Efficient ridge tracking algorithms for computing Lagrangian coherent structures in fluid dynamics applications

by

Douglas M. Lipinski

B.S., University of Wisconsin - Madison, 2007

A thesis submitted to the

Faculty of the Graduate School of the

University of Colorado in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

Department of Applied Mathematics

2012

This thesis entitled:
Efficient ridge tracking algorithms for computing Lagrangian coherent structures in fluid dynamics applications
written by Douglas M. Lipinski
has been approved for the Department of Applied Mathematics

Kamran Mohseni

Keith Julien

James Meiss

Date

The final copy of this thesis has been examined by the signatories, and we find that both the content and the form meet acceptable presentation standards of scholarly work in the above mentioned discipline.

Lipinski, Douglas M. (Ph.D., Applied Mathematics)

Efficient ridge tracking algorithms for computing Lagrangian coherent structures in fluid dynamics applications

Thesis directed by Prof. Kamran Mohseni

Lagrangian coherent structures (LCS) are recently defined structures used to analyze transport in dynamical systems with general time dependence. LCS techniques have seen increasing use over the past decade, but several factors have limited their application to highly complex and three-dimensional flows. In this dissertation, I study the computation of LCS in the context of fluid dynamics applications. The primary examples used here are axisymmetric simulations of swimming jellyfish, a three-dimensional ocean current simulation, a three-dimensional hurricane simulation, and various test cases and analytically defined flows. All these flows involve complicated dynamics and fluid transport that can be analyzed using LCS to reveal the flow structures and underlying transport behavior.

The main contribution of this dissertation is the development and application of a class of efficient algorithms for computing LCS in a given velocity field. Large computational time has been a major hurdle to the widespread adoption of LCS techniques, especially in three dimensions. The ridge tracking algorithms presented here take advantage of the definition of LCS as codimension-one manifolds by avoiding computations in parts of the domain away from the LCS surfaces. By detecting and tracking LCS through the domain of interest, the computational order is reduced from $\mathcal{O}\left(1/\delta x^n\right)$ to $\mathcal{O}\left(1/\delta x^{n-1}\right)$ in n-dimensional problems. In three dimensions, this algorithm is used to compute the LCS in a warm-core ring in the Gulf of Mexico and a hurricane simulation, revealing a new type of LCS structure in the boundary layers of these geophysical vortices. The transport of these structures is analyzed and found to enhance the potential for diffusive mixing in these flow regions through the generation of small length scales.

Dedication

This dissertation and the years of work it represents are dedicated to my wife, family, and friends. I am infinitely grateful for all their loving help and support. I am particularly thankful for the patience and understanding of my wife and the friends and colleagues with whom I have shared many hours of enjoyable discussion, research related and otherwise.

Acknowledgements

I am pleased to acknowledge the many helpful conversations and useful advice received from many colleagues, especially Prof. Kamran Mohseni, Prof. James Meiss, and Dr. Brock Mosovsky. I am also grateful for the teaching and encouragement of the University of Colorado - Boulder, Applied Mathematics faculty as well as all the help from the department staff that I received along the way.

Contents

Tables

Table

Figures

Figure

Chapter 1

Introduction

There is, perhaps, no process more central to the understanding of realistic fluid flows than the transport and mixing caused by their velocity fields. The mixing of fuel and oxygen in combustion, chemical mixing in manufacturing processes, thermal convection, efficient trajectory planning in the ocean and atmosphere, even the energy cycle of a hurricane; all these and other examples too numerous to list rely on the transport and mixing of physical quantities in fluid flows. This subject has fascinated the dynamical systems community for decades, permeating the entire field. After all, dynamics implies motion and therefore transport; mixing is an almost inevitable consequence in any realistic system.

The field of dynamical systems concerns the study of systems governed by equations of the type

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, t) \tag{1.1}$$

where $\mathbf{x}$ represents the phase space of the system. Techniques have long existed to analyze discrete time maps and homogeneous (non time-dependent) or periodic dynamical systems (e.g. [88, 92]). Early work primarily involved analysis of fixed points, limit cycles, orbit stability, stable and unstable manifolds, and periodic orbits. In particularly, Mackay et al. investigated the transport in Hamiltonian systems using area preserving maps [61]. Invariant curves and Cantor sets (Cantori) provide an outline of the transport during the iteration of such maps. In the context of continuous time systems, the stable and unstable manifolds of hyperbolic fixed points have been widely analyzed because they separate phase space into invariant regions of qualitatively distinct behavior [1].

More recently, lobe dynamics theory has been developed to describe the interplay between stable and unstable manifolds that form heteroclinic/homoclinic tangles. If a time periodic perturbation is added to a homogeneous dynamical system containing heteroclinic or homoclinic connections, it may break these connections, creating a complex tangle of lobes that act to transport trajectories between regions of phase space [76]. Since most fluid flows are time dependent, such periodic flows more closely mimic real world flows than homogeneous systems. However, time periodicity is a significant constraint and many of the results are framed in terms of the infinite time behavior of the system which may not be defined (or relevant) for physical systems.

Realistic fluid flows and practical applications do not typically meet the assumptions for classical dynamical systems theories to apply. Fluid flows may be highly time dependent, aperiodic, and only defined over some finite time interval. Worse yet, the velocity field may be defined as data files resulting from data collection (e.g. particle image velocimetry, PIV) or as the solution of a computational fluid dynamics (CFD) problem. Additionally, in some fluid flows the choice of reference frame is not obvious either. For example, a vortex ring is logically considered either in the frame of the vortex ring fixed in a steady background flow or in the frame of a quiescent fluid with a steadily translating vortex ring. More complicated cases may have many possible choices of reference frame and cause difficulty for any type of traditional dynamical systems analysis involving hyperbolic fixed points and their manifolds since changing the reference frame creates and destroys fixed points. In fact, any orbit may made a fixed point by considering a reference frame attached to that orbit.

Many steps have been taken to overcome these issues during the past decade. In particular, the issue of reference frame choice has been tackled by using Lagrangian (as opposed to Eulerian) methods. Lagrangian methods depend on the relative motion of nearby trajectories and are therefore frame independent. Additionally, much of the focus in applications has shifted to the *finite time* analysis of transport and mixing within and between coherent structures. While it is perhaps most natural to think of coherent structures as regions in a flow with qualitatively similar behavior, many publications have referred to the *boundaries* of these regions as coherent structures. For consistency and simplicity this latter usage is applied here. In 2000, Haller and Yuan proposed defining Lagrangian coherent structures (LCS) as material lines with locally the longest or shortest linear stability times, giving rise to stable and unstable material surfaces [37]. Haller and Yuan draw many parallels between the stable/unstable material lines and the well known unstable/stable manifolds of homogeneous and time periodic dynamical systems. This was one of the first attempts to enable systematic and unambiguous detection of coherent structures in general fluid flows.

In an influential paper from 2005, Shadden et al. defined LCS as ridges of the finite time Lyapunov exponent (FTLE) field [85], a definition that is commonly used today. The FTLE is the finite time version of the traditional Lyapunov exponent and measure stretching in the flow. Intuitively, it is expected that separatrices in the flow (such as vortex boundaries or separation/reattachment lines for airfoils) will undergo the locally highest stretching as particles on either side of the separatrix diverge from each other. For this same reason, LCS are expected to be barriers to transport. This has been observed in countless examples since the publication of Shadden et al.'s paper. Several prominent examples include high frequency radar data in Monterey Bay [86], jellyfish swimming [56, 73, 95], vortex shedding behind an airfoil [54], three-dimensional turbulence [33] and geophysical flows [49, 50, 82].

To use this definition of LCS (ridges in the FTLE field), one first defines the flow map:

$$\mathbf{\Phi}_{t_0}^{T}(\mathbf{x}_0) = \mathbf{x}_0 + \int_{t_0}^{t_0+T} \mathbf{v}(\mathbf{x}(t), t)dt \tag{1.2}$$

where $\mathbf{x}_0$ is the initial position, t_0 is the initial time, T is the integration time, and $\mathbf{\Phi}$ is the flow map. This map projects fluid particles from their initial position at time t_0 to a final location at time $t_0 + T$ and is typically computed via numerical integration of particle trajectories. Once the flow map has been computed, the deformation tensor is defined as

$$\Delta = (\nabla_{\mathbf{x}_0}\mathbf{\Phi})^{\intercal}(\nabla_{\mathbf{x}_0}\mathbf{\Phi}) \tag{1.3}$$

where $^\intercal$ denotes the transpose operator and $\nabla_{\mathbf{x}_0}$ is the gradient with respect to the initial particle positions. Finally, the FTLE is computed as

$$\sigma_{t_0}^{T}(\mathbf{x}_0) = \frac{1}{|T|} \ln \sqrt{\lambda_{\max}(\Delta)} \tag{1.4}$$

where σ is the FTLE and $\lambda_{\max}(\Delta)$ is the largest eigenvalue of Δ. Note that T may be positive or negative, corresponding to forward or backward time integration. This results in two types of LCS for any flow, the forward LCS reveal repelling structures and the backward LCS reveal attracting structures.

In most applications, the LCS are then visualized by making contour plots of the FTLE field. In such cases, the FTLE ridges are visible just as mountain ridges are visible on topographic maps. However, in complex flows and particularly in three dimensions it may be necessary to precisely compute the LCS ridges. Shadden et al. give two potential definitions of a ridge, curvature ridges and second derivative ridges where second derivative ridges are a subset of curvature ridges. In this dissertation, I use an n-dimensional ridge definition that simplifies to the second derivative ridge definition in two dimensions. A similar definition was used by Lekien et al. to define LCS in n-dimensional systems [51].

Definition *Ridge*: A *ridge* of a C^2 function F is a codimension-one surface S satisfying:

(1) $\mathbf{n} \cdot \nabla F = 0$ for all points on S where $\mathbf{n}$ is a unit vector normal to S.

(2) $\mathbf{n}^\intercal H \mathbf{n} = \min_{\|\mathbf{u}\|=1} (\mathbf{u}^\intercal H \mathbf{u}) < 0$ for all points on S where H is the Hessian matrix associated with F.

Using an equivalent definition in two dimensions, Shadden et al. proved several useful properties of LCS, the most important of which is an estimate of the flux through LCS surfaces. Their result states that along LCS surfaces

$$\frac{dL}{dt} = \frac{\langle \hat{\mathbf{t}}, \nabla\sigma \rangle}{\langle \hat{\mathbf{n}}, \Sigma\hat{\mathbf{n}} \rangle} \left\langle \hat{\mathbf{t}}, \frac{\partial \hat{\mathbf{n}}}{\partial t} - J\hat{\mathbf{n}} \right\rangle + \mathcal{O}\left(1/\left|T\right|\right) \tag{1.5}$$

where L is the distance of a particle from the LCS, $\hat{\mathbf{t}}$ is the unit tangent to the ridge, $\hat{\mathbf{n}}$ is the unit normal, and J is the Jacobian matrix of the velocity field [85]. The first term in this expansion is small for well defined ridges (large curvature normal to the ridge and small slope along the ridge) so for sufficiently well defined ridges and long enough integration time, T, the flux through LCS is expected to be negligible and LCS form nearly invariant manifolds.

Haller recently revealed that the proof of Equation 1.5 given by Shadden et al. [85] is invalid without some additional assumptions [35]. The proof relies on three additional assumptions laid out in Appendix C of [35]. In general, these extra assumptions are non-trivial to verify and Haller gives several examples where ridges in the FTLE field may have large flux through them or do not correspond to any identifiable coherent structure [35]. To remedy this, Haller presents a variational formulation for LCS based on locating material lines that maximize the normal repulsion or attraction rate over some time interval. These LCS are guaranteed to be invariant because they are *defined* as material lines and they are also guaranteed to be repelling or attracting rather than a result of large stretching caused by shear as can be the case with FTLE ridges.

While Haller's variational formulation for LCS offers guarantees of invariance and normally repelling or attracting structures, it also has several disadvantages. For example, in this context, LCS are invariant over the chosen time interval, but if the time interval is shifted the LCS must be recomputed and the LCS will not typically be invariant from one time interval to the next. If LCS are computed using this method over time interval $[t_a, t_b]$, the repelling LCS will be initially computed at time t_a and must be numerically advected in time to find their position at any other time in this interval. This advection step is always unstable due to the repelling nature of the LCS and the length of the LCS will decrease exponentially during the advection, resulting in very different lengths of the LCS curves at opposite ends of the time interval. Finally, and most importantly, there are major difficulties in extending this definition to dimensions greater than two. No computational method has been developed to actually use this technique in three dimensions.

Other methods of defining LCS have also been proposed and implemented. Some groups have used eigenvectors of the Perron-Frobenius operator to define coherent structures as nearly invariant sets [21, 29, 30, 72]. Other groups have use the so called M function [63]. However, these methods also come with their own drawbacks and complications. For the remainder of this dissertation, LCS will refer to ridges of the FTLE field. I have chosen to use this method despite its uncertainties and limitations for the simple reason that it is the most commonly used technique and has proven successful in a wide range of applications over the past seven years. When combined with an analysis of drifter trajectories, FTLE based LCS offer a good means of identifying coherent structures and a high level of confidence in the results.

Despite their widespread use, there are several problems with the methods most commonly used to compute LCS. Of primary concern is the large computational cost which can even exceed the cost of solving the CFD problem to compute the desired velocity field. The standard algorithm for computing LCS proceeds as follows:

Algorithm 1 Standard LCS algorithm

Set integration time, T

Initialize drifter grid, $\mathbf{x}_0$, with spacing δx

for $t = t_0$ **to** t_f **do**

  for $s = -1, 1$ **do**

    Advect drifters in time, $\mathbf{x} = \mathbf{\Phi}_t^{sT}(\mathbf{x}_0)$

    Approximate $\nabla_{\mathbf{x}_0}\mathbf{\Phi}$ via finite differences

    Compute Δ, $\lambda_{\max}(\Delta)$, and σ at every point in the domain

    if $s = -1$ **then**

      $\sigma_{\text{backward}} \leftarrow \sigma$

    else

      $\sigma_{\text{forward}} \leftarrow \sigma$

    end if

  end for

end for

View LCS as contour plots of σ_{backward} and σ_{forward}

Unlike many numerical algorithms that involve eigenvalue problems, Δ is only an n-dimensional tensor where n is the dimension of the space (typically two or three) so the largest eigenvalue can be computed very quickly either analytically or using the power method. Most of the computational time for algorithm 1 is spent advecting particles and interpolating the velocity field if it is defined by data files. A typical example in three dimensions may use a grid with around 500^3 points and the integration time may require 50 advection steps. Additionally, the LCS need to be computed at multiple times, perhaps 500 time steps. If a fourth-order Runge-Kutta (RK4) advection scheme is used, this situation will require 25 trillion individual particle advection steps (including intermediate RK4 steps) and velocity interpolations. Although the problem is embarrassingly parallel since no communication between particles is required during the advection stage, better algorithms are required to make LCS truly computationally attractive.

Visualizing the results of three-dimensional LCS creates its own problems. In two-dimensions, it is often sufficient to simply view contour plots of the FTLE fields and the LCS surfaces are never explicitly found. However, this method fails dramatically in three dimensions. The best that can be done is to plot level sets of some high FTLE value or view two-dimensional slices of the three-dimensional FTLE field. However, if the LCS surfaces are explicitly computed it is possible to sensibly visualize the entire LCS surfaces or easily computed their intersection with viewing planes or other surfaces of interest.

Several past papers have attempted to improve the computation efficiency of LCS algorithms by using adaptive mesh refinement (AMR) algorithms that refine the computational mesh near the LCS [31, 79]. In such algorithms, an initial, sparse grid computation is performed to obtain a coarse estimate of the FTLE values. Then, various methods may be used to determine the location of LCS surfaces within this coarse grid and iteratively refine the grid in those regions. Garth et al. refine the mesh based on the estimated error of successively refined interpolants [31] while Sadlo and Peikert locate ridges and refine the mesh by finding points where the directional derivative changes sign and the second derivative is negative [79]. These AMR algorithms are effective at reducing the computational load, but still result in computing many FTLE values away from the LCS and see speed ups of about 1.4 to 10 times.

Additional attempts at improving LCS efficiency have been made by reusing flow map computations in subsequent time steps [5]. Since the LCS are often desired at many different times and the time step may be smaller than the integration time used this leads to overlapping integrations. For example, if the LCS are desired at times $t = \{0.0, 0.1, 0.2, ...\}$ and an integration time of $T = 1.0$ is to be used, the standard approach would be to compute a series of flow maps $\{\Phi_{0.0}^{10.0}, \Phi_{0.1}^{10.1}, \Phi_{0.2}^{10.2}, \ldots\}$ for each time the FTLE field is required. However, it is also possible to compute the flow maps $\{\Phi_{0.0}^{0.1}, \Phi_{0.1}^{0.2}, \Phi_{0.2}^{0.3}, \ldots\}$ and use the fact that $\Phi_{0.0}^{10.0} = \Phi_{9.9}^{10.0} \circ \Phi_{9.8}^{9.9} \circ \Phi_{9.7}^{9.8} \cdots \Phi_{0.0}^{0.1}$. This composition of flow maps technique has the potential for large efficiency gains, but only if there is a significant overlap in the integration times and many time steps are desired. The method also comes at the cost of greatly increased memory usage to store all the necessary flow maps.

Another recent paper has reformulated the FTLE problem as an Eulerian level set problem involving the solution of a Liouville equation [52]. This allows the use of many previously developed high order accurate schemes for the resulting hyperbolic PDE's. Although there are many existing techniques for solving such hyperbolic systems, the time step used in the Eulerian method must obey a CFL condition to ensure stability while the time step in more commonly used Lagrangian techniques is typically determined only by the required accuracy. The larger time steps and easily implemented high-order accurate methods (such as Runge-Kutta methods) mean that Lagrangian techniques are typically faster.

This dissertation presents a new method to decrease the cost associated with LCS computations, a *ridge tracking algorithm* [57,58]. This algorithm operates by detecting some initial points on the LCS surfaces and then tracking the FTLE ridges through space, eliminating almost all FTLE computations away from the LCS. Since the ridges are codimension-one structures, the ridge tracking algorithm reduces the order of computation from $\mathcal{O}\left(1/\delta x^n\right)$ to $\mathcal{O}\left(1/\delta x^{n-1}\right)$ in n-dimensional systems and provides speed ups of more than an order of magnitude in typical examples. It is well suited to flows that require high resolution or contain sparse LCS structures.

Lagrangian coherent structures provide important insight and opportunities for analysis of realistic, three-dimensional fluid flows. A ridge tracking algorithm can accurately compute Lagrangian coherent structures in complex flows while improving the computational order and simplifying LCS visualization and analysis. The remaining chapters of this dissertation present evidence in defense of this thesis.

Chapter 2 discusses some preliminary considerations concerning the computation and understanding of LCS that have not been reported by other researchers. Chapter 3, presents the LCS for swimming jellyfish and reveals new flow structures while confirming previously held beliefs about the feeding and swimming nature of two different classes of jellyfish. Chapter 4 presents a gridless ridge tracking algorithm in detail for two-dimensional problems. An extension to three dimensions is also presented, but the complexity involved in higher dimensional mesh generation slows computations. Chapter 5 presents a grid-based ridge tracking algorithm to enable efficient mesh generation and surface meshing for three-dimensional problems. Analysis of this algorithm provides error bounds for the computed LCS locations and algorithm performance is verified for several examples. In three dimensions, the ridge tracking algorithm decreases the computational time from $\mathcal{O}\left(1/\delta x^3\right)$ to about $\mathcal{O}\left(1/\delta x^{2.1}\right)$. Finally, Chapter 6 presents the application of the three dimensional ridge tracking algorithm to eddies in the Gulf of Mexico and a numerical simulation of a hurricane. The LCS in the boundary layers of these geophysical vortices reveal a new type of LCS structure, dubbed "checkerboard LCS", that are shown to influence the fluid stretching and transport in these regions.

Chapter 2

LCS and FTLE preliminaries

Before discussing the main results of this dissertation, it is useful to discuss some details of the FTLE computations and the conceptual perception of LCS in general that have not been discussed previously. In particular, the use of the appropriate coordinate system for FTLE computations and some conceptual failings when extending the understanding of LCS in two dimension to three dimensions. Although it seems obvious that the appropriate coordinate system must be used in all calculations, several past publications have failed to do so. Additionally, it is common to make the analogy between LCS and the stable and unstable manifolds of homogeneous systems. This analogy is often quite apt in two dimensional systems, but in three dimensions there are situations where the analogy fails. As long as researchers account for these factors, LCS remain a useful tool for flow structure analysis.

2.1 The FTLE and coordinate systems

Despite the common use of FTLE values to define and compute coherent structures, several errors have appeared in past publications. For example, FTLE values for an incompressible flow must be non-negative (see Appendix A for proof of this property), but negative values have been reported (e.g. Figure 6 of [85]). In at least one other publication, two-dimensional Cartesian FTLE computations were carried out while axisymmetric assumptions were made elsewhere in the same paper, and the specifics of the computations are often neglected [73,84]. One must use the correct form of the gradient operator depending on the coordinate system when computing the FTLE values. Failing to do so causes potential errors in the resulting FTLE values. This is highlighted below with analytical and computational examples of several axially symmetric flows.

In cylindrical and spherical coordinates (or any general curvilinear coordinate system), the gradient operator is more complicated than in cartesian coordinates. Symmetry assumptions may further complicate this picture, but it is essential to use the appropriate gradient operator when computing the gradient of the flow map. If initial coordinates are denoted (x_0, y_0, z_0) and final coordinates $\Phi_{t_i}^{t_i+T}(\mathbf{x}_0) = (x_f, y_f, z_f)$, then

$$\nabla\Phi = \begin{bmatrix} \frac{\partial x_f}{\partial x_i} & \frac{\partial x_f}{\partial y_i} & \frac{\partial x_f}{\partial z_i} \\ \frac{\partial y_f}{\partial x_i} & \frac{\partial y_f}{\partial y_i} & \frac{\partial y_f}{\partial z_i} \\ \frac{\partial z_f}{\partial x_i} & \frac{\partial z_f}{\partial y_i} & \frac{\partial z_f}{\partial z_i} \end{bmatrix} \tag{2.1}$$

in cartesian coordinates. In cylindrical coordinates, (r, θ, z),

$$\nabla\Phi = \begin{bmatrix} \frac{\partial r_f}{\partial r_i} & \frac{1}{r_i}\frac{\partial r_f}{\partial \theta_i} & \frac{\partial r_f}{\partial z_i} \\ r_f\frac{\partial \theta_f}{\partial r_i} & \frac{r_f}{r_i}\frac{\partial \theta_f}{\partial \theta_i} & r_f\frac{\partial \theta_f}{\partial z_i} \\ \frac{\partial z_f}{\partial r_i} & \frac{1}{r_i}\frac{\partial z_f}{\partial \theta_i} & \frac{\partial z_f}{\partial z_i} \end{bmatrix}. \tag{2.2}$$

Finally, in spherical coordinates, (ρ, ϕ, θ), with azimuthal angle ϕ and polar angle θ,

$$\nabla\Phi = \begin{bmatrix} \frac{\partial \rho_f}{\partial \rho_i} & \frac{1}{\rho_i \sin(\theta_i)}\frac{\partial \rho_f}{\partial \phi_i} & \frac{1}{\rho_i}\frac{\partial \rho_f}{\partial \theta_i} \\ \rho_f \sin(\theta_f)\frac{\partial y_f}{\partial \rho_i} & \frac{\rho_f \sin(\theta_f)}{\rho_i \sin(\theta_i)}\frac{\partial \phi_f}{\partial \phi_i} & \frac{\rho_f \sin(\theta_f)}{\rho_i}\frac{\partial \phi_f}{\partial \theta_i} \\ \rho_f\frac{\partial \theta_f}{\partial \rho_i} & \frac{\rho_f}{\rho_i \sin(\theta_i)}\frac{\partial \theta_f}{\partial \phi_i} & \frac{\rho_f}{\rho_i}\frac{\partial \theta_f}{\partial \theta_i} \end{bmatrix}. \tag{2.3}$$

Symmetry assumptions may eliminate terms from these expressions, but the appropriate coordinate system *must* be used.

2.1.1 FTLE examples

A recent publication used velocity data obtained from digital particle image velocimetry (DPIV) to compute FTLE values and transport near a swimming jellyfish [73]. In this case, only a two dimensional slice of the velocity field is available and particles are assumed to move two-dimensionally in this plane. However, to analyze volumetric transport, additional assumptions can be made such as axisymmetric flow. As long as the velocity field has been accurately measured or computed, the assumption of axial symmetry does not change the particle trajectories (they remain in the same plane). However, it does change the FTLE values by introducing the potential for geometric stretching as particles move radially away from the axis of symmetry. This case is discussed in more detail below.

In this swirl free axisymmetric case it is particularly easy to make a mistake in computing the FTLE. In this situation, all motion occurs in the rz-plane, but the geometric stretching in the θ direction must still be accounted for. $\nabla\Phi$ simplifies to

$$\begin{bmatrix} \dfrac{\partial r_f}{\partial r_i} & 0 & \dfrac{\partial r_f}{\partial z_i} \\ 0 & \dfrac{r_f}{r_i} & 0 \\ \dfrac{\partial z_f}{\partial r_i} & 0 & \dfrac{\partial z_f}{\partial z_i} \end{bmatrix}. \tag{2.4}$$

where the r_f/r_i term accounts for the geometric stretching. If this term is neglected and the FTLE values are computed as if the flow were in a two-dimensional cartesian coordinate system incorrect values may be obtained. In fact, the eigenvalues of Δ for swirl free, axisymmetric flow are $\lambda_1 = (r_f/r_i)^2$, λ_2, λ_3 where λ_2 and λ_3 are the eigenvalues of

$$\begin{bmatrix} \left(\dfrac{\partial r_f}{\partial r_i}\right)^2 + \left(\dfrac{\partial z_f}{\partial r_i}\right)^2 & \dfrac{\partial r_f}{\partial r_i}\dfrac{\partial r_f}{\partial z_i} + \dfrac{\partial z_f}{\partial r_i}\dfrac{\partial z_f}{\partial z_i} \\ \dfrac{\partial r_f}{\partial r_i}\dfrac{\partial r_f}{\partial z_i} + \dfrac{\partial z_f}{\partial r_i}\dfrac{\partial z_f}{\partial z_i} & \left(\dfrac{\partial r_f}{\partial z_i}\right)^2 + \left(\dfrac{\partial z_f}{\partial z_i}\right)^2 \end{bmatrix}. \tag{2.5}$$

If the geometric stretching is ignored, the FTLE results will be incorrect wherever λ_1 is the dominant eigenvalue. The following examples demonstrate the possible errors in this case.

