INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids 2007; **55**:283–306
Published online 14 March 2007 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/fld.1460

Adapting simulation codes to compute unstable solutions and bifurcation behaviour

Beau B. Grande[1], Simon J. Tavener[2,*,†] and James W. Thomas[2]

[1]*Stevensville, MD, U.S.A.*
[2]*Department of Mathematics, Colorado State University, Fort Collins, CO 80523, U.S.A.*

SUMMARY

Simulation codes for solving large systems of ordinary differential equations suffer from the disadvantage that bifurcation-theoretic results about the underlying dynamical system cannot be obtained from them easily, if at all. Bifurcation behaviour typically can be inferred only after significant computational effort, and even then the exact location and nature of the bifurcation cannot always be determined definitively. By incorporating relatively minor changes to an existing simulation code for the Taylor–Couette problem, specifically, by implementing the Newton–Picard method, we have developed a computational structure that enables us to overcome some of the inherent limitations of the simulation code and begin to perform bifurcation-theoretic tasks. While a complete bifurcation picture was not developed, three distinct solution branches of the Taylor–Couette problem were analysed. These branches exhibit a wide variety of behaviours, including Hopf bifurcation points, symmetry-breaking bifurcation points, turning points and bifurcation to motion on a torus. Unstable equilibrium and time-periodic solutions were also computed.

Received 30 October 2005; Revised 14 January 2007; Accepted 16 January 2007

KEY WORDS: unstable solutions; periodic orbits; Taylor–Couette flow

1. INTRODUCTION

Consider a parameter-dependent non-linear dynamical system described by a system of ordinary differential equations

$$\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t} = \mathbf{f}(\mathbf{u}, \lambda) \tag{1}$$

*Correspondence to: Simon J. Tavener, Department of Mathematics, Colorado State University, Fort Collins, CO 80523, U.S.A.
†E-mail: tavener@math.colostate.edu

where $\mathbf{u} \in \mathbb{R}^N$ and $\mathbf{f} : \mathbb{R}^N \times \mathbb{R} \to \mathbb{R}^N$. For the particular application we have in mind, Equation (1) results from the spatial discretization of a system of time-dependent partial differential equations, but in general this need not be the case.

One approach to obtain an understanding of this dynamical system is to perform direct time simulation, viewing the dynamical system as an initial value problem. In this case, an initial condition, boundary conditions and time horizon are specified. The equations are then integrated by some suitable method over the time horizon to obtain the state of the system at the desired time. Such a computational routine is often called a 'time-stepper' in the literature. By their very nature, time-steppers are iterative, continually advancing the solution one step forward in time for each iteration.

Much information about the dynamical system can be obtained in this way. However, as a means of understanding equilibria and their bifurcation structure, simulation codes have drawbacks. Near bifurcation points, the rate of convergence of the time-stepper algorithm becomes arbitrarily slow, causing the computational expense to grow significantly. As a result, information about solution bifurcations is usually limited to simply observing the change in the solution. The location of any bifurcations, in terms of the parameter λ, is difficult to establish with accuracy. Furthermore, the computation of unstable solutions is impossible with a time-stepper code, since such codes converge only to stable solutions. However, knowledge of unstable solutions is often desired to understand more fully the system's behaviour. These limitations of the time-stepper are due to the fact that bifurcations, the rate of convergence and the stability of solutions are all connected to the eigenvalues of the linearization of the problem, as is explained in Appendix A.

There are many situations, such as in industrial settings, where considerable time and effort have been expended developing an accurate and efficient time-stepper for a given system. It would be beneficial to be able to use a simulation code in a 'black box' fashion to compute unstable solutions and bifurcation behaviour that are not otherwise obtainable from such a code.

Some progress has already been made in this direction. Tuckerman and Barkley [1] have developed methods for transforming an existing time-stepper code of a particular form into a bifurcation tool, although their techniques are not applicable to arbitrary time-steppers. Also relevant is the work of Koronaki *et al.* [2], who have written a program that enables existing time-stepper codes that model tubular reactors to perform bifurcation studies. They use a method called the recursive projection method (RPM), originally developed by Shroff and Keller [3].

This paper describes the modification of an existing time-stepper code to perform bifurcation analysis of the Taylor–Couette problem. While many time simulation codes for the Taylor–Couette problem have been written, we focus on the code written by Adair [4], which we have adapted to perform bifurcation style analysis by implementing the Newton–Picard method of Lust *et al.* [5].

The organization of the paper is as follows. Section 2 gives details of the Newton–Picard method and shows how the eigenvalue information can be used to adapt an existing time-stepper. Details of how the eigenvalues of the linearized problem affect the performance of the time-stepper, and how this information can be used for a bifurcation study of the dynamical system are provided in Appendix A. In Section 3, we present the numerical method employed by Adair's time-stepper code for the Taylor–Couette problem. Sections 4 and 5 give implementation details of the adapted code as well as some results of numerical experiments conducted for equilibrium and time-periodic solutions, respectively. Finally, Section 6 summarizes our results and highlights our conclusions.

DOI: 10.1002/fld

2. THE NEWTON–PICARD METHOD

We used the Newton–Picard method developed by Lust *et al.* [5–9] to adapt a Taylor–Couette time-stepper code. The basic idea is to reformulate the fixed point iteration as a root-finding problem to which Newton's method can be applied. The calculation is then split into two parts. The first part corresponds to the 'slow' or 'dangerous' modes of the problem whose solution is computed in a small-dimensional subspace by direct matrix methods. The other (complementary) part of the solution is found in terms of a power series approximation. We first discuss the application of the Newton–Picard method to finding equilibrium solutions, followed by the extension to compute time-periodic solutions.

2.1. Computing equilibrium solutions

The discretization of the time-dependent axisymmetric Navier–Stokes equations using the techniques to be described in Section 3 reduces to performing an iteration of the form

$$\mathbf{u}^{n+1} = \mathbf{u}^n + \Delta t \mathbf{f}(\mathbf{u}^n, \lambda) \equiv \mathbf{F}(\mathbf{u}^n, \lambda) \tag{2}$$

and finding the fixed points of the mapping $\mathbf{F}$.

The convergence of the iteration slows arbitrarily as an eigenvalue of the Jacobian matrix approaches the unit circle in the complex plane, and it diverges if an eigenvalue lies outside the unit circle. The Newton–Picard method overcomes these difficulties by instead finding fixed points of $\mathbf{F}$ using Newton's method to find roots of the equation $\mathbf{r}(\mathbf{u}) \equiv \mathbf{F}(\mathbf{u}) - \mathbf{u}$. This entails solving the system

$$(M^v - I)\Delta\mathbf{u}^v = -\mathbf{r}(\mathbf{u}^v), \quad v = 0, 1, 2, \ldots \tag{3}$$

for $\Delta\mathbf{u}^v$, and following this by the update

$$\mathbf{u}^{v+1} = \mathbf{u}^v + \Delta\mathbf{u}^v$$

Here, the superscript v counts the Newton iteration, and for convenience we have denoted the Jacobian matrix $\mathbf{F}_\mathbf{u}(\mathbf{u}^v, \lambda)$ by M^v.

System (3) may be large and using direct matrix methods would be costly. The ingenuity of the Newton–Picard method lies in using direct methods to solve only a small-dimensional problem in a subspace of $\mathbb{R}^N$. This subspace is the invariant subspace of $\mathbb{R}^N$ which corresponds to the slowly decaying or non-decaying modes of the Jacobian matrix $\mathbf{F}_\mathbf{u}(\mathbf{u}^*, \lambda)$, i.e. the subspace spanned by the (generalized) eigenvectors of the Jacobian matrix corresponding to the largest eigenvalues in modulus. A projected iteration is used instead of direct methods in the orthogonal complement of this low-dimensional subspace.

More precisely, let $\mathbf{u}^*$ be an isolated solution of $\mathbf{r}(\mathbf{u}) = \mathbf{0}$, and let $\mathscr{B}$ be a small neighbourhood of $\mathbf{u}^*$. Let $M(\mathbf{u}) = \mathbf{F}_\mathbf{u}(\mathbf{u}, \lambda)$ for $\mathbf{u} \in \mathscr{B}$ and denote its eigenvalues by $\mu_1, \ldots, \mu_N$, ordered by decreasing modulus and repeated according to multiplicity. Let $\delta > 0$ be a small, user-defined number, and assume that for every $\mathbf{u} \in \mathscr{B}$, precisely p eigenvalues of $M(\mathbf{u})$ lie outside the circle of radius $1 - \delta$ in the complex plane. Assume further that no eigenvalue has modulus equal to $1 - \delta$. Hence, for every $\mathbf{u} \in \mathscr{B}$, we have

$$|\mu_1| \geqslant |\mu_2| \geqslant \cdots \geqslant |\mu_p| > 1 - \delta > |\mu_{p+1}| \geqslant \cdots \geqslant |\mu_N|$$

Lust points out [6] that this assumption is typically observed in practice.

