathbf{y}}$ 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.

Table 1: The governing PDE's for incompressible flow with heat transfer are shown in dimension-less form, including the Navier-Stokes equations with the Boussinesq approximation, the continuity equation, and a heat balance.

Momentum	$\frac{dv}{dt} + v \cdot \nabla v = -\nabla P + \nabla^2 v + GrTe_z$
Total Mass	$\nabla \cdot v = 0$
Thermal Energy	$\frac{dT}{dt} + v \cdot \nabla T = \frac{1}{Pr}\nabla^2 T$

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 of this infrastructure along with a brief discussion on the software tools used. We refer the reader to [11] for a recent paper describing the CFD simulator employed. This procedure can be summarized in these 3 steps, with details to follow.

1. The finite element mesh is generated and a graph partitioning tool is employed to distribute the mesh among the processors.

2. A parallel finite element CFD simulator is used to compute a steady-state solution.

3. A parallel eigen-solver determines the linear stability from a linearization of a steady-state solution.

The CUBIT [12] 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 [13], 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 based CFD simulator we used is 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 method [14] (GLS). 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] eigen-solver 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 preconditioners. 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 some of the specific details associated with the representative steady-flow problem discussed previously.

2.

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 seconds 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 on each sub-domain (with one subdomain per processor) an ILU preconditioner is used with a fill-in factor of 7. The fill-in factor is a parameter that allows the preconditioner

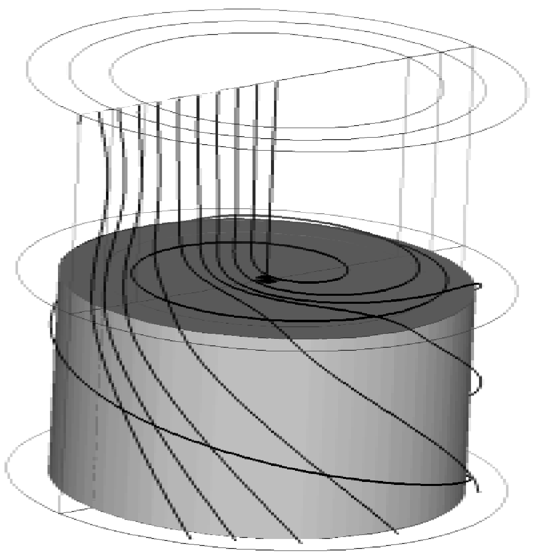

Figure 2: Visualization of the three-dimensional steady-state flow solution. Streamlines enter the top, pass over the heated disk, and leave through the annular region.

to retain more non-zeros than the sparse Jacobian; in this case the incomplete LU factorization process can create up to 7 times as many. An average GMRES solve required 80 iterations and 30 seconds (including the time to construct the preconditioner) to reach a drop in the scaled residual of 10^{-3}. The steady-state solution at $Gr = 15000$ was reached from a trivial initial guess in 7 minutes using 2 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 calculation described in this section.

4 Stability Analysis Calculation

If we linearize the equation

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

about the steady-state $(\mathbf{y}_0, \tau_0)$ to small perturbations $e^{\lambda t}\mathbf{z}$, we obtain the generalized eigenvalue problem $\mathbf{Jz} = \lambda\mathbf{Bz}$ where the Jacobian and mass matrices are $\mathbf{J} = \mathbf{f}_{\mathbf{y}}(\mathbf{y}_0, \mathbf{0}, \tau_0)$ and $\mathbf{B} = -\mathbf{f}_{\dot{\mathbf{y}}}(\mathbf{y}_0, \mathbf{0}, \tau_0)$ respectively. We denote the order of the matrices $\mathbf{J}$ and $\mathbf{B}$ by n. Because we use a GLS discretization scheme, the generalized eigenvalue problem can be written as

$$\begin{pmatrix} \mathbf{L} & -\mathbf{C} \\ \mathbf{C}^T + \mathbf{G} & \mathbf{K} \end{pmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{p} \end{bmatrix} = \begin{pmatrix} \mathbf{M} & \mathbf{0} \\ \mathbf{N} & \mathbf{0} \end{pmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{p} \end{bmatrix} \lambda \tag{1}$$

