The Abdus Salam
International Centre for Theoretical Physics

United Nations
Educational, Scientific and
Cultural Organization

IAEA
International Atomic Energy Agency

2247-11

School and Conference on Computational Methods in Dynamics

20 June - 8 July, 2011

A Survey of methods computing (un)stable manifolds of vector fields

John Guckenheimer and more authors
Department of Mathematics, Cornell University
Ithaca NY
United States of America

A survey of methods for computing (un)stable manifolds of vector fields

B. Krauskopf & H.M. Osinga
Department of Engineering Mathematics, University of Bristol,
Queen's Building, Bristol BS8 1TR, UK

E.J. Doedel
Department of Computer Science, Concordia University,
1455 Boulevard de Maisonneuve O.,
Montréal Québec, H3G 1M8 Canada

M.E. Henderson
IBM Research, P.O. Box 218, Yorktown Heights, NY 10598, USA

J. Guckenheimer & A. Vladimirsky
Department of Mathematics, Cornell University,
Malott Hall, Ithaca, NY 14853–4201, USA

M. Dellnitz & O. Junge
Institute for Mathematics, University of Paderborn,
D-33095 Paderborn, Germany

Final draft of April 22, 2004

Keywords: stable and unstable manifolds, numerical methods, Lorenz equation.

Abstract

The computation of global invariant manifolds has seen renewed interest in recent years. We survey different approaches for computing a global stable and unstable manifold of a vector field, where we concentrate on the case of a two-dimensional manifold. All methods are illustrated with a single example — the two-dimensional stable manifold of the origin in the Lorenz system.

1 Introduction

Many applications give rise to mathematical models in the form of a system of ordinary differential equations. Well-known examples are periodically forced oscillators and the Lorenz system (introduced in Section 1.1); see, for example, [21, 37, 54] for further references. Such a dynamical system can be written in the general form

$$\frac{d\mathbf{x}}{dt} = f(\mathbf{x}), \tag{1}$$

where $\mathbf{x} \in \mathbb{R}^n$ and the map $f : \mathbb{R}^n \mapsto \mathbb{R}^n$ is sufficiently smooth. We remark that, in general, the function f will depend on parameters. However, we assume that all parameters are fixed and use (1) as the appropriate setting for the discussion of global manifolds.

The goal is to understand the overall dynamics of system (1). To this end, one needs to find special invariant sets, namely the equilibria, periodic orbits, and possibly invariant tori. Already to find these objects (with the possible exception of equilibria) one will need to use numerical techniques. Furthermore, if these invariant sets are of saddle type then they come with global stable and unstable manifolds. For example, the stable and unstable manifolds $W^s(\mathbf{x}_0)$ and $W^u(\mathbf{x}_0)$ of a saddle equilibrium $\mathbf{x}_0$ are defined as

$$\begin{aligned} W^s(\mathbf{x}_0) &:= \{\mathbf{x} \in \mathbb{R}^n \mid \lim_{t\to\infty} \phi^t(\mathbf{x}) = \mathbf{x}_0\} \\ W^u(\mathbf{x}_0) &:= \{\mathbf{x} \in \mathbb{R}^n \mid \lim_{t\to\infty} \phi^{-t}(\mathbf{x}) = \mathbf{x}_0\}, \end{aligned}$$

respectively, where ϕ^t is the flow of (1). Hence, trajectories on the stable (unstable) manifold converge to $\mathbf{x}_0$ in forward (backward) time. Knowing these manifolds is crucial as they organize the dynamics on a global scale. For example, stable manifolds may form boundaries of basins of attraction, and it is well known that intersections of stable and unstable manifolds lead to complicated dynamics and chaos.

Generally, global stable and unstable manifolds can not be found analytically. Furthermore, they are not implicitely defined, meaning that it is not possible to find them as the zero-set of some function. Hence, points on global invariant manifolds cannot be found 'locally'. Instead, these manifolds must be 'grown' from local knowledge, for example from linear information near a fixed point $\mathbf{x}_0$.

It is the purpose of this paper to review different numerical techniques that have recently become available to compute these global objects. We review five algorithms in detail and characterize their properties using a common test-case example, namely, the *Lorenz manifold* which is introduced now.

1.1 The Lorenz manifold

The Lorenz system [38] is a classic example of a vector field with a chaotic attractor. It is given as

$$\left\{ \begin{array}{rcl} \dot{x} &=& \sigma(y - x), \\ \dot{y} &=& \varrho x - y - xz, \\ \dot{z} &=& xy - \beta z, \end{array} \right. \tag{2}$$

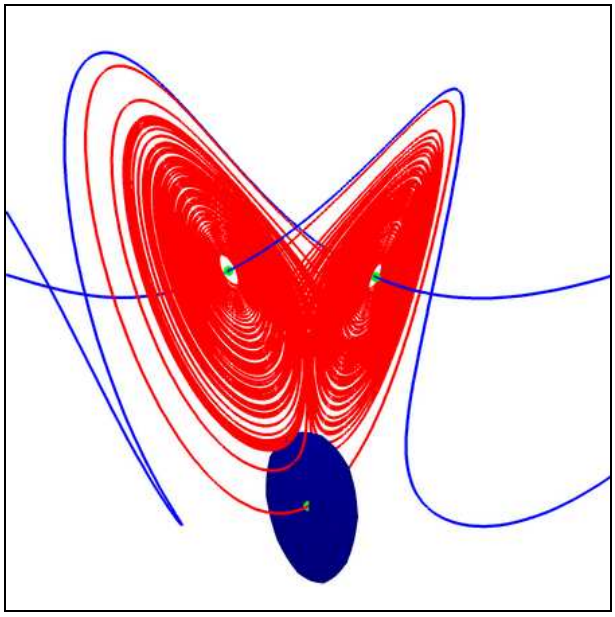

Figure 1: The unstable manifold $W^u(\mathbf{0})$ (red curve) accumulates on the butterfly-shaped Lorenz attractor. The blue disk is the linear approximation $E^s(\mathbf{0})$ of the Lorenz manifold $W^s(\mathbf{0})$. Also shown are the two equilibria at the centres of the 'wings' of the butterfly and their one-dimensional stable manifolds (blue curves).

where we fix the parameters at the standard choice $\sigma = 10$, $\varrho = 28$ and $\beta = 8/3$, for which one finds the famous butterfly-shaped Lorenz attractor. Note that the Lorenz system (2) has the symmetry $(x, y, z) \mapsto (-x, -y, z)$ of rotation around the z-axis. In particular, the z-axis is invariant under the flow.

The origin is a saddle point of (2) with real eigenvalues $-\beta$ and $-\frac{\sigma+1}{2} \pm \frac{1}{2}\sqrt{(\sigma+1)^2 + 4\sigma(\rho-1)}$, that is, approximately -22.828, -2.667 and 11.828. The origin is contained in the Lorenz attractor, so that its one-dimensional unstable manifolds $W^u(\mathbf{0})$ can be used to approximate the Lorenz attractor; this is illustrated in Figure 1 where $W^u(\mathbf{0})$ is shown in red. At the centers of the 'wings' of the butterfly are two more equilibria of (2), approximately at $(\pm 8.485, \pm 8.485, 27)$, which are each other's image under the rotational symmetry. Each of these equilibria has one negative real eigenvalue, giving rise to a one-dimensional stable manifold, and an unstable pair of complex conjugate eigenvalues with positive real part. Figure 1 shows all equilibria of (2) in green, together with their one-dimensional global manifolds. As mentioned, the red curve is the unstable manifold $W^u(\mathbf{0})$ of the origin, whose closure is the Lorenz attractor. The blue curves are the stable manifolds of the two other equilibria. The blue disk lies in the linear eigenspace $E^s(\mathbf{0})$ of the origin.

The Lorenz attractor, that is, the red curve in Figure 1 conveys the chaotic nature of the system, but does not give any information on the overall organization of the phase space of (2). This role is played by the two-dimensional stable manifold $W^s(\mathbf{0})$ of the origin — which we refer to as the *Lorenz manifold* from now on. The Lorenz manifold $W^s(\mathbf{0})$ is tangent at 0 to the eigenspace $E^s(\mathbf{0})$ spanned by the eigenvectors of -22.828 and -2.667. This is a generic property of stable and unstable manifolds; see Section 1.2. Note the large difference

in magnitude between the two stable eigenvalues, leading to a dominance of the strong stable manifold, which is tangent to the eigenspace of the the eigenvalue -22.828.

The Lorenz manifold has a number of astonishing properties. Imagine that the little blue disk in Figure 1 'grows' to become the Lorenz manifold $W^s(\mathbf{0})$, but without ever intersecting the red unstable manifold $W^u(\mathbf{0})$. In other words, the Lorenz manifold stays 'in between' trajectories on the Lorenz attractor, but 'spirals' simultaneously into both wings of the butterfly. Now imagine how trajectories on this manifold must be able to pass from one wing to the other. Any finitely grown part of $W^s(\mathbf{0})$ is topologically still a two-dimensional disk, but one with a particularily intriguing embedding into $\mathbb{R}^3$. The geometry of $W^s(\mathbf{0})$ can only truly be appreciated if one can draw an image of it.

The first, hand-drawn image of the Lorenz manifold appeared in [1] and the first published computer generated image appeared in [23]. Not in the least due to its intriguing nature, the Lorenz manifold has become a much-used test-case example for evaluating algorithms that compute two-dimensional (un)stable manifolds of vector fields. For each of the methods discussed in this paper we present an image of the computed Lorenz manifold that is always taken from a viewpoint along the line spanned by the vector $(\sqrt{3}, 1, 0)$ in the (x, y)-plane.

1.2 Stable and unstable manifolds

In order to explain the different methods for computing two-dimensional (un)stable manifolds, we need to introduce some notation. To keep the exposition simple, we consider here the case of a global (un)stable manifold of a hyperbolic saddle point $\mathbf{x}_0 \in \mathbb{R}^n$ of (1). Furthermore, we present all theory and the different methods for the case of an unstable manifold. This is not a restriction, because a stable manifold can be computed as an unstable manifold when time is reversed in system (1).

Suppose now that $f(\mathbf{x}_0) = \mathbf{0}$ and for some $1 < k < n$ the Jacobian $Df(\mathbf{x}_0)$ of f at $\mathbf{x}_0$ has k eigenvalues with positive real parts and $(n-k)$ eigenvalues with negative real parts (counted with multiplicity). The Stable and Unstable Manifold Theorem states that a local unstable manifold $W^u_{\rm loc}(\mathbf{x}_0)$ exists in a neighborhood of $\mathbf{x}_0$. Furthermore, $W^u_{\rm loc}(\mathbf{x}_0)$ is as smooth as f and tangent to the unstable (generalized) eigenspace $E^u(\mathbf{x}_0)$ of $Df(\mathbf{x}_0)$ at $\mathbf{x}_0$. This means that we may define the global unstable manifold $W^u(\mathbf{x}_0)$ as

$$W^u(\mathbf{x}_0) \;=\; \{\mathbf{x} \in \mathbb{R}^n \mid \lim_{t \to -\infty} \phi^t(\mathbf{x}) = \mathbf{x}_0\} = \bigcup_{t>0} \phi^t(W^u_{\rm loc}(\mathbf{x}_0))\,. \tag{3}$$

Hence, $W^u(\mathbf{x}_0)$ is a k-dimensional (immersed) manifold, defined as the globalization of $W^u_{\rm loc}(\mathbf{x}_0)$ under the flow ϕ^t. Note that the local stable manifold $W^s_{\rm loc}(\mathbf{x}_0)$ and the stable manifold $W^s(\mathbf{x}_0)$ are similarly related with respect to the reversed direction of time, namely

$$W^s(\mathbf{x}_0) \;=\; \{\mathbf{x} \in \mathbb{R}^n \mid \lim_{t \to \infty} \phi^t(\mathbf{x}) = \mathbf{x}_0\} = \bigcup_{t<0} \phi^t(W^s_{\rm loc}(\mathbf{x}_0))\,. \tag{4}$$

This indeed shows that it is sufficient to consider only the case of an unstable manifold, possibly after reversing time.

Definition (3) already suggests a method for computing $W^u(\mathbf{x}_0)$: take a small $(k-1)$-sphere (or other 'outflow boundary' such as an ellipsoid) $S_\delta \subset W^u_{\rm loc}(\mathbf{x}_0)$ with radius δ around

$\mathbf{x}_0$ and 'grow' the manifold $W^u(\mathbf{x}_0)$ by evolving S_δ under the flow ϕ^t. As starting data, one can take $S_\delta \subset E^u(\mathbf{x}_0)$ or a higher-order approximation of $W^u_{\rm loc}(\mathbf{x}_0)$.

In the special case $k = 1$ of computing a one-dimensional manifold, this method works well, because it boils down to evolving two points at distance δ from $\mathbf{x}_0$ under the flow. This can be done reliably by numerical integration of (1), so that computing one-dimensional unstable manifolds is straightforward. The one-dimensional manifolds in Figure 1 were computed in this way.

However, the above method of evolving a $(k-1)$-sphere S_δ with $k \geq 2$ under the flow ϕ^t generally gives very poor results. This is so because S_δ will typically deform very rapidly under ϕ^t. In particular, it will stretch out along the strong unstable directions (if present). Furthermore, S_δ is a continuous object that will have to be discretized by some mesh. Any mesh on S_δ will deteriorate rapidly under the flow ϕ^t, so that it will not be a good representation of $W^u(\mathbf{x}_0)$ as a k-dimensional manifold.

1.3 Different approaches to computing $W^u(\mathbf{x}_0)$

It is quite a challenge to compute a global unstable manifold $W^u(\mathbf{x}_0)$ of dimension at least two. Indeed simple numerical integration of the flow is not sufficient (except in very special cases) — dedicated algorithms are needed for this task. Before we describe some recent methods in more detail, we first explain the underlying approaches in general terms. It is useful to consider for this purpose different parametrizations of $W^u(\mathbf{x}_0)$.

We concentrate in this survey on the first nontrivial case $k = 2$ of a two-dimensional unstable manifold. While all methods could be used in principle to compute higher-dimensional manifolds, almost all implementations are for $k = 2$. Furthermore, visualizing higher-dimensional manifolds remains a serious challenge. The different methods use the idea of growing $W^u(\mathbf{x}_0)$ from a local neighborhood of $\mathbf{x}_0$. They differ in how they ensure that a good mesh representing $W^u(\mathbf{x}_0)$ is computed during this growth process.

Consider as starting data a small smooth closed curve $S_\delta \subset W^u_{\rm loc}(\mathbf{x}_0)$, also referred to as a (topological) circle in what follows, of points that all lie at distance δ from $\mathbf{x}_0$. (As was mentioned, one can take $S_\delta \subset E^u(\mathbf{x}_0)$ if δ is small enough.) The goal is to find a 'nice' parametrization of $W^u(\mathbf{x}_0)$ in terms of the starting data S_δ.

As we have seen above, the parametrization

$$W^u(\mathbf{x}_0) = \{\phi^t(S_\delta)\}_{t\in\mathbb{R}} \tag{5}$$

is not practical, because the $\phi^t(S_\delta)$ are typically not nice smooth closed curves for all t. Indeed, they soon tend to look like very elongated ellipses.

