Physica D 252 (2013) 22–33

Contents lists available at SciVerse ScienceDirect

Physica D

journal homepage: www.elsevier.com/locate/physd

A parallel algorithm for the computation of invariant tori in large-scale dissipative systems

J. Sánchez *, M. Net

Dept. de Física Aplicada, Universitat Politècnica de Catalunya, Jordi Girona Salgado s/n., Campus Nord. Mòdul B4, 08034 Barcelona, Spain

HIGHLIGHTS

- A parallel algorithm to compute invariant tori of large-scale systems is presented.
- The new method is compared with a previous serial algorithm.
- The two methods are applied to the thermal convection of a binary mixture.
- Speed-ups around the number of processors employed are achieved for the new method.
- The comparison with the old method shows speed-ups above the number of processors.

ARTICLE INFO

Article history:
Received 13 June 2012
Received in revised form
11 February 2013
Accepted 18 February 2013
Available online 27 February 2013
Communicated by H.A. Dijkstra

Keywords:
Invariant tori
Parallel algorithms
Continuation methods
Generalized Poincaré maps
Variational equations
Newton–Krylov methods

ABSTRACT

A parallelizable algorithm to compute invariant tori of high-dimensional dissipative systems, obtained upon discretization of PDEs is presented. The size of the set of equations to be solved is only a small multiple of the dimension of the original system. The sequential and parallel implementations are compared with a previous method (Sánchez et al. (2010)) [11], showing that important savings in wall-clock time can be achieved. In order to test it, a thermal convection problem of a binary mixture of fluids has been used. The new method can also be applied to problems with very low rotation numbers, for which the previous is not suitable. This is tested in two examples of two-dimensional maps.

1. Introduction

The spatial discretization of systems of elliptic–parabolic partial differential equations leads to large-scale systems of differential-algebraic equations (DAEs) which, in many cases, take the form

$$M\dot{x} = f(x, \lambda), \quad (x, \lambda) \in \mathcal{U} \subset \mathbb{R}^n \times \mathbb{R}, \tag{1}$$

where M is a linear operator, possibly singular, n the dimension, and λ a distinguished parameter. The computation of its invariant manifolds is essential to understand the global dynamics induced by the system (see [1] for a recent review of applications in Fluid Mechanics). Algorithms to find fixed points [2–6], periodic orbits [7–10] and invariant tori [11] of large-scale problems, and to study their stability [12–15] have been developed in the past. The calculation of coefficients of normal forms at fixed points and periodic orbits has also been considered, for instance, in [16,17], and that of portions of two-dimensional unstable manifolds of periodic orbits in [18].

The method to compute invariant two-dimensional tori described in [11] consists in finding a single point of the invariant curve of a Poincaré section located at its intersection with a second transversal section. The point is found as the fixed point of a *synthesized* map, which is defined as the intersection of the second section with the curve interpolating the first powers of the Poincaré map of the initial point, which fall inside a small ball centered at it. Due to this definition high powers of the Poincaré map may be necessary to complete the computation of the *synthesized* map, for instance, in the case of low rotation numbers. Moreover the computation of the required powers is a sequential process.

A parallel algorithm is proposed here, which reduces the computing time of the previous. It consists in finding a set of points which interpolate an arc of the invariant curve in a Poincaré section. The invariance condition is now that the interpolating curve,

* Corresponding author. Tel.: +34 934017981; fax: +34 934016090.
E-mail addresses: juan.j.sanchez@upc.edu, sanchez@fa.upc.edu (J. Sánchez).

0167-2789/$ – see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.physd.2013.02.008

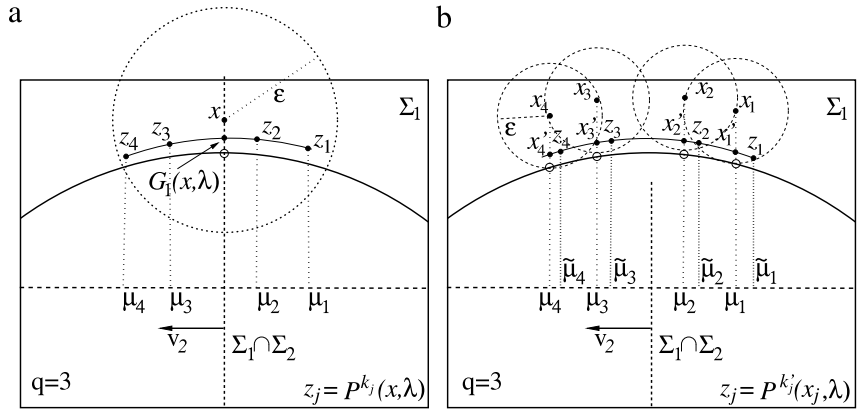

Fig. 1. Schemes of the maps used to compute invariant tori. In the two plots the long solid line represents the invariant curve in Σ_1. (a) For the map G_1, the intersection with $\Sigma_1 \cap \Sigma_2$ (empty circle) is the point we want to approximate. (b) For G_2, the desired points (empty circles) are those at the stations μ_1 to μ_{q+1}. The interpolation degree is $q = 3$ in both schemes. In (a) the distance between x and the curve containing the points z_j would be small if x is a good prediction of a point on the torus. It has been exaggerated in the figure for clarity purposes. The same holds for the distances between the x_j and the curve passing through the z_j in (b).

passing through the first power of each point of the initial set, and falling near to them, be close to the initial arc. Several implementations are possible, three of which are detailed in the text and final appendices. Since only the first returning power of each point is needed, and they are independent of each other, the computational cost decreases, and they can be calculated in parallel. Moreover, the new method can also be applied to very low-rotation-number cases, for which the first method is too expensive or could be not suitable due to extrapolation errors. The new algorithm is based on previous ideas for low-dimensional systems, which can be found, among others, in [19–22]. The details on how to compute Poincaré maps, and the action of their Jacobians were given in [11], and will not be repeated here. For this purpose, it is assumed that a time integrator is available to solve the initial value problem (1), and the corresponding first variational equation

$$M\dot{y} = D_x f(x,\lambda)y + D_\lambda f(x,\lambda)\mu, \quad (y,\mu) \in \mathbb{R}^n \times \mathbb{R}, \tag{2}$$

for given initial conditions $x(0) = x_0$ and $y(0) = y_0$, and a constant μ. Only the action of the Jacobian of a Poincaré map is required because matrix-free implementations of Newton–Krylov methods are used to solve the nonlinear systems.

In order to perform the numerical experiments, we have applied the method to the thermal convection of a binary mixture, filling a two-dimensional rectangular domain. It is known that the onset of convection is subcritical and oscillatory below a determined negative value of the separation ratio, and that, depending on the aspect ratio, it can give rise to very complex dynamics.

To see that the method can be used for very low rotation numbers, some tests have also been performed for two two-dimensional maps.

Section 2 describes the previous and new algorithms to compute invariant tori. The main test problem is presented in Section 3, with a summary of previous results in Section 4. The results of the comparison of the two methods occupy Section 5, and the two-dimensional examples and the conclusions Sections 6 and 7.

2. Two algorithms to compute invariant two-tori of ODEs

The intersection of an invariant two-dimensional torus with a hyperplane Σ_1, which cuts it transversally is locally (near one of the points of the intersection) an arc of a curve. The algorithms described here try to compute a point on this arc, or the discretization of an small portion of the arc. The first was described in detail in [11] but has also been included here for comparison purposes. In both cases a map G will be defined such that the solution is obtained by applying Newton–Krylov methods to the equation

$$x - G(x,\lambda) = 0. \tag{3}$$

The reason for the fast convergence of the linear solvers for this particular problem, and for the computation of periodic orbits was given in [11,8]. The essential idea is that, since the system is dissipative, as for the integration of the systems of DAEs considered, the spectrum of G is tightly clustered around zero so that the Jacobian $I - D_x G(x,\lambda)$ is near a low-rank perturbation of the identity. In this situation Krylov methods (GMRES [23] for instance) are known to converge in a low number of iterations.

2.1. The first method

Let $P : \mathcal{V} \subset \Sigma_1 \to \Sigma_1$ be the Poincaré map defined on the hyperplane Σ_1 given by $v_1^\top(x - x^{\Sigma_1}) = 0$, with $\|v_1\|_2 = 1$. The intersection of Σ_1 with an invariant torus of the system (1) is invariant under the map P. Let Σ_2 be another hyperplane, given by $v_2^\top(x - x^{\Sigma_2}) = 0$, with $\|v_2\|_2 = 1$, transversal to both Σ_1 and the invariant two-torus. Then we define a map

$$G_1(x,\lambda) : \mathcal{U}_1 \subset (\Sigma_1 \cap \Sigma_2) \times \mathbb{R} \to \Sigma_1 \cap \Sigma_2, \tag{4}$$

such that its fixed points are intersections of the invariant curve and Σ_2, as follows (see Fig. 1a). If x is a point on the intersection $\Sigma_1 \cap \Sigma_2$, and $B(x,\varepsilon)$ is the ball of radius ε centered at x, a time integration of (1) with initial condition x is started to find the first $q+1$ powers ($q = 3$ in Fig. 1a) of the Poincaré map

$$z_j = P^{k_j}(x,\lambda), \quad \text{with } j = 1,\dots,q+1 \text{ and}$$
$$k_1 < k_2 < \cdots < k_{q+1}, \tag{5}$$

which fall inside $B(x,\varepsilon)$. Then the intersection of Σ_2 with the polynomial which interpolates these points defines the map G_1. If

$$\mu_j = v_2^\top(P^{k_j}(x,\lambda) - x^{\Sigma_2}), \quad j = 1,\dots,q+1, \tag{6}$$

are the projections of the points z_j onto the line $x = x^{\Sigma_2} + \mu v_2$, then

$$G_1(x,\lambda) = \sum_{j=1}^{q+1} l_j(0) z_j, \tag{7}$$

where the $l_j(0)$ are the Lagrange interpolation polynomials of degree q, based on the nodes μ_j, evaluated at $\mu = 0$.

By differentiating (7) the following expression for the action of the Jacobian of G_1 is obtained:

$$DG_1(x,\lambda)(\Delta x,\Delta\lambda)=\sum_{i=1}^{q+1}\left[l_i(0)DP^{k_i}(x,\lambda)+P^{k_i}(x,\lambda)\times\sum_{j=1}^{q+1}\partial_{\mu_j}l_i(0)v_2^TDP^{k_j}(x,\lambda)\right](\Delta x,\Delta\lambda). \tag{8}$$

Hence the action of the Jacobian of G_1 on $(\Delta x,\Delta\lambda)$ reduces to that of the Poincaré map.