DOI: 10.1002/fld

Let $\mathscr{P}$ denote the p-dimensional invariant subspace of the Jacobian $M(\mathbf{u})$ corresponding to the eigenvalues $\mu_1, \ldots, \mu_p$. Let V_p be an $N \times p$ real matrix whose columns form an orthonormal basis for $\mathscr{P}$. Next, let $\mathscr{Q} = \mathscr{P}^\perp$ and suppose that V_q is a real $N \times (N-p)$ matrix whose columns form an orthonormal basis for $\mathscr{Q}$. Let $P = V_p V_p^{\mathrm{T}}$ and $Q = V_q V_q^{\mathrm{T}} = I - P$ be orthogonal projectors of $\mathbb{R}^N$ onto $\mathscr{P}$ and $\mathscr{Q}$, respectively. This gives the direct sum decomposition $\mathbb{R}^N = \mathscr{P} \oplus \mathscr{Q}$, and hence each $\mathbf{u} \in \mathbb{R}^N$ can be written uniquely as

$$\mathbf{u} = P\mathbf{u} + Q\mathbf{u}$$

where $P\mathbf{u} \in \mathscr{P}$ and $Q\mathbf{u} \in \mathscr{Q}$. It will often be convenient to work with local coordinates in $\mathscr{P}$ and $\mathscr{Q}$. Utilizing the matrices V_p and V_q, we change coordinates by setting

$$\bar{\mathbf{p}} = V_p^{\mathrm{T}}\mathbf{u} \in \mathbb{R}^p, \quad \bar{\mathbf{q}} = V_q^{\mathrm{T}}\mathbf{u} \in \mathbb{R}^{N-p}$$

from which we have $\mathbf{u} = V_p\bar{\mathbf{p}} + V_q\bar{\mathbf{q}}$. Similarly, we can write $\Delta\mathbf{u} = V_p\Delta\bar{\mathbf{p}} + V_q\Delta\bar{\mathbf{q}}$. Substituting this decomposition into the Newton's method equation (3) and multiplying by $[V_q \ V_p]^{\mathrm{T}}$ yields the system

$$\begin{bmatrix} V_q^{\mathrm{T}}(M^\nu - I)V_q & 0 \\ V_p^{\mathrm{T}}M^\nu V_q & V_p^{\mathrm{T}}(M^\nu - I)V_p \end{bmatrix} \begin{bmatrix} \Delta\bar{\mathbf{q}}^\nu \\ \Delta\bar{\mathbf{p}}^\nu \end{bmatrix} = - \begin{bmatrix} V_q^{\mathrm{T}}\mathbf{r}(\mathbf{u}^\nu) \\ V_p^{\mathrm{T}}\mathbf{r}(\mathbf{u}^\nu) \end{bmatrix} \tag{4}$$

Here we have used the fact that $V_p^{\mathrm{T}}V_q = 0$ and $V_q^{\mathrm{T}}V_p = 0$ since $\mathscr{P}$ and $\mathscr{Q}$ are orthogonal, and that $V_q^{\mathrm{T}}M^\nu V_p = 0$ since $\mathscr{P}$ is an invariant subspace of M^ν. Note that system (4) is block lower triangular. Thus, we can solve for $\Delta\bar{\mathbf{q}}^\nu$ first and use this value in the resulting equation to find $\Delta\bar{\mathbf{p}}^\nu$. Then, we get

$$\bar{\mathbf{p}}^{\nu+1} = \bar{\mathbf{p}}^\nu + \Delta\bar{\mathbf{p}}^\nu$$

$$\bar{\mathbf{q}}^{\nu+1} = \bar{\mathbf{q}}^\nu + \Delta\bar{\mathbf{q}}^\nu$$

and hence

$$\mathbf{u}^{\nu+1} = V_p\bar{\mathbf{p}}^{\nu+1} + V_q\bar{\mathbf{q}}^{\nu+1}$$

2.1.1. Solution of $\Delta\bar{\mathbf{q}}^\nu$. From (4), the system that determines $\Delta\bar{\mathbf{q}}^\nu$ is

$$(V_q^{\mathrm{T}}M^\nu V_q - I)\Delta\bar{\mathbf{q}}^\nu = -V_q^{\mathrm{T}}\mathbf{r}(\mathbf{u}^\nu) \tag{5}$$

Hereafter, we refer to system (5) as the $\mathscr{Q}$-equations. This is a large system that we wish to avoid solving directly. Since the spectral radius of $V_q^{\mathrm{T}}M^\nu V_q$ is less than one by construction, we know that the Neumann series for $(V_q^{\mathrm{T}}M^\nu V_q - I)^{-1}$ is convergent, and hence

$$(V_q^{\mathrm{T}}M^\nu V_q - I)^{-1} = -\sum_{k=0}^{\infty} (V_q^{\mathrm{T}}M^\nu V_q)^k \tag{6}$$

We approximate this series by taking the first ℓ terms to get

$$\Delta\bar{\mathbf{q}}^\nu \approx \sum_{k=0}^{\ell-1} (V_q^{\mathrm{T}}M^\nu V_q)^k V_q^{\mathrm{T}}\mathbf{r}(\mathbf{u}^\nu)$$

DOI: 10.1002/fld

This series expansion can be viewed as the following iteration:

$$\begin{aligned} \Delta\bar{\mathbf{q}}^{[0]} &= \mathbf{0} \\ \Delta\bar{\mathbf{q}}^{[k]} &= V_q^{\mathrm{T}} M V_q \Delta\bar{\mathbf{q}}^{[k-1]} + V_q^{\mathrm{T}}\mathbf{r}, \quad k = 1, \ldots, \ell \end{aligned} \tag{7}$$

and then setting $\Delta\bar{\mathbf{q}}^{\nu} = \Delta\bar{\mathbf{q}}^{[\ell]}$. Note that in (7) we have used the superscripts $[k]$ to count the iteration number, *not* the Newton step number of (4). Multiplying (7) by V_q and using $V_q\bar{\mathbf{q}} = \mathbf{q} \in \mathbb{R}^N$ yields the Picard iteration

$$\begin{aligned} \Delta\mathbf{q}^{[0]} &= \mathbf{0} \\ \Delta\mathbf{q}^{[k]} &= QM\Delta\mathbf{q}^{[k-1]} + Q\mathbf{r}, \quad k = 1, \ldots, \ell \\ \Delta\mathbf{q}^{\nu} &= \Delta\mathbf{q}^{[\ell]} \end{aligned} \tag{8}$$

Since $Q = I - P = I - V_p V_p^{\mathrm{T}}$, we need only to use the small matrix V_p when executing (8). In fact, for numerical stability the matrix V_p is not used directly, rather the projection onto $\mathcal{Q}$ is performed *via* a modified Gram–Schmidt method (without normalization). Rather than using a fixed number ℓ of iterations, we chose to iterate until the norm of the residual

$$\| QM\Delta\mathbf{q} + Q\mathbf{r} - \Delta\mathbf{q} \|$$

falls below a user-defined tolerance.

If $\ell > 1$, then each iteration requires the evaluation of the matrix–vector product $M^{\nu}\Delta\mathbf{q}$, where M^{ν} is the Jacobian matrix $\mathbf{F}_{\mathbf{u}}(\mathbf{u}^{\nu}, \lambda)$. Such a matrix–vector product can be approximated by the finite difference (9). If $\mathbf{v}$ is any vector of the appropriate dimension, the matrix–vector product $\mathbf{F}_{\mathbf{u}}(\mathbf{u}, \lambda)\mathbf{v}$ is approximated by

$$\mathbf{F}_{\mathbf{u}}(\mathbf{u}, \lambda)\mathbf{v} \approx \frac{\mathbf{F}(\mathbf{u} + \varepsilon\mathbf{v}, \lambda) - \mathbf{F}(\mathbf{u}, \lambda)}{\varepsilon} \tag{9}$$

where ε is determined by the user. Using this approximation requires two calls to the time-stepper code for each iteration, an additional cost associated with using the iteration (8) with $\ell \geqslant 2$ that is not present in the $\ell = 1$ case.

2.1.2. Solution of $\Delta\bar{\mathbf{p}}^{\nu}$. The system of equations that defines $\Delta\bar{\mathbf{p}}^{\nu}$ can be written as

$$V_p^{\mathrm{T}}(M^{\nu} - I)V_p\Delta\bar{\mathbf{p}}^{\nu} = -V_p^{\mathrm{T}}\mathbf{r}(\mathbf{u}^{\nu}) - V_p^{\mathrm{T}}M^{\nu}V_q\Delta\bar{\mathbf{q}}^{\nu} \tag{10}$$

which we will hereafter refer to as the $\mathcal{P}$-equations. Note that the updated value of $\Delta\bar{\mathbf{q}}^{\nu}$ appears on the right-hand side of this equation. Since the $\mathcal{P}$-equations (10) form a small (p-dimensional) system, we can solve for $\Delta\bar{\mathbf{p}}^{\nu}$ using any method. However, system (10) is singular at bifurcation and fold points, and is very ill-conditioned near these points. For these reasons, we solve (10) by using a least-squares method based on the singular value decomposition (SVD), which has the advantage of being numerically stable [10].

2.1.3. Computing a basis of $\mathcal{P}$. As the parameter λ is varied, the solution $\mathbf{u}^*$ changes as do the eigenvalues of the Jacobian matrix. If some eigenvalues move closer to the boundary of the unit circle in the complex plane and move outside the circle of radius $1 - \delta$, then the dimension of the

DOI: 10.1002/fld

invariant subspace $\mathscr{P}$ will need to be increased. The Newton–Picard method continually monitors these eigenvalues and builds a basis for $\mathscr{P}$ appropriately. It is also possible that some eigenvalues, originally outside the circle of radius $1-\delta$, move inside that circle as λ is varied. In this case the dimension of $\mathscr{P}$ should be decreased.

2.2. Computing time-periodic solutions

We now turn to computing time-periodic solutions of

$$\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t} = \mathbf{f}(\mathbf{u}, \lambda) \tag{11}$$

where we assume that $\mathbf{f}$ is autonomous. This assumption implies that a periodic solution of the system is entirely determined by the initial condition $\mathbf{u}(0) = \mathbf{u}_0$ and the period of the solution, which we denote by T.

