High-order Adaptive Method for Computing Two-dimensional Invariant Manifolds of Maps

Jacek K. Wróbel & Roy H. Goodman
E-mail: jacek.wrobel@njit.edu, roy.goodman@njit.edu
New Jersey Institute of Technology
Newark, NJ
September 1, 2011

Abstract

We present efficient and accurate numerical methods for computing invariant manifolds of maps which arise in the study of dynamical systems. In order to decrease the number of points needed to compute a given surface, we propose using higher-order approximation techniques developed for the field of computer-aided geometric design (CAGD). We use Bézier curves and triangles, fundamental objects in surface design, to create adaptive method. The method is based on tolerance conditions derived from properties of triangular Bézier patches. We develop and test the method for an ordinary parametric surface and adapt it to compute invariant manifolds of three-dimensional maps.

Keywords: invariant manifold, computer-aided geometric design, numerical algorithm.

1 Introduction

Iterated maps are ubiquitous in the study of dynamics, arising either as models of physical, biological, or economic systems themselves or as reductions of continuous-time dynamics, e.g., as Poincaré maps. Of fundamental importance in understanding these dynamical systems are invariant manifolds (stable or unstable) emanating from fixed points and periodic orbits. These manifolds act as barriers between different regions of the phase space and exert a significant influence on the dynamics through their topology. In general, such manifolds can not be expressed as closed-form parametric curves or surfaces, nor as the level sets of some function, and therefore must be approximated numerically.

We consider a few families of three-dimensional maps, for instance, volume-preserving maps that are useful in understanding the motion of passive tracers in fluids and magnetic field line configurations. These maps possess features that are well-understood in the two-dimensional case but not yet completely understood in three. These include such phenomena as transport, the breakup of heteroclinic connections, and the existence of invariant tori.

In recent work [8], we have developed high-order adaptive methods for computing one-dimensional invariant manifolds using Catmull-Rom splines, an interpolation scheme based on composite Bézier curves. This improved on previous methods, e.g. [11, 12] which tend
to either place too many points near, or else fail even to resolve, segments of the manifolds of high curvature. While these previous methods are based on piecewise-linear interpolation, our methods, based on tools from computer-aided geometric design (CAGD), are better able to follow the curvature of these manifolds, maintaining higher accuracy with fewer calls to the underlying map. The present paper describes a similar high-order method for two-dimensional manifolds.

This paper is organized as follows. Section 2 introduces the basic terminology and the objects for which we develop approximation algorithms. In Section 3, we describe existing methods: the parameterization method and the method due to Krauskopf and Osinga, and highlight their shortcomings. Section 4 gives a description of the CAGD tools used to construct the algorithm. Topics covered include Bézier curves and triangular Bézier patches, a quasi-interpolation methods based on cubic Bézier curves and quartic Bézier patches and their adaptive versions. In Section 5, we perform numerical tests of the adaptive quasi-interpolation based on quartic Bézier patches. Section 6 contains a description of the implementation of these tools in the context of computing two-dimensional invariant manifolds. In Section 7, we perform various numerical tests showing the convergence of the method. Section 8 contains a discussion summarizing the advantages of the algorithm, as well as remaining challenges.

2 Background

A discrete-time iterated map is a dynamical system $x_{j+1} = f(x_j)$ where, for simplicity, we assume $f : \mathbb{R}^n \to \mathbb{R}^n$ is diffeomorphism, and as smooth as we need. For the present work, we set $n = 3$. We assume that $f$ has a hyperbolic fixed point $x^*$, i.e., $f(x^*) = x^*$ such that the linearized matrix $Df(x^*)$ has three eigenvalues $\lambda_{s1}, \lambda_{s2}$ and $\lambda_u$ satisfying either $0 < |\lambda_{s1}|, |\lambda_{s2}| < 1 < |\lambda_u|$ or else three eigenvalues $\lambda_s$ and $\lambda_{u1}, \lambda_{u2}$ satisfying $0 < |\lambda_s| < 1 < |\lambda_{u1}|, |\lambda_{u2}|$. These assumptions assure the existence of two-dimensional stable and one-dimensional unstable manifolds or two-dimensional unstable and one-dimensional stable manifolds of $x^*$ in $\mathbb{R}^3$ which are invariant with respect to the map $f$; see Figure 1.

The stable manifold,

$$W^s(x^*) = \{ x | \exists \mu > 0, c > 0 \text{ s.t. } \forall k \geq 0, \| f^k(x) - x^* \| < ce^{-\mu k} \}$$

is defined as the set of points which approach $x^*$ at an exponential rate in forward iterates of the map. The manifold is tangent to the stable eigenspace of the linearized system at $x^*$ and its global extension can be derived by applying the inverse mapping $f^{-1}$ to a local piece.

The unstable manifold,

$$W^u(x^*) = \{ x | \exists \mu > 0, c > 0 \text{ s.t. } \forall k \leq 0, \| f^k(x) - x^* \| < ce^{-\mu k} \}$$

is defined as the set of points which approach $x^*$ at an exponential rate in backward iterates of the map. The assumptions on the eigenvalues ensure this convergence is exponential. The manifold is tangent to the unstable eigenspace of the linearized system at $x^*$ and its global extension can be derived by applying the forward mapping to a local piece.

Most numerical methods constructed to compute invariant manifolds use the same basic idea: the global structure (of individual branches for a 1D manifold) of the unstable manifold is found by repeatedly applying the mapping to an existing portion of the manifold, for example see [8]. The stable manifold can similarly be found by iterating $f^{-1}$, when it exists.
Figure 1: (Schematic) Right: Two and one-dimensional invariant manifolds of a hyperbolic fixed point and the annular portion of the manifold between curves $\Gamma$ and $\Gamma'$ on the right. Left: The annular domains for a parametrization method.

For a two-dimensional manifold, consider two simple, closed curves $\Gamma, \Gamma' \subset W^u$, which enclose the fixed point without intersecting each other. Let $W^u[\Gamma, \Gamma']$ denote the closed annular region of $W^u$ with boundary $\Gamma \cup \Gamma'$; see Figure 1. For any given simple, closed curve $\Gamma_0 \in W^u$ that encloses $x^*$, the set of its images $\{\Gamma_k = f^k(\Gamma_0) : k \in \mathbb{Z}\}$ partition $W^u$ into a family of finite annular regions disjoint except for their boundary curves. We refer to the closed connected component of $W^u$ between $\Gamma_k$ and $\Gamma_{k+1}$ as the $k$th primary annulus $U_k$, i.e.,

$$U_k = W^u[\Gamma_k, f(\Gamma_k)].$$

Therefore, in general we can write

$$U = \bigcup_{k=-\infty}^{\infty} U_k.$$

Our method is based on repeatedly computing an approximation to annulus $U_{k+1}$ from a previously computed approximation to annulus $U_k$. It thus requires that one of these annuli, referred to as the initial primary annulus $U_0$ be computed by some other method; see Section 3. Let

$$U_{\text{init}} = \bigcup_{k=-\infty}^{0} U_k.$$

We call this the “initial portion” of the manifold, and it does not deviate greatly from the unstable subspace.

The problem of computing a finite portion of an unstable manifold of a three-dimensional map can be reduced to that of simply computing a parametric surface in the following manner. Given an already computed annulus $U_k$ that has been endowed with parameterization $U_k = U_k(r, \theta)$, $(r, \theta) \in [r_k, r_{k+1}] \times [0, 2\pi]$, then the next segment is simply

$$U_{k+1}(r, \theta) = f(U_k(r, \theta)),$$

so that $U_{k+1}(r, \theta)$ is a parametric surface depending on the same parameters.