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 up-winded mass matrix, $\mathbf{L}$ is the sum of the discretized diffusion, nonlinear 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 $\text{Real}(\lambda) < 0$ for all the eigenvalues of (1). Hence, computing approximations to the right-most eigenvalues 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 (1) into the standard eigenvalue problem

$$\mathbf{T}_s\mathbf{z} = (\mathbf{J} - \sigma\mathbf{B})^{-1}\mathbf{Bz} = \gamma\mathbf{z}, \quad \gamma = \frac{1}{\lambda - \sigma}. \tag{2}$$

The above formulation maps the infinite eigenvalues of (1) (arising from singular $\mathbf{B}$) to zero. By selecting the pole σ near the imaginary axis, the right-most eigenvalues are mapped by $\mathbf{T}_s$ into 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. Although a natural choice is to select a zero pole, the resulting transformation might miss a Hopf bifurcation (complex conjugate pair of eigenvalues that cross into the right half of the complex plane). This occurs, for instance, when the distance to the Hopf bifurcation is greater than the distance to other (perhaps stable) eigenvalues of (1). The paper [23] discusses these issues in some detail.

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 [24, 25] depends strongly upon the spectral condition number.

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

$$\mathbf{T}_c\mathbf{z} = (\mathbf{J} - \sigma\mathbf{B})^{-1}(\mathbf{J} - \mu\mathbf{B})\mathbf{z} = \gamma\mathbf{z}, \quad \gamma = \frac{\lambda - \mu}{\lambda - \sigma}. \tag{3}$$

We call μ the zero of the Cayley transform. In contrast to shift-invert transform, the Cayley transform maps any eigenvalues of (1) 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 (1) and choose $\mu > \sigma$, 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 ten) spectral condition number.

The last two paragraphs describe a delicate balancing act. On the one hand, the ability to compute the right-most eigenvalues (λ's) requires that the Cayley transformation map these values to γ's 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 is negatively impacted 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 shift-invert spectral transformation 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

$$(\mathbf{J} - \sigma\mathbf{B})\mathbf{v} = (\mathbf{J} - \mu\mathbf{B})\mathbf{x} \tag{4}$$

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

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 $\mathbf{Jz} = \mathbf{Mz}\lambda$ using the Cayley transformation and IRAM.

bounded (via a careful choice of σ and μ) implies that the Cayley system results in a better conditioned set of linear equations.

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 `pdneupd` to implement the Cayley transformation. We refer the reader to [19] for full details about the software and underlying algorithm.

Figure 3 lists the scheme used for computing several (say k) right-most eigenvalues of (1). 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 eigenvalues (θ's approximating γ's) of the m by m Hessenberg matrix are mapped back to the system defined by (1) via the inverse Cayley transformation resulting in approximations $\hat{\lambda}$'s. The eigensolver is terminated when these k rightmost 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 Arnoldi'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 ten.

We remark that there are two iterations—an outer and an 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 (4) arising from applying $\mathbf{T}_c$. We use a GMRES iteration for solving this linear set of equations. The next section discuss 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 the Arnoldi's method, how accurately they will be calculated, and how expensive the calculation will be. The major consideration is the size of $|\gamma|$ 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 to $\sigma - \text{Real}(\lambda_i)$.

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 fixed values of $\sigma = 20$ and $\mu = 80$ and four imaginary portions of λ. Note that as the real part of λ decreases, $|\gamma|$ approaches one. The σ and μ values in this plot map the real eigenvalues in the range of $-40 < \text{Real}(\lambda) < 20$ 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 $\text{Real}(\lambda) < -40$, 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 $2\sigma - \mu < \text{Real}(\lambda_i) < \sigma$.

