o appear in: Proceedings of the 11th International Conference on Numerical Methods in luid Dynamics, edited by D.L. Dwoyer, M.Y. Hussaini, and R.G. Voigt (Springer-Verlag, 1989)

STEADY-STATE SOLVING VIA STOKES PRECONDITIONING; RECURSION RELATIONS FOR ELLIPTIC OPERATORS

Laurette S. Tuckerman
Center for Nonlinear Dynamics and Department of Physics
University of Texas, Austin Texas 78712

INTRODUCTION

Recent experiments on Rayleigh-Benard convection in a cylindrical geometry have yielded a rich variety of results concerning wavelength selection, breaking of axisymmetry, and phase turbulence [1,2]. Motivated by these experiments, we have developed a program which numerically simulates the full three-dimensional Boussinesq equations in a cylinder of large aspect ratio.

In order to study the bifurcations undergone by the system, we have brought to bear the full arsenal of tools for studying dynamical systems: time-dependent integration, linear stability analysis, and steady-state continuation. In the past, these tasks have generally been accomplished by using entirely separate programs. However, in [3,4] it was shown that a time-dependent code could easily be adapted to perform linear stability analysis, by iterating the equations linearized about a numerically computed steady state, thus calculating the most unstable (or least stable) eigenvectors via the power method. Here we will show that a time-dependent code can be given the additional capability for direct calculation of stable and unstable steady states via Newton's method.

Our steady-state solver relies on a fast and implicit method for solving the linear Stokes problem, i.e. for inverting the elliptic operators arising from time-stepping of the viscous terms, enforcing incompressibility, and imposing boundary conditions. The need for a fast implicit Stokes solver is not limited to the steady-state code: it is well known that, when integrating the incompressible Navier-Stokes or Boussinesq equations, the linear terms present the most stringent time-stepping criterion, and should therefore be treated implicitly. In the next section, we will present a general result concerning the existence of recursion relations for elliptic operators, and we will demonstrate its application to the solution of Poisson's equation in a cylindrical geometry. The third section will describe the steady-state solver.

INVERSION OF ELLIPTIC OPERATORS VIA RECURSION RELATIONS

We use a pseudospectral method [5,6,7], in which functions are represented by:

$$f(r,\theta,z) = \sum_{k=0}^{K} \sum_{m=0}^{M} \sum_{n=0}^{N} f_{kmn}\, T_k(r)\, e^{im\theta}\, T_n(z) \qquad (1)$$

where T_k, T_n are Chebyshev polynomials, and parity restrictions are imposed on $k+m$. Numerical spectral methods, like analytic methods, are difficult to adapt to nonperiodic geometries. Here rigid boundaries at the top, bottom, and sidewalls must be added to the complication of curvilinear coordinates and coordinate

singularities. A further challenge arises from the large aspect ratio of the cylinder: the algorithm must be economical in the number K of gridpoints or functions in the radial direction, a direction which is neither periodic (like θ) nor Cartesian (like z).

At the core of our algorithm is a fast solver for elliptic equations such as:

$$\nabla^2 f = g \tag{2}$$

A well known recursion relation [5,7] can be used to reduce the upper triangular matrix corresponding to the (Cartesian) second derivative acting on a Chebyshev series to a tridiagonal diagonally-dominant matrix. Only one spatial direction can be reduced in this way: when two directions are nonperiodic, a Haidvogel-Zang decomposition [7,8] is indicated, in which the (Cartesian-like) z direction is reduced to tridiagonal form and the r direction treated by eigenvector-eigenvalue decomposition, leading to an operation count of $O(3KMN) + O(2K^2MN)$. However, since $K >> N$, what is needed is a recursion relation for the *radial* direction instead.

We have shown in [9] that the standard recursion relation is only one instance of a much larger class of matrices which can be reduced to banded form. The relevant result is:

THEOREM: Let R be an upper triangular matrix of the form:

$$R(k,l) = \left\{ \begin{array}{ll} \sum_{j=1}^{J} S(k,j)\, T(j,l) & 1 \le k < l \le K \\ R(k,k) & 1 \le k = l \le K \\ 0 & \text{otherwise} \end{array} \right\} \tag{3}$$

with $J < K$. Suppose that the J by J matrices $\hat{S}^k$ defined by $\hat{S}^k(i,j) \equiv S(k+i,j)$ for $1 \le i, j \le J$ are invertible for all $k < K-J$ Then there exists an invertible banded matrix B, depending only on S, with J nonzero super-diagonals, such that BR is also banded with J nonzero super-diagonals.