Clearly any numerical method employing this idea by using the same number of points to approximate the fundamental domain $U_0$ and all its forward images will have several disadvantages. First, the distribution of points along each piece is not controlled. A few iterations of the fundamental domain may produce very closely spaced points in some regions while large gaps between points occur elsewhere. Next, the shape of a primary annulus tends...
to change rapidly with the number of iterations of the initial segment. The number of points
required to resolve a later annulus is generally much greater than that required to resolve the
previous annuli; see Figure 2. These arguments suggest that it is better to place the points
adaptively along the manifold.

Figure 2: First, fifth, tenth and fifteenth primary annulus from a two-dimensional manifold
of a hyperbolic fixed point.

We focus our work on adaptive methods for computing invariant manifolds. These meth-
ods are able to adapt the distribution and number of points on each annulus. This avoids
large gaps between successive points on a segment without placing too many points on that
annulus. It can also avoid using far too many points in the smaller annuli while still resolving
the larger annuli. Our goal is to generate an approximation of the manifold which is smoothly
resolved with a minimum number of points.

3 Existing Methods

Two existing, widely cited methods suited for computing two-dimensional manifolds of maps
are the parametrization method and the method by Krauskopf and Osinga.

The Parameterization Method is the basis for the recent numerical computations of
heteroclinic manifolds of maps by Mireles-James and Lomelí [14, 15]. In a neighborhood
of the fixed point, the manifolds $W^u$ and $W^s$ are represented as power series, with the
coefficients determined by the manifold’s invariance under the map $f$. This construction has
been used for both rigorous analytical calculations as well as numerical approximation; see,
e.g., [1, 2, 3, 7].

This method provides a truncated form of an explicit parametric representation of the
manifold. If $f$ is analytic, the power series has infinite radius of convergence, but, in practice,
due to roundoff error, the numerical radius of convergence may be quite small. Even though
the method is very accurate near the origin a complementary method is needed in order to
compute larger portion of the manifolds. We use the parameterization method to compute
an approximation to $U_0$, and in fact to calculate all of $U_{init}$, defined in the previous section.
Note that the parameterization does not prescribe a way to choose which points on the
manifold to compute, nor how to interpolate between them, which is the principal aim of our
method.

The Krauskopf-Osinga Method presented in [12] is a method for computing the two-
dimensional unstable manifolds of a hyperbolic fixed point of an iterated map in three di-


dimensions. Authors drop the idea of iterating a fundamental domain and instead they “grow” the manifold at the same rate in each direction. The algorithm computes one-dimensional intersection curve of the unstable manifold with a finite number of two-dimensional leaves of a linear foliation.

Their attempt to grow the manifold uniformly is intended to alleviate the exponentially anisotropic growth if $1 < |\lambda_1| < |\lambda_2|$. It is designed to deal with the problem that fundamental annuli grow at different rates dependent on direction. However, it has major shortcomings which we describe below. Two aspects of this method can cause it to fail.

1. Its dependence on the foliation breaks down if the manifold is ever tangent to a leaf.

2. If the map has rotation, points far from $x^*$ may map to points nearer to $x^*$. Points on a given ring that we intend to compute may depend on points outside that ring which are not yet computed.

Krauskopf and Osinga acknowledge point 1, but we were unable to find any recognition of point 2 in their work.

The method should work when the fixed point has eigenvalues with largely differing modulus, as well as in the case of complex conjugate eigenvalues. However, our experiments with this method show that this is not true in general and the method can fail or produce a manifold of poor quality. For example, the case, in point 2 above, is a common behavior of nonlinear maps and it can occur not only in the neighborhood of the fixed point. The behavior is easily seen in maps with a rotation component. For instance, this is one of the reasons why the method fails for maps on which we test our proposed method in Section 7.

Moreover, their method is based on an interpolation by piecewise-planar triangular patches which thus has only quadratic accuracy. These interpolation errors are then magnified exponentially by the stretching action of the map, as we saw in our computation of one-dimensional manifolds [8]. Using higher-order interpolation should significantly improve computations.

4 Geometric Modeling Tools

Motivated by (1), we delay considering the problem of computing a two-dimensional manifold and instead focus on the simpler problem of approximating a parametric surface. Here we introduce a few tools from CAGD that we use to construct the numerical method. A good introduction to this material is given by Farin [6].

4.1 Bézier Curves

Bézier curves provide a convenient form for computations involving polynomial curves. A degree-$n$ Bézier curve defined in terms of $n+1$ control points $\{p_0, p_1, \ldots, p_n\}$, $n = 3$ being the most common choice, defined over the parameter interval $t \in [0, 1]$. It interpolates the first and last points $p_0$ and $p_n$ at parameter values 0 and 1, respectively, while the other points merely influence the curve’s shape. An important property of these curves is the convex hull property, namely that the curve lies inside the convex hull of the control points.

This fundamental object in geometric modeling can be constructed using Bernstein polynomials defined as

$$B_k^n(t) = \binom{n}{k} t^k (1-t)^{n-k}; \quad k = 0, \ldots, n; \quad 0 \leq t \leq 1. \quad (2)$$
A Bézier curve has parametric form given by the convex affine combination (i.e., with positive weights summing to one) of \( n + 1 \) control points \( p_0, \ldots, p_n \):

\[
\gamma(t) = \sum_{k=0}^{n} p_k B_k^n(t). \tag{3} \]

The polygon with consecutive vertices \( p_0, \ldots, p_n \) is the control polygon of the curve. An example of a cubic Bézier curve given by control points \( p_0, p_1, p_0^3 \), and \( p_0^3 \) is shown in Figure 3. Note that in general the boundary of the convex hull and the control polygon are not the same object.

Bézier curves are invariant under affine transformation of the independent variable \( t \). Setting

\[
t = (u - a) / (b - a), \tag{4} \]

defines a curve \( \gamma(u) = \gamma(t(u)) \) over the interval \([a, b] \) that is identical to \( \gamma(t) \) parameterized over \([0, 1] \).

Finally, by Equation (3), the tangent vectors to the curve at \( p_0 \) and \( p_n \) are

\[
T_0 = n(p_1 - p_0) \quad \text{and} \quad T_n = n(p_n - p_{n-1}), \tag{5} \]

respectively—this formula is modified slightly if the parameterization in Equation (4) is used. This can be observed in Figure 3.

Bézier curves are widely used in CAGD because, in addition to the above properties, there exist efficient algorithms for evaluating them (forward-differencing and the de Casteljau algorithm), and for performing other calculations such as finding their intersections \([16, 17] \) or their arc length \([9] \).

**The de Casteljau Algorithm** is a recursive method to evaluate a Bézier curve \( P_{[a,b]} \) at a parameter value \( t \in (a, b) \). The geometric interpretation of this algorithm is shown in Figure 3. The control points of \( P_{[a,b]} \) are labeled as \( p_0, p_1, p_2, \) and \( p_3 \). \( P(t) \) is calculated by iterated linear interpolation. At step \( i + 1 \), each line segment in the trellis is split in the ratio \((t - a)/(b - t)\), i.e.,

\[
p_j^{i+1} = ((t - a)p_j^i + (b - t)p_{j+1}^i)/(b - a) \tag{6} \]

and \( p_3 \), the final point computed is the value of \( \gamma(t) \) as defined in Equation (3).

The de Casteljau algorithm subdivides a Bézier curve \( P_{[a,b]} \) into two shorter Bézier curves \( P_{[a,t]} \) and \( P_{[t,b]} \) whose union is \( P_{[a,b]} \) with control polygons \( \{p_0, p_1, p_0^3\} \) and \( \{p_3, p_2, p_2^3, p_3\} \).

**Composite Bézier Curves** are constructed from multiple Bézier curves pieced together in order to generate shapes that are too complex for a single low-degree curve. In concatenating Bézier curves, we need to control the smoothness of the composite curve. Let \( p_0, \ldots, p_3 \) and \( p_3, \ldots, p_6 \) be the Bézier control points of two cubic curve segments \( P_{[a,c]} \) and \( Q_{[c,b]} \). Since they share the point \( p_3 \), their union clearly forms a continuous curve. The composite curve may, however, form a corner. To ensure that the two pieces meet smoothly, more care is called for.