It must be noticed that x and Δx in Eqs. (7) and (8) are in manifolds of dimension $n-2$, which must be parameterized. This is done as follows. Suppose that j_1 is the index of the largest component of v_1, j_2 is that of v_2 different from j_1, and let $R:\Sigma_1\cap\Sigma_2\to\mathbb{R}^{n-2}$ be the orthogonal projection from $\Sigma_1\cap\Sigma_2$ onto the subspace $\{x\in\mathbb{R}^n\mid x_{j_1}=0,\ x_{j_2}=0\}\equiv\mathbb{R}^{n-2}$. The diffeomorphism R drops the components j_1 and j_2 from a point in $\Sigma_1\cap\Sigma_2$, and its inverse fills these components such that the resulting point is in $\Sigma_1\cap\Sigma_2$. Then instead of working with the map G_1 we work with

$$\bar G_1(\bar x,\lambda)=R(G_1(R^{-1}(\bar x),\lambda)):\bar{\mathcal{U}}_1\times\mathbb{R}\subset\mathbb{R}^{n-2}\times\mathbb{R}\to\mathbb{R}^{n-2}.$$

The actions of the Jacobians of R and R^{-1} are trivial. The same kind of parameterization is needed for the next method, but we will skip these implementation details in the description.

2.2. The second method

The new method consists in finding a set of points which represents a discretization of an arc of the invariant curve (see Fig. 1b). In the same situation as for the previous map, let $\mu_1,\dots,\mu_{q+1}$ be $q+1$ fixed coordinates along the line $x=x^{\Sigma_2}+\mu v_2$, and Σ_2^j the hyperplanes given by $v_2^\top(x-x^{\Sigma_2})=\mu_j$ parallel to Σ_2. Then, if $\mathcal{H}=\Sigma_1\cap\Sigma_2^1\times^{q+1}\cdots\times\Sigma_1\cap\Sigma_2^{q+1}$ we define a map

$$G_2(X,\lambda):\mathcal{U}_2\subset\mathcal{H}\times\mathbb{R}\to\mathcal{H} \tag{9}$$

as follows. Given a value of ε, if $X=(x_1,\dots,x_{q+1})\in\mathcal{U}_2$, $q+1$ time integrations of (1) are started with initial conditions x_j, $j=1,\dots,q+1$, to find the first power of the Poincaré map on each x_j falling inside $B(x_j,\varepsilon)$. Let the powers be k_j', and let define $z_j=P^{k_j'}(x_j,\lambda)$, $Z=(z_1,\dots,z_{q+1})$ and $\tilde\mu_j=v_2^\top(z_j-x^{\Sigma_2})$, $j=1,\dots,q+1$ (see Fig. 1b). The points z_j are interpolated to find the polynomial of degree q

$$Q(\mu)=\sum_{i=0}^{q}\alpha_i\mu^i, \tag{10}$$

with $\alpha_i\in\mathbb{R}^n$. If $A=(\alpha_0,\dots,\alpha_q)\in\mathbb{R}^{n\times(q+1)}$, by imposing that the polynomial passes through $(\tilde\mu_j,z_j)$,

$$z_j=\sum_{i=0}^{q}\alpha_i\tilde\mu_j^i,$$

or in matrix form, $Z=A\tilde V$, with

$$\tilde V=\begin{pmatrix}1&\cdots&1\\ \tilde\mu_1&\cdots&\tilde\mu_{q+1}\\ \cdots&\cdots&\cdots\\ \tilde\mu_1^q&\cdots&\tilde\mu_{q+1}^q\end{pmatrix} \tag{11}$$

the Vandermonde matrix associated with the $\tilde\mu_j$, the coefficients of the polynomial are obtained as $A=Z\tilde V^{-1}$.

Evaluating Q at the positions μ_j to obtain $x_j'=Q(\mu_j)$ is computing the product $X'=AV$ with $X'=(x_1',\dots,x_{q+1}')$, and V the Vandermonde matrix associated with the μ_j. Therefore the definition of the new map G_2 is

$$G_2(X,\lambda)=X'=Z(X,\lambda)\tilde V(X,\lambda)^{-1}V, \tag{12}$$

where the dependence of Z and $\tilde V$ on X and λ has been explicited. Each fixed point of $G_2(X,\lambda)$ approximates an arc of the invariant curve in Σ_1.

The first method was described in terms of Lagrange interpolation, but can also be written in power form as in Eq. (10). Then $G_1(x,\lambda)=\alpha_0$.

The action of the Jacobian of $G_2=Z\tilde V^{-1}V$ also reduces, essentially, to that of the Poincaré map. The details are given in Appendix A. Here we only show the final result. If $\Delta X=(\Delta x_1,\dots,\Delta x_{q+1})$ then

$$\begin{aligned}DG_2(X,\lambda)(\Delta X,\Delta\lambda)=&\big[DZ(X,\lambda)(\Delta X,\Delta\lambda)\\&-Z(X,\lambda)\tilde V(X,\lambda)^{-1}D\tilde V(X,\lambda)(\Delta X,\Delta\lambda)\big]\\&\times\tilde V(X,\lambda)^{-1}V,\end{aligned} \tag{13}$$

where

$$\begin{aligned}DZ(X,\lambda)(\Delta X,\Delta\lambda)=&\,(DP^{k_1'}(x_1,\lambda)(\Delta x_1,\Delta\lambda),\dots,\\&DP^{k_{q+1}'}(x_{q+1},\lambda)(\Delta x_{q+1},\Delta\lambda)),\end{aligned}$$

$$D\tilde V(X,\lambda)(\Delta X,\Delta\lambda)=\begin{pmatrix}0&\cdots&0\\1&\cdots&1\\2\tilde\mu_1&\cdots&2\tilde\mu_{q+1}\\\cdots&\cdots&\cdots\\q\tilde\mu_1^{q-1}&\cdots&q\tilde\mu_{q+1}^{q-1}\end{pmatrix}\times\begin{pmatrix}\eta_1&\cdots&0\\\vdots&\ddots&\vdots\\0&\cdots&\eta_{q+1}\end{pmatrix}, \tag{14}$$

and $\eta_j=v_2^\top DP^{k_j'}(x_j,\lambda)(\Delta x_j,\Delta\lambda)$. In short, $DG_2=[DZ-Z\tilde V^{-1}D\tilde V]\tilde V^{-1}V$.

At the beginning of this section the fast convergence of GMRES to solve systems with matrix $M=I-D_xG(x,\lambda)$ when $D_xG(x,\lambda)$ is contractive was recalled. An explanation was given in the case of periodic orbits in [8], and in more general situations, when the spectrum of M consists of several close clusters of eigenvalues in [24]. Since G_2 involves the time evolution of parabolic equations it will be contractive, except along the unstable manifold of its fixed points, which we assume will have, at most, a low dimension. Moreover, the computation of G_2 and the action of its Jacobian can be done in parallel because the calculation of each $z_j=P^{k_j'}(x_j,\lambda)$ or $DP^{k_j'}(x_j,\lambda)(\Delta x_j,\Delta\lambda)$ is an independent process. In order to have an optimal parallelism we also need that the number of iterations required to solve systems with matrix $M=I-DG_2(X,\lambda)$ be almost independent of q. From [24] we know that, if M has clusters of eigenvalues, at each iteration of GMRES the residual is reduced by a factor, which depends only on a common relative radius of the discs containing each cluster and on the maximal distance between them. In addition, the asymptotic error constant depends on the maximal distance of the *outliers* (the eigenvalues out of the clusters) from the clusters. Therefore we have to see that when q changes, the number of clusters and their relative positions do not change. In Appendix B we give an outline of a proof of this fact. In any case, the numerical experiments have confirmed the almost independence of q of the number of iterations of the solver.

Instead of defining $z_j=P^{k_j'}(x_j,\lambda)$, with k_j' the first power of the Poincaré map on x_j falling inside $B(x_j,\varepsilon)$, it could be defined as the first power for which $P^{k_j'}(x_j,\lambda)\in B(x_i,\varepsilon)$ for some $1\le i\le q+1$. In addition, a condition on its projection onto the line $x=x^{\Sigma_2}+\mu v_2$

could also be required to ensure that it does not fall too far away from the initial μ_i. This would give rise to other variants of the method.

Polynomial interpolation has been used in the presentation of the method because it provides a simple expression for G_2 and its Jacobian. It is assumed that the degree will be small because the total dimension of the systems to be solved, $(q+1)(n-2)$, must be limited for the algorithms to be efficient. The maximum degree used in the numerical experiments has been five. In Appendix C we give the details on how to compute G_2, and the action of its Jacobian in case of spline interpolation, which could be used if, for some reason, the number of points $q+1$ required to approximate the arc must be large. For instance close to the breakdown of the tori when they lose regularity.

Another possibility is using least squares polynomial fitting. This can be specially useful in the case of unstable tori. The algorithm defined by the map G_2 in Eq. (12) computes the powers $z_j = P^{k'_j}(x_j, \lambda)$. If the torus is unstable (think of a two-dimensional unstable manifold for the invariant curve by the Poincaré map) the component of each z_j along the unstable direction can grow in any of the two opposite directions. Then the interpolation polynomial could be highly oscillatory giving rise to poor results. To avoid this, we describe in Appendix D how to compute the map, and the action of its Jacobian in the case of polynomial fitting.

The maps G_1 and any of the three versions of G_2 depend on the radius ε, and the degree of the interpolation polynomial. G_2 depends also on the position of the μ_i. Several strategies can be implemented to include adaptability when the parameter λ is changed during a continuation process, and to discard, for instance, a point too close to a previous one in case of G_1, which would introduce large errors in the interpolation.

In the case of ε, a starting value can be given and, once a torus has been obtained, it can be set to a fraction of the diameter of the invariant curve passing through the points found. The continuation process will always start from an initial known condition. The initial diameter, and ε can be estimated, for instance, from a time integration, by computing enough Poincaré sections to complete a turn on the invariant curve. If the continuation starts close to a Neimark–Sacker bifurcation, the diameter grows proportionally to $\sqrt{\lambda - \lambda_c}$, λ_c being the value of the parameter at the bifurcation. During the continuation of the tori the diameter can be estimated by calculating the distances from the initial conditions to the powers of the Poincaré map obtained while computing the maps G_i. If ε is not changed, and the diameter of the invariant curve grows, the powers of the Poincaré map required to fall inside the balls $B(x_j, \varepsilon)$ will also grow, increasing the computational cost. If the diameter decreases the interpolation might lose sense because the points might be at distances of the order of the size of the invariant curve.

Adaptability in the degree of the polynomial could also be implemented. When the full invariant curve is computed by expanding it in Fourier series, the number of harmonics can be adapted to control the error of the approximation. This was implemented in [25], where two different estimators for the error were provided. One was based on controlling the size of the tail in the Fourier expansion, and the other on the evaluation of the invariance condition on a mesh of points. In [21] a method was implemented to compute invariant curves, similar to that described here. The segment of invariant curve interpolated was that determined by a set of points on it and their images by the map. Therefore the segment could be relatively long compared with the total length of the invariant curve, requiring a large number of interpolation points. Then, instead of polynomial interpolation, splines were used. The interpolation error was estimated by increasing in one unit the number of points employed in the splines. This number was then increased or decreased depending on a threshold for the error.

