Commun Nonlinear Sci Numer Simulat 18 (2013) 1734–1745

Contents lists available at SciVerse ScienceDirect

Commun Nonlinear Sci Numer Simulat

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

High-order adaptive method for computing two-dimensional invariant manifolds of three-dimensional maps

Jacek K. Wróbel *, Roy H. Goodman *

Department of Mathematical Sciences, New Jersey Institute of Technology, University Heights, Newark, NJ 07102, USA

ARTICLE INFO

Article history:
Received 31 August 2011
Received in revised form 11 October 2012
Accepted 26 October 2012
Available online 28 November 2012

Keywords:
Invariant manifold
Computer-aided geometric design
Numerical algorithm

ABSTRACT

An efficient and accurate numerical method is presented for computing invariant manifolds of maps which arise in the study of dynamical systems. A quasi-interpolation method due to Hering-Bertram et al. is used to decrease the number of points needed to compute a portion of the manifold. Bézier triangular patches are used in this construction, together with adaptivity conditions based on properties of these patches. Several numerical tests are performed, which show the method to compare favorably with previous approaches.

1. Introduction

We consider the problem of computing two-dimensional invariant manifolds of iterated maps $f : \mathbb{R}^3 \to \mathbb{R}^3$ following our recent work [1] that introduced accurate and efficient adaptive methods for computing such manifolds in $\mathbb{R}^2$. These methods, based on high-order local splines, developed for the field of computer-aided geometric design (CAGD), greatly exceed the accuracy and efficiency of existing methods. The methods of [1] do not generalize straightforwardly to higher dimensions, for reasons we will describe in detail. Thus we look to CAGD for methods that will be useful in this case.

The one-dimensional invariant manifold in [1] is approximated using a Catmull–Rom spline, an interpolation scheme based on composite Bézier curves. This improves on previous methods that use piecewise linear interpolation, e.g. [2–4] which tend to either place too many points near, or else fail even to resolve, segments of the manifolds with large curvature. Tests in [1] show our methods are better able to follow the curvature of these manifolds, maintaining higher accuracy with fewer calls to the map.

In [1], the one-dimensional manifold is computed using an interpolation method over an adaptively-generated non-uniform grid. An interpolating piecewise-defined surface is defined by function values at discrete points, but it must be smooth and even continuous across one-dimensional edges, which is difficult to achieve. This generally requires the use of higher-degree polynomials and the solution of a very large linear system over the whole surface to calculate the coefficients. In spite of the high-degree polynomials involved, such interpolation methods usually suffer from artifacts, e.g. they appear puckered or wrinkled when compared to the true surface. In addition, the necessity of solving a global linear system means that adaptively adding points on one portion of a surface changes the coefficients everywhere on the surface. Quasi-interpolation methods apply weaker conditions and can generate surfaces without such artifacts. They permit localized refinement without requiring the solution of a global linear system. We apply an adaptive quasi-interpolating scheme using triangular Bézier patches, developed by Hering-Bertram et al. [5] to approximate the manifolds.

* Corresponding authors. Present address: Department of Mathematics and Center for Computational Science, Tulane University, 6823 St. Charles Avenue, New Orleans, LA 70118, USA (J.K. Wróbel).

E-mail addresses: wrobel@njit.edu (J.K. Wróbel), goodman@njit.edu (R.H. Goodman).

1007-5704/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.cnsns.2012.10.017

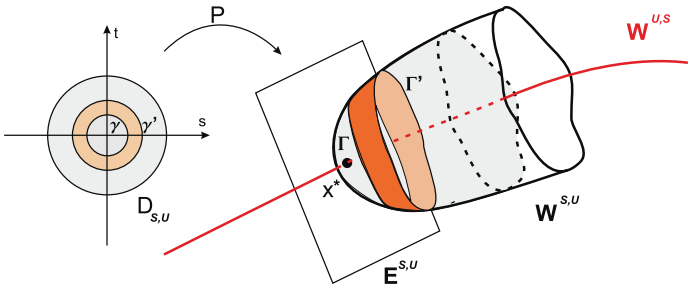

Fig. 1. (Schematic) Right: one- and two-dimensional invariant manifolds of a hyperbolic fixed point $\mathbf{x}^*$ and the initial primary annulus $\mathbf{U}_0$ between Γ and $\Gamma\prime$. Left: pre-images in parameter space.

While many algorithms exist for approximating unstable manifolds for differential equations, summarized in [6], fewer published methods exist for iterated maps, e.g. [4,7–9]. We identify several situations in which our method outperforms existing methods and, indeed, where existing methods fail to compute the manifold altogether. We perform quantitative numerical tests of the algorithm and compare it with the method [4,7]. With that said, important cases of maps exist that present difficulty to all such methods, including ours. One purpose of this paper is to explicitly lay out the technical difficulties that an effective method must overcome.

This paper is organized as follows. Section 2 introduces the basic dynamical systems objects for which we develop approximation algorithms. In Section 3, we describe existing methods: the parameterization method and the geometric level set method (GLSM) due to Krauskopf and Osinga, and highlight their shortcomings in order to display the need for a new method. In particular, we construct an example for which GLSM is able to compute only a small finite portion of the manifold. Section 4 introduces the CAGD tools used to construct the algorithm, including triangular Bézier patches and the concept of quasi-interpolation. It briefly describes two quasi-interpolation methods constructed using these ideas. 5 describes the implementation of these tools in the context of computing two-dimensional invariant manifolds. Section 6 contains numerical tests demonstrating the performance of this method. Section 7 contains a discussion summarizing the advantages of the algorithm and some remaining challenges.

2. Dynamical systems background

Iterated maps and invariant manifolds. We consider iterated maps of the form $\mathbf{x}_{j+1} = f(\mathbf{x}_j)$. We assume the diffeomorphism $f : \mathbb{R}^3 \to \mathbb{R}^3$ is as smooth as needed and has a hyperbolic fixed point $\mathbf{x}^*$ with a two-dimensional unstable manifold and a one-dimensional stable manifold, i.e., $f(\mathbf{x}^*) = \mathbf{x}^*$ such that the linearized matrix $\mathbf{F} = Df(\mathbf{x}^*)$ has eigenvalues $\lambda_{u_1}, \lambda_{u_2}$, and λ_s satisfying $0 < |\lambda_s| < 1 < |\lambda_{u_1}|, |\lambda_{u_2}|$; see Fig. 1.[1]

The unstable manifold,

$$W^{\mathrm{u}}(\mathbf{x}^*) = \{\mathbf{x} \in \mathbb{R}^n : f^k(\mathbf{x}) \to \mathbf{x}^* \text{ as } k \to -\infty\}$$

is defined as the set of points which converge to $\mathbf{x}^*$ under iterates of the map f^{-1}. The above hyperbolicity assumptions ensure exponential convergence. The manifold $W^{\mathrm{u}}(x^*)$ is tangent to $E^{\mathrm{u}}(\mathbf{x}^*)$, the unstable eigenspace of the linearized system at $\mathbf{x}^*$.

Proper loops and fundamental domains. Most numerical methods compute invariant manifolds by repeatedly applying the map f to an existing portion of the manifold—a fundamental domain.