Note that if $\mathbf{u}(t)$ is a periodic solution of (11), then so is $\mathbf{u}(t+\tau)$ for any $\tau \in \mathbb{R}$. Hence, there is a non-uniqueness associated with periodic solutions, and any point of the orbit can be used as the point at time $t=0$. We are free to choose this point by introducing an auxiliary equation $\psi(\mathbf{u}_0, T) = 0$, which is called a phase condition. Examples of phase conditions used in practice can be found in [11]. Hence, a periodic solution can be found by solving the two-point boundary value problem

$$\begin{aligned} \frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t} &= \mathbf{f}(\mathbf{u}, \lambda) \\ \frac{\mathrm{d}T}{\mathrm{d}t} &= 0 \\ \mathbf{u}(T) &= \mathbf{u}_0 \\ \psi(\mathbf{u}_0, T) &= 0 \end{aligned} \tag{12}$$

Let $\boldsymbol{\phi}(\mathbf{u}_0, t)$ denote the flow function, that is, the solution of the initial value problem (11) at time t for a given initial condition $\mathbf{u}_0$. This solution $\boldsymbol{\phi}$ is found by the time-stepper code, stepping forward in time until time t is reached. A standard technique to solve the two-point boundary value problem (12) is to use a shooting approach, which involves using Newton's method to solve the system

$$\begin{aligned} \mathbf{r}(\mathbf{u}_0, T) \equiv \boldsymbol{\phi}(\mathbf{u}_0, T) - \mathbf{u}_0 &= \mathbf{0} \\ \psi(\mathbf{u}_0, T) &= 0 \end{aligned} \tag{13}$$

In particular, at each Newton iteration we solve the system

$$\begin{bmatrix} M^v - I & \boldsymbol{\phi}_T^v \\ \psi_{\mathbf{u}}^v & \psi_T^v \end{bmatrix} \begin{bmatrix} \Delta\mathbf{u}_0^v \\ \Delta T^v \end{bmatrix} = - \begin{bmatrix} \mathbf{r}(\mathbf{u}_0^v, T^v) \\ \psi(\mathbf{u}_0^v, T^v) \end{bmatrix} \tag{14}$$

and then set

$$\begin{aligned} \mathbf{u}_0^{v+1} &= \mathbf{u}_0^v + \Delta\mathbf{u}_0^v \\ T^{v+1} &= T^v + \Delta T^v \end{aligned}$$

DOI: 10.1002/fld

As before, the superscript v counts the Newton iterations, the matrix M is the Jacobian $\mathrm{d}\boldsymbol{\phi}/\mathrm{d}\mathbf{u} = \mathbf{G}_{\mathbf{u}}$, and all derivatives in (14) are evaluated at $(\mathbf{u}_0^v, T^v)$. As in the equilibrium case, we can approximate the action of M^v on a vector $\mathbf{v}$ by a finite difference

$$M^v\mathbf{v} \approx \frac{\boldsymbol{\phi}(\mathbf{u}_0^v + \varepsilon\mathbf{v}, T^v) - \boldsymbol{\phi}(\mathbf{u}_0^v, T^v)}{\varepsilon}$$

This evaluation of $M^v\mathbf{v}$ will require $2k$ calls to the time-stepper code, where k is the number of time steps required to integrate over a single period, starting with slightly different initial conditions. Since $\boldsymbol{\phi}(\mathbf{u}_0, t)$ solves $\mathrm{d}\mathbf{u}/\mathrm{d}t = \mathbf{f}(\mathbf{u})$ by definition, we conclude that

$$\boldsymbol{\phi}_T^v = \mathbf{f}(\boldsymbol{\phi}(\mathbf{u}_0^v, T^v))$$

The evaluation of the two derivatives $\psi_{\mathbf{u}}^v$ and ψ_T^v depends on the form of the phase condition $\psi(\mathbf{u}_0, T)$ used.

Rather than solving system (14) directly, which is expensive when N is large, we apply the Newton–Picard method. Our assumptions are analogous to those of the equilibrium case. Let $(\mathbf{u}_0^*, T^*)$ be an isolated solution of (13), let $\mathscr{B}$ be a small neighbourhood of $(\mathbf{u}_0^*, T^*)$, and let $M(\mathbf{u}_0, T) = \boldsymbol{\phi}_{\mathbf{u}}(\mathbf{u}_0, T)$ for $(\mathbf{u}_0, T) \in \mathscr{B}$. Denote the eigenvalues of M by μ_i, $i = 1, \ldots, N$, and assume that for all $(\mathbf{u}_0, T) \in \mathscr{B}$ precisely p eigenvalues lie outside the circle of radius $1 - \delta$ for some small $\delta > 0$. As in the equilibrium case, the parameter δ is determined by the user.

As before, let $\mathscr{P}$ denote the invariant subspace of M spanned by the eigenvectors and generalized eigenvectors of M which correspond to the eigenvalues which lie outside the circle of radius $1 - \delta$. Let V_p be an orthogonal matrix whose columns span $\mathscr{P}$; the orthogonal projector onto $\mathscr{P}$ is given by $P = V_p V_p^{\mathrm{T}}$. Also, let $\mathscr{Q} = \mathscr{P}^{\perp}$ with orthonormal basis V_q and orthogonal projector $Q = V_q V_q^{\mathrm{T}} = I - P$. For $\Delta\mathbf{u}_0 \in \mathbb{R}^N$, this yields the unique decomposition

$$\Delta\mathbf{u}_0 = V_p\Delta\bar{\mathbf{p}} + V_q\Delta\bar{\mathbf{q}} \quad \text{where } \Delta\bar{\mathbf{p}} \in \mathbb{R}^p \text{ and } \Delta\bar{\mathbf{q}} \in \mathbb{R}^{N-p}$$

Substituting this decomposition into system (14) and multiplying the first N equations by $[V_q\ V_p]^{\mathrm{T}}$, we obtain the system

$$\begin{bmatrix} V_q^{\mathrm{T}}(M^v - I)V_q & 0 & V_q^{\mathrm{T}}\boldsymbol{\phi}_T^v \\ V_p^{\mathrm{T}}M^vV_q & V_p^{\mathrm{T}}(M^v - I)V_p & V_p^{\mathrm{T}}\boldsymbol{\phi}_T^v \\ \psi_{\mathbf{u}}^vV_q & \psi_{\mathbf{u}}^vV_p & \psi_T^v \end{bmatrix} \begin{bmatrix} \Delta\bar{\mathbf{q}}^v \\ \Delta\bar{\mathbf{p}}^v \\ \Delta T^v \end{bmatrix} = - \begin{bmatrix} V_q^{\mathrm{T}}\mathbf{r}(\mathbf{u}_0^v, T^v) \\ V_p^{\mathrm{T}}\mathbf{r}(\mathbf{u}_0^v, T^v) \\ \psi(\mathbf{u}_0^v, T^v) \end{bmatrix} \tag{15}$$

As in the equilibrium case, we have used the fact that $V_p^{\mathrm{T}}V_q = 0$ and $V_q^{\mathrm{T}}V_p = V_q^{\mathrm{T}}M^vV_p = 0$. Once $\Delta\bar{\mathbf{q}}^v$, $\Delta\bar{\mathbf{p}}^v$ and ΔT^v have been found, we set

$$\mathbf{u}_0^{v+1} = \mathbf{u}_0^v + V_p\Delta\bar{\mathbf{p}}^v + V_q\Delta\bar{\mathbf{q}}^v$$
$$T^{v+1} = T^v + \Delta T^v$$

The vector

$$\mathbf{b}^* \equiv \left.\frac{\partial\boldsymbol{\phi}}{\partial T}\right|_{(\mathbf{u}_0^*, T^*)}$$

DOI: 10.1002/fld

is an eigenvector of the Monodromy matrix $M(\mathbf{u}_0^*, T^*)$ corresponding to the eigenvalue 1 [6]. Hence, $\mathbf{b}^* \in \mathscr{P}$ by our assumption above, which implies that $V_q^{\mathrm{T}}\mathbf{b}^* = 0$. Thus, the term $V_q^{\mathrm{T}}\boldsymbol{\phi}_T^v$ appearing in Equation (15) is usually quite small and converges to zero as the Newton iteration converges. Lust *et al*., therefore neglect this term, and so will we.

With this simplification, the solution of (15) is computed as in the equilibrium case. We first solve

$$V_q^{\mathrm{T}}(M^v - I)V_q \Delta\bar{\mathbf{q}}^v = -V_q^{\mathrm{T}}\mathbf{r}(\mathbf{u}_0^v, T^v)$$

for $\Delta\bar{\mathbf{q}}^v$ *via* an iteration based on the Neumann series expansion for $[V_q^{\mathrm{T}}(M^v - I)V_q]^{-1}$. Next, $\Delta\bar{\mathbf{p}}^v$ and ΔT^v are found by directly solving the small system

$$\begin{bmatrix} V_p^{\mathrm{T}}(M^v - I)V_p & V_p^{\mathrm{T}}\boldsymbol{\phi}_T^v \\ \psi_{\mathbf{u}}^v V_p & \psi_T^v \end{bmatrix} \begin{bmatrix} \Delta\bar{\mathbf{p}}^v \\ \Delta T^v \end{bmatrix} = - \begin{bmatrix} V_p^{\mathrm{T}}[\mathbf{r}(\mathbf{u}_0^v, T^v) + M^v V_q \Delta\bar{\mathbf{q}}^v] \\ \psi(\mathbf{u}_0^v, T^v) + \psi_{\mathbf{u}}^v V_q \Delta\bar{\mathbf{q}}^v \end{bmatrix} \tag{16}$$

The basis V_p is computed in precisely the same way as in the equilibrium case.

3. THE UNDERLYING TIME-STEPPER CODE

The time-stepper we adapted is a code to simulate Taylor–Couette flow written by Ron Adair [4], which he named TAY. The Taylor–Couette problem concerns the flow of an incompressible fluid confined to the annular region between two concentric cylinders and driven by the rotation of one or both of the cylinders. The problem has been studied extensively and is one of the classic problems of hydrodynamic stability.

3.1. Governing equations and numerical method

The Navier–Stokes equations describe the flow of a viscous incompressible fluid. Let the radii of the inner and outer cylinders be r_1 and r_2, respectively, and the angular velocities of the inner and outer cylinders be Ω_1 and Ω_2, respectively. The Navier–Stokes equations may be written in non-dimensional form by choosing the characteristic length to be the gap width $d = r_2 - r_1$, the characteristic velocity to be the velocity of the inner cylinder $r_1\Omega_1$. In order to simplify the explanation of the Chorin–Temam projection scheme, rather than let these characteristic length and velocity scales implicitly define the length scale, we define the diffusive time scale d^2/ν which means that the Reynolds number multiplies the convective terms only in the non-dimensionalized Navier–Stokes equations below. The reference pressure scale is $\rho\nu r_1\Omega_1/d$, where ν denotes the viscosity of the fluid and ρ the (constant) density. We define the inner and outer Reynolds numbers to be

$$R_{\mathrm{i}} = \frac{r_1\Omega_1 d}{\nu} \quad \text{and} \quad R_{\mathrm{o}} = \frac{r_2\Omega_2 d}{\nu}$$

respectively, and the geometric parameters, radius ratio $\eta = r_1/r_2$, aspect ratio $\Gamma = h/d$, where h is the height of the cylinders and $\beta = r_1/d$.