2.1.1.1 Sliding-expanding cylinder

This section presents a simple example where geometric stretching is critically important, the flow around a sliding-expanding cylinder as presented by Lagerstrom [47]. In this incompressible flow, a cylinder of radius, R, is impulsively started sliding along its axis with constant velocity, U. At the same time, the radius of the cylinder starts expanding at a rate of $R = Ct^n$. The fluid is initially at rest. Only the case $n = 1/2$ is considered here. For this case, a length scale, L, is defined from the relation

$$L = \frac{Ut}{R^2}. \tag{2.6}$$

and the Reynolds number is

$$Re = \frac{UL}{\nu} \tag{2.7}$$

for kinematic viscosity ν.

As shown by Lagerstrom [47], the geometric constraints of the problem make it simple to determine the radial velocity,

$$\frac{v}{U} = \frac{1}{2Ur}\frac{d(R^2)}{dt} = \frac{L}{2r}. \tag{2.8}$$

By introducing the alternative coordinates

$$\xi = \frac{r}{2\sqrt{\nu t}}, \quad \eta = \frac{R}{2\sqrt{\nu t}} = \frac{\sqrt{Re}}{2}, \tag{2.9}$$

one may obtain the axial velocity as

$$\frac{u}{U} = \frac{g(\xi,\eta)}{g(\eta,\eta)} \tag{2.10}$$

where

$$g(\xi,\eta) = \int_{\xi}^{\infty} e^{-\sigma^2}\sigma^{(2\eta^2-1)}d\sigma = \frac{1}{2}\Gamma_U(\eta^2,\xi^2) \tag{2.11}$$

and Γ_U is the upper incomplete gamma function,

$$\Gamma_U(x,a) = \int_{a}^{\infty} e^{-s}s^{x-1}ds. \tag{2.12}$$

In this problem, due to the axial symmetry and independence of the z-coordinate, the gradient of the flow map becomes

$$\nabla\Phi = \begin{bmatrix} \dfrac{\partial r_f}{\partial r_i} & 0 & 0 \\ 0 & \dfrac{r_f}{r_i} & 0 \\ \dfrac{\partial z_f}{\partial r_i} & 0 & 1 \end{bmatrix}. \tag{2.13}$$

Computing the FTLE values requires computing the trajectories to find $\dfrac{\partial r_f}{\partial r_i}$, $\dfrac{r_f}{r_i}$, and $\dfrac{\partial z_f}{\partial r_i}$. This example considers only the forward FTLE values ($T > 0$) since $r_f/r_i < 1$ (and therefore $(r_f/r_i)^2$ cannot be the largest eigenvalue) for all backward FTLE computations.

To simplify this process, non-dimensional variables are defined as follows.

$$z^* = z/L, \quad r^* = r/L, \quad t^* = \frac{t}{L^2/\nu}, \quad u^* = u/U, \quad v^* = v/U \tag{2.14}$$

The variable ξ may then be expressed as $\xi = \dfrac{r^*}{2\sqrt{t^*}}$. It is worth noting that $\nabla\Phi$ is unchanged by this coordinate scaling so the FTLE value of each trajectory is unaffected by the nondimensionalization. Transforming the velocities to dimensionless variables gives the ODEs for particle trajectories as

$$\frac{dr^*}{dt^*} = \frac{Re}{2r^*}, \tag{2.15}$$

$$\frac{dz^*}{dt^*} = Re\frac{g(\xi,\eta)}{g(\eta,\eta)}. \tag{2.16}$$

From this point on, for notational simplicity, I will use only dimensionless variables for this example and drop the * notation.

Integrating Equation 2.15 gives the radial position as a function of time,

$$r(t) = \sqrt{r_i^2 + Re(t - t_i)}, \tag{2.17}$$

for initial time t_i and initial radius r_i. This gives

$$\frac{dr_f}{dr_i} = \left(1 + \frac{ReT}{r_i}\right)^{-1/2}, \tag{2.18}$$

$$\frac{r_f}{r_i} = \left(1 + \frac{ReT}{r_i}\right)^{1/2}, \tag{2.19}$$

for integration time T. It is also worth noting that the Jacobian of the flow, $\det(\nabla\Phi)$, is zero, as it is for all incompressible flows (see Meiss, p. 63 for details [64]). The axial coordinate is more complicated since the axial velocity is governed by viscous diffusion rather than geometric constraints. However, the axial position can be written as

$$z(t) = z_i + \frac{Re}{g(\eta,\eta)} \int_{t_i}^{t} g\left(\xi,\eta\right) ds, \tag{2.20}$$

noting that ξ is a function of t and r while η is a constant. Taking the derivative with respect to r_i and reversing the order of integration and differentiation gives

$$\frac{dz_f}{dr_i} = \frac{Re}{g(\eta,\eta)} \int_{t_i}^{t} \frac{d}{dr_i} g\left(\xi,\eta\right) ds. \tag{2.21}$$

Finally, using $\dfrac{d}{dr_i}g(\xi,\eta) = \dfrac{\partial g}{\partial \xi}\dfrac{\partial \xi}{\partial r_i}$, $Re = 4\eta^2$, Equation 2.17, and much algebra and manipulation gives

$$\frac{dz_f}{dr_i} = \frac{r_i\eta^2}{g(\eta,\eta)} \int_{r_f^2/(4t_f)}^{r_i^2/(4t_i)} \frac{e^{-s}s^{\eta^2-2}}{1-\eta^2/s} ds \tag{2.22}$$

It is simple to show that for all physically relevant parameters (i.e. outside the cylinder radius), $\eta^2/s < 1$ so I expand $1/(1-\eta^2/s)$ as a Taylor series and use this approximation in the the above integral:

$$\begin{aligned}
\frac{dz_f}{dr_i} &= \frac{r_i\eta^2}{g(\eta,\eta)} \int_{r_f^2/(4t_f)}^{r_i^2/(4t_i)} e^{-s}s^{\eta^2-2} \sum_{k=0}^{\infty} (\eta^2/s)^k ds &(2.23)\\
&= \frac{r_i\eta^2}{g(\eta,\eta)} \sum_{k=0}^{\infty} \eta^{2k} \int_{r_f^2/(4t_f)}^{r_i^2/(4t_i)} e^{-s}s^{\eta^2-k-2} ds &(2.24)\\
&= \frac{r_i\eta^2}{g(\eta,\eta)} \sum_{k=0}^{\infty} \eta^{2k}\Gamma\left(\eta^2-k-1, \frac{r_f^2}{4t_f}, \frac{r_i^2}{4t_i}\right) &(2.25)
\end{aligned}$$

where $\Gamma(x,a,b)$ is the incomplete gamma function,

$$\Gamma(x,a,b) = \int_a^b e^{-s}s^{x-1} dx. \tag{2.26}$$

Note that the convergence of the sum in Eq. 2.25 may be very slow near the cylinder where η^2/s approaches 1. At larger radii, using only a few terms of the series gives a very good approximation.

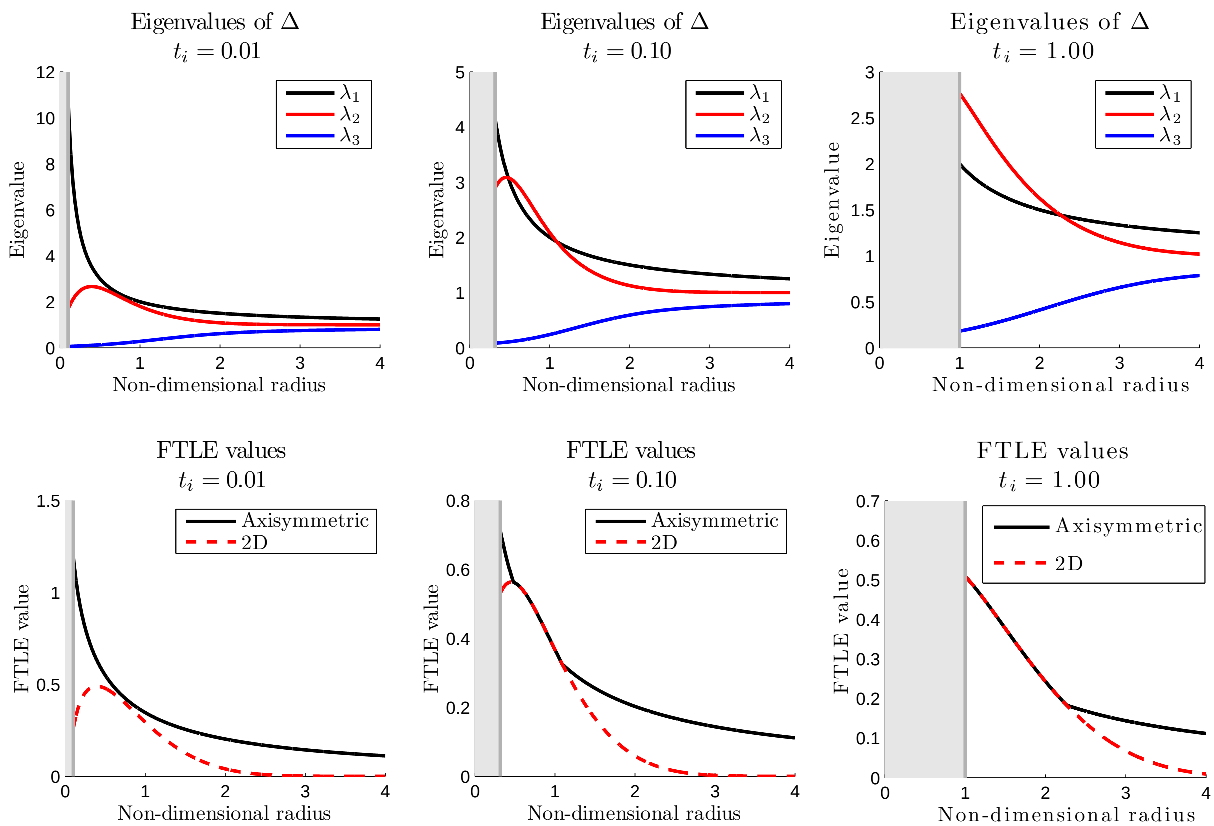


Figure 2.1: Eigenvalue and FTLE results for the sliding-expanding cylinder example. The top row shows the eigenvalues of the axisymmetric deformation tensor and the bottom row shows the FTLE values resulting from the incorrect two-dimensional FTLE calculation as well as the correct axisymmetric calculation. $Re = 1$ and $T = 1$ in all figures. The initial time is $t_i = 0.01$, 0.1, and 1 from left to right. The gray area at the left of each figure represents the cylinder.

There are two components to this flow, the radial motion that is caused by the cylinder expansion and the axial motion that is induced by viscous forces. As seen in Figure 2.1, for small initial time, the geometric stretching due to radial motion may determine the FTLE values in the entire domain. This means that by inappropriately using Cartesian coordinates to compute the FTLE values, incorrect values are found everywhere in the domain. Furthermore, for $t_i = 0.01$, the two-dimensional FTLE calculation actually results in a peak in the FTLE values at about $r = 0.395$. This peak may then be incorrectly interpreted to be an LCS in cartesian coordinates where there is none.

For larger initial times, the cylinder has expanded further and the geometric stretching of the coordinate system is less pronounced. The viscous diffusion of axial velocity has also spread further into the domain, allowing shear effects to create larger amounts of stretching.

2.1.1.2 Hill's spherical vortex

Hill's spherical vortex is a canonical example of an analytical flow that satisfies the Euler equations of fluid motion [38, 48]. This axisymmetric vortex is defined by the stream function

$$\psi = \begin{cases} \frac{-3}{4} U r^2 \left(1 - \frac{x^2 + r^2}{a^2}\right) & \text{for} \quad x^2 + r^2 < a^2 \\ \frac{1}{2} U r^2 \left(1 - \frac{a^3}{(x^2 + r^2)^{3/2}}\right) & \text{for} \quad x^2 + r^2 \geq a^2 \end{cases} \tag{2.27}$$

such that the axial and radial velocities are respectively given by

$$u = -\frac{1}{r}\frac{\partial \psi}{\partial r}, \tag{2.28}$$

$$v = \frac{1}{r}\frac{\partial \psi}{\partial x}. \tag{2.29}$$

These equations give a steady, spherical vortex ring of radius a moving through a uniform background flow of velocity U. All the vorticity in the flow is concentrated within the spherical region $x^2 + r^2 < a$ and the velocity field is not time dependent. The vortex contains a stagnation point at the front and rear of the vortex connected by streamlines (or, in dynamical systems terms, a heteroclinic connection between fixed points). This completely separates the vortex interior from the surrounding flow.

Since Hill's spherical vortex is a steady flow with very straightforward transport dynamics, time dependence is created by adding a sinusoidal perturbation to the axial velocity field. The new velocity field is given by

$$u = \frac{-1}{r}\frac{\partial \psi}{\partial r} + \varepsilon \sin(\omega t) \tag{2.30}$$

$$v = \frac{1}{r}\frac{\partial \psi}{\partial x}. \tag{2.31}$$

We will use parameters

$$U = 1, \quad a = 1, \quad \varepsilon = 0.2, \quad \omega = 5, \tag{2.32}$$

and LCS integration time $T = 5$. This perturbed velocity field is no longer a solution of the Euler equations, but exhibits transport similarities to vortices seen in experimental flows. In particular, the time dependent perturbation breaks the heteroclinic connection between the stagnation points at the front and rear of the vortex and enables transport across the vortex boundary similar to that seen in Shadden et al. [84]. The resulting forward FTLE values for one set of parameters are show in Fig. 2.2. In the absence of the time dependent perturbation, the FTLE field has a strong ridge along the vortex boundary, in this case at $x^2 + r^2 = 1$. The time dependent perturbation causes the manifold to detach from one of the stagnation points, forming a series of folds that enable transport into and out of the vortex via lobe dynamics [76].

In this case, computing the FTLE values as though the flow were two-dimensional Cartesian flow results in errors due to the neglected effects of geometric stretching in the axisymmetric coordinates. The top part of Figure 2.2 shows the FTLE field as computed using axisymmetric coordinates (Equation 2.4) while the bottom inappropriately uses a two-dimensional Cartesian FTLE computation. Regions that differ between the two are outlined in black. The error in the FTLE values is up to 10.95% of the maximum FTLE value.

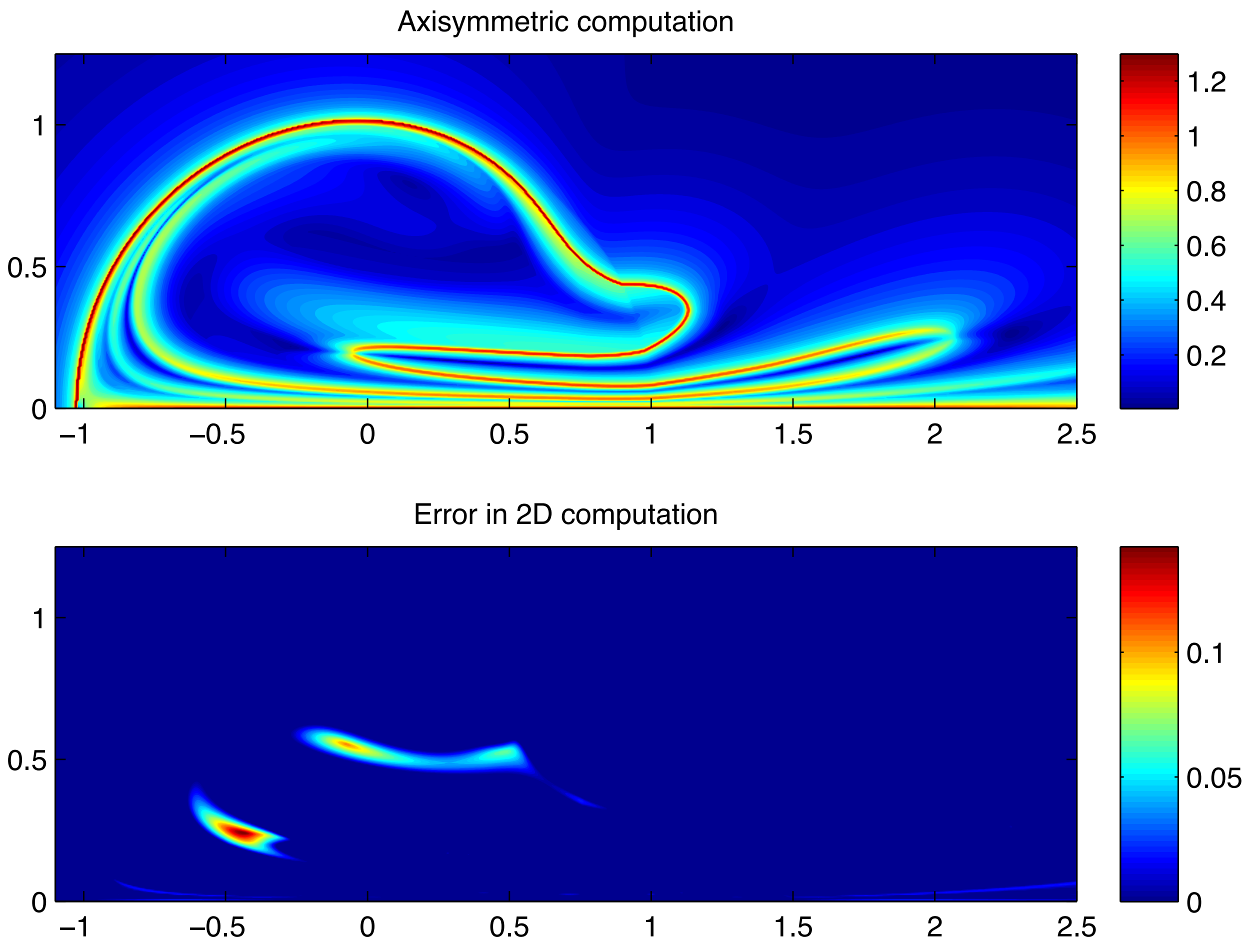


Figure 2.2: The forward FTLE field for Hill's spherical vortex. The FTLE values are shown at top and the error caused by improperly using a two-dimensional FTLE computation is shown at bottom. The unperturbed vortex is a sphere of radius 1 centered at the origin.

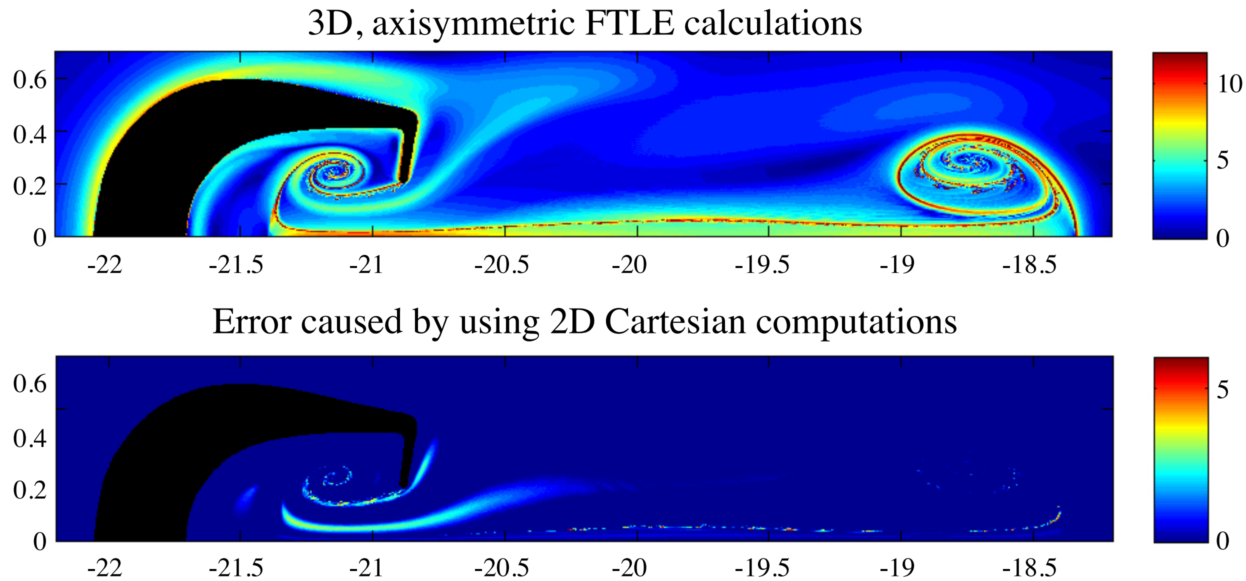


Figure 2.3: Forward FTLE values (top) and the corresponding error (bottom) caused by neglecting geometric stretching in the axisymmetric coordinates for the swimming jellyfish *S. tubulosa*.

2.1.1.3 Axisymmetric jellyfish swimming

Swimming jellyfish have been a popular subject for FTLE based LCS investigation [55, 56,73,95] and are discussed in detail in Chapter 3. In this section, the FTLE field is presented for an axisymmetric jellyfish simulation (species: *Sarsia tubulosa*) as a final example of the errors caused by inappropriately using two-dimensional Cartesian FTLE computations. This example has previously been discussed by Mohseni and collaborators [55,56,81]. The jellyfish swimming motion was digitized from videos of the actual swimming jellyfish and used as input to a CFD code to compute the resulting velocity field [81]. The velocity field is then used to compute the FTLE field and perform any further analysis. For a detailed discussion of LCS created by swimming jellyfish see Chapter 3.

S. tubulosa is a prolate jellyfish that ejects a strong vortex along the axis of symmetry during each swimming stroke. This vortex quickly moves downstream, away from the jellyfish, and enables high momentum transfer and rapid propulsion. In this simulation, the jellyfish has a swimming frequency of 1 Hz and and integration of $T = 0.4$ s was used for the FTLE computations. The forward FTLE values are shown in the top section of Figure 2.3 for one time step. The jellyfish body is drawn in black and the vortex is clearly visible in the FTLE field downstream of the jellyfish. Computing the FTLE field as if the flow were two-dimensional cartesian results in the error shown in the bottom half of Figure 2.3. The absolute error is depicted and the reaches values as large as 50.3% of the largest FTLE values. Furthermore, there are large errors very near to some of the FTLE ridges, possibly effecting attempts to precisely locate the LCS in the flow.

2.1.2 FTLE coordinate system conclusions

The geometric stretching that appears in curvilinear coordinate systems can significantly influence FTLE values. This is not to say that the FTLE values change with the coordinate system, rather, that care must be exercised when using non-cartesian coordinate systems. In the cases presented above, neglecting the geometric stretching term that appears in the gradient of the flow map can result in significant errors, up to 50% of the maximum FTLE values in the FTLE field around a swimming jellyfish.

As LCS techniques continue to become more sophisticated, accuracy of the the FTLE values as well as the eigenvalues and eigenvectors of the deformation tensor becomes increasingly important. Incorrect values may affect the computed LCS locations or even cause false positives or false negatives in LCS detection. This is especially important when the specific eigenvalues or directions of the eigenvectors are crucial, such as in the variational approach to LCS as recently proposed by Haller [35]. Once this necessity is clear, it is simple to compute and use the appropriate gradient operator in any coordinate system and avoid such issues entirely.

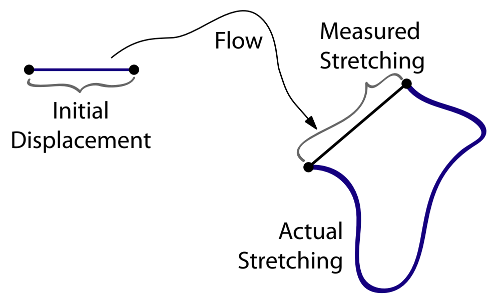


Figure 2.4: An initial line segment may be stretched into a curve, leading to an underestimation of the stretching.

2.2 Conceptual problems and other LCS difficulties

Despite the widespread use and successful implementation of LCS computations, several difficulties remain that must be kept in mind. Firstly, the standard LCS algorithm (see Algorithm 1) approximates FTLE values using finite differences. Typical schemes use a second order accurate central difference, but accurately resolving the FTLE field requires extremely high resolution. Secondly, The FTLE field does not distinguish between hyperbolic stretching and shear, which can lead to shear LCS. Finally, there are several conceptual issues that occur when considering LCS in three dimensions. These issues are further discussed below.

2.2.1 Choice of FTLE resolution

It is often desirable to compute LCS in a domain with a high aspect ratio. A typical example would be oceanic or atmospheric flows where there is often a difference of one to two orders of magnitude between the horizontal domain size and the vertical domain size. Simulations of such geophysical domains often use a grid that offers much higher resolution in the vertical direction than in the horizontal to properly resolve the velocity field. It is tempting to use a similar non-uniform grid for FTLE computations, however, this is likely to cause several problems.

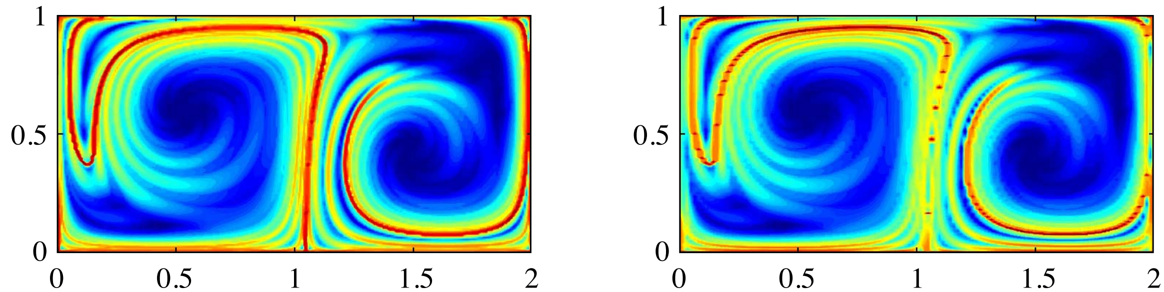


Figure 2.5: The forward FTLE field in a double gyre flow. The left image uses $dx = dy = 1/2^7$, the right uses $dx = 1/2^6$, $dy = 1/2^8$ (same total number of points). The apparent breaks in central the FTLE ridge at right are due to the difference in the x- and y-resolution.