First, define a *proper loop* [10] to be a smooth, simple, closed curve $\Gamma \subset W^{\mathrm{u}}$ such that Γ bounds a surface $W^{\mathrm{u}}_\Gamma \subset \mathbf{int}(W^{\mathrm{u}})$ which is a trapping region: $f^{-1}(\mathbf{cl}(W^{\mathrm{u}}_\Gamma)) \subset \mathbf{int}(W^{\mathrm{u}}_\Gamma)$, where $\mathbf{int}(W^{\mathrm{u}}_\Gamma)$ and $\mathbf{cl}(W^{\mathrm{u}}_\Gamma)$ refer, respectively, to the interior and the closure of the set W^{u}_Γ inside W^{u}. Since f is invertible, it maps proper loops to proper loops. Given two proper loops $\Gamma, \Gamma' \subset W^{\mathrm{u}}$ such that $\Gamma \cap \Gamma' = \varnothing$ and $\Gamma \subset \mathbf{int}(\Gamma')$, let $W^{\mathrm{u}}[\Gamma, \Gamma']$ denote the closed annular region of W^{u} with boundary $\Gamma \cup \Gamma'$; similarly, $W^{\mathrm{u}}(\Gamma, \Gamma')$ denotes an open annular region; see Fig. 1. The half-open annular region $W^{\mathrm{u}}[\Gamma, f(\Gamma))$ defines a *fundamental domain*.

For any given proper loop $\Gamma_0 \in W^{\mathrm{u}}$, the set of fundamental domains $\mathbf{U}_k = W^{\mathrm{u}}[\Gamma_k, \Gamma_{k+1})$ where $\{\Gamma_k = f^k(\Gamma_0) : k \in \mathbb{Z}\}$, forms a partition of W^{u}:

$$W^{\mathrm{u}} = \bigcup_{k=-\infty}^{\infty} \mathbf{U}_k; \quad \mathbf{U}_j \cap \mathbf{U}_k = \begin{cases} \mathbf{U}_j; & j = k; \\ \varnothing; & j \neq k. \end{cases}$$

We refer to the fundamental domain $\mathbf{U}_k$ as the kth primary annulus and note that $\mathbf{U}_{k+1} = f(\mathbf{U}_k)$. Denoting the numerical approximation to $\mathbf{U}_k$ by U_k, the numerical method generates a sequence of approximations

$$U_{k+1} \approx f(U_k). \tag{1}$$

[1] The case $0 < |\lambda_{s_1}|, |\lambda_{s_2}| < 1 < |\lambda_u|$, with 2D stable and 1D unstable manifolds is handled analogously using f^{-1} in place of f.

Fig. 2. The first, fifth, tenth and fifteenth primary annuli from a two-dimensional manifold of a hyperbolic fixed point for map (10).

Define the "initial portion" of the manifold as

$$W^{\mathrm{u}}_{\mathrm{init}} = \bigcup_{k=-\infty}^{0} \mathbf{U}_k. \tag{2}$$

The parametrization method (see Section 3) can accurately approximate $W^{\mathrm{u}}_{\mathrm{init}}$ and is used to define the approximation U_0 needed to seed iteration (1).

It is natural to parameterize each annulus in polar (r,θ) coordinates. Instead, we introduce a radial-like variable t and parameterize $\mathbf{U}_k$ using $(t,\theta) \in [k, k+1) \times [0, 2\pi)$—see Section 3 for an explanation of the parameter t.[2] Then the next annulus can be parameterized

$$\mathbf{U}_{k+1}(t+1,\theta) = f(\mathbf{U}_k(t,\theta)). \tag{3}$$

This reduces the problem of computing W^{u} to that of simply computing a parametric surface.

The simplest way to compute each annulus would be to specify a collection of points on the annulus $\mathbf{U}_0$ and to generate the approximation U_n using the nth image of those points. This leads to several problems. First, the distribution of points on each annulus is not controlled and simple iteration may place points very closely on some parts of $\mathbf{U}_n$, with large gaps between points on other parts. Further, the dynamics lead to exponential and anisotropic growth, so that the number of points necessary to resolve $\mathbf{U}_n$ increases with n; see Fig. 2. Thus a method that can adaptively select the distribution and number of points on each annulus is required.

3. Existing methods

Two existing, widely cited methods suited for computing two-dimensional manifolds of maps are the parametrization method and the geodesic level set method (GLSM).

The Parameterization Method. This method is the basis for the recent numerical computations of heteroclinic manifolds of maps [9,11] and has been used as both an analytical and a numerical tool [12–15]. Near the fixed point $\mathbf{x}^*$, the manifold W^{u} is represented by a power series in parameters $(\sigma,\tau) = (r\cos\theta, r\sin\theta)$, with coefficients determined by the manifold's invariance under the map f and its tangency to the unstable subspace.

If f is analytic, the series has an infinite radius of convergence, but roundoff error usually makes the numerical radius of convergence quite small. While the method is very accurate near the origin, a complementary method is needed to compute larger portions of the manifolds. We use the parameterization method to compute an approximation to $\mathbf{U}_0$, and, in fact, all of $W^{\mathrm{u}}_{\mathrm{init}}$ of Eq. (2). The parameterization method does not prescribe a way to choose which points on the manifold to compute, nor how to construct the surface between them, which is the principal aim of our method. Another method for computing $W^{\mathrm{u}}_{\mathrm{init}}$ is due to Homburg et al. [16].

The Geodesic Level-Set Method. This method, due to Krauskopf and Osinga [4,7], drops the idea of iterating a fundamental domain in favor of "growing" the manifold at a uniform rate in each direction. The computed manifold consists of a concentric family of annuli, with each annulus triangulated as in Fig. 3a (W^{u} is shown as a plane in this figure for clarity). The algorithm begins by defining a "foliation," a set of planes ("leaves") that intersect in the stable subspace of $\mathbf{x}^*$; see Fig. 3b. At each step, the algorithm adds a ring of points which are connected with line segments to form a topological annulus immediately exterior to the computed manifold. It searches for points on the next annulus approximately satisfying two conditions: each point lies on a leaf of the foliation and lies at a fixed geodesic distance Δ from the previous annulus. The added points are, of course, images of points on the previously computed triangulated manifold.

The method features two forms of adaptation. First, as the distance from $\mathbf{x}^*$ to the outer annulus increases, so does the distance between adjacent leaves of the foliation, so additional leaves are added to keep the distance between leaves below a specified tolerance $\Delta_{\mathcal{F}}$. Second, if the algorithm fails to find a complete ring of new points, it decreases Δ and tries again.

This attempt to grow the manifold uniformly is intended to alleviate a major shortcoming of iterating fundamental domains: exponentially anisotropic growth in the case that the unstable eigenvalues of $\mathbf{F}$ satisfy $1 <| \lambda_{u_2} |<| \lambda_{u_1} |$. The authors give an example of a map (the fattened Arnol'd map) whose unstable manifold grows a long finger under iterations which

[2] In polar coordinates $0 \leqslant r < \infty$, while in this coordinate system $-\infty < t < \infty$.

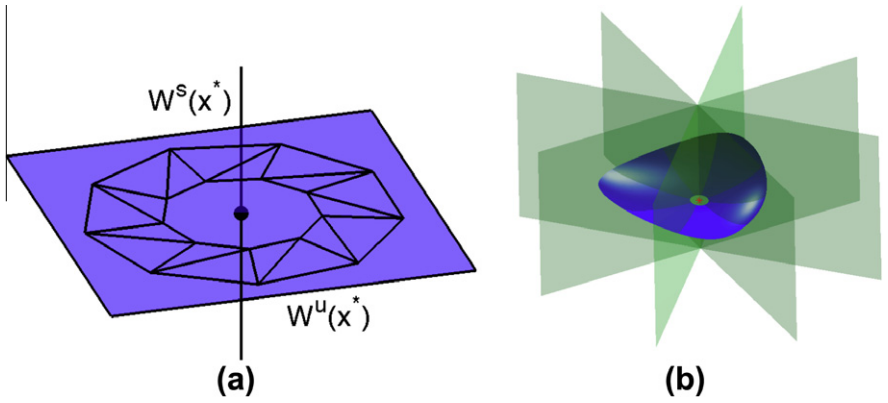

Fig. 3. Schematic application of the GLSM. (a) One triangulated annulus on $W^{\mathrm{u}}(\mathbf{x}^*)$, shown as a plane for clarity. (b) A portion of the manifold (blue) and a few foliation leaves (green). (For interpretation of the references to colour in this figure caption, the reader is referred to the web version of this article.)

subsequently re-enters the neighborhood of the fixed point. They show that GLSM computes a more useful portion of the manifold. GLSM, however, has major shortcomings which may make it, too, fail:

1. Its dependence on the foliation breaks down if the manifold is ever tangent to a leaf (the foliation condition).
2. In order to compute a new ring of points for the manifold, each point on the desired ring must have a pre-image that has already been computed. The method fails if there does not exist a proper loop at a non-trivial positive distance from the thus-far computed manifold. We demonstrate this numerically below.
3. The method requires setting a spatial scale for the distance between leaves before beginning the computation. We found in [1] that W^{u} can develop large curvature after just a few iterates ($O(10^6$ in one standard example we tried). Setting such a scale would either under-resolve certain features or else require a very large number of points. This would be a problem for the ACT map (11) studied in Section 6.3.