The TAY code is capable of computing fully three-dimensional flows in the Taylor–Couette problem. For our adapted code, however, we focused on those flows which are axisymmetric. We were therefore unable to compute bifurcations that broke this invariance property. However, fully 3D computations in the parameter regions studied here detected only axisymmetric flows, supporting

DOI: 10.1002/fld

our decision to make this restriction. Let (u, v, w) denote the non-dimensional radial, azimuthal and axial velocities, respectively, and let p denote the pressure. Using the non-dimensionalization defined above, the axisymmetric Navier–Stokes equations in cylindrical coordinates, are

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial r^2} + \frac{1}{(r+\beta)}\frac{\partial u}{\partial r} + \frac{\partial^2 u}{\partial z^2} - \frac{u}{(r+\beta)^2} - R_{\mathrm{i}}\left(u\frac{\partial u}{\partial r} + w\frac{\partial u}{\partial z} - \frac{v^2}{(r+\beta)}\right) - \frac{\partial p}{\partial r} \tag{17}$$

$$\frac{\partial v}{\partial t} = \frac{\partial^2 v}{\partial r^2} + \frac{1}{(r+\beta)}\frac{\partial v}{\partial r} + \frac{\partial^2 v}{\partial z^2} - \frac{v}{r^2} - R_{\mathrm{i}}\left(u\frac{\partial v}{\partial r} + w\frac{\partial v}{\partial z} + \frac{uv}{(r+\beta)}\right) \tag{18}$$

$$\frac{\partial w}{\partial t} = \frac{\partial^2 w}{\partial r^2} + \frac{1}{(r+\beta)}\frac{\partial w}{\partial r} + \frac{\partial^2 w}{\partial z^2} - R_{\mathrm{i}}\left(u\frac{\partial w}{\partial r} + w\frac{\partial w}{\partial z}\right) - \frac{\partial p}{\partial z} \tag{19}$$

$$\frac{\partial u}{\partial r} + \frac{u}{(r+\beta)} + \frac{\partial w}{\partial z} = 0 \tag{20}$$

which were solved in the domain $\Omega = \{(r, z) : 0 \leqslant r \leqslant 1, 0 \leqslant z \leqslant \Gamma\}$. The ends of the cylinders were fixed, and no-slip velocity boundary conditions are imposed at the ends and at the cylindrical walls.

Finite differences, specifically centred differences, were used to approximate all spatial derivatives, while the Chorin–Témam projection scheme [12, 13] was used to implement the temporal discretization. Our development here follows [14]. The time derivatives were approximated by first order finite differences, so that at time step $n+1$ (that is, at time $t = (n+1)\Delta t$), we used the approximation

$$\left(\frac{\partial u}{\partial t}\right)^{n+1} \approx \frac{u^{n+1} - u^n}{\Delta t}$$

and similarly for the derivatives $\partial v/\partial t$ and $\partial w/\partial t$. The Chorin–Témam scheme evaluates all velocities appearing on the right-hand side of (17)–(19) at time step n, while the pressure terms are evaluated at time step $n+1$. If we let

$$\begin{aligned}
F_1^n &= u^n + \Delta t\left[\frac{\partial^2 u^n}{\partial r^2} + \frac{1}{(r+\beta)}\frac{\partial u^n}{\partial r} + \frac{\partial^2 u^n}{\partial z^2} - \frac{u^n}{(r+\beta)^2} - R_{\mathrm{i}}\left(u^n\frac{\partial u^n}{\partial r} + w^n\frac{\partial u^n}{\partial z} - \frac{(v^n)^2}{(r+\beta)}\right)\right] \\
F_2^n &= v^n + \Delta t\left[\frac{\partial^2 v^n}{\partial r^2} + \frac{1}{(r+\beta)}\frac{\partial v^n}{\partial r} + \frac{\partial^2 v^n}{\partial z^2} - \frac{v^n}{(r+\beta)^2} - R_{\mathrm{i}}\left(u^n\frac{\partial v^n}{\partial r} + w^n\frac{\partial v^n}{\partial z} + \frac{u^n v^n}{(r+\beta)}\right)\right] \\
F_3^n &= w^n + \Delta t\left[\frac{\partial^2 w^n}{\partial r^2} + \frac{1}{(r+\beta)}\frac{\partial w^n}{\partial r} + \frac{\partial^2 w^n}{\partial z^2} - R_{\mathrm{i}}\left(u^n\frac{\partial w^n}{\partial r} + w^n\frac{\partial w^n}{\partial z}\right)\right]
\end{aligned} \tag{21}$$

DOI: 10.1002/fld

then Equations (17)–(19) are completely discretized as

$$\begin{aligned} u^{n+1} &= F_1^n - \Delta t \frac{\partial p^{n+1}}{\partial r} \\ v^{n+1} &= F_2^n \\ w^{n+1} &= F_3^n - \Delta t \frac{\partial p^{n+1}}{\partial z} \end{aligned} \tag{22}$$

To determine the pressure, the continuity equation (20) is evaluated at time step $n+1$. Substituting (22) into (20), we obtain

$$\begin{aligned} 0 &= \frac{\partial u^{n+1}}{\partial r} + \frac{u^{n+1}}{(r+\beta)} + \frac{\partial w^{n+1}}{\partial z} \\ &= \frac{\partial}{\partial r}\left(F_1^n - \Delta t \frac{\partial p^{n+1}}{\partial r}\right) + \frac{1}{(r+\beta)}\left(F_1^n - \Delta t \frac{\partial p^{n+1}}{\partial r}\right) + \frac{\partial}{\partial z}\left(F_3^n - \Delta t \frac{\partial p^{n+1}}{\partial z}\right) \end{aligned}$$

and hence,

$$\frac{\partial^2 p^{n+1}}{\partial r^2} + \frac{1}{(r+\beta)}\frac{\partial p^{n+1}}{\partial r} + \frac{\partial^2 p^{n+1}}{\partial z^2} = \frac{1}{\Delta t}\left(\frac{\partial F_1^n}{\partial r} + \frac{F_1^n}{(r+\beta)} + \frac{\partial F_3^n}{\partial z}\right) \tag{23}$$

We recognize this last equation as a Poisson equation in cylindrical coordinates for the pressure p^{n+1}. It ensures that the velocity field is divergence free. On the boundary we use the numerical boundary condition for the pressure given by [12]. The TAY code solves (23) along with the boundary condition iteratively by the Gauss–Seidel method. Once the pressure p^{n+1} is known, it is used in (22) to compute the velocity field at time step $n+1$.

Let $\mathbf{x} = (\mathbf{u}, \mathbf{v}, \mathbf{w}) \in \mathbb{R}^N$. The Chorin projection scheme, as part of its solution algorithm, enforces the condition that $\mathbf{x}$ have zero (discrete) divergence. We therefore consider the Chorin–Temam projection scheme applied to the spatially discretized axisymmetric Navier–Stokes equations as a fixed point iteration of the form

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \mathbf{f}(\mathbf{x}), \quad \mathbf{x} \in \mathbb{R}^N_{\mathrm{sol}} \tag{24}$$

where $\mathbf{x} \in \mathbb{R}^N_{\mathrm{sol}}$ are discretely divergence free. We apply the Newton–Picard method to this iteration. In terms of time-stepping, the TAY code is an iteration of the form

$$\mathbf{x}^{n+1} = \mathbf{x}^n + \Delta t\, \mathbf{f}(\mathbf{x}^n) \equiv \mathbf{F}(\mathbf{x}^n), \quad n = 0, 1, \ldots \tag{25}$$

The map $\mathbf{F}$ is the discrete solution operator over one time step, integrating the Navier–Stokes equations forward in time by Δt. The domain of $\mathbf{F}$ is the subspace of $\mathbb{R}^N_{\mathrm{sol}}$ consisting of those vectors with zero (discrete) divergence. If $\mathbf{x}^n$ is not divergence free, as may happen, for example, during the course of the eigenvalue calculation when the action of the Jacobian matrix on an arbitrary given vector is to be determined by differencing, the result $\mathbf{x}^{n+1}$ is none-the-less divergence free. This may have reduced the efficiency of the implicitly restarted Arnoldi procedure used in the calculation of a basis for $\mathscr{P}$.

DOI: 10.1002/fld

4. NUMERICAL RESULTS FOR EQUILIBRIUM SOLUTIONS

4.1. Implementation details

Several implementation issues had to be addressed before the Newton–Picard method could be used to adapt the TAY code successfully and to produce a working bifurcation code. First, note that the Chorin–Témam scheme is a mapping of the velocity field only. The pressure term is dependent only on the velocity field at the previous time step and not on any previous pressure values. Thus, the time-stepper is properly viewed as the mapping $\mathbf{F} : (u, v, w)_{\text{sol}} \to (u, v, w)_{\text{sol}}$ where $(u, v, w)_{\text{sol}}$ are discretely divergence free rather than $\mathbf{F} : (u, v, w, p) \to (u, v, w, p)$.

The Jacobian matrix $\mathbf{F}_\mathbf{u}$ is most properly the projection of the matrix $\mathbf{F}_\mathbf{u}$ into the divergence-free subspace $\mathbb{R}^N_{\text{sol}}$. There are three different techniques for linearizing the map $\mathbf{F}$.

(1) Explicitly constructing the Jacobian matrix *by differencing*; i.e. approximating the ith column of $\mathbf{F}_\mathbf{u}$ by

$$[\mathbf{F}_\mathbf{u}]_i \approx \frac{\mathbf{F}(\mathbf{u} + \varepsilon \mathbf{e}_i) - \mathbf{F}(\mathbf{u})}{\varepsilon}$$

for some small $\varepsilon > 0$. Note $\mathbf{F}(\mathbf{u} + \varepsilon \mathbf{e}_i)$ will be divergence free by construction.

(2) Finite differences may also be used as in Equation (9) to approximate the multiplicative action of the Jacobian on a vector $\mathbf{v}$, i.e.

$$[\mathbf{F}_\mathbf{u}]\mathbf{v} \approx \frac{\mathbf{F}(\mathbf{u} + \varepsilon \mathbf{v}) - \mathbf{F}(\mathbf{u})}{\varepsilon}$$

Note that this is discretely divergence free even if $\mathbf{v}$ is not.