Figure 4 also indicates that eigenvalues with large imaginary parts are mapped to small $|\gamma|$. Therefore it is difficult to compute approximations to

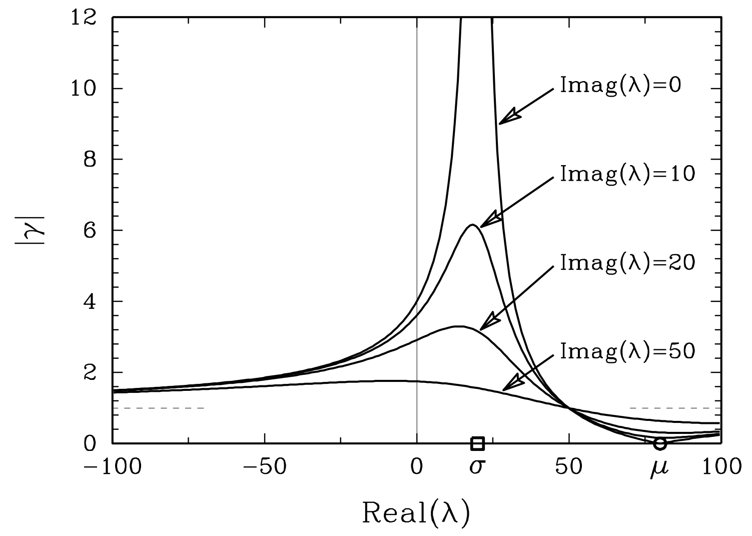

Figure 4: Plot of the transformation of the eigenvalues in the physical system to those in the Cayley transformed system. The magnitude of the transformed eigenvalue is plotted against the real part of λ for three different imaginary contributions to λ.

eigenvalues with large imaginary parts. 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 tradeoff 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.

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 very large 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 three right-most eigenvalues and eigenvectors versus the convergence tolerance used for the linear solver. The eigensolver residual error is defined as

$$\frac{||\mathbf{J}\hat{\mathbf{z}} - \hat{\lambda}\mathbf{B}\hat{\mathbf{z}}||}{||\mathbf{B}\hat{\mathbf{z}}||} \tag{5}$$

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 (5) is independent of the scaling of the data. The linear solver uses the criterion

$$\frac{||\mathbf{b} - \mathbf{A}\mathbf{x}_j||}{||\mathbf{B}\mathbf{v}||} < \eta \tag{6}$$

where $\mathbf{A} = \mathbf{J} - \sigma\mathbf{B}$ and $\mathbf{b} = (\mathbf{J} - \mu\mathbf{B})\mathbf{v}$ from (4), 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 i-th ($1 \leq i \leq m$) 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 (5). The residual error of the rightmost eigenvalue pair (denoted by the solid line) stops decreasing $\eta \approx 10^{-3}$. The residual error of the next two eigenvalues stop 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 that $\max(|\gamma|)$ 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

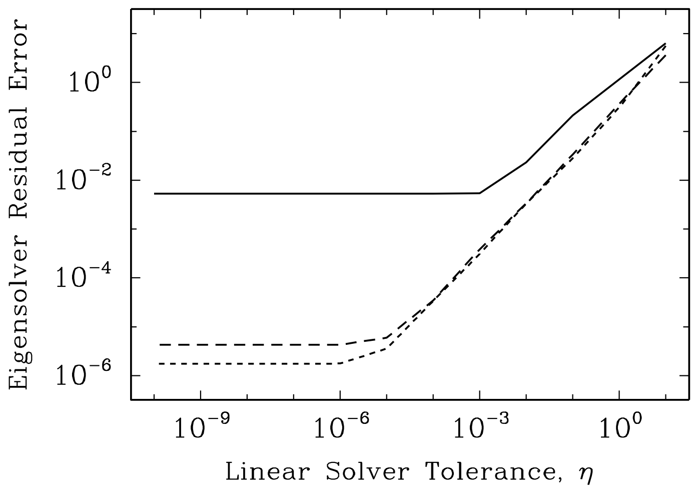

Figure 5: The error in the eigenvalue calculation for the three rightmost eigenvalues is shown to be a function of the acceptance criterion of the iterative linear solver. The eigensolver residual error and linear solver tolerance is given in (5) and (6).

iterations needed for a solve. However, as σ is increased so do the residuals (5). For this problem, a choice of $\sigma = 20$ provides a balance between efficiency and accuracy. The trends seen as a function of σ point to a remedy for systems where the preconditioned linear solver is not able 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 (5) is sufficiently small for the rightmost eigenvalues.