The FTLE typically uses first order finite differences to approximate the stretching in the flow. It is known that the approximation is always an underestimate of the true stretching since a line of fluid will in general be stretched into a curve, but the first order finite difference always estimates the stretching based on a straight line (see Figure 2.4). This underestimate is likely to be worse for lower resolution, thus if the resolution is chosen to be higher in the vertical direction than the horizontal, stretching due to initial displacements in the horizontal direction will be under-emphasized. This can also lead to the appearance of breaks in the ridges while displaying the FTLE ridges.

Figure 2.5 shows the forward FTLE field in a double gyre flow with $A = 0.1$, $\varepsilon = 0.1$, $\omega = 2\pi/10$, and an integration time of $T = 1.2$. This same flow will be revisited as a test case throughout this dissertation and consists of two counter-rotating gyres with a periodic perturbation. For a detailed description of the flow see Section 4.2.6.1. The left image was computed with $dx = dy = 1/2^7$ while the right image was computed with $dx = 1/2^6$ and $dy = 1/2^8$. The breaks seen in the central ridge of the FTLE field at right result when different resolution is used in the x- and y-directions. This effect occurs because particle pairs that straddle the LCS rapidly separate from one another in this system. However, the large spacing used in the x-direction significantly underestimates the stretching in the flow, leading to FTLE values that are around 30% lower when the LCS falls between dx-separated particles than when if falls between dy-separated particles.

Additionally, increasing the resolution gives a much better estimate of the true FTLE values. As long as the particle advection scheme is sufficiently accurate, the FTLE ridge position is determined up to the resolution of the spatial grid, but the FTLE values can change significantly as the resolution is increased. The true stretching may be much larger than what is detected at low or intermediate resolutions. As the resolution is increased, the resulting FTLE values will converge to the true FTLE value. This occurs when the spacing used to compute the FTLE field reaches the thickness of the LCS. In finite time, some finite thickness must always be expected for LCS, it is only in the limit as $t \to \infty$ that manifolds of zero thickness can be exist.

As an example of the dependence of the FTLE on resolution, consider the same double gyre system that is pictured in Figure 2.5. Using a uniform spacing of $dx = dy = 1/2^8$ gives FTLE values of up to 0.3993 and the FTLE field appears to be well resolved. Increasing the resolution reveals the same FTLE ridges, but the maximum FTLE values increase. Choosing a particular point of interest, it is possible to plot the convergence of the FTLE value. I choose the point $(1.064564595, 0.5)$, located on the central ridge seen in Figure 2.5 and plot the FTLE value at this point as a function of grid resolution in Figure 2.6. As the grid resolution increases, the FTLE value increases approximately as $\log(\delta x)$, until the true FTLE value is approached around $\delta x \approx 10^{-6}$. This gives a good indication that the underlying hyperbolic structure has a finite thickness of about 10^{-6} at this point in space and for this integration time ($T = 1.2$). As the integration time, T, is increased, the ridge thickness decreases.

Although it appears that the qualitative features of the FTLE field (i.e. the ridges) are fairly robust to changes in resolution, this is clearly not the case for the actual FTLE values and their derivatives. Methods that rely on the values of the FTLE or its derivatives should take this in to account. Additionally, the FTLE error on the ridges decreases slowly, only as $\log(\delta x)$, if central differences are used to approximate the gradient of the flow map. This is

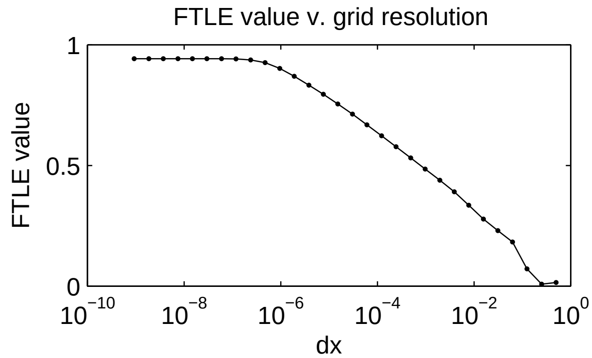


Figure 2.6: The computed FTLE value at $(1.064564595, 0.5)$ (on the central ridge seen in Figure 2.5) as a function of grid spacing. The plot is semi-log in x. The FTLE value increases as $\log(\delta x)$ for decreasing grid spacing before converging to 0.9425.

likely because the FTLE is a linear method and approximates the linear component of the stretching in the flow. Once particles have separated far enough that linear approximations are no longer accurate, significant error can be expected in the resulting FTLE values. The finite size Lyapunov exponent (FSLE, which also helps avoid saturation effects), and nonlinear methods such as M-functions may not have this same issue.

2.2.2 Shear and FTLEs

One of the premises behind the FTLE formulation is that the detected stretching is exponential in time. However, the actual formulation does not ensure this in any way. In fact, any type of stretching will be revealed as elevated FTLE values. It is well known that FTLE ridges may correspond to regions of high shear where there is large linear stretching [35]. In finite time, linear stretching may even completely overwhelm any exponential stretching. This is sometimes undesirable since FTLE ridges due to shear are neither repelling nor attracting structures.

Several attempts have been made to identify only the hyperbolic LCS with varying degrees of success. The method recently proposed by Haller [35] to ensure finite time hyperbolicity seems particularly appropriate, but this variational approach cannot be readily applied to three-dimensional flows [26]. It may also be possible to determine the character of LCS by examining the sharpness of the FTLE ridges. Experience has shown that hyperbolic LCS are typically much sharper ridges (more negative principle curvature) than the FTLE ridges created by shear flow. Additionally, lengthening the integration time typically acts to further sharpen the hyperbolic ridges while this effect may be absent in shear ridges.

2.2.3 Conceptual problems of LCS in three-dimensions

It is common to make an analogy between LCS and the stable and unstable manifolds of hyperbolic fixed points in homogeneous or periodic systems. There are numerous cases where this analogy is appropriate and in many two-dimensional homogeneous or periodic flows the FTLE ridges very closely match the stable and unstable manifolds for the system. However, care should be taken when extending this idea to general flows.

While it is true that in slowly time varying systems this approximation may still be valid, in turbulent or otherwise complex flows, there may not even be an obvious frame of reference and the notion of fixed points no longer makes sense. In fact, one of the most obvious examples where the FTLE field fails to give good results in a simple homogeneous linear saddle.

$$\begin{aligned} \dot{x} &= x \\ \dot{y} &= -y \end{aligned} \tag{2.33}$$

This system (and any linear system) gives a constant FTLE field and admits no LCS despite having a fixed point at the origin and stable/unstable manifolds on the y/x axes as shown in Figure 2.7. However, a change in the frame of reference can result in a different fixed point or no fixed points at all and the apparent structure of the system is changed. For example, a moving coordinate system may be used that moves with a given trajectory. The

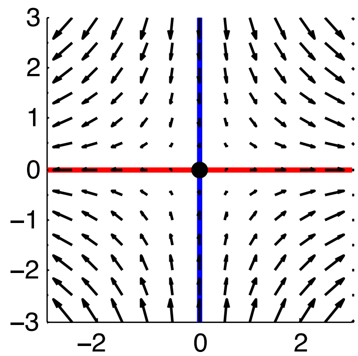


Figure 2.7: The velocity field of the linear saddle given by Equation 2.33. The stable and unstable manifolds of the origin are shown in blue and red respectively.

transformation

$$\tilde{x} = x + e^{t} - 1 \tag{2.34}$$

$$\tilde{y} = y + e^{-t} - 1 \tag{2.35}$$

creates a coordinate system, $(\tilde{x}, \tilde{y})$ that is aligned with the (x, y) coordinates at $t = 0$ and mirrors the movement of a trajectory starting at $(x, y) = (1, 1)$. This coordinate transformation gives the new dynamical system

$$\begin{aligned} \dot{\tilde{x}} &= \tilde{x} - 1 \\ \dot{\tilde{y}} &= -\tilde{y} + 1 \end{aligned} \tag{2.36}$$

that has a fixed point at $(1, 1)$. On the other hand, Lagrangian measures such as the FTLE are frame independent since they deal with the relative motion of trajectories. Trajectories in the initial frame are given by $(x_0 e^{t}, y_0 e^{-t})$ and the resulting FTLE field has values of $\sigma_{t_0}^{T}(\mathbf{x}) = 1$. The transformed coordinates result in trajectories of $((\tilde{x}_0 + 1)e^{t} - 1, (\tilde{y}_0 - 1)e^{-t} + 1)$ and the same FTLE field of of $\sigma_{t_0}^{T}(\tilde{\mathbf{x}}) = 1$.

Additionally, when three-dimensional homogeneous systems admit hyperbolic fixed points the dimensions of the stable and unstable manifold sum to three. Thus, one manifold is a two-dimensional surface and one is a one-dimensional curve. Most Lagrangian techniques for time dependent flows are designed to find codimension-one manifolds (e.g. two-dimensional surfaces in a three-dimensional flow). There is no hope of finding a 1D curve in three-space while looking for two-dimensional ridge surfaces of the FTLE field. Furthermore, codimension-two manifolds do not present barriers to transport. Just as flow may easily pass around a point on a plane, a curve in three-space may easily be bypassed. However, these codimension-two manifolds may still play a very important role in determining the flow structure. For example, the Lorenz system (see Lorenz, [60]) given by

$$\begin{aligned}\dot{x} &= \sigma(y - x),\\ \dot{y} &= x(\rho - z) - y,\\ \dot{z} &= xy - \beta z,\end{aligned} \tag{2.37}$$

where $\sigma = 10$ $\beta = 8/3$, and $\rho = 28$ has a hyperbolic fixed point at the origin with a two-dimensional stable manifold surface and a one-dimensional unstable manifold which manifests as the famous Lorenz attractor (see Fig. 2.8, details on how this attractor was computed are included in Appendix B).

In the Lorenz system, all trajectories that begin near enough to the origin are attracted to the unstable manifold of the origin. This manifold dominates the long term dynamics of all such trajectories, yet it is only a one-dimensional manifold in a three-dimensional domain. The two-dimensional stable manifold is still an invariant manifold and divides the phase space, but it does not reveal much useful information about the orbits at large times.

2.3 Consequences

The FTLE field has often given meaningful results when used to compute LCS. However, it is evident that accurately computing the FTLE or related measures such as the deformation tensor is a difficult task which may require extremely high resolution. Fortunately, the FTLE ridges are accurately located even if there is significant error in the actual FTLE values.

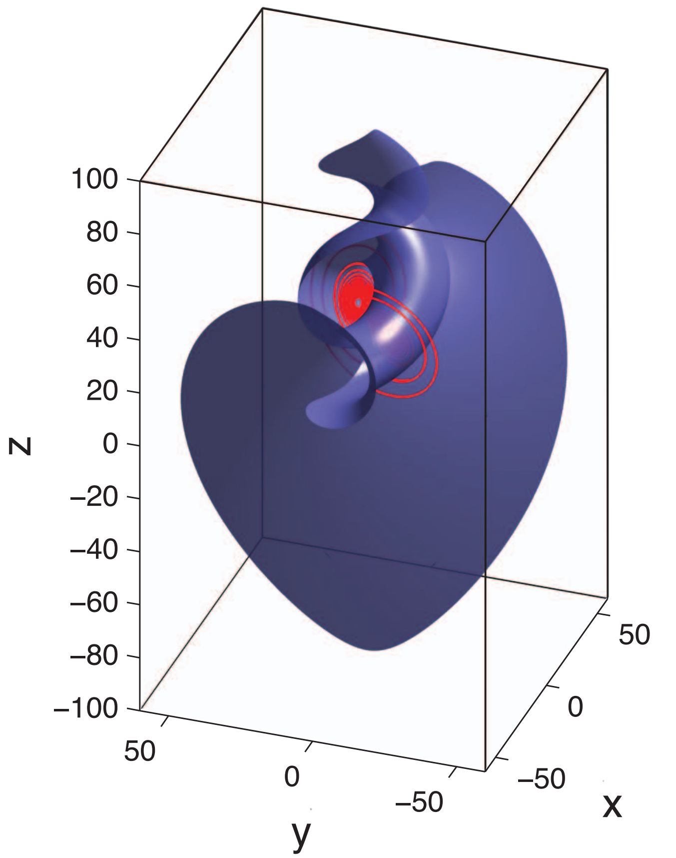


Figure 2.8: A part of the stable (blue surface) and unstable (red curve) manifolds of the Lorenz system. For details on how these manifolds were computed, see Appendix B.

Current LCS techniques are not equipped to detect codimension-two manifolds such as the unstable manifold of the origin in the Lorenz system (Eq. 2.37). Such manifolds may be of importance in compressible flows or flow separation/attachment on an airfoil [20]. These structures may be more easily located as attracting structures under particle advection.

The above discussion points out many factors that must be considered by LCS users. The coordinate system considerations discussed in Section 2.1 are easily dealt with by using the appropriate gradient operator. However, the remaining issues are not so easily solved. In this dissertation, the choice of FTLE resolution is handled by ensuring that the LCS locations do not change with increasing grid resolution for all applications. The ridge tracking algorithm presented in later chapters does not rely on accurately computing the FTLE values, as long as the ridges are accurately located. I make no attempt to distinguish between shear LCS and hyperbolic LCS in this dissertation. Rather, all LCS studies are paired with an analysis of drifter trajectories near the LCS to determine the true flow behavior near the important structures. Additionally, regions of large shear may bound important flow structures (e.g. jets and currents) so it is not necessarily advisable to filter them out. Finally, the conceptual issues involved in drawing analogies between stable and unstable manifolds of hyperbolic fixed points and LCS in three dimensions are beyond the scope of this dissertation.

In all LCS computations, one must be aware that there is some uncertainty in the results and the precise cause of the LCS (i.e. high shear versus hyperbolic separation). However, numerous examples such as those cited in the introduction have borne out the usefulness of LCS for identifying flow structures and fluid transport. Supplementing LCS analysis with an examination of nearby particle trajectories ensures a full and correct understanding of any LCS results as particle trajectories can be used to determine the causes of the fluid stretching identified by LCS and ensure low flux through LCS.

Chapter 3

Lagrangian coherent structures generated by jellyfish feeding and swimming

3.1 Introduction

Two distinct types of jellyfish propulsion are well known, corresponding to the two different jellyfish morphologies shown in Figure 3.1 [6]. Prolate species such as *Sarsia tubulosa* primarily use a jetting type of propulsion with large jet velocities immediately behind the velar aperture. Their swimming is characterized by a quick acceleration during the contraction phase of swimming followed by gliding with relatively small acceleration. However, the once widely accepted jetting model fails to explain the swimming patterns seen in prolate species such as *Aequorea victoria*. These jellyfish use a paddling or rowing motion to swim and produce more diffuse vortex pairs that are shed from the bell margins during the contraction phase.

Prolate, jetting jellyfish retract their tentacles during swimming and feed by extending their tentacles while drifting passively [62]. Swimming is used to escape predators or to ambush prey. Swimming and feeding are disparate activities since extending the tentacles during swimming could greatly decrease swimming performance. On the other hand, oblate, paddling jellyfish leave their tentacles extended during swimming and the vortices produced during swimming travel through the extended tentacles [6,7,10,12,27]. Prey in the fluid near the bell region of paddling jellyfish have even been observed to be carried into contact with the tentacles by the vortices formed during swimming [13]. For these reasons, swimming complements feeding in paddling jellyfish by helping to draw prey into the tentacles.

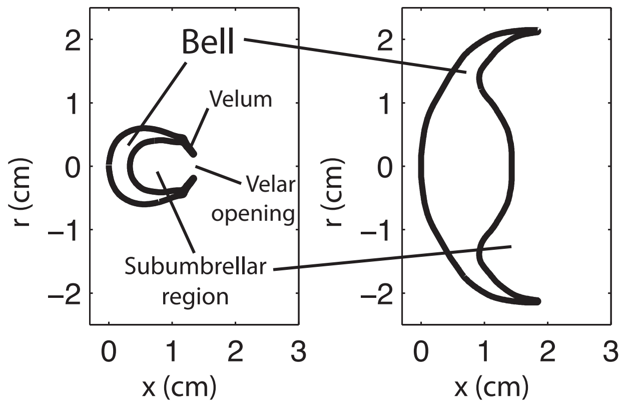


Figure 3.1: Prolate (left) and oblate (right) jellyfish body shapes corresponding to the jetting and paddling swimming types discussed in this chapter.

It has been shown that rowing propulsion is a necessary adaptation for larger jellyfish due to morphological constraints and energy efficiency considerations. For jetting propulsion, the force necessary for propulsion increases faster with size than the available muscle force to provide the jetting motion [16]. Oblate jellyfish make up for this by employing paddling type propulsion. Models show that the production of stopping vortices during the relaxation phase of paddling type propulsion allows large jellyfish to swim effectively despite their morphological constraints [16]. Specifically, the stopping vortex partially cancels the starting vortex, reducing the induced drag on the oblate jellyfish and increasing swimming efficiency. The smaller, prolate species have lower drag due to their shape and further decrease drag by retracting their tentacles while swimming [7]. These factors, combined with more rapid bell contractions, make jetting jellyfish much more proficient swimmers than their oblate relatives [18].

Recently, jet and vortex propulsion have become a focus in the areas of underwater maneuvering and locomotion of bio-engineered vehicles. A vortex thruster loosely mimicking jellyfish propulsion was proposed by Mohseni [66, 67]. These vortex thruster have been built and tested on underwater vehicles and are further discussed in Krieg and Mohseni [45]. Similar to a swimming jellyfish, these vortex thrusters draw in fluid and eject it through the same orifice and are capable of producing very strong vortices (formation time of up to 15, see Section 3.3.1 for a discussion of formation numbers). Mohseni et al. [68] and Dabiri and Gharib [17] also reported numerical and experimental results for jets formed from a nozzle with temporally varying exit velocity and diameter respectively.

This chapter presents the LCS seen in the results of numerical simulations of jellyfish swimming as well as several examples of particle motion in the resulting flow. The jellyfish examined are *Aequorea victoria*, a paddling or rowing type of jellyfish and *Sarsia tubulosa*, a jetting type of jellyfish. The velocity field was computed by Sahin and Mohseni [80, 81] in previous work and is used here to compute the LCS and analyze the flow transport behavior and fluid structures. For completeness, a very brief overview of the numerical methods used by Sahin and Mohseni is presented in Section 3.2.2. The use of computational fluid dynamics (CFD) data instead of an empirical velocity field from digital particle image velocimetry (DPIV) or similar, results in higher resolution of the LCS as well as greater accuracy in subsequent calculations. Additionally, there are significant difficulties in obtaining high quality results from DPIV for swimming jellyfish. DPIV results are only available for the time during which the jellyfish is properly oriented within the field of view, perhaps only a few swimming cycles depending on many factors. Additionally, the resolution and accuracy of DPIV depends on the concentration of particles in a given region. In general, the distribution of particles may be highly non-uniform. Particles are drawn toward certain flow structures, just as dye is drawn into vortices in dye visualization experiments, but other areas of the flow may be left with few particles. None of these difficulties are present when using CFD data. To generate the CFD runs, it is only necessary to capture a complete swimming cycle on video. The periodic swimming motion may then be determined up to the resolution of the camera used and iterated for as many swimming cycles as desired.

As expected, *S. tubulosa* produces strong vortices along the axis of symmetry. These vortices move quickly away from the jellyfish, providing a high momentum transfer for rapid swimming while negating opportunities for feeding while swimming. In fact, if *S. tubulosa* were to extend its tentacles during swimming, they would create additional drag and could negatively impact swimming performance. Conversely, the structures formed by the paddling jellyfish, *A. victoria*, transport fluid from the outer bell surface to linger in the tentacle region, enhancing feeding opportunities since the flow passes through the region where the tentacles drift.

Previously unobserved flow structures in the subumbrellar region of *S. tubulosa* are revealed by the LCS in this study. As previously mentioned, *A. victoria* produces a starting and stopping vortex during each swimming cycle. Since these vortices are ejected together during the contraction phase they interact and influence each other. *S. tubulosa* also produces a stopping vortex during the relaxation phase. However, only the starting vortex is ejected during contraction while the stopping vortex dissipates within the subumbrellar cavity.

Finally, the pressure on the subumbrellar wall of both jellyfish as well as the velocity profiles in the wake and across the velar opening are examined. *Sarsia tubulosa* produces a nearly uniform jet through the velar opening and an equally uniform pressure along the subumbrellar wall. This type of swimming could be approximated by a slug model or a piston-cylinder arrangement due to the nearly uniform velocity profile at the velar opening. On the other hand, *A. victoria* produces a more complicated wake and pressure profile.

3.2 Methods

3.2.1 Jellyfish motion.

The motion of each jellyfish was determined from videos of physical specimens of the swimming jellyfish in other work, it is briefly discussed here for completeness. These videos were provided by Dr. Sean Colin (Roger Williams University) and are further discussed in [81]. The jellyfish were placed in filtered sea water within a glass vessel of sufficient size to allow each jellyfish to swim freely. The outline of the bell was illuminated using a planar laser directed through the central axis. Fluorescein dye was injected near the bell to enhance the illumination.

After recording, each frame of the video was analyzed and the body motion of the jellyfish was determined. The geometry of the jellyfish was approximated using NURBS curves [74] and Fourier-series interpolation in time was used to create a numerical model of the periodic contraction of the swimming jellyfish. In contrast to the relatively short time frames allowed by using DPIV, this model allows the analysis of many periods of swimming via numerical solution for the flow around the jellyfish.

3.2.2 Numerical procedure.

For completeness, a brief overview of the numerical procedure used in the computation of the velocity field produced by the swimming jellyfish is included here. This work was done by collaborators and is not a new contribution of this dissertation. For complete details of the numerical procedure, including code validation, see Sahin et al. [81]. The flow field was computed based on the periodic swimming model derived from the videos of swimming jellyfish. The surrounding velocity field was computed using an arbitrary Lagrangian-Eulerian (ALE) [39] method developed for this purpose. In this method, the mesh follows the moving boundary between the fluid and the jellyfish body and the cylindrically symmetric governing equations are solved on a moving, unstructured quadrilateral mesh. The pressure is solved on a staggered grid, eliminating the need for pressure boundary conditions since pressure is defined only at interior points. The mesh motion is determined by solving the linear elasticity equation at each time to avoid remeshing [23, 43] and the linear systems produced by the discretization are solved using the GMRES method [78] combined with several preconditioners.

The mesh type (an unstructure, moving quadrilateral mesh) creates significant complications for velocity interpolation during particle advection for LCS computations since even locating the mesh element that contains a given point is non-trivial. To address this issue efficiently, an alternating digital tree (ADT) is used to search the domain for the element that contains a given drifter particle [4]. The ADT recursively divides the space in half so that at each node in the tree, only one branch must be searched. Since the nodes of the mesh elements are listed in counter clockwise order and the elements are convex, a point $\mathbf{p}$ is inside (including the boundary) an element if and only if

$$\hat{\mathbf{z}} \cdot [(\mathbf{v}_i - \mathbf{p}) \times (\mathbf{v}_{(i \bmod 4)+1} - \mathbf{v}_i)] \geq 0 \;\forall\; i \in \{1, 2, 3, 4\}$$

for vertices $\{\mathbf{v}_i\}$.

Once the element containing a particle is found, the velocity must be interpolated onto the drifter particle. Since the elements are generally non-rectangular, simple linear interpolation is difficult possible. Instead, perspective projection is used to map the quadrilateral element and the point of interest onto the unit square. Bilinear interpolation is then used to approximate the velocity at a point.

3.3 Results

3.3.1 *Sarsia tubulosa*

S. tubulosa employs a jetting type of propulsion. The swimming cycle consists of three phases, a rapid contraction, relaxation, and a coasting phase. The jetting nature of *S. tubulosa*'s propulsion, forming a single vortex ring with each swimming pulse, is clearly reflected in the LCS seen in Figure 3.2. During the contraction phase, a vortex is ejected in the positive x-direction along the axis of symmetry. This will be referred to as the starting vortex. A notable feature of this Figure 3.2 is the presence of very complex flow structures within the subumbrellar region of the jetting jellyfish. This region is very difficult to view experimentally with DPIV or other techniques since it is occluded by the jellyfish body. However, the numerically generated velocity data includes the bell interior. During the relaxation phase of swimming (Figure 3.2(c)), a stopping vortex of opposite sense to the ejected vortex forms within the subumbrellar cavity. In this jellyfish, the presence of the velum traps the stopping vortex within the subumbrellar cavity. Since the stopping vortex is not ejected, it cannot interact with the ejected starting vortex. This is the first time this stopping vortex has been observed in jetting jellyfish, likely due to the difficulty in imaging the subumbrellar cavity of jetting jellyfish during experiments. In fact, it has been recently stated that no stopping vortex is formed in a jetting swimmer [94]. This vortex may in fact be present, but difficult to detect in jetting as well as paddling jellyfish.

S. tubulosa is very efficient at producing a strong vortex. The formation time of the vortices quantifies this efficiency. Gharib et al. defined the dimensionless formation time as

$$T^* = \frac{\overline{U_e} t}{D_e} \tag{3.1}$$

for a jet of velocity U_e through a nozzle of constant diameter D_e over a time t where the overbar denotes a running average [32] . However, for a non-constant exit diameter it is necessary to consider an integral form of this equation. Calculations for a slug of fluid ejected from an orifice with time varying diameter and velocity were presented by Mohseni [65]. Using a similar approach, Dabiri and Gharib [17] derived

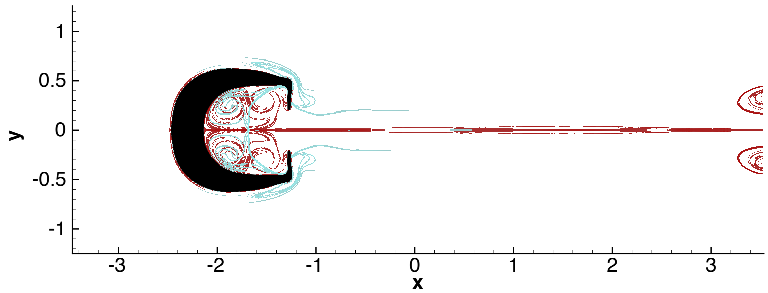
$$T^* = \int_0^t \frac{U_e(\tau)}{D_e(\tau)} d\tau \tag{3.2}$$

In the case of constant density flow, such as a jellyfish swimming in water, conservation of mass allows this to be expressed as

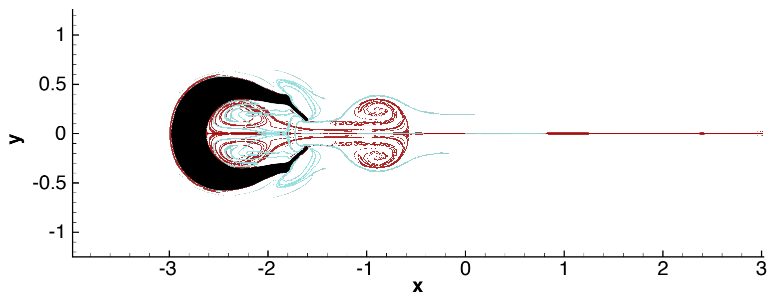
$$T^* = \int_0^t \frac{U_e(\tau)}{D_e(\tau)} d\tau = -\frac{\pi}{4} \int_0^t D_e(\tau) \frac{dV_c(\tau)}{d\tau} d\tau \tag{3.3}$$

where V_c is the volume of the cavity, i.e. the subumbrellar volume.