(3) The multiplicative action of the Jacobian matrix may also be approximated by solving the linearized Navier–Stokes equations directly using numerical methods that are consistent with those used for the non-linear problem.

One must decide which of these three options is optimal under the given circumstances. For testing purposes, we implemented all three (see below), but for final computations, finite differences (9) were used.

The TAY code uses the iterative Gauss–Seidel method to solve the Poisson equation (23) for the pressure terms. Upon discretization of the pressure and velocities, solving this equation amounts to solving a matrix equation. For a fixed cylinder geometry, the matrix of coefficients is always the same, regardless of the inner or outer Reynolds numbers. Rather than using the existing Gauss–Seidel solver in TAY, we instead utilized a matrix inverse to solve the discrete Poisson equation for the pressure. Solving the Poisson equation (23) then amounts to a simple matrix–vector multiplication, which can be performed efficiently using BLAS routines.

The software package ARPACK [15] was used in our adapted code to compute the dominant eigenvalues of the Jacobian matrix and the corresponding Schur vectors used as a basis for the dominant invariant subspace.

Finally, because the TAY code uses the explicit Euler scheme to approximate the time derivative along with centred differences for the convective term, a small time step increment Δt was required. Our numerical experiments typically used $\Delta t = 0.00005$. Such a small value of Δt introduces unintended consequences into the program. As discussed in Appendix A, the eigenvalues $\{\mu_k\}$ of $\mathbf{F}_\mathbf{u}$ are

DOI: 10.1002/fld

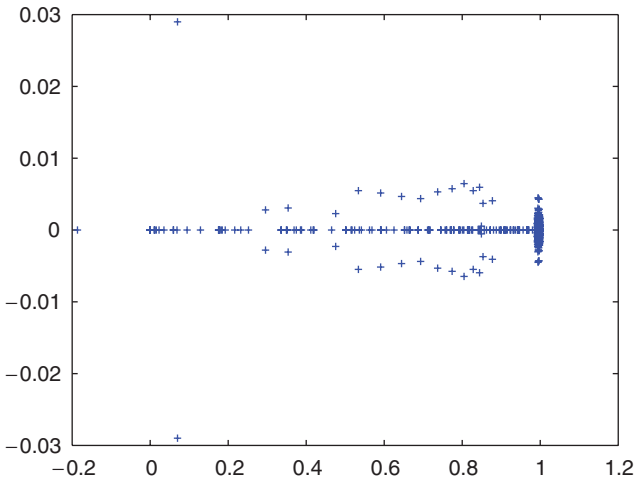

Figure 1. Eigenvalue clustering near 1 in the complex plane.

related to those $\{\alpha_k\}$ of $\mathbf{f_u}$ by

$$\mu_k = 1 + \Delta t \alpha_k, \quad k = 1, 2, \ldots, N$$

When Δt is very small, this causes the eigenvalues of $\mathbf{F_u}$ to be highly clustered near 1. Typically, over half of the eigenvalues have modulus greater than 0.99. Figure 1 shows a representative plot of the entire spectrum of the Jacobian matrix, plotted in the complex plane, where the clustering near 1 is evident.

There are several consequences of this eigenvalue clustering. Firstly, it increases the cost of the eigenvalue calculations in ARPACK, since Arnoldi methods are better suited for computing well-separated eigenvalues. A fixed basis size was used for the invariant subspace $\mathscr{P}$ rather than dynamically adapting the size of the basis as originally proposed in the Newton–Picard method. Secondly, the Neumann series (6) converges quite slowly since the spectral radius of $V_q^{\mathrm{T}} M V_q - I$ is still very close to 1. Finally, the clustering makes the matrix $V_q^{\mathrm{T}} M V_q - I$ very poorly conditioned. Since the value of $\Delta\mathbf{q}$ is used on the right-hand side of the $\mathscr{P}$-equations (10), this ill-conditioning requires that the $\mathscr{Q}$-equations (5) be solved very accurately, typically two or three orders of magnitude more accurately than the overall solution. We chose to use a simplified Newton's method, where the Jacobian matrix was fixed at the beginning of the iterations and only updated if the norm of the increment $\Delta\mathbf{u}$ increased.

4.2. Verification

As mentioned above, three different methods can be used to linearize $\mathbf{F}$. All three methods were used to verify the eigenvalue calculations and all three gave results that agreed to within the error tolerance. Further eigenvalue verification was made using MATLAB on the entire explicitly constructed Jacobian.

Stable equilibrium solutions computed with our adapted code were compared with those obtained directly from the TAY code. Typical results are summarized in Table I, where a convergence

DOI: 10.1002/fld

Table I. Comparison of Newton–Picard solutions and those directly from the TAY code.

(R_i, R_o)	(η, Γ)	$r \times z$ grid	*Residual norm*	*Solution difference norm*
$(300, -81)$	$(0.615, 2.4)$	11×25	8.4×10^{-9}	1.55×10^{-5}
$(300, -88)$	$(0.615, 2.4)$	11×25	2.0×10^{-9}	2.46×10^{-5}
$(122, 0)$	$(0.5, 1.0)$	21×21	1.34×10^{-9}	1.36×10^{-6}
$(258, 0)$	$(0.7, 0.7)$	21×15	2.65×10^{-10}	3.59×10^{-6}

tolerance of 10^{-5} was used. The first column gives the particular Reynolds numbers used and the second column gives the geometric parameters of the problem. The third column gives the spatial rz-discretization grid used. The fourth column gives the value of $\|\mathbf{F}(\mathbf{u}) - \mathbf{u}\|$, which represents how close the computed solution is to being a fixed point of the map $\mathbf{F}$. The last column indicates the norm of the difference of the solution computed by the adapted code and the corresponding solution computed from TAY. All norms are sup-norms.

Finally, bifurcation results obtained by the adapted code were compared with those that appear in Mullin *et al*. [16] and in Schulz *et al*. [17]. The locations of the bifurcation points computed by our adapted code compared favourably with their results.

4.3. Numerical results

In the numerical experiments performed with the adapted code, the inner and outer Reynolds numbers played the role of the bifurcation parameter λ in all of the experiments. Only one of them was allowed to change at a time. A typical numerical experiment with the code was performed as follows. An initial solution was generated using the TAY code, and sufficient time allowed to ensure that the solution reached equilibrium. Next, one of the Reynolds numbers was changed slightly, usually by 1 or 2%. The initial solution was used as the starting guess for the adapted code to compute the solution at the new Reynolds number. This process was repeated, changing one of the Reynolds numbers slightly and using the previously computed solution as the starting guess for the new Reynolds number. Keller's pseudo-arclength continuation method [18] was used to traverse turning points.

In all of the numerical experimental results presented in this chapter, the aspect ratio was fixed at 2.4 and the radius ratio was 0.615. The inner Reynolds number was held constant at $R_i = 300$ (except for one experiment noted below), while the outer Reynolds number was varied over the interval $-320 \leqslant R_o \leqslant 0$. The time step increment was fixed at $\Delta t = 0.00005$. A Newton iteration convergence tolerance of 10^{-5} was used, and the size of the basis of the invariant subspace $\mathscr{P}$ was chosen to be 15. The complete bifurcation structure is not given here.

4.3.1. Four-cell flows. The behaviour of four-cell flows in this region of parameter space is extremely complicated and has confounded a complete description. A stable, steady symmetric four-cell flow was established at $R_i = 300$, $R_o = -320$ using a spatial discretization of $\Delta r = \Delta z = 0.1$. This solution was not established by a quasi-static increase in Reynolds numbers, but rather by starting the time integration with a given set of initial conditions. Figure 2 is a representative velocity cross-section plot of this symmetric four-cell solution.

Increasing R_o from -320, a single real eigenvalue of the Jacobian matrix became equal to 1 at $R_o = -189$, indicating that the symmetric four-cell solution loses stability at that point. In the

DOI: 10.1002/fld

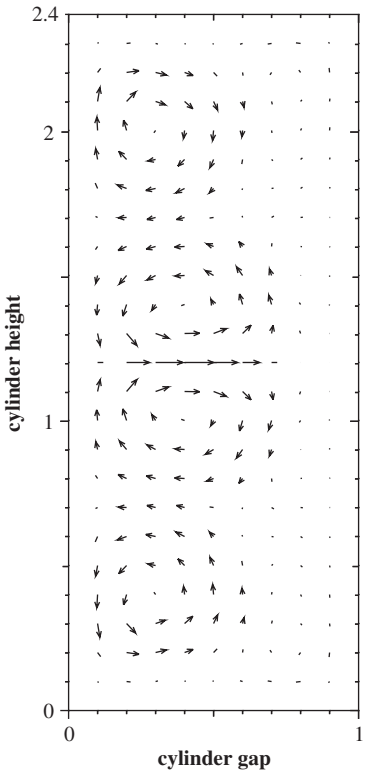

Figure 2. Symmetric four-cell Taylor vortex flow velocity cross-section.

absence of symmetry or other invariance property, an eigenvalue of unity generically indicates a turning point. Because the four-cell flow possesses a reflectional symmetry about the midplane, it is also possible for a symmetry-breaking bifurcation to occur at $R_o = -189$. Figure 3 shows the stable solution $R_o = -180$. A close inspection reveals that this solution, although still possessing four cells is asymmetric, the slight asymmetry being most visible near the midplane $z = 1.2$. The asymmetry is quite subtle at first, and even far from the bifurcation point it is not readily apparent to the naked eye. This illustrates one of the pitfalls of direct simulation codes: visual inspection of plots may fail to identify some bifurcation points.

For parameter values $R_o > -189$, there are two solution branches to follow. Increasing R_o from -189 to -100 along the stable asymmetric four-cell branch emanating from the symmetry-breaking bifurcation point showed no further bifurcation points, the asymmetry simply became more pronounced. At $R_o = -99.25$, our code detected a turning point, and that the asymmetric four-cell solution lost stability as the turning point was traversed.

Similarly, there were no bifurcation points along the branch of unstable symmetric four-cell flows as R_o was increased $R_o = -189$, until a turning point was detected at $R_o = -34.1$.