In order to define the parametrization of $W^u(\mathbf{x}_0)$ as a family of the nicest topological circles possible, recall that the geodesic distance $d_g(\mathbf{x}, \boldsymbol{y})$ is defined as the arclength of the shortest path in $W^u(\mathbf{x}_0)$ connecting $\mathbf{x}$ and $\boldsymbol{y}$, called a *geodesic*. Consider now the geodesic parametrization of $W^u(\mathbf{x}_0)$ given by

$$W^u(\mathbf{x}_0) = \{S_\eta\}_{\eta>0} \quad \text{where} \quad S_\eta := \{\mathbf{x} \in W^u(\mathbf{x}_0) \mid d_g(\mathbf{x}, \mathbf{x}_0) = \eta\}\,. \tag{6}$$

The geodesic parametrization (6) is entirely in terms of the geometry of $W^u(\mathbf{x}_0)$, and not in terms of the dynamics on the manifold. Since $W^u(\mathbf{x}_0)$ is a smooth manifold tangent to $E^u(\mathbf{x}_0)$

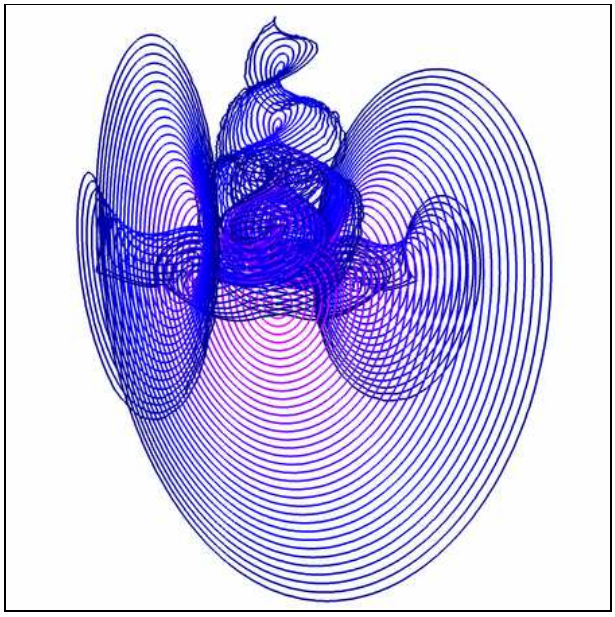

Figure 2: The Lorenz manifold computed with the method of Guckenheimer & Worfolk [23] up to geodesic distance 180; the computed approximate geodesic level sets are at increasing radial distances from the origin with steps of 5.0 in between, which is indicated by a color change from magenta (small) to blue (large).

at $\mathbf{x}_0$, there must be some $\eta_{\max} > 0$ so that the geodesic level sets $\mathcal{S}_\eta$ for $0 < \eta \leq \eta_{\max}$ are all smooth closed curves without self-intersection, that is, topological circles; see, for example, [53]. We also refer to geodesic level sets for $\eta \leq \eta_{\max}$ as *geodesic circles*. Up until $\eta_{\max}$, the geodesic parametrization (6) is geometrically the nicest parametrization, because its elements, the geodesic circles, are the nicest possible topological circles on $W^u(\mathbf{x}_0)$. For the Lorenz manifold, apparently $\eta_{\max} = \infty$. However, the case of a finite $\eta_{\max}$ is possible and it typically involves a non-smooth geodesic circle; see [35] for details.

The idea of computing $W^u(\mathbf{x}_0)$ as a sequence of geodesic circles goes back to Guckenheimer and Worfolk [23]. Starting with a small geodesic circle (or ellipse) S_δ around $\mathbf{x}_0$, they rescale the vector field so that the component tangential to the last computed geodesic level set is practically zero, retaining only the radial part. Then the flow of the rescaled radial vector field is used to evolve (a sufficient number of points on) this geodesic circle by integration over a suitably small and fixed integration time (now corresponding to geodesic distance up to a rescaling of the radial part of the vector field). Figure 2 shows 36 approximate geodesic circles of the Lorenz manifold computed with this method up to geodesic distance 180. The output was produced in the DsTool [4] software environment; the manifold could be rendered as a two-dimensional surface by post-processing the data. Notice that the last few computed circles show some ripples near the z-axis. This happens where the vector field is almost tangent to the geodesic circles, so that the rescaling to determine the radial part of the vector field becomes unstable. More generally, the method from [23] requires that the vector field points radially outward *everywhere* along each geodesic circle, which is why this approach has difficulties in case $Df(\mathbf{x}_0)$ has complex conjugate eigenvalues.

The method by Krauskopf and Osinga [34, 35], discussed in detail in Section 2, also computes $W^u(\mathbf{x}_0)$ as a sequence of geodesic circles, but does not rescale the vector field. Instead, the idea is to find the next geodesic circle in a local (and changing) coordinate system given by hyperplanes perpendicular to the present geodesic circle. Determined by certain accuracy parameters, a suitable number of mesh points on the next geodesic circle is computed by solving appropriate boundary value problems. During the computation the interpolation error stays bounded, so that the overall quality of the mesh is guaranteed.

A different approach is to reparametrize time so that the flow with respect to the new time progresses with the same speed along all trajectories through S_δ, meaning that the same arclength is covered per unit time along all trajectories. One also speaks of arclength integration. We then have the new parametrization of $W^u(\mathbf{x}_0)$ given by

$$W^u(\mathbf{x}_0) = \{A_\eta\}_{\eta>0} \quad \text{with} \quad A_\eta := \{\mathbf{x} \in W^u(\mathbf{x}_0) \mid d_a(\mathbf{x}, \mathbf{x}_0) = \eta\}\,, \tag{7}$$

where $d_a(\mathbf{x}, \boldsymbol{y})$ denotes the arclength distance between two points $\mathbf{x}$ and $\boldsymbol{y}$ on the same trajectory; we set $d_a(\mathbf{x}, \boldsymbol{y}) = \infty$ if $\mathbf{x}$ and $\boldsymbol{y}$ are not on the same trajectory. This parametrization can be considered as the best in terms of dynamically defined topological circles on $W^u(\mathbf{x}_0)$.

Johnson *et al.* [27] use essentially this parametrization by trajectory arclength, but consider integration in the product of time and phase space. They start with a uniform mesh on a first small circle $A_\delta \in E^u(\mathbf{x}_0)$ and then integrate at each step the present mesh points up to a specified arclength. This leads to a new circle, on which a uniform mesh is then constructed by interpolation between the integration points. Figure 3 shows the Lorenz manifold computed with this method up to an approximate arclength distance of 200. The method is quite fast since it involves only direct integration and redistribution of points by interpolation. On the other hand, it is difficult to control the interpolation error, which is determined by the (unknown) dynamics on $W^u(\mathbf{x}_0)$.

An altogether different parametrization of $W^u(\mathbf{x}_0)$ is the dual parametrization to (5) and (7) in terms of the trajectories through a fixed $S_\delta \subset E^u(\mathbf{x}_0)$, which is formally given as

$$W^u(\mathbf{x}_0) = \{B_p\}_{p \in S_\delta} \quad \text{where} \quad B_p := \{\phi^t(p) \mid t \in \mathbb{R}\}\,. \tag{8}$$

Notice that, in the case of a two-dimensional manifold $W^u(\mathbf{x}_0)$ considered here, parametrization (8) is a one-parameter family of trajectories.

The method by Doedel, discussed in detail in Section 3, computes two-dimensional (un)stable manifolds by following trajectories B_p as a boundary value problem where the initial condition $p \in S_\delta$ is parametrized with one of the free continuation parameters. This method is very accurate and flexible by allowing for different boundary conditions at the other end point of the trajectory B_p, which includes specifying a fixed arclength L of the trajectory. During a computation, mesh points are distributed along the trajectories to maintain the accuracy of the computation.

The method of Henderson [25], discussed in detail in Section 4, also considers parametrization (8) of $W^u(\mathbf{x}_0)$ by orbits. However, the manifold is constructed directly as a two-dimensional object by computing fat trajectories. A fat trajectory is a string of polyhedral patches along a trajectory, where the size of the patches is given by local curvature information. When a fat trajectory reaches the prescribed total arclength L, the boundary of the computed part of the manifold is determined. Then a suitable starting point for the next fat

Figure 3: The Lorenz manifold computed with the method of Johnson, Jolly & Kevrikidis [27] up to a total trajectory arclength of about 200.

trajectory is found and the computation continues. When no more possible starting points exist, the computation stops.

The method by Guckenheimer and Vladimirsky [22], discussed in detail in Section 5, locally models $W^u(\mathbf{x}_0)$ as the graph of a function g that satisfies a quasi-linear partial differential equation (PDE) expressing the tangency of the vector field f to the graph of g. The PDE is discretized in an Eulerian framework and the manifold is approximated by a triangulated mesh. At each step one new point is added to the mesh, leading to a new simplex whose other vertices are previously known mesh points. An Ordered Upwind Method determines where the next point/simplex is added and the ordering of new simplices is based on the arclength of the trajectories.

The method by Dellnitz and Hohmann [7, 8], discussed in detail in Section 6, is complementary to the previous methods in that it computes an outer approximation of the manifold by boxes of the same dimension n as the phase space of (1). This method uses the time-τ map of the flow ϕ^t for some fixed $\tau > 0$. A subdivision algortihm first finds a covering of $W^u_{\rm loc}(\mathbf{x}_0)$ with n-dimensional boxes of suitably small diameter. This local box covering is then globalized in steps by adding new boxes (of the same small size) that are 'hit' under the time-τ map by the present collection of boxes. The practical problem is to reliably detect when the image of one box intersects another box (for example, by using test points). If a-priori bounds on the local growth rate of the vector field are known then it is possible to compute a rigorous box covering of $W^u(\mathbf{x}_0)$; see [29].

In the following sections we present the different algorithms in more detail, again illustrated with the computation of the Lorenz manifold $W^s(\mathbf{0})$.

2 Approximation by geodesic level sets

The method of Krauskopf and Osinga [34, 35] approximates a global (un)stable manifold as a sequence of geodesic circles of the parametrization (6). Only the case of a two-dimensional unstable manifold of a saddle point in a three-dimensional space is presented here. However, the method can be formulated in terms of computing a k-dimensional manifold of a vector field in $\mathbb{R}^n$, and has been implemented to compute two-dimensional (un)stable manifolds of saddle points and saddle periodic orbits in a phase space of any dimension; see the examples in [34, 35] and also in [42, 43]. Variants of this method exist to compute global manifolds of maps; see [32, 33].

The method completely steps away from evolving an existing mesh. Instead, new mesh points are computed by means of solving appropriate boundary value problems; see Section 2.1. The boundary conditions predetermine where the new mesh points need to be added in order to achieve a prescribed mesh quality. This method is as independent of the dynamics as possible and it grows the manifold as a sequence of discretized geodesic circles until $\eta_{\max}$ is reached where the geodesic level sets are no longer smooth circles; see Section 1.3.

To be more specific, let M_i denote a circular list of mesh points from which a continuous topological circle C_i is formed by connecting neighboring points of M_i by line segments. The mesh points in M_i are computed to ensure that C_i is a good approximation (according to prespecified accuray parameters) of an appropriate geodesic circle S_{η_i}. The manifold $W^s(\mathbf{0})$ is then approximated up to a prescribed geodesic distance L by the triangulation formed by the total mesh $\mathcal{M} = \cup_{0 \le i \le l} M_i$, where $l \in \mathbb{N}$ depends on L and the accuracy parameters.

The start data is a uniform mesh M_0 on an initial small geodesic circle $S_{\eta_0} = S_\delta \subset E^u(\mathbf{x}_0)$ at some prescribed distance δ from 0. The method then computes at each step i a new circular list M_{i+1} that approximates the next level set $S_{\eta_{i+1}}$. In other words, at every step a new band is added to $W^u(\mathbf{x}_0)$; the width of this band is determined by the curvature of geodesics. The method stops when the prespecified fixed geodesic distance L from 0 is reached.

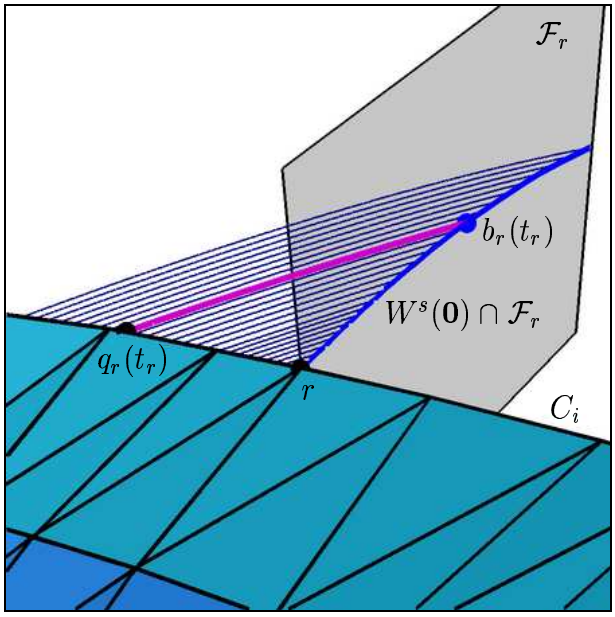

Figure 4: The boundary value problem formulated for a mesh point r on the geodesic level set C_i is solved by a family of trajectories, starting at $q_r(t)$ on C_i and ending at $b_r(t)$ in $\mathcal{F}_r$, that is parametrized by integration time t. There is a unique first orbit such that $\| b_r(t_r) - r \| = \Delta_i$. The image shows actual data for the Lorenz manifold $W^s(\mathbf{0})$ of Figure 5 where $C_i \approx S_{\eta_i}$ with $\eta_i = 32.75$ and $\Delta_i = 4.0$.

2.1 Finding a new point in M_{i+1}

Let us consider the task of finding M_{i+1} at some prescribed increment Δ_i from a known circular list M_i representing S_{η_i}. The circular list M_{i+1} is constructed pointwise. Let $r \in M_i$ and consider the (half)plane $\mathcal{F}_r$ through r that is (approximately) perpendicular to C_i at r. Then $W^u(\mathbf{x}_0) \cap \mathcal{F}_r$ is a well-defined one-dimensional curve locally near r, which is parametrized by the time it takes to reach $W^u(\mathbf{x}_0) \cap \mathcal{F}_r$ by integration from C_i. Points in $W^u(\mathbf{x}_0) \cap \mathcal{F}_r$ can be found by solving the two-point boundary value problem

$$q_r(t) \in C_i, \tag{9}$$

$$b_r(t) := \phi^t(c_r(t)) \in \mathcal{F}_r, \tag{10}$$

where the integration time t is a free parameter. The situation is shown in Figure 4 with actual data for the Lorenz manifold $W^s(\mathbf{0})$ presented in Section 2.3. Note that for an unstable manifold $t \geq 0$ and for a stable manifold $t \leq 0$.

The point $b_r(t_r) \in \mathcal{F}_r$ is uniquely defined by the property that t_r is the smallest integration time (in absolute value) for which $\| b_r(t_r) - r \| = \Delta_i$. If Δ_i is small enough then $b_r(t_r)$ exists and can be found by continuation of the trivial solution $b_r(0) = q_r(0) = r$ for $t = 0$ while checking for the first zero of the test function

$$\Delta_i - \| b_r(t) - r \| . \tag{11}$$

When the first zero is found then $b_r(t_r) = b_r(t)$ is the candidate for a point in M_{i+1}; see Figure 4.

2.2 Mesh adaptation

Once all candidate points in M_{i+1} have been found, all for the same Δ_i, then it is decided whether the step size Δ_i was appropriate. To this end, it is checked that the curvature of (approximate) geodesics through all points $r \in M_i$ was not too large. This is done with a criterion that was originally introduced for one-dimensional global manifolds of maps [26]. Let α_r denote the angle between the line through r and $b_r(t_r)$ and the line through p_r and r, where $p_r \in M_{i-1}$ is the associated point of M_{i-1} on the approximate geodesic. The step of geodesic distance Δ_i was acceptable if both

$$\alpha_r < (\alpha_{\max}, \quad \text{and} \tag{12}$$

$$\Delta_i \cdot \alpha_r < (\Delta\alpha)_{\max} \tag{13}$$

hold for all $r \in M_i$. In this case M_{i+1} is accepted and step i is complete. If there is some $r \in M_i$ that fails either (12) or (13) then Δ_i is halved and step i is repeated with this smaller Δ_i. Similarly, Δ_i may be doubled if for every $r \in M_i$ both α_r and $\Delta_i \cdot \alpha_r$ are well below the respective upper bounds in (12) or (13), say, below $\alpha_{\min}$ and $(\Delta\alpha)_{\min}$ respectively. The parameters $\alpha_{\min}$, $\alpha_{\max}$, $(\Delta\alpha)_{\min}$, and $(\Delta\alpha)_{\max}$ implicitely determine the mesh adaptation along geodesics and are fixed by the user before a computation.

It is important to ensure that C_{i+1} is also a good approximation of $S_{\eta_{i+1}}$. In other words, neighboring points of M_{i+1} may not be too close or too far from each other. When two neighboring points of M_i lead to two neighboring points of M_{i+1} at more than the prespecified distance $\Delta_{\mathcal{F}}$ from each other, then a new point is added in between. This is not done by interpolating between points of M_{i+1} but by applying step i of Section 2.2 for finding a new point in M_{i+1} to the middle point on C_i. In other words, no interpolation is ever performed between points that are more than distance $\Delta_{\mathcal{F}}$ apart. In order to ensure proper order relations between directly neighboring points of M_{i+1} a point is removed if two neighboring points in M_{i+1} lie closer together than a prespecified distance $\delta_{\mathcal{F}}$.

The mesh adaptation as decribed ensures that the overall error of a computation up to a prescribed geodesic distance L is bounded. This means that the computed piece of the manifold lies in an ε-neighborhood of $W^u(\mathbf{x}_0)$, provided the accuray parameters are chosen small enough; see [35] for the proof.