The position of the μ_j can also be changed during the continuation process. Assuming they are ordered, the length $|\mu_1 - \mu_{q+1}|$ can be increased or decreased according to the variation of ε. The relative position of the μ_j is also important. They can be homothetic to a mesh of Gauss–Chebyshev or Gauss–Lobatto points to minimize the interpolation error.

The initial selection of the μ_j is essential to have convergence of the Newton's iterations together with an efficient parallelism. Ideally, the CPU time to compute each z_j (see Fig. 1b and Eq. (12)) should be essentially the same. This will be the case if the x_j are close enough, due to the continuity with respect to the initial conditions. On the other side, if ε' is the maximal distance among the x_j, and $\varepsilon' < \varepsilon$ as shown in Fig. 2a, the x'_j could be evaluated by extrapolation of the z_j, with a possible large error leading to failure in the convergence of Newton's method. If ε is too small the time integration required to obtain the z_j can be too large, and the method too inefficient (imagine Fig. 2b with the x_j closer to the curve passing through the z_j, and a very small radius of the balls).

In the numerical experiments we have kept q and the position of the μ_j fixed to avoid many differences between the two methods when comparing them, but we have implemented variation of ε.

In the description of the maps it has been shown how to compute the action of their Jacobians, which implies knowing how to calculate the product by the Jacobian of the Poincaré map. The latter is explained in detail in [11,26]. A word of caution is necessary here. Approximating the products $DG(x, \lambda)(u, \mu)$ (directional derivatives) by finite differences may lead to problems in the convergence of the linear iterative methods. The reason is that the error in the approximation has two parts. The first corresponds to the truncation error that, for centered differences with step h is $O(h^2)$, the second, due to the error in the evaluation of the map, is $O(h^{-1})$. Since computing them involves time integrations, it may be difficult to obtain good approximations unless very good time-steppers are used. This is uncommon when solving PDEs. When a linear system $Ax = b$, with $A = D_x G$, is solved by an iterative method, for instance by GMRES [23], the way of detecting a bad approximation of the products by A is to compare the estimation of the residual $\|b - Ax\|$ provided by GMRES at the end of the iterations, with the actual residual. The latter will be orders of magnitude larger than the former for bad approximations.

In [11] it was already mentioned that, for the methods presented here to make sense, the assumption of reducibility of the invariant curve (the possibility of reducing the linearized map to constant coefficients by a suitable change of variables) is essential [27,28]. Otherwise, the differential of the return map is not well defined in general, and non-typical behavior can be observed [29]. Some comments can also be found there and in [30,31] on the objects which are found depending on the Diophantine properties of the rotation number of the invariant curve, and on how the tori can be destroyed.

3. Thermal convection in binary fluid mixtures

The thermal convection of a binary mixture, filling a two-dimensional rectangular domain Ω heated from below has been used as a test problem. It was also the example in [11]. The equations of the system are the conservation of mass, momentum and energy, and the evolution of one of the concentrations (the denser has been chosen here) [32]. The units used to put them in non-dimensional form are the height of the domain h, the thermal diffusion time h^2/κ, κ being the thermal diffusivity, the temperature difference between the top and bottom sides ΔT, and $\bar{C}(\bar{C}-1)D'\Delta T/D$, $\bar{C}$ being the volume-average concentration, $D > 0$ the mass diffusion coefficient, and D' the thermal diffusion coefficient. In non-dimensional units $\Omega = [0, \Gamma] \times [0, 1]$, Γ being the width l to the height ratio, and x and y are the horizontal and vertical coordinates, respectively.

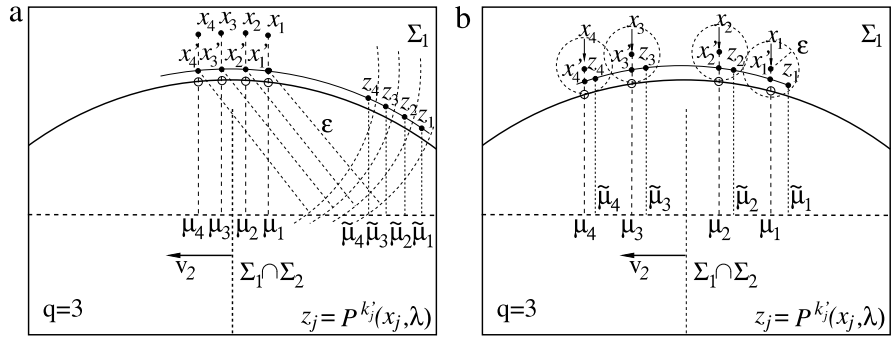

Fig. 2. Extreme cases to be avoided. (a) ε too large compared with ε', leading to extrapolation. (b) ε too small leading to very long time integrations to find the points z_j.

The basic conductive and linearly stratified state, which is a solution of the equations for any value of the parameters, is given by zero velocity $\mathbf{v}_b = 0$, and non-dimensional linear profiles for the temperature $T_b = T_b(0) - y$, and the concentration $C_b = C_b(0) - y$. The values $T_b(0)$ and $C_b(0)$ are related constants because of the boundary conditions defined below.

The Boussinesq approximation of the equations for the perturbation $(\mathbf{v}, \Theta, \Sigma)$, of the basic state $(\mathbf{v}_b, T_b, C_b)$, are

$$\partial_t \mathbf{v} + (\boldsymbol{v}\cdot\nabla)\mathbf{v} = -\nabla\pi + \sigma\nabla^2\mathbf{v} + \sigma Ra(\Theta + S\Sigma)\hat{e}_y,$$
$$\partial_t\Theta + (\boldsymbol{v}\cdot\nabla)\Theta = \nabla^2\Theta + v_y,$$
$$\partial_t\Sigma + (\boldsymbol{v}\cdot\nabla)\Sigma = L(\nabla^2\Sigma - \nabla^2\Theta) + v_y,$$
$$\nabla\cdot\mathbf{v} = 0,$$

where $\mathbf{v} = (v_x, v_y)$. The problem depends on the aspect ratio, the non-dimensional Rayleigh, Prandtl and Lewis numbers, and the separation ratio, defined as

$$\Gamma = \frac{l}{h}, \qquad Ra = \frac{\alpha g \Delta T h^3}{\kappa\nu},$$
$$\sigma = \frac{\nu}{\kappa}, \qquad L = \frac{D}{\kappa}, \qquad S = \frac{\bar{C}(1-\bar{C})\beta D'}{\alpha D}$$

respectively. In the definitions of the parameters ν means the kinematic viscosity, and α and β (taken positive) are the thermal and solutal expansion coefficients, respectively. The Prandtl and Lewis numbers represent ratios of time scales involving the viscous, thermal and solutal diffusions, but the separation ratio is also related with the temperature and concentration gradients of the flow. As usual, the Rayleigh number represents the ratio between the gravitational potential energy released when the light fluid rises (and the heavier falls), and the viscous and thermal dissipation of energy. The physical parameters must be evaluated at $\bar{C}$, $\bar{T}$, and $\bar{\rho}$.

In the continuation experiments we fix $\Gamma = 4$, $\sigma = 0.6$, $L = 0.03$ and $S = -0.1$. The last three values correspond to a mixture of two isotopes of Helium in liquid state. According with the definition of S, $D' < 0$, and initially the concentration gradient is stabilizing in opposition to the destabilizing temperature gradient. If S is below a negative critical value, as it is in the test problem, the primary bifurcation from the basic state is a Hopf bifurcation, and when convection sets in, the denser component tends to migrate towards the hotter region. The Rayleigh number, which is proportional to the difference between the bottom and top temperatures, will be the control parameter.

The boundary conditions taken are non-slip for the velocity field ($\mathbf{v} = 0$ on $\partial\Omega$), constant temperatures at the top and bottom sides, insulating lateral sides, and non-porous boundaries.

The above equations are rewritten in terms of a stream-function, ψ, i.e., $\mathbf{v} = (-\partial_y\psi, \partial_x\psi)$, and an auxiliary function $\eta = \Sigma - \Theta$. They are

$$\partial_t\nabla^2\psi + J(\psi, \nabla^2\psi) = \sigma\nabla^4\psi + \sigma Ra\,[(S+1)\partial_x\Theta + S\partial_x\eta],$$
$$\partial_t\Theta + J(\psi, \Theta) = \nabla^2\Theta + \partial_x\psi,$$
$$\partial_t\eta + J(\psi, \eta) = L\nabla^2\eta - \nabla^2\Theta,$$

with $J(f, g) = \partial_x f\,\partial_y g - \partial_y f\,\partial_x g$, and the boundary conditions become $\psi = \partial_n\psi = \partial_n\eta = 0$ on $\partial\Omega$, $\Theta = 0$ on $y = 0, 1$, and $\partial_x\Theta = 0$ on $x = 0, \Gamma$. In this way the incompressibility condition is identically fulfilled, the boundary conditions for Θ and Σ decouple, and the number of unknowns is reduced.

The group of symmetries of this system is $\mathbb{Z}_2 \times \mathbb{Z}_2$ generated by the reflections with respect to the vertical, and horizontal mid-planes, i.e., changing x by $\Gamma - x$ and the sign of ψ, or changing y by $1-y$ and the sign of all three functions, leaves the system invariant. These symmetries give rise to several pitchfork bifurcations of fixed points, periodic orbits, and also of invariant tori.

To obtain the numerical solutions, the functions ψ, Θ, and η are approximated by a pseudo-spectral method. Collocation on a mesh of $n_x \times n_y = 64 \times 16$ Gauss–Lobatto points has been used in all the calculations shown. This gives a total dimension $n = 3072$. This mesh is enough to have a good accuracy in the interval of Ra considered because the solutions are smooth. Finer resolutions were used in [11,10] to check the results. The stiff system of ODEs obtained after the spatial discretization can be written as $B\dot{u} = Lu + N(u)$, where the vector $u = (\psi_{ij}, \Theta_{ij}, \eta_{ij})$ contains the values of ψ, Θ and η at the mesh of collocation points. The operators L and N represent the linear and nonlinear terms in the equations. They are integrated by using fixed-time-step sixth-order BDF-extrapolation formulas

$$(\gamma_0 B - \Delta t L)\,u^{n+1} = \sum_{i=0}^{k-1}\left(\alpha_i B u^{n-i} + \Delta t\,\beta_i N(u^{n-i})\right)$$

where the superscripts indicate the time step. The coefficients, up to order six, are given in [8], and a comparison with other high-order time integration methods was performed in [33]. The initial points required to start the time integration are obtained by a fully implicit BDF method, using the subroutine DLSODPK of the ODEPACK package [34].