Two adjacent curve segments \( P \) and \( Q \) are said to be \( C^k \)-continuous at their common end point \( c \) (or, to have \( k \)th order parametric continuity) if

\[
P(c) = Q(c), P'(c) = Q'(c), \ldots, P^{(k)}(c) = Q^{(k)}(c). \]


Equation (5) demonstrates that the composite curve is \( C^1 \) if and only if
\[
\frac{(p_3 - p_2)}{(c - a)} = \frac{(p_4 - p_3)}{(b - c)}.
\] (7) \{eq:c0continuity\}
Similar conditions exist for \( C^2 \) and higher continuity.

### 4.2 Triangular Bézier Patches

Here we consider the most natural generalization of the above ideas to two-dimensional surfaces. Consider a triangle \( T \) with vertices \( p_1, p_2, p_3 \) (i.e., 3 non-collinear points) and a fourth point \( p \), all in two-dimensional Euclidean space \( \mathbb{E}^2 \). Then \( p \) may be written uniquely as a barycentric combination of these three vertices, i.e.,
\[
p = u p_1 + v p_2 + w p_3,
\] (8) \{eq:barycentric\}
where \( u, v \) and \( w \) must satisfy \( u + v + w = 1 \). The coefficients \( U := (u, v, w) \) are called the barycentric coordinates of \( p \) with respect to the triangle \( T \). If the coordinates satisfy \( u, v, w > 0 \), then the point \( p \) lies inside the triangle \( T \). A point outside the triangle has at least one negative coordinate.

**The Bernstein polynomial** \( B^n_i \) of degree \( n \) over a triangle, in terms of the barycentric coordinates \( U := (u, v, w) \), is defined by
\[
B^n_i(U) = \binom{n}{i} u^i v^j w^k; \quad |i| = n; i = (i, j, k)^T;
\]
where the trinomial coefficients are defined as \( \binom{n}{i} = \frac{n!}{i! j! k!} \). The Bernstein polynomials satisfy the recursion relation
\[
B^n_{i,j,k} = u B^n_{i-1,j,k} + v B^n_{i,j-1,k} + w B^n_{i,j,k-1},
\] (9) \{eq:recursionBern\}
with the convention that \( B^n_{i,j,k} = 0 \) if any of its subscripts is negative.
**Triangular Bézier patches** are polynomials of the form

\[ b^n(U) = \sum_{|i|=n} b_i B^n_i(U). \]  

(10) \{eq:Bernstein-BezierPatch\}

This sum has \( p = \frac{1}{2}(n+1)(n+2) \) terms. The \( b_i \) are called the Bézier ordinates (control points) of \( b^n \). The piecewise planar interpolant of the points \((i/n, b_i)\) is the Bézier control net of \( b^n \). The Bézier patch \( b^n \) is located inside the convex hull of its defining ordinates, the triangulated surface defined by the control net. The boundary curves of the patch \( b^n \) are (univariate) Bézier polynomials and their control polygons are the boundaries of the control net.

![Cubic Bézier patch](image)

Figure 4: Cubic Bézier patch (red) with the control net (blue) and the domain triangle (left). \{fig:CubicBezierTriangle\}

Triangular Bézier patches can be combined to build piecewise-defined complex surfaces, where each patch corresponds to a triangle of a tessellation of the \((s, t)\) parameter plane. The main problem in constructing such surfaces is maintaining their continuity. This is harder than for Bézier curves because only the three corner points of a Bézier patch are interpolated, while continuity conditions must hold along the whole boundary. When the surface is given by patches of different resolution, even \( C^0 \) continuity may be problematic. In order to maintain \( C^1 \) continuity between two patches, their control nets have to satisfy algebraic conditions similar to Equation (7).

**The de Casteljau Algorithm for Triangular Patches** is a direct generalization of the corresponding algorithm for curves. As it does for curves, the de Casteljau algorithm uses repeated linear interpolation to evaluate Bézier triangles.

The recursion formula (9) defines the de Casteljau algorithm for evaluating a Bézier triangle \( b^n \) at barycentric coordinate \( U := (u, v, w) \). Recursively defining auxiliary points

\[ b'_{i,j,k} = u b_{i-1,j,k} + v b_{i-1,j-1,k} + w b_{i,j-1,k-1}, \]

where \( i + j + k = n - l \) \((i, j, k \geq 0)\), and \( b^0_{i,j,k} = b_{i,j,k} \), then \( b^n(U) = b^n_{0,0,0}(u, v, w) \). A scheme of the algorithm is shown in the left part of Figure 5. It is a numerically stable and efficient method for evaluating a Bézier triangle, both faster and more accurate than using formula (10).
Subdividing Triangular Bézier Patches. The de Casteljau algorithm for triangular patches subdivides a patch into three subtriangles each with two vertices along the edges of the original patch, a third vertex at the evaluated point, and control points at the points of the auxiliary net, as in the right image of Figure 5. By subdividing the patch at a point on the triangle’s edge, the subdivision produced is degenerate and instead results in two subtriangles. Several applications of this 2-to-1 split can be used to split the triangle into four congruent subtriangles $T_0 - T_3$ as shown in Figure 6.

4.3 Quasi-Interpolation Schemes

Quasi-interpolation is a method for constructing an approximation from data that has attractive smoothness properties and local control, at the expense of sacrificing exact interpolation of the data. We consider the quasi-interpolation operator constructed by spline functions and, in particular, discrete quasi-interpolation (dQI). This Quasi-Interpolant (QI) of a function $f(x)$ can be defined as an operator of the form

$$Qf = \sum_{j \in J} \mu_j(f)B_j; \text{ where } \mu_j(f) = \sum_{k=-q}^{q} w_{j,k}f(x_{j+k})$$

where $\{B_j : j \in J\}$ is a basis of some space of splines of degree $m \leq 2q + 1$. Let $P_m$ be the space of polynomials of degree at most $m$. Quasi-interpolation is defined by imposing the condition that $Q$ is exact for functions $f \in P_m$, i.e., $Qp = p$ for all $p \in P_m$. As a consequence of this property, the approximation order is formally $O(h^{m+1})$ on smooth functions, where $h$ is the maximum steplength of the partition.

Quasi-Interpolation based on Cubic Bézier Curves. The one-dimensional (curve-drawing) method presented here is meant to describe the basic ideas needed for the two-
dimensional (surface-drawing) methods introduced in what follows, which we will not describe in detail, but only point out how they differ.

For \( I = [a, b] \), we let \( \mathcal{S}_3(I, X_N) \) be the space of piecewise cubic polynomials on the uniform partition \( X_N = \{x_i = a + ih, i = 0, \ldots, N\} \) with grid size \( h = \frac{b-a}{N} \), where \( N \) is multiple of 3. Define the set of disjoint subintervals \( I_i = [x_i, x_{i+3}] \) for \( i = 0, 3, 6, \ldots, N-3 \). On each subinterval, as the basis of this space, we use the cubic Bernstein polynomials,

\[
\mathcal{B}_k^3(x) \equiv B_k^3(t(x)), \ k = 0, \ldots, 3,
\]

where \( B_k^3(t(x)) \) is a cubic Bézier curve. Note that the method requires four buffer points outside the interval \( I \) for each composite curve interpolates every third point of data and approximates between them. For \( f \in C(I) \), quasi-interpolating spline \( Qf \) is \( C^1 \)-continuous and for each \( p \in \mathcal{P}_3, Qp = p \).

**Adaptive Quasi-Interpolating Cubic Splines.** The quasi-interpolation scheme based on cubic Bézier curves is a \( C^1 \)-continuous, completely local, and fourth-order. We now describe how to add adaptivity, necessary for efficiently approximating a curve whose curvature and complexity vary greatly along its length. The method constructed uses regularly sampled curve based on the cubic spline scheme introduced above. This adaptive quasi-interpolation construction provides local control; it offers flexibility for local resolution adaptation while maintaining \( C^1 \)-continuity.