5.2 Using Aztec

Another issue associated with preconditioned 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 ILUT preconditioner with considerable fill-in was selected so that the preconditioner required almost 4 times more memory than the matrix itself. The preconditioner is computed once and reused for each iteration of Arnoldi's method needed by the eigensolver.

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 pre-calculated preconditioner) was recorded with the number of processors being varied from 100 to 1000. The message of this

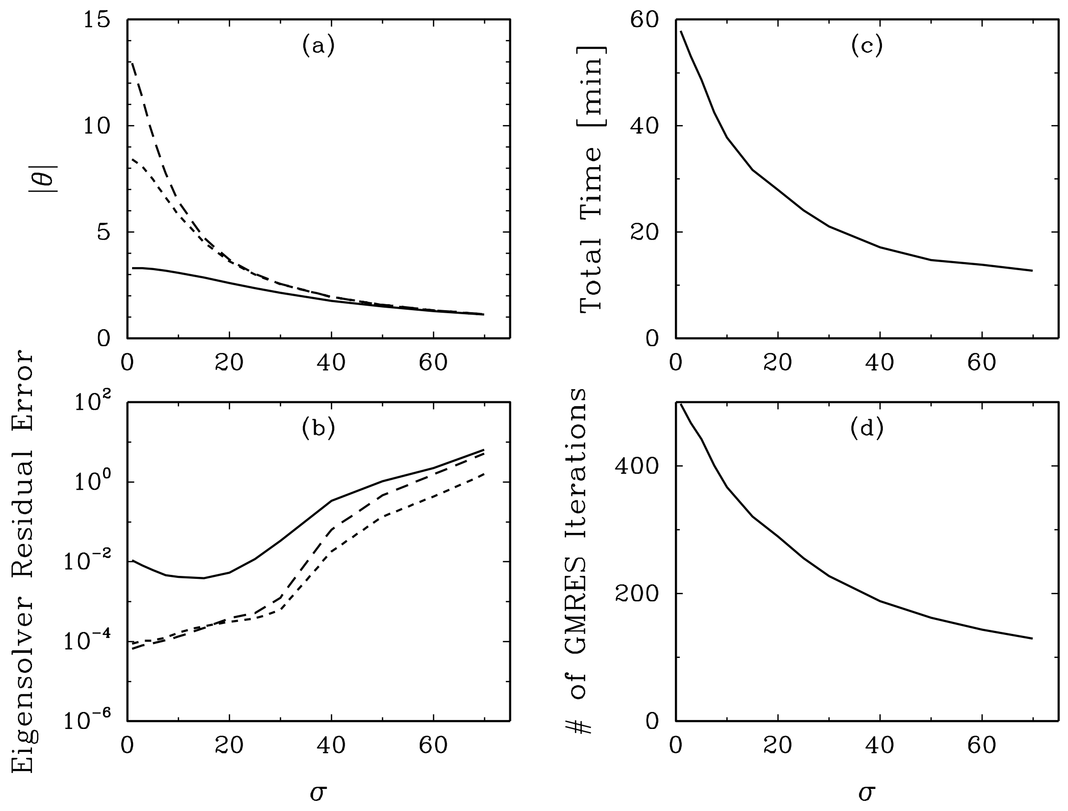

Figure 6: The model eigenvalue calculation is repeated for several values of the σ parameter in the Cayley transformation. The effect of σ on (a) the magnitude of the three largest complex θ pairs, (b) the residual errors (5) associated with the three largest complex θ pairs in the transformed system, (c) the time for the calculation, and (d) the average number of GMRES iterations needed for each of the 24 linear solve are shown.