Krauskopf and Osinga discuss point 1, but we have found no reference to point 2 in the literature. We encountered the second problem when we attempted to apply this method to the volume-preserving Hénon map in Section 6.2 and were then able to construct a simpler illustrative example of this failure.

Counterexample. Section 5.3 of Ref. [4] contains the following numerical test of the GLSM algorithm. It considers a map of the form

$$f(\mathbf{x}) = \phi \circ \Lambda \circ \phi^{-1}(x), \tag{4}$$

where $\Lambda(\mathbf{x}) = \Lambda \cdot \mathbf{x}$ and Λ is a diagonal matrix satisfying the hyperbolicity assumption and ϕ maps the plane $z = 0$ to a curved surface Σ, in such a way that $W^u = \Sigma$. It shows that GLSM grows this manifold uniformly, and that the pointwise error decreases under stricter error tolerances. Thus this would seem to be preferred over computing iterates of fundamental domains. However, we present here an example where GLSM fails.

Consider the linear map

$$\mathbf{x}_{n+1} = f(\mathbf{x}_n) = \begin{pmatrix} 0 & -AB & 0 \\ A/B & 0 & 0 \\ 0 & 0 & 1/2 \end{pmatrix} \mathbf{x}_n \text{ with } B > A > 1. \tag{5}$$

The origin is a fixed point and $W^{\mathrm{u}}(0)$ is simply the xy-plane. The circle $x^2 + y^2 = r^2;\ z = 0$ is not a proper loop, as it intersects its own image in four places. The ellipse

$$\Gamma_0 = \left\{ (x, y, z) \middle| \frac{1}{B^2} x^2 + y^2 = r^2; \quad z = 0 \right\}$$

is, by contrast, a proper loop. Its image $f(\Gamma_0)$ is the ellipse $\frac{1}{B^2} x^2 + y^2 = A^2 r^2$; see Fig. 4.

To understand the action of GLSM applied to this system, recall that an *offset curve* of a smooth curve $\gamma \subset \mathbb{R}^2$ is a curve $O_\Delta(\gamma)$ consisting of points in $\mathbb{R}^2$ a fixed normal distance Δ from γ. A closed convex curve has two such offset curves, and $O_\Delta(\gamma)$ will refer to the one *exterior* to γ. If γ is smooth and convex, so is $O_\Delta(\gamma)$.[3] The action of GLSM, when applied to an example in which $W^{\mathrm{u}}(0)$ lies in the plane is to define a sequence of curves[4]

[3] This may be false for interior offset curves or if γ is non-smooth or non-convex.

[4] More precisely, it defines discrete sets of points lying on a sequence of curves.

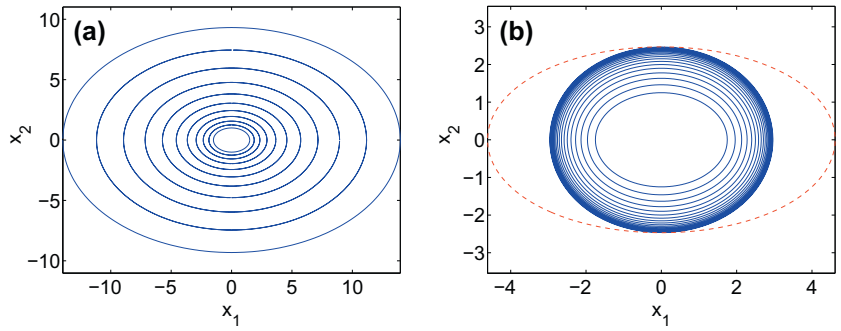

Fig. 4. (a) Iterates of a proper loop under map (5) with $A = 5/4,\ b = 3/2$. (b) The family of recursively defined offset curves that converge to χ_∞ which is tangent to $f(\chi_\infty)$, dashed.

$$\chi_{k+1} = O_{\Delta_k}(\chi_k); \quad \chi_0 = \Gamma_0.$$

To proceed, it must be able to find an offset Δ_k such that χ_{k+1} has a pre-image interior to χ_k. If Δ_k is chosen to be constant, then the curves χ_k become increasingly circular with each increasing k and, for large enough k, χ_{k+1} is not a proper loop. In this case, the algorithm fails.

GLSM does not, however, use constant Δ: if the algorithm finds no pre-image for a point on the offset curve, it reduces Δ until it succeeds. We have written a program that at each step finds the largest Δ_k for which $f^{-1}(O_{\Delta_k}(\chi_k)) \subset \mathbf{int}\chi_k$. Fig. 4b shows the results of a numerical experiment in which the algorithm finds $\Delta_k \to 0$ exponentially at the rate $\Delta_k \propto e^{-1.23k}$. Thus there exists a limiting curve such that $\chi_k \to \chi_\infty$.

GLSM fails in a similar manner when the matrix in Eq. (5) has two distinct real unstable eigenvalues and nearly parallel eigenvectors. In this case, shear, rather than rotation, prevents the circle from being a proper loop.

Moreover, GLSM has other faults. It is based on an interpolation by piecewise-planar triangular patches and thus has quadratic accuracy. This coarse interpolation error is magnified exponentially upon iteration, as demonstrated numerical testing of one-dimensional unstable manifold algorithms in [1]. In addition, GLSM is inefficient: to find a point at a given distance on a certain leaf, it uses a search algorithm, meaning that it computes and discards many points, leading to further inefficiencies in the method.

Other methods. A survey of methods for computing invariant manifolds in the case of *vector fields* appeared in [17]. In general, computation of the invariant manifolds of continuous dynamical systems differs from that for discrete systems. In order to handle differential equations by a method formulated for discrete time systems, one may use a corresponding time-τ map.

Reference [17] describes two methods that are applicable specifically to maps, GLSM and the **Box Covering** algorithm [8], which approximates W^u as a subset of $\mathbb{R}^3$ by dividing space into cubes and keeping track of which cubes contain at least one point on the manifold. Because the method represents the surface as a pixelated (more exactly, voxelated) object, it may be difficult to use it to compute quantities such as curvature.

4. Necessary ingredients from CAGD

We describe here the specific ideas from CAGD that are used in approximating the manifold. Two good texts on these CAGD concepts, with references to the original research, are [18,19]. We refer to the method we derive as the adaptive Bézier quasi-interpolation (ABQ) method.

The approximate annulus U_k is parameterized in polar-like (t, θ) coordinates with $k \leqslant t \leqslant k+1$, and $0 \leqslant \theta < 2\pi$. The word "data" is used below to refer to the function values $f(U_k(t, \theta))$ used to construct U_{k+1}:

Type-1 triangulation. The surface is defined by piecewise polynomial patches, defined over a *Type 1* triangulation of the parametric region, that is, a uniform rectangular grid in which each rectangle is split into triangles by a diagonal line, all with the same orientation; Fig. 5a. The adaptive scheme produces a "pseudo-regular" triangulation; Fig. 5c.