4.3.2. Three-cell flows. Three-cell flows arise as disconnected branches of solutions in this apparatus, and as such they cannot be obtained by quasi-static increase in the rotation rates of

DOI: 10.1002/fld

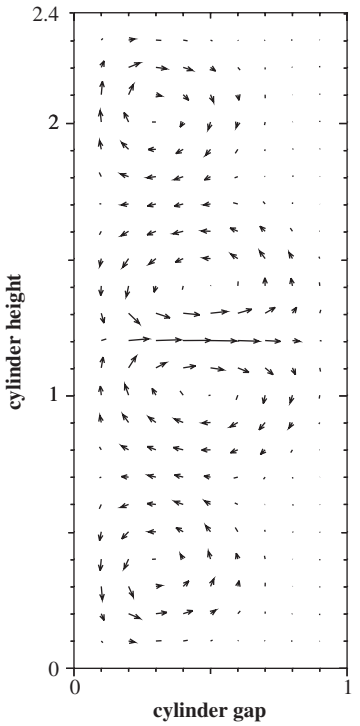

Figure 3. Asymmetric four-cell Taylor vortex flow velocity cross-section.

the inner and outer cylinder, but must be obtained through special choices of the initial conditions. Once a three-cell flow was established by these means at $R_i = 300$, $R_o = -85$ it was used as the starting solution for another limited set of numerical experiments. A spatial discretization of $\Delta r = \Delta z = 0.05$ was used for these calculations.

On increasing the outer Reynolds number from -85, our code detected a turning point near $R_o = 1$. Fixing $R_o = 0$ and decreasing the inner Reynolds number from $R_i = 300$, we found another turning point at $R_i = 295.4$. The location of these two turning points compares favourably with a more extensive independent numerical investigation of the three-cell conducted with the finite-element code ENTWIFE [19]. The small differences were assumed to be due to the different spatial discretizations employed in the two codes.

5. RESULTS FOR TIME-PERIODIC SOLUTIONS

5.1. Implementation details

The adapted code is also capable of computing time-periodic solutions and bifurcations along branches of those solutions. All of the previous implementation issues apply, but there are two additional considerations which arise for time-periodic solutions.

DOI: 10.1002/fld

First, the multiplicative action of the Monodromy matrix is approximated by a finite difference of the form

$$M\mathbf{v} \approx \frac{\boldsymbol{\phi}(\mathbf{u}_0 + \varepsilon\mathbf{v}, T^v) - \boldsymbol{\phi}(\mathbf{u}_0, T^v)}{\varepsilon} \tag{26}$$

Unlike the equilibrium case, where evaluating the finite difference requires only two calls to the time-stepper map $\mathbf{F}$, evaluating the right-hand side of (26) requires integrating over the entire periodic orbit. Since typically we performed computations using $\Delta t = 0.00005$, several thousand applications of the time-stepper map $\mathbf{F}$ were required to approximate $M\mathbf{v}$. Thus, time-periodic solutions were much more computationally expensive than equilibrium solutions.

Secondly, a phase condition is required to fix the point at time $t = 0$, since any point on the periodic orbit can be used as the starting point. Rather than using an explicit phase condition of the form $\psi(\mathbf{u}_0, T) = 0$ and solving Equation (16), we instead implicitly imposed a phase condition by solving the under-determined system

$$[V_p^{\mathrm{T}}(M^v - I)V_p \;\; V_p^{\mathrm{T}}\boldsymbol{\phi}_T^v] \begin{bmatrix} \Delta\bar{\mathbf{p}}^v \\ \Delta T^v \end{bmatrix} = -[V_p^{\mathrm{T}}[\mathbf{r}(\mathbf{u}_0^v, T^v) + M^v V_q \Delta\bar{\mathbf{q}}^v]] \tag{27}$$

by a least-squares method, utilizing LAPACK routines which solve the least squares problem using the SVD of the matrix. The least squares solution returned is the one with minimal norm, and it is this imposition of minimal Euclidean norm that implicitly determines a phase condition. See Lust [6] for more details on how this condition is close to an optimal one.

5.2. *Verification*

As in the equilibrium case, the cylinder geometry $\eta = 0.615$, $\Gamma = 2.4$ and a spatial discretization $\Delta r = \Delta z = 0.1$ was used to verify the adapted code's ability to calculate the period of a time-periodic solution. For this cylinder geometry, the solution at $(R_\mathrm{i}, R_\mathrm{o}) = (300, -91)$ is periodic. We can observe this from the time series plots (generated by TAY) of the azimuthal velocity at one particular gridpoint, in this case at $(r, \theta, z) = (0.3, 0, 1.1)$. The time series plot is given in Figure 4, and from this we estimate that the period is approximately 0.133, in terms of non-dimensional time units.

To compute the period of this solution using the Newton–Picard method, the solution from TAY at an arbitrary time was used as the starting solution. The period computed by the adapted code is 0.132, consistent with the period derived from the time series plot. Note that because of the implicit phase condition, the solution vector returned by the adapted code will generally be different from the starting solution vector, and thus a direct comparison between solution vectors was not made, unlike the equilibrium case.

5.3. *Numerical results*

Numerical experiments for time-periodic solutions were conducted using the same geometric parameters and error tolerances as the equilibrium solution experiments. In particular, we studied a Taylor–Couette geometry with radius ratio $\eta = 0.615$ and aspect ratio $\Gamma = 2.4$ on a finite difference grid with $\Delta r = \Delta z = 0.1$.

DOI: 10.1002/fld

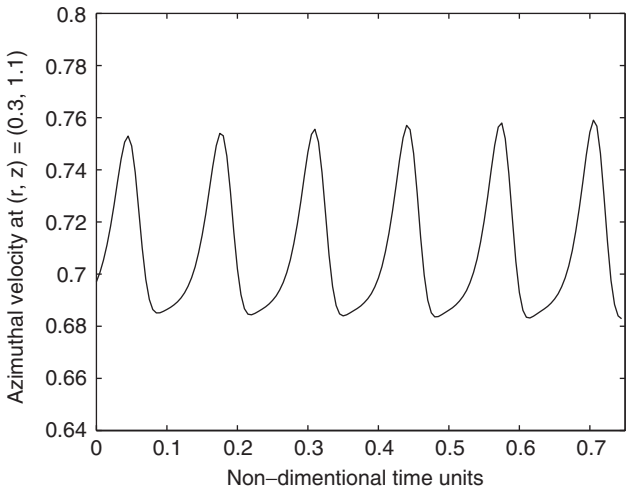

Figure 4. Azimuthal velocity time series plot at gridpoint $(r, z) = (0.3, 1.1)$ for the solution at $R_i = 300$, $R_o = -91$.

We began with a flow observed at $R_i = 300$, $R_o = -80$, which is a stable, steady two-cell Taylor vortex flow. Like the four-cell flows before, this two-cell flow is symmetric about the midplane. As the value of R_o was decreased from -80 to -91, a Hopf bifurcation was detected at $R_o = -90.6$. A complex conjugate pair of eigenvalues crossed the line $\Re e(z) = 1$ and the branch of steady two-cell Taylor vortex flows lost stability to a branch of time-periodic solutions.

As described above, the period of the solution at $R_o = -91$ was computed to be 0.132, which is consistent with the period calculated by examining a time series plot produced by the original simulation code. The initial period of solutions along the time-periodic branch is related to the imaginary part of the complex conjugate eigenpair at the Hopf bifurcation point [20], and an initial period of 0.132 was predicted, again in very close agreement with our results.

The code ENTWIFE was again used for the purposes of verification. In this instance the code ENTWIFE also detected a Hopf bifurcation point along this branch of two-cell flows, but at a slightly different outer Reynolds number, specifically at $R_o = -98.5$. This represents an 8.7% relative difference between our results and those of ENTWIFE, which difference was believed to be due to the different spatial resolution of the flow in the two codes.

Following this branch of time-periodic solutions, we found that a complex conjugate pair of Floquet multipliers crosses the unit circle at approximately $R_o = -92.5$. This implies that there is a bifurcation to motion on a torus, and that the time-periodic solution branch loses stability to a branch of time-dependent solutions with two incommensurate frequencies. Because the time-dependent solution branch emanating from the torus bifurcation point is stable, we could use the TAY code to obtain more information about it. Figure 5 shows a time series plot of the azimuthal velocity at a fixed gridpoint for $R_o = -94$, just beyond the bifurcation point. It is clear the solution is no longer simply periodic. It is difficult to discern by visual inspection how many independent frequencies appear in Figure 5. Applying a discrete Fourier transform to the data, using 560 data points, we obtained the power spectrum pictured in Figure 6. It is evident that there are two well-defined peaks, indicating two frequencies and confirming the torus bifurcation.

DOI: 10.1002/fld

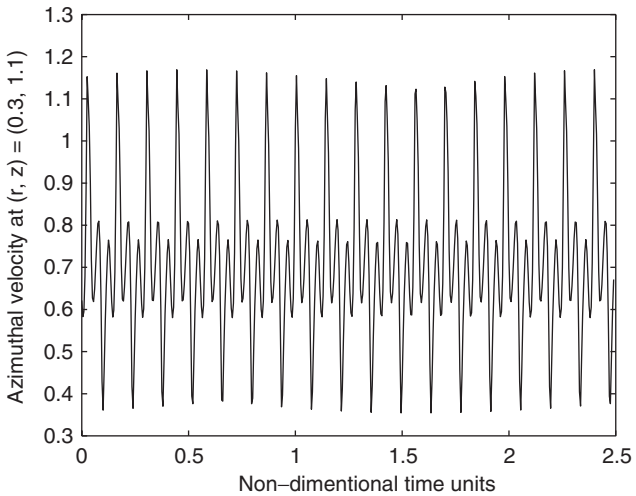

Figure 5. Azimuthal velocity time series plot at gridpoint $(r, z) = (0.3, 1.1)$ for the solution at $R_o = -93$.