4. Invariant tori

A detailed explanation of all the results obtained for this problem can be found in [11]. We give here just a brief summary. Fig. 3 shows a part of the bifurcation diagram for the thermal convection problem, which contains the main branch of periodic orbits, and the bifurcated branches of tori. The Euclidean norm of the solution is plotted versus Ra. The points shown correspond to the intersection of the solution with the hyperplane Σ_1 if it is a periodic orbit, and with $\Sigma_1 \cap \Sigma_2$ as defined for the map G_1 if it is a torus. Solid and dashed lines mean stable and unstable solutions, respectively.

The horizontal line in Fig. 3a corresponds to the basic state which loses stability at a subcritical Hopf bifurcation, where a branch of periodic orbits emerges. It becomes stable after a saddle–node, and, very near, there is a Neimark–Sacker bifurcation giving rise to the branch of invariant tori we have computed for testing purposes. A detail of the branch of tori is given in Fig. 3b. It

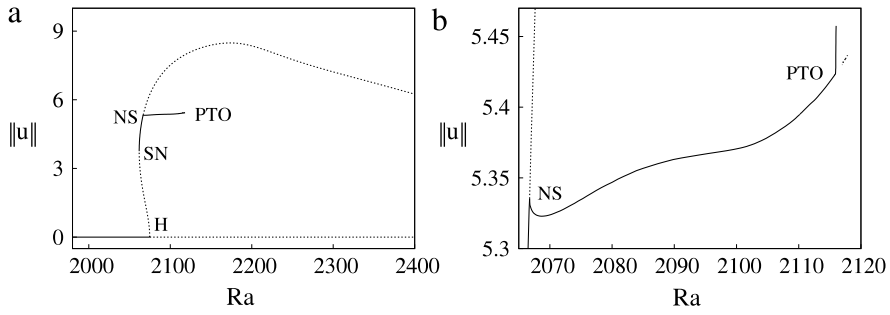

Fig. 3. (a) Bifurcation diagram. The horizontal line corresponds to the basic state. It loses stability at a subcritical Hopf bifurcation (H). The periodic orbits become stable at a saddle–node (SN) and again unstable at a Neimark–Sacker (NS) bifurcation. (b) Detail of the branch of invariant tori. It ends on a pitchfork bifurcation of invariant tori (PTO). Only one of the two stable branches is shown after the bifurcation.

starts at $Ra \approx 2066.74$, and it is stable up to a pitchfork bifurcation at $Ra \approx 2115.92$. The rotation number ρ decreases along this main branch when Ra is increased, starting near but below 1/6, and ending close to 1/8. The Appendix in [11] gives details on the method used to compute it, and on error estimates. Only one of the two stable branches of tori after the pitchfork was computed, and shown in the diagrams. This stable branch was continued up to a 1/8 resonance interval at $2116.18 < Ra < 2116.20$. The periodic orbits in this region were also computed by continuation methods. After the 1/8-phase-locking interval there are two period doubling bifurcations at $Ra \approx 2118.40$ and $Ra \approx 2118.55$, and finally, a breakdown of the tori at $Ra \approx 2118.6$. The strange attractors at $Ra = 2118.6$ and $Ra = 2118.7$ were described, and it was also possible to compute a small portion of the unstable branch after the pitchfork bifurcation at $Ra \approx 2115.92$ (see Fig. 3b). It was started with an initial condition obtained by a continuation of the stable branch with a large arclength step, which allowed to cross a gap in the unstable branch where no solution was found. This gap is probably due to a breakdown of the tori. Neither was it possible to continue the branch backward in Ra down to the pitchfork bifurcation.

5. Comparison of the two methods

To test the new algorithm and compare it with the first method, the interval $Ra \in [2100, 2115]$ on the main branch of invariant tori was selected because at some points it requires high powers of the Poincaré map when the first method (map G_1) is used. This makes this portion specially expensive to compute. We have implemented the variant of the second method described in Section 2. In this range of Ra the diameter of the invariant curve is in the interval [4.8, 5.9], the radius of the balls ε, which defines the maps, was set initially to 0.36, and to 0.075 times the estimation of the diameter during the continuation for both methods. The initial conditions for G_2 were taken inside a ball of radius $\varepsilon' = 0.3$ to ensure that the powers required to return close to them (fall in each $B(x_j, \varepsilon)$) were the same for most of the tori computed. The stations μ_i were chosen homothetic to a mesh of Gauss–Chebyshev points. The codes were written in FORTRAN, and MPI [35] was used to implement the parallel version. They were run on a cluster of Intel I7 processors at 2.67 GHz.

Before reporting the results on the comparison of the computing times required by the two methods, it is possible to advance what should be expected by making a few assumptions which are trivial or have been confirmed by the computations. Suppose that the same degree, q, of interpolation is used for both methods, with the same value of ε, then:

1. The first method requires computing increasing returning powers $k_1 < k_2 < \cdots < k_{q+1}$ of the Poincaré map on a single initial condition x. This is a sequential process.
2. The second method requires the first returning powers k'_j for a collection of initial conditions x_j. They can be computed in parallel and, provided the x_j are close enough, and ε is not too small, all the k'_j would be the same by the continuity of Poincaré's map. This is important to balance the load of the different processors involved in the computation in the parallel version of this algorithm. Therefore one can expect $k'_1 = \cdots = k'_{q+1} = k_1$, with k_1 the first returning power of the first method.
3. If the second method is implemented by using parallelism, and the previous assumption is fulfilled, the wall-clock time required to compute the map G_2 or the action of its Jacobian is reduced by a factor k_{q+1}/k_1 if it is compared with that of G_1.
4. If the number of iterations of Newton's method and the linear solver are the same for both methods, the time (wall-clock) to obtain the fixed points of the map G_2 would be reduced essentially by a factor k_{q+1}/k_1 if compared with that of G_1.
5. If the returning powers of the first method satisfy $k_2 \approx 2k_1, \ldots, k_{q+1} \approx (q+1)k_1$, then the factor in the previous statement is $k_{q+1}/k_1 \approx q+1$. If $k_2 > 2k_1, \ldots, k_{q+1} > (q+1)k_1$ then $k_{q+1}/k_1 > q+1$, and larger accelerations could be expected.
6. If $k'_1 = \cdots = k'_{q+1}$ the speed-up of the second method (wall clock time for the sequential code)/(wall clock time for the parallel code) would be about $q+1$.

Summarizing, the speed-up factor (ratio of the wall-clock times) of the parallel implementation of the second method to the sequential can be expected to be essentially $q+1$, because almost all the time is spent in the time integration, and any other task is negligible. The acceleration factor when the parallel version of the second method is compared with the first should be at least, $q+1$.

The assumption $k'_1 = \cdots = k'_{q+1} = k_1$ in the second item of the above list is fulfilled in most of the points computed during the numerical experiments. That of the fourth item is also satisfied, the number of iterations of Newton's method and the linear solver does not increase with the degree of interpolation. They seem to be quite independent of q if the points interpolated are close enough. Finally, we have seen that, in general, $k_2 > 2k_1, \ldots, k_{q+1} > (q+1)k_1$ (see item 5). Hence $k_{q+1}/k_1 > q+1$.

Table 1 summarizes the results of the tests. The first group of four rows corresponds to the first algorithm, the second group to the sequential version of the second method, and the third to its parallel version. This is described by the first and second columns. The rest of the columns contain the interpolation degree q, the wall-clock time in minutes required to do the calculations, the number of solutions computed in the interval $Ra \in [2100, 2115]$, the time required per solution computed, and two ratios of computing times. Ratio_1 is, for the first eight rows, the total time divided by the total time of the parallel version of the second algorithm with the same q. Ratio_2 is the same ratio but for the average time required to compute a solution. In the last four rows the ratios are the times divided by that of $q = 2$ in the same group, in order to compare the times required by the different interpolations with the second parallel algorithm.

The continuation algorithm incorporates a control of the arclength step-size. Therefore, the number of solutions calculated in

Table 1
Comparison of the wall-clock times for the different algorithms, implementations, and interpolation degrees. See text for details.

Meth.	Version	q	Time	N. sol.	Time/N. sol.	Ratio$_1$	Ratio$_2$
1	Serial	2	3725	19	196	4.97	4.71
1	Serial	3	3960	19	208	5.27	4.99
1	Serial	4	5019	19	264	7.34	7.34
1	Serial	5	5664	19	298	7.97	7.97
2	Serial	2	2255	18	125	3.01	3.01
2	Serial	3	2830	18	157	3.76	3.76
2	Serial	4	3563	19	188	5.21	5.21
2	Serial	5	4333	19	228	6.09	6.09
2	Parallel	2	749	18	42	1.00	1.00
2	Parallel	3	752	18	42	1.00	1.00
2	Parallel	4	684	19	36	0.91	0.87
2	Parallel	5	711	19	37	0.95	0.90

each case can be different. Since the test to increase or decrease the step-size uses the Euclidean norm, we have taken the maximal arc-length step-size as $0.05\sqrt{q+1}$, i.e., proportional to the square root of the number of points used in the interpolation. With this we managed to obtain a number of solutions of 18 or 19 in all runs. This makes the two ratios shown be different in some rows.

The comparison of the first and second groups of four lines in Table 1 shows that, even without parallelism, the second algorithm is computationally cheaper than the first. Since the number of iterations of Newton's method and the linear solver are essentially the same, the decrease of CPU time is due to the lower powers of the Poincaré map required by the second method. As stated before we found that, on average, $k_j > jk_1$ during the computation of the tori.

The comparison of the second and third groups of four lines indicates, as should be expected, that the parallelism consisting in sending each computation of the z_j of the second method to a different processor divides the wall-clock time by $q+1$. This is so because almost all the computing time is employed in the time integration of the system, and each of them is independent of the rest. In parallel computing, speed-up refers usually to how much a parallel algorithm is faster than the best available sequential algorithm. Therefore, since this is the second method we can say that the speed-up of the new algorithm is $q+1$. There is however a limitation. Since we are dealing with high-dimensional systems one must avoid the total size of the systems to be solved, $(q+1)(n-2)$, from growing too much, and then q must be limited. Then, one cannot expect a massive parallelization, unless this be possible in the time integration algorithm.

The last four rows show that the computing time is almost independent of the degree of interpolation q, at least for the low degrees considered. This indicates that the number of iterations of the solvers does not increase with q. It has also been checked from the output of the codes.

Finally, the comparison of the first and third groups of four lines shows another acceleration factor we were interested in studying. We have seen that factors above $q+1$ could be expected if $k_j > jk_1$. This was the case in our computations, and ratios 4.71, 4.99, 7.34 and 7.97 were obtained for $q+1 = 3, 4, 5$, and 6, if Ratio$_2$ is used to measure it. They depend on the distance $|k_j - jk_1|$, which depends on the rotation number of the tori, the value of ε and the position of the μ_j. This implies that they can be different in each portion of the interval of continuation. The values given here are averages in the range of *Ra* considered to do the test.