Triangular Bézier patches. The polynomial patch on each triangle is a piecewise bivariate triangular Bézier polynomial of degree n, which is defined in terms of $\binom{n+1}{2}$ *control points* $\mathbf{b}_{i,j,k}$, using the Bernstein polynomials over local barycentric coordinates as a basis. These are the most natural generalization of Bézier curves.

The de Casteljau algorithm is an efficient method for evaluating and manipulating Bézier patches which is based on iterated linear interpolation of the control points. It is used extensively in the implementation of the approximation algorithm; see Fig. 6. The algorithm is also employed by the subdivision algorithm described below. Bézier triangles and the de Casteljau algorithm were both discovered and developed, independently, by Bézier and de Casteljau in the 1950's and 1960's [20–23].[5]

[5] Bézier and de Casteljau worked, respectively, at the French automakers Citroën and Renault, and their work was for many years a trade secret [18, Ch. 1, by P. Bézier].

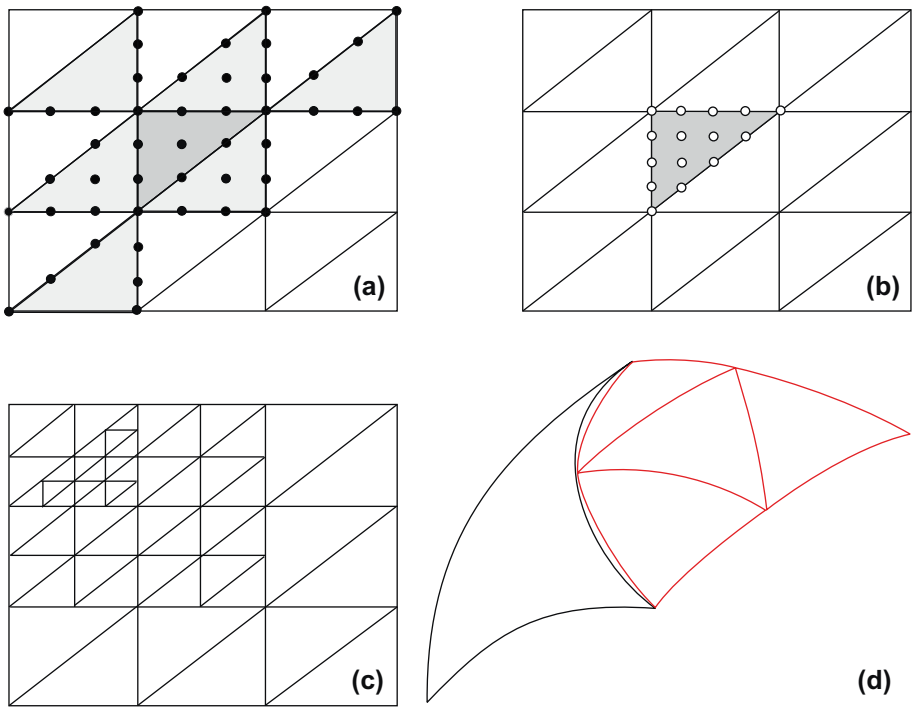

Fig. 5. (a) A type-1 triangulation and the 46 domain points used to construct control points of a single (darkest) Bézier triangle. (b) The resulting 15 control points. (c) A pseudo-regular type-1 triangulation. (d) Discontinuity of composite Bézier patches with different resolutions.

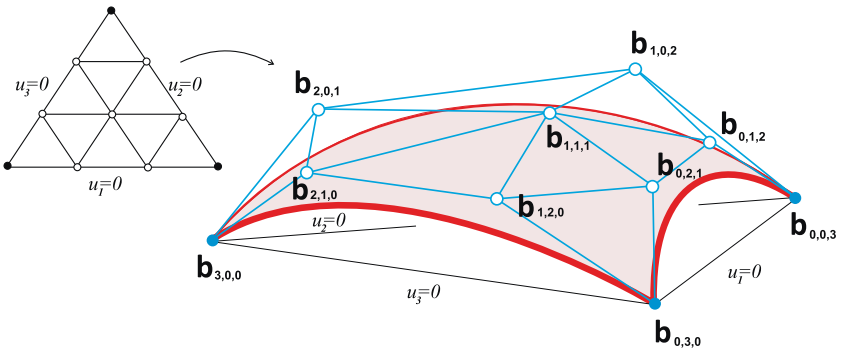

Fig. 6. Cubic Bézier patch (red), its control net (blue), and domain triangle (left). The u_j are *barycentric coordinates*. (For interpretation of the references to colour in this figure caption, the reader is referred to the web version of this article.)

Quasi-interpolation on regular and pseudo-regular triangulations. The approximating surface is a *quasi-interpolant* of the data. Quasi-interpolation refers to a family of approximations that satisfy weaker conditions than interpolation, namely that the approximation is exact when applied to polynomials of a certain degree or below. The Bézier control points for a given patch are determined from the data on that patch and on the neighboring patches, without having to solve a system of equations over the entire data set as would be necessary, for example for the better-known C^2-cubic spline. Quasi-interpolation was first described by de Boor and Fix [24].

The quasi-interpolation method used is due to Sorokina et al. [25]. Each annulus is triangulated using a type-1 triangulation. The resulting surface is C^1 on each fundamental domain. Cubic patches are insufficient to satisfy both the quasi-interpolation condition and the C^1 condition, so quartic Bézier patches are required. Ten data points are computed per triangle, and forty-six data points from a triangle and its nearest neighbors are used to compute the fifteen control points that define the quartic Bézier patch; Fig. 5a and b.

Because the control points on a given triangle depend on the data points on neighboring triangles, care must be taken for triangles along the annulus edges. In particular, a row of "buffer triangles" is included during the calculation. In these triangles, the data is chosen to maintain C^1 continuity with the interior triangles and to vanish, along with its derivative, at its outer boundary.

To approximate W^{u}, we use an adaptive modification of Sorokina's quasi-interpolant, due to Hering-Bertram et al. [5]. The algorithm in this method estimates the approximation error on each patch T and, where the estimate exceeds a tolerance, the three edges of the patch are bisected, producing four triangular patches of half the side-length. The method works by identifying the set of triangles $\mathcal{T}_{\mathrm{fail}}$ for which the error exceeds a prescribed tolerance. Letting $g(x)$ be the function being approximated and $s_0(x)$ be the quasi-interpolant, the algorithm defines a correction function

$$\Delta_0(x) = \begin{cases} g(x) - s_0(x); & x \in T_i \in \mathcal{T}_{\mathrm{fail}} \\ 0, & \text{otherwise}. \end{cases}$$

It then computes a quasi-interpolant δ_0 of Δ_0 and adds this to the previously computed $s_0(x)$ to form the improved approximation $s_1(x)$, iterating this procedure until the tolerance condition is met everywhere.

Some notes on this algorithm: first, although the above description defines a global correction, the correction is in practice applied only locally, and the different levels of correction are stored in a recursive quadtree data structure. Second, the correction at each step extends beyond the set $\mathcal{T}_{\mathrm{fail}}$, by one row of triangles on each side, so that the quasi-interpolating correction δ_0 varies smoothly between the regions where it is non-zero and those where it is zero. Finally, an important and non-trivial accomplishment of [5] is that the quasi-interpolant is C^1 across boundaries connecting grids of two different refinements. For an example of a composite surface that fails to be even C^0 at such a junction; see Fig. 5d.

5. Numerical implementation

Parameterizing the annuli. For simplicity assume, $\mathbf{x}^* = 0$. An approximate primary annulus on $W^{\mathrm{u}}(0)$ is constructed using the linearization matrix $\mathbf{F}$.