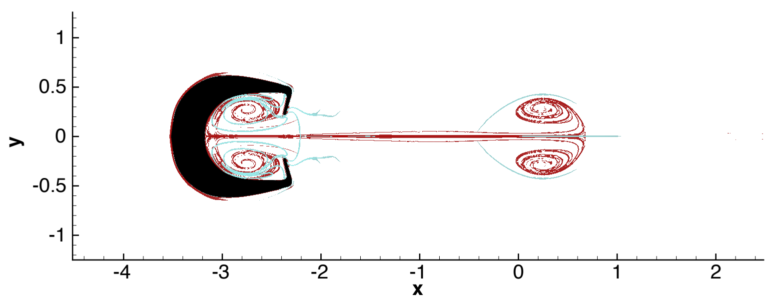
A larger formation time indicates the formation of a more energetic vortex, where an impulsively started piston-cylinder arrangement produces a formation time of about four, after which, additional fluid expulsion results in a trailing jet behind the vortex [32]. *S. tubulosa* produces a vortex with formation time $T^* \approx 7$. This large formation number was expected as discussed by Dabiri et al. [15] and is made possible largely by the decrease in diameter of the velar opening during the contraction phase of swimming (see Figure 3.3).



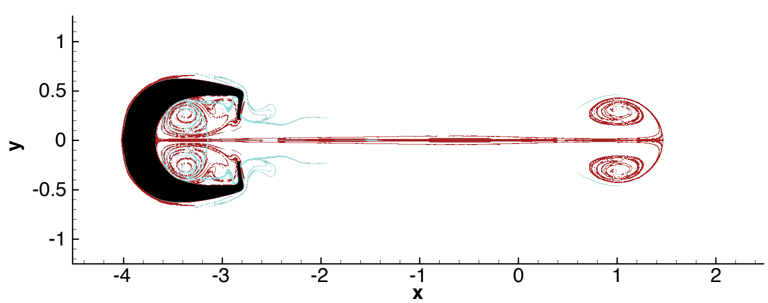
(a) time = 2.01s



(b) time = 2.26s



(c) time = 2.51s



(d) time = 2.76s

Figure 3.2: LCS for one swimming cycle of *Sarsia tubulosa*. The forward LCS are shown in cyan and the backward are dark red.

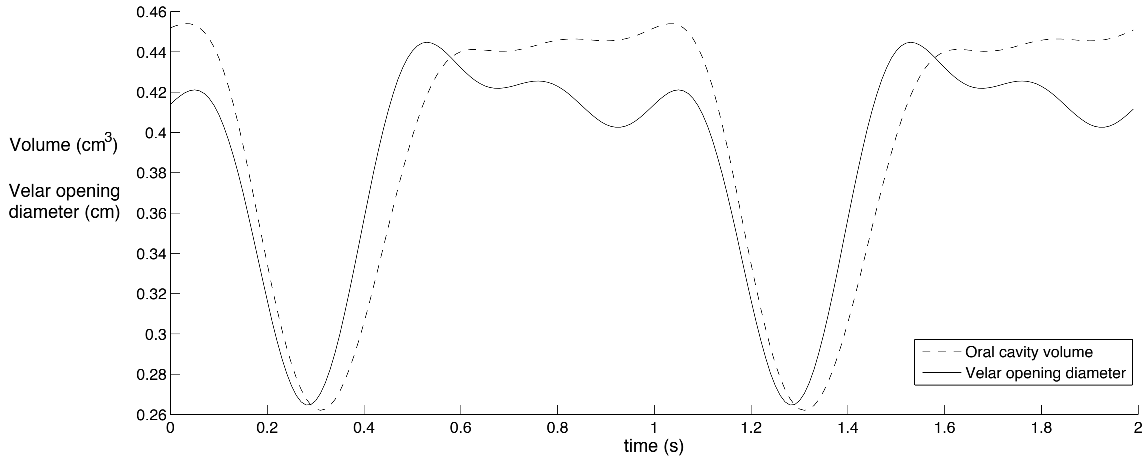


Figure 3.3: The subumbrellar volume and velar opening diameter during two swimming cycles for *Sarsia tubulosa*

A trailing jet represents a decreased efficiency of momentum and energy transfer [46]. The vortex ring produced by *S. tubulosa* has no such trailing jet, meaning the vortices are formed with maximal efficiency.

The axial velocity in the wake is plotted in Figure 3.5. This plot represents one swimming cycle and shows that the velocity disturbances are concentrated near the axis of symmetry. In fact, the disturbances quickly decay to negligible levels beyond the jellyfish radius. As a vortex is produced, a large axial velocity appears at the velar opening of the jellyfish which then decays as the vortex moves away from the jellyfish. Also, the velocity in the wake is almost entirely positive (to the right) due to the jetting nature of the swimming. The strong jet can also be seen in the plot of axial velocity across the velar opening as well as the pressure on the subumbrellar wall (Figures 3.6 and 3.7). Notice in particular, that the velocity across the velar opening forms a very uniform jet, with only a small shear layer near the velum. This is in contrast to the profile for *A. victoria*, discussed below. Additionally, the swimming cycle for *S. tubulosa* is logically divided into 3 parts, a strong contraction, a relaxation phase and then a brief coasting phase before the next contraction. These three phases correspond to a decrease then increase in the subumbrellar volume followed by a time

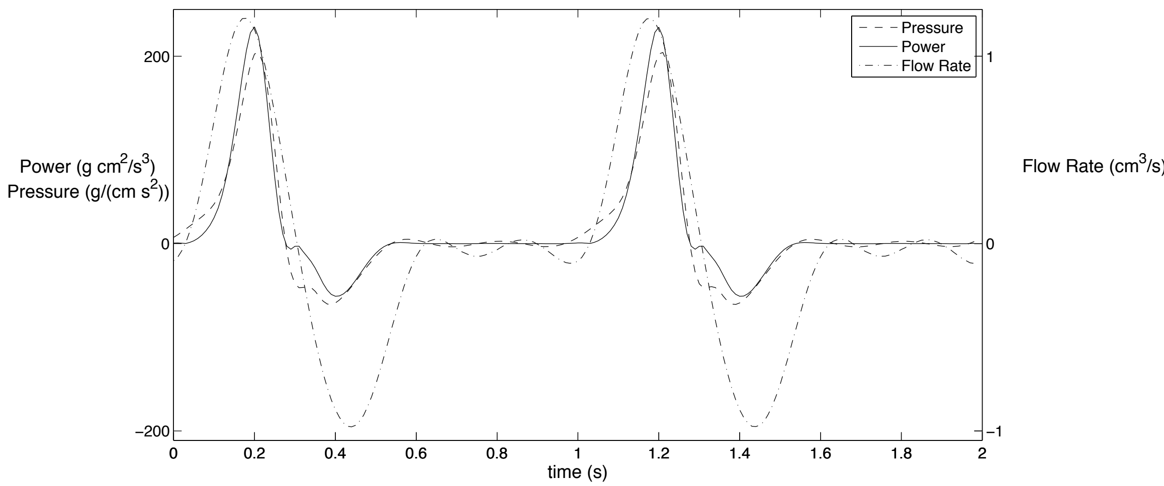


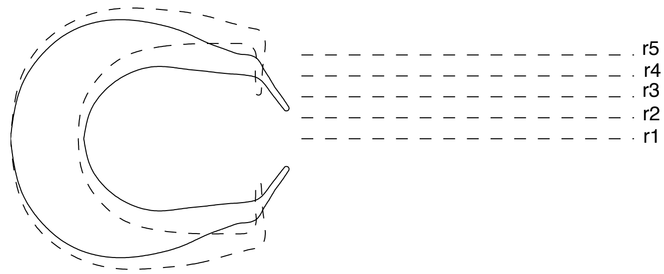
Figure 3.4: The power output (P_{out}), pressure (p) and flow rate (Q) during two swimming cycles for *Sarsia tubulosa*

of nearly constant volume as seen in Figure 3.3.

By noting that *S. tubulosa* is a jetting swimmer, propelling itself via a jet created by pressurizing the subumbrellar cavity, one may estimate the power output as for a biological (such as a heart) or mechanical pump as done by O'Dor for squid [69]. The power output is simply given by $P_{out} = pQ$ where p is the subumbrellar pressure and Q is the jet flow rate. Since the pressure and jet velocity are very uniform in space (see Figures 3.6 and 3.7), one may use the average value at each time step without losing much accuracy. Q, p and P_{out} are plotted in Figure 3.4.

The average power output is found to be about 16 g·cm/s^3. To account for body size, power is divided by mass to the 5/3 power (see Daniel [18]) where this *S. tubulosa* has a mass of about 0.65 g as calculated from the volume of the body and the assumption of neutral buoyancy (1 g/cm^3). This results in an average power output of about 0.33 W·kg$^{-5/3}$ which is within the range of 0.2-0.75 W·kg$^{-5/3}$ for the experimentally measured power outputs for the jellyfish *Gonionemus vertens* and *Stomotoca atra* as reported by Daniel [19].

Since flux across LCS is typically negligible (as mentioned in Chapter 1 this should be



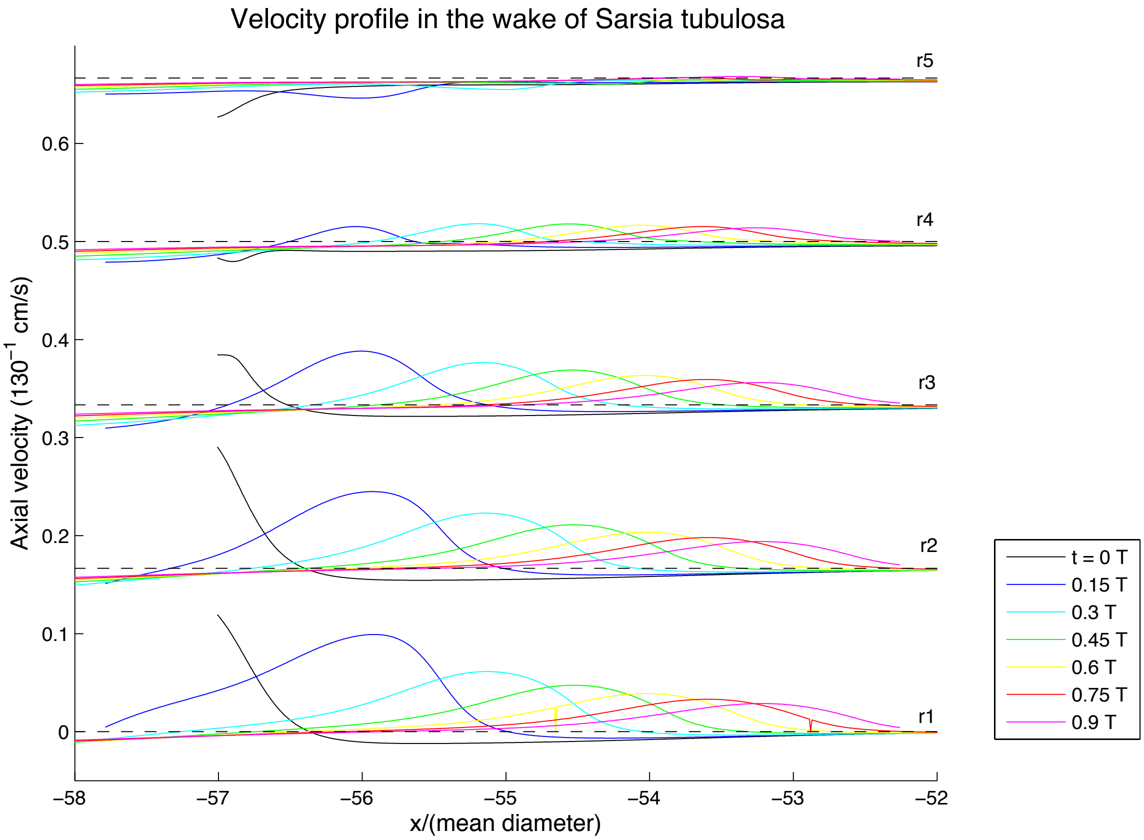
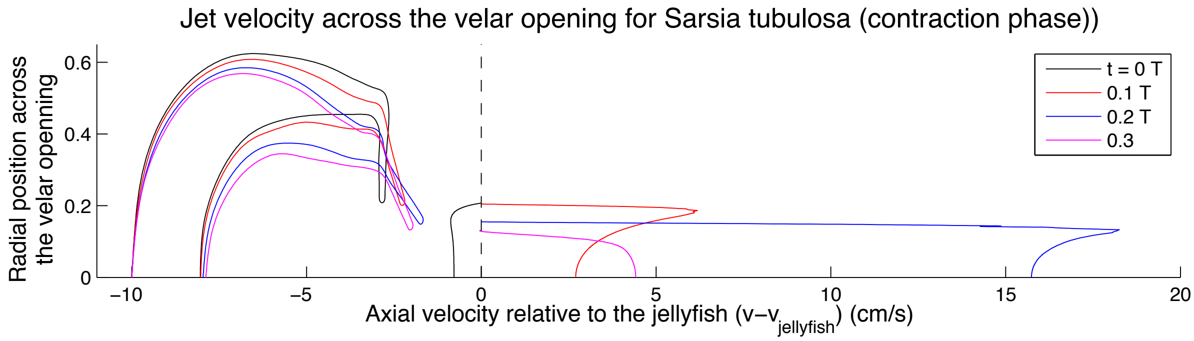
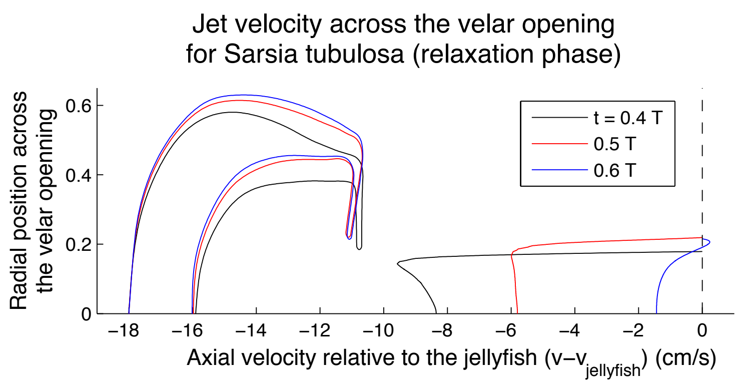


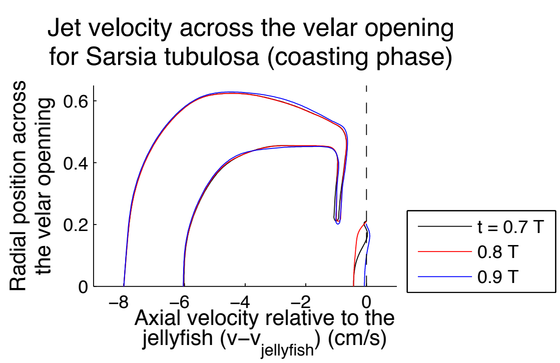
Figure 3.5: The axial velocity along lines of constant radius in the wake of *Sarsia tubulosa*. The radii chosen, along with the jellyfish geometry at minimum and maximum diameter are shown in the figure on the left. The radius has been scaled by the mean diameter and the velocity has been scaled by a factor of 130^{-1} to fit within the figure. Many times, from $t = 0T$ to $0.9T$ ($T = 1$s $=$ one paddling cycle), representing one paddling cycle are plotted, with darker lines representing the most recent times



(a) Contraction phase

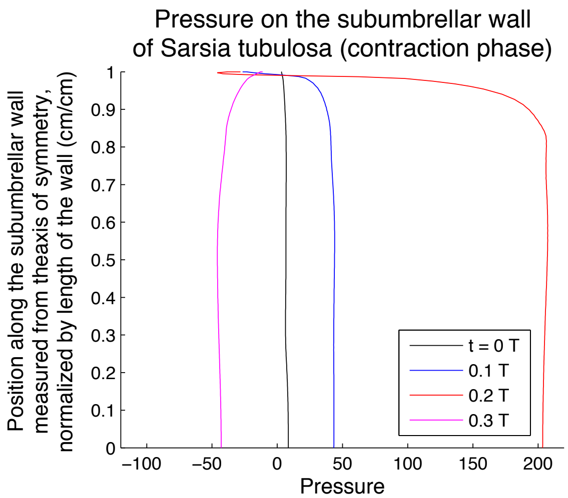


(b) Relaxation phase

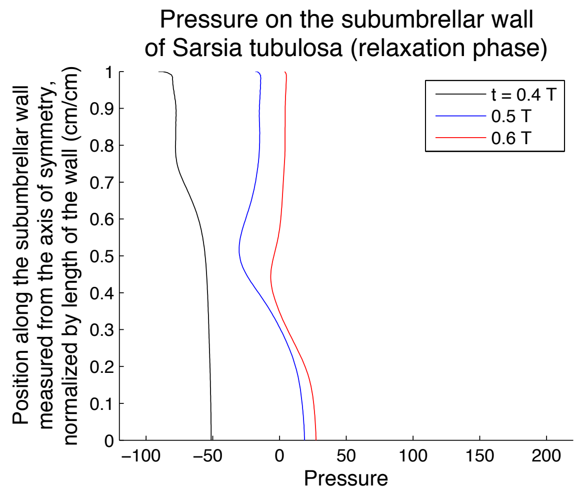


(c) Coasting phase

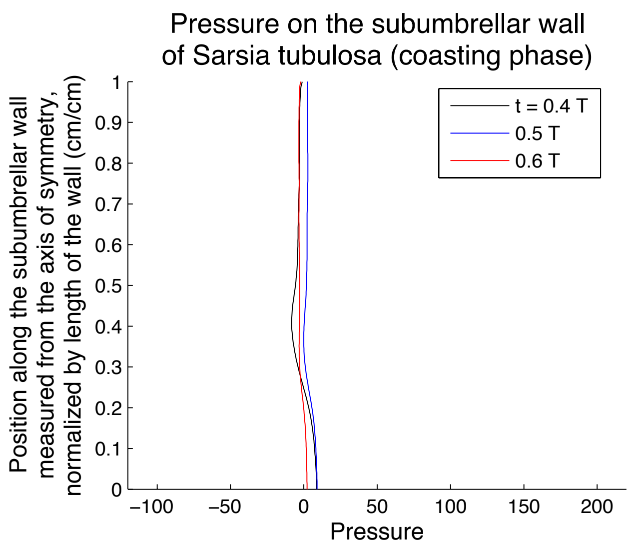
Figure 3.6: The axial velocity of the flow relative to the jellyfish across the velar opening of *Sarsia tubulosa*. Times are presented for one complete paddling cycle.



(a) Contraction phase



(b) Relaxation phase



(c) Coasting phase

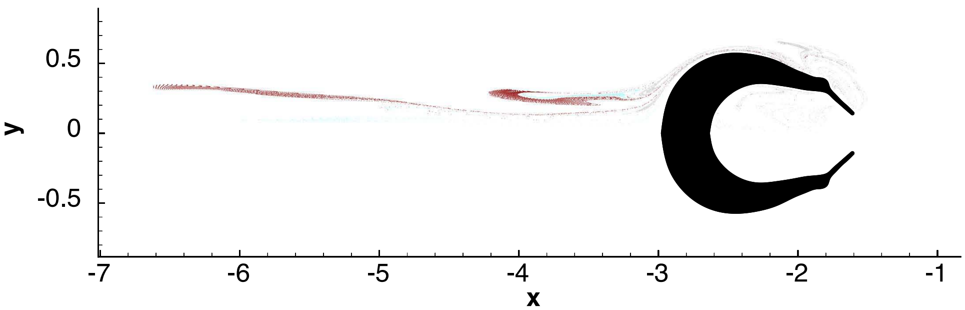
Figure 3.7: The pressure on the subumbrellar wall of *Sarsia tubulosa* for one swimming cycle. The position is normalized by the length of the subumbrellar wall and pressure is given in g cm^{-1} s^{-2}).

verified using particle trajectories or other methods), they largely govern transport in a given flow [85]. In fact, the intersections of forward and backward LCS often divide the flow into lobes that have distinct transport characteristics. Using these lobes, it is simple to determine where the particles in a vortex come from. Figures 3.8-3.10 show the motion of passive tracers placed in the flow of the swimming *S. tubulosa*. In this jetting jellyfish, the LCS are complicated and evolve very quickly during the contraction phase so the groups of tracers that end up in a vortex are not clearly separated upstream of the jellyfish (Figure 3.8(a)). As the jellyfish swims, the gray group of tracers is pulled into the subumbrellar cavity while the cyan and dark red groups collect outside the bell (see Figure 3.9(b)). Then, just before the contraction phase, the cyan group is pulled into the subumbrellar cavity while the dark red group remains outside (Figure 3.10(a)). The dark red and cyan groups of tracers merge and are ejected with the vortex ring, at which point the dark red group of tracers begins to be wrapped into the vortex ring as well (Figures 3.10(b)-3.10(c)). The ejected drifters travel away from the jellyfish very quickly as they are carried with the traveling vortex ring.

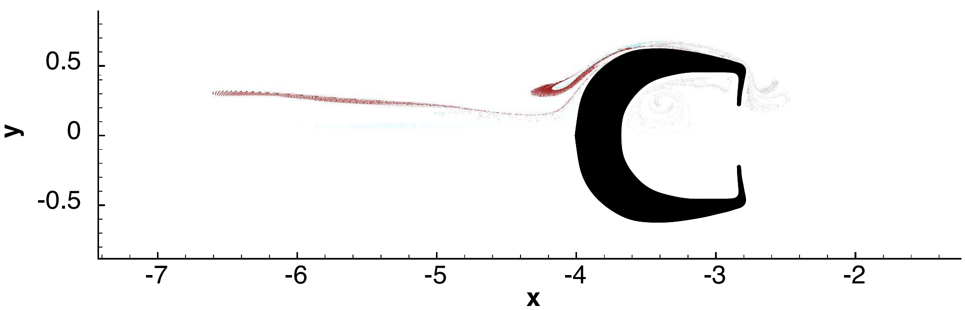
The Strouhal number is used to quantify the rate of separation of a vortex from the jellyfish. It is defined as $St = \frac{fL}{v}$ where f is frequency, L is the mean jellyfish radius and v is the rate of vortex separation from the jellyfish. *S. tubulosa* has a Strouhal number of about 0.10, meaning that the vortices separate from the jellyfish by about 10 radii per swimming cycle. This presents little opportunity for feeding during swimming and is discussed in more detail below.