6. Low rotation number examples

The two methods presented become very expensive when the rotation number is very low, because the powers of the Poincaré map required to return to a vicinity of the initial points can be very large for small ε. In this case, instead of completing a full turn along the invariant curve to obtain the $z_j = P^{k'_j}(x_j, \lambda)$, one only needs to compute a single fixed low power k' of each x_j, $z_j = P^{k'}(x_j, \lambda)$, to reconstruct the invariant curve, because each $z_j = P^{k'}(x_j, \lambda)$ will be close to its corresponding x_j. If the rotation number is extremely low, the only requirement is that k' be large enough to separate each z_j from x_j to avoid them to be too close. Something similar could be done, in principle, with the first method. Given an initial x, the $q+1$ consecutive powers $z_1 = P(x, \lambda)$, $z_2 = P^2(x, \lambda)$, $\ldots$, $z_{q+1} = P^{(q+1)}(x, \lambda)$ could be used to define $G_1(x, \lambda)$. Again, if the rotation number is low enough the z_j will be close to the initial x. The problem is that all the z_j will be at the same side of the hyperplane Σ_2, and an extrapolation will be needed to obtain $G_1(x, \lambda)$. Unless the rotation number be very small, this can lead to the same difficulties explained for the map G_2 in the case of large ε (see Fig. 2a). The possible large errors in the extrapolation can lead to failure in the convergence of Newton's method.

To check this particular variant in the case of the second method, two examples of two-dimensional maps have been used as tests. The first is a driven logistic map defined by

$$\begin{aligned} &T : S^1 \times \mathbb{R} \to S^1 \times \mathbb{R}, \\ &T(\theta, x) = (\theta + \omega, 1 - a(\theta)x^2), \\ &a(\theta) = a_0 + \gamma \sin(2\pi\theta) \end{aligned} \tag{15}$$

with $S^1 = \{\mathbb{R} \mod 1\}$. It depends on three parameters, a_0, γ, and the rotation number ω. The second component is the logistic map but with a non-constant parameter $a(\theta)$, θ having quasiperiodic dynamics. This map is part of a family studied in [36], used there to show how double precision numerical simulations can lead to erroneous interpretations of the results. More recently, it was used in [37] to illustrate computer assisted proofs of normally hyperbolic invariant manifolds. In particular, the existence of invariant curves of T was rigorously proved.

Continuations with respect to two parameters (γ and ω) are shown in Fig. 4. The parameterizing coordinate (see Fig. 1b) is $\mu = \theta$, the number of points forming a Gauss–Lobatto mesh is $q+1 = 5$, and the common power of the map T used to define G_2 is $k' = 1$. The attractors shown in each plot were found by iterating T, starting with one of the points found by method 2. This can be done in this case because the invariant objects are stable. If they are unstable, and a larger portion or a full invariant curve is needed, it can be found by successive extrapolations of the points found by method 2 followed by a refinement by the same method. This procedure would follow the invariant curve.

The first row corresponds to the continuation with respect to $\gamma \in [0, 0.25]$, with $a_0 = 0$, $\omega = 10^{-3}\varphi$, and $\varphi = (\sqrt{5}-1)/2$ the golden mean minus one. The initial invariant curve for $\gamma = 0$ is $x = 1$. The arrows in the left plots indicate the direction of variation of the parameter. The different curves correspond to equally spaced values of the continuation parameter. The same holds for the rest of plots. The symbols in all figures are the points computed with the new algorithm. In Fig. 4a there are two groups of coordinates μ_i, one around $\theta = 0.35$ and the other around $\theta = 0.7$. In both cases the width of the interval of $\mu = \theta$ is 0.02 as can be seen in Fig. 4b, where a detail of the points around $\theta = 0.7$ is shown, and the Gauss–Lobatto mesh is visible. For this continuation there is no special reason to choose a particular position of the μ_i.

The second row of Fig. 4 corresponds to the continuation with respect to $\gamma \in [0.28, 0.67]$, decreasing it from $\gamma = 0.67$, with $a_0 = 0.85$, $\omega = 10^{-2}\varphi$, the μ_i around $\theta = 0.9$, and a width of 0.01. The initial condition can be reached by following the path $a_0 = 0.55$, $\gamma \in [0, 0.67]$, starting from the constant invariant curve $x = (\sqrt{1+4a_0}-1)/2a_0$ at $\gamma = 0$, and then $a_0 \in [0.55, 0.85]$, $\gamma = 0.67$. In this case the method is able to find invariant objects which

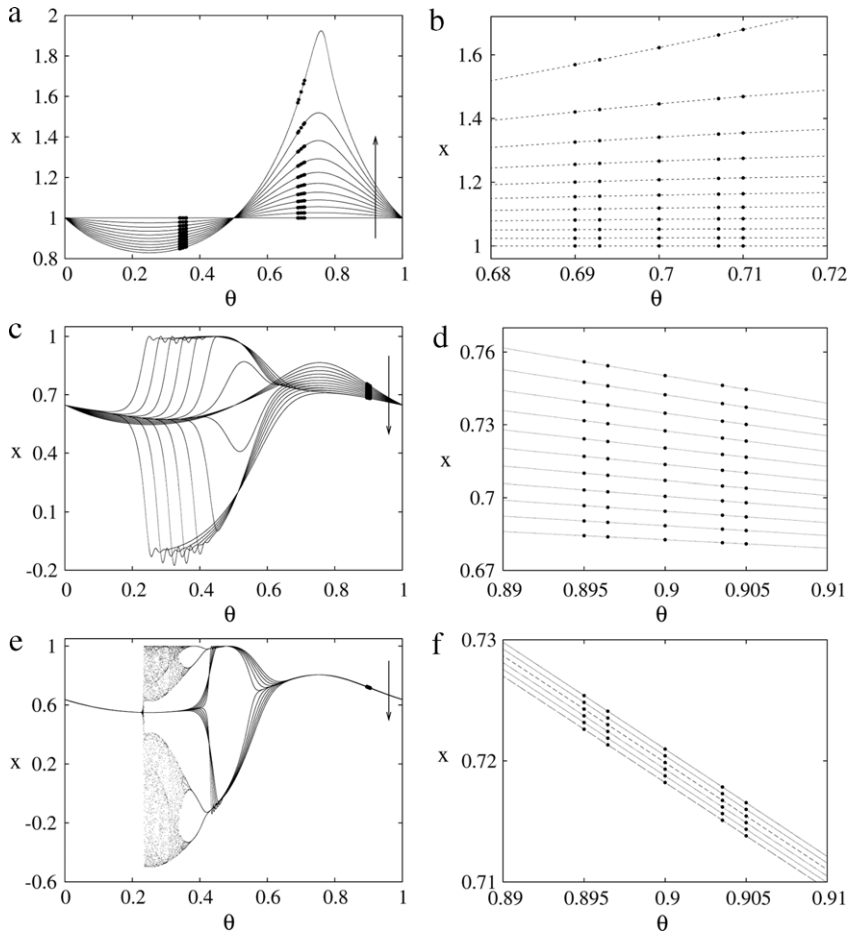

Fig. 4. Continuations for the map (15) (left plots) and details of the points computed by method 2. See text for the description of the plots.

are no longer single-valued curves. The invariant curve *doubles* because there is an interval of values of θ for which $a(\theta)$ is beyond the first period doubling of the logistic map ($a = 3/4$). The limits between which the curve is not single-valued are not just given by the solutions of the equation $a(\theta) = 3/4$ because there is a delay due to the slow variation of the parameter. An example of how to compute these limits, for other values of the parameters, can be found in [37]. The authors give a detailed explanation of the reasons why the attractor in their Figure 6 looks so different when it is computed in double precision (it seems chaotic), and in extended precision (a smooth curve). When T is iterated, starting with (θ_1, x_1), the error of x_j is multiplied at each step by a factor $|2a(\theta_j)x_j|$. After n iterations the factor becomes $\prod_{j=1}^{n} |2a(\theta_j)x_j| = \exp(\sum_{j=1}^{n} \ln |2a(\theta_j)x_j|)$. If $S_n = \sum_{j=1}^{n} \ln |2a(\theta_j)x_j|$, the Lyapunov exponent is $\lambda = \lim_{n\to\infty} S_n/n$. Suppose that (θ_1, x_1) is on a smooth invariant curve with $\lambda < 0$, but that the sequence S_n decreases oscillating to $-\infty$, such that there are indices $n_1 < n_2$ with $S_{n_1} < 0$, $S_{n_2} < 0$ but $S_{n_1} < S_{n_2}$. Then during the iterations between n_1 and n_2 the amplification factor of the error becomes $\exp(S_{n_2} - S_{n_1})$, and the number of decimal significant digits lost is $n_{dl} = (S_{n_2} - S_{n_1})/\ln 10$. This is the case for the map T. The lower α the larger n_{dl} can be, because for very small α, depending on the values of a_0 and γ, $|2a(\theta_j)x_j| > 1$ during many consecutive iterations.

For the values we have used, the doubled curves have a noticeable numerical noise if T is iterated in double precision. Therefore, each curve shown in Fig. 4c, d was computed by iterating using quad-double precision (approximately 64 decimal digits). The same holds for Fig. 4e, f. The attractors computed by using both precisions differ only inside the doubled interval, close to $\theta = 0.5$. This is why method 2 can still be used without increasing the numerical resolution. Moreover, the position of the μ_i must be out of the doubled portion. If not, the method might fail even using the second power of the map ($k' = 2$).

Finally, last row of Fig. 4 shows a continuation with respect to the rotation number in the interval $\omega \in [10^{-3}, 10^{-2}]\varphi$, decreasing from $10^{-2}\varphi$, with $\gamma = 0.6$ and $a_0 = 0.9$, and the μ_i as in the previous example. The initial conditions can be found following a path similar to that described before, or just by iterating the map T to find points with θ close to the μ_i, interpolating them if necessary. The last invariant object in Fig. 4e is no longer the union of two smooth single-valued curves. Therefore, its computation is not possible by means of the Fourier expansion method, or by any other based on the assumption that there is a global parameterization of the invariant curve. Each smooth branch of the curves in Fig. 4c, e could still be computed with the Fourier method applied to an even power of the map. In this last case the attractors look very different in double precision. It was checked that the curves shown, computed with quad-double precision, do not change when the number of decimal digits is increased up to 500.