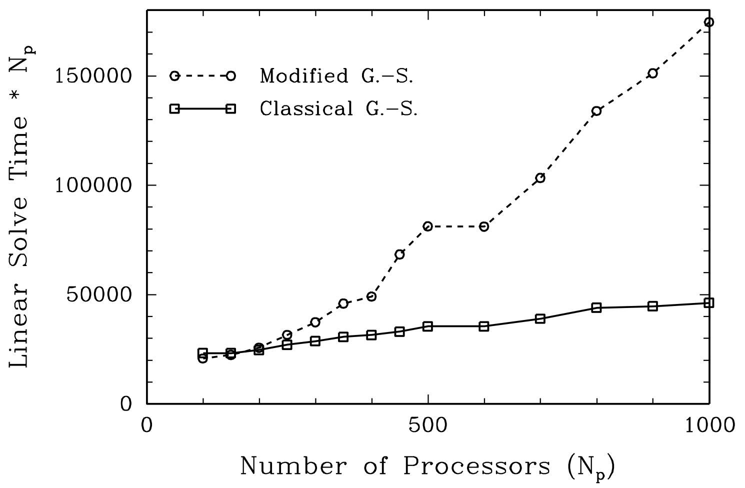

Figure 7: The parallel efficiency of two stable orthogonalization schemes in the GMRES algorithm is compared. The extra communications of the Modified Gram-Schmidt (MGS) approach scale poorly with number of processors, while the extra operations of the 2-step Classical Gram-Schmidt (CGS) approach scale well.

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 classical Gram-Schmidt (CGS) scheme scales much better than the modified Gram-Schmidt (MGS) scheme. It should be pointed out that the inter-processor communication rate of the Sandia-Intel Tflop computer is very fast compared to 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.

Table 2: Mesh resolution studies at $Gr = 15000$ on the six eigenvalues with largest real parts. The coarse mesh results would indicate a stable solution, but the finest mesh shows that two pairs of eigenvalues have positive real parts.

Number of Unknowns	First Eigenvalue	Second Eigenvalue	Third Eigenvalue
.25 Million	$-0.08 \pm 25.33\imath$	$-0.49 \pm 9.63\imath$	$-1.44 \pm 5.96\imath$
.50 Million	$0.35 \pm 25.16\imath$	$-0.05 \pm 9.50\imath$	$-1.13 \pm 5.91\imath$
1 Million	$0.57 \pm 25.06\imath$	$0.21 \pm 9.36\imath$	$-0.98 \pm 6.01\imath$
2 Million	$0.73 \pm 25.02\imath$	$0.39 \pm 9.31\imath$	$-0.85 \pm 6.03\imath$
4 Million	$0.84 \pm 24.94\imath$	$0.50 \pm 9.22\imath$	$-0.78 \pm 6.08\imath$
2D mesh .50 Million	$1.06 \pm 24.69\imath$		

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 PDE model. The results of such a study are shown in Table 2. 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 250 thousand to 4 million unknowns. Since the first eigenmode was determined through visualization to be axisymmetric, a final calculation on a very fine 2D axisymmetric mesh of 0.5 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 (which is used in all 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 meshes give 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.

There was no attempt to calculate a convergence rate with mesh since the data does not fall on smooth a 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 nonlinear 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 the calculated eigenvalues. These results imply that very fine meshes and very accurate linear and nonlinear solves may be needed to pinpoint the exact parameter value of a Hopf bifurcation in 3D 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 finer meshes are only needed to locate the parameter values to a higher degree of accuracy.

Some more details on the 4 million unknown calculation 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 hours of CPU time on 1024 processors. The leading eigenvalues of the sparse matrix with over 500 million nonzero elements where calculated in under 5 hours, where each linear solve required about 12 minutes. The linear solves used row-sum scaling, the ILUT preconditioner with a fill-in factor of 8, and required an average of 825 iterations of (un-restarted) GMRES to converge.

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. At the moment, there are 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 eigenvalue problem [27] where at the cost of computing a sparse direct factorization, reliability of the eigen-solver can be determined.

6 Reactor Analysis

In this section we apply the linear stability analysis capability that has been presented 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 8 shows how the six eigenvalues that have largest real part at $Gr = 15000$ evolve from $Gr = 10000$ to $Gr = 16000$. 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 2, 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 get damped 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 the 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 bifurcation is a mode 2 symmetry breaking, where there is up-flow in two quadrants of the disk and down-flow in the two others. Again, this flow structure precesses around reactor in time. It is interesting that the modes 0, 1, and 2 symmetry breakings occur at 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 3D capability. And since Cartesian coordinates were used to model the system, the fact that the solution was axisymmetric did not simplify the calculations in any way.