Matrices that differentiate polynomial expansions, or which multiply derivatives by powers of the independent variable (e.g. r) are all of form (3) (see the Appendix of [5]), as are sums of such operators. In these cases S (and T) are monomials, i.e. $S(k,j) = k^{\alpha_j}$, for integers α_j, with $\alpha_i \neq \alpha_j$ if $i \neq j$. Since the monomials k^{α_j} are linearly independent over any range of k, the conditions of the theorem are satisfied. In the tau method, the last rows of the matrix of form (3) are replaced by full rows corresponding to boundary conditions. If only one row is changed (as here) then the bandwidth of BR remains $J+1$ except for the last row. The proofs of these assertions are straightforward [9].

Let us now apply the theorem to the cylindrical Laplacian acting on (1). We define the operators:

$R \equiv r\, \partial_r\, r\, \partial_r$, which is of form (3) and can be reduced by the banded matrix B,

$D_{\theta\theta} \equiv \partial_{\theta\theta}$, which is diagonal with elements $-m^2$, and

$Z \equiv \partial_{zz} = V^{-1} D_{zz} V$, where D_{zz} is diagonal with elements σ_n.

$R, B, D_{\theta\theta}, Z, D_{zz}, V$, and V^{-1} can either be considered to be matrices of order K, M, or N, or extended to order (KMN) via, e.g., $R(k,m,n;\ k',m',n') = R(k,k')\ \delta(m,m')\ \delta(n,n')$.

Equation (2) is then written as:

$$(r^{-2}R + r^{-2}D_{\theta\theta} + Z)\, f = g$$

We first diagonalize in z:

$$(r^{-2}R + r^{-2}D_{\theta\theta} + D_{zz})\, Vf = V g$$

and then multiply through by the matrices B and r^2:

$$A\, Vf \equiv (BR + B\, D_{\theta\theta} + Br^2\, D_{zz})\, Vf = Br^2\, V g \tag{4}$$

For each θ Fourier mode and z eigenfunction, i.e. for each m and n, this can be written as:

$$A_{mn}\,(Vf)_{mn} \equiv \left(BR - Bm^2 + Br^2\,\sigma_n\right)(Vf)_{mn} = Br^2\;(Vg)_{mn}$$

All boundary conditions are imposed via the tau method: those in z are built into the matrix Z (i.e. into V, V^{-1}, and D_{zz}) while radial boundary conditions are inserted into the matrices A_{mn}. Since B, BR, and r^2 are all tridiagonal and diagonally dominant, the $K \times K$ matrices A_{mn} are pentadiagonal, with an extra full boundary row, and can be inverted via LU decomposition. The requirements of the algorithm are summarized in the following table.

Operator	Storage	Time for multiplication by operator
V and V^{-1}	$2N^2$	$2KMN^2$
Br^2	$5K$	$5KMN$
A^{-1}	$6KMN$	$6KMN$

For $m \geq 2$, regularity conditions at $r = 0$ must be imposed [6,10]. We do this by setting $f = r^2 h$ and defining:

$$R' \equiv r^{-1}\partial_r\, r\,\partial_r\, r^2 = r^{-2} R\, r^2$$

which is also of form (3). Diagonalizing (2) in z and multiplying by the same matrix B, we obtain:

$$A'\,V\,h \equiv \left(BR' + B\,D_{\theta\theta} + Br^2 D_{zz}\right) V\,h = B\,V\,g$$

which is solved in the same way as (4).

The algorithm for solution of the Stokes problem is built around this Poisson solver. The cylindrical vector Laplacian is decoupled into three scalar elliptic operators [6]. The technique outlined above is used to invert operators such as $I - \varepsilon\nabla^2$, as well as ∇^2. Finally, the velocity field is made divergence-free by the influence matrix method [7,11] with full tau correction, generalized to a cylindrical geometry with two nonperiodic directions [10].

STEADY-STATE SOLVER

A steady-state solver presents several advantages over a time-dependent code. The first is speed: root-finding algorithms such as Newton's method converge quadratically, while time-stepping necessarily follows the dynamics, converging linearly to steady states. The second advantage is more fundamental: a steady-state solver can compute unstable steady states, which a time-dependent code cannot (except for the case in which the instability arises via a symmetry-breaking bifurcation [3,4]). In addition, the steady-state solver may be coupled to standard continuation methods for tracking bifurcations in parameter space, and is exempt from critical slowing down.