Since the rotation number is $\omega = 6.18 \times 10^{-4}$ in some of these examples, it takes around $1/\omega = 1618$ iterations to return to a vicinity of an initial point, if a complete turn along the invariant curve is required. If $q + 1 = 5$, as in the calculations shown, this means that each evaluation of G_1 with a small ε could require the same order of iterations of T instead of just the 5 needed by G_2 with $k' = 1$. The problems with the propagation of rounding errors should also be added depending on the parameters explored. This makes the direct application of method 1 very expensive, even if this could be done without extended precision. The modified

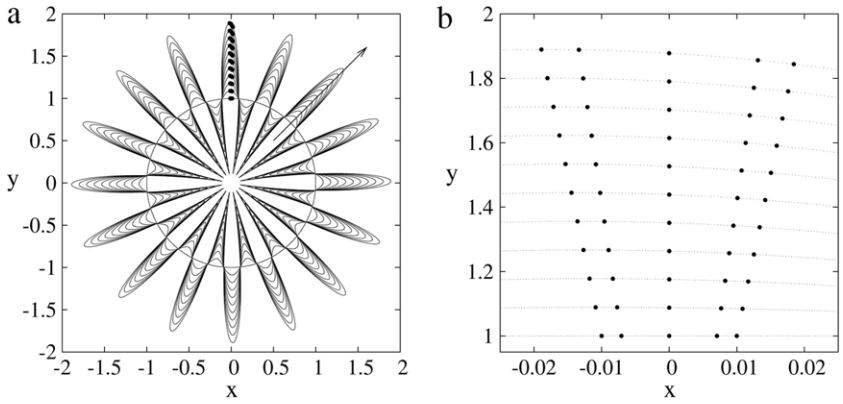

Fig. 5. Continuations for the map (16), and detail of the points computed by method 2.

version of G_1 consisting in using the first $q+1$ consecutive powers of the Poincaré map, described at the beginning of this section, could be used instead. The calculation would be sequential, and the risks of using extrapolation present.

The second map considered is defined by

$$T : S^1 \times \mathbb{R} \to S^1 \times \mathbb{R},$$
$$T(\theta, r) = (\theta + \omega,\ 1 + \lambda(r-1) + \gamma\cos(m\theta)). \tag{16}$$

Now $S^1 = \{\mathbb{R} \mod 2\pi\}$, and the map depends on λ, γ, ω, and m. There is an invariant curve, which can be found analytically. It is given by

$$r = 1 + \gamma\frac{\cos(m(\theta-\omega)) - \lambda\cos(m\theta)}{1+\lambda^2 - 2\lambda\cos(m\omega)}.$$

The continuations with respect to $\gamma \in [0, 0.09]$ can be seen in Fig. 5 for $\lambda = 0.9$, $\omega = 2\pi\varphi 10^{-4}$ and $\varphi = (\sqrt{5}+1)/2$ the golden mean. Instead of representing (θ, r), $(x, y) = r(\cos\theta, \sin\theta)$ is plotted. The points computed are on lines of constant θ, in the interval $\pi/2 \pm 0.01$. Although the dynamics of this map is not as rich as that of the previous example, it can be used to test methods for computing tori, because the number of oscillations of $r(\theta)$, m, can be controlled, and there is a closed expression for the invariant curve. The Fourier series method, for instance, would require just three non-zero coefficients.

7. Conclusions

A new method has been presented to compute invariant tori for large-scale systems obtained upon the discretization of dissipative partial differential equations, although it is also useful and may be competitive against those employed for low-dimensional systems of ordinary differential equations. Three versions, differing in the way a segment of the invariant curve is approximated, are described in detail, two of them in the final appendices. The implementation based on polynomial interpolation, has been compared with a previous method reporting accelerations of the computations by a factor above $q+1$, q being the degree of interpolation. Since the parallelization is trivial, the comparison of the serial and parallel versions of the new method show speed-ups around $q+1$. The only requirement to be optimal is that the computing time of each integration be almost the same, and this can always be accomplished if the initial conditions (i.e. the position of the μ_j) are close enough, and the radius ε is not too large. These conditions are necessary, in any case, to have a good approximation of the invariant curve.

There is an obvious limitation on the parallelism of the algorithm. Since the number of points $q+1$ must be kept small to avoid solving extremely large-scale systems, the parallelism is restricted to a low number of processors unless it is also employed in the time integration, but this is not part of the algorithm.

Since the new method requires lower powers of the Poincaré map, it is better suited for the computation of weakly unstable invariant tori than the method described in [11], and it can also be applied to problems with low rotation numbers for which the first is not adequate.

Other variants of this method are possible. Changing the way the z_j are selected, as explained before, or the kind of approximation of the invariant curve (interpolation, splines, least squares, minimax polynomials, trigonometric or Chebyshev expansions, collocation, etc.) would lead to similar methods. The only problem can be finding an easy expression for the action of the Jacobian. On the other hand, instead of a pure Newton's method, globalized versions can be employed when good initial conditions are not available.

Close to the breakdown of the tori, when the invariant curve wrinkles, the local single-valued parameterization represented in Fig. 1 might be impossible. In this case a parameterization based on approximations of the arc-length of the curve could be used. For instance, a description of chordal spline interpolation can be found in [38,39].

The main example shown here was of dimension $n = 3072$, and the time needed to do the computations with the first method was still acceptable. For larger problems it is essential to have more efficient algorithms like those described here.

Acknowledgments

We thank C. Simó for very helpful discussions during the preparation of this manuscript. This research has been supported by Spain Ministerio de Ciencia e Innovación, and Generalitat de Catalunya under projects MTM2010-16930 and 2009-SGR-67, respectively.

Appendix A. Derivatives of the map G_2

In this section we proof Eq. (13). If $G_2(X, \lambda) = X' = Z(X, \lambda)\,\tilde{V}(X, \lambda)^{-1}V$ (in short, $X' = Z\tilde{V}^{-1}V$), and since V does not depend on X and λ, we can compute the derivatives of $Y \equiv X'V^{-1} = Z\tilde{V}^{-1}$. Let $Y = (y_1, \ldots, y_{q+1})$ with $y_i \in \mathbb{R}^n$ and $Y \in \mathbb{R}^{n\times(q+1)}$, and let denote the j component of y_i by y_{ij}, and the same for X, Z, ΔX, etc. We want to compute the action $DY(X, \lambda)(\Delta X, \Delta\lambda)$, i.e.,

$$(DY(X, \lambda)(\Delta X, \Delta\lambda))_{ij} = \sum_{\substack{1\le k\le n\\ 1\le l\le q+1}} \frac{\partial y_{ij}}{\partial x_{kl}}\Delta X_{kl} + \frac{\partial y_{ij}}{\partial\lambda}\Delta\lambda \equiv \mathcal{D}y_{ij}$$

where the operator

$$\mathcal{D} = \sum_{\substack{1\le k\le n\\ 1\le l\le q+1}} \Delta X_{kl}\frac{\partial}{\partial x_{kl}} + \Delta\lambda\frac{\partial}{\partial_\lambda},$$

which acts componentwise on vectors and matrices ($(\mathcal{D}Y)_{ij} \equiv \mathcal{D}y_{ij}$), has been introduced.

Let v_{ij} and $\bar v_{ij}$ be the elements of $\tilde V$ and its inverse $\tilde V^{-1}$ respectively. The following relations hold

$$\sum_{l=1}^{q+1} v_{il}\bar v_{lj} = \delta_{ij}, \qquad \sum_{l=1}^{q+1}(\mathcal{D}v_{il}\bar v_{lj} + v_{il}\mathcal{D}\bar v_{lj}) = 0,$$

$$\mathcal{D}\bar v_{rj} = -\sum_{k=1}^{q+1}\bar v_{rk}\sum_{l=1}^{q+1}\mathcal{D}v_{kl}\bar v_{lj},$$

i.e., $\mathcal{D}\tilde V^{-1} = -\tilde V^{-1}\mathcal{D}\tilde V\tilde V^{-1}$. Since $y_{ij} = \sum_{r=1}^{q+1} z_{ir}\bar v_{rj}$,

$$\begin{aligned}(\mathcal{D}Y)_{ij} &= \sum_{r=1}^{q+1}\left(\mathcal{D}z_{ir}\bar v_{rj} + z_{ir}\mathcal{D}\bar v_{rj}\right)\\ &= \sum_{r=1}^{q+1}\left(\mathcal{D}z_{ir}\bar v_{rj} - z_{ir}\sum_{k=1}^{q+1}\bar v_{rk}\sum_{l=1}^{q+1}\mathcal{D}v_{kl}\bar v_{lj}\right)\\ &= \left(\mathcal{D}Z\,\tilde V^{-1} - Z\tilde V^{-1}\,\mathcal{D}\tilde V\,\tilde V^{-1}\right)_{ij}, \quad \text{and}\end{aligned}$$

$$(\mathcal{D}X')_{ij} = \left((\mathcal{D}Z\ - Z\tilde V^{-1}\,\mathcal{D}\tilde V)\tilde V^{-1}V\right)_{ij}.$$

Finally, since the elements of $\tilde V$, v_{ij} depend on the $\tilde\mu_k$, which in turn depend on the x_l and λ (in our case $\tilde\mu_k$ only depends on x_k), we have

$$\mathcal{D}v_{ij} = \sum_{l=1}^{q+1}\frac{\partial v_{ij}}{\partial\tilde\mu_l}\mathcal{D}\tilde\mu_l.$$

The Vandermonde matrix $\tilde V$ has elements $v_{ij} = \tilde\mu_j^{i-1}$, and $\tilde\mu_l = v_2^\top(z_l - x^{\Sigma_2})$, hence

$$\frac{\partial v_{ij}}{\partial\tilde\mu_l} = (i-1)\tilde\mu_j^{i-2}\delta_{jl},$$

$$\mathcal{D}\tilde\mu_l = v_2^\top\mathcal{D}z_l = v_2^\top DP^{k'_l}(x_l,\lambda)(\Delta x_l,\Delta\lambda) \equiv \eta_l,$$

and therefore $\mathcal{D}v_{ij} = (i-1)\tilde\mu_j^{i-2}\eta_j$. Compare now with Eq. (14).

Appendix B. Conditioning of the Jacobian of G_2

An idea of the distribution of the eigenvalues of $DG_2(X,\lambda)$ is necessary to see that the results in [24] can be used to establish the fast convergence of GMRES solving systems with matrix $I - DG_2(X,\lambda)$, and the independence of q of the number of iterations. For this purpose we assume that the points $x_1,\dots,x_{q+1}$, which approximate an arc of a torus, are inside a ball $B(x_0, m\varepsilon)$ centered at a point x_0 on the torus and close, for instance, to the barycenter of the x_j, and of radius m times that of the balls defining the map G_2, for some m. Suppose also, for simplicity, that all the powers k'_j are the same, and equal to a common value k.