2.3 The Lorenz manifold approximated by geodesic circles

Figure 5 shows the Lorenz manifold $W^s(\mathbf{0})$ represented by a total of 75 bands and with total geodesic distance 154.75. The manifold was computed starting with a mesh M_0 of 20 points on $S_\delta \subset E^s(\mathbf{0})$ with $\delta = 1.0$. The computation was initiated with $\Delta_1 = 0.25$ and the mesh was generated using the accuracy parameters $\alpha_{\min} = 0.3$, $\alpha_{\max} = 0.4$, $(\Delta\alpha)_{\min} = 0.1$, $(\Delta\alpha)_{\max} = 1.0$, $\Delta_{\mathcal{F}} = 2.0$, and $\delta_{\mathcal{F}} = 0.67$. The coloring illustrates the geodesic distance from the origin, where blue is small, green is intermediate and red is large. The manifold was rendered as a two-dimensional surface with the visualization package Geomview [41]; other illustrations of the Lorenz manifold can be found in [35, 36, 44] and animations with [35, 36].

Figure 5(a) shows the entire computed part of the Lorenz manifold from the common viewpoint; notice the similarity with the geodesic level sets in Figure 2. Figure 5(b) shows

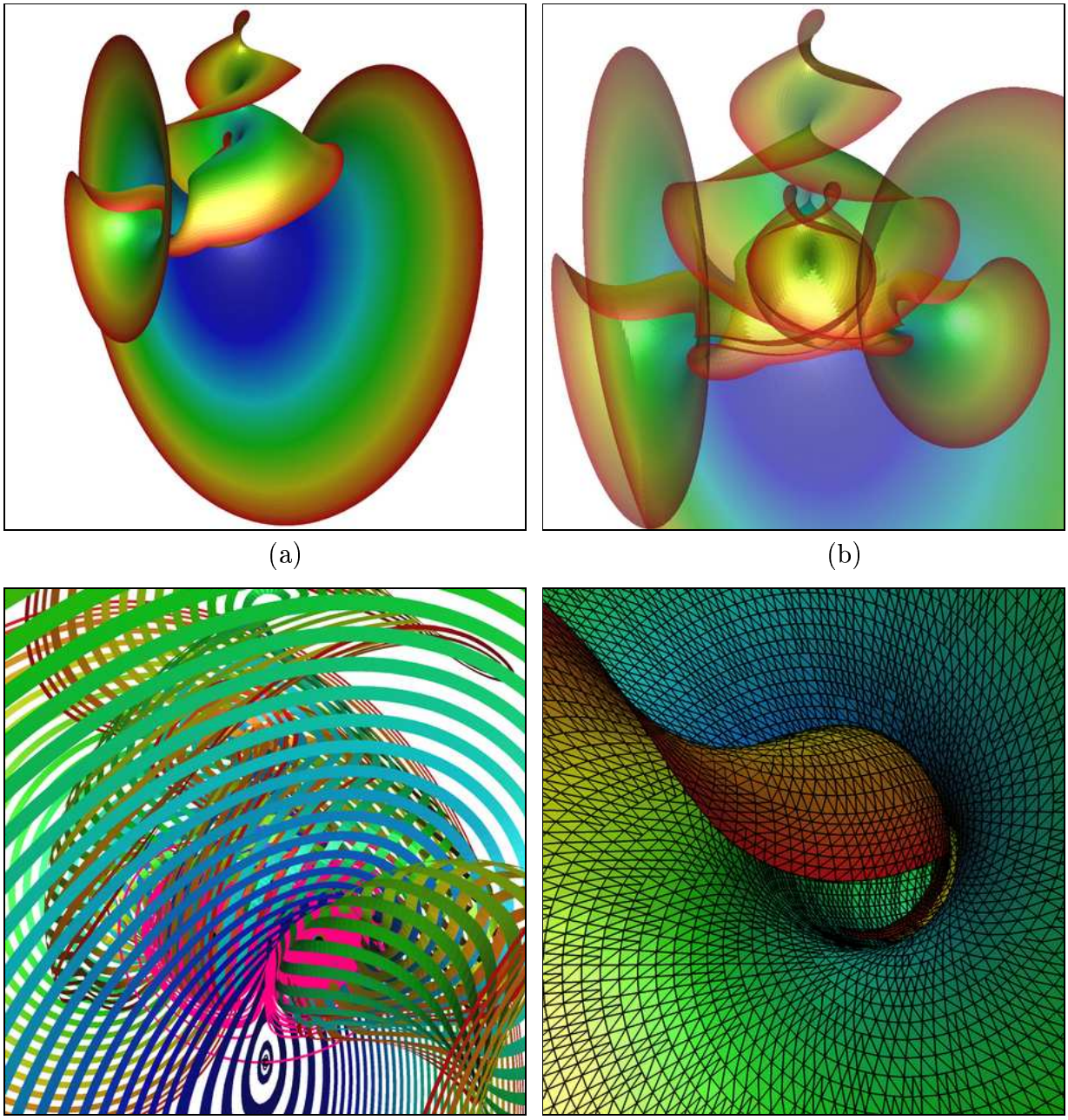

Figure 5: The Lorenz manifold computed with the method of Krauskopf & Osinga up to geodesic distance 154.75. Panel (a) shows the entire manifold, panel (b) an enlargement where the manifold is transparent, panel (c) a further enlargement near the Lorenz attractor (in magenta) where only every second band is shown, and panel (d) the computed mesh when looking into the outer scroll.

an enlargement of the Lorenz manifold where the manifold is now transparent. This brings out the detail of the manifold, in particular, the development of a pair of extra helices

that follow the main helix along the z-axis. Notice that points of the same color are on the same geodesic circle, which shows that points on $W^s(\mathbf{0})$ that are close to the origin in Euclidean distance need not be close to the origin in geodesic distance. Figure 5(c) shows a further enlargement near the Lorenz attractor, which is illustrated in magenta by plotting the unstable manifold $W^u(\mathbf{0})$. In this image only every second band is shown to obtain a see-through effect, showing clearly how the Lorenz manifold 'rolls' into the Lorenz attractor.

Figure 5(d) gives an impression of the computed mesh with an enlargement looking into one of the outer scrolls. Geodesic circles can be seen as spiraling curves (between bands of the same color). The approximate geodesics are the curves that point approximately radially out in the image. They are perpendicular to the geodesic circles, and locations where points were added can be identified as starting points of new approximate geodesics. Notice that the last six bands are closer together. The image illustrates how the distance between geodesic circles is determined by the curvature along geodesics, while the mesh distribution on the geodesic circles is allowed to vary between $\delta_{\mathcal{F}} = 0.67$ and $\Delta_{\mathcal{F}} = 2.0$.

3 BVP continuation of trajectories

It seems very natural to use parametrization (8) for defining a one-parameter family that describes the unstable manifold $W^u(\mathbf{x}_0)$ of a saddle equilibrium $\mathbf{x}_0$ of (1). An approximation to $W^u(\mathbf{x}_0)$ could then be attempted by simple integration of Eq. (1) for a sufficient number of initial conditions that lie on the circle (or ellipse) S_δ of small radius δ in the stable eigenspace $E^u(\mathbf{x}_0)$ centered at $\mathbf{x}_0$. However, as was already explained in Section 1.3, this procedure does not generally produce $W^u(\mathbf{x}_0)$ as a surface. The main problem is to properly space the initial conditions around the circle, so that the result gives a reasonable distribution of the computed trajectories along the stable manifold. This is caused by the fact that the continuation procedure for computing the initial orbit is sensitive to the dependence of the entire orbit on the initial condition.

The method of Doedel uses numerical continuation to address this problem. The step size in the continuation procedure measures the change of the *entire trajectory* (and various parameters), and not just the change in the initial conditions. It is this key property of continuation that generally results in a reasonable distribution of trajectories along the stable manifold.

In this section we only consider the computation of one-parameter families of trajectories, which together describe a two-dimensional (un)stable manifold of a fixed point. Most existing continuation algorithms can handle the computation of such one-dimensional families (also called *solution branches*); see, for example, [5, 16, 17, 31, 48, 52], and [37, Chapter 10]. The continuation method described here was implemented in the continuation package AUTO [14, 15, 18] by specifying the respective driver files.

Continuation algorithms have also been developed for the higher-dimensional case; see, for example, [2, 24]. Hence, this method could be applied, in principle, equally well to compute manifolds of dimension larger than two.

3.1 Pseudo-arclength continuation

Let us begin with a discussion of some basic notions of continuation. Consider the finite-dimensional equation

$$F(X) = 0, \qquad F \ : \ \mathbb{R}^{m+1} \ \rightarrow \ \mathbb{R}^m, \tag{14}$$

where F is assumed to be sufficiently smooth. This equation has one more variable than it has equations. Given a solution X_0, one has, generically, a locally unique solution branch that passes through X_0. To compute a next point, say, X_1, on this branch, one can use Newton's method to solve the extended system

$$\begin{aligned} F(X_1) &= 0, & (15)\\ (X_1 - X_0)^* \dot{X}_0 &= \Delta s. & (16) \end{aligned}$$

Here $\dot{X}_0$ is the unit tangent to the path of solutions at X_0, the symbol * denotes transpose, and Δs is a step size in the continuation procedure. The vector $\dot{X}_0$ is a null vector of the $m \times (m+1)$-dimensional Jacobian matrix $F_X(X_0)$, and it can be computed at little cost [16]. This continuation method is known as Keller's *pseudo-arclength method* [31]. The size of the pseudo-arclength step Δs is normally adapted along the branch, depending, for example, on the convergence history of Newton's method.

The continuation procedure is well posed near a *regular solution* X_0, that is, if the null space of $F_X(X_0)$ is one-dimensional. Namely, in this case the Jacobian of the entire system (15)–(16) at X_0, that is, the $(m+1) \times (m+1)$ matrix

$$\begin{pmatrix} F_X(X_0) \\ \dot{X}_0^* \end{pmatrix} \tag{17}$$

is nonsingular. The Implicit Function Theorem then guarantees that a locally unique solution branch passes through X_0. This branch can be parametrized locally by Δs. Moreover, for Δs sufficiently small, and for sufficiently accurate initial approximation (for example, when taking $X_1^{(0)} = X_0 + \Delta s \dot{X}_0$), Newton's method for solving Eqs. (15)–(16) converges.

3.2 Boundary value problem formulation

When computing a branch of solutions to an ODE of the form (1) one must keep in mind that (1) has infinitely many solutions and boundary or integral constraints must be imposed. Furthermore, the pseudo-arclength constraint (16) is then typically given in functional form; more details can be found in [17]. This means that the possibly unknown total integration time T is embedded in the equations. To this end, the vector field (1) is rescaled so that integration always takes place over the interval $[0,1]$, and the actual integration time T appears as a parameter. Hence, in this context, Eqs. (15)–(16) take the form

$$\begin{aligned} \mathbf{x}_1'(t) &= \hat{f}\big(\mathbf{x}_1(t), \lambda_1\big), & (18)\\ b(\mathbf{x}_1(0), \mathbf{x}_1(1), \lambda_1) &= 0, & (19)\\ \int_0^1 q(\mathbf{x}_1(s), \lambda_1)\, ds &= 0, & (20)\\ \int_0^1 \big(\mathbf{x}_1(\tau) - \mathbf{x}_0(\tau)\big)^* \dot{\mathbf{x}}_0(\tau)\, d\tau + (\lambda_1 - \lambda_0)^* \dot{\lambda}_0 &= \Delta s, & (21) \end{aligned}$$

where the dimension of λ_1 must be chosen consistently with the dimensions of the boundary conditions (19) and the integral constraints (20) in order to ensure a one-dimensional family of solutions. Equations (15)–(16) must be solved for $X_1 = (\mathbf{x}_1(\cdot), \lambda_1)$, given a solution $X_0 = (\mathbf{x}_0(\cdot), \lambda_0)$ and the path tangent $\dot{X}_0 = (\dot{\mathbf{x}}_0(\cdot), \dot{\lambda}_0)$. That is, in a function space setting, Eqs. (18)–(20) correspond to the equation $F(X) = 0$, as in Eq. (14). Note that the dimension $(m+1)$ of $X = (x(\cdot), \lambda)$ may be much larger than the dimension n of the phase space of (1). In particular, λ always contains the parameter T, which may or may not vary during the continuation; see Section 3.3 for specific examples. If $\lambda = T$ then $\hat{f}\big(\mathbf{x}_1(t), \lambda_1\big) = Tf\big(\mathbf{x}_1(t)\big)$.

In each continuation step, Eqs. (18)–(21) are solved by a numerical boundary value algorithm. Here, the package AUTO [14, 15, 18] is used, which uses piecewise polynomial collocation with Gauss-Legendre collocation points (also called *orthogonal collocation*), similar to COLSYS with adaptive mesh selection [3, 6, 49]. In combination with continuation, this allows the numerical solution of 'difficult' orbits. Moreover, for the case of periodic solutions, AUTO determines the characteristic multipliers (or Floquet multipliers) that determine asymptotic stability and bifurcation properties, as a by-product of the decomposition of the Jacobian of the boundary value collocation system [17, 20]; see also [39].

3.3 BVP continuation of the (un)stable manifold of an equilibrium

Consider now the situation that (1) has a saddle equilibrium $\mathbf{x}_0$ with a two-dimensional unstable manifold, meaning that the Jacobian $Df(\mathbf{x}_0)$ has exactly two eigenvalues μ_1 and μ_2 with positive real part. Suppose further that $\mathbf{v}_1$ and $\mathbf{v}_2$ are the associated (generalized) eigenvectors. We are looking for solutions of the system

$$\begin{aligned} \mathbf{x}'(t) &= Tf\big(\mathbf{x}(t)\big), && (22)\\ \mathbf{x}(0) &= \mathbf{x}_0 + \delta(\cos(\theta)\mathbf{v}_1 + \sin(\theta)\mathbf{v}_2), && (23) \end{aligned}$$

which is a combination of Eqs. (18) and (19) with $\lambda = (\theta, T)$. Note that in Eqs. (22)–(23) the continuation equation corresponding to Eq. (21) (or Eq. (16)) has been omitted, even though it is an essential part of the continuation procedure. The continuation equation will also not be written explicitly in subsequent continuation systems.

If the eigenvalues μ_1 and μ_2 are real, then it is advantageous to choose the initial condition on the ellipse that is given by the ratio of the eigenvalues as

$$\mathbf{x}(0) = \mathbf{x}_0 + \delta\left(\cos(\theta)\frac{\mathbf{v}_1}{|\mu_1|} + \sin(\theta)\frac{\mathbf{v}_2}{|\mu_2|}\right). \qquad (24)$$

In other words, in the continuation Eq. (23) is replaced by Eq. (24).

Obvious starting data for the system (22)–(23) consist of a value of θ ($0 \le \theta < 2\pi$), $T = 0$, and $\mathbf{x}(t) = \mathbf{x}_0 + \delta(\cos(\theta)\mathbf{v}_1 + \sin(\theta)\mathbf{v}_2)$, that is, $\mathbf{x}(t)$ is constant. An actual trajectory for a specific value of θ can now be obtained using continuation as well. While this may seem superfluous, it has the added benefit that the output files of this first step in AUTO are then compatible with subsequent continuation steps. In this continuation step, system (22)–(23) is solved for $X = (\mathbf{x}(\cdot), T)$, keeping the angle θ fixed. Here, $T > 0$ for an unstable manifold and $T < 0$ for a stable manifold since then integration is backward (or negative) in time.

Once a single orbit is obtained up to a desired length, defined by a suitable end-point condition, then this orbit is continued numerically as a boundary value problem where the initial condition on the small circle (or ellipse) is now a component of the continuation variable. In this way, the family (8) of such orbits on (part of) the stable manifold $W^u(\mathbf{x}_0)$ is approximated. The simplest way to do this is to fix T in the continuation system (22)–(23) after the first step and allow θ, the angle of the starting point on S_δ to vary freely. It is important to note that θ is not used as the sole continuation parameter. Instead each continuation step is taken in the full continuation variable $X = (\mathbf{x}(\cdot), \theta)$, so that the continuation step size includes variations along the entire orbit. Also, θ is one of the variables solved for in each continuation step and it is not fixed *a priori*.

Instead of keeping T fixed, there are other ways to perform the continuation. For example, one can constrain the end point $\mathbf{x}(1)$ as one wishes. This is done by adding to system (22)–(23) the equation

$$g(\mathbf{x}(1), \theta, T) - \alpha \;=\; 0. \tag{25}$$

Here g is an appropriate functional, chosen to control the end point in a desirable manner, for example, by requiring one coordinate to have a particular fixed value. The continuation variable can now be taken as $X = (\mathbf{x}(\cdot), \theta, T)$, while α is kept fixed.