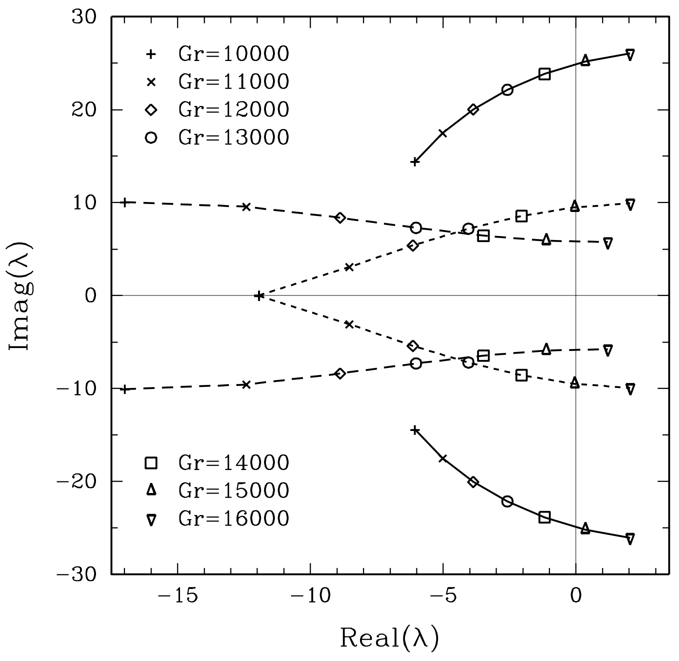

Figure 8: The tracking of the six largest eigenvalues as a function of parameter indicate a Hopf bifurcation near 14800.

7 Summary and Conclusions

A massively parallel code for calculating steady-states of incompressible and reacting flows (MPSalsa) has been linked to a library (using 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 3D 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 sophisticated linear algebra 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 nonzero 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 3D incompressible flow PDE's 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 the next generation reactors.

As mentioned at the end of the introduction, 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 eigen-solve; and the ability to rigorously verify that the right-most eigenvalue has been computed.

Acknowledgements We would like to thank Beth Burroughs, David Day, Louis Romero, John Shadid and Ray Tuminaro for helpful discussions.

References

[1] A. Fortin, M. Jardak, J.J. Gervais, and R. Pierre. Localization of Hopf bifurcations in fluid flow problems. *Int. J. Numer Methods Fluids*, 24(11):1185–1210, 1997.

[2] J.J. Gervais andD. Lemelin and R. Pierre. Some experiments with stability analysis of discrete incompressible flows in the lid-driven cavity. *Int. J. Numer Methods Fluids*, 24(11):1185–1210, 1997.

[3] Marek Morzyński, Konstantin Afanasiev, and Frank Thiele. Solution of the eigenvalue problems resulting from global non-parallel flow stability analysis. *Computer Methods in Applied Mechanics and Engineering*, 169:161–176, 1999.

[4] Kurt Lust. *Numerical Bifurcation Analysis of Periodic Solutions of Partial Differential Equations*. PhD thesis, Katholieke Universitet Leuven, Leuven, Belguim, December 1997.

[5] Guatam M. Shroff and Herbert B. Keller. Stabilization of unstable projections: The recursive projection method. *SIAM J. Numerical Analysis*, 30(4):1099–1120, August 1993.

[6] G. Evans and R. Greif. A numerical model of the flow and heat transfer in a rotating disk chemical vapor deposition reactor. *J. Heat Transfer ASME*, 109:928–935, 1987.

[7] S. Kieda D.I. Fotiadis and K.F. Jensen. Transport phenomena in vertical reactors for metalorganic vapor phase epitaxy. *J. Crystal Growth*, 102:441–470, 1990.

[8] C. Weber, C. van Opdorp, and M. de Keijser. Modeling of gas-flow patterns in a symmetric vertical vapor-phase-epitaxy reactor allowing asymmetric solutions. *J. Appl. Phys.*, 67:2109–2118, 1990.

[9] W. G. Breiland and Greg H. Evans. Design and verification of nearly ideal flow and heat transfer in a rotating disk chemical vapor deposition reactor. *Journal of the Electrochemical Society*, 138:1806–1816, 1991.

[10] L. A. Romero and A. G. Salinger. Stability analysis of flow in the rotating disk reactor. *in preparation*, 1999.