Each $z_j = P^k(x_j,\lambda)$ is in $B(x_j,\varepsilon)$, and $\tilde\mu_j$ and μ_j are, respectively, the projections of z_j and x_j onto the line $x = x^{\Sigma_2} + \mu v_2$ (see Fig. 1), then $\tilde\mu_j = \mu_j + \delta_j$ with $|\delta_j| < \varepsilon$. This implies that $V = \tilde V + \mathcal{O}(\varepsilon)$, and $\tilde V^{-1}V = I + \mathcal{O}(\varepsilon)$. Then the map G_2 can be written as $G_2 = Z(X,\lambda)\tilde V(X,\lambda)^{-1}V = Z(X,\lambda) + \mathcal{O}(\varepsilon)$, and

$$DG_2(X,\lambda) = DZ(X,\lambda) + \mathcal{O}(\varepsilon).$$

The time integrations to find $P^k(x_j,\lambda)$, $j = 1,\dots,q+1$, are independent, and then the Jacobian DZ has a block-diagonal structure with blocks $DP^k(x_j,\lambda)$. Since the x_j are within a distance $m\varepsilon$ we can also write

$$DP^k(x_j,\lambda) = DP^k(x_0,\lambda) + \mathcal{O}(\varepsilon).$$

Hence DG_2 acts as $q+1$ copies of $DP^k(x_0,\lambda)$ plus a perturbation of order ε. The map P^k is computed by evolving a system of parabolic PDEs, and then the Jacobian $DP^k(x_0,\lambda)$ will have the spectrum tightly clustered around zero, with p eigenvalues (p much less than the total dimension n) at a distance from the origin greater than a certain $\beta < 1$. If the torus being computed is unstable some of these latter will be out of the unit circle. By introducing a small perturbation (for small enough ε) each eigenvalue of multiplicity $q+1$ splits into $q+1$ simple eigenvalues inside a small disk [40]. Therefore, the matrix $I - DG_2$ has a cluster centered at $+1$ containing most of its spectrum and p more clusters, each one containing $q+1$ eigenvalues, and which can be considered of radius less than β for small enough ε. Under these conditions, the results in [24] apply, and after kp iterations of GMRES a reduction in norm of the residual

$$\|r_{kp}\| \le C(\sigma^{p-1}\beta)^k\|r_0\|$$

can be expected, r_0 being the initial residual, C a constant independent of k, and σ the relative maximal distance between clusters. It must be stated that this bound is not necessary sharp and faster convergences are possible.

The difference, when the degree of the interpolation polynomial q is changed, is just the number of eigenvalues inside each disk, not the number of clusters or their relative positions, assuming the new points x_j are still in $B(x_0, m\varepsilon)$. Therefore, the convergence of GMRES is not affected by changing q. This situation is completely different to what happens when periodic orbits are computed by multiple shooting. The integrations for the subintervals can be computed in parallel, but the number of iteration of GMRES grows proportionally to the number of subintervals m and no speed-up is achieved when using parallelism. The reason is that the eigenvalues of the multiple-shooting matrix spread in circles because they are the m-th roots of that corresponding to the single-shooting method. Hence a preconditioner is required in this case to keep the iterations independent of m [10].

Appendix C. Spline interpolation

The definition of $G_2(X,\lambda)$ is given here for cubic spline interpolation. With the notation of Section 2 and following [41], assume $\tilde\mu_1 < \cdots < \tilde\mu_{q+1}$ (if not they must be reordered), let $I_i = [\tilde\mu_{i-1},\tilde\mu_i]$, $h_i = \tilde\mu_i - \tilde\mu_{i-1}$, and $\theta = (\mu - \tilde\mu_{i-1})/h_i \in [0,1]$ if $\mu \in I_i$. The cubic spline which passes through the points $(\tilde\mu_i, z_i)$, $i = 1,\dots,q+1$, with $z_i \in \Sigma_1 \cap \Sigma_2^i \subset \mathbb{R}^n$, is given, in I_i, by

$$\begin{aligned}S(\mu) = &-\frac{h_i^2}{6}(\theta^3 - 3\theta^2 + 2\theta)M_{i-1} + \frac{h_i^2}{6}(\theta^3 - \theta)M_i\\ &+ (1-\theta)z_{i-1} + \theta z_i\end{aligned}$$

with $M_i = S''(\tilde\mu_i)$. The continuity of the first derivatives at the inner nodal points gives the equations

$$\begin{aligned}&h_iM_{i-1} + 2(h_i + h_{i+1})M_i + h_{i+1}M_{i+1}\\ &\quad = 6\left(\frac{z_{i+1} - z_i}{h_{i+1}} - \frac{z_i - z_{i-1}}{h_{i-1}}\right), \quad i = 2,\dots,q.\end{aligned}$$

Two additional conditions must be added to completely determine the spline. The available options, which can easily be expressed in terms of the M_i, are natural spline ($M_1 = M_{q+1} = 0$), parabolic runout spline ($M_1 = M_2$, $M_q = M_{q+1}$), cubic runout spline ($M_1 = 2M_2 - M_3$, $M_{q+1} = 2M_q - M_{q-1}$), or M_1 and M_{q+1} can be approximated by finite differences from the values z_i (of fourth order if the

accuracy of the spline has to be preserved). In any case the relation between the z_i and the M_i can be written as

$$(M_1, \ldots, M_{q+1})W_1 = (z_1, \ldots, z_{q+1})W_2,$$

where the matrices $W_i \in \mathbb{R}^{(q+1)\times(q+1)}$, $i = 1, 2$, are tridiagonal in most cases. If the spline is evaluated at the nodes μ_j and $x'_j = S(\mu_j)$ we have

$$x'_j = -\frac{h_{i_j}^2}{6}(\theta_j^3 - 3\theta_j^2 + 2\theta_j)M_{i_j-1} + \frac{h_{i_j}^2}{6}(\theta_j^3 - \theta_j)M_{i_j} + (1-\theta_j)z_{i_j-1} + \theta_j z_{i_j}$$

with $\theta_j = (\mu_j - \tilde{\mu}_{i_j-1})/h_{i_j}$, if $\mu_j \in I_{i_j}$, i.e., there are matrices $W_i \in \mathbb{R}^{(q+1)\times(q+1)}$, $i = 3, 4$ such that

$$(x'_1, \ldots, x'_{q+1}) = (M_1, \ldots, M_{q+1})W_3 + (z_1, \ldots, z_{q+1})W_4.$$

Hence

$$(x'_1, \ldots, x'_{q+1}) = (z_1, \ldots, z_{q+1})(W_2W_1^{-1}W_3 + W_4),$$

where the x'_i and z_i are column vectors, and

$$X' = G_2(X, \lambda) = Z(W_2W_1^{-1}W_3 + W_4),$$

which is the alternative to Eq. (12).

The action of DG_2 is similar to that corresponding to polynomial interpolation, i.e.,

$$\begin{aligned} DG_2(X, \lambda)(\Delta X, \Delta\lambda) = {} & DZ(W_2W_1^{-1}W_3 + W_4) \\ & + Z\left(DW_2W_1^{-1}W_3 + W_2W_1^{-1}\right. \\ & \left. \times(-DW_1W_1^{-1}W_3 + DW_3) + DW_4\right), \end{aligned}$$

where DZ and DW_i represent

$$DZ \equiv DZ(X, \lambda)(\Delta X, \Delta\lambda),$$

$$DW_i \equiv DW_i(X, \lambda)(\Delta X, \Delta\lambda) = \sum_{j=1}^{q+1} \eta_j \frac{\partial}{\partial\tilde{\mu}_j} W_i,$$

with $\eta_j = v_2^\top DP^{k'_j}(x_j, \lambda)(\Delta x_j, \Delta\lambda)$.

Appendix D. Polynomial fitting

The definition of $G_2(X, \lambda)$ is given here for least squares polynomial fitting. As in the case of polynomial interpolation, given a value of ε, if $X = (x_1, \ldots, x_{q+1}) \in \mathcal{U}_2 \subset \mathbb{R}^{n\times(q+1)}$ such that the projection of x_j onto the line $x = x^{\Sigma_2} + \mu v_2$ is μ_j, i.e.,

$$v_2^\top(x_j - x^{\Sigma_2}) = \mu_j, \quad j = 1, \ldots, q+1, \tag{17}$$

a larger collection of points $(\hat{\mu}_j, \hat{x}_j)$, $j = 1, \ldots, q'+1$, with $q' > q$ can be obtained by polynomial interpolation at a set of new nodes $\hat{\mu}_j$. If $\hat{X} = (\hat{x}_1, \ldots, \hat{x}_{q'+1}) \in \mathbb{R}^{n\times(q'+1)}$, and $V \in \mathbb{R}^{(q+1)\times(q+1)}$ and $\hat{V} \in \mathbb{R}^{(q+1)\times(q'+1)}$ are the Vandermonde matrices

$$V = \begin{pmatrix} 1 & \cdots & 1 \\ \mu_1 & \cdots & \mu_{q+1} \\ \cdots & \cdots & \cdots \\ \mu_1^q & \cdots & \mu_{q+1}^q \end{pmatrix}, \quad \text{and}$$

$$\hat{V} = \begin{pmatrix} 1 & \cdots & 1 \\ \hat{\mu}_1 & \cdots & \hat{\mu}_{q'+1} \\ \cdots & \cdots & \cdots \\ \hat{\mu}_1^q & \cdots & \hat{\mu}_{q'+1}^q \end{pmatrix}, \tag{18}$$

then $\hat{X} = XV^{-1}\hat{V}$.

Let $z_j = P^{k'_j}(\hat{x}_j, \lambda)$ for $j = 1, \ldots, q'+1$, be the first power of the Poincaré map on $\hat{x}_j$ falling inside the ball $B(\hat{x}_j, \varepsilon)$, $Z = (z_1, \ldots, z_{q'+1})$, and $\tilde{\mu}_j = v_2^\top(z_j - x^{\Sigma_2})$, $j = 1, \ldots, q'+1$ (as for the map G_2 in Fig. 1b).

The overdetermined system of equations to fit a polynomial of degree q

$$Q(\mu) = \sum_{i=0}^{q} \alpha_i \mu^i,$$

with $\alpha_i \in \mathbb{R}^n$, to the points z_j, is $Z = A\tilde{V}$, where $A = (\alpha_0, \ldots, \alpha_q) \in \mathbb{R}^{n\times(q+1)}$, and with

$$\tilde{V} = \begin{pmatrix} 1 & \cdots & 1 \\ \tilde{\mu}_1 & \cdots & \tilde{\mu}_{q'+1} \\ \cdots & \cdots & \cdots \\ \tilde{\mu}_1^q & \cdots & \tilde{\mu}_{q'+1}^q \end{pmatrix} \tag{19}$$

the Vandermonde matrix associated with the $\tilde{\mu}_j$, $\tilde{V} \in \mathbb{R}^{(q+1)\times(q'+1)}$. The normal equations to find the least squares polynomial of degree q are

$$A\tilde{V}\tilde{V}^\top = Z\tilde{V}^\top.$$

Therefore the definition of the map is

$$G_2(X, \lambda) = X' = AV = Z\tilde{V}^\top(\tilde{V}\tilde{V}^\top)^{-1}V, \tag{20}$$

where the dependence of Z and $\tilde{V}$ on X and λ has no been explicited. This is the second alternative to Eq. (12).