Another possibility is to impose an integral constraint along the orbit, namely adding to (22)–(23) the equation

$$\int_0^1 h(\mathbf{x}(s), \theta, T)\; ds - L \;=\; 0. \tag{26}$$

Now h is an appropriate functional, chosen to control the orbit in a desirable manner. The continuation variable can again be taken as $X = (\mathbf{x}(\cdot), \theta, T)$, but now keeping L fixed. A particularly useful choice is $h(\mathbf{x}, \lambda, T) = T||f\big(\mathbf{x}, \lambda\big)||$, which results in the total arclength of the orbit being kept fixed during the continuation. Finally, it is entirely possible to use a combination of end-point conditions and integral constraints, but this will not be used here.

3.4 The Lorenz manifold as a family of trajectories

Figure 6 shows an enelargement near the origin of the orbits that were continued on the Lorenz manifold $W^s(\mathbf{0})$ (for negative T). The angle θ is allowed to vary from 0 to 2π, so that the initial condition varies along the ellipse in the middle of the image, which is defined by (24) with $\delta = 5.0$, $\mu_1 = -22.828$ and $\mu_1 = -2.667$. All orbits have the same arclength and the coloring is in terms of the total integration time T along each trajectory. In other words, the coloring gives an indication of the speed of the flow along trajectories, where red is fast and green is slower. The flow is fastest along the strong stable manifold, which is located in the middle of the red region. Note that the disctribution of points is much denser near the top and bottom of the ellipse, that is, near the invariant z-axis, which ensures a good distribution of orbits over the Lorenz manifold $W^s(\mathbf{0})$.

Figure 7(a)–(c) shows the Lorenz manifold $W^s(\mathbf{0})$ covered by 2284 trajectories of arclength 250, where the ellipse of initial conditions is as in Figure 6. The number of mesh points along each trajectory was NTST = 75, with NCOL = 4 collocation points in each mesh interval. Figure 7(a) shows the entire computed part of the Lorenz manifold from

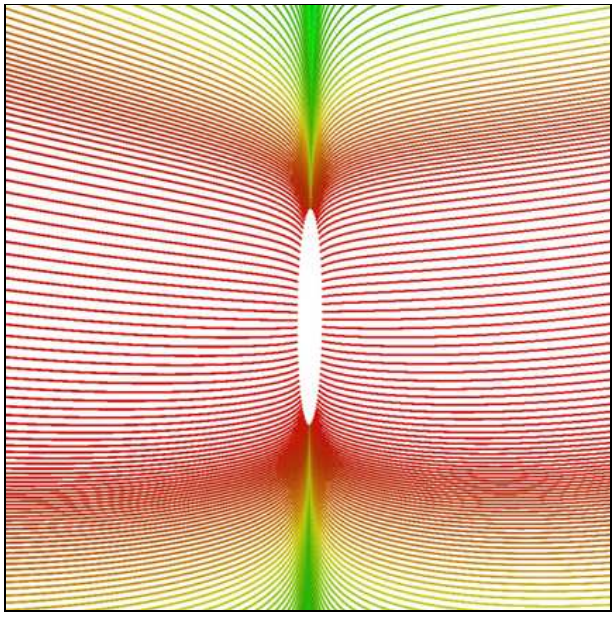

Figure 6: Continued trajectories on $W^s(\mathbf{0})$ near the origin starting from the ellipse (24) with $\delta = 5.0$, $\mu_1 = -22.828$ and $\mu_2 = -2.667$; the coloring is according to integration time T, where red indicates faster and green slower flow.

the common viewpoint. The coloring changes from blue to red according to the mesh point number along a trajectory, which gives an impression of the arclength of trajectories. Figures 7(b) and (c) show enlargements where the coloring shows the total integration time T along trajectories. As in Figure 6, this indicates the speed of the flow; the strong stable manifold is located in the red region of fast flow. In Figure 7(b) every fourth trajectory is rendered as a thin tube. This results in a better sense of depth so that an impression is given of how trajectories lie in phase space to form $W^s(\mathbf{0})$. Figure 7(c) is an enlargement of Figure 7(a) (though with a different color scheme) showing how the manifold forms a scroll.

Figure 7(d) illustrates the flexibility of the method by showing part of the Lorenz manifold computed by numerical continuation of solutions to the boundary value problem (22)–(23) and (25) for the choice $g(\mathbf{x}, \lambda, T) = x$. This results in the x-coordinate of the end point $\mathbf{x}(1)$ being kept fixed during the continuation, and it was set to $x = -25$ in the computation. For an appropriate choice of α, the continuation procedure then naturally leads to non-monotonous variation of θ, thereby allowing the computation of a scroll-like structure on the stable manifold. In Figure 7(d) the origin is the point on the right from which all trajectories emerge.

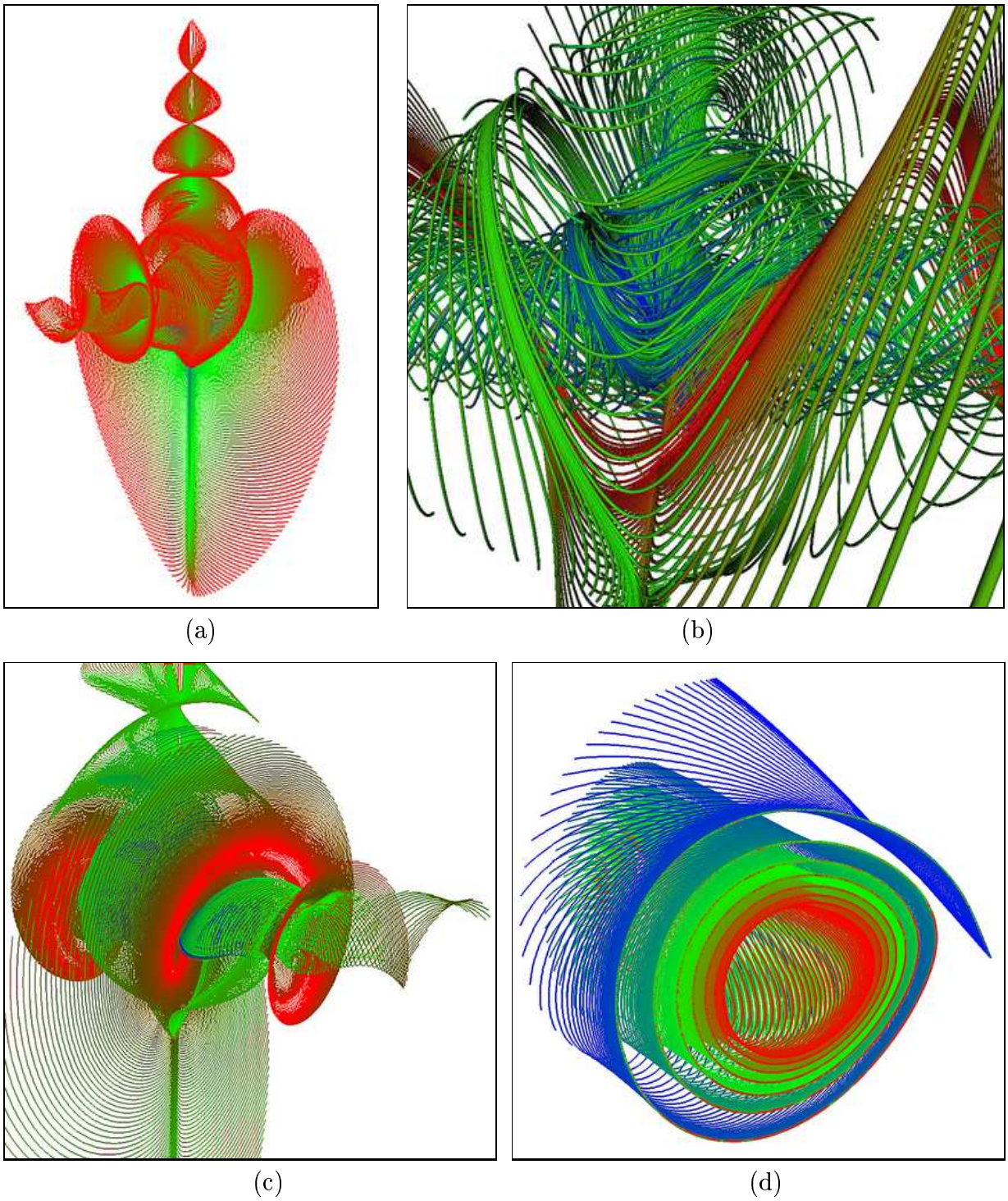

Figure 7: The Lorenz manifold computed with the continuation method of Doedel. Panels (a), (b) and (c) show the manifold where the arclength of the trajectories is fixed at $L = 250$. In panel (a) the coloring indicates the arclength along trajectories and in panels (b) and (c) the coloring is according to the total integration time T of trajectories; the strong stable manifold lies inside the red region. Panels (a) and (c) show all trajectories, while panels (b) shows only every fourth trajectory as a tube. Panel (d) demonstrate that only a part of interest of the stable manifold may be computed, such as a part of the main scroll; this was done by fixing $x = -25$ at the end point of trajectories.

4 Computation of fat trajectories

The method of Henderson [25] computes a compact piece of a k-dimensional invariant manifold by covering it with k-dimensional spherical balls in the tangent space, centered at a set of well-distributed points. This set is found by computing so-called fat trajectories, which are trajectories augmented with tangent and curvature information at each point. The centers of the balls are points on the fat trajectory, and the radius is determined by the curvature.

For the implemented case of computing a two-dimensional unstable manifold $W^u(\mathbf{x}_0)$ of a saddle point of (1), the method starts with a small circle $S_\delta \subset E^u(\mathbf{x}_0)$ and at every step circular disks are added along a fat trajectory with a fixed total arclength (from $\mathbf{x}_0$) of L. Initially all fat trajectories start on S_δ, but at later stages fat trajectories begin at points interpolated where two fat trajectories move too far from each other. The method stops when $W^u(\mathbf{x}_0)$ has been covered up to the prescribed arclength L.

4.1 Fat trajectories on the global stable manifold

The method requires a basis for the tangent space and the curvatures in that basis to construct the disks. As was mentioned in the introduction, invariant manifolds are not defined locally, so that there is no local way of determining the tangent space or curvature for a given a point on the invariant manifold. This information is known at points on the initial curve $S_\delta \subset E^u(\mathbf{x}_0)$, because the tangent to S_δ is known, and if the flow is transverse to the initial curve S_δ, f can be used as the second tangent. The circle S_δ (or possibly an ellipse) may be chosen to be transverse to the flow for sufficiently small δ. The curvature information is obtained using the second derivative tensor.

The tangent and curvature can be 'transported' over $W^u(\mathbf{x}_0)$ by deriving and solving evolution equations for them along a trajectory. To this end, one writes the parametrization (5) in the form

$$x(t,\sigma) = c(\sigma) + \int_0^t f(\mathbf{x}(s,\sigma))ds\,, \tag{27}$$

where $c(\sigma)$ parametrizes S_δ with the one-dimensional parameter σ. Then the tangent space at $\mathbf{x}(t,\sigma)$ is spanned by $\mathbf{x}_\sigma$ and $\mathbf{x}_t = f$, and the curvatures are $\mathbf{x}_{\sigma\sigma}$, $\mathbf{x}_{t\sigma} = f_{\mathbf{x}}\mathbf{x}_\sigma$ and $\mathbf{x}_{tt} = f_{\mathbf{x}}f$. Evolution equations for these quantities can be found by differentiating (27)

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{x} = f, \tag{28}$$

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{x}_\sigma = f_{\mathbf{x}}\mathbf{x}_\sigma, \tag{29}$$

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{x}_{\sigma\sigma} = f_{\mathbf{x}}\mathbf{x}_{\sigma\sigma} + f_{\mathbf{x}\mathbf{x}}\mathbf{x}_\sigma\mathbf{x}_\sigma\,. \tag{30}$$

Note that, even if $\mathbf{x}_\sigma$ is orthogonal to f at the initial point, there is no reason to expect the basis to remain orthogonal. In [25], equations are derived for the evolution of a local parametrization which does remain orthonormal and has minimal change in the basis along the trajectory. (This is analogous to finding Riemannian normal coordinates in gravitation,

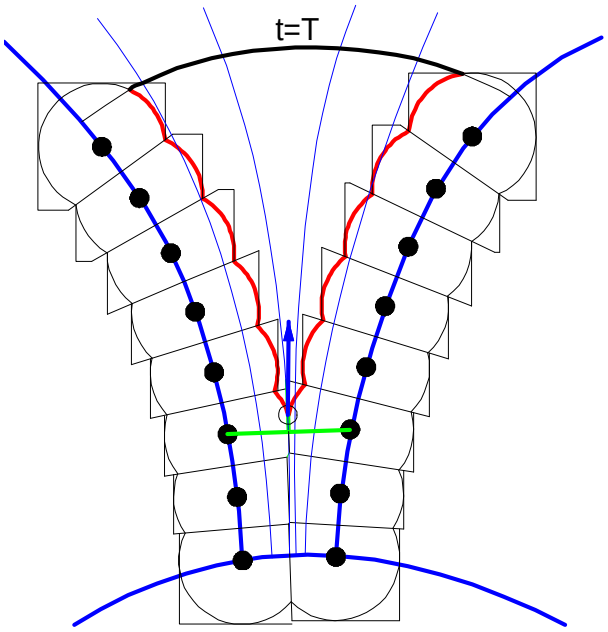

Figure 8: Two adjacent fat trajectories starting from S_δ. A new fat trajectory is started from the point where the two fat trajectories separate. This point can be found by interpolation between two suitable mesh points, which is indicated by the green lines.

where trajectories play the role of geodesics [40].) If the tangents in the local parametrization are $\mathbf{x}_0$ and $\mathbf{x}_1$, they evolve according to

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{x}_0 = f_{\mathbf{x}}\mathbf{x}_0 - \mathbf{x}_0^T f_{\mathbf{x}}\mathbf{x}_0 \ \mathbf{x}_0 - \mathbf{x}_1^T f_{\mathbf{x}}\mathbf{x}_0 \ \mathbf{x}_1 \,, \tag{31}$$

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{x}_1 = f_{\mathbf{x}}\mathbf{x}_1 - \mathbf{x}_0^T f_{\mathbf{x}}\mathbf{x}_1 \ \mathbf{x}_0 - \mathbf{x}_1^T f_{\mathbf{x}}\mathbf{x}_1 \ \mathbf{x}_1 \,. \tag{32}$$

4.2 Interpolation points on the invariant manifold

The method starts with a set of well-distributed points on the initial curve S_δ, which can be found using the algorithm described in [24]. At each such point on S_δ an orthonormal basis for the invariant manifold and second derivatives of the manifold in that basis are computed, and used as initial conditions for finding a set of disks along a fat trajectory. Because trajectories may move apart from each other, these disks will generally not cover $W^u(\mathbf{x}_0)$; see Figure 8. This means that additional fat trajectories must be started at suitable points until $W^u(\mathbf{x}_0)$ is covered. In order to generate a well-spaced set of points on $W^u(\mathbf{x}_0)$, one chooses a starting point from the boundary of the computed part of the manifold.

The method in [24] represents the boundary of the union of disks $\{D_i\}$ using polygons related to the Voronoi regions of the centers of the disks. A disk D_i consists of a center $\mathbf{x}(t_i, \sigma_i)$ (a point on a fat trajectory), the orthonormal basis for the tangent space of the manifold $\mathbf{x}_0(t_i, \sigma_i)$ and $\mathbf{x}_1(t_i, \sigma_i)$, a radius R_i, and polygon P_i. The polygon P_i is represented by a list of vertices in the tangent space and edges joining them (this actually works in arbitrary dimensions). The polygons are constructed in such a way that each edge of P_i

which crosses the boundary of D_i corresponds to a neighboring disk D_j. The situation is sketched in Figure 8.

Suppose that part of $W^u(\mathbf{x}_0)$ is represented this way, and a new disk D_i is to be added. P_i is initially a square centered at the origin with sides $2R_i$, and for each disk D_j which intersects the new disk D_i complementary half spaces are subtracted from P_i and P_j. The projection of D_j into the tangent space at $\mathbf{x}_i$ is approximated by a disk of radius R_j centered at the projection of $\mathbf{x}_j$. If R_i and R_j are small enough so that the distance between the tangent space and the manifold is small (this depends on the curvature of $W^u(\mathbf{x}_0)$), then this is a good approximation. This pair of disks in the tangent space at $\mathbf{x}_i$ defines a line containing the intersection of the circles bounding the disks, and one subtracts from P_i a half space bounded by this line. The same approach is used to update P_j by projecting $\mathbf{x}_i$ into the tangent space at $\mathbf{x}_j$.