3.3.2 *Aequorea victoria*

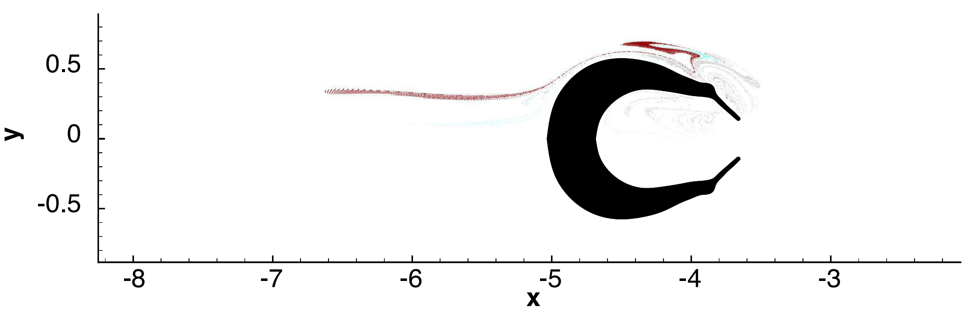
The paddling propulsion of *A. victoria* is very different from the jetting propulsion of *S. tubulosa*. This paddling or rowing propulsion produces two vortices during each swimming cycle that are ejected together during the contraction phase. This results in more energy efficient swimming, but cannot provide the fast accelerations and rapid swimming seen in jetting jellyfish. The LCS produced by *A. victoria* are markedly different as well. The



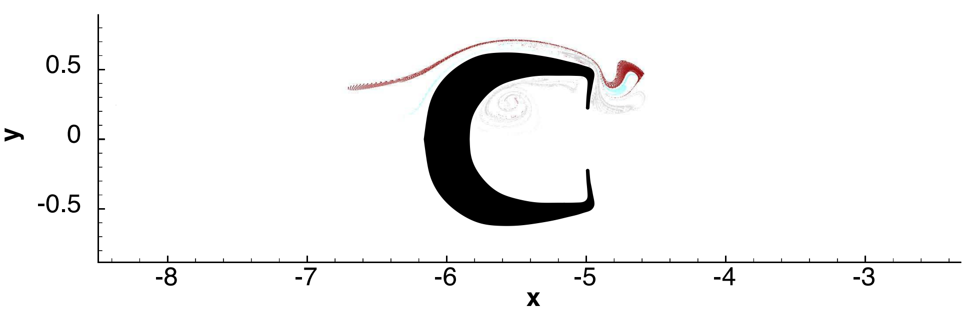
(a) time = 2.26s



(b) time = 2.76s

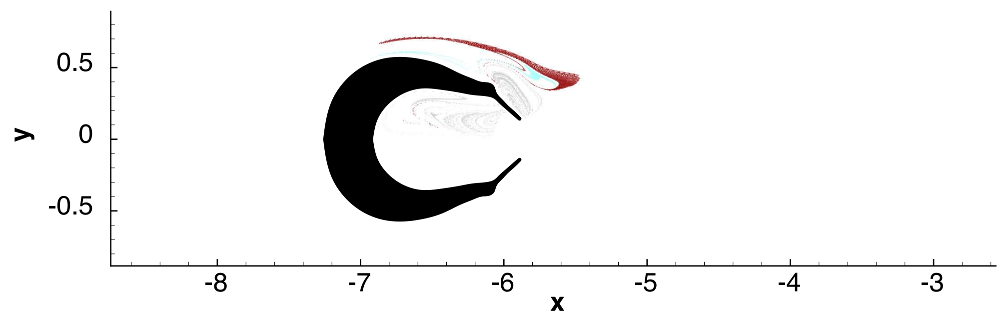


(c) time = 3.26s

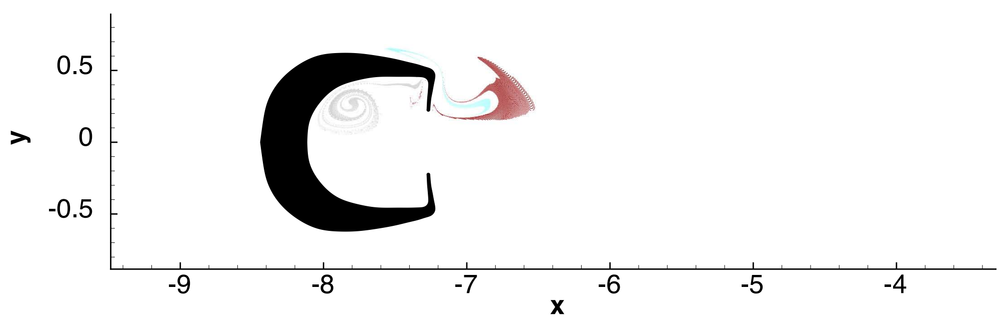


(d) time = 3.76s

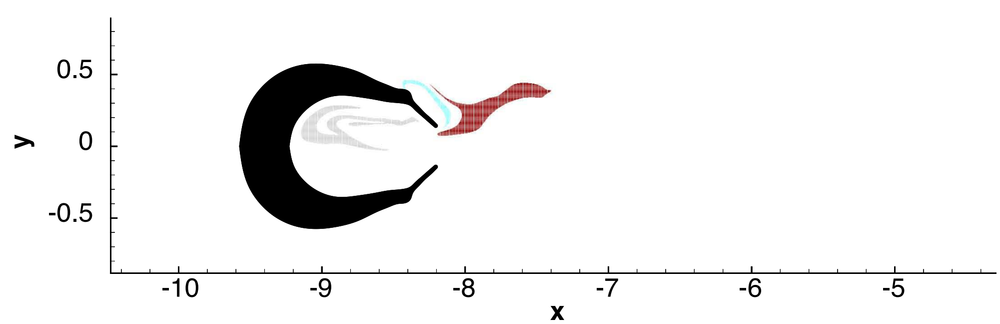
Figure 3.8: Passive tracer particles for *Sarsia tubulosa*. There are three groups of tracer particles, colored gray, cyan and dark red, which are ejected as part of the same vortex.



(a) time = 4.26s

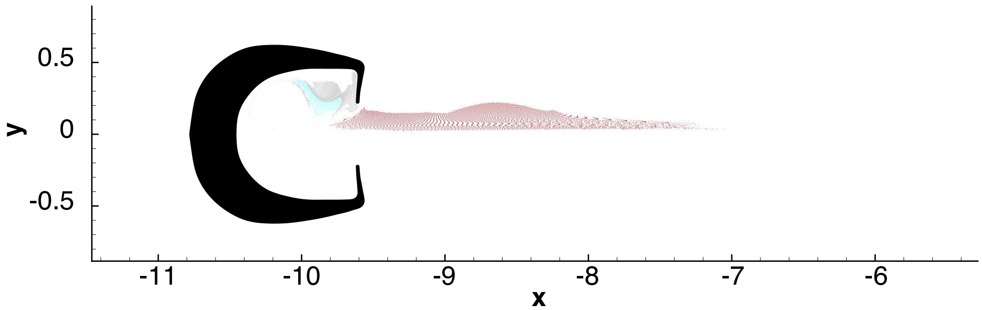


(b) time = 4.76s

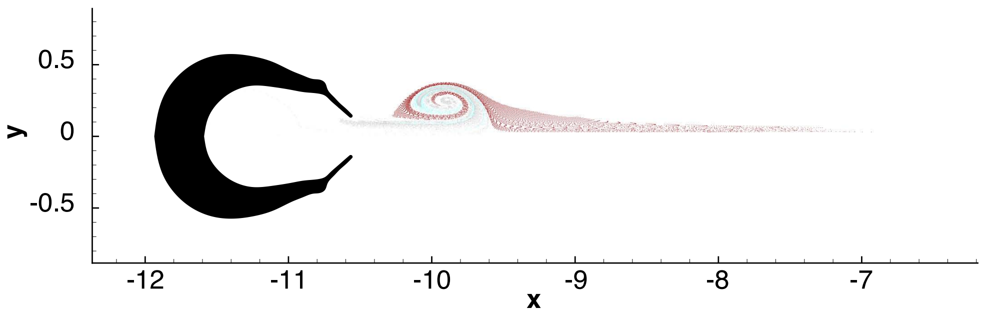


(c) time = 5.26s

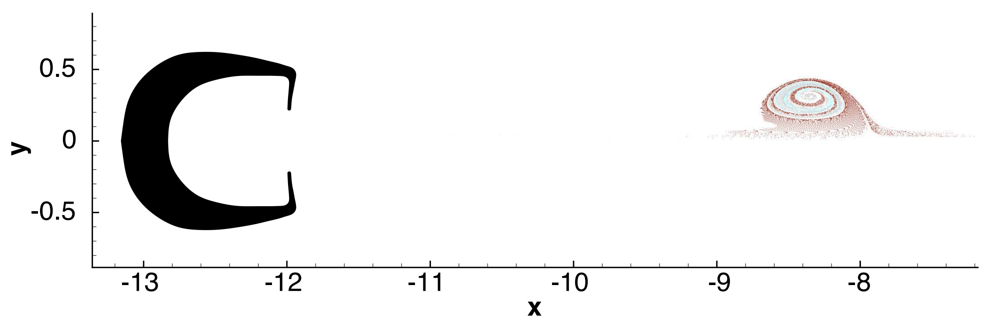
Figure 3.9: Passive tracer particles for *Sarsia tubulosa*. There are three groups of tracer particles, colored gray, cyan and dark red, which are ejected as part of the same vortex.



(a) time = 5.76s



(b) time = 6.26s



(c) time = 6.76s

Figure 3.10: Passive tracer particles for *Sarsia tubulosa*. There are three groups of tracer particles, colored gray, cyan and dark red, which are ejected as part of the same vortex.

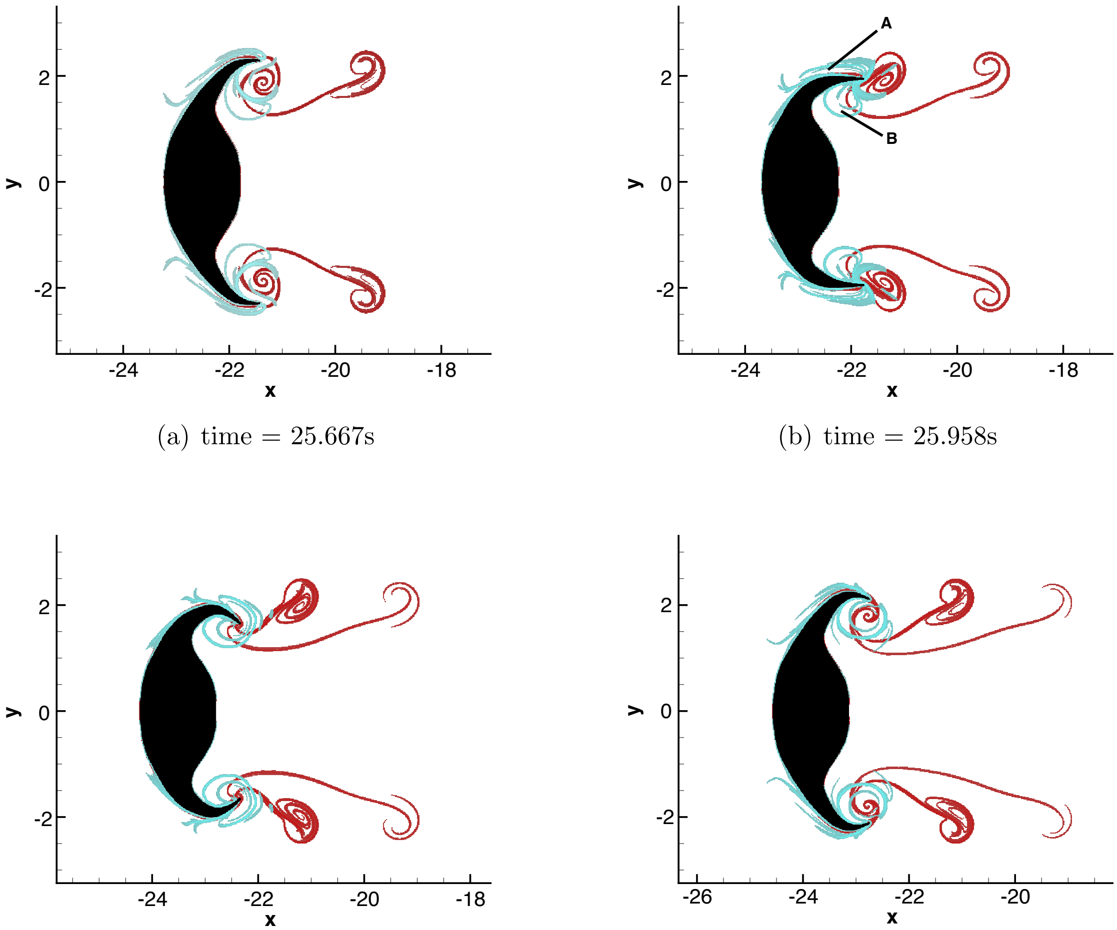
forward and backward LCS can be seen in Figure 3.11.

Since *A. victoria* does not use a jetting motion to swim, the concept of a vortex formation time does not make much sense. However, naively defining a vortex formation time by considering the volume in the subumbrellar region results in a formation number of $T^* \approx 0.07$. Clearly *A. victoria* has not optimized it's swimming to produce the most powerful vortices since even nozzles of constant diameter are capable of producing vortices with $T^* \approx 4$ [32].

The axial velocity in the wake is plotted in Figure 3.12. Near the axis of symmetry, the flow moves toward the jellyfish, but near the bell margins, the axial flow velocity oscillates with the swimming strokes. Beyond the bell margins, the flow disturbances quickly decay, leaving the flow far from the axis of symmetry largely undisturbed. The axial velocity across the velar opening and the pressure on the subumbrellar surface of the jellyfish are also presented in Figures 3.13 and 3.14. Note that again there are large changes in velocity across the velar opening as well as the pressure on the jellyfish occur near the bell margins. Additionally, the swimming cycle for *A. victoria* is equally split into a contraction phase and a relaxation phase as it uses a rowing motion to propel itself forward.

These results are all significantly different than those seen for *S. tubulosa* which produced velocity disturbances only near the axis of symmetry and displayed a very uniform pressure profile and velocity across the velar opening. Additionally, there is no coasting phase for *A. victoria*. Each swimming cycle is a continuous transition from contraction to relaxation and back to contraction. Due to these fundamental differences in locomotion, it is not possible to calculate the power output of *A. victoria* in the same way as was done for *S. tubulosa*.

The resulting forward and backward LCS for *A. victoria* are shown in Figure 3.11. The backward LCS show attracting material manifolds and reveal vortical structures that look very similar to the results of dye visualization experiments as seen in Dabiri et al. [16] and Costello et al. [11]. As discussed by Dabiri et al. [16], the paddling type of jellyfish



(a) time = 25.667s

(b) time = 25.958s

(c) time = 26.250s

(d) time = 26.542s

Figure 3.11: LCS for one swimming cycle of *Aequorea victoria*. The forward LCS are shown in cyan and the backward are dark red.

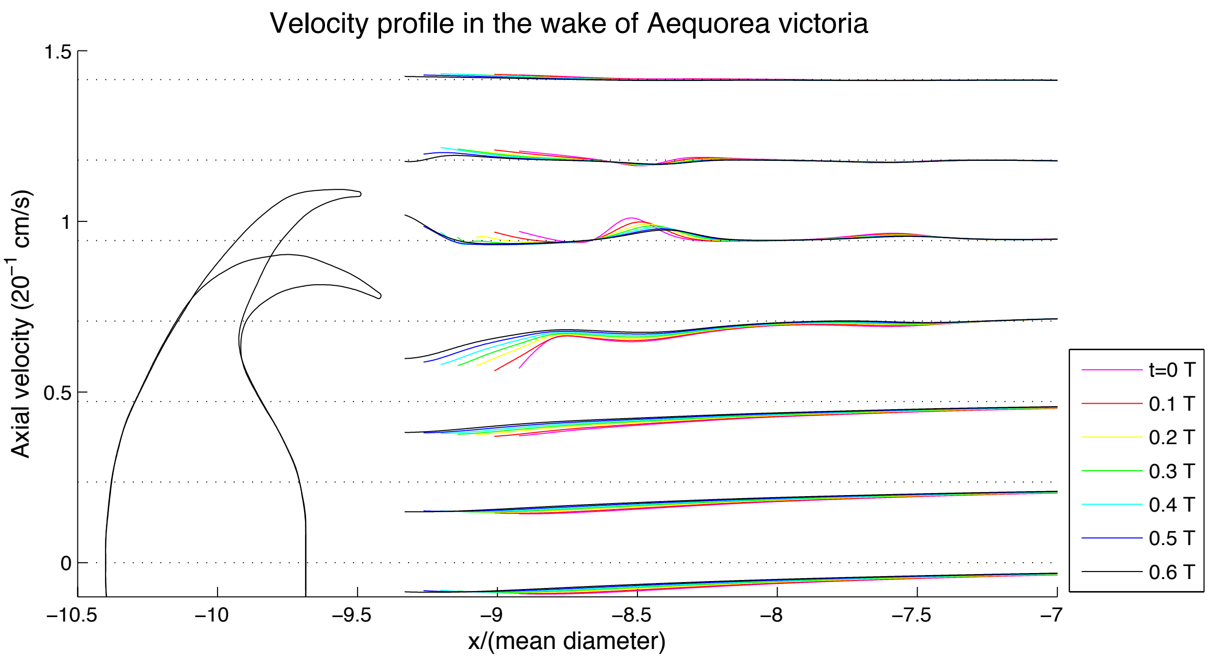
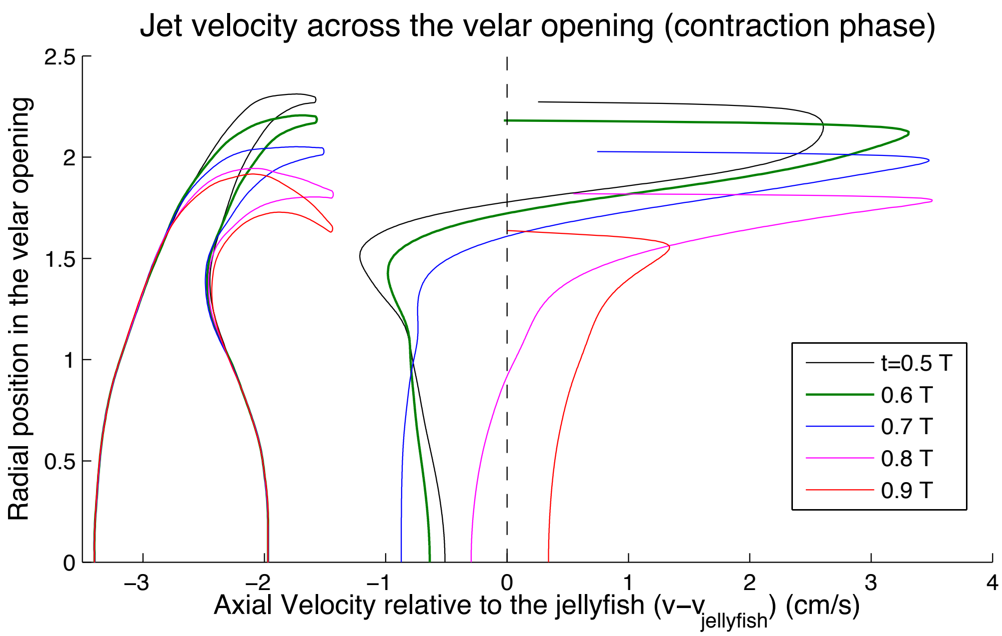
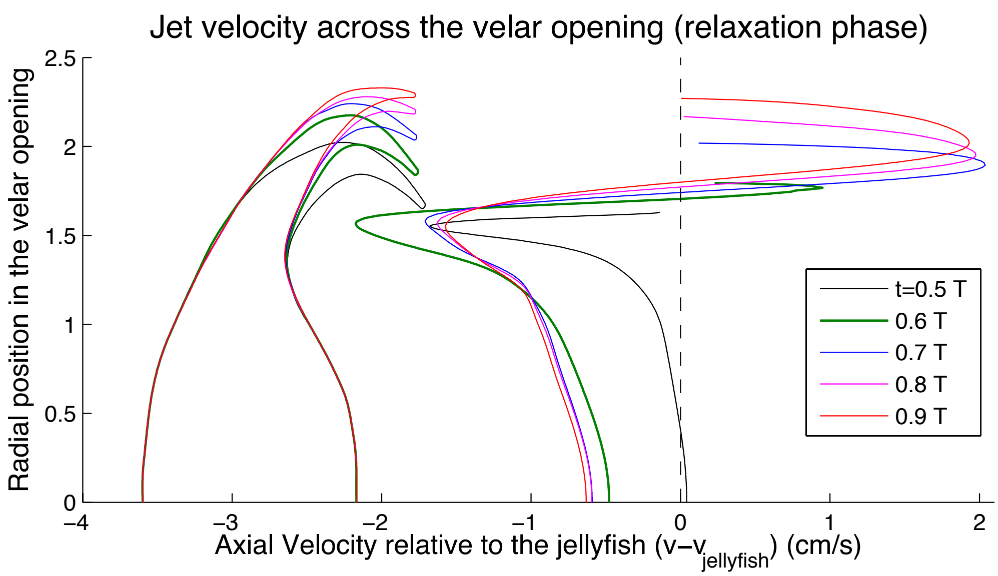


Figure 3.12: The axial velocity along lines of constant radius in the wake of *Aequorea victoria*. The radius has been scaled by the mean diameter and the velocity has been scaled by a factor of 20^{-1} to fit within the figure. Many times are plotted from $t = 0T$ to $t = 0.8T$ where T is one swimming cycle (1.17 s). For reference, the jellyfish geometry at maximum and minimum diameter are also plotted.

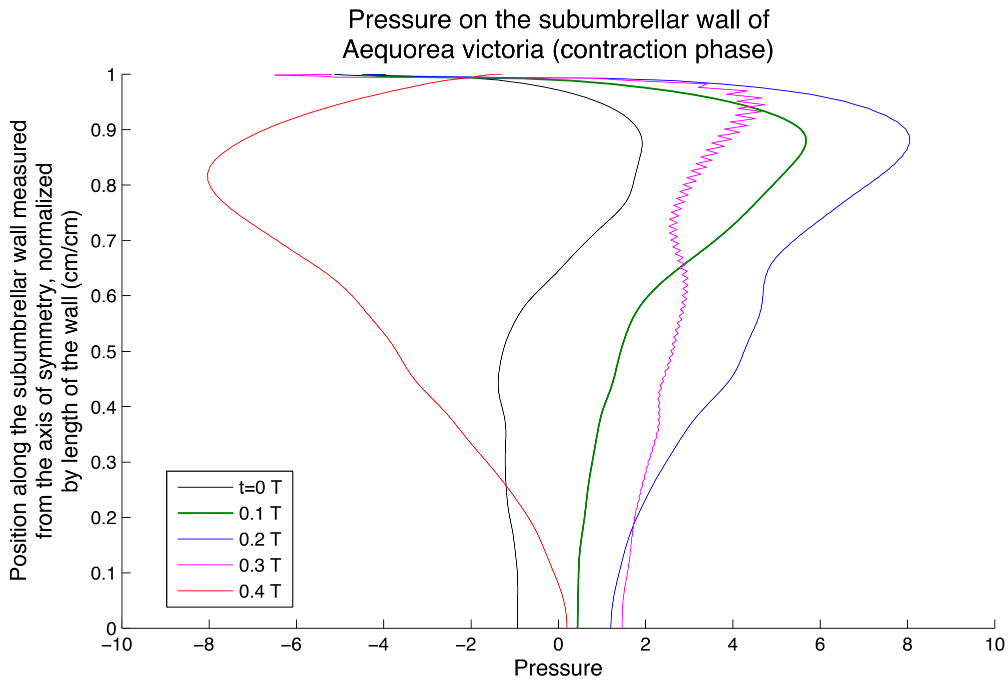


(a) Contraction phase

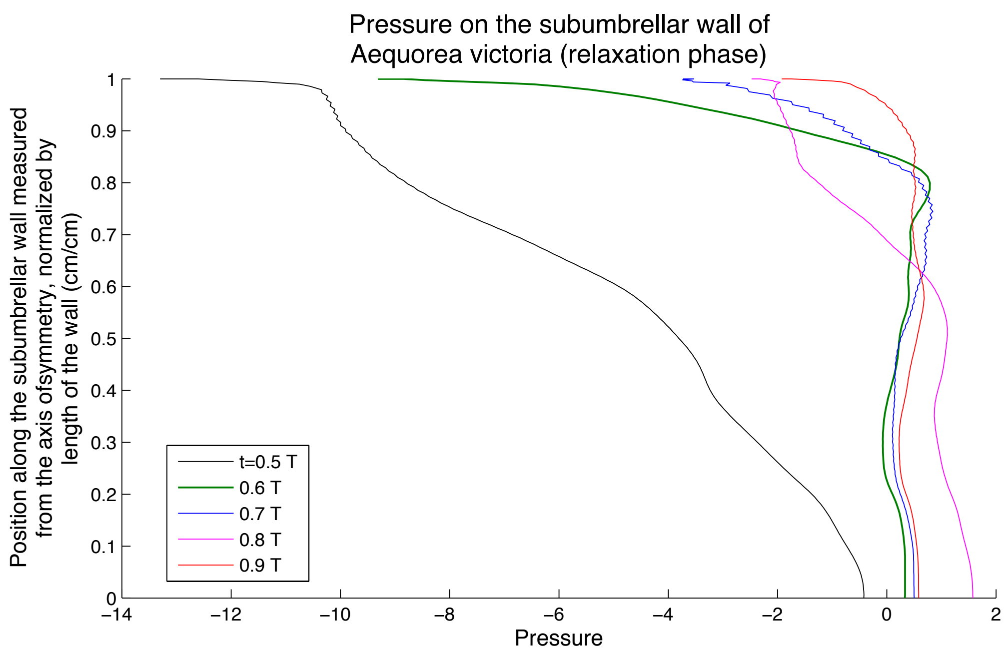


(b) Relaxation phase

Figure 3.13: The axial velocity of the flow relative to the jellyfish across the velar opening of *Aequorea victoria*. Times are presented for one complete paddling cycle.



(a) Contraction phase



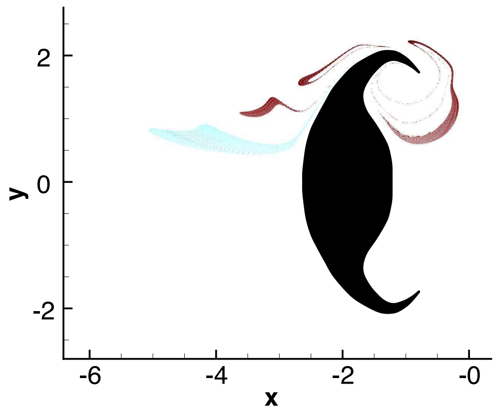
(b) Relaxation phase

Figure 3.14: The pressure on the subumbrellar wall of *Aequorea victoria* for one swimming cycle. The position is normalized by the length of the subumbrellar wall and pressure is given in g cm^{-1} s^{-2}.

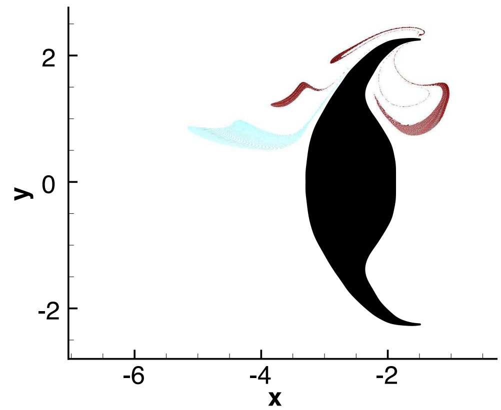
creates two vortices of opposite rotation during each swimming cycle. During the relaxation phase, Figure 3.11(d), a stopping vortex forms in the subumbrellar region with a rotation that draws fluid towards the jellyfish along the axis. Then, during the contraction phase, Figure 3.11(b), a starting vortex of opposite sense is formed very near the first and these two vortices are ejected from the jellyfish together. Once the vortices have been ejected, the weaker stopping vortex acts to cancel out some of the vorticity from the starting vortex. This improves the swimming efficiency of the jellyfish [16]. This swimming motion is repeated periodically, generating a series of vortices and propelling the jellyfish forward. Unlike *S. tubulosa*, where only the starting vortex has been previously noted, both of these vortices have been observed before and are known to play a key role in the swimming of paddling jellyfish such as *A. victoria*.

Figures 3.15 and 3.16 show the motion of passive tracers placed in the flow which end up in the ejected vortices of *A. victoria*. Note that in Figure 3.11(b), during the contraction phase, the forward LCS have formed a loop along the outer surface of the bell which is labeled A in the figure as well as a loop in the subumbrellar region (labeled B). Tracers that end up in lobe A begin upstream of the jellyfish in one coherent group and are swept around the tip of the bell in one cycle (see Figs 3.15(f)-3.16(b)). On the other hand, the tracers in lobe B are more dispersed until they group together in Figure 3.15(e). As a contraction takes place, lobe A is swept into the subumbrellar region, along with the tracers contained therein, and combines with lobe B, merging the two groups of tracers. From here, the tracers are immediately ejected with the next vortex pair.