[11] J.N. Shadid. A fully-coupled Newton-Krylov solution method for parallel unstructured finite element fluid flow, heat and mass transport. *IJCFD*, 12:199–211, 1999.

[12] T.D. Blacker, S. Benzley, S. Jankovich, R. Kerr, J. Kraftcheck, R. Kerr, P. Knupp, R. Leland, D. Melander, R. Meyers, S. Mitchell, J.Shepard, T.Tautges, and D. White. *CUBIT Mesh Generation Environment Users Manual Volume 1.* Sandia National Laboratories, Albuquerque, NM 87185, 1999. revised.

[13] B. Hendrickson and R. Leland. The Chaco user's guide: Version 2.0. Technical Report SAND94–2692, Sandia National Labs, Albuquerque, NM, June 1995.

[14] J. R. Hughes, L. P. Franca, and G. M. Hulbert. A new finite element formulation for computational fluid dynamics: VIII. the Galerkin/least-squares method for advective-diffusive equations. *Comp. Meth. App. Mech. and Eng.*, 73:173–189, 1989.

[15] J.N. Shadid, R.S. Tuminaro, and H.F. Walker. An inexact Newton method for fully coupled solution of the Navier-Stokes equations with heat and mass transport. *Journal of Computational Physics*, 137:155–185, 1997.

[16] A. G. Salinger, J. N. Shadid, S. A. Hutchinson, G. L. Hennigan, K. D. Devine, and H. K. Moffat. Massively parallel computation of 3d flow and reactions in chemical vapor deposition reactors. Technical Report SAND97–3092, Sandia National Laboratories, Albuquerque, NM, 1997.

[17] A. G. Salinger, J. N. Shadid, S. A. Hutchinson, G. L. Hennigan, K. D. Devine, and H. K. Moffat. Analysis of gallium arsenide deposition in a horizontal chemical vapor deposition reactor using massively parallel computations. *J. Crystal Growth*, 203:516–533, 1999.

[18] K. J. Maschhoff and D. C. Sorensen. P_ARPACK: An efficient portable large scale eigenvalue package for distributed memory parallel architectures. In Jerzy Wasniewski, Jack Dongarra, Kaj Madsen, and Dorte Olesen, editors, *Applied Parallel Computing in Industrial Problems and Optimization*, volume 1184 of *Lecture Notes in Computer Science*, Berlin, 1996. Springer–Verlag.

[19] R. B. Lehoucq, D. C. Sorensen, and C. Yang. *ARPACK USERS GUIDE: Solution of Large Scale Eigenvalue Problems with Implicitly Restarted Arnoldi Methods.* SIAM, Phildelphia, PA, 1998.

[20] S. A. Hutchinson, L. V. Prevost, R. S. Tuminaro, and J. N. Shadid. Aztec user's guide: Version 2.0. Technical report, Sandia National Laboratories, Albuquerque, NM, 1998.

[21] T. G. Mattson and G. Henry. An overview of the Intel TFLOPS supercomputer. *Intel Technology Journal*, 1, 1998.

[22] K. Meerbergen, A. Spence, and D. Roose. Shift-invert and Cayley transforms for the detection of rightmost eigenvalues of nonsymmetric matrices. *BIT*, 34:409–423, 1994.

[23] K. A. Cliffe, T. J. Garratt, and A. Spence. Eigenvalues of the discretized Navier-Stokes equation with application to the detection of Hopf bifurcations. *Advances in Computational Mathematics*, 1:337–356, 1993.

[24] Y. Saad. *Iterative Methods for Sparse Linear Systems.* PWS, Boston, MA, 1996.

[25] Anne Greenbaum. *Iterative Methods for Solving Linear Systems.* SIAM, Philadelphia, PA., 1997.

[26] Karl Meerbergen and Alastair Spence. Implicitly restarted Arnoldi with purification for the shift–invert transformation. *Mathematics of Computation*, 218:667–689, 1997.

[27] R. G. Grimes, J. G. Lewis, and H. D. Simon. A shifted block Lanczos algorithm for solving sparse symmetric generalized eigenproblems. *SIAM J. Matrix Analysis and Applications*, 15(1):228–272, January 1994.