If $\mathbf{F}$ has two unstable eigenvalues $\lambda_{u_1} > \lambda_{u_2} > 1$, with eigenvectors $\vec{\mathbf{v}}_1$ and $\vec{\mathbf{v}}_2$, the region

$$\mathbf{U}_0 = \left\{ \lambda_{u_1}^t \vec{\mathbf{v}}_1 \cos\theta + \lambda_{u_2}^t \vec{\mathbf{v}}_2 \sin\theta \mid (t,\theta) \in [0,1] \times [0,2\pi) \right\} \tag{6}$$

is bounded by proper loops $\Gamma_0 = \mathbf{U}_0|_{t=0}$ and $\Gamma_1 = \mathbf{U}_0|_{t=1} = f(\Gamma_0)$ and is thus a parameterized fundamental domain. Because the growth in the $\vec{\mathbf{v}}_1$-direction is exponentially faster than in the $\vec{\mathbf{v}}_2$-direction, almost all of the discrete set of points on Γ_1 will lie nearer to the $\vec{\mathbf{v}}_1$-direction, in relative terms, than their pre-images on Γ_0, and this trend will worsen exponentially with iteration—the so-called clustering problem. To compensate for both anisotropic growth and gridpoint clustering, it is useful to scale the eigenvectors such that $\|\vec{\mathbf{v}}_1\| = (\lambda_{u_2}/\lambda_{u_1})^n \|\vec{\mathbf{v}}_2\|$ where n is the planned number of iterates.

If $\mathbf{F}$ has a pair of complex unstable eigenvalues

$$\lambda_{u_1} = \rho e^{i\Theta} \text{ and } \lambda_{u_2} = \rho e^{-i\Theta}, \tag{7}$$

with eigenvectors $\vec{\mathbf{v}}_1 = \vec{\mathbf{u}} + i\vec{\mathbf{w}}$ and $\vec{\mathbf{v}}_2 = \vec{\mathbf{v}}_1^*$, a fundamental domain is

$$\mathbf{U}_0 = \left\{ \beta\rho^t \left(\vec{\mathbf{u}} \cos(\theta + \Theta t) - \vec{\mathbf{w}} \sin(\theta + \Theta t) \right) \middle| (t,\theta) \in [0,1] \times [0,2\pi) \right\}, \tag{8}$$

with $\Gamma_0 = \beta(\vec{\mathbf{u}} \cos\theta - \vec{\mathbf{w}} \sin\theta)$, $\Gamma_1 = \beta\rho(\vec{\mathbf{u}} \cos(\theta + \Theta) - \vec{\mathbf{w}} \sin(\theta + \Theta))$, and β chosen small enough that the parameterization method provides a sufficiently accurate approximation.

Although the above discussion is for the linearized map $\mathbf{F}$, the numerical code uses the same region of parameter space to implement the parameterization method. The parameterizations given in Eqs. (6) and (8) are used to compute $W^{\mathrm{u}}_{\mathrm{local}}$, which is globalized using Eq. (3).

Triangulating U_0. Care must be taken in triangulating the initial primary annulus U_0. The obvious method would be to use a uniform cartesian grid with gridlines corresponding to lines of constant t and lines of constant θ. This approach fails for two reasons. First, to achieve continuity between domains U_0 and U_1, the two parameterizations must share common grid points along their shared boundary Γ_1. Consider the linear map with complex unstable eigenvalues as in Eq. (7). Choose grid points with $t = 0$ and $\theta_k = 2\pi k/n$ on Γ_0. The images of these points lie on Γ_1 with angles $\theta = \theta_k + \Theta$. Since Θ is not, in general an integer multiple of $2\pi/n$, these points do not match the grid points from U_0. Second, if Θ is large, the triangulation on the

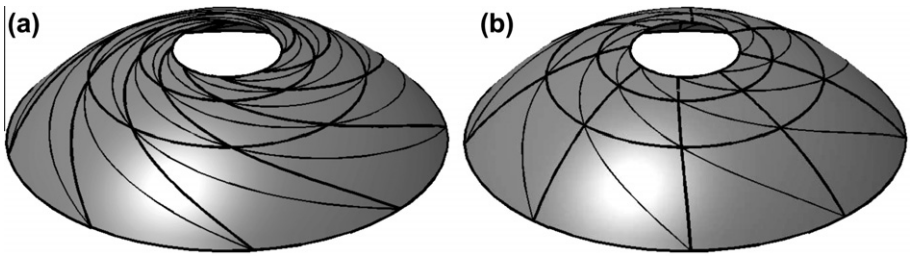

Fig. 7. (a) Schematic of a non-orthogonal type-1 triangulation in (t,θ) parameters. (b) Schematic of a more closely-orthogonal triangulation in (t,ψ) parameters.

manifold (as opposed to the triangulation in parameter space) will be formed by triangles with a large aspect ratio that twist around the manifold, as in Fig. 7a.

The solution to both these problems is to replace the gridlines of constant θ in parameter space with sheared gridlines, along which $\psi = \theta + \omega t$ is constant, with ω chosen as follows. Find a rational number p/q such that

$$\Theta \approx 2\pi p/q. \tag{9}$$

Then choose $\omega = \Theta - \frac{2\pi}{q}$. This straightens the triangulation, as seen in Fig. 7b, and aligns the grid points along the boundaries.

Joint patching. Because the algorithm constructs each primary annulus separately, smoothness, and even continuity conditions may not hold at the annulus boundaries—even though the function values on boundary grid points from both sides are identical. This occurs when the resolution at the two sides of the boundary differ (see Fig. 5d). The magnitude of this gap is generally smaller than the tolerance used for refining the triangulation. To fix this, the triangles on both sides are refined to the same level using the de Casteljau algorithm [18]. This is straightforward and computationally inexpensive. While the resulting surface is only C^0 across the boundary, numerical tests show no higher errors here than elsewhere on the surface and the resulting graphics show no visible artifacts along the boundary. Algorithms that produced C^1-continuity ran much slower with no discernible benefit.

Distance control. Due to the anisotropic growth of the manifold under iterations of f, the ABQ method may compute some portions of W^{u} outside the region of interest before finding other needed points of the manifold closer to $\mathbf{x}^*$. When the domains $\mathbf{U}_k$ grow at a very nonuniform rate, these faraway points may be pre-images of the desired points, and we cannot avoid computing them. In many cases, however, it can be shown analytically that the orbits of points outside some compact region $C \subset \mathbb{R}^3$ never re-enter C. Thus, it may be efficient to instruct the algorithm not to refine portions of U_k that lie outside of C or not to refine the surface for regions that are far from $\mathbf{x}^*$ in arc-length along grid lines of constant ψ as shown in Fig. 7.

6. Numerical tests

Testing a method for computing two-dimensional invariant manifolds is harder than for a one-dimensional manifold. In [9] the accuracy of the parameterization method is tested by evaluating a residual. The power series is constructed to satisfy some nonlinear equation, and the residual measures how well the numerically-evaluated series does so. This is a weaker condition than a direct measurement of the distance between the exact and computed manifolds. The residual is controlled only in a small neighborhood of the fixed point, so that iteration must be used beyond that neighborhood.

Below, we describe the results of numerical tests showing the performance of the proposed method on the global portion of the invariant manifolds for three model systems.

6.1. Example 1: a map with explicit unstable manifold

We compare ABQ with GLSM quantitatively using the map from Section 5.3 of reference [4], Eq. (4), with $\Lambda = \mathrm{diag}(2.1, 15.3, 0.6)$ and $\varphi(x,y,z) = (x, y, z - 0.2x^2 + 0.1y^2)$. This map satisfies the hyperbolicity assumption of Section 2, with two unstable eigenvalues of greatly varying magnitude, so that naïvely iterating the map on fundamental domains leads to highly anisotropic growth and clustering of points.