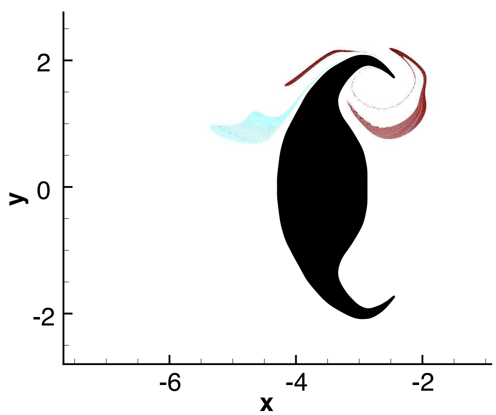
Since particles (such as food) are collected at the core of the vortex, the ejected vortices and the particles contained therein remain at about the same radius as the bell margin and the jellyfish's tentacles as the jellyfish moves upstream. In fact, *Aequorea victoria* swims with $St \approx 1.1$, meaning that the particles in a vortex separate downstream from the jellyfish at a rate of only about 0.9 radii per swimming cycle, in contrast to the 10 radius separation seen for *S. tubulosa*. This provides an excellent opportunity for the jellyfish to feed and offers



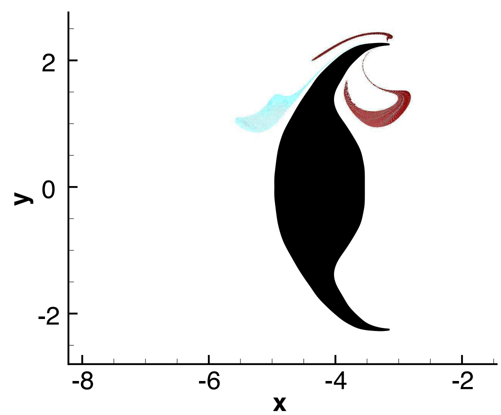
(a) time = 25.142s



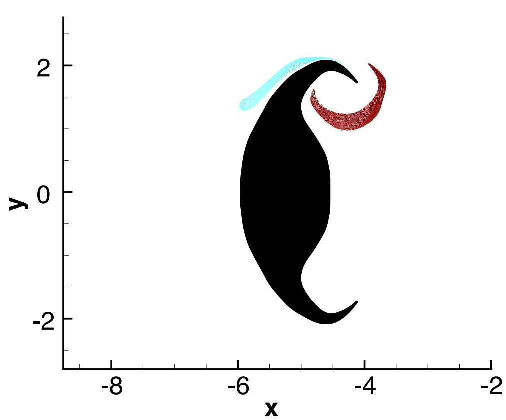
(b) time = 25.725s



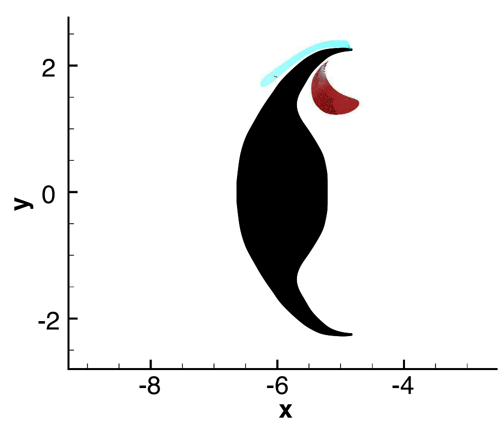
(c) time = 26.308s



(d) time = 26.892s

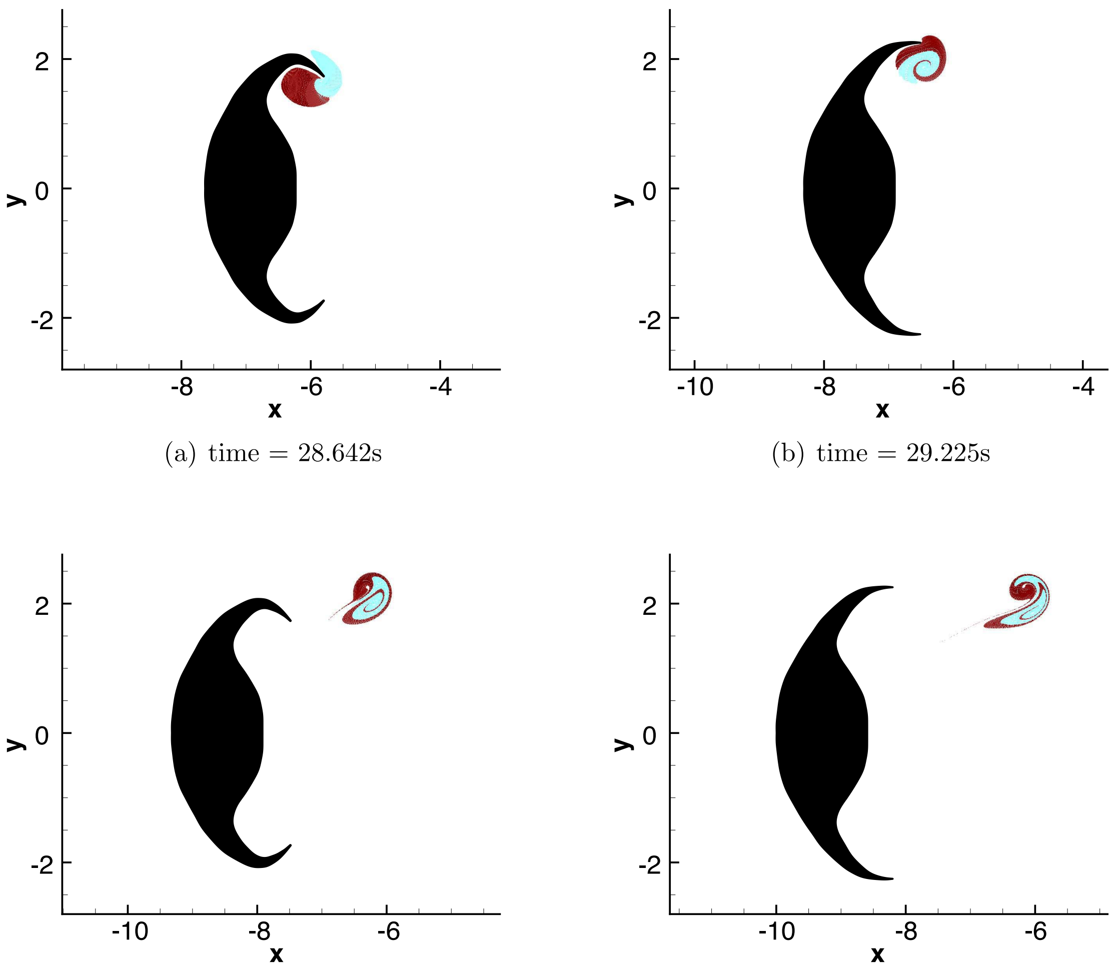


(e) time = 27.475s



(f) time = 28.058s

Figure 3.15: Passive tracer particles for *Aequorea victoria*. There are two groups of tracer particles, colored cyan and dark red, which are ejected as part of the same vortex.



(a) time = 28.642s

(b) time = 29.225s

(c) time = 29.808s

(d) time = 30.392s

Figure 3.16: Passive tracer particles for *Aequorea victoria*. There are two groups of tracer particles, colored cyan and dark red, which are ejected as part of the same vortex.

a plausible explanation for why *A. victoria* swims with its tentacles extended.

3.4 Jellyfish LCS conclusions

Sarsia tubulosa and *Aequorea victoria* fall into two different categories based on their method of swimming. For jellyfish, feeding and swimming are interdependent activities where effective feeding is completely dependent on bringing prey into contact with the tentacles or oral arms where it may be captured. These two groups have addressed this problem in very different ways.

The jetting jellyfish, *Sarsia tubulosa*, retracts its tentacles while swimming and feeds primarily by ambushing its prey. Swimming is used primarily to escape predators or relocate to a new feeding location. To this end, jetting jellyfish have optimized their swimming to move quickly, despite the extra energy costs.

S. tubulosa takes advantage of its jetting motion to produce large accelerations to escape predators and reposition itself for feeding. *Aequorea victoria*, however, experiences much lower accelerations and uses swimming as an extension of its feeding mechanism. In fact, the velocity profile in the wake of *A. victoria* (Figure 3.12) shows significant negative velocities in the wake near the axis of symmetry. This indicates large added mass effects which inhibit acceleration of the jellyfish.

Aequeria victoria feeds by swimming with its tentacles extended. In fact, oblate jellyfish spend almost all their time swimming with tentacles extended [7]. This is effective because each swimming stroke acts to transport fluid that may contain food into the region of the tentacles in a way that enables prey capture (see Figure 3.16(c)).

The relatively high Strouhal number seen for *A. victoria*, $St \approx 1.1$, indicates that the ejected vortices separate at a rate of about 0.9 jellyfish radii per swimming cycle (see Figure 3.11). During this time, the particles entrained in the vortices are transported through the tentacles, presenting an excellent opportunity for prey capture. Furthermore, as the starting and stopping vortices interact, the vortices are stretched (see Figure 3.16(d)) and partially cancel each other due to viscous effects and vorticity diffusion, decreasing the circulation. This creates a relatively slowly rotating and translating vortex, further enhancing the chance for prey capture.

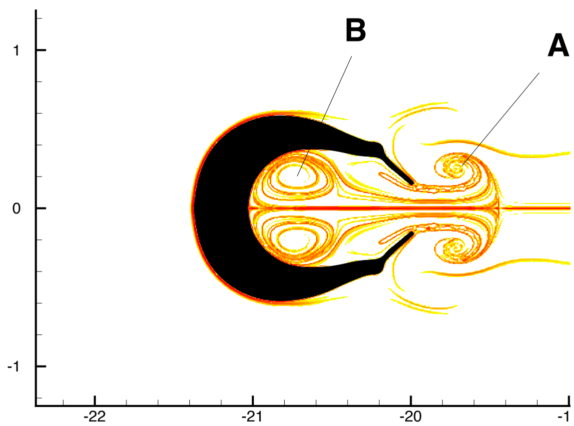
S. tubulosa swims with a much lower Strouhal number, $St \approx 0.10$, than *A. victoria*. This means that the ejected vortices move about 10 radii away from the jetting jellyfish during each swimming cycle. This rapid transport of fluid away from the jellyfish body offers little opportunity for prey capture during swimming since any prey in the surrounding flow is quickly transported out of range of the tentacles.

A. victoria primarily feed on small, soft-bodied zooplankton [14], which, to a good approximation, may be expected to largely drift along with the surrounding flow. Haller and Sapsis [36] have recently examined transport of inertial particles in a general flow and Peng and Dabiri [73] have recently completed a study of the transport of inertial particles in the flow around an oblate jellyfish, *Aurelia aurita*, using particle LCS and they find regions of transport that are very similar to those seen for passive tracers. Furthermore, inertial and finite size effects decrease with prey size. If the fluid inside lobes A and B seen in Figure 3.11(b) contains potential prey, at least a large portion of the prey will be drawn through *A. victoria*'s tentacles during swimming. In particular, lobe A is responsible for most of the transport from the upstream region into the subumbrellar region and past the tentacles. Dabiri and Peng [73] found 64-91% (depending on the parameters used) of the volume of capture regions for passive tracers (analogous to lobe A) was still a capture region for inertial particles. Food particles that impact the outer bell surface of the jellyfish are likely to be entrained in lobe A and potentially captured and eaten.

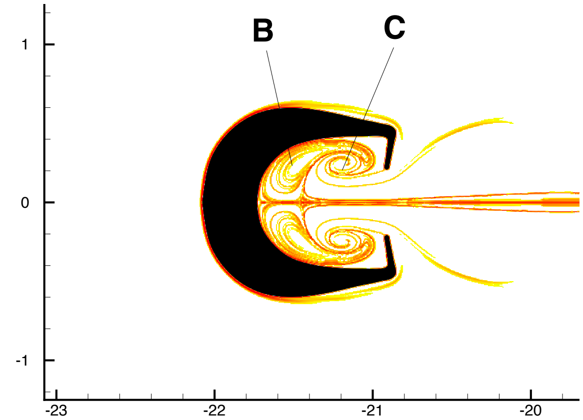
The wake structures produced by these two jellyfish also help to explain some additional features of the anatomy of oblate and prolate jellyfish. Oblate jellyfish commonly have well developed and prominent oral arms extending from the center of the bell while these structures are absent in prolate species [11]. In fact, the presence of well developed oral arms or tentacles extended in the flow in a jetting species could add drag and decrease swimming performance. On the other hand, paddling jellyfish take advantage of feeding structures, such as tentacles and oral arms, by feeding while swimming.

The LCS seen in this chapter reveal complex fluid structures in the subumbrellar region of *S. tubulosa* (see Figure 3.2). The subumbrellar region even contains a stopping vortex that forms during the relaxation phase of swimming. However, unlike in paddling jellyfish, this stopping vortex remains inside the subumbrellar cavity and therefore cannot influence the development of the ejected vortex. Figure 3.17 shows the development and dissipation of the stopping vortex for *Sarsia tubulosa*. This figure displays only the backward LCS so that activity in the subumbrellar cavity is clear. As starting vortex A is ejected during contraction, the previous stopping vortex, B, sits deep within the subumbrellar cavity (Figure 3.17(a)). During relaxation a new stopping vortex, C is formed and begins to push B along the walls of the cavity (Figure 3.17(b)) until C resides deep within the cavity and B has been pushed near the velar opening (Figure 3.17(c)). At this point, B is no longer recognizable as a well defined vortex. In fact, as B is pushed along the wall, C interacts with the wall to create a secondary vorticity of an opposite sense to vortex C. This secondary vorticity overwhelms vortex B so that region D in Figure 3.17(d) actually has vorticity of opposite sense to the stopping vortex. Finally, a new contraction begins and the fluid in region D (Figure 3.17(d)) is ejected as part of a new starting vortex.

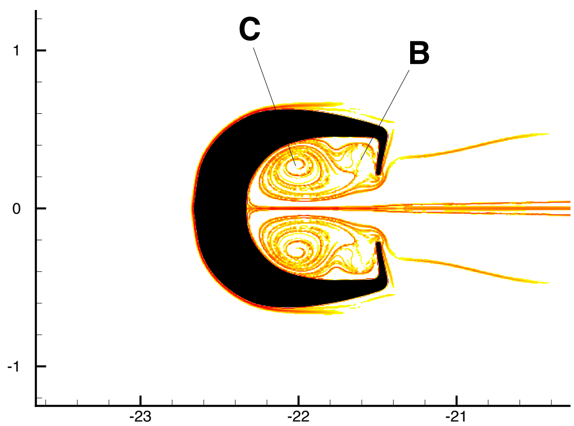
A. victoria ejects both vortices together during the contraction phase while *S. tubulosa* ejects only the starting vortex while the stopping vortex remains within the bell. The different morphologies of the two jellyfish produce this distinction. In *A. victoria*, each vortex is formed by the shear layer being shed off the tip of the bell. However, the velum of *S. tubulosa* alters the way the vortex formation so that the shear layer formed during relaxation is shed into the bell and the stopping vortex moves deep within the velar cavity, instead of remaining near the velar opening, so it is not ejected during contraction. This stopping vortex in *S. tubulosa* has not been observed in DPIV or dye visualization experiments, most likely due to its confinement within the bell and the difficulty of imaging this area in experiments.



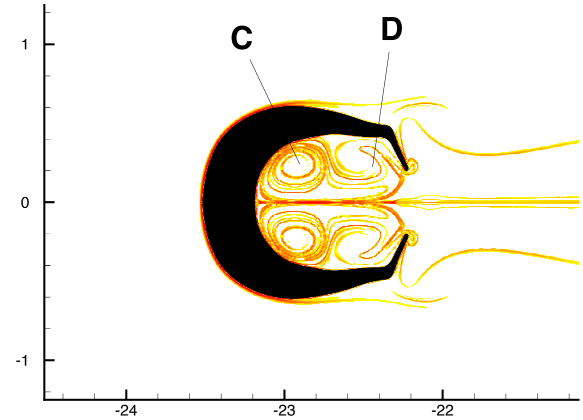
(a) Starting vortex A and stopping vortex B



(b) Old stopping vortex B and new stopping vortex C



(c) Old stopping vortex B and new stopping vortex C



(d) Stopping vortex C and region D which will be ejected

Figure 3.17: Backward LCS for *Sarsia tubulosa* showing the development of the stopping vortex. During contraction, (a), starting vortex A is ejected while stopping vortex B from the previous relaxation sits deep in the subumbrellar cavity. During relaxation, (b), a new stopping vortex, C, is formed as fluid is drawn into the subumbrellar region and stopping vortex C pushes the weaker stopping vortex B out of the way, along the walls of the cavity. In (c), stopping vortex C has settled deep within the subumbrellar cavity and B has been pushed to the area near the velar opening. B is no longer recognizable as a well defined vortex. Finally, in (d), a new contraction begins. C is deep within the subumbrellar cavity while fluid in the area of D, including the remnants of B, will be ejected with the next starting vortex.

Its effect on energy requirements and swimming efficiency remains to be seen.

The LCS analysis of this chapter, coupled with studies of drifter motion and pressure/velocity profiles gives new insight into the fluid structures and swimming behavior of these jellyfish. However, the computations required for this analysis required many days to run, even for these two-dimensional problems. Most three-dimensional problems are beyond the reach of these techniques without investing in significant computing resources. Developing efficient algorithms to overcome this limitation and enable three-dimensional LCS computations on standard desktop computers is the focus of the following chapters.

Chapter 4

Gridless Ridge Tracking Algorithms

4.1 Introduction

The nature of LCS, codimension-one ridges in the FTLE field, makes their computation a natural candidate for a more efficient algorithm. Since the ridges are the only part of the FTLE field that are needed for LCS computations, any FTLE values that are computed away from the ridges are essentially wasted computational time. There are two broad classes of algorithms that are well suited for this situation. The first is adaptive mesh refinement (AMR). By starting with a coarse grid of points on which the FTLE field is calculated and refining the grid only in areas where ridges are detected, it is possible to achieve very high resolution near the ridges with large computational savings when compared to a uniform mesh at the same resolution. There are many possible variations to this type of algorithm, but at its heart, it only requires a criterion to determine where the mesh should be refined. Two recent papers provide examples of such criteria [31, 79].

A second type of algorithm involves detecting and tracking a ridge in the FTLE field. In this type of algorithm, there must be some initial way to detect a point on the ridge, after which the ridge may be "grown" in space. This chapter focuses on gridless ridge tracking algorithms. These algorithms do not use a pre-determined grid. Rather, they dynamically generate a mesh to approximate the location and orientation of the LCS ridges in space.

I first present a two-dimensional ridge tracking algorithm for LCS computations. This algorithm uses the fact that LCS are coherent in time as well as space to speed computations.

In the case of a time dependent flow field, the LCS at one time step may be used to provide an estimate of the location of the LCS at the next time step. LCS are known to often be nearly invariant manifolds with the fluid flux through LCS on the order of numerical error [85]. This means that, to a good approximation, LCS are simply advected with the flow. In the algorithm presented here, this property is leveraged by using the LCS at time t to predict the location of LCS at time $t + \delta t$. A small correction is then made to ensure that the new points at time $t + \delta t$ are actually on the LCS and then the rest of the LCS are extracted via the ridge tracking algorithm.

This two-dimensional ridge tracking algorithm is applied to two example problems: the time dependent double gyre flow and the jetting jellyfish *Sarsia tubulosa* of Chapter 3. The algorithm provides a speed up of up to 81.6 times in the double gyre example, almost two orders of magnitude.

Next, the two-dimensional ridge tracking algorithm is extended to three dimensions. This involves significant complications because a surface mesh must be generated to represent the LCS surfaces in three-dimensional problems. Although the resulting algorithm accurately computes and tracks surfaces, generating the surface mesh while ensuring mesh consistency (e.g. no self intersections) quickly becomes at least as expensive as the standard LCS algorithm.

4.2 A two-dimensional gridless ridge tracking algorithm

This section focuses on the extraction of LCS in two-dimensions via a ridge tracking algorithm. This algorithm has the advantage of computing the FTLE field at a greatly reduced number of points so it provides a very large speedup over methods which compute the FTLE field over the entire domain. Additionally, since fewer points are used, less memory is required for the computations and since the output consists only of the one-dimensional LCS curves rather than the entire two-dimensional FTLE field, the output files are also much smaller.

The basic gridless ridge tracking algorithm is outlined in Algorithm 2 and is presented in detail in the following sections. The main difference between the two- and three-dimensional algorithms is in the implementation of the mesh generation step of Algorithm 2.

Algorithm 2 Gridless ridge tracking algorithm

```
Set integration time, T, and spatial step, δx
for t = t_0 to t_f do
  Every N^th step, perform initial ridge and orientation detection
  while new ridge points are found do
    Discard FTLE values below threshold
    Generate mesh connecting new points to the ridge
    Step forward tangent to the ridge by δx and check for new ridge points
  end while
  Advect some ridge points forward to next time
  Readjust positions to accurately locate new initial points
end for
Save or plot the parameterized LCS ridges
```

4.2.1 The first time step

The first time step requires locating at least one point on the LCS to be extracted, as well as the orientation of the corresponding ridge in the FTLE field. This is accomplished by computing the FTLE value along lines that crisscross the domain. These lines of FTLE values then have local maxima where they cross a ridge in the FTLE field so ridges are initialized at these points. To obtain a more accurate estimate of the ridge location, this algorithm uses the locally maximum value, plus the value on either side to approximate the FTLE values around the ridge with a parabola and locate the maximum of this parabola (see Figure 4.1). A threshold may also be set to ensure that these points have at least some minimum FTLE value (experience shows that an appropriate threshold is usually in the

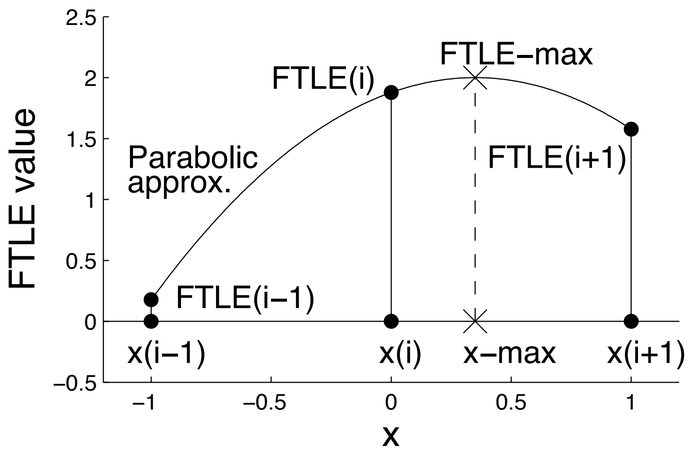


Figure 4.1: The actual local maximum of the FTLE is estimated with a parabolic approximation of the ridge.

range of 50-80% of the maximum detected FTLE value) since only the "strongest" LCS are desired.

The next step is to find the orientation of the ridges to begin stepping along the ridge to extract the LCS. For each initial point, this is accomplished by computing the FTLE value of the eight points surrounding the point at a distance of δx. The point with the maximum FTLE value is selected and its neighbors are used to form a parabolic approximation of the FTLE values and provide a better estimate of the location of the highest surrounding point. Finally, the orientation of the ridge is determined by the vector from the initial point to this highest surrounding point. The algorithm then begins stepping along the ridge in both directions from this initial point.

4.2.2 Mesh generation

In two-dimensional problems, the LCS are one-dimensional ridges and mesh generation is nearly trivial. One simply connects new ridge points to the previous point with a line segment and ensures that the line segment does not intersect any existing line segments and the new point is not within some tolerance (say δx) of any other ridge points. If two points are closer than the tolerance, the the corresponding ridges are merged at those points. This process will be much more complicated in higher dimensions.

4.2.3 Tracking the ridge

Given a point on the ridge and the orientation of the ridge, the algorithm tracks the ridge by taking a step in the direction of the given orientation, computing the FTLE value at three points on a line normal to the step direction, approximating the FTLE value by a parabola and estimating the location and value of the true local max along this normal line. Figure 4.2 contains a graphical representation of this process which may be more clear. To prevent the trajectory from jumping from one side of the true ridge to the other, after the second step the trajectory updates are based on the average of the last two steps. Experience also dictates that the spacing of the points on the normal line (used to compute the parabolic approximation) should be about 1/2 the step size along the ridge or they may be adjusted adaptively based on the curvature of the ridge as computed from the previous approximation. Finally, when making the parabolic approximation, one must ensure that the parabola has the appropriate concavity (concave down) and that the maximum occurs between the points on the normal line. If either of these tests fails, it is best to step to the point on the normal line with the maximum FTLE value. If these tests fail because the ridge has ended or the algorithm has lost the ridge, the stopping criteria (presented below) will prevent the ridge from growing any further.

4.2.4 Stopping criteria

It is important to know when to stop tracking a ridge because either the ridge has ended or the algorithm has lost the ridge. Three criteria cause the algorithm to stop. The criteria and their results are as follows:

(1) The end of the ridge leaves the domain of the computations: stop tracking the ridge

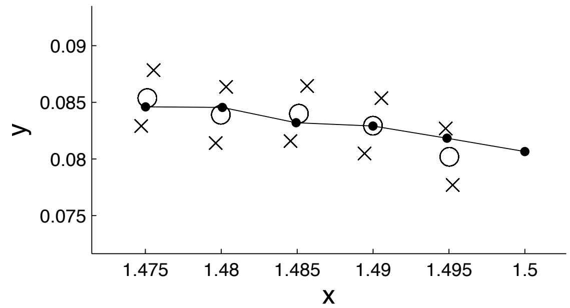


Figure 4.2: The first several steps along a ridge, beginning at the right end. The estimated next point, based on the ridge trajectory is shown as a circle, the other points on the normal line are shown as $\times$'s and the actual trajectory is drawn as a line.