Given a set of regularly gridded data points \( f_i^l \) over \( x_i^0 \) for \( i = -2, \ldots, N+2 \), let \( s_0 = Qf \). Analogously, we define a sequence of locally supported splines \( s_l \) on finer grids, \( x_i^l = a + i2^{-l}h \) for \( i = -2, \ldots, 2^l N + 2 \) and \( l = 1, 2, \ldots \). In this hierarchy, every subinterval in the domain of \( s_l \) overlaps with two subintervals of \( s_{l+1} \).

The algorithm starts by calculating the spline \( s_0 \) at the coarsest base level, where the finer representations are needed only for local evaluation, on demand. As the condition for refining the segment \( s_l|_{[x_i^l, x_{i+3}^l]} \), the method uses the maximum error of the quasi-interpolant at all the two non-interpolated points over \( x_{i+1}^l \) and \( x_{i+2}^l \), i.e.

\[
\varepsilon([x_i^l, x_{i+3}^l]) := \max_{k \in \{1, 2\}} \{||f_{i+k}^l - s_l(x_{i+k}^l)||\}.
\]

All segments with an approximation error exceeding a prescribed tolerance are then split.
Algorithm 1 The algorithm can be defined as follows:

- For spline \( s_l \) compute the approximation error at all data points \( f^l_i, i \neq 3j \).
- Recursively identify the set of segments in \( s_l \) to be split, using a refinement condition based on Equation (13).
- On these segments, use the approximation error as the new data values by locally evaluating the spline and the underlying function on the next finer grid, i.e., \( f^{l+1}_i := (f^l - s_l)|_{x^{l+1}_i} \) and all data points \( f^{l+1}_i \) located outside this area set to zero.
- Compute the next finer approximation level \( s_{l+1} \) for the region influenced by the nonzero \( f^{l+1}_i \); see Figure 7.

The algorithm repeats until the approximation error decreases below a prescribed tolerance. This strategy results in a coarse approximation \( s_0 \) and a sequence of locally supported correction functions \( s_1, \ldots, s_L \), each of which vanishes with \( C^1 \) continuity at the boundaries of its support; see Figure 7. Hence, the final representation

\[
s_{\text{adaptive}} := \sum_{l=0}^{L} s_l \tag{14}
\]

is \( C^1 \)-continuous and can be locally evaluated.

Figure 7: A step of the adaptive method; (a) coarse approximation \( s_l \) of given data points (red dots); the middle dashed segment requires correction. (b) locally supported correction functions \( s_{l+1} \) on a finer grid.

Quasi-Interpolation based on Quartic Bézier Patches was introduced by Sorokina and Zeilfelder in [18]. This \( C^1 \) quasi-interpolant is defined over a uniform lattice of points in parameter space, connected via a type-1 triangulation—a square lattice with one diagonal per box, all aligned in the same direction, splitting each square into a top and bottom triangles \( T \) and \( B \) as in figure 8. The method uses piecewise Bézier patches for its basis, and sets the Bézier coefficients in a manner analogous to equation (12).

The method is analogous to the quasi-interpolation based on cubic Bézier curves introduced above. A method based on cubic triangular patches does not contain enough free parameters in order satisfy the algebraic conditions for (i) \( C^1 \)-continuity between patches and (ii) cubic exactness: \( Qp = p, \forall p \in \mathcal{P}_3 \), so the method uses quartic patches, requiring
fifteen control points per patch, but uses just ten data points per patch, the same number of points in the cubic stencil.

We consider the space $S_1^4$ of quartic $C^1$-splines on a type-1 triangulation $\triangle$, see Figure 8 defined by

\[ S_1^4(\triangle) = \{ s \in C^1(\mathbb{R}^2) : s|_T \in \mathcal{P}_4, \forall T \in \triangle \} \]

where $\mathcal{P}_4 = \text{span}\{x^i y^j : i + j = 0, \ldots, 4\}$ denotes the space of bivariate polynomials of total degree four. We use the piecewise Bernstein-Bézier representation of the splines, i.e., for each spline $s \in S_1^4(\triangle)$, the polynomial patch $s|_T \in \mathcal{P}_4$ on a triangle $T = P_0 P_1 P_2 \in \triangle$ is given by

\[ s|_T = \sum_{i+j+k=4} c_{i,j,k} B^4_{i,j,k}(u,v,w), \]  

(15) \{eq:quartic_BB\}

where $B^4_{i,j,k}$ are the quartic Bernstein polynomials defined by (10) associated with $T$ and $u, v, w$ are barycentric coordinates with respect to its vertices. We associate the Bézier ordinates $c_{i,j,k}$ relative to $T$ with the domain points $\xi_{i,j,k} := (iP_0 + jP_1 + kP_2)/4$ in $T$. For the quasi-interpolation scheme, we also need the domain points associated with a cubic polynomial on $T$: $\eta_{i,j,k} := (iP_0 + jP_1 + kP_2)/3$.

Sorokina and Zeilfelder in [18] determine the Bézier coefficients of the resulting quartic continuous spline $Qf$ on $\triangle$. For each triangle $T$ in $\triangle$, the polynomial piece $Qf|_T \in \mathcal{P}_4$ is obtained from data points lying on $T$ and its six nearest neighbors with the opposite orientation; see Figure 8. Moreover, their construction of the Bézier ordinates of $Qf$ ensures $C^1$ continuity. As the method is exact for $f \in \mathcal{P}_3$, there exists a constant $K$ such that for each triangle $T$ in $\triangle$ for every $f \in C^4(\mathbb{R}^2)$,

\[ ||Qf - f||_T \leq K ||D^4 f||_{Th^4}. \]  

(16) \{eq:quasi_accuracy\}

Figure 8: Left: filled dots show domain points required to construct control points of a single Bézier triangle; Right: open dots show the resulting control points.

Adaptive Quasi-Interpolating Quartic Splines. In [10], Hering-Bertram et al. present an adaptive quasi-interpolation quartic spline construction based on the above quartic spline scheme, in a manner similar to the one dimensional Bézier interpolant we presented earlier. The construction is $C^1$ can be applied to both parametric and scalar surfaces which is not
always true for interpolation methods in higher dimensions. The mesh is refined in a pseudo-regular manner a local type-1 triangulation. At $T$-nodes, a point where two meshes of different refinement meet, it is difficult to achieve continuity, much less smoothness; see Figure 9. In a similar manner to our 1D interpolant above, this means that corrections must be made on a region slightly larger than where the errors occur.

Figure 9: Bézier patches of different resolution, one level difference.

Given a set of regularly gridded data points $f_{i,j}^0$ over cubic domain points $\eta_{i,j}^0$ for $i = 0, \ldots, N_x$ and $j = 0, \ldots, N_y$, let $s_0$ denote the spline surface approximation on the coarsest grid. Analogously, we define a sequence of locally supported splines $s_l$ for $i = 0, \ldots, 2^l N_x$ and $j = 0, \ldots, 2^l N_y$. In this hierarchy, every triangular patch $T$ in the domain of $s_l$ overlaps with four domain patches of $s_{l+1}$; as in Figure 6.

The algorithm starts by calculating the spline approximation $s_0$ at a coarsest level and follows the same strategy as Algorithm 1, above. In this case the method uses as its refinement condition the maximum error of $s_l|_T$ at all sites on the mesh $\eta_{i,j}^l$ inside $T$. Let

$$\varepsilon_{ij} = f_{i,j}^l - s_l(\eta_{i,j}^l)$$

and

$$\varepsilon(T) := \max_{\eta_{i,j}^l \in T} \{|\varepsilon_{i,j}|\}. \quad (17)$$

Triangles whose approximation error exceeds a prescribed tolerance, as well as their nearest neighbors require correction.