We implement GLSM with tolerances $\Delta = \Delta_{\mathcal{F}} = 0.3$, an initial annulus in E^u of outer radius 0.05 and 40 foliation leaves. The manifold is grown to a geodesic distance 12.14 at its outer edge, which contains 494 points; see Fig. 8a. The computed manifold contains 9320 points, it has errors below the values in Table 1 of [4] and maximum error below 3×10^{-3}, measured at mesh points, at the midpoints of mesh edges and at the center of each triangle. Because the GLSM algorithm searches for well-spaced points on W^{u}, the number of calls to the map f is many times higher than the number of points forming the computed manifold.

We use ABQ to compute a slightly larger piece of the manifold with approximately the same maximum error, 3×10^{-3}. To compensate for both anisotropic growth and gridpoint clustering we follow the method discussed following Eq. (6) and scale the eigenvectors such that $\|\vec{\mathbf{v}}_i\| = \lambda_i^{-n} R_{\mathrm{outer}}$, where $R_{\mathrm{outer}} = 12.15$ is the outer radius, in parameter space of the computed manifold. We use $n = 8$ fundamental annuli, and a tolerance condition 2×10^{-3}. Since $\lambda_{u_2}/\lambda_{u_1} \approx 7.3$, the proper loop at the inner edge of the manifold has major and minor axes 3×10^{-2} and 4×10^{-9}, an aspect ratio of about 1.26×10^{-7}. The computed manifold consists of 1338 Bézier triangular patches based on 8273 data points; see Fig. 8b. The number of calls to the map f is a fraction higher because of the buffer points at the edge of each computed annulus. Calculation and rendering take under a minute on the author's laptop while the calculation using GLSM took more than a day on the same computer, although we made little attempt to optimize this program.

6.2. Example 2: volume-preserving Hénon map

Following [9,26], we consider the volume-preserving Hénon map:

$$f(x,y,z) = \begin{pmatrix} \alpha + \tau x + z + ax^2 + bxy + cy^2 \\ x \\ y \end{pmatrix}. \tag{10}$$

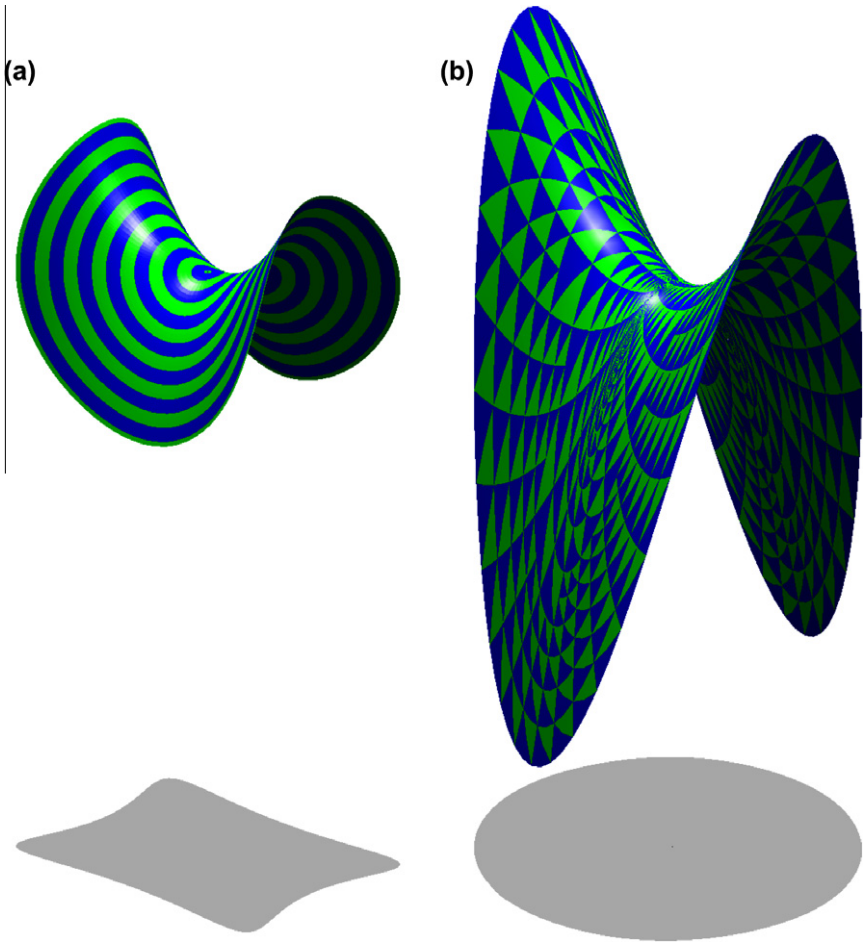

Fig. 8. The unstable manifold of $f = \varphi \circ \Lambda \circ \varphi^{-1}$, (a) by GLSM method, alternating color every three annuli, (b) by ABQ method with $\mathbf{tol} = 2 \times 10^{-3}$. Both subfigures show the projection of the map onto the subspace $z = 0$, demonstrating that GLSM computes a surface of a certain size measured geodesically, while ABQ computes up to a certain radius in parameter space.

This has two hyperbolic fixed points $\mathbf{x}_\pm = (x_\pm, x_\pm, x_\pm)$ where $x_\pm = -\tau/2 \pm \sqrt{\tau^2 - 4\alpha}/2$. For parameters $(a, b, c, \alpha, \tau) = (0.44, 0.21, 0.35, -0.25, -0.3)$, the manifolds $W^{\mathrm{u}}(\mathbf{x}_+)$ and $W^{\mathrm{s}}(\mathbf{x}_+)$ are, respectively two- and one-dimensional, while $W^{\mathrm{u}}(\mathbf{x}_-)$ and $W^{\mathrm{s}}(\mathbf{x}_-)$ are, respectively, one- and two-dimensional. The two unstable eigenvalues of $\mathbf{F}(\mathbf{x}_+)$ are complex, as are the stable eigenvalues of $\mathbf{F}(\mathbf{x}_-)$.

The parametrization method described in Section 3 is used to compute U_0 for $W^{\mathrm{u}}(\mathbf{x}_+)$, truncating the series $P(r, \theta)$ at degree 60. Since the map (10) is explicitly invertible, the method is also used to compute $W^{\mathrm{s}}(\mathbf{x}_-)$.

Computed manifolds are shown in Fig. 9a. For these parameter values, the dynamics near the fixed point feature relatively modest growth and compression with eigenvalues $|\lambda_s| = 0.8482$, $|\lambda_{u_{1,2}}| = 1.1737$. The rotational component of the map is visualized by following the trajectory of one triangle over several iterates in Fig. 9b. To compensate for the rotation, we take $p/q = 1/3$ in Eq. (9). The manifold looks like an onion with a wide "bulb" near the fixed point and a narrow "stalk." Wrapped around the stalk are three rapidly growing regions interspersed with slower-growing portions that fold back inward.

It is shown in [26] that the map satisfies the conditions described for using "distance control" as sketched in Section 5; see Fig. 9c. We test the convergence of the numerical algorithm as follows. We compute the "exact" initial $\mathbf{U}_0(t, \theta)$ using the parameterization method. For a given tolerance $\mathbf{tol}_j$, we compute $U_k(t, \theta; \mathbf{tol}_j)$ using ABQ, and compute the maximum error by $\mathbf{err}_j(t, \theta) = |f^k(\mathbf{U}_0(t, \theta)) - U_k(t, \theta; \mathbf{tol}_j)|$ over a large number of computed points. Fig. 9d shows the computed error in U_{15} decays rapidly with the number of triangular patches used to resolve U_k with decreasing tolerance. The parameterization of the surfaces makes direct comparison simple.

Fig. 10 shows an attempt to calculate $W^{\mathrm{u}}(\mathbf{x}_+)$ using GLSM. In red is $W^{\mathrm{u}}_{\mathrm{init}}$ computed by the parameterization method, and in green are the rings added by GLSM. The distance between rings decreases rapidly with each iteration, and it becomes impossible to find a proper loop at a non-trivial geodesic distance from the computed boundary χ_∞. The blue curve is $f(\chi_\infty)$, which is tangent to χ_∞ near the bottom of the image. This is the same failure mechanism as illustrated in Fig. 4.