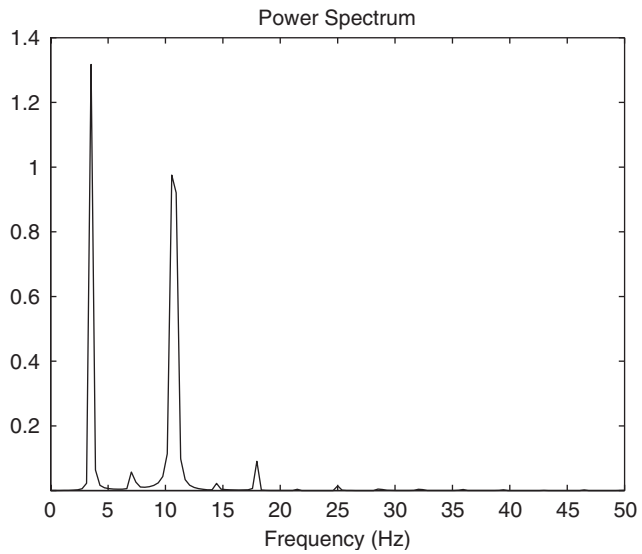

Figure 6. Power spectrum *via* finite Fourier transform of the time series plot in Figure 5.

The adapted code cannot follow the branch of solutions emanating from the torus bifurcation point; it only can detect when this bifurcation occurs. The reason is that the solutions are not truly time-periodic (in exact arithmetic), and the Newton–Picard method is only applicable to periodic or equilibrium solutions.

We instead can follow the branch of unstable time-periodic solutions, something that, to our knowledge, no other Taylor–Couette code is able to do. Further decreasing the outer Reynolds

DOI: 10.1002/fld

number, we compute that a single real Floquet multiplier crosses the unit circle through -1 at $R_o \approx -95.6$. This indicates that the unstable time-periodic solution undergoes a period-doubling bifurcation, resulting in an unstable time-periodic solution with twice the period.

6. CONCLUSIONS

The Newton–Picard and related methods have been proposed as computational wrappers around existing time-stepper codes, which can treat the time-stepper code as a black-box about which little needs to be known [3, 6]. The main goal of this work was to test this claim for a large-scale application when the time-stepper solves a partial differential equation. We conclude, in the case of the TAY time-stepper code which solves the Navier–Stokes equations in cylindrical coordinates, that this is not quite the case.

When implemented as a true black-box wrapper, early versions of the code calculated spurious eigenvalues of large modulus resulting in an incorrect basis for the invariant subspace $\mathscr{P}$ and the algorithm to fail to converge. It was only through a complete understanding of the numerical method implemented within the TAY code that we were able to determine the cause of these spurious eigenvalues. In general when using a true black-box time-stepper, such intimate knowledge is either unobtainable or difficult to extract.

Once the eigenvalue calculations were corrected, the resulting program was too slow to make a practical bifurcation code. Segments of the TAY solver routine, in particular the Poisson equation solver, were rewritten in order to increase the speed of the code. The resulting adapted program cannot therefore be considered to be a true wrap-around of the TAY code. Knowledge about the time-stepper's numerical algorithms and the ability to rewrite parts of the code were required in order to obtain a useful computational tool.

The Newton–Picard method works well for problems where the spectrum of the linearized operator is well separated. Unfortunately, the linearization of the TAY code produces spectra where many eigenvalues are clustered near 1. However, there appears to be no way to tell *a priori* whether a given black-box time-stepper will produce a well-separated spectrum, unless one understands the details of its computational techniques.

While our results suggest that *some* information about the time-stepper code and/or its solution algorithm is required to form a working adapted code, the forgoing discussion is not meant to suggest that these Newton–Picard methods are of little value. In fact, these methods can be quite powerful when coupled with a time-stepper code that has been written with the intent of using it within a Newton–Picard method. The work of Love on the Taylor–Couette problem and Kolmogorov flows [21] and the work of Tiesinga *et al*. on the driven cavity problem [22] are examples where a time-stepper was written specifically to be used in conjunction with these methods. We conclude that it remains unclear whether these methods can be applied successfully to an *arbitrary* large-scale time-stepper code, one that was not written necessarily to take specific advantage of these Newton–Picard methods.

Our work has illuminated many of the difficulties in adapting an existing time-stepper code by using the Newton–Picard method. Others who wish to accomplish similar adaptations of time-steppers using the Newton–Picard method may benefit from our results. In particular, one should be aware of the following.

1. A small value of Δt will cause eigenvalue clustering near 1, regardless of whether an explicit or implicit method is used in the time-stepper. This clustering presents the most severe

DOI: 10.1002/fld

obstacle for large-scale implementation of the Newton–Picard method, for the following reasons:

(a) The clustering causes the original iteration scheme to be slowly convergent, since the rate of convergence is related to the spectral radius of the Jacobian of the iteration map.
(b) The computation of the dangerous eigenvalues *via* ARPACK is more expensive when the eigenvalues are clustered. A fixed basis size for $\mathscr{P}$ is recommended in this case.
(c) The $\mathscr{Q}$-equations are poorly conditioned when the eigenvalues are clustered around 1, requiring their solution to be several orders of magnitude more accurate than the convergence criterion of the Newton iteration.

2. Some information about the solution scheme used in the time-stepper must be known to correctly interpret the eigenvalue information returned by the adapted code. This requires that connections between eigenvalues of the Jacobian of the iteration map and the stability of solutions of the original dynamical system be established. For example, we needed to know the details of the Chorin–Témam projection scheme to correctly identify the iteration map as a mapping of the velocity field only, rather than the velocity–pressure field.
3. It is an open question whether time-steppers for even larger problems (say $N = O(10^4)$ or $O(10^5)$) are amenable to adaptation by the Newton–Picard method, even if the eigenvalue clustering near 1 is not present.

APPENDIX A

In this section, we present some fundamental results from bifurcation and fixed point theory. These results highlight why time-steppers are poorly suited for performing bifurcation studies and what properties the modified method should possess to overcome their inherent limitations.

A.1. Equilibrium solutions

Consider again the parameter-dependent dynamical system

$$\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t} = \mathbf{f}(\mathbf{u}, \lambda) \tag{A1}$$

where for each t, $\mathbf{u}(t) \in \mathbb{R}^N$ and $\mathbf{f} : \mathbb{R}^N \times \mathbb{R} \to \mathbb{R}^N$. We assume that $\mathbf{f}$ is autonomous.

The main result which governs the stability of equilibrium solutions is presented in the following theorem, which is an amalgamation of [20, Theorems 9.5 and 9.7].

Theorem A.1
If all eigenvalues of the Jacobian matrix $\mathbf{f}_\mathbf{u}(\mathbf{u}^*, \lambda^*)$ have negative real parts, then the equilibrium solution $\mathbf{u}^*$ of (A1) is asymptotically stable. If at least one eigenvalue of $\mathbf{f}_\mathbf{u}(\mathbf{u}^*, \lambda^*)$ has positive real part, then $\mathbf{u}^*$ is unstable.

Note that an equilibrium solution $\mathbf{u}^*$ can lose stability at a bifurcation point as the parameter λ is varied when either a real eigenvalue or a complex conjugate pair of eigenvalues crosses the imaginary axis, at (generically) a limit point or a Hopf bifurcation point, respectively.

DOI: 10.1002/fld

A.2. Mathematics of time-steppers

A time-stepper code to solve Equation (A1) is obtained by a suitable discretization of the temporal derivative $\mathrm{d}\mathbf{u}/\mathrm{d}t$. After imposing this discretization, we can view the time-stepper as an iterative mapping of the form

$$\mathbf{u}^{n+1} = \mathbf{F}(\mathbf{u}^n, \lambda), \quad \mathbf{F} : \mathbb{R}^N \times \mathbb{R} \to \mathbb{R}^N \tag{A2}$$

where the variable n counts the time step. Equilibrium solutions $\mathbf{u}^*$ of (A1) correspond to fixed points of $\mathbf{F}$ and can be approximated by iterating (A2) until the norm $\|\mathbf{u}^{n+1} - \mathbf{u}^n\|$ of successive iterates falls below a user-defined tolerance.

The convergence of iteration (A2) is intimately related to the eigenvalues of the Jacobian matrix $\mathbf{F}_\mathbf{u}(\mathbf{u}^*)$. Indeed, if the spectral radius of the Jacobian matrix is strictly less than one, then the iteration will converge (locally) to a fixed point $\mathbf{u}^*$ as a consequence of the following more general result.

Theorem A.2 (*Adapted from Krasnosel'skii et al.* [*23, p. 21*])
Let $\mathbf{F}$ be an operator from a Banach space into itself and let $\mathbf{u}^*$ be a fixed point of $\mathbf{F}$. If $\mathbf{F}$ is Fréchet differentiable at $\mathbf{u}^*$ and if the spectral radius $\rho(\mathbf{F}_\mathbf{u}(\mathbf{u}^*))<1$, then the sequence $\{\mathbf{u}^n\}$ defined by

$$\mathbf{u}^{n+1} = \mathbf{F}(\mathbf{u}^n), \quad n = 0, 1, \ldots \tag{A3}$$

converges to $\mathbf{u}^*$, provided that $\mathbf{u}^0$ is sufficiently close to $\mathbf{u}^*$. Moreover, for every $\varepsilon>0$, there exists a constant C, depending only on $\mathbf{u}^0$ and ε but not on n, such that

$$\|\mathbf{u}^n - \mathbf{u}^*\| \leqslant C\,(\rho(\mathbf{F}_\mathbf{u}(\mathbf{u}^*)) + \varepsilon)^n \tag{A4}$$

Conversely, if the spectral radius $\rho(\mathbf{F}_\mathbf{u}(\mathbf{u}^*)) \geqslant 1$, then the sequence $\{\mathbf{u}^n\}$ defined above does not converge to $\mathbf{u}^*$.

To establish a connection between bifurcations of equilibrium solutions of (A1) and convergence properties of the iteration (A2), (see also [24]) note that Theorems A.1 and A.2 show that the eigenvalues of both Jacobian matrices $\mathbf{f}_\mathbf{u}$ and $\mathbf{F}_\mathbf{u}$ play critical roles.

If we denote the eigenvalues of $\mathbf{f}_\mathbf{u}$ by $\{\alpha_k\}$ and the corresponding eigenvalues of $\mathbf{F}_\mathbf{u}$ by $\{\mu_k\}$, $k = 1, 2, \ldots, N$, then the eigenvalues of the two Jacobian matrices are related by

$$\mu_k = 1 + \Delta t\,\alpha_k \tag{A5}$$

From this, we obtain the following result which connects the location of eigenvalues of the Jacobian matrix $\mathbf{F}_\mathbf{u}$ with stability of equilibrium solutions of (A1).