With these polygons a point on the boundary of the union can easily be found. Any point on $\delta D_i \cap P_i$ is near the boundary of the union (the distance to the boundary is controlled by the distance between the tangent space and the manifold at the radius). Points on the boundary where two disks meet correspond to points where an edge of P_i crosses δD_i (the point obtained is in the tangent space of the manifold and must be projected onto the manifold).

If one considers the part of the invariant manifold that is not yet covered (that is, the exterior of the union of neighborhoods, $t < T$), one can define something resembling a constrained minimization problem (it lacks a global objective function) which looks for a point in this region that lies furthest back in time under the flow. With a mild assumption about the shape of the region (it must be a topological ball), such a minimal point must exist. It must lie on the boundary of the region at the intersection of two disks. This point is a 'minimum' if the flow vector extended backwards intersects the interior of the edge joining the centers of the intersecting disks. (This is, in fact, Guckenheimer and Vladimirksy's upwinding criterion; see Section 5.) One can easily find candidate points on the boundary from the edges of the polygons, and checking the upwinding criterion is a matter of computing a projection. One can then either interpolate tangents and curvatures from the disks' centers (the method used in the computations shown in Figure 9) or use a homotopy (as Doedel uses in AUTO [14, 15, 18]) to move from the fat trajectory from S_δ through the center of one of the disks to the fat trajectory which starts on S_δ and passes underneath the interpolation point.

This interpolation to find new starting points for fat trajectories completes the algotithm. It computes a covering of the manifold $W^u(\mathbf{x}_0)$ with disks centered at well-spaced points. Provided the disks are sufficiently small compared to the curvature, the algorithm is guaranteed to terminate, and all points lie on trajectories that originate on the initial curve S_δ or at points interpolated from nearby trajectories.

The fat trajectory, with its string of disks and polygons, is integrated until a prespecified total arclength L is reached. This is repeated for all the points on the initial curve. (The total integration time T of fat trajectories varies with the initial condition.)

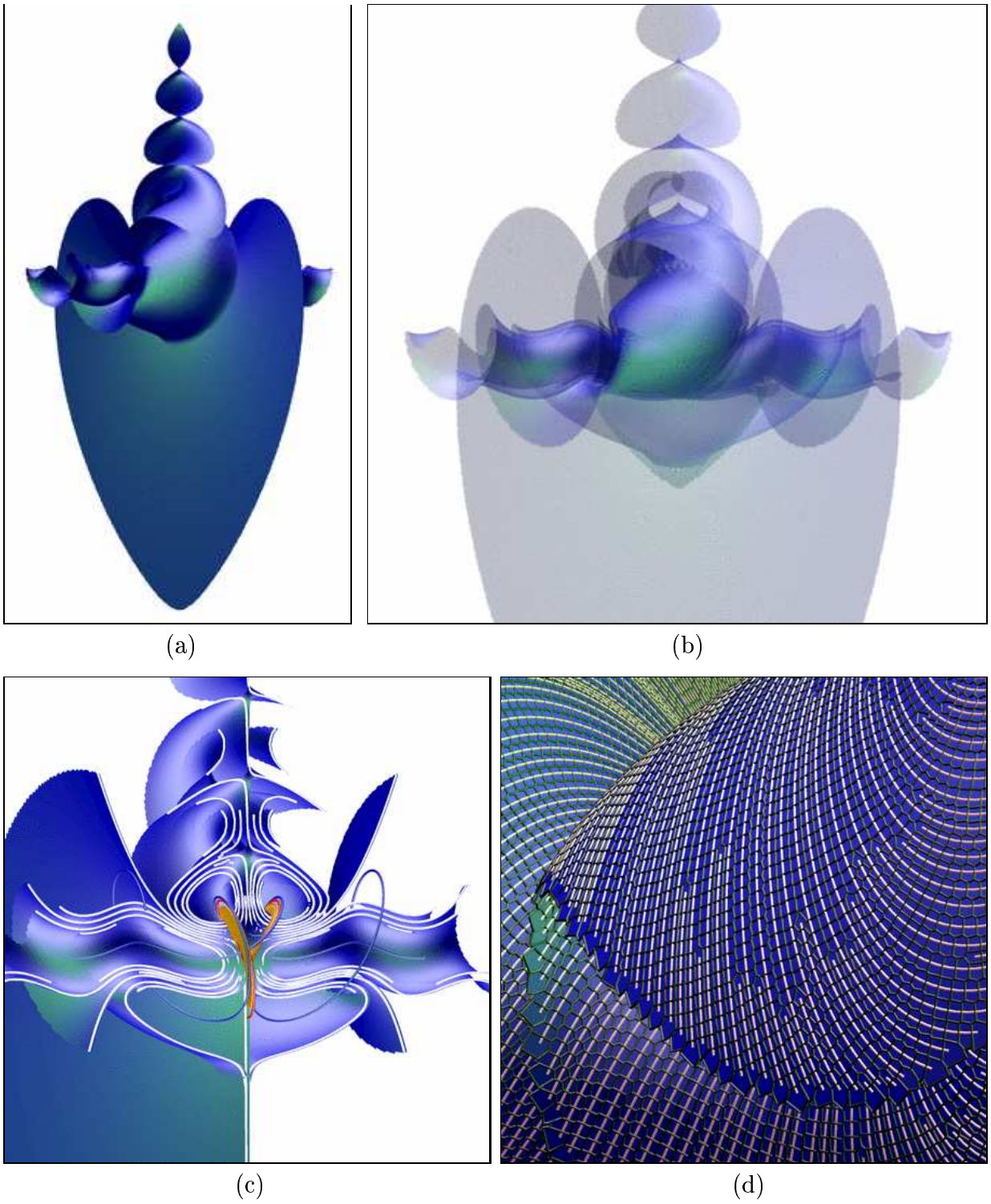

Figure 9: The Lorenz manifold computed with the method of Henderson up to a total trajectory arclength of 240. Panel (a) shows a view of the entire manifold, panel (b) a transparent enlargement near the main scroll, panel (c) shows the part of the manifold for $x < 0$ together with the Lorenz attractor and the one-dimensional stable manifolds of the two other equilibria, and panel (d) gives an impression of the computed mesh.

4.3 The Lorenz manifold covered by fat trajectories

Figure 9 shows the Lorenz manifold $W^s(\mathbf{0})$ computed (using integration backward in time) up to a total trajectory arclength of 240. The step was controlled so that the distance between the tangent space and $W^u(\mathbf{x}_0)$ over each disk was less than 0.5. The scaled time step along trajectories was 0.01 (many more than one time step is taken between sucessive points on a fat trajectory), and no radius is greater than 2.0. The result was a total of 221,210 disks. Figure 9(a) shows the entire computed part of the Lorenz manifold from the common viewpoint. Figure 9(b) shows an enlargement of the Lorenz manifold near the central region where the manifold is now transparent. Notice the different 'sheets' of manifold in the scroll and the extra helices forming around the z-axis. This complicated structure of the Lorenz manifold is further illustrated in Figure 9(c) where only the half of $W^s(\mathbf{0})$ with negative x-coordinate is shown. The intersection curves of the manifold with the plane $\{x = 0\}$ are shown in white. Also shown is the on-dimensional unstable manifold $W^u(\mathbf{0})$ (red curve) accumulating on the the Lorenz attractor (yellow) and the stable manifolds (blue curves) of the other two equilibria.

Figure 9(d) gives an impression of the computed mesh. The fat trajectories are the white curves and they are surrounded by the polygons that make up the Lorenz manifold. Clearly visible are points where new fat trajectories are started from interpolated data. The boundary of the manifold at termination simply consists of the disks that are distance L from $\mathbf{x}_0$ (measured along trajectories).

5 PDE formulation

Another method for approximating invariant manifolds of hyperbolic equilibria was introduced by Guckenheimer and Vladimirsky [22]. Their approach locally models a codimension-one invariant manifold as the graph of a function g satisfying a quasi-linear PDE that expresses the tangency of the vector field f of (1) to the graph of g. The PDE is then discretized in an Eulerian framework and the manifold is approximated by a triangulated mesh. We denote by $\mathcal{M}$ the triangulated approximation of the 'known' part of the manifold. It can be extended by adding simplices at the current polygonal boundary $\partial\mathcal{M}$ in a locally-outward direction in the tangent plane. The discretized version of the PDE is then solved to obtain the correct slope for the newly added simplices. To avoid solving the discretized equations simultaneously, an Ordered Upwind Method (OUM) is used to decouple the system: the causality is ensured by ordering the addition/recomputation of new simplices based on the lengths Λ of the vector field's trajectories.

Two key ideas provide for the method's efficiency:

1. The use of Eulerian discretization ensures that 'geometric stiffness', a high non-uniformity of separation rates for nearby trajectories on different parts of the manifold, does not affect the quality of the produced approximation: new simplices constructed at the current boundary $\partial\mathcal{M}$ are as regular as is compatible with the previously constructed mesh.

2. Since OUM is non-iterative, the PDE-solving step of the method is quite fast.

5.1 Tangency condition

The method is explained here for a two-dimensional manifold $W^u(\mathbf{x}_0)$ of a saddle point $\mathbf{x}_0$ in $\mathbb{R}^3$; see [22] for more details. Let $(\boldsymbol{u}, g(\boldsymbol{u})) = (u_1, u_2, g(u_1, u_2))$ be a local parametrization of the manifold of (1). Then the vector field f should be tangential to the graph of $g(u_1, u_2)$, that is,

$$\left[\frac{\partial}{\partial u_1} g(u_1, u_2), \frac{\partial}{\partial u_2} g(u_1, u_2), -1\right] \cdot f\left(u_1, u_2, g(u_1, u_2)\right) = 0. \tag{33}$$

The above first-order quasi-linear PDE can be solved to 'grow' the manifold in steps, because the Dirichlet boundary condition is specified on the boundary $\partial\mathcal{M}$ of the piece of the manifold computed in previous steps. The initial boundary is chosen by discretizing a small circle or ellipse $S_\delta \subset E^u(\mathbf{x}_0)$ that is transverse to f, so that the vector field is outward pointing everywhere.

Unlike a general quasi-linear PDE, Eq. (33) always has a smooth solution as long as the chosen parameterization remains valid. Thus, switching to local coordinates when solving the PDE avoids checking the continued validity of the parameterization.

In [22] the PDE formulation (33) is extended to approximate two-dimensional manifolds in $\mathbb{R}^n$. A similar characterization can be used for general $k-$dimensional invariant manifolds in $\mathbb{R}^n$, but the current numerical implementation relies on $k = 2$.

The PDE approach for characterizing invariant surfaces goes back to at least the 1960s. The existence and smoothness of solutions for equations equivalent to (33) were the subjects of Sacker's analytical perturbation theory [50] and later served as a basis for several numerical methods, for example, those in [12, 13, 19]. However, all this work was done for the computation of invariant tori. There are two very important distinctions between the PDE methods for tori and the method presented in this section:

1. These prior methods assume the existence of a coordinate system in which the invariant torus is indeed *globally* a graph of a function $g : \mathbb{T}^k \mapsto \mathbb{R}^{n-k}$. This implies the availability of a global mesh, on which the PDE can be solved. For invariant manifolds of hyperbolic equilibria such a mesh is not available a priori and has to be constructed in the process of 'growing' the approximation $\mathcal{M}$.

2. For the invariant tori computations, the solution function g has periodic boundary conditions; hence, the discretized equations are inherently coupled and have to be solved simultaneously.
   For the approximation of $W^u(\mathbf{x}_0)$ all characteristics of the PDE start at the initial boundary (chosen in $E^u(\mathbf{x}_0)$) and run 'outward'. Knowledge of the direction of information flow can be used to decouple the discretized system, resulting in a much faster computational method.

5.2 Eulerian discretization

To enable decoupling of the discretized system, our discretization of Eq. (33) at a 'new' mesh point $\boldsymbol{y}$ has to be 'upwinding', i.e. it should use only previously-computed mesh points straddling $\boldsymbol{y}$'s approximate trajectory. For a two-dimensional invariant manifold in $\mathbb{R}^3$,

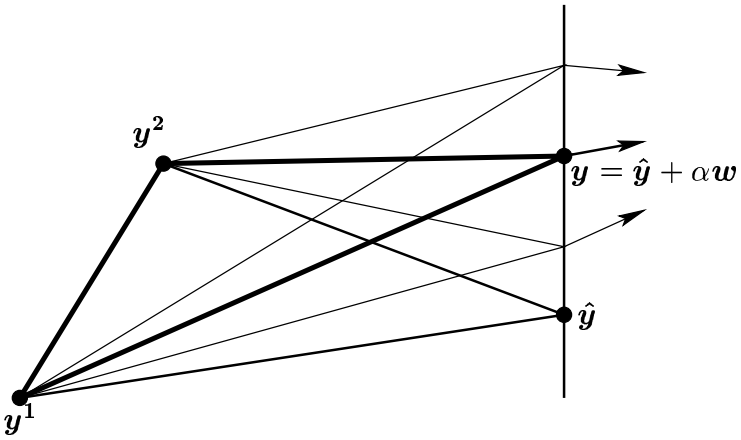

Figure 10: Geometric interpretation of Eq. (34). The search space for finding a suitable α is one-dimensional and given by the normal direction $\boldsymbol{w}$, which corresponds to the codimension of the manifold.

let $G(u_1, u_2)$ be a piecewise-linear numerical approximation of the local parameterization $g(u_1, u_2)$. Consider a simplex $\boldsymbol{y}\boldsymbol{y^1}\boldsymbol{y^2}$, where $\boldsymbol{y^i} = (u_1^i, u_2^i, G(u_1^i, u_2^i)) = (\boldsymbol{u}^i, G(\boldsymbol{u}^i))$ and $\boldsymbol{y} = (u_1, u_2, G(u_1, u_2)) = (\boldsymbol{u}, G(\boldsymbol{u}))$. Suppose that the vertices $\boldsymbol{y^1}$ and $\boldsymbol{y^2}$ are two adjacent mesh points on the discretization of the current manifold boundary, called *AcceptedFront* (thus, $G(\boldsymbol{u}^1)$ and $G(\boldsymbol{u}^2)$ are known and can be used in computing $G(\boldsymbol{u})$). If $\boldsymbol{u}$ is chosen so that the simplex $\boldsymbol{u}\boldsymbol{u}^1\boldsymbol{u}^2$ is well-conditioned, then $\boldsymbol{y} = (\boldsymbol{u}, G(\boldsymbol{u}))$ can be determined from the PDE. Define the unit vectors $\boldsymbol{P_i} = \frac{\boldsymbol{u}-\boldsymbol{u}^i}{\|\boldsymbol{u}-\boldsymbol{u}^i\|}$ and let P be the square invertible matrix with the $\boldsymbol{P_i}$s as its rows. The directional derivative of G in the direction $\boldsymbol{P_i}$ can be computed as $v_i(\boldsymbol{u}) = (G(\boldsymbol{u}) - G(\boldsymbol{u}^i)) / \|\boldsymbol{u} - \boldsymbol{u}^i\|$, for $i = 1, 2$. Therefore, $\nabla g(\boldsymbol{u}) \approx \nabla G(\boldsymbol{u}) = P^{-1}\boldsymbol{v}$, where $\boldsymbol{v} = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}$. This yields the discretized version of Eq. (33) as

$$\left[P^{-1}\boldsymbol{v}(\boldsymbol{u})\right]_1 f_1(\boldsymbol{u}, G(\boldsymbol{u})) \; + \; \left[P^{-1}\boldsymbol{v}(\boldsymbol{u})\right]_2 f_2(\boldsymbol{u}, G(\boldsymbol{u})) \; = \; f_3(\boldsymbol{u}, G(\boldsymbol{u})). \tag{34}$$

This nonlinear equation can be solved for $G(\boldsymbol{u})$ by the Newton-Raphson method or any other robust zero-solver. In addition, it has an especially simple geometric interpretation if the local coordinates are chosen so that $G(\boldsymbol{u}^1) = G(\boldsymbol{u}^2) = 0$. Namely, we reduce the problem to finding the correct 'tilt' of the simplex $\boldsymbol{y}\boldsymbol{y^1}\boldsymbol{y^2}$ with respect to the simplex $\hat{\boldsymbol{y}}\boldsymbol{y^1}\boldsymbol{y^2}$ with $\hat{\boldsymbol{y}} = (\boldsymbol{u}, 0)$. Hence, solving Eq. (34) is equivalent to finding $\alpha \in \mathbb{R}$ such that $f(\hat{\boldsymbol{y}} + \alpha\boldsymbol{w})$ lies in the plane defined by $\boldsymbol{y^1}$, $\boldsymbol{y^2}$, and $\boldsymbol{y} = \hat{\boldsymbol{y}} + \alpha\boldsymbol{w}$, where $\boldsymbol{w}$ is the unit vector normal to $\hat{\boldsymbol{y}}\boldsymbol{y^1}\boldsymbol{y^2}$; see Figure 10. A similar discretization and geometric interpretation can be derived for the general case of $k \geq 2$ and $n \geq 3$ [22].