To describe our steady-state solver, we represent the differential equation by:

$$\partial_t u = Lu + Nu$$

Here L represents the terms (usually linear) in the equation that are to be integrated by an implicit method, and N represents the terms (usually nonlinear) to be integrated explicitly. We wish to solve:

$$Lu + Nu = 0 \qquad (5)$$

Our time-dependent program uses backwards Euler time-stepping for L and forwards Euler time-stepping for N (higher-order time discretizations could also be used):

$$u^{n+1} - u^n = \Delta t\,(Lu^{n+1} + Nu^n)$$

$$u^{n+1} = (I - \Delta t\, L)^{-1}\,(I + \Delta t\, N)\,u^n$$

$$\frac{1}{\Delta t}(u^{n+1} - u^n) = \frac{1}{\Delta t}[\,(I - \Delta t\, L)^{-1}(I + \Delta t\, N) - I\,]\, u^n \qquad (6)$$

$$\equiv E_{\Delta t}\, u^n$$

The crucial observation is that:

$$E_{\Delta t} = \frac{1}{\Delta t}(I - \Delta t\, L)^{-1}\,[\,(I + \Delta t\, N) - (I - \Delta t\, L)\,]$$

$$= (I - \Delta t\, L)^{-1}\,(N + L)$$

so that, as long as $I - \Delta t\, L$ is invertible,

$$E_{\Delta t}\, u = 0 \qquad (7)$$

has the same solutions as (5) for any Δt. The limit $\Delta t \to 0$ yields equation (5), but (7) is numerically more tractable if much larger values of Δt are used, as we shall see below.

We see from (6) that we can act with $E_{\Delta t}$ on a vector by using the time-stepping program. In order to solve (7) by Newton's method, we shall also need the Jacobian $DE_{\Delta t}(U)$ of $E_{\Delta t}$ at U, i.e. the linearization of $E_{\Delta t}$ about U. If L is linear and:

$$Nu = (u\cdot\nabla)\,u$$

then:
$$DN(U)\,u = (U\cdot\nabla)\,u + (u\cdot\nabla)\,U$$

and $DE_{\Delta t}(U)$ is obtained by substituting $DN(U)$ for N in (6). The time-stepping code already computes Nu; with slight modifications it can compute $DN(U)u$ as well. (Indeed, the substitution $N \to DN(U)$ is also precisely that required to perform linear stability analysis [3,4]).

One step of Newton's method consists of solving:

$$DE_{\Delta t}(u^n)\,(u^{n+1} - u^n) = -E_{\Delta t}\, u^n \qquad (8) \quad *$$

for $u^{n+1} - u^n$. The matrix $DE_{\Delta t}(u^n)$ is large and full, necessitating iterative solution of equation (8). However, we can act rapidly with $DE_{\Delta t}(u^n)$ on a vector: the pseudospectral method provides a fast way to multiply by $DN(U)$ in gridspace via Fourier transforms [5,7], while our Poisson/Stokes solver efficiently operates with $(I - \Delta t\, L)^{-1}$. Iterative techniques are able to fully exploit this advantage.

Because $DE_{\Delta t}(u^n)$ is neither symmetric nor definite, iterative solution of (8) is problematic [7,12]. Despite the lack of guarantees, we have experimented with Lanczos and Orthomin methods implemented in the NSPCG (Nonsymmetric Preconditioned Conjugate Gradient) software package [12]. We obtain extremely good results from the biconjugate gradient squared (BCGS) algorithm, a Lanczos-type method.

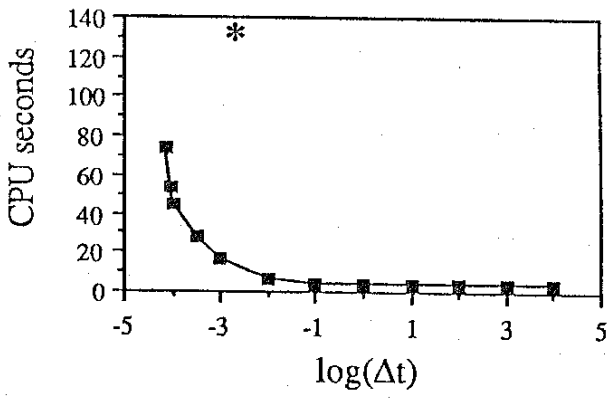
In the figure below, we compare the time needed to solve (7) for different values of Δt on a Cray X-MP/24. The parameters of the problem are: aspect ratio $\Gamma = 5$, resolution $(K,M,N) = (49,0,15)$, and reduced Rayleigh number $\varepsilon = (Ra - Ra_c)/Ra_c = 1.3$ The initial condition for Newton's method is a steady state at $\varepsilon = 1.2$ (other parameters the same). For details see [13].