The final $C^1$-continuous approximating surface is given as a sum of the coarse spline approximation $s_0$ and locally supported correction functions $s_1, \ldots, s_L$, each of which vanishes with $C^1$ continuity at the boundaries of its support. For more details of above method see [10].

5 Numerical Implementation and Test of the Adaptive Scheme

In this section we describe our implementation of the Adaptive Quasi-Interpolating scheme described in Subsection 4.3. The goal is to adaptively approximate parametric surfaces of varying geometric complexity, where the local resolution is driven by the approximation error. The construction of high-quality spline surfaces is enhanced by the flexibility of adaptive pseudo-regular triangular meshes.

The first step of the method is to generate the Quartic Quasi-Interpolation approximation surface. Hence, for our purposes we define the type-1 triangulation on a rectangular parametric domain. Remember that the method constructs approximating surface over any triangle $T$ based on data from $T$’s neighbors; see Figure 8. Thus if $T$ is on the boundary, it needs data from an additional strip. If the parametric region is rectangular and bounded, it will require buffer points on all four sides. If it is doubly periodic, no buffers are needed. In the present case of annular region, buffers are required in the radial ($r$) direction but not the periodic $\theta$ direction.
For the general and annular surface some additional work has to be done to generate
adaptable corrections close the boundary. The approximation surface is created only over
the inner triangles, so that for any correction surface, over the same domain, we need to
maintain one external row of domain triangles (three extra rows of data). Forming the
correction over a single domain triangle requires error data $\varepsilon_{i,j}$ from equation (17) over the
extended triangulation. If no data exists, we need to estimate it. The simplest estimate, that
the error $\varepsilon_{i,j}$ vanishes outside of the original domain, fails: the calculated error persists close
to the boundary and does not decay when the further adaptation steps are applied. In some
cases, this error even propagates into the domain.

Instead, we construct approximate error data $\varepsilon_{i,j}$ at points that extend the data from the
domain needing refinement into the neighboring rows of triangles. We assume that the data
and the derivative the interior boundary of the extended domain match those of the original
domain, and at the exterior edge, we assume the data and its derivative vanishes vanish; see Figure 10. We use a cubic Hermite polynomial to satisfy these four conditions, using
formula (5) for the derivatives. This approach works well; the error close to the boundary
decays with the same rate as inside of the domain.

This extrapolation is necessary only when correction is needed to refine a triangle close to
the boundary of the domain. At interior points these data are still needed and have already
been computed during the test for refinement.

The adaptive method produces a $C^1$ approximation surface using Bézier triangles of
different sizes. To render the surface, each Bézier patch is triangulated by small planar
triangle. To avoid the appearance of cracks in the surface—see figure 9 each triangle has to
be perfectly matched with its neighbors. Thus, the resolution of the planar triangulation is
determined by the mesh of the finest Bézier patch.

We implement and test methods described in Subsections 4.3 on surfaces with explicit
parameterization and periodicity in 0, 1, or 2 directions. This allows us to understand
behavior of the implemented method and to gain a bit more control over the testing processes.
In Figure 11a we can easily notice at least three levels of the adaptive refinement. The flat
region close to the boundary of the disk has a lower resolution, whereas the middle, curly
and folded region has the highest. In examples, Figure 11b and 11c, we can easily notice two
levels of the refinement (really, there are two in Figure 11b and three in 11c). This test shows
that the method itself and its implementation perform well at refining surfaces where needed
without creating any pinholes or artifacts. More details, such as formulas for the parametric
model surfaces and numerical convergence studies, are in [19].
6 Details of the Numerical Implementation: Two-dimensional Invariant Manifold Calculation

6.1 Parameterization

Assuming the initial primary annulus $U_0 = U_0(r, \theta)$ is parameterized as in Section 3, let $U_1(r, \theta) = f(U_0(r, \theta))$ for some $r_0 \leq r \leq r_1$ and $0 \leq \theta \leq 2\pi$. Then considering this portion of the manifold as a topologically annular surface centered at the fixed point it is natural to use polar coordinates.

The choice of the initial parameterization is suggested by the problem of computing unstable manifolds. Consider a linear map of the form $f(x) = Ax$. Clearly this has a hyperbolic fixed point at the origin with one stable and two unstable directions. Let $\lambda_1$ and $\lambda_2$ be its unstable eigenvalues and $\vec{v}_1, \vec{v}_2$ be associated eigenvectors, respectively, i.e., the unstable invariant subspace $E^u = \text{span}\{\vec{v}_1, \vec{v}_2\}$. The initial primary annulus of the unstable manifold can be written as $U_0 = W^u[\Gamma_0, \Gamma_1]$, where

$$\Gamma_0 = r_0\vec{v}_1 \cos \theta + r_0\vec{v}_2 \sin \theta$$
$$\Gamma_1 = A\Gamma_0 = r_0|\lambda_1|\vec{v}_1 \cos(\theta + \theta_0) + r_0|\lambda_2|\vec{v}_2 \sin(\theta + \theta_0)$$

with $r_0 = \text{const}$ and $\theta_0 = \text{arg}(\lambda_1/\lambda_2)$. The annulus creates the elliptical ring which can be written in polar coordinates as

$$U_0(t, \theta) = (a(t)\vec{v}_1 \cos \theta + b(t)\vec{v}_2 \sin \theta)$$

where $a(t) = r_0|\lambda_1|^t$ and $b(t) = r_0|\lambda_2|^t$ for $\theta \in [0, 2\pi]$ and $t \in [0, 1]$; see Figure 12.

For a complex conjugate pair of eigenvalues $\lambda_1 = \bar{\lambda}_2$, the relative growth of an annulus under the linear map is isotropic since $a(t) = b(t)$. Additionally $\theta_0 \neq 0$ and rotation about this angle is associated with the map.

This example suggests a way to parameterize the initial primary annulus $U_0$ for a general map $f$. Near the fixed point (which we can assume is $x^* = 0$), the map $f(x)$ is approximately given by $x_{n+1} = Df(0)x_n$, so we can approximate $U_0$ by an appropriate annulus of the unstable eigenspace using the above parameterization, up to a correction due to the nonlinearity of $f$. If the initial primary annulus is parameterized by this algorithm then by
mathematical induction, this parameterization can be used for all successive annuli $U_n$; we define $U_{n+1}(r, \theta) = f(U_n(r-1, \theta-\theta_0))$.

$$v_1/c^{1/2}$$

$$v_2/c^{1/2}$$

$$v_3/c^{1/2}$$

$$v_1/c^{1/2}$$

$$v_2/c^{1/2}$$

$$v_3/c^{1/2}$$

(a) \hspace{1cm} (b)

![Figure 12: Union of primary annuli with the initial primary annulus $U_0$ shaded, (a) for a real pair of eigenvalues with slightly anisotropic growth, (b) for a complex pair of eigenvalues with isotropic growth.](fig:Eigenellipse)

6.2 Notation

Let $U_n$ be the numerical approximation to the $n$th true annulus $U_n$. This approximation is defined adaptively as the composite quartic Bézier surface in two steps. Given is the initial set of data points defined over a uniform grid in parameter space

$$X_n^0 = \{x_k^n\}_{k=0}^N$$

and the underlying type-1 triangulation. On each annulus the initial type-1 triangular mesh is chosen identically, consisting of three columns of triangles per annulus, where the first and the last column are the buffer triangles for the quasi-interpolation method. First, we define the operator $A$ that adaptively, based on a specified tolerance $\text{tol}$, adds points to the initial set generating a pseudo-regular parameter grid $T_n^A$ and the associated points $X_n^A$, i.e.,

$$(T_n^A, X_n^A) = A(T_n^0, X_n^0, \text{tol}).$$

Next, we define an operator $QI$ that constructs the composite quartic Bézier surface over the pseudo-regular parametric grid $T_n^A$ and the corresponding set of points $X_n^A$, i.e.,

$$U_n = QI(T_n^A, X_n^A).$$

A composite of these two operators we denote $A$-$QI$. Additionally, we introduce two pieces of information

$$T_{n+1}^0 = T_n^0 + (1, \theta_0) = \{(t_k^n + 1, \theta_k^n + \theta_0)\}_{k=0}^N$$

and $X_{n+1}^0 = f(X_n^0) = f^n(X_0^0)$. \hspace{6cm} \{eq:notation_3d\}

Thus we may construct the approximate invariant manifold using an inductive procedure. First, we use the refinement operator $A$, $(T_0^A, X_0^A) = A(T_0^0, X_0^0, \text{tol})$, and the quasi-interpolation operator $QI$ to construct the initial primary annulus $U_0 = QI(T_0^A, X_0^A)$. Next, for given $U_n$, we use the operator $A$ to find $T_{n+1}^A$ and $X_{n+1}^A$ and the operator $QI$ to construct $U_{n+1} = QI(T_{n+1}^A, X_{n+1}^A)$. 
6.3 The Initial Primary Annulus

In order to initialize any adaptive method for computing an invariant manifold, one first has to determine the initial primary annulus $U_0$, an approximation to the true annulus $U_0$, as is described in Section 2.