The described discretization procedure is similar in spirit to an *implicit Euler's method* for solving initial value problems since $\boldsymbol{y^1}$ and $\boldsymbol{y^2}$ are assumed to be known and the vector field is computed at the to-be-determined point $\boldsymbol{y}$. In solving first-order PDEs, a fundamental condition for the numerical stability requires that the mathematical domain of dependence

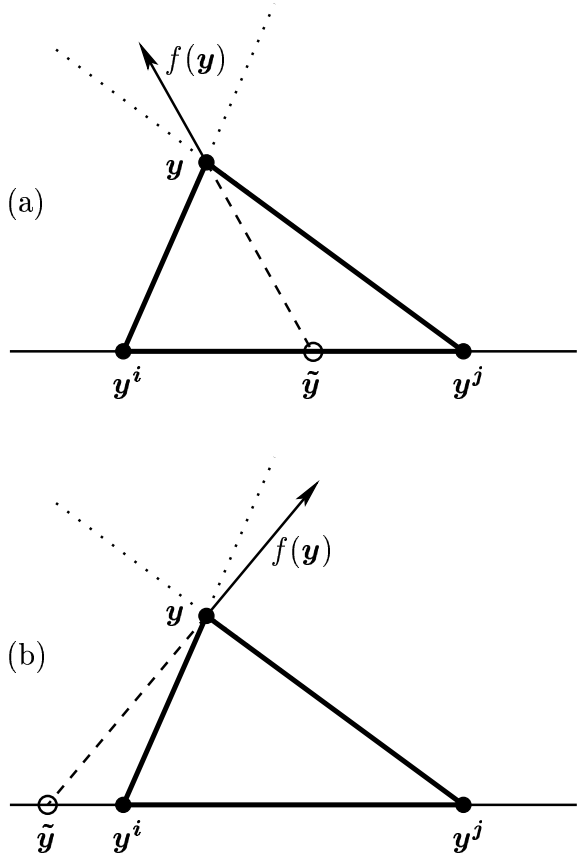

Figure 11: An acceptable (a) and an unacceptable (b) approximation of $f(\boldsymbol{y})$; The range of upwinding directions is shown by dotted lines; the local linear approximation to the trajectory is shown by a dashed line; $\tilde{\boldsymbol{y}}$ is its intersection with the line $\boldsymbol{y^i y^j}$. In the second case the upwinding criterion is not satisfied and the update for $\boldsymbol{y}$ should be computed using another segment of $AcceptedFront$.

should be included in the numerical domain of dependence. Since the characteristics of PDE (33) coincide with the trajectories of the vector field, $G(\boldsymbol{u})$ should be computed using the triangle through which the corresponding (approximate) trajectory runs. Thus, having computed $\boldsymbol{y} = (\boldsymbol{u}, G(\boldsymbol{u}))$ by (34) using two adjacent mesh points $\boldsymbol{y^i}$ and $\boldsymbol{y^j}$, we need to verify an additional *upwinding condition*: the linear approximation to the trajectory of $\boldsymbol{y}$ should intersect the line $\boldsymbol{y^i y^j}$ at a point $\tilde{\boldsymbol{y}} = (\tilde{\boldsymbol{u}}, G(\tilde{\boldsymbol{u}}))$ that lies between $\boldsymbol{y^i}$ and $\boldsymbol{y^j}$; see Figure 11. An equivalent formulation is that $f(\boldsymbol{y})$ should point *from* the newly computed simplex $\boldsymbol{y y^i y^j}$.

Algebraically, if $\boldsymbol{y}$ solves (34), then $f(\boldsymbol{y}) = \beta_1(\boldsymbol{y} - \boldsymbol{y^i}) + \beta_2(\boldsymbol{y} - \boldsymbol{y^j})$; thus, the upwinding criterion above simply requires $\beta_1, \beta_2 \geq 0$. In this case the discretization is locally second-order accurate and the arclength $\Lambda(\boldsymbol{y})$ of the trajectory up to the point $\boldsymbol{y}$ can be approximated as

$$\Lambda(\boldsymbol{y}) \;\approx\; \|\boldsymbol{y} - \tilde{\boldsymbol{y}}\| + \Lambda(\tilde{\boldsymbol{y}}) \;\approx\; \frac{\|f(\boldsymbol{y})\|}{\beta_1 + \beta_2} + \beta_1 \Lambda(\boldsymbol{y^i}) + \beta_2 \Lambda(\boldsymbol{y^j}). \tag{35}$$

Numerical evidence indicates that the resulting method is globally first-order accurate [22].

5.3 Ordered Upwind Method

Ordered Upwind Methods (OUMs) were originally introduced for static Hamilton-Jacobi-Bellman PDEs [51]. In [22] the same idea of *space-marching* for boundary value problems is used to solve Eq. (33). All mesh points are divided into those that are *Accepted*, that is, already fixed as belonging to the approximation $\mathcal{M}$, and those *Considered*, which are in a tentative position adjacent to the current polygonal manifold boundary $\partial\mathcal{M}$, called the *AcceptedFront*. A tentative position can be computed for each *Considered* mesh point $\boldsymbol{y}$ under the assumption that its trajectory intersects $\partial\mathcal{M}$ in some neighborhood $\mathcal{N}(\boldsymbol{y})$ of that point. In other words, $\boldsymbol{y}$ is updated by solving Eq. (34) for a 'virtual simplex' $\boldsymbol{y}\boldsymbol{y}^{\boldsymbol{i}}\boldsymbol{y}^{\boldsymbol{j}}$ such that $\boldsymbol{y}^{\boldsymbol{i}}\boldsymbol{y}^{\boldsymbol{j}} \in \partial\mathcal{M} \bigcap \mathcal{N}(\boldsymbol{y})$ and the upwinding criterion is satisfied. All *Considered* points are sorted based on the approximate trajectory arclengths $\Lambda(\boldsymbol{y})$ defined by (35). The method starts with $\partial\mathcal{M}$ discretizing a small ellipse in $E^u(\boldsymbol{x_0})$. That initial boundary is surrounded by a single 'layer' of *Considered* mesh points (also in $E^u(\boldsymbol{x_0})$).

A typical step of the algorithm consists of picking the *Considered* $\bar{\boldsymbol{y}}$ with the smallest Λ and making it *Accepted*. This operation modifies $\partial\mathcal{M}$ ($\bar{\boldsymbol{y}}$ is included, and the mesh points that are no longer on the boundary are removed) and causes a possible recomputation of all the not-yet-*Accepted* mesh points near $\bar{\boldsymbol{y}}$. If $\boldsymbol{y}^{\boldsymbol{i}}$ is adjacent to $\bar{\boldsymbol{y}}$ and $\boldsymbol{y}^{\boldsymbol{i}}\bar{\boldsymbol{y}}$ is on the boundary, then the mesh is locally extended by adding a new *Considered* mesh point $\boldsymbol{y}$ connected to $\boldsymbol{y}^{\boldsymbol{i}}\bar{\boldsymbol{y}}$ in a tangent plane. To maintain good aspect ratios of newly-created simplices, the current implementation relies on an 'advancing front mesh generation' method similar to [46]. Other local mesh-extension strategies can be implemented similarly to methods in [47] or [24].

The vector field near $\partial\mathcal{M}$ determines the order in which the correct 'tilts' for tentative simplex-patches are computed and the *Considered* mesh points are *Accepted*. This ordering has the effect of reducing the approximation error (since a mesh point $\boldsymbol{y}$ first computed from a relatively far part of $\mathcal{N}(\boldsymbol{y})$ is likely to be recomputed before it gets *Accepted*). The default stopping criterion is to enforce $\Lambda(\bar{\boldsymbol{y}}) \leq L$, so that the algorithm terminates when the maximal approximate arclength L is reached. Other stopping criteria (for example, based on Euclidean or geodesic distance or the maximum number of simplices) can be used as well. Current algorithmic parameters include L, the size N of the neighborhood N, and the desired simplex size Δ. As in the original OUMs, the computational complexity of the algorithm is $O(M \log M)$, where $M = O(L^2/\Delta^2)$ is the total number of mesh points and the $(\log M)$ factor results from the necessity to maintain a sorted list of *Considered* mesh points. A detailed discussion of the algorithmic issues can be found in [22].

5.4 The Lorenz manifold computed with the PDE formulation

Figure 12 shows the Lorenz manifold $W^s(\mathbf{0})$ computed up to an approximate total arclength of $L = 174$. The computation was started from $S_\delta \subset E^s(\mathbf{0})$ with $\delta = 2.0$, $\Delta = 0.6$ and $N = 4\Delta$, which resulted in the total of 271469 mesh points. The coloring shows arclength along trajectories where blue is small and red is large. The manifold was rendered as a two-dimensional surface with MATLAB; other illustrations can be found in [22] and animations at `http://www.math.cornell.edu/~vlad/manifold_movies/lorenz.html`.

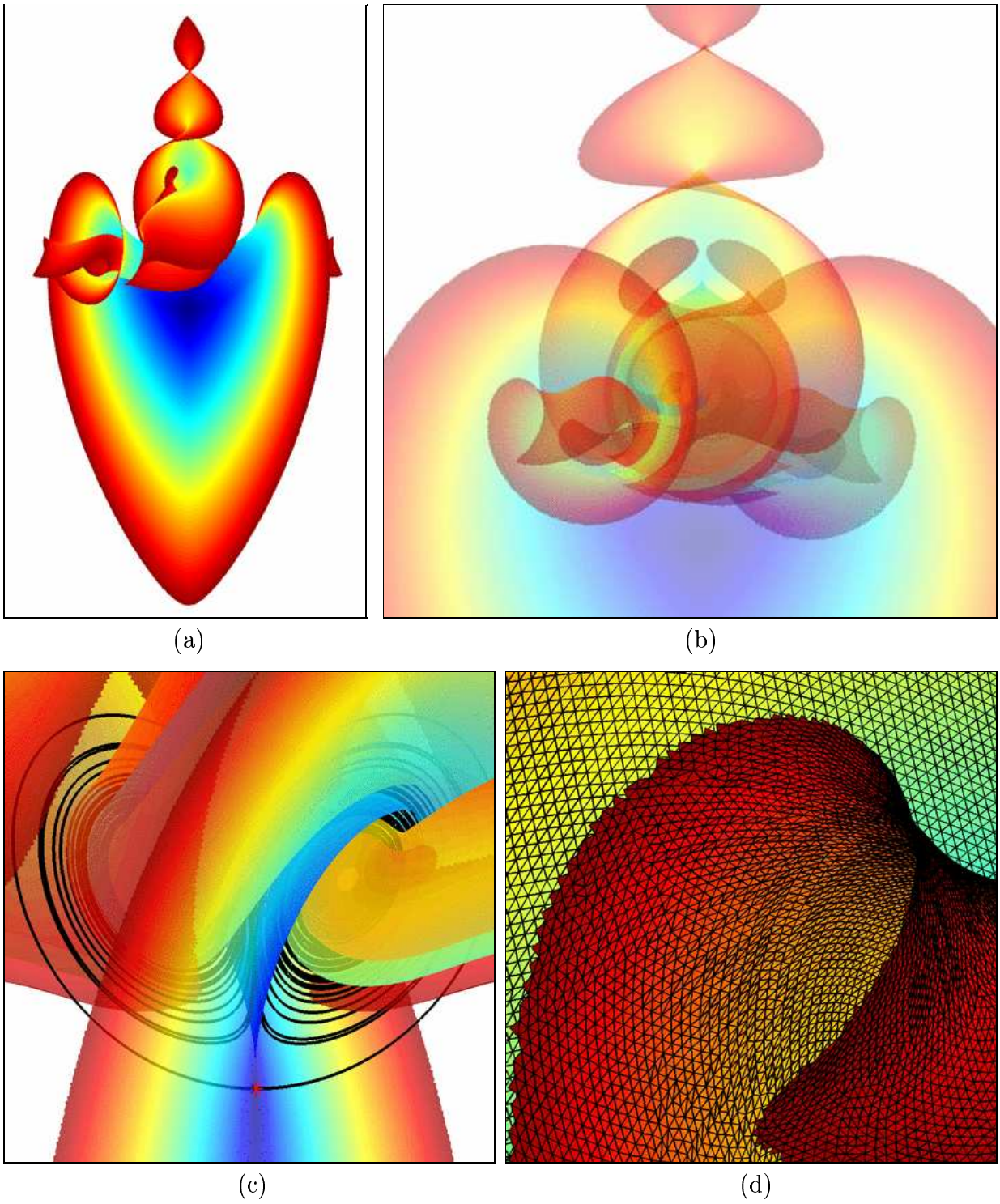

Figure 12: The Lorenz manifold computed with the method of Guckenheimer and Vladimirsky up to a total trajectory arclength of about 174. Panel (a) shows a view of the entire manifold, panel (b) an enlargement near the main scroll where the manifold is shown transparent, panel (c) shows how the manifold interacts with the Lorenz attractor, and panel (d) gives an impression of the computed mesh.

Figure 12(a) shows the entire computed part of the Lorenz manifold from the common viewpoint. Figure 12(b) is an enlargement near the central scrolls where the manifold is now shown transparent. Clearly visible are two secondary spirals forming near the positive z-axis. The coloring is such that points of the same color are equally far away from the origin in arclength along trajectories. Figure 12(c) is a further enlargement near the unstable manifold $W^u(\mathbf{0})$ accumulating on the Lorenz attractor. This clearly shows how the Lorenz manifold 'rolls' into both wings of the Lorenz attractor, creating different sheets that do not actually intersect the shown trajectories representing the unstable manifold $W^s(\mathbf{0})$.

Figure 12(d) gives an enlarged impression of the computed mesh looking into one of the outer scrolls. The simplices of the mesh are sufficiently uniform in spite of the complicated geometry of the manifold they represent. The red boundary of the computed manifold is not a smooth curve, because it is formed simply by the last simplices that were added locally.

6 Box covering

In contrast to the techniques described so far, the method of Dellnitz and Hohmann [7, 8] presented in this section approximates invariant manifolds by objects of the same dimension as the underlying phase space. It first produces an *outer covering* of a local unstable manifold by a finite collection of sets. This covering is then 'grown' in order to cover larger parts of the manifold analogously to what is described in Sections 2 and 5. In combination with set-oriented multilevel techniques for the computation of invariant sets, such as periodic orbits, attractors and general chain recurrent sets, the technique allows, in principle, for the computation of manifolds of arbitrary dimension, where the numerical effort is essentially determined by the dimension of the manifold. In combination with rigorous techniques for the implementation of this approach, it is possible to compute *rigorous coverings* of the considered object. For a more detailed exposition of the general method see [7, 8, 10, 11]. The algorithm is implemented in the software package GAIO [10].

6.1 The box covering algorithm

The box covering algorithm applies to a discrete-time dynamical system, that is, to a diffeomorphism D. In the context of approximating global manifolds, it can compute the unstable manifold of an (unstable) invariant set of D in a compact region of interest Q. In this section, we explain how this method can be used for the computation of a two-dimensional (un)stable manifold of a saddle $\mathbf{x}_0$ in $\mathbb{R}^3$.

Here, the diffeomorphism $D : \mathbb{R}^3 \to \mathbb{R}^3$ is given by the time-τ map of the vector field (1). For an unstable manifold $\tau > 0$, while for a stable manifold $\tau < 0$ to account for reversing time. Numerically, the map D may be realized by classical one-step integration schemes. Since the algorithm involves integration over short time intervals only, typically the requirements in terms of accuracy or preservation of structures of the underlying vector field f are rather mild. The diffeomorphism D then has a hyperbolic saddle fixed point $\bar{\mathbf{x}} = \mathbf{x}_0$ and, because of the reversal of time, $\bar{\mathbf{x}}$ has a two-dimensional unstable manifold $W^u(\bar{\mathbf{x}})$, which is identical to the (un)stable manifold of $\mathbf{x}_0$.

The idea of the algorithm is as follows. Imagine a finite partition $\mathcal{P}$ of Q. The method

first finds a (small) collection $\mathcal{C}_0 \subset \mathcal{P}$ that covers the local unstable manifold $W^u_{\mathrm{loc}}(\bar{\mathbf{x}})$. This local covering of $W^u(\bar{\mathbf{x}})$ is extended in steps, where in each step the sets in the current collection $\mathcal{C}_k$ are mapped forward under D. All sets in $\mathcal{P}$ that have an intersection with the images of $\mathcal{C}_k$ are added to the current collection of sets, yielding $\mathcal{C}_{k+1}$.