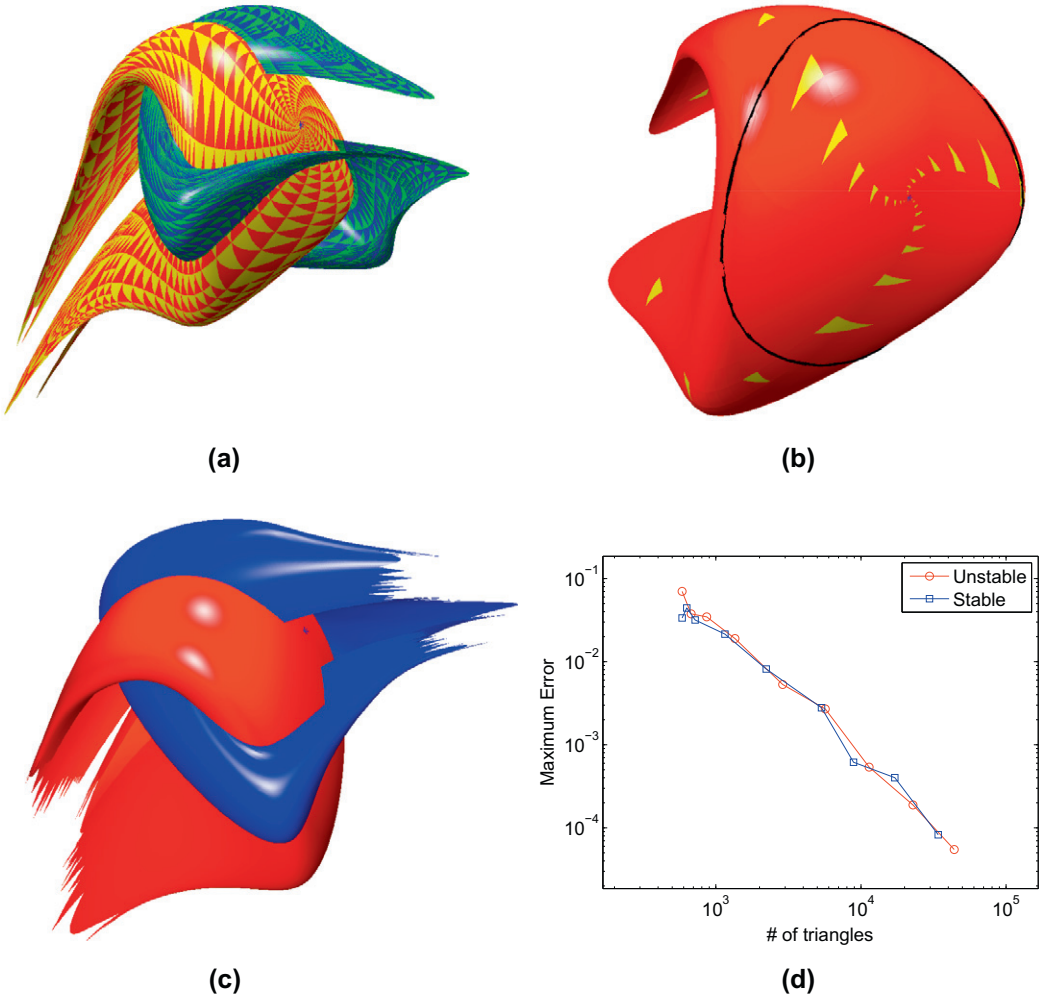

Fig. 9. Computed manifolds of the Hénon map (10) with a 1×15 initial triangulated mesh and **tol** $= 10^{-3}$. (a) W^s (red/yellow) and W^u (blue/green) showing 15 iterations of the initial annulus. The initial annulus is computed to machine precision by the parameterization method truncated to degree 35. (b) Computation showing the orbit of one patch highlighted in yellow. (c) Computation using distance cutoff to avoid computing too far along the stalk. (d) Convergence of the AQI method for this example with tolerance decreasing from **tol** = 0.1 to **tol** = 0.1×2^{-16}. (For interpretation of the references to colour in this figure caption, the reader is referred to the web version of this article.)

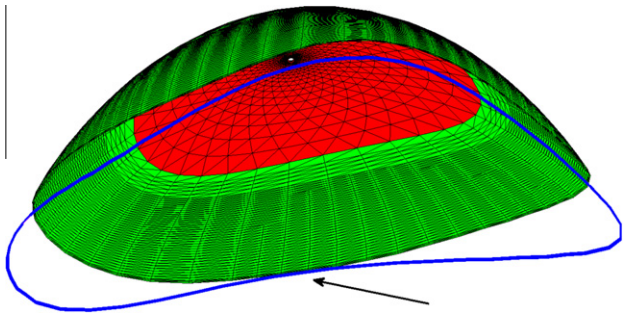

Fig. 10. The initial portion of W^u computed by the parameterization method (red) and GLSM (green), with $f(\chi_\infty)$ in blue, and an arrow indicating the point where χ_∞ and $f(\chi_\infty)$ are tangent. The curve χ_∞ is also drawn as a black line in Fig. 9b. (For interpretation of the references to colour in this figure caption, the reader is referred to the web version of this article.)

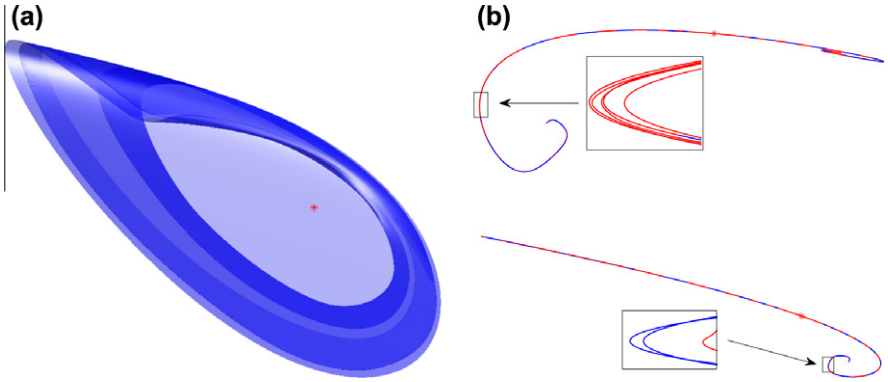

Fig. 11. (a) The unstable manifold of the ACT map (11). (b) Intersections of the manifold with planes containing the stable direction. The aspect ratio is not preserved in the closeups.

6.3. Example 3: Arneodo–Coullet–Tresser map

The next example, the Arneodo–Coullet–Tresser map (ACT), shows that the method can accurately resolve a typical feature of unstable manifolds: sharp folds where the manifold has very high curvature and doubles back on itself many times in a small region. Accurate computations of such structures in the two-dimensional dissipative Hénon map were made in [1]. This map has been extensively studied by Du et al., e.g. in [27]. The quadratic ACT map is

$$f(x,y,z) = \begin{pmatrix} ax - \omega b(y-z) \\ \frac{b}{\omega}x + a(y-z) \\ cx - dx^2 + ez \end{pmatrix} \tag{11}$$

with fixed points at the origin and at $\mathbf{x}^* = \left(x_1, \frac{a^2+b^2-a}{\omega b}x_1, \frac{(a-1)^2+b^2}{\omega b}x_1\right)$ with $x_1 = \frac{bc-(1-e)\left((a-1)^2+b^2\right)}{\omega bd}$. With parameter values $(a,b,c,d,e,\omega) = (0.2, 0.5, 0.5, 1, 1, 4)$, the manifold $W^{\mathrm{u}}(\mathbf{x}^*)$ is two-dimensional with complex-conjugate unstable eigenvalues. Fig. 11a shows the computed manifold with these parameters. Transparency is used to show its folds which are similar to those in the attractor for the Rössler ODE system. Fig. 11b shows intersections of $W^{\mathrm{u}}(\mathbf{x}_*)$ with planes containing the stable subspace of $\mathbf{x}^*$, highlighting the well-resolved fine structure. GLSM, which has a pre-set resolution, cannot compute such folds.