The action of the Jacobian of this G_2 can be computed as before. If $W = \tilde{V}\tilde{V}^\top$ then $G_2(X, \lambda) = Z\tilde{V}^\top W^{-1}V$, and

$$\begin{aligned} & DG_2(X, \lambda)(\Delta X, \Delta\lambda) \\ & = \left[DZ\tilde{V}^\top + Z\left(D\tilde{V}^\top - \tilde{V}^\top W^{-1}(D\tilde{V}\tilde{V}^\top + D\tilde{V}^\top\tilde{V})\right)\right]W^{-1}V, \end{aligned}$$

where the identity $DW^{-1} = -W^{-1}DWW^{-1}$ has been used, DZ and $D\tilde{V}$ represent, respectively,

$$\begin{aligned} DZ \equiv DZ(X, \lambda)(\Delta X, \Delta\lambda) = {} & (DP^{k'_1}(\hat{x}_1, \lambda)(\Delta\hat{x}_1, \Delta\lambda), \ldots, \\ & DP^{k'_{q'+1}}(\hat{x}_{q'+1}, \lambda)(\Delta\hat{x}_{q'+1}, \Delta\lambda)), \end{aligned}$$

with $\Delta\hat{X} = (\Delta\hat{x}_1, \ldots, \Delta\hat{x}_{q'+1}) = \Delta XV^{-1}\hat{V}$, and

$$\begin{aligned} & D\tilde{V} \equiv D\tilde{V}(X, \lambda)(\Delta X, \Delta\lambda) \\ & = \begin{pmatrix} 0 & \cdots & 0 \\ 1 & \cdots & 1 \\ 2\tilde{\mu}_1 & \cdots & 2\tilde{\mu}_{q'+1} \\ \cdots & \cdots & \cdots \\ q\tilde{\mu}_1^{q-1} & \cdots & q\tilde{\mu}_{q'+1}^{q-1} \end{pmatrix} \begin{pmatrix} \eta_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \eta_{q'+1} \end{pmatrix}, \end{aligned}$$

with $\eta_j = v_2^\top DP^{k'_j}(\hat{x}_j, \lambda)(\Delta\hat{x}_j, \Delta\lambda)$ for $j = 1, \ldots, q'+1$.

References

[1] G. Kawahara, M. Uhlmann, L. van Veen, The significance of simple invariant solutions in turbulent flows, Annu. Rev. Fluid Mech. 44 (1) (2012) 203–225.

[2] G.F. Carey, K. Wang, W. Joubert, Performance of iterative methods for Newtonian and generalized Newtonian flows, Int. J. Numer. Methods Fluids 9 (1989) 127–150.

[3] G.M. Shroff, H.B. Keller, Stabilization of unstable procedures: the recursive projection method, SIAM J. Numer. Anal. 30 (4) (1993) 1099–1120.

[4] P.N. Brown, Y. Saad, Hybrid Krylov methods for nonlinear systems of equations, SIAM J. Sci. Stat. Comput. 11 (3) (1990) 450–481.

[5] M.J. Molemaker, H.A. Dijkstra, Multiple equilibria and stability of the North-Atlantic wind-driven ocean circulation, in: E. Doedel, L.S. Tuckerman (Eds.), Numerical Methods for Bifurcation Problems and Large-scale Dynamical Systems, in: The IMA Volumes in Mathematics and its Applications, vol. 119, Springer, 2000, pp. 35–65.

[6] L.S. Tuckerman, D. Barkley, Bifurcation analysis for timesteppers, in: E. Doedel, L.S. Tuckerman (Eds.), Numerical Methods for Bifurcation Problems and Large-Scale Dynamical Systems, in: IMA Volumes in Mathematics and its Applications, vol. 119, Springer–Verlag, 2000, pp. 453–466.

[7] K. Lust, D. Roose, A. Spence, A. Champneys, An adaptive Newton–Picard algorithm with subspace iteration for computing periodic solutions, SIAM J. Sci. Comput. 19 (4) (1998) 1188–1209.
[8] J. Sánchez, M. Net, B. García-Archilla, C. Simó, Newton–Krylov continuation of periodic orbits for Navier–Stokes flows, J. Comput. Phys. 201 (1) (2004) 13–33.
[9] M. Net, J. Sánchez, Symmetric periodic orbits and global dynamics of quasi-periodic flows in an $O(2)$ equivariant system: two-dimensional thermal convection, Internat. J. Bifur. Chaos 15 (12) (2005) 3953–3972.
[10] J. Sánchez, M. Net, On the multiple shooting continuation of periodic orbits by Newton–Krylov methods, Internat. J. Bifur. Chaos 20 (1) (2010) 1–19.
[11] J. Sánchez, M. Net, C. Simó, Computation of invariant tori by Newton–Krylov methods in large-scale dissipative systems, Physica D 239 (2010) 123–133.
[12] K. Meerbergen, D. Roose, Matrix transformations for computing rightmost eigenvalues of large sparse non-symmetric eigenvalue problems, IMA J. Numer. Anal. 16 (3) (1996) 297–346.
[13] D. Barkley, R.D. Henderson, Floquet stability analysis of the periodic wake of a circular cylinder, J. Fluid Mech. 322 (1996) 215–241.
[14] B. García-Archilla, J. Sánchez, C. Simó, Krylov methods and test functions for detecting bifurcations in one parameter-dependent partial differential equations, BIT 46 (4) (2006) 731–757.
[15] M. Net, F. Garcia, J. Sánchez, On the onset of low-Prandtl-number convection in rotating spherical shells: non-slip boundary conditions, J. Fluid Mech. 601 (2008) 317–337.
[16] G.M. Lewis, W. Nagata, Double Hopf bifurcation in the differentially heated rotating annulus, SIAM J. Appl. Math. 63 (3) (2003) 1055–1229.
[17] J. Sánchez, M. Net, J. Vega, Amplitude equations close to a triple-$(+1)$ bifurcation point of D_4-symmetric periodic orbits in $O(2)$-equivariant systems, Discrete Contin. Dyn. Syst. B 6 (6) (2006) 1357–1380.
[18] L. van Veen, G. Kawahara, M. Atsushi, On matrix-free computation of 2d unstable manifolds, SIAM J. Sci. Comput. 33 (1) (2011) 25–44.
[19] C. Kaas-Petersen, Computation, continuation, and bifurcation of torus solutions for dissipative maps and ordinary differential equations, Physica D 25 (1–3) (1987) 288–306.
[20] C. Simó, Effective computations in Hamiltonian dynamics, in: Cent ans après les Méthodes Nouvelles de H. Poincaré, Société Mathématique de France, 1996, pp. 1–23.
[21] C.-J. Yu, J.-E. Lee, An algorithm to approach invariant curves, Physica D 149 (2001) 30–42.
[22] À. Jorba, Numerical computation of the normal behaviour of invariant curves on n-dimensional maps, Nonlinearity 14 (2001) 943–976.
[23] Y. Saad, M.H. Schultz, GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems, SIAM J. Sci. Stat. Comput. 7 (1986) 865–869.
[24] S.L. Campbell, I.C.F. Ipsen, C.T. Kelley, C.D. Meyer, GMRES and the Minimal Polynomial, BIT 36 (4) (1996) 664–675.
[25] À. Jorba, E. Olmedo, On the computation of reducible invariant tori on a parallel computer, SIAM J. Appl. Dyn. Syst. 8 (4) (2009) 1382–1404.
[26] C. Simó, Analytical and numerical computation of invariant manifolds, in: C. Benest, C. Froeschlé (Eds.), Modern Methods in Celestial Mechanics, Editions Frontières, 1990, pp. 285–330. Also at http://www.maia.ub.es/dsg/2004/.
[27] À. Jorba, C. Simó, On the reducibility of linear differential equations with quasiperiodic coefficients, J. Differential Equations 98 (1992) 111–124.
[28] À. Jorba, R. Ramírez-Ros, J. Villanueva, Effective reducibility of quasi-periodic linear equations close to constant coefficients, SIAM J. Math. Anal. 28 (1997) 178–188.
[29] L. van Veen, The quasi-periodic doubling cascade in the transition to weak turbulence, Physica D 210 (2005) 249–261.
[30] D.G. Aronson, M.A. Chory, G.R. Hall, R.P. McGehee, Bifurcations from an invariant circle for two-parameter families of maps of the plane: a computer-assisted study, Commun. Math. Phys. 83 (1982) 303–354.
[31] H. Broer, C. Simó, J. Tatjer, Towards global models near homoclinic tangencies of dissipative diffeomorphisms, Nonlinearity 11 (1998) 667–770.
[32] S.R. de Groot, P. Mazur, Non-Equilibrium Thermodynamics, Dover Publications, Amsterdam, 1962.
[33] F. Garcia, M. Net, B. García-Archilla, J. Sánchez, A comparison of high-order time integrators for the Boussinesq Navier–Stokes equations in rotating spherical shells, J. Comput. Phys. 229 (2010) 7997–8010.
[34] A.C. Hindmarsh, ODEPACK, a systematized collection of ODE solvers, in: R.S. Stepleman, et al. (Eds.), Scientific Computing, North-Holland, Amsterdam, 1983, pp. 55–64.
[35] Message Passing Forum, Mpi: A message-passing interface standard, Tech. rep., University of Tennessee, Knoxville, TN, USA (1994).
[36] H. Broer, C. Simó, R. Vitolo, Chaos and quasi-periodicity in diffeomorphisms of the solid torus, Discrete Contin. Dyn. Syst. B 14 (2010) 871–905.
[37] M.J. Capiński, C. Simó, Computer assisted proof for normally hyperbolic invariant manifolds, Nonlinearity 25 (2012) 1997–2026.
[38] M.S. Floater, T. Surazhsky, Parameterization for curve interpolation, in: K. Jetter, M. Buhmann, W. Haussmann, R. Schaback, J. Stoeckler (Eds.), Topics in Multivariate Approximation and Interpolation, Elsevier, 2006, pp. 39–54.
[39] M.S. Floater, Chordal cubic spline interpolation is fourth-order accurate, IMA J. Numer. Anal. 26 (1) (2006) 25–33.
[40] T. Kato, Perturbation theory for linear operators, in: Classics in Mathematics, second ed., Springer-Verlag, 1995.
[41] A. Ralston, P. Rabinowitz, A First Course in Numerical Analysis, second edition, McGraw-Hill, 1978.