More formally, let $\mathcal{P}_0, \mathcal{P}_1, \ldots$ be a nested sequence of successively finer partitions of Q: We take $\mathcal{P}_0 = \{Q\}$ and each element $P \in \mathcal{P}_{\ell+1}$ is contained in an element $P' \in \mathcal{P}_\ell$ and $\operatorname{diam}(P) \leq \gamma \operatorname{diam}(P')$ for some fixed number $0 < \gamma < 1$.

The algorithm consists of two main steps:

1. *Initialization:* Compute an initial covering $\mathcal{C}_0^{(k)} \subset \mathcal{P}_{\ell+k}$ of the local unstable manifold $E^u(\bar{\mathbf{x}})$ of $\bar{\mathbf{x}}$. This can be achieved by applying a subdivision algorithm for the computation of relative global attractors to the element $P \in \mathcal{P}_\ell$ containing $\bar{\mathbf{x}}$ for some suitable ℓ; see [7].

2. *Growth:* From the collection $\mathcal{C}_j^{(k)}$, $j = 0, 1, \ldots$, the next collection is obtained by setting

$$\mathcal{C}_{j+1}^{(k)} = \{P \in \mathcal{P}_{\ell+k} : f(Q) \cap P \neq \emptyset \text{ for some set } Q \in \mathcal{C}_j^{(k)}\}.$$

   This is repeated until no more sets are added to the current collection, that is, until $\mathcal{C}_j^{(k)} = \mathcal{C}_{j+1}^{(k)}$.

We can show that this method converges to a certain subset of $W^u(\bar{\mathbf{x}})$ in Q. Namely, let $W_0 = W^u_{\mathrm{loc}}(\bar{\mathbf{x}}) \cap P$, where P is the element in $\mathcal{P}_\ell$ containing $\bar{\mathbf{x}}$ and define

$$W_{j+1} = D(W_j) \cap Q, \quad j = 0, 1, 2, \ldots.$$

Then we have the following convergence result (see [7]):

1. the sets $C_j^{(k)}$ are coverings of W_j for all $j, k = 0, 1, \ldots$;

2. for fixed j and $k \to \infty$, the covering $C_j^{(k)}$ converges to W_j in Hausdorff distance.

In general, one cannot guarantee that the algorithm leads to an approximation of the entire set $W^u(\bar{\mathbf{x}}) \cap Q$. This is due to the fact that parts of $W^u(\bar{\mathbf{x}})$ that do not lie in Q may map into Q. In this case, the method will indeed not cover all of $W^u(\bar{\mathbf{x}}) \cap Q$.

Under certain hyperbolicity assumptions on $W^u(\bar{\mathbf{x}})$ it is possible to obtain statements about the speed of convergence in terms of how the Hausdorff distance between the covering and the approximated subset of $W^u(\bar{\mathbf{x}})$ depends on the diameter of the sets in the covering collection; see [11] for details.

6.2 Realization of the method

The efficiency of the growth part of the algorithm significantly depends on the realization of the collections $\mathcal{P}_\ell$. In the implementation the $\mathcal{P}_\ell$ are partitions of Q into *boxes*

$$B(c, r) = \left\{y \in \mathbb{R}^d : |y_i - c_i| \leq r_i \text{ for } i = 1, \ldots, d\right\},$$

where $c, r \in \mathbb{R}^d$, $r_i > 0$, are the center and the sizes of the box $B(c, r)$, respectively. Moreover, only partitions are used that result from bisecting the initial box Q repeatedly, where in this

process of bisecting the relevant coordinate direction is changed systematically (typically, the bisected coordinate direction is varied cyclically).

Starting with $\mathcal{P}_0 = \{Q\}$, this process yields a sequence $\mathcal{P}_\ell$ of partitions of Q, that can efficiently be stored in a binary tree. Note that it is easy to store arbitrary subsets of the full partition $\mathcal{P}_\ell$ just by storing the corresponding part of the tree. In fact, in the initialization of the algorithm one starts with a single box on a given level ℓ, so that the stored tree consists of a single leaf. Whenever sets are added to the current collection, the corresponding paths are added to the tree. Figure 13 illustrates the first three growth steps for the computation of a covering of the Lorenz manifold on level 18 of the tree (all other parameters are as described in Section 6.3 below). The yellow box in Figure 13(a) was created in the initialization step and then grown in one step to obtain the blue boxes. Panels (b) and (c) show two further growth steps, where the covering of the previous step is again shown in yellow.

The hierarchical storage scheme has another crucial computational advantage in that it is easier to decide which boxes are 'hit' by mapping the boxes that were added in the previous step of the continuation algorithm. Namely, for each of these boxes $B \in \mathcal{P}_{\ell+k}$ one needs to compute the set $\mathcal{F}(B) = \{B' \in \mathcal{P}_{\ell+k} \mid D(B) \cap B' \neq \emptyset\}$. Since B contains an uncountable number of points, this problem must be discretized. The obvious approach is to choose a finite set $\mathcal{T}$ of *test points* in B and to approximate $\mathcal{F}(B)$ by $\tilde{\mathcal{F}}(B) = \{B' \in \mathcal{P}_{\ell+k} \mid D(\mathcal{T}) \cap B' \neq \emptyset\}$. Using the tree structure, the determination of the box that contains the image of a test point can be accomplished with a complexity that only depends logarithmically on the number of boxes in $\mathcal{P}_\ell$ [8].

6.3 Box covering of the Lorenz manifold

Figure 14 shows a box covering of the Lorenz manifold $W^s(\mathbf{0})$. For the computation the time-τ map of the Lorenz system (2) was considered with $\tau = -0.1$. This map is realized by the classical Runge-Kutta scheme of fourth order with a fixed step size of -0.01. The region of interest Q is a box with sizes $(70, 70, 70)$ and center $(10^{-1}, 10^{-1}, 10^{-1})$; this offset centering is for a practical reason: it avoids having the origin on the edge of a box. Level $\ell = 27$ of the tree was used and 16 growth steps were performed, starting from a single box containing the origin (i.e. $k = 0$). In each growth step, an equidistant grid of 125 test points in each box was mapped forward. The resulting object contains more than 4 million boxes.

Figure 14(a) shows the entire computed part of the Lorenz manifold from the common viewpoint in a transparent rendering. A non-transparent rendering is shown in Figure 14(c). Figure 14(b) shows an enlargement near the central region. Because the methods is using the time-τ map of the Lorenz system (2), the Lorenz manifold first grows initially mainly in the direction of the strong unstable direction until the boundary of the box of interest is reached. This can be seen nicely in Figure 13. Later steps of the growth process then start to build up the other part of the manifold, resulting in the image in Figure 14(a) and (c). As a result, the helical part of the manifold near the positive z-axis is less pronounced.

The further enlargement near the scroll of the manifold in Figure 14(d) gives a local impression of the box covering. Notice that covering of the manifold has a thickness of several box diameters at the end of the scroll.

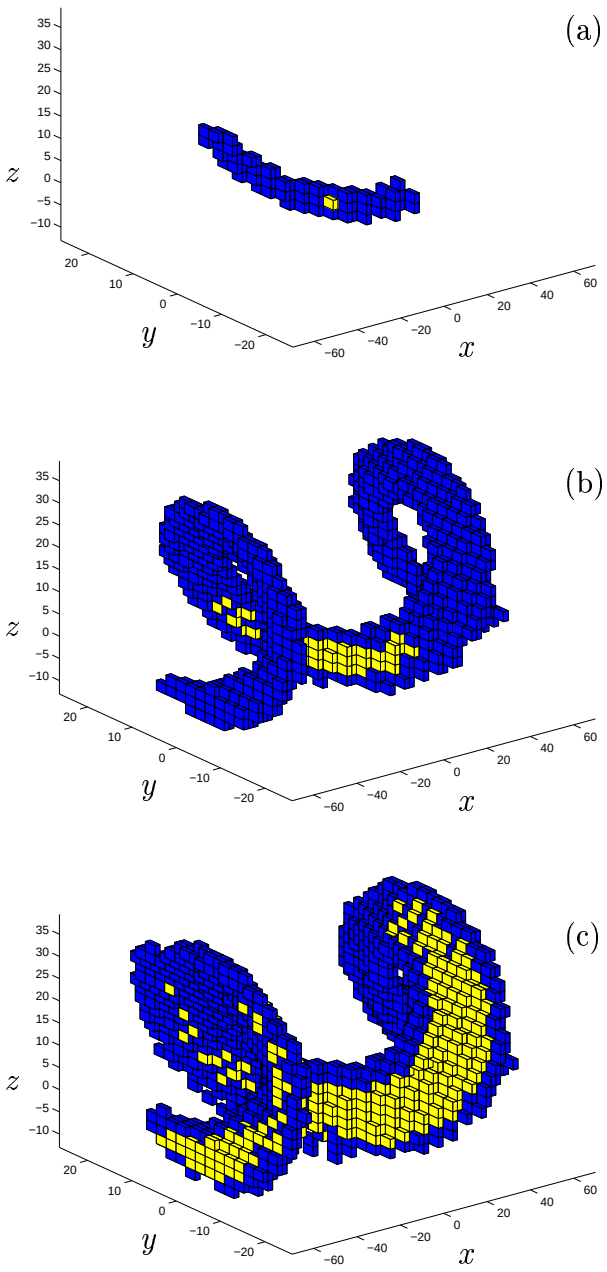

Figure 13: Coverings of the Lorenz manifold during the first three growth steps are shown in panels (a)–(c), where the covering of the previous step (the initialization box in the case of (a)) is shown in yellow.

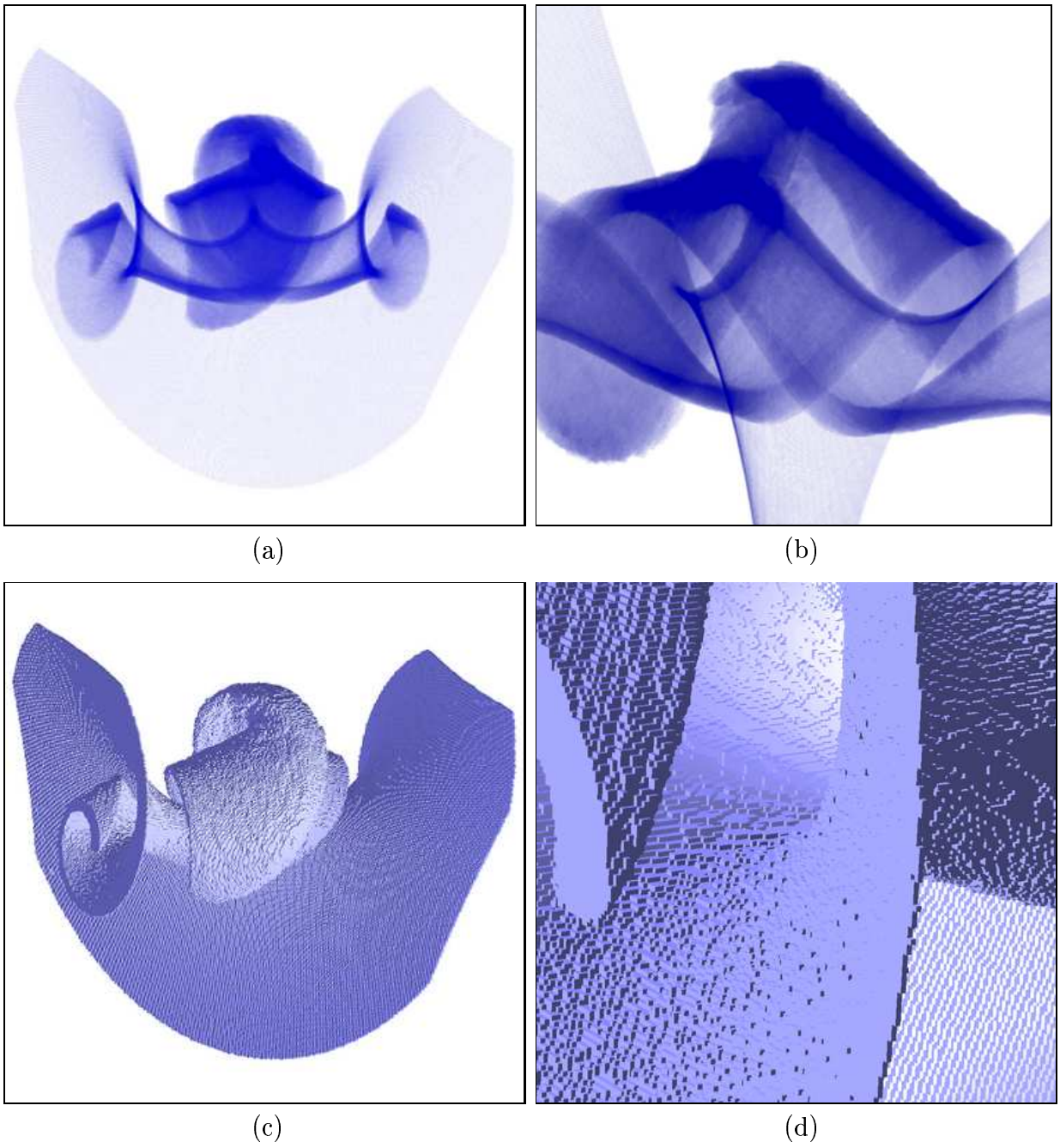

Figure 14: The Lorenz manifold computed with the box covering method of Dellnitz and Hohmann. In panels (a) and (b) the manifold is rendered transparently, while panels (c) and (d) show the box covering in more detail. The common viewpoint is chosen in panels (a) and (c). Panel (b) shows an enlargement near the z-axis, and panel (d) gives a closer look at the computed boxes.

7 Discussion

After a recent flurry of research activity, several complementary methods are available today to compute global (un)stable manifolds in applications. While these methods are still somewhat under development and testing, we hope that this survey will encourage the reader to consider computing such global objects in systems arising in applications.

Each of the methods presented in the previous sections is based on a particular point of view of characterizing a global (un)stable manifold. Common to all is the idea that the manifold must be 'grown' from local information near the saddle point, and the difference is in how this is done. The choice of method will generally depend on the application one has in mind and on the particular questions one wants to answer. This discussion is intended to give an indication of the specific properties of the different approaches.

Approximation by geodesic level sets. The method by Krauskopf and Osinga [34, 35] is presently implemented for two-dimensional manifolds of saddle points and saddle periodic orbits in a phase space of arbitrary dimension; see also [42, 43]. This implementation approximates the manifold linearly between mesh points, while the boundary value problems (9)–(10) are solved by single shooting. It would be possible to use higher order interpolation between mesh points and collocation for solving the boundary value problems. The method produces a very regular mesh that consists of (approximate) geodesic circles and approximate geodesics. This means that the manifold is rendered as a geometric object, independently of the dynamics on it. The mesh is, in fact, constructed so regularly that it can be interpreted as a crochet pattern. This allows one to produce a real-life model of the Lorenz manifold; see [45] for details. During a computation the interpolation error is controlled by prescribed mesh quality parameters, so that the correctness of the method can be proved; see [35] for details.

The price one has to pay for obtaining a guaranteed 'geometric mesh' is that one needs to set up and continue a boundary value problem for each new mesh point. This makes the method more expensive compared to other methods. With the non-optimized present implementation and the accuracy parameters as in Section 2.3, computing the Lorenz manifold up to geodesic distance 140 takes about 10 minutes, while the larger image in Figure 5 with 69,900 mesh points took 40 minutes and 47 seconds on an 800MHz Pentium III machine with 256Mb RAM.

Because it is based on the geodesic parametrization (6), the method works as long as the geodesic level sets of this parametrization remain smooth circles. While this is not an obstruction for computing the Lorenz manifold, there are examples where the computation stops when a geodesic circle seizes to be smooth; see [35]. Furthermore, the method stops when it encounters an equilibrium or a periodic orbit on (the closure of) the (un)stable manifold.

An implementation for global (un)stable manifolds of dimension three would already be quite challenging. First of all, geodesic level sets are spheres in this case, on which one needs to compute a regular mesh. Secondly, the method would require multi-parameter continuation to continue the boundary value problems (9)–(10).

BVP continuation of trajectories. The method by Doedel is arguably the most straightforward one. The continuation calculations can be carried out using the standard boundary value continuation capabilities of AUTO. This means that all that is required are rather standard AUTO equations and parameter files. The orbits that make up the manifold are computed very accurately, due to the high accuracy of the orthogonal collocation method, which is superconvergent for the solution at the mesh points and for scalar variables. Furthermore, the boundary value continuation algorithms in AUTO, written in the f77 or C programming language, are rather efficient, so that the calculations can generally be done in relatively little computer time. For example, the Lorenz manifold in Figure 7 was computed on a ?? MHz Linux laptop in ?? seconds. (SEBIUS: please provide info.)