Solving (7) to a prescribed level of accuracy usually requires four Newton steps, independent of Δt. However, the number of conjugate gradient iterations per Newton step may vary greatly. For $\Delta t > 10^{-1}$, about 35 iterations per Newton step are required, leading to a total CPU time of 4 seconds. For $\Delta t < 10^{-2}$, the number of conjugate gradient iterations begins to climb steeply, until, for $\Delta t = 7\times10^{-5}$, as many as 780 iterations are required for one Newton step. For $\Delta t < 7\times10^{-5}$ (and also for $\Delta t > 10^{5}$) the method breaks down.

Effectively, $(I - \Delta t\, L)^{-1}$ acts as a *preconditioner* for the poorly conditioned matrices $L + DN(u^n)$. For $\Delta t \to 0$, the preconditioning becomes weaker as $(I - \Delta t\, L)^{-1} \to I$, while for $\Delta t \to \infty$, we have $(I - \Delta t\, L)^{-1} \to (-\Delta t\, L)^{-1}$.

* Here, the superscript n refers to Newton iteration and not to time stepping

For purposes of comparison, we note that the time-stepping code, limited by the Courant condition to $\Delta t \leq 2\times10^{-3}$, will converge within the same accuracy to the steady state in 130 CPU seconds, shown as the asterisk in the figure. The combination of quadratic convergence, preconditioning, and biconjugate gradient squared iteration has brought about a drastic reduction in computation time, with almost no additional coding effort.



ACKNOWLEDGMENTS

I thank the Center for Numerical Analysis at the University of Texas at Austin for making the NSPCG package available, and especially Tom Oppe for spending many hours patiently explaining its workings to me. Discussions with D. Barkley are gratefully acknowledged. The author was supported by the NSF Mathematical Sciences Postdoctoral Research Fellowship program and the Office of Naval Research Nonlinear Dynamics Program. Computational resources for this work were provided by the University of Texas System Center for High Performance Computing.

REFERENCES

[1] V. Croquette, P. Le Gal, A. Pocheau, and R. Guglielmetti *Europhys. Lett.* **1** (8) 393 (1986).

[2] V. Steinberg, G. Ahlers, and D.S. Cannell, *Physica Scripta* **T9**, 97 (1985).

[3] L.S. Tuckerman and P.S. Marcus, in *Proceedings of the Ninth Intl. Conf. on Num. Meth. in Fluid Dyn.*, ed. by Soubbaramayer and J.P. Boujot (Springer-Verlag, Berlin, 1985).

[4] P.S. Marcus and L.S. Tuckerman, *J. Fluid Mech.* **185**, 1 (1987) and **185**, 31 (1987).

[5] D. Gottlieb and S.A. Orszag, *Numerical Analysis of Spectral Methods: Theory and Applications* (SIAM Press, Philadelphia, 1977).

[6] A.T. Patera and S.A. Orszag, in *Nonlinear Problems: Present and Future*, ed. by A.R. Bishop, D.K. Campbell, and B. Nicolaenko (North-Holland, Amsterdam, 1982).

[7] C. Canuto, M.Y. Hussaini, A. Quateroni, and T.A. Zang, *Spectral Methods in Fluid Dynamics* (Springer-Verlag, Berlin, 1988).

[8] D.B. Haidvogel and T.A. Zang, *J. Comp. Phys.* **30**, 167 (1979).

[9] L.S. Tuckerman, submitted to *J. Comp. Phys.*

[10] L.S. Tuckerman, *J. Comp. Phys.*, in press.

[11] L. Kleiser and U. Schumann, in *Proc. of the Third GAMM Conf. on Num. Meth. in Fluid Mech.*, ed. by E.H. Hirschel (Vieweg and Sohn, Braunschweig, 1980).

[12] T.C. Oppe, W.D. Joubert, and D.R. Kincaid, Publ. #216 of the Center for Numerical Analysis, University of Texas at Austin.

[13] L.S. Tuckerman and D. Barkley, *Phys. Rev. Lett.*, **61** (1988).