(2) The ridge hits the start or end of another ridge: join the two ridges into one

(3) The FTLE value on the ridge falls below the threshold value (e.g. $0.8\max(\sigma)$, where $\max(\sigma)$ is the largest FTLE value that has been found): either the ridge is ending or the algorithm has lost the ridge so stop tracking the ridge

4.2.5 Later time steps

Once all the ridges have ended by meeting one of the stopping criteria, the algorithm advances to compute the LCS at the next time step. To avoid the cost of initially detecting points on the ridges one may advect some points on the ridge forward to the next time. For efficiency, roughly every 15^{th} point along the ridge is used, but this is a tune-able parameter that may be used to help optimize the algorithm. Additionally, neighboring pairs of points are used so that the orientation of the new ridge may be easily determined and some error may be eliminated by taking their average position as the estimated new ridge position. Two additional points on the line normal to the new ridge are used with the parabolic approximation to more precisely locate the new ridge. Once these new initial points and orientations have been computed, any points below the threshold value are discarded and the algorithm proceeds exactly as for the first time step.

Even if a few of the advected points miss the new LCS ridges the other points on the ridge will grow the ridge to fill in the gap. In practice, this rarely happens and all the advected points lie very near the the new ridge. Also, it is possible (and likely) that entirely new LCS will be created elsewhere in the flow domain. This algorithm will not detect these new ridges so it is necessary to occasionally repeat the initial detection part of the algorithm as performed in the first step. The frequency of repeating this step is entirely dependent on whether or not it is deemed acceptable to miss a newly created LCS for a few time steps. Also, if a new ridge appears during this re-initialization and it is critical that all ridges are detected, the previous time steps may be re-computed by the same method (only advecting the points backwards in time) until the ridge is no longer present.

4.2.6 Results

To demonstrate the capabilities of this algorithm it is applied to two examples and compared to the FTLE field over the entire domain. The first is an example that has become a standard test case for computing LCS, the time dependent double gyre. Secondly, the algorithm is applied to a swimming jellyfish. In both examples, the ridge tracking algorithm provides a significant increase in performance.

4.2.6.1 The time dependent double gyre

The time dependent double gyre consists of two counter-rotating gyres with a periodic perturbation. This flow has become a commonly used test case for LCS papers since its use in 2005 by Shadden et al. [85]. The velocity field for this system is given by the stream function

$$\psi = A\sin(\pi f(x,t))\sin(\pi y) \tag{4.1}$$

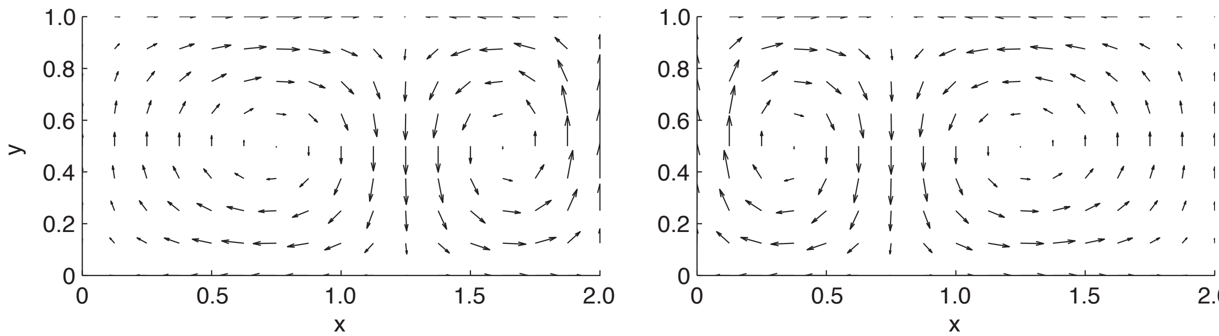


Figure 4.3: The double gyre velocity field at maximum deflection to the right and left. The time dependent perturbation causes the gyres to oscillate in the horizontal direction.

where

$$f(x,t) = a(t)x^2 + b(t)x, \tag{4.2}$$

$$a(t) = \epsilon \sin(\omega t), \tag{4.3}$$

$$b(t) = 1 - 2\epsilon \sin(\omega t). \tag{4.4}$$

$$\tag{4.5}$$

The velocity is then given by

$$u = -\frac{\partial \psi}{\partial y}, \quad v = \frac{\partial \psi}{\partial x}. \tag{4.6}$$

This results in a left hand gyre rotating clockwise and a right hand gyre rotating counter-clockwise in the closed domain $[0, 2] \times [0, 1]$ as shown in Figure 4.3. This velocity field will be used as a test case in several examples in this dissertation. This section uses the parameters $A = 0.1$, $\epsilon = 0.1$, $\omega = 2\pi/10$, and integration time $T = \pm 15$ which gives a system with an oscillation period of 10.

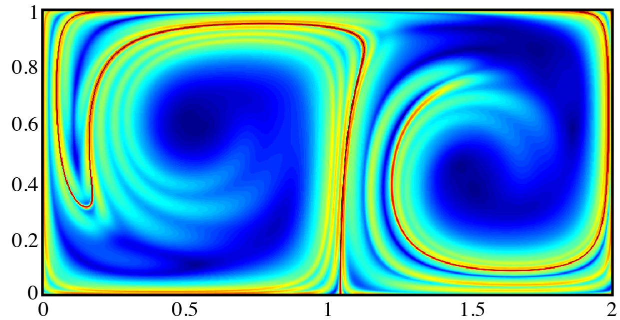
For reference, the forward and backward FTLE field are shown in Figures 4.4(a) and 4.4(b) respectively. This field was computed with grid spacing of 0.0025, resulting in an 801×401 grid and the computation took 20.4 seconds per time step to run. FTLE values were computed using central differencing on this grid resulting in a finite difference spacing of $\Delta x = 0.005$. This grid spacing was chosen to be 1/2 the step size used in the ridge tracking algorithm. Since the ridge tracking algorithm can correct the position of steps along the ridge by up to 1/2 step size, this grid spacing provides a fair comparison between the two methods.

The LCS extracted from this same system by the ridge tracking algorithm are shown in Figure 4.5. The results are shown only for time $t = 0$, but are typical of all times and are indistinguishable from the ridges seen in Figure 4.4 when plotted on top of the FTLE field. For this example, a step size along the ridges of 0.005 is used, a spacing of points used to compute parabolic approximations of 0.0025, and FTLE values were computed with central differencing on a four point stencil with spacing $\Delta x = 0.005$ (the same as the spacing used above to compute the full FTLE field). Finally, a threshold value of 80% of the maximum computed FTLE value was used.

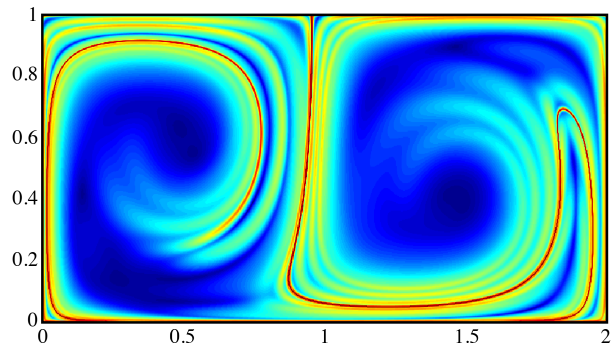
As mentioned previously, it is necessary to re-initialize the ridge tracking algorithm occasionally to ensure that any newly formed LCS are captured. For this system, the LCS were computed at time steps of $\Delta t = 0.02$ and re-initialized 10 times per period (at $t = 0, 1, 2, 3, ...$). Since the length of the LCS ridges varies with time, so does the computation time. To account for this, the average computational time is used for performance comparisons. The overall average time step for this run takes just 0.25 s and represents an 81.6 times speedup over the full FTLE calculation. This speedup is a combination of the time saved during the computations and also much less disk I/O due to the much smaller output files.

4.2.6.2 Jellyfish swimming

In the second example, the ridge tracking algorithm is used to examine the flow created by a swimming jellyfish. Th *Sarsia tubulosa* species discussed in detail in Chapter 3. The main differences between this flow and the double gyre flow of the previous section are the complexity of the LCS and the fact that the flow around the swimming jellyfish is stored in data files output from a CFD code, rather than analytically defined.



(a)



(b)

Figure 4.4: The forward (a) and backward (b) FTLE fields for the time dependent double gyre system at time $t = 0$ with $A = 0.1$, $\epsilon = 0.1$, $\omega = 2\pi/10$, and $T = -15$.

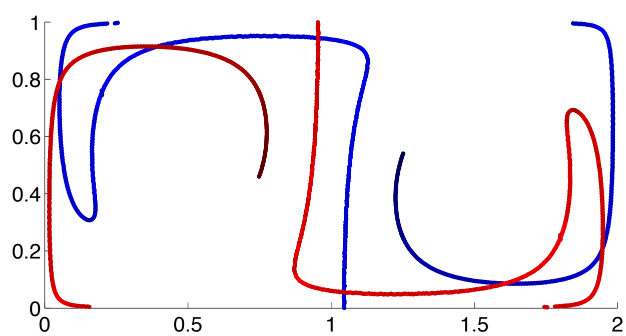


Figure 4.5: The extracted LCS with the ridge tracking algorithm for the time dependent double gyre system at time $t = 0$ with $A = 0.1$, $\epsilon = 0.1$, $\omega = 2\pi/10$, and $T = \pm 15$. Forward LCS are shown in blue and backward LCS are shown in red. Also, the relative height of the ridge is indicated by the shade of the color. As the ridge height decreases to the threshold value, the color fades to black.

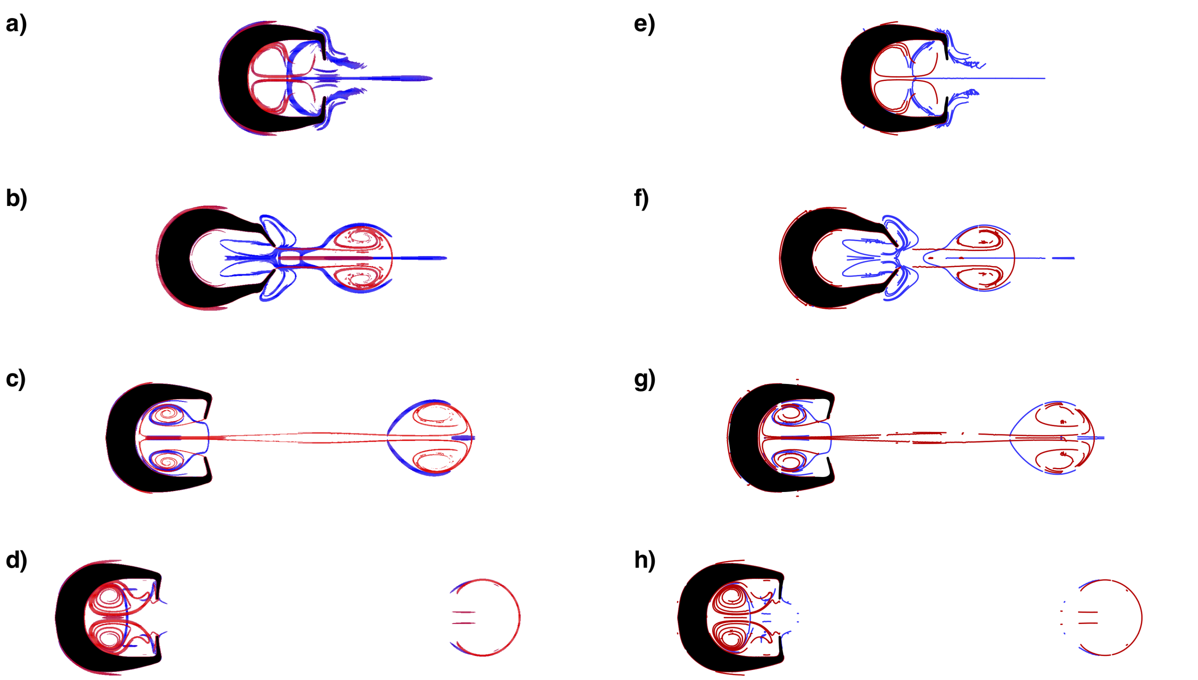


Figure 4.6: The LCS of *S. tubulosa* for several time steps as computed with both the standard algorithm as well as the two-dimensional gridless ridge tracking algorithm. Forward LCS are shown in blue and backward are red. The results of the standard algorithm are shown in a-d and the ridge tracking algorithm are show in e-h. The four different time steps are evenly spaced at 0.25 second intervals and make up one complete swimming cycle (1 second).

In cases where the velocity field is stored in data files instead of analytically defined, several additional considerations must be made to implement this ridge tracking algorithm. First and foremost, depending on the size and format of the velocity files, velocity read ins may actually be a limiting factor for algorithm performance. It is very important to choose an efficient data format to minimize velocity read in time. In this case, the netCDF data format is used [91]. This format is commonly used in several areas of scientific research, particularly the geophysical sciences, and has proven to be efficient and flexible.

Secondly, a ridge tracking algorithm requires computing FTLE values for each step along a ridge. If this is done by reading in the velocity for each time step as it is needed during the advection process, then the velocity files may each need to be read in hundreds of times as the algorithm iteratively steps along the ridges. To avoid this, the ridge tracking algorithm should only be applied to situations where the velocity field for one complete period of integration can fit in memory.

The standard algorithm was used to compute the full FTLE field for comparison. The thresholded contour plots of the forward and backward FTLE fields for this jellyfish are presented in Figure 4.6a-d. This computation was performed at mesh spacing of 0.0025 with an integration time of 0.40 seconds (40 timesteps). On average, computing and writing the FTLE field for each timestep took 874.6 seconds or about 14.6 minutes.

The resulting LCS from the ridge tracking algorithm are also shown in Figure 4.6e-h. The ridge tracking algorithm took an average of 60.8 seconds per time step to run and captures all the major features of the LCS seen in the full FTLE field. This represents a speed up of 14.4 times. as the time to compute each time step was cut from 14.6 minutes to just over 1 minute. While this is not as drastic a speedup as seen for the double gyre example, it is still a substantial savings and a speed up of more than an order of magnitude.

In both sets of figures, all the major LCS present at this time are clearly visible. In fact, because the results of the standard LCS computation are often visualized with a thresholded contour plot, they often do not appear as clear lines where the ridges are present, but rather as narrow regions of high FTLE value. The ridge tracking algorithm suffers from no such drawback. Although there is a slight loss of detail in the vortex core seen in figures 4.6f and g, a single, a one dimensional line along each ridge is extracted. This is well demonstrated in Figure 4.6. For example, the results of the ridge tracking algorithm display the structures in the bell in Figure 4.6d,h more cleanly than the standard algorithm.

This algorithm is very well suited for picking out the major LCS. As the integration time is increased, the LCS that are revealed become increasingly complex and close together. The total length of LCS present in the domain increases as well. In many applications such as flow control and for identifying major flow structures only the major LCS need to be computed. For these purposes, this algorithm provides a large time savings over the standard algorithm.

4.3 A three-dimensional gridless ridge tracking algorithm

The first half of this chapter discussed a gridless ridge tracking algorithm for computing LCS in two-dimensional flows. The results were very good and the the algorithm proved to be accurate and provide significant speed ups. However, it is much more expensive to compute the LCS in three-dimensional flows. Unfortunately, the above algorithm is non-trivial to extend to three dimensions. The primary difficulties lie in advancing along the ridges (which become surfaces in three dimensions) and generating a surface mesh to represent the results. This section uses a surface tracking method based on an advancing front meshing algorithm to create meshes of triangles on an arbitrary surface. The surface tracking algorithm is able to mesh a surface of arbitrary topology given only an initial point and normal vector on the surface. However, the computational cost of generating the surface mesh and ensuring mesh consistency quickly outweighs the savings of avoiding unnecessary FTLE computations.

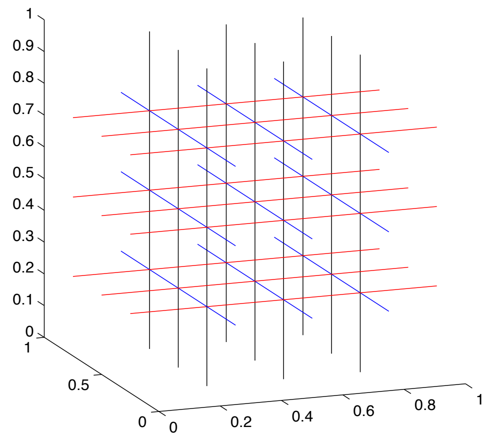


Figure 4.7: Points on the ridge surfaces are initially detected by computing the FTLE values on a grid of lines through the domain. FTLE values are computed at a spacing of δx along these lines.

4.3.1 The first time step

The first time step is nearly identical to the two-dimensional algorithm presented above. Initial points on the ridge surfaces are located by computing the FTLE values on a series of lines that cress cross the domain. In three dimensions the lines look like those shown in Figure 4.7. Local maxima along the lines are assumed to correspond to FTLE ridge crossings and the same parabolic approximation is used to precisely locate the initial ridge points (see Figure 4.1). Again, only the strongest LCS are kept by setting a threshold of 50-80% of the maximum detected FTLE value.

The next step is to find the orientation of the surface in order to begin growing the mesh to extract the LCS. For each initial point, this may be accomplished by approximating the Hessian of the FTLE field at that point. The eigenvector of the Hessian matrix corresponding to the most negative eigenvalue is normal to the LCS surface.

4.3.2 The advancing front method

Given a point on the surface and the normal to the surface a set of six equilateral triangles is initialized in the tangent plane at this point (see Figure 4.8). The side length

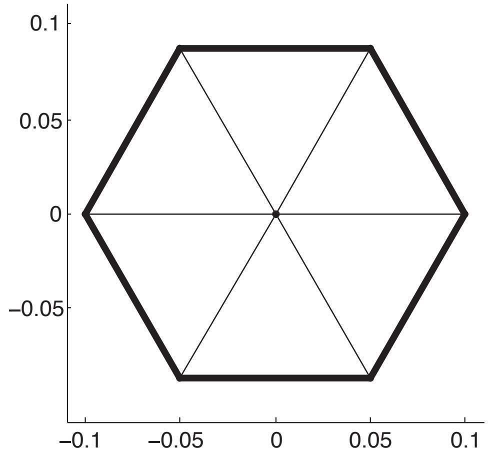


Figure 4.8: Initialization of the surface based on a tangent plane approximation. The mesh front is shown in bold.

of the triangles, δx, is a user defined parameter and depends on the expected maximum curvature of the surface. Each of the new points is then adjusted in the normal direction to lie on the true LCS surface. Given point $\mathbf{x}_i$ and normal vector $\hat{\mathbf{n}}_i$, this is achieved by checking the FTLE values at $\mathbf{x}_i \pm \frac{\delta x}{2}\hat{\mathbf{n}}_i$. The location of the maximum FTLE value in the normal direction is determined by the parabolic approximation through these three points. This is analogous to the adjustment steps made above in the two-dimensional algorithm.

Each edge on the mesh front $(\mathbf{x}_i)$ has an associated normal and tangent vector $(\hat{\mathbf{n}}_i\ \hat{\mathbf{t}}_i)$ determined by the triangle containing the edge. The normal vector is normal to the triangle and the tangent vector lies in the plane of the triangle, is normal to the mesh front, and points in the direction of front growth.

To advance the mesh front, triangles are progressively added to the edges of the front. If an acute angle is detected on the front, a new edge is added across the angle. If this new edge is longer than a user specified tolerance (say $1.2\delta x$), two edges are added instead. Otherwise, each edge is stepped forward by adding an isosceles triangle with two sides of length δx to the existing edge. It is necessary to check if the new node is within some tolerance of an existing node, in which case the two nodes are merged. As the mesh front advances, the number of edges may grow or decrease accordingly (see Fig 4.9).

It is also necessary to ensure that new nodes do not result in self-intersections of the mesh. Different parts of the mesh front may merge and the front may split into two different fronts. For example, if the surface is a cylinder the tracking algorithm will result in a mesh front that intersects itself opposite the point of initialization as shown in Figure 4.10. In this case, points in close proximity are combined and front edge neighbors are redefined to prevent self-intersection. In this way, the initial single front may split into two or more separate fronts (see Figure 4.10).

Finally, if the FTLE values fall below the chosen threshold at some point on the front, that point is considered an edge point and stopped. The rest of the front continues advancing. In this way, the mesh front may split upon encountering a hole in the surface and rejoin on the other side of the hole as shown in Figure 4.10.

4.3.3 Subsequent time steps

Once all the surfaces have been computed, the algorithm moves on to compute the LCS at the next time step. To avoid the cost of initially detecting points on the surfaces some points on the surface are advected forward to the next time. This is done exactly analogously to the two-dimensional case discussed above. A subset of the ridge points are selected. Triads of points are used so that the ridge orientation can be determined and the new ridge points are adjusted using the parabola approximation.

4.3.4 Results

Testing with two analytical velocity fields was performed to validate the three dimensional gridless ridge tracking algorithm. The two cases presented here are the double gyre flow and ABC flow. Each case has been previously examined in detail and the results reported here match well with previously reported LCS for these velocity fields.

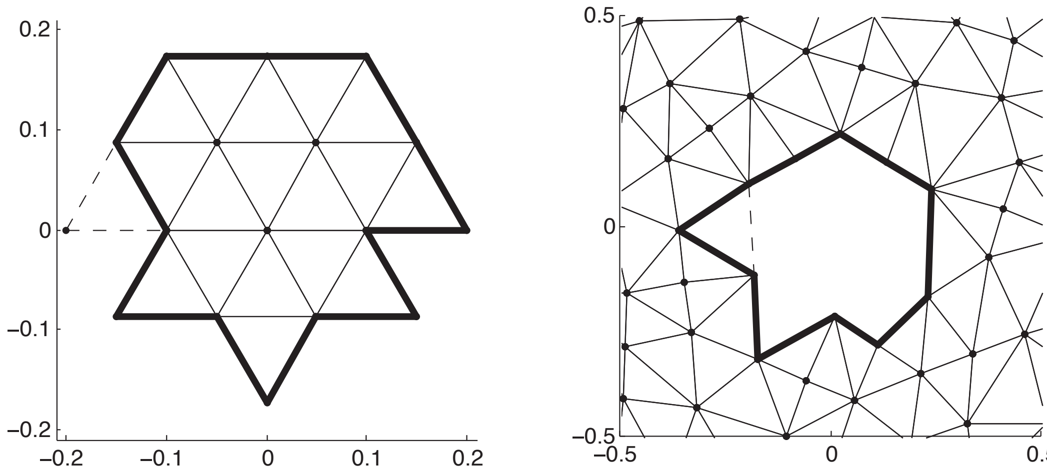


Figure 4.9: As the mesh front lengthens or shortens the number of edges may grow or decrease. The left image shows an expanding mesh front with an increasing number of edges while the right shows a contracting mesh front with a decreasing number of edges. The next edges that will be added are drawn as dashed lines.

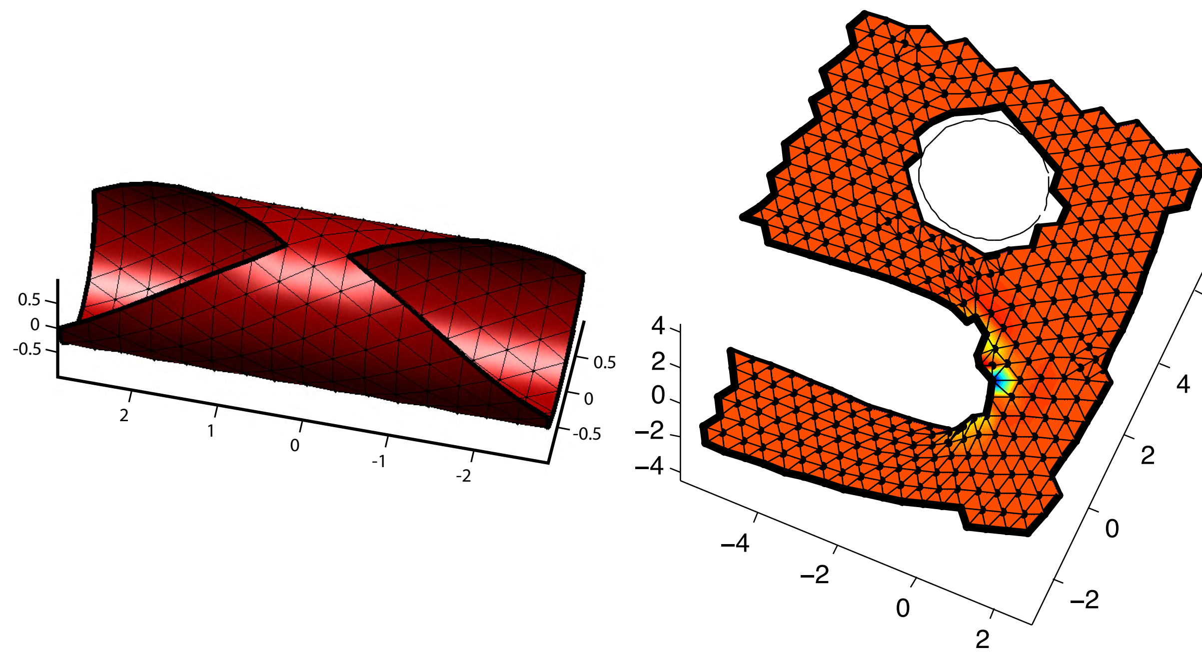


Figure 4.10: Two parts of the mesh front may intersect, resulting in two disjoint fronts. The advancing front meshing algorithm is also able to mesh around holes and discontinuities in the surface.

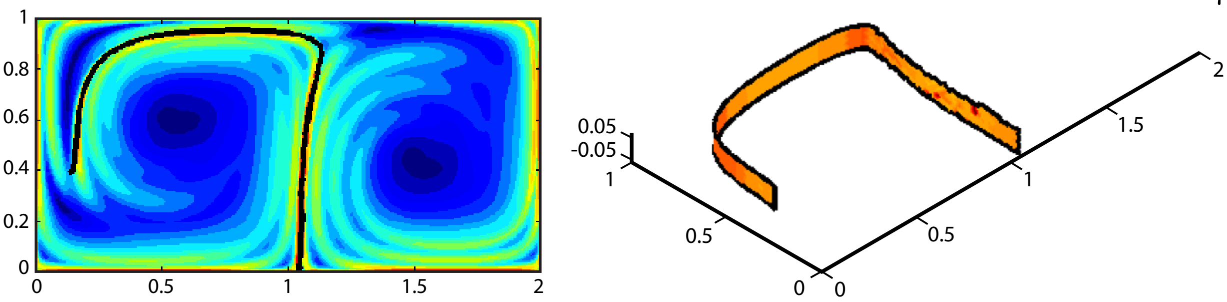


Figure 4.11: The full FTLE field (left) and the resulting LCS surface (right). The surface is drawn in black over the FTLE field at left and aligns with the most prominent LCS.