Crucially, this method requires the initial primary annulus $U_0$ in a parametric form. Hence, the parameterization method described in Section 3 establishes the fundamentals of the method. The stable or unstable manifolds of the map $f$ near a fixed point $x^*$ is represented by a parametric function $P(r, \theta)$. In practice, the initial annulus requires $r$ small, as per the discussion in Section 3. For $U_0$ to be approximated with the necessary accuracy, the initial annulus must be taken close to the fixed point, where the truncated series of the higher-order approximation is valid.

Choosing $\gamma_0$ as described in Subsection 6.1, the first ring around the origin in the parametric space, the closed curve $\Gamma_0$ on the unstable manifold (the interior boundary curve of the initial annulus $U_0$) is determined by the expansion $P$ as $\Gamma_0 = P(\gamma_0)$. Here, $\gamma_i$ is a closed curve enclosing the origin in $(r, \theta)$-parameter space, whereas $\Gamma_i$ its image in physical space, i.e., $\Gamma_i = P(\gamma_i)$; see Figure 1. The ring $\gamma_1$ in the parametric space is determined by the choice of the ring $\gamma_0$ by Equation 13 and its image $\Gamma_1 = P(\gamma_1)$ defines the other boundary of the initial annulus. Then the whole initial primary annulus $U_0 = W[\Gamma_0, \Gamma_1]$ is determined by the annulus in parametric space and by the expansion $P$. Note, that the high accuracy of the expansion $P$ gives $\Gamma_1 = f(\Gamma_0)$. The annulus between $\gamma_0$ and $\gamma_1$ is defined for parameters $(r, \theta) \in [r_0, |\lambda| r_0] \times [0, 2\pi]$ or equivalently $(t, \theta) \in [0, 1] \times [0, 2\pi]$; see Subsection 6.1.

In order to construct an accurate approximation $U_0$ we sample our parametric domain $(t, \theta)$ on the uniform grid with underlying type-1 triangulation mesh; see Figure 8. This grid defines the sets $T_0^0$ and $\lambda_0^0 = P(T_0^0)$. The composite of the operators $A$ and $QI$, $A-QI(T_0^0, \lambda_0^0, tol)$, generates the approximation $U_0$ as a composite Bézier surface.

In the case of complex eigenvalues $\lambda_1 = \bar{\lambda}_2$, a rotation about the angle $\theta_0 = \arg(\lambda_1/\lambda_2)$ is associated with the map. Any point from one boundary curve $\Gamma_0$ given by $P(r_0, \theta)$ for $\theta \in [0, 2\pi]$ is associated with its image point (under the map $f$) at the other boundary curve $\Gamma_1$ given by $P(r_1, \theta)$ for $\theta \in [\theta_0, \theta_0 + 2\pi]$. We want to construct a continuous surface, so we need to match these points in the parametric space. Unfortunately, if we try to directly match a point with its image under the map $f$, a straightforward type-1 triangulation on the $(r, \theta)$ grid will produce long narrow triangles in physical space. Further, a triangulation on the rectangular grid it will cause inconsistency between triangular patches on the consecutive annuli; see Figure 13a. In order to avoid this problem we need to carefully choose the parameter values for the initial primary annulus $U_0$.

Let $m$ be the closest integer to $2\pi/\theta_0$, i.e., $m = \text{round}(2\pi/\theta_0)$. Then after $m$ applications of the map, the images of a point make approximately one rotation around the fixed point. Were we to fix an orthogonal grid in $(\theta, r)$ parameter space for the annulus $U_0$, then the images of the lines of constant $\theta$ in this parameterization would themselves be far from radial. Thus we parameterize $U_0$ in a slightly different way. Let

$$\omega_0 = \theta_0 - \frac{2\pi}{m}.$$ 

Next, sample the expansion $P(r, \theta)$ on the sheared grid, i.e., $\theta \in [0, 2\pi]$ for $\gamma_0$ and $\theta \in [\omega_0, \omega_0 + 2\pi]$ for $\gamma_1$; see Figure 13b. This parameterizes the surface such that the triangulation is well-adapted for its geometry; Figure 13.
Figure 13: Schematic settings for the initial primary annulus $U_0$, (a) five primary annuli in a physical space: dashed line—the edges over a sheared grid with $\theta_0$, solid bars—edges over a rectangular grid, (b) the triangulation of the sheared grid in the parametric space, the buffer dash triangles in the first and the last columns, (c) four primary annuli in a physical space, rotation of a primary annulus about $\theta_0$ under the map $f$ with a multiplicity $m = 3$ and a correction $\omega_0$; $N_T = 12$ then every mapping the rotation about four triangles.

Additionally, we need to make sure that the matching is valid for each triangle of the triangulation, thus the number of triangles in each column also depends on $\theta_0$ and must be a multiple of $m$. For each application of the map, the annular mesh is rotated about $N_T/m$ triangles, where $N_T$ is the number of triangles in a column.

Note that the proper choice of the parameter values is crucial only for the initial primary annulus $U_0$ and it guaranties a continuous strip of Bézier triangles on the whole manifold. During the process of creating further annuli, the algorithm uses only the parameters $\mathcal{T}$ in their barycentric coordinates with respect to the triangular mesh, so the phase term is irrelevant.

### 6.4 Resolving a Primary Annulus

We restrict our attention to the specific problem of finding the unknown primary annulus $U_{n+1}$ which is the approximate image under the map $f$ of an already-resolved primary annulus $U_n = A-QI(\mathcal{T}_n^0, \mathcal{A}_n^0, \text{tol})$, i.e., $U_{n+1} \approx f(U_n)$. At each primary annulus we start with a mesh $(\mathcal{T}_n^0, \mathcal{A}_n^0)$ identical to that defined on the initial primary annulus. We apply operators $A$ and $QI$ to produce a resolved surface $U_{n+1}$ that approximates $f(U_n)$ with sufficient accuracy. The computed portion of the manifold is then given by

$$U \approx \bigcup_{n=0}^{N} U_n,$$

which is piecewise $C^1$, but not yet continuous.
6.5 Joint Patching

The proposed method is quite capable of resolving a single primary annulus of a manifold given a previously completed annulus. The annuli $U_n$ and $U_{n+1}$ may be individually resolved but the composite surface formed by their union $U_n \cup U_{n+1}$ may lose continuity along the boundary. This happens when independently resolved annuli are represented with two different resolutions close to their common boundary; see Figure 9. The size of the gap is generally smaller than the tolerance which was used to generate these annuli. Therefore, additional steps are necessary to ensure that the composite surface formed by two successive primary annuli is at least continuous. This can be easily accomplished by refining the edges between annuli to the finest resolution on both sides and replacing the points on the edge with the coarser mesh with values from the finer mesh. The boundary curves are composite Bézier curves, so they can be refined using the one-dimensional de Casteljau algorithm of Subsection 4.1. This approach is straightforward and is computationally inexpensive. The resulting manifold has only $C^0$ continuity across the annular joints $\Gamma_n$, but we found that achieving stronger smoothness greatly slowed the computation without producing more accurate or better-looking manifolds.