7. Discussion

The adaptive Bézier quasi-interpolation method combines locally-defined, higher-degree splines from CAGD with local adaptivity to efficiently construct numerical approximations to unstable manifolds of iterated maps. Iterated maps are but one type of dynamical system that may possess stable and unstable manifolds. The ABQ method should be adaptable to computing manifolds in other types of dynamical systems (e.g. ordinary differential equations and delay differential equations) following the discussion in [28], for example by computing the invariant manifolds of a time-τ map or Poincaré map.

What is more, fixed points are not the only invariant sets that may have stable and unstable manifolds. For example hyperbolic periodic orbits in continuous-time systems, and hyperbolic invariant tori in discrete time systems may each have them. The parameterization method has been used to compute local invariant manifolds in these cases [29–32], and the ABQ method might be extended to efficiently extend the invariant manifolds away from the invariant tori.

A fundamental obstacle to efficiently computing invariant manifolds is posed by anisotropic growth. The simplest approach to growing a manifold, and the one taken here, is iterating the map over a sequence of fundamental domains. Krauskopf and Osinga identified this obstacle, e.g. in [4], and proposed a method that would grow the manifold at a uniform rate in all directions. Unfortunately, we have identified scenarios in which this approach will terminate unsuccessfully, since the boundary of the computed manifold may not be a proper loop. This is in addition to less serious problems discussed in [28]. Therefore it remains to find an algorithm that avoids both the problems caused by non-proper loops and by anisotropic growth.

While there is still no method that can provide accurate and efficient results in all cases, the numerical tests applied here show the ABQ method can generate highly accurate manifolds.

Software

The authors have written a small suite of MATLAB programs that implement the methods described in this article. These are also available from the author's website http://web.njit.edu/goodman/Numerics.html.

Acknowledgments

Thanks to Gershon Elber, Rafael de la Llave, Hector Lomelí, James Meiss, Jason Mireles-James, and Denis Zorin for helpful discussions, to the anonymous referees, whose suggestions improved this paper enormously, and to Casayndra Basarab for her careful reading of the manuscript. The authors were supported by the NSF under Grant Nos. DMS-0807284, DMS-0639270.

References

[1] Goodman RH, Wróbel JK. High-order bisection method for computing invariant manifolds of two-dimensional maps. Int J Bifurcation Chaos 2011;21:2017–42.
[2] Carter JA. Bisection method for computing invariant manifolds of 2-D maps; Preprint, posted, with the author's permission. <http://web.njit.edu/goodman/publications/carter.pdf>.
[3] Hobson D. An efficient method for computing invariant manifolds of planar maps. J Comput Phys 1993;104:14–22.
[4] Krauskopf B, Osinga H. Globalizing two-dimensional unstable manifolds of maps. Int J Bifurcation Chaos 1998;8:483–503.
[5] Hering-Bertram M, Reis G, Zeilfelder F. Adaptive quasi-interpolating quartic splines. Computing 2009;86:89–100.
[6] Krauskopf B, Osinga H, Doedel EJ, Henderson ME, Guckenheimer J, Vladimirsky A, et al. A survey of methods for computing (un)stable manifolds of vector fields. Int J Bifurcation Chaos 2004;15:763–92.
[7] Krauskopf B, Osinga H. Growing 1D and quasi-2D unstable manifolds of maps. J Comput Phys 1998;146:404–19.
[8] Dellnitz M, Hohmann AA. Subdivision algorithm for the computation of unstable manifolds and global attractors. Numer Math 1997;75(3):293–317.
[9] Mireles-James JD, Lomelí H. Computation of heteroclinic arcs with application to the volume preserving Hénon family. SIAM J Appl Dyn Syst 2010;9:919–53.
[10] Lomelí HE, Meiss JD. Heteroclinic primary intersections and codimension one Melnikov method for volume-preserving maps. Chaos 2000;10:109–21.
[11] Mireles-James JD. Quadratic volume-preserving maps: (un)stable manifolds, hyperbolic dynamics, and vortex-bubble bifurcations. Int J Bifurcation Chaos; submited for publication.
[12] Cabré X, Fontich E, de la Llave R. The parameterization method for invariant manifolds I: manifolds associated to non-resonant subspaces. Indiana Univ Math J 2003;52:283–328.
[13] Cabré X, Fontich E, de la Llave R. The parameterization method for invariant manifolds II: regularity with respect to parameters. Indiana Univ Math J 2003;52:329–60.
[14] Cabré X, Fontich E, de la Llave R. The parameterization method for invariant manifolds III: Overview and applications. J Differ Equ 2005;218:444–515.
[15] Franceschini V, Russo L. Stable and unstable manifolds of the Henón mapping. J Stat Phys 1981;25:757–69.
[16] Homburg AJ, Sands D, de Vilder R. Computing invariant sets. Int J Bifurcation Chaos 2003;13:497–504.
[17] Krauskopf B, Osinga H, Doedel EJ, Henderson ME, Guckenheimer J, Vladimirsky A, et al. A survey of methods for computing (un)stable manifolds of vector fields. Int J Bifurcation Chaos 2005;15:763–92.
[18] Farin G. Curves and surfaces for CAGD: a practical guide. Fifth. San Francisco, CA: Morgan-Kaufmann; 2002.
[19] Goldman R. Pyramid algorithms: a dynamic programming approach to curves and surfaces for geometric modeling. Morgan-Kaufmann; 2003.
[20] de Casteljau P. Outillages methodes calcul. Tech. Rep. Citroën, Paris; 1959.
[21] de Casteljau P. Courbes et surfaces à pôles. typeTech. Rep.; Citroën; addressParis; 1963.
[22] Bézier P. Définition numérique des courbes et surfâces I. Automatisme 1966;XI:625–32.
[23] Bézier P. Définition numérique des courbes et surfâces II. Automatisme 1967;XII:17–21.
[24] de Boor C, Fix GJ. Spline approximation by quasiinterpolants. J Approx Theory 1973;8:19–45.
[25] Sorokina T, Zeilfelder F. An explicit quasi-interpolation scheme based on C^1 quartic splines on type-1 triangulations. Comput Aided Geom Des 2008;25:1–13.
[26] Lomelí HE, Meiss JD. Quadratic volume-preserving maps. Nonlinearity 1998;11:557–74.
[27] Du BS, Li MC, Malkin M. Topological horseshoes for Arneodo–Coullet–Tresser maps. Regul Chaotic Dyn 2006;11:181–90.
[28] Krauskopf B, Osinga H. Two-dimensional global manifolds of vector fields. Chaos 1999;9:768–74.
[29] Haro A, de la Llave R. Parameterization method for the computation of invariant tori and their whiskers in quasi-periodic maps: explorations and mechanisms for the breakdown of hyperbolicity. SIAM J Appl Dyn Syst 2007;6:142–207.
[30] Haro, A, de la Llave R. A parameterization method for the computation of invariant tori and their whiskers in quasi-periodic maps: numerical algorithms. Discrete Cont Dyn – B 2006;6:1261–300.
[31] Lomelí HE, Meiss JD. Heteroclinic intersections between invariant circles of volume-preserving maps. Nonlinearity 2003;16:1573.
[32] Wysham D, Meiss JD. Iterative techniques for computing the linearized manifolds of quasiperiodic tori. Chaos 2006;16:023129.