Proposition A.3
With the above cumulative notation:

(a) Iteration (A2) is convergent if and only if

$$\Re e(\alpha_k) < -\frac{\Delta t}{2}|\alpha_k|^2 \tag{A6}$$

DOI: 10.1002/fld

for all $k = 1, \ldots, N$. In particular, if iteration (A2) is convergent with fixed point $\mathbf{u}^*$, then the equilibrium solution $\mathbf{u}^*$ of (A1) is stable.

(b) If at least one eigenvalue μ of the Jacobian matrix $\mathbf{F}_\mathbf{u}(\mathbf{u}^*)$ has real part $\Re e(\mu) > 1$, then the equilibrium solution $\mathbf{u}^*$ of (A1) is unstable. Moreover, the equilibrium solution $\mathbf{u}^*$ loses stability if and only if an eigenvalue μ of $\mathbf{F}_\mathbf{u}$ crosses the line $\Re e(z) = 1$ in the complex plane.

Notice that the criteria for stability of the equilibrium and of the fixed point iteration are not identical. If the eigenvalue of the Jacobian $\mathbf{f}_\mathbf{u}(\mathbf{u}^*)$ closest to the unstable right half of the complex plane is real, then this is not an issue, but if $(\mathbf{u}^*)$ is a stable equilibrium at which the Jacobian has a conjugate eigenvalue pair with small negative real part but large complex part, the fixed point iteration may none-the-less be unstable if Δt is too large.

The fact that time-steppers cannot compute unstable solutions follows immediately from part (a) of Proposition A.3. That the rate of convergence of the time-stepper becomes arbitrarily slow near bifurcation point follows from the estimate (A4) and the fact that near bifurcation points the spectral radius of $\mathbf{F}_\mathbf{u}$ is near 1.

A.3. Time-periodic solutions

Let $\mathbf{u}^*(t)$ be a time-periodic solution of (A1) with period T and initial condition $\mathbf{u}^*(0) = \mathbf{u}_0$. The stability of $\mathbf{u}^*(t)$ is determined by the eigenvalues of the Jacobian matrix $\boldsymbol{\phi}_\mathbf{u}(\mathbf{u}^*, T)$, where $\boldsymbol{\phi}(\mathbf{u}^*, T)$ is the flow of the system at a time T with initial condition $\mathbf{u}^*$. This matrix is called the Monodromy matrix and its eigenvalues are termed Floquet multipliers. The main result which governs the stability of $\mathbf{u}^*(t)$ is given below.

Theorem A.4 (*from Seydel* [*11*])
For fixed parameter λ, let $\mathbf{u}^*(t)$ be a time-periodic solution of (A1) with period T, and let M denote the Monodromy matrix $\boldsymbol{\phi}_\mathbf{u}(\mathbf{u}^*, T)$. Denote the N Floquet multipliers by $\mu_1, \ldots, \mu_N$, repeated according to multiplicity. One of them is identically equal to 1, say $\mu_N = 1$. The other $N - 1$ Floquet multipliers determine the local stability of $\mathbf{u}^*(t)$ as follows. If $|\mu_k| < 1$ for all $k = 1, \ldots, N-1$, then $\mathbf{u}^*(t)$ is stable. If at least one Floquet multiplier μ_k for some k has modulus greater than 1, then $\mathbf{u}^*(t)$ is unstable.

The particular time-stepper for the Taylor–Couette problem we are studying is of the form

$$\mathbf{u}^{n+1} = \mathbf{u}^n + \Delta t \mathbf{f}(\mathbf{u}^n, \lambda) \equiv \mathbf{F}(\mathbf{u}^n, \lambda) \tag{A7}$$

Finding a time-periodic solution of (A1) amounts to finding a fixed point of the map $\mathbf{G} = \mathbf{F}^k$ for some positive integer k, where $\mathbf{F}^k$ denotes the k-fold composition of $\mathbf{F}$ with itself. The value of k is chosen so that $k\Delta t \approx T$, say $k = [T/\Delta t]$, where $[x]$ represents the greatest integer less than or equal to x. In our numerical experiments, the value of Δt is very small, which makes the corresponding value of k quite large, typically on the order of several thousand. An alternative and preferrable strategy (see e.g. [25]) when Δt is not required to be small is to fix k and vary Δt so as to enable T to be approximated more exactly.

Eigenvalues of the Jacobian matrix $\mathbf{G}_\mathbf{u}(\mathbf{u}^*)$ are approximations to the Floquet multipliers of the Monodromy matrix $\boldsymbol{\phi}_\mathbf{u}(\mathbf{u}^*, T)$. A bifurcation occurs if at least one Floquet multiplier crosses the unit circle in the complex plane. The type of bifurcation depends on the number of Floquet

DOI: 10.1002/fld

multipliers crossing the unit circle and how they cross it [20]. A real Floquet multiplier can cross at $+1$ resulting in a turning point in the path of periodic solutions, a real Floquet multiplier can cross at -1 resulting in a period-doubling bifurcation, or a complex conjugate pair of Floquet multipliers can cross at $e^{\pm i\theta}$ for some value of θ that is not an integer multiple of π leading to motion on a torus.

REFERENCES

1. Tuckerman LS, Barkley D. Bifurcation analysis for timesteppers. In *Numerical Methods for Bifurcation Problems and Large-Scale Dynamical Systems*, Doedel E, Tuckerman LS (eds), IMA Volumes in Mathematics and its Applications, vol. 119. Springer: Berlin, 2000.
2. Koronaki ED, Boudouvis AG, Kevrekidis IG. Enabling stability analysis of tubular reactor models using PDE/PDAE integrators. *Computers and Chemical Engineering* 2003; **27**:951–964.
3. Shroff GM, Keller HB. Stabilization of unstable procedures: the Recursive Projection Method. *SIAM Journal on Numerical Analysis* 1993; **30**(4):1099–1120.
4. Adair RH. Simulations of Taylor–Couette flow. *Ph.D. Thesis*, Colorado State University, 1997.
5. Lust K, Roose D, Spence A, Champneys AR. An adaptive Newton–Picard algorithm with subspace iteration for computing periodic solutions. *SIAM Journal on Scientific Computing* 1998; **19**(4):1188–1209.
6. Lust K. Numerical bifurcation analysis of periodic solutions of partial differential equations. *Ph.D. Thesis*, Katholieke Universiteit Leuven, 1997.
7. Lust K, Roose D. Newton–Picard methods with subspace iteration for computing periodic solutions of partial differential equations. *Zeitschrift für Angewandte Mathematik und Mechanik* 1996; **76**(Suppl. 2):605–606.
8. Lust K, Roose D. Computation and bifurcation analysis of periodic solutions of large-scale systems. In *Numerical Methods for Bifurcation Problems and Large-Scale Dynamical Systems*, Doedel E, Tuckerman LS (eds), IMA Volumes in Mathematics and its Applications, vol. 119. Springer: Berlin, 2000.
9. Roose D, Lust K, Champneys A, Spence A. A Newton–Picard shooting method for computing periodic solutions of large-scale dynamical systems. *Chaos, Solitons and Fractals* 1995; **5**(10):1913–1925.
10. Trefethen LN, Bau III D. *Numerical Linear Algebra*. SIAM: Philadelphia, 1997.
11. Seydel R. *From Equilibrium to Chaos*: *Practical Bifurcation and Stability Analysis*. Elsevier: New York, Amsterdam, London, 1988.
12. Chorin AJ. Numerical solution of the Navier–Stokes equations. *Mathematics of Computation* 1968; **22**(104): 745–762.
13. Témam R. Sur l'approximation de la solution des équations de Navier–Stokes par la méthode des pas fractionaires. *Archive for Rational Mechanics and Analysis* 1969; **32**:135–153.
14. Griebel M, Dornseifer T, Neunhoeffer T. *Numerical Simulation in Fluid Dynamics*: *A Practical Introduction*. SIAM: Philadelphia, 1998.
15. Lehoucq RB, Sorensen DC, Yang C. ARPACK *User's Guide*: *Solution of Large Scale Eigenvalue Problems with Implicitly Restarted Arnoldi Methods*. SIAM: Philadelphia, 1998. ARPACK software available at http://www.caam.rice.edu/software/ARPACK
16. Mullin T, Toya Y, Tavener SJ. Symmetry breaking and multiplicity of states in small aspect ratio Taylor–Couette flow. *Physics of Fluids* 2002; **14**(8):2778–2787.
17. Schulz A, Pfister G, Tavener SJ. The effect of outer cylinder rotation on Taylor–Couette flow at small aspect ratio. *Physics of Fluids* 2003; **15**(2):417–425.
18. Keller HB. Numerical solution of bifurcation and nonlinear eigenvalue problems. In *Applications of Bifurcation Theory*, Rabinowitz PH (ed.). Academic Press: New York, 1997.
19. Cliffe KA. *ENTWIFE* (*Release 6.3*) *Reference Manual*: *ENTWIFE*, *INITIAL DATA AND SOLVER DATA COMMANDS*, 1996. See http://www.entwife.com
20. Hale J, Koçak H. Dynamics and bifurcations. In *Texts in Applied Mathematics*, vol. 3. Springer: Berlin, New York, Heidelberg, 1991.
21. Love P. Bifurcations in Kolmogorov and Taylor-vortex flow. *Ph.D. Thesis*, California Institute of Technology, 1998.
22. Tiesinga G, Wubs FW, Veldman AEP. Bifurcation analysis of incompressible flow in a driven cavity by the Newton–Picard method. *Journal of Computational and Applied Mathematics* 2002; **140**:751–772.

DOI: 10.1002/fld

23. Krasnosel'skii MA, Vainikko GM, Zabreiko PP, Rutitskii YB, Stetsenko VY. *Approximate Solution of Operator Equations*. Wolters-Noordhoff Publishing: Groningen, 1972.
24. Davidson BD. Large-scale continuation and numerical bifurcation for partial differential equations. *SIAM Journal on Numerical Analysis* 1997; **34**(5):2008–2027.
25. Van Leemput P, Lust KWA, Kevrekidis IG. Coarse-grained numerical bifurcation analysis of lattice Boltzmann models. *Physica D* 2005; **210**:58–76.

DOI: 10.1002/fld