In the case of the map with rotation, this part has to be done carefully. Because of the rotation, matching of the boundary Bézier triangles is not automatic. At every annular joint we need to track the rotation angle, i.e., the number of triangles about which the consecutive annulus is rotated; see Figure 13b.

6.6 Distance Control along the Manifold

The method discussed so far is based on simple forward mapping of a primary annulus. The images of such an annulus do not grow uniformly in all radial directions; see Figures 14a and 14b. Away from the fixed point, the shapes of successive images could be very complicated. For example, the image of the initial primary annulus can grow a “finger” that stretches and that could wind around the computed portion of the manifold several times. We discuss such behavior of the invariant manifold in case of the first map in the next section.

We partially solved this problem in our implementation of A-QI. In order to accelerate the computation we can neglect the situation when errors above the tolerance occur at points with sufficiently large geodesic distance from the fixed point. Using properties of Bézier triangles we can control the distance between each quartic triangle and the fixed point along the manifold; for example see Figure 15b. Note that the edges of Bézier triangles form composite Bézier curves situated on the manifold, and their length can be easily approximated using the Gravesen algorithm [9]. This distance is not the same as the geodesic distance; however it can be used to bound the distance between a point along the manifold and the fixed point. Similarly, for the drawing process, we can avoid plotting parts that have drifted far away. An improvement of this step may significantly improve the proposed method.

7 Numerical Tests

Testing a method for computing two-dimensional invariant manifolds is much harder than testing in the one-dimensional case. The authors of the previous methods briefly introduced in Section 3 provide some numerical tests of their methods.
Mireles and Lomelí in [15] tested the parameterization method, providing the quantitative error analysis by evaluating a residual. However, these tests show the validity of the method locally inside the series convergence region for $P(r, \theta)$. Each new point on the manifold is generated by the repeated matching of a point taken from initial primary annulus, so there is no interpolation error and error growth is due solely to errors on $U_0$ and to roundoff. Krauskopf and Osinga perform even fewer quantitative tests on their methods, and simply use them to visualize a few examples of invariant manifolds without publishing any numerical error analysis.

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

### 7.1 Example 1: Volume-Preserving Hénon Map

First we follow Mireles-James and Lomelí [15] and consider the volume-preserving Hénon family introduced in [13]. These maps are defined by

\[
\begin{pmatrix}
  f_1(x, y, z) \\
  f_2(x, y, z) \\
  f_3(x, y, z)
\end{pmatrix}
= \begin{pmatrix}
  \alpha + \tau x + z + ax^2 + bxy + cy^2 \\
  x \\
  y
\end{pmatrix}, \tag{19}
\]

where $a + b + c = 1$ to guarantee volume preservation. The relevance of the volume-preserving property is to guarantee that a generic fixed point has the features of interest: namely a two dimensional unstable manifold and one dimensional stable manifold or vice versa. The map is quadratic with an explicit quadratic inverse.

The Hénon map has two hyperbolic fixed points $f(x_\pm, x_\pm, x_\pm) = (x_\pm, x_\pm, x_\pm)$ where $x_\pm = -\tau/2 \pm \sqrt{\tau^2 - 4\alpha}/2$. Its Jacobian matrix is

\[
Df(x, y, z) = \begin{pmatrix}
  \tau + 2ax + by & 2cy + bx & 1 \\
  1 & 0 & 0 \\
  0 & 1 & 0
\end{pmatrix}.
\]

As the determinant of the Jacobian is identically one; the matrix $Df$ will generically have either three real eigenvalues, or one real eigenvalue and one complex conjugate pair. Since the product of the three must be one, it has either a two dimensional stable eigenspace, or a two dimensional unstable eigenspace. For the parameters used, the fixed point $x_+$ has a two dimensional unstable manifold, while the stable manifold of $x_-$ is two-dimensional. In both cases, the relevant eigenvalues form a complex conjugate pair.

We implement the parametrization method described in Section 3 to approximate the initial annulus of $W^u(x_+)$, truncating the series $P(r, \theta)$ at order 60. Since the inverse map may be found explicitly we can also use the method in order to compute $W^s(x_-)$.

The complex conjugate pair of eigenvalues of the Hénon map imply that the map has a rotation component. Because of this, as described in Subsection 6.3 we define a sheared parametric grid in order to generate the approximation $U_0$ that guarantees consistency between annuli along the manifold. In order to match the points of the triangulation with their images under the map, the number of triangles in each column of triangulation should be a multiple of 3; see Figure 13.

The manifold looks like “an onion” where wider part around the fixed point is the bulb and the rapidly-growing “fingers” create its stalk; see Figures 13. If we unroll the onion we
Figure 14: Two-dimensional stable and unstable manifolds of the Hénon map given by the Equation (19) for \((a, b, c, \alpha, \tau) = (0.44, 0.21, 0.35, -0.25, -0.3)\), \(|\lambda_s| = 0.8482\), \(|\lambda_u| = 1.1737\), \(\text{tol} = 10^{-3}\) and the starting triangular mesh \(1 \times 15\), (a) red-yellows patches—stable, blue-green – unstable manifold, the initial portion and 15 primary annuli with \(r_0 = 0.5\), (b) 38 primary annuli of the stable manifold for \(r_0 = 0.01\), with the orbit of one patch highlighted in yellow; (c) schematic of the geodesic distance from the fixed point along one of the manifolds, (d) convergence of the proposed method tested on manifolds presented in (a) with tolerance from \(\text{tol} = 0.1\) to \(\text{tol} = 0.1 \times 2^{-16}\).

see parts that grow fast and parts that grow slowly; see Figure 14. During the calculation process we notice that image of a single Bézier triangle under the map rotates around the fixed point. Next, it hits one of the three “fingers”, and at each further iterate, its image cycles through these three fingers; see Figures 14b and 14c. This is caused by the influence of the nonlinear part of the map. In regions where the linear part of the map dominates, the dynamics are dominated by rotation around the fixed point. In regions where nonlinear part dominates, the “fingers” start to grow. After a few steps the primary annulus has a complex shape with some parts fairly close to the fixed point but also with “fingers” far away.

This phenomenon suggests it should be helpful to employ some distance control in computing and rendering a manifold. We use the idea described in Subsection 6.6 which slightly improves the speed of the computation and its visualization; see Figure 15. Lomelí and Meiss in [13] prove that all bounded orbits of the volume-preserving Hénon map are contained in a cube and that points outside this cube go to infinity as \(n \to \pm \infty\). Because of this, we can neglect parts of the manifold that have drifted away outside of the cube.
Figure 15: Two-dimensional stable and unstable manifolds of the Volume-Preserving Hénon map given by the Equation (19); (a) \((a, b, c, \alpha, \tau) = (0.5, 0, 0.5, 0, -0.3)\), eigenvalues s.t. \(|\lambda_s| = 0.9536, |\lambda_u| = 1.0487, r_0 = 0.3\) and 59 primary annuli are generated with \(\text{tol} = 10^{-4}\), the initial triangular mesh \(1 \times 18\), the distance cutoff along the manifold 2.1; (b) \((a, b, c, \alpha, \tau) = (0.44, 0.21, 0.35, -0.25, -0.3)\), eigenvalues \(|\lambda_s| = 0.8482, |\lambda_u| = 1.1737, r_0 = 1.1\), 13 primary annuli generated with \(\text{tol} = 10^{-3}\) and the triangular mesh \(1 \times 36\), the distance cutoff along the manifold is 5.