The method is very flexible in that it allows for different boundary conditions at the endpoint of a trajectory. This means that one can compute only a part of interest of the manifold, as was illustrated in Figure 7(d). However, the manifold cannot be 'grown', so that the continuation must be repeated if a larger part of the manifold is desired.

While visualizing or even animating the computed trajectories gives much insight into the geometry of the manifold, it would require substantial post-processing to produce a nice mesh representation of the manifold as a two-dimensional object. In particular, the density of the orbits may be high in areas where the further evolution of the trajectories depends sensitively on the current state. For example, in Figure 7(a) and (c) the density of the orbits is high along a curve in the direction of the z-axis, that is, the direction of the weakly stable eigenvector.

Computation of fat trajectories. While also essentially computing trajectories, the method of Henderson [25] does produce a nice mesh representation by 'fattening' the trajectories with a string of polygonal patches. The method tends to minimize the need for interpolation. When interpolation is needed there is a guarantee that appropriate points exist, and at those points information is available which allows higher order interpolation or the generation of an interpolating trajectory. The method is presently implemented for computing two-dimensional manifolds. It would be possible to use it for computing higher-dimensional manifolds, because updating the computed boundary works in all dimensions [24]. However, solving the minimization problem and interpolation would have to be generalized. (MIKE: please provide status of the implementation.)

The computation of a fat trajectory is more expensive than straighforward integration, because it adds equations for the evolution of the tangent space and curvatures. However, the implementation of updating the computed boundary is quite efficient; see also [24]. The overall algorithm is relatively fast. For example, the Lorenz manifold in Figure 9 was computed on a ?? in about ?? minutes. (MIKE: please provide info.)

Finally, the algorithm may encounter a geometric problem. It must be able to distinguish between mesh points on different sheets of the invariant manifold, for example, where a trajectory returns close to itself. This can be done by checking the values of t and σ at the centers of the disks, but it demands sufficiently small disks so that those quantities vary only little across each disk. This requirement may results in many more mesh points being computed than is necessary to obtain a geometrically smooth manifold. This geometric problem occurs when trajectories spiral tightly, as is the case, for example, on the unstable manifolds of the two equilibria on the wings of the Lorenz attractor.

PDE formulation. The PDE approach by Guckenheimer and Vladimirsky [22] leads to a very efficient numerical method for computing a mesh representation of a global (un)stable manifold. The computational cost of this method is largely independent of the geometric stiffness present in the system. For example, the Lorenz manifold in Figure 12 was computed in under 90 seconds on a Pentium III 850 MHz processor with 256Mb RAM.

The constructed approximation $\mathcal{M}$ is 'causal', that is, it contains approximate trajectories for all of the mesh points on $\partial\mathcal{M}$. As a result, the method is not restricted to manifolds where the level curves of the geodesic distance remain smooth. In particular, the method can be used for approximating manifolds containing homoclinic and heteroclinic trajectories; see [22] for examples. Another advantage of the PDE method is that it is readily adaptable to a continuation framework. When a vector field is studied under parameter variation, an approximation of the manifold for one parameter value can be used to build a global parametrization for nearby parameter values. This reduces the cost of locally extending the mesh near $\partial\mathcal{M}$ at every step. (ALEXANDER: has this continuation approach actually been implemented/tried? If not, i would prefer it at the end of the section as a possible extension.)

The computational cost of adding each simplex is proportional to the codimension $(n - k)$ of the manifold. When approximating manifolds of high codimension, this is clearly a disadvantage compared to other methods for which this cost is proportional to the dimension of the manifold k. A second limitation of the method is that the constructed approximation is globally only first-order accurate, in contrast with, for example, the second-order accuracy of computing fat trajectories.

The current implementation of the PDE approach works for two-dimensional manifolds in a phase space of arbitrary dimension. An adaptive implementation for $k \geq 2$ will have to employ a robust algorithm for higher-dimensional mesh extension, which remains a challenge.

Box covering. The box covering algorithm of Delnitz and Hohmann [7, 8, 10, 11] constructs a covering of (part of) the global invariant manifold. This covering consists of a collection of small boxes. The method is formulated for discrete time systems, and differential equations can be handled by considering a corresponding time-τ-map. It allows for the computation of (un)stable manifolds of arbitrary invariant sets. It is possible (and implemented in GAIO) to compute manifolds of arbitrary dimension. The 'thickness' of the covering depends on the contraction rate transverse to the manifold. The stronger the contraction, the fewer 'box-layers' along the manifold will be produced. In particular, the algorithm needs to be modified in order to apply it to Hamiltonian systems [30].

The key implementational issue, namely how to compute the image of a given box, is typically discretized by mapping a (finite) set of test points in each box. Evidently, depending on the properties of the underlying map, the choice of these points determines the quality of the resulting covering. Using too few points may lead to missing boxes, while using too many slows down the computation. There exist strategies for a near-optimal choice of these points. In the case that Lipschitz estimates of the dynamical system are available, one may compute rigorous coverings. In this case, it can be ensured that the manifold is contained inside the union of the sets in the constructed covering [10]. (OLIVER: right ref?)

The overall computational cost is quite high when good resolution, that is, many boxes are required. For example, the Lorenz manifold in Figure 14 of more that 4 million boxes took ?? on a ?? machine (OLIVER: please check!) Since the numerical cost depends on

the dimension of the manifold, for manifolds of dimension larger than two it may only be feasible to compute rather coarse approximations.

Acknowledgements

The authors thank Mike Jolly for providing the image in Figure 3 of the Lorenz manifold computed with the method in [27] and seen from the common viewpoint.

References

[1] Abraham, R. H. & Shaw, C. D. [1982–1985] *Dynamics — the geometry of behavior, Part three: global behavior* (Aerial Press, Santa Cruz).

[2] Allgower, E. L. & Georg, K. [1996] "Numerical path following," *Handbook of Numerical Analysis* Vol. 5, eds. Ciarlet, P. G. & Lions, J. L. (North Holland Publishing), pp. 3–207.

[3] Ascher, U. M., Mattheij, R. M. M. & Russell, R. D. [1995] *Numerical Solution of Boundary Value Problems for Ordinary Differential Equations.* (SIAM).

[4] Back, A., Guckenheimer, J. , Myers, M.R., Wicklin, F.J. & Worfolk, P. A. [1992] "DsTool: Computer assisted exploration of dynamical systems," newblock *Notices Amer. Math. Soc.* **39** 303.

[5] Beyn, W.-J., Champneys, A., Doedel, E. J., Govaerts, W., Sandstede, B. & Kuznetov, Yu. A. [2002] "Numerical continuation and computation of normal forms," *Handbook of Dynamical Systems* Vol. 2m ed. Fiedler, B. (Elsevier Science), pp. 149–219.

[6] De Boor, C. & Swartz, B. [1973] "Collocation at Gaussian points," *SIAM J. Numer. Anal.* **10**, 582–606.

[7] Dellnitz, M. & Hohmann, A. [1996] "The computation of unstable manifolds using subdivision and continuation," *Nonlinear Dynamical Systems and Chaos PNLDE* 19, eds. Broer, H. W., Van Gils, S. A., Hoveijn, I. & Takens, F. (Birkhäuser, Basel), pp. 449–459.

[8] Dellnitz, M. & Hohmann, A. [1997] "A subdivision algorithm for the computation of unstable manifolds and global attractors," *Numer. Math.* **75**, 293–317.

[9] Dellnitz, M., Hohmann, A., Junge, O. & Rumpf, M. [1997] "Exploring invariant sets and invariant measures," *Chaos* **7**(2), 221–228.

[10] Dellnitz, M., Froyland, G. & Junge, O. [2001] "The algorithms behind GAIO – Set oriented numerical methods for dynamical systems," *Ergodic Theory, Analysis, and Efficient Simulation of Dynamical Systems*, ed. Fiedler, B. (Springer-Verlag, Berlin), pp. 145–174; software available at `http://www.djs2.de/gaio`.

[11] Dellnitz, M. & Junge, O. [2002] "Set oriented numerical methods for dynamical systems," *Handbook of Dynamical Systems II: Towards Applications*, eds. Fiedler, B., Iooss, G. & Kopell, N. (World Scientific, Singapore), pp. 221–264.

[12] Dieci, L. & Lorenz, J. [1995] "Computation of Invariant Tori by the Method of Characteristics," *SIAM J. Num. Anal.* **32**(5), 1436–1474.

[13] Dieci, L., Lorenz, J., & Russell, R. D. [1991] "Numerical calculation of invariant tori," *SIAM J. Sci. Stat. Comput.* **12**(3), 607–647.

[14] Doedel, E. J. [1981] "AUTO, a program for the automatic bifurcation analysis of autonomous systems," *Congr. Numer.* **30**, 265–384.

[15] Doedel, E. J., Champneys, A. R., Fairgrieve, T. F., Kuznetsov, Yu. A., Sandstede, B. & Wang, X. J. [1997] "AUTO97: Continuation and bifurcation software for ordinary differential equations," available via `http://cmvl.cs.concordia.ca/`.

[16] Doedel, E. J., Keller, H. B. & Kernévez, J. P. [1991] "Numerical analysis and control of bifurcation problems: I," *Int. J. Bifurcation and Chaos* **1**(3), 493–520.

[17] Doedel, E. J., Keller, H. B. & Kernévez, J. P. [1991] "Numerical analysis and control of bifurcation problems: II," *Int. J. Bifurcation and Chaos* **1**(4), 745–772.

[18] Doedel, E. J., Paffenroth, R. C., Champneys, A. R., Fairgrieve, T. F., Kuznetsov, Yu. A., Oldeman, B. E., Sandstede, B. & Wang, X. J., [2000] "AUTO2000: Continuation and bifurcation software for ordinary differential equations," available via `http://cmvl.cs.concordia.ca/`.

[19] Edoh, K. D., Russell, R. D., & Sun, W., "Orthogonal Collocation for Hyperbolic PDEs & Computation of Invariant Tori," Australian National Univ., Mathematics Research Report No. MRR 060-95.

[20] Fairgrieve, T. F. & Jepson, A. D. [1991] "O. K. Floquet multipliers," *SIAM J. Numer. Anal.* **28**(5), 1446–1462.

[21] Guckenheimer, J. & Holmes, P. [1986] *Nonlinear Oscillations, Dynamical Systems and Bifurcations of Vector Fields* (Springer-Verlag, New York), second edition.

[22] Guckenheimer, J. & Vladimirsky, A. [2004] "A fast method for approximating invariant manifolds," *SIAM J. Appl. Dyn. Sys.*, in press.

[23] Guckenheimer, J. & Worfolk, P. [1993] "Dynamical systems: Some computational problems," *Bifurcations and Periodic Orbits of Vector Fields*, ed. Schlomiuk, D. (Kluwer Academic Publishers), pp. 241–277.

[24] Henderson, M. E. [2002] "Multiple parameter continuation: Computing implicitly defined k–manifolds," *Int. J. Bifurcation and Chaos* **12**(3), 451–476.

[25] Henderson, M. E. [2003] "Computing invariant manifolds by integrating fat trajectories," Technical Report RC22944, IBM Research.

[26] Hobson, D. [1993] "An efficient method for computing invariant manifolds of planar maps," *J. Comput. Phys.* **104**(1), 14–22.

[27] Johnson, M. E., Jolly, M. S. & Kevrekidis, I. G. [1997] "Two-dimensional invariant manifolds and global bifurcations: some approximation and visualization studies," *Num. Alg.* **14**(1–3), 125–140.

[28] Johnson, M. E., Jolly, M. S. & Kevrekidis, I. G. [2001] "The Oseberg transition: visualization of global bifurcations for the Kuramoto-Sivashinsky equation," *Int. J. Bifurcation and Chaos* **11**(1), 1–18.

[29] Junge, O. [2000] "Rigorous discretization of subdivision techniques," in B. Fiedler, K. Gröger, and J. Sprekels (Eds.) *Proc. Int. Conf. Diff. Eqs.* **2** (World Scientific, Singapore), pp. 916–918.

[30] Junge, O. [2000] *Mengenorientierte Methoden zur numerischen Analyse dynamischer Systeme* (Shaker, Aachen).

[31] Keller, H. B. [1977] "Numerical solution of bifurcation and nonlinear eigenvalue problems," *Applications of Bifurcation Theory*, ed. Rabinowitz, P. H. (Academic Press), pp. 359–384.

[32] Krauskopf, B. & Osinga, H. M. [1998] "Globalizing two-dimensional unstable manifolds of maps," *Int. J. Bifurcation and Chaos* **8**(3), 483–503.

[33] Krauskopf, B. & Osinga, H. M. [1998] "Growing 1D and quasi 2D unstable manifolds of maps," *J. Comp. Phys.* **146**(1), 404–419.

[34] Krauskopf, B. & Osinga, H. M. [1999] "Two-dimensional global manifolds of vector fields," *Chaos* **9**(3), 768–774.

[35] Krauskopf, B. & Osinga, H. M. [2003] "Computing geodesic level sets on global (un)stable manifolds of vector fields," *SIAM J. Appl. Dyn. Sys.* **4**(2), 546–569.

[36] Krauskopf, B. & Osinga, H. M. [2004] "The Lorenz manifold as a collection of geodesic level sets," *Nonlinearity* **17**(1), C1–C6.

[37] Kuznetsov, Yu. A. [1998] *Elements of Applied Bifurcation Theory.* (Springer Verlag, NY), second edition.

[38] Lorenz, E. N. [1963] "Deterministic nonperiodic flows," *J. Atmosph. Sci.* **20**, 130–141.

[39] Lust, K. [2001] "Improved numerical Floquet multipliers," *Int. J. Bifurcation and Chaos* **11**(9), 2389–2410.

[40] Misner, C. W., Thorne, K. S., Wheeler, J. A. [1970] *Gravitation* (W. H. Freeman and Company, San Francisco).

[41] Phillips, M., Levy, S. & Munzner, T. [1993] "Geomview: An interactive geometry viewer," *Notices of the Amer. Math. Soc.* **40**, 985–988.

[42] Osinga, H. M. [2000] "Non-orientable manifolds of periodic orbits," *Proc. Int. Conf. Differential Eqations, Equadiff 99 (Berlin)* Vol. 2, eds. Fiedler, B., Gröger, K. & Sprekels, J. (World Scientific, Singapore), pp. 922–924.

[43] Osinga, H. M. [2003] "Non-orientable manifolds in three-dimensional vector fields," *Int. J. Bifurcation and Chaos* **13**(3), 553–570.

[44] Osinga, H. M. & Krauskopf, B. [2002] "Visualizing the structure of chaos in the Lorenz system," *Computers and Graphics* **26**(5), 815–823.

[45] Osinga, H. M. & Krauskopf, B. [2004] "Crocheting the Lorenz manifold," *The mathematical intelligencer*, to appear (`http://www.enm.bris.ac.uk/anm/preprints/2004r03.html`).

[46] Peraire, J., Peiro, J. & Morgan, K. [1999] "Advancing Front Grid Generation," *Handbook of Grid Generation*, eds. Thompson, J. F., Soni, B. K.,& Weatherill, N. P. (CRC Press), Chapter 17.

[47] Rebay, S. [1993] "Efficient unstructured mesh generation by means of Delaunay triangulation and Bowyer-Watson algorithm," *J. Comp. Phys.* **106**(1), 125–138.

[48] Rheinboldt, W. C. [1986] *Numerical analysis of parametrized nonlinear equations*, University of Arkansas Lecture Notes in the Mathematical Sciences, (Wiley-Interscience).

[49] Russell, R. D. & Christiansen, J. [1978] "Adaptive mesh selection strategies for solving boundary value problems," *SIAM J. Numer. Anal.* **15**, 59–80.

[50] Sacker, R.J. [1965] "A new approach to the perturbation theory of invariant surfaces," *Comm. Pure Appl. Math.* **18**, 717–732.

[51] Sethian, J.A. & Vladimirsky, A. [2004] "Ordered Upwind Methods for Static Hamilton-Jacobi Equations: Theory & Applications," *SIAM J. Num. Anal.*, in press.

[52] Seydel, R. [1995] *From Equilibrium to Chaos. Practical Bifurcation and Stability Analysis* (Springer-Verlag, New York), second edition.

[53] Spivak, M. [1979] *Differential Geometry* (Publish or Perish, ??), second edition.

[54] Strogatz, S.H. [1994] *Nonlinear Dynamics and Chaos* (Addison-Wesley, ??).