Due to the rotation, the orbit of a point on the primary annulus rotates around the manifold. At each step, its angle increases by about \(2\pi/3\). It is possible for there to exist points on \(W^u\) that are further from \(x^*\) than are their own images under \(f\), when measured by geodesic distance; see point \(p_1\) in Figure 14c. This could happen close to the fixed point when the behavior of the manifold is driven by the linear part of the map, or further away where nonlinear effects dominate. Our numerical test shows that is the case in this example. Because of this, the Krauskopf-Osinga method fails to compute this manifold for the second reason enumerated on page 5. This manifold can not be grown uniformly in the radial direction while controlling geodesic distance of growth.

7.1.1 Convergence Test

In order to show the convergence of the proposed method, we generate composite surface representations of the manifold for several decreasing values of the refinement condition and check how the difference between the approximation and the points mapped straightforwardly (and slowly!) from the highly accurate local approximation to the manifold, measured using the maximum \((L^\infty)\) norm, decays; see Figure 14d. It is clear that the proposed method converges well. The test confirms our expectation for the proposed method.

7.2 Example 2: Arneodo-Coullet-Tresser Map

This family of maps was introduced by Arneodo, Coullet and Tresser, motivated by the study of strange attractors in a family of differential equations on \(\mathbb{R}^3\) with homoclinic points of Shilnikov type. Their numerical computations show some interesting phenomena in the dynamical behavior of these maps, such as a period doubling cascade and a strange attractor.
More detailed bifurcation analysis is described by Du et al. in [5]. The ACT map is given by

\[
\begin{align*}
\begin{pmatrix}
ax - \omega b(y - z) \\
\frac{b}{\omega}x + a(y - z) \\
\frac{c}{\omega}x - dx^2 + ez
\end{pmatrix}
\end{align*}
\]

(20) \{eq:ACTmap\}

with fixed points at the origin, \(x_0\), and at \(x^* = \left( x_1, \frac{a^2 + b^2 - a}{\omega b} x_1, \frac{(a-1)^2 + b^2}{\omega b} x_1 \right) \) for \( x_1 = \frac{bc - (1 - e)(a-1)^2 + b^2}{\omega b d} \).

The local stability of the fixed points depends on the choice of parameters. Du et al. in [5], similarly to Lomelí and Meiss for the volume-preserving Hénon map, showed that the non-wandering set and the set of bounded orbits of the ACT map both lie in a finite box.

![Two-dimensional unstable manifold of the ACT map given by the Equation (20) with \((a, b, c, d, e, k, \omega) = (0.2, 0.5, 0.5, 1, 1, 2, 4)\) associated with nontrivial fixed point, eigenvalue s.t. \(|\lambda_u| = 1.20\) and 30 primary annuli generated for \(r_0 = 0.02\), \(\text{tol} = 10^{-5}\) and the initial mesh \(1 \times 36\), (b) sections of the manifold with planes containing the stable direction. The zoomed boxes do not preserve aspect ratio.](fig:ACTmanifold)

Figure 16: (a) Two-dimensional unstable manifold of the ACT map given by the Equation (20) with \((a, b, c, d, e, k, \omega) = (0.2, 0.5, 0.5, 1, 1, 2, 4)\) associated with nontrivial fixed point, eigenvalue s.t. \(|\lambda_u| = 1.20\) and 30 primary annuli generated for \(r_0 = 0.02\), \(\text{tol} = 10^{-5}\) and the initial mesh \(1 \times 36\), (b) sections of the manifold with planes containing the stable direction. The zoomed boxes do not preserve aspect ratio.

For certain parameter values, see caption of the Figure 16, the ACT map has a pair of hyperbolic fixed points and, associated with each, pair of complex eigenvalues; i.e., the map has a rotation component. As in the previous example, we use the sheared parametric grid in order to generate the approximation \(U_0\). The unstable manifold of the ACT map associated with nontrivial fixed point \(x^*\) is shown in Figure 16a, we notice that it does not grow far away from fixed point. Subsequent primary annuli stay fairly close to \(x^*\) while rotating around it. The unstable manifold seems to be bounded.

Due to the rotation, stretching and folding the unstable manifold presents similar dynamics as the Rössler, a familiar model with chaotic dynamics and an attractor. The manifold displays the same type of banding as the attractor; it has a half-twist in it, which makes it look somewhat like a Möbius strip. Note also that since \(W^u\) is bounded, but expanding due the eigenvalue \(\lambda_u\) it folds back on itself. Points on \(W^u\) may have large distance from \(x^*\) along the manifold but short Euclidean distance from \(x^*\).

Obtaining an approximate initial annulus \(U_0\) accurate to 15 digits via the parameterization method required a very small radius \(r \approx 0.02\) in the power series expansion, so the portion
of $W^u$ representable by this method is very small. To resolve the interesting features (e.g., intersection of manifolds or folding) much further away, a method like ours is required.

7.2.1 Cross-Sections

Figure 16a shows that the unstable manifold of the ACT map folds back and forth while rotating around the fixed point. The folding creates at least four visible layers. Increasing the number of iterations in the computation would generate more such features. The folds of the manifold suggest looking more closely at its intersection with a plane containing the stable direction of the fixed point, a leaf of the foliation presented in Section 3. The two sample sections are presented in Figure 16b, along with closeup views. Sections of consecutive annuli are drawn using alternating red and blue. The section curves fold back and forth making sharp turns. Our computations show that for both cross-sections curvature can takes values up to order $10^9$. In the boxes we magnify spots where the folds creates multiple layers of the manifold. This confirms the complexity of the manifold of the ACT map. The method of Krauskopf-Osinga [12] requires setting a spatial scale at the beginning of the computation and thus would either under-resolve certain features or else require a very large number of points.

8 Discussion

The method proposed here for computing unstable manifolds of the three-dimensional maps incorporates ideas from computer aided geometric design. The basic approach of our computation is forward mapping of a local portion of an invariant manifold. The strategy of this study has been to improve this technique.

We have subjected the method to more stringent tests than had been previously presented. The results show that the method achieves not only significant improvements in the accuracy of the calculation but can also handle the problems where the other methods have failed. High accuracy and near-$C^1$ continuity of the approximation, pleasant visual appearance, and reduction of the number of calls of the map $f$ allow us to see the method as the successor of these previously proposed. The method performs exceptionally well in the case of a bounded invariant manifold, even with highly varying complexity, smoothly resolving places of high curvature or exponential stretching. In the case of an unbounded, nonuniformly growing invariant manifold it also does well, however, during the process of computation it performs some unnecessary calculations: the pieces of the manifold that are far away from the fixed point are refined to no real purpose. Distance control along the manifold reduces the number of computations, partially solving the problem, but not eliminating it completely, while significantly speeding up rendering. Iterating fundamental annuli requires at each step a region of parameter space. Removing this restriction while maintaining the high order of accuracy could lead to a significant improvement in these computations.

As we found for one-dimensional methods, there are other costs involved with the implementation of this higher-order method, in comparison, with methods based on bivariate linear interpolation. In particular, one piece of a plane triangular interpolant can be plotted using only its values at the three corner points, with high-level graphing software filling in the points at the planar triangle. To render a quartic Bézier patch, one first samples the patch at a finite number of uniformly triangulated barycentric coordinates, and then plots a
planar triangular interpolant through those points. Each of these points is evaluated by the de Casteljau algorithm for Bézier patches. The resolution of the triangulation is determined by the scale of the most-finely resolved Bézier triangles, so that the number of facets used to resolve a given Bézier triangle is inversely proportional to the square of the resolution level. This procedure constructs a continuous surface without cracks or pinholes.

Acknowledgments

For helpful discussions the authors would like to thank Denis Blackmore, Gershon Elber, Martin Hering-Bertram, Rafael de la Llave, Hector Lomelí, Jason Mireles-James, James Meiss, Hinke Osinga, and Denis Zorin. The author received support from NSF DMS-0807284.

A Software

The authors have written a small set of MATLAB programs that implement the methods described in this article and have included them in the supplementary material with the publication. These are also available from the author’s website


References