4.3.4.1 Double Gyre flow

The same double gyre flow discussed above in Section 4.2.6.1 is used here. The flow is extended to three-dimensions by simply setting the z-velocity to

$$w = 0,$$

resulting in LCS that are independent of z. The parameters $A = 0.1$, $\epsilon = 0.1$, and $\omega = 2\pi/10$ are used. An integration time of $T = 10$ is used since this algorithm struggles to track the more complex LCS resulting from longer integration times.

The resulting surface is shown in Figure 4.11 and aligns with the expected LCS ridge in the two-dimensional FTLE field. The resulting surface has 825 nodes and 1403 triangles and required 19,248 particle advections to generate the mesh. An equivalent fully three dimensional computation would require 200,000 particle advections to achieve equivalent resolution. However, very little performance improvement is seen due to the large amount of time spent generating and checking the surface mesh.

4.3.4.2 ABC flow

Arnold-Beltrami-Childress (ABC) flow is an analytically defined velocity field that is known to exhibit chaotic trajectories [22]. This flow has also been used as a test case in several other investigations of coherent structures in three-dimensions (e.g. by Haller [34]).

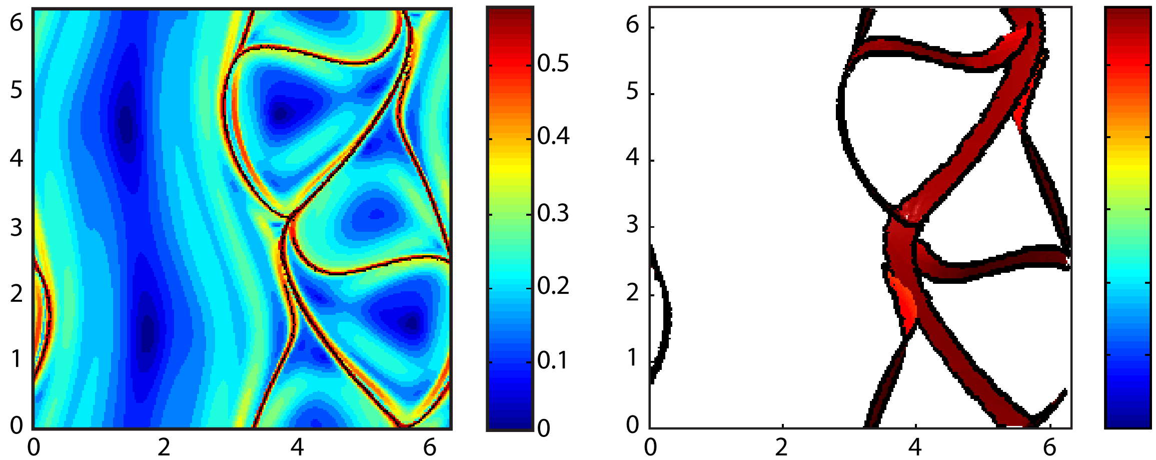


Figure 4.12: LEFT: The FTLE field for ABC flow at $z = 0$ with $T = -10$. The black lines represent the intersection of the computed surfaces with the $z = 0$ plane. RIGHT: The LCS surfaces for ABC flow for $-0.2 \le z \le 0.2$ with $T = -10$. There is excellent agreement between the computed surfaces and the FTLE field.

The general ABC flow is given by

$$
\begin{aligned}
u &= A\sin(z) + C\cos(y) \\
v &= B\sin(x) + A\cos(z) \\
w &= C\sin(y) + B\cos(x)
\end{aligned} \tag{4.7}
$$

and typically $A = 1$, $B = \sqrt{2/3}$, and $C = \sqrt{1/3}$. An integration time of $T = -10$ (backward LCS) is used for this example. The velocity field is periodic in all directions and so are the corresponding LCS. A two-dimensional slice of the FTLE field at $z = 0$ is shown in Figure 4.12. The surface tracking algorithm is then used to extract the LCS surfaces near this plane.

Figure 4.12 also shows the LCS surfaces for the ABC flow. The surfaces have been restricted to $-0.2 \le z \le 0.2$ for ease of visualization and there is excellent agreement between the LCS surfaces and the FTLE field. The LCS surfaces are comprised of 9985 triangles and 5627 nodes resulting from about 130,000 particle advections. The full three-dimensional LCS computations of equivalent resolution would require over 1 million particle advections.

Again, no significant speed up is seen due to the mesh generation costs.

4.4 Gridless surface tracking conclusions

Finding Lagrangian coherent structures is a computationally intensive process and as the problems to which LCS are applied grow in size, the computational cost becomes prohibitive. The gridless ridge tracking algorithms presented in this chapter are intended to address this problem. For two-dimensional flows, the gridless algorithm achieves a speed of more than 80 times for a simple analytical velocity field and almost 15 times for the swimming jellyfish. Additionally, all the major LCS were successfully captured in each example. The speed up is achieved without a loss of detail or accuracy in finding the most significant LCS in a system. This ridge tracking algorithm also provides additional savings in terms of memory usage and size of the output files.

Three-dimensional flows require vastly more computational time than their two-dimensional counterparts and the three-dimensional ridge tracking algorithm attempts to extend the two-dimensional results the these flows. Although good accuracy is achieved in the test cases shown above, the complexity of generating and checking a surface mesh eliminate any time savings in the computations. Perhaps more sophisticated meshing algorithms that make use of adaptive triangle size and spatial tree searches for mesh integrity checking can improve upon this result. However, such improvements will come at the cost of ease of implementation. For now, a different approach is suggested. The following chapter presents a grid-based ridge tracking algorithm for three-dimensional LCS computations that enables efficient mesh generation and large savings in computational time.

Chapter 5

Grid-based Ridge Tracking Algorithm

5.1 Introduction

As discussed in Chapter 4, one of the largest hurdles to more widespread use of LCS techniques is the large time required for particle advections used to compute the FTLE. This chapter presents a grid-based ridge tracking algorithm to simplify mesh generation in three-dimensional LCS computations. In fact, most of this algorithm may be used for $n > 0$ dimensions, but the surface triangulation used here is specific to two-dimensional surfaces in a three-dimensional domain. All computations are performed on a predetermined orthogonal grid to simplify implementation and surface triangulation. The gridless algorithm of Chapter 4 requires significant computational time to generate surface meshes, but using a fixed grid enables efficient generate of surface triangulations via a lookup table similar to the marching cubes algorithm that is used for computing isosurfaces [59]. FTLE ridges are initially detected by computing the FTLE values on a series of lines across the domain. Local maxima along these lines occur at the FTLE ridge crossings. Once a series of points on the ridges have been detected, nearby points are tested to see if they are also on the ridge. This process is repeated to track the ridges through the entire domain. By performing computations only near the LCS surfaces, the order of the algorithm is reduced from $\mathcal{O}\left(1/\delta x^3\right)$ to about $\mathcal{O}\left(1/\delta x^2\right)$. In contrast to the two-dimensional ridge tracking algorithm, the detected ridge points are not advected to initialize the next time step. This is because the ridge detection step scales as $\mathcal{O}\left(1/\delta x\right)$ and is not an significant part of the computational cost when compared to the surface tracking step that scales as $\mathcal{O}\left(1/\delta x^2\right)$. Additionally, it is not necessary to detect the ridge orientation for the grid based algorithm.

The algorithm is verified using several test cases, including a time dependent double gyre, Arnold-Beltrami-Childress (ABC) flow, and a swimming jellyfish. These results establish the computational order of the algorithm. The scaling of computational time versus area of the LCS surfaces is also studied and scales as $C + \mathcal{O}\left(A_{\text{LCS}}\right)$ where C is a constant initialization cost and A_{LCS} is the area of the LCS surfaces. Finally, the ridge tracking algorithm offers several other advantages beyond the savings in computational time. Since the LCS surfaces are directly computed, visualization of the LCS is simplified and the output files store only the LCS surface triangulation and so are much smaller than storing the entire FTLE field.

5.2 A grid-based surface tracking algorithm

The computational savings seen by using a ridge tracking algorithm come from avoiding unnecessary computations away from the FTLE ridges. This process is divided into three steps as described in Algorithm 3: detecting initial points on the ridge surfaces, tracking the ridges through space, and triangulating the ridge points into a ridge surface. The initial ridge detection is handled by detecting where lines through the domain cross the ridge surfaces. The ridges are then iteratively grown by searching for nearby ridge points until no new ridge points are found. Finally, the use of a predetermined grid throughout this process allows the use of a lookup table to efficiently generate a triangulation of the resulting LCS surfaces.

Algorithm 3 Grid-based ridge tracking algorithm

Set integration time, T, and spatial step, δx

for $t = t_0$ **to** t_f **do**

  Perform initial ridge detection

  while new ridge points are found **do**

    Check surrounding points for additional ridge points

    Ensure FTLE values exceed threshold

    Add points to list of ridge points

  end while

  Generate surface triangulation from lookup table

end for

Save or plot the parameterized LCS surfaces

5.2.1 Initial ridge detection

If a hiker walks in a straight line, constantly monitoring his altitude, he will reach a locally maximum altitude upon crossing a ridge in the terrain. Similarly, if the FTLE is known along a line through a three-dimensional domain, local maxima along the line occur where the line crosses FTLE ridges. This algorithm restricts computations to a fixed grid with spacing δx and initially detects the FTLE ridges by computing the FTLE values along a set of lines that cross the domain just as in the gridless ridge tracking algorithm of Chapter 4 and shown in Figure 4.7. The number and spacing of the lines is determined by the user and depends on the expected spatial extend of the LCS in the flow. If an LCS surface does not intersect any of the lines used and is isolated from other LCS it may be missed entirely.

Once the FTLE values along these lines are known, local maxima on each line are detected and grid ridge points are defined as follows:

Definition *Grid Ridge Point*: Given an orthogonal grid in $\mathbb{R}^n$ with coordinate directions $\mathbf{e}_i : i \in \{1, ..., n\}$; a grid point $\mathbf{x}_0$ is a *grid ridge point* of a function F if $F(\mathbf{x}_0) \geq F(\mathbf{x}_0 \pm \mathbf{e}_i)$

for at least one $i \in \{1, ..., n\}$.

Lekien et al. used a similar ridge definition for LCS in n-dimensions, but also required that the deformation tensor, Δ, have one eigenvalue greater than 1 and all other eigenvalues less than one (requiring the LCS to generate Lagrangian stretching in only one dimension) [51]. The local maxima on each line are grid ridge points and FTLE ridge tracking is initialized at these points. In the limit as grid spacing goes to zero, the grid ridge points converge to coordinate local maxima, defined as:

Definition *Coordinate local maximum*: A point $\mathbf{x}_0$ is a coordinate local maximum of the function F with respect to coordinate direction $\mathbf{e}_i$ if there is a value $\varepsilon > 0$ such that $F(\mathbf{x}_0) > F(\mathbf{x}_0 + \delta \cdot \mathbf{e}_i)$ for all $\delta < \varepsilon$.

Note that coordinate local maxima are not typically ridge points (unless the gradient along the ridge is zero or the ridge is aligned with a coordinate direction), but the are typically very close to a ridge, at least for well defined ridges. This property, as well as convergence of grid ridge points to coordinate local maxima, is proven below in Section 5.3.

It is also desirable to set a threshold for the FTLE ridge values at this time. Only detecting ridges with FTLE values above some threshold ensures that only the strongest LCS are revealed. This is often done by restricting the ridges to have FTLE values above a percentage of the maximum FTLE values that are detected. As in Chapter 4, $50 - 80\%$ of the maximum value is typically an adequate choice. Although the grid ridge points may exhibit false positives in the sense that such points may not necessarily correspond to true ridges defined as

Definition *Ridge*: A *ridge* of a C^2 function F is a codimension-one surface S satisfying

(1) The vectors $\mathbf{n} \cdot \nabla F = 0$ for all points on S where $\mathbf{n}$ is a unit vector normal to S.

(2) $\mathbf{n}^\intercal H \mathbf{n} = \min_{\|\mathbf{u}\|=1} (\mathbf{u}^\intercal H \mathbf{u}) < 0$ for all points on S where H is the Hessian matrix associated with F.

experience has shown that such occurrences are limited and typically do not alter the topology of the resulting ridge surfaces.

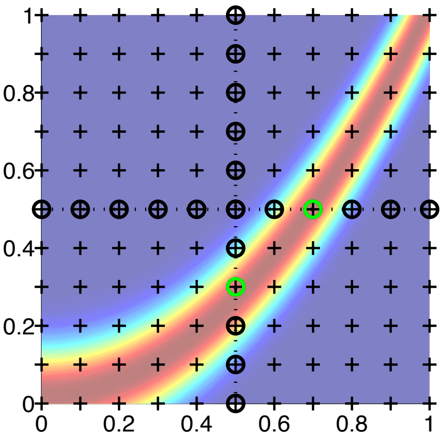
5.2.2 Ridge tracking

The heart of this algorithm lies in tracking the ridges outward from the initially detected grid ridge points. A schematic of this process for a two dimensional example is shown in Figure 5.1. The neighbors of each initially detected grid ridge point are checked to see if any meet the criteria to be grid ridge points (Figure 5.1b). In three dimensions, points in the grid are indexed by (i, j, k) for the x, y, and z directions. For each newly detected grid ridge point, (i, j, k), the neighboring points in $[i-1, i+1] \times [j-1, j+1] \times [k-1, k+1]$ are checked to see if any are grid ridge points. If new grid ridge points are found that lie above the FTLE threshold, the neighbors of those points are checked. This process is repeated until no new grid ridge points are found.

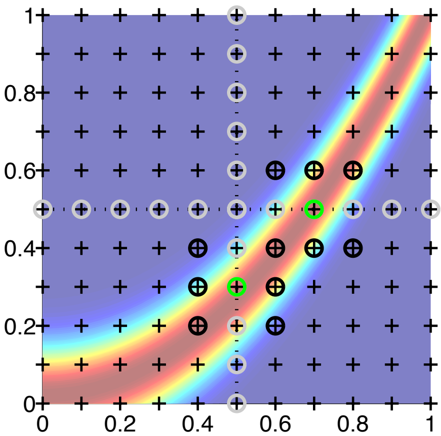
It is important to note that this algorithm detects *grid ridge points* as defined above, and therefore detects *coordinate local maxima* (also defined above). If the ridge height is not constant (i.e. $\nabla F \neq \mathbf{0}$ on the ridge) and the ridges are not aligned with a coordinate direction, the grid ridge points will not converge to the actual ridges. Section 5.3 provides an error bound and corresponding proofs addressing this issue, but for well defined ridges the error is typically smaller than reasonable grid spacings.

5.2.3 Surface triangulation

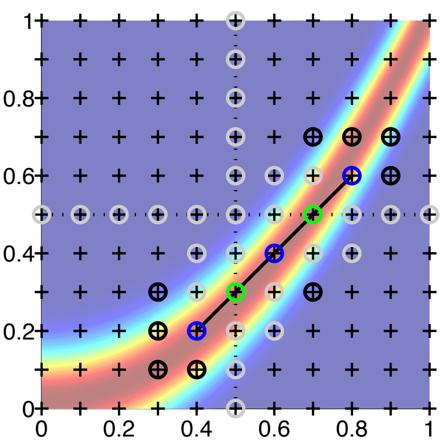
At the end of the ridge tracking process, a large list of grid ridge points is obtained. These points must be connected into a surface triangulation for visualization and further analysis. Initial attempts at developing a three-dimensional ridge tracking algorithm revealed that gridless techniques requiring surface meshing where both difficult to implement and computationally expensive because of the surface meshing process (see Chapter 4). By using a grid-based coordinate system, it is possible to very quickly generate a surface triangulation



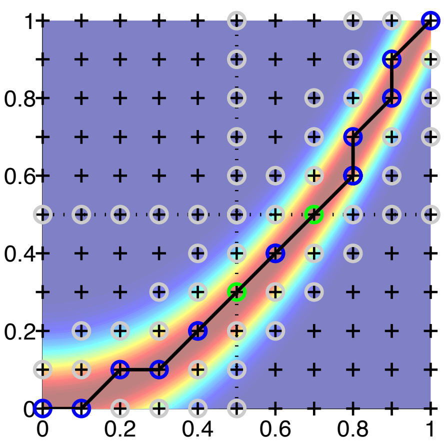
(a) Initial grid ridge points detection step, green circles are grid ridge points.



(b) The next set of points to check for ridges (black ○'s) and previously checked points (gray ○'s).



(c) New grid ridges (blue), new points to check (black) and previously checked points (gray).



(d) The resulting LCS.

Figure 5.1: The ridge tracking process in two dimensions. The background color represents the FTLE field, +'s mark grid points. The three-dimensional process is analogous, but a surface triangulation is used instead of line segments.

from a lookup table similar to the process used in the marching cubes algorithm that is popular for isosurface generation [59].

After all the grid ridge points have been found, each cubic element in the domain is given an 8-bit number corresponding to the configuration of the grid ridge points on the element's 8 vertices. The 256 possible configurations are listed in a lookup table which directly converts the 8-bit number to a triangulation of the grid ridge points in the element.

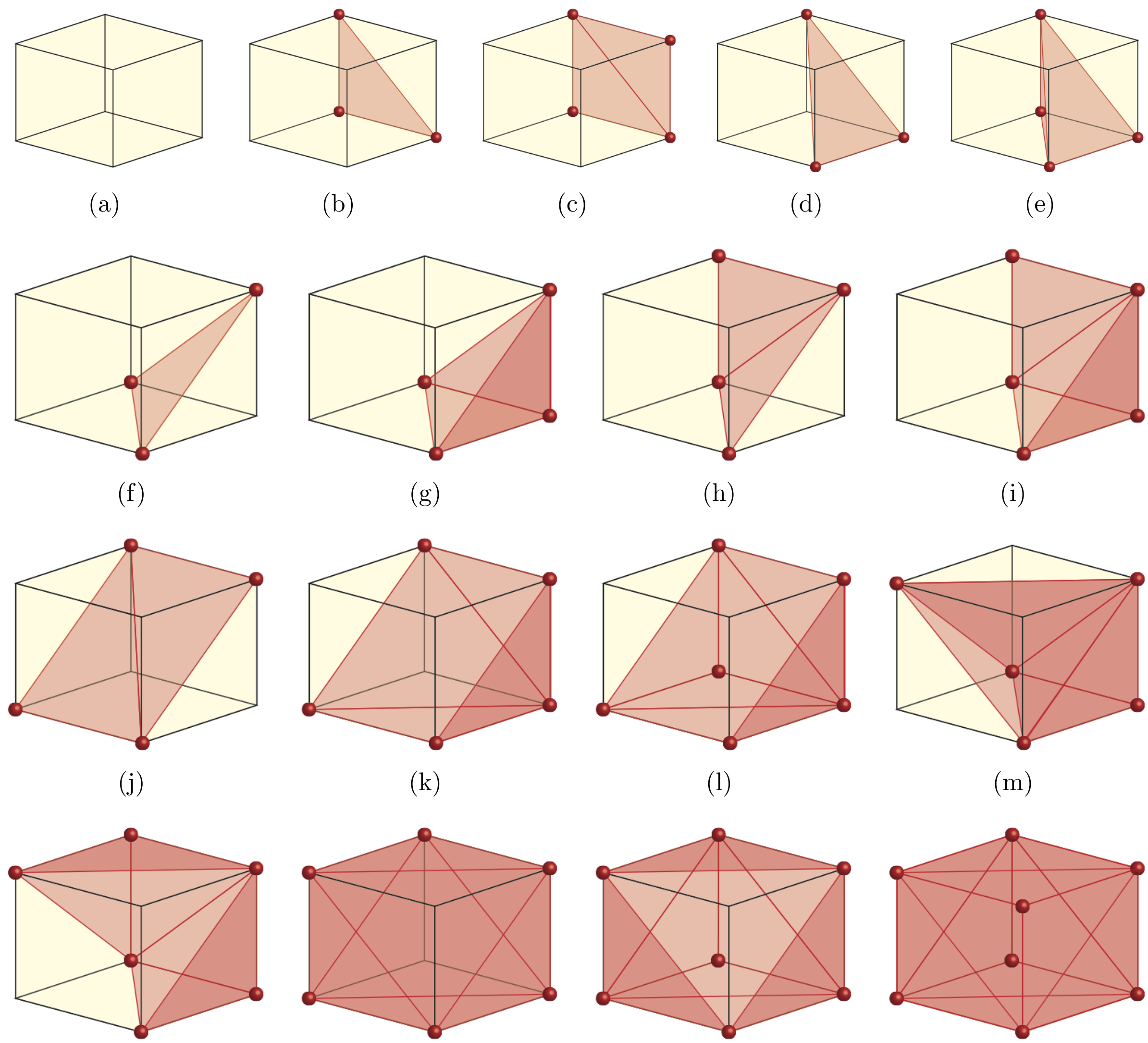
The lookup table is generated by using reflections and rotations to reduce the problem to the 17 canonical cases shown in Fig. 5.2. Elements containing two or fewer grid ridge points contain no LCS surface triangles while elements containing 3 or more grid ridge points are added to the surface triangulation.

The lookup table makes the surface triangulation step in this algorithm extremely efficient, but there is some ambiguity in the precise triangulation that should be used for certain ridge point configurations. This is a well known problem in isosurface construction and is commonly dealt with by making assumptions about the field at the subgrid scale and testing additional points within the element. Similar tests may be possible for the FLTE ridges, but surface patches were chosen to err in favor of adding extra triangles to these ambiguous cases surfaces to avoid gaps in the LCS surfaces.

5.3 Algorithm properties

This section focuses on the error and convergence of the grid-based ridge tracking algorithm. Well defined ridges are detected by the ridge tracking algorithm and the error in the location of the ridges is typically very small. The theorems contained below rely on the ridge definition presented above in Section 5.2. A few important properties of ridges are summarized here for convenience:

- ∇F is parallel to the ridge.
- The eigenvectors $\{\mathbf{v}_i\}$ of the Hessian H of F form a complete, orthonormal basis for



(a) (b) (c) (d) (e) (f) (g) (h) (i) (j) (k) (l) (m) (n) (o) (p) (q)

Figure 5.2: The canonical cases used in the lookup table for surface triangulation. All other configurations can be generated via reflections and rotations of these cases.

the space $\mathbb{R}^n$ since H is symmetric.

- The eigenvector $\mathbf{v}_1$ associated with the minimum eigenvector λ_1 of H is normal to the ridge.

To prove that the ridge tracking algorithm accurately detects ridges, it is first shown that grid ridges converge to coordinate local maxima and then shown that well defined ridges always have a nearby coordinate local maximum. An explicit bound on the distance between a ridge and the nearest coordinate local maximum is given.

I first show that coordinate local maxima are approximated by grid ridge points.

Theorem 1. *Let F be a C^2 continuous function and let L be a line parallel to grid direction $\mathbf{e}_i$ and through grid points $\{\mathbf{x}_j\}$. Assume $\mathbf{x}^*$ on L is a local maximum of F in the $\mathbf{e}_i$ direction and the second derivative of F in the $\mathbf{e}_i$ direction is negative. There exist an ε such that if the grid spacing, dx satisfies $dx < \varepsilon$, then $\|\mathbf{x}^* - \mathbf{x}^g\| \leq dx$ for some grid ridge point $\mathbf{x}^g$*

Proof. Since $\mathbf{x}^*$ is a local maximum in the $\mathbf{e}_i$ direction, there is a value $\varepsilon_1 > 0$ such that for all $|c_1| < \varepsilon_1$, $F(\mathbf{x}^* + c_1\mathbf{e}_i) < F(\mathbf{x}^*)$. Since F is C^2 continuous and $\partial^2 F/\partial \mathbf{e}_i{}^2 < 0$, $\exists\, \varepsilon \in (0, \varepsilon_1)$ such that

$$\left.\frac{\partial^2 F}{\partial \mathbf{e}_i{}^2}\right|_{\mathbf{x}^*+c\mathbf{e}_i} < 0 \;\forall\; |c| < \varepsilon. \tag{5.1}$$

This implies that $\partial F/\partial \mathbf{e}_i$ is monotonically decreasing over the interval $c \in (-\varepsilon, \varepsilon)$.

Next, choose a grid spacing, $dx = \varepsilon/2 < \varepsilon$. This ensures that at least five grid points lie in the interval $(\mathbf{x}^* - \varepsilon\mathbf{e}_i, \mathbf{x}^* + \varepsilon\mathbf{e}_i)$ and at least two lie to each side of $\mathbf{x}^*$.

If $\mathbf{x}^*$ is a grid point, then the neighboring grid points have lower values of F since $dx < \varepsilon < \varepsilon_1$ so this grid point is a grid ridge point and there is a grid ridge point less than dx from $\mathbf{x}^*$.

If $\mathbf{x}^*$ is not a grid point, label the nearest four grid points $\mathbf{x}_1$, $\mathbf{x}$, $\mathbf{x}_3$, $\mathbf{x}_4$, with $\mathbf{x}^*$ lying between $\mathbf{x}$ and $\mathbf{x}_3$. Since $\partial F/\partial \mathbf{e}_i$ is monotonically decreasing over the interval and $\partial F/\partial \mathbf{e}_i|_{\mathbf{x}^*} = 0$, $F|_{\mathbf{x}_1} < F|_{\mathbf{x}}$ and $F|_{\mathbf{x}_3} > F|_{\mathbf{x}_4}$. If $F|_{\mathbf{x}} > F|_{\mathbf{x}_3}$, $\mathbf{x}$ is a grid ridge point. If

$F|_{\mathbf{x}} < F|_{\mathbf{x}_3}$, $\mathbf{x}_3$ is a grid ridge point. If $F|_{\mathbf{x}} = F|_{\mathbf{x}_3}$, both $\mathbf{x}$ and $\mathbf{x}_3$ are grid ridge points. In any of these cases, since $\mathbf{x}^*$ lies between $\mathbf{x}$ and $\mathbf{x}_3$, there is a grid ridge point less than dx from $\mathbf{x}^*$. $\square$

Theorem 1 proves that as grid spacing goes to zero, there is a grid ridge point that converges to the coordinate local maxima of F. Therefore, the ridge tracking algorithm detects coordinate local maxima. The relationship between ridges and coordinate local maxima is shown below. First, two additional properties must be shown that will be used in the proof of an error bound for the ridge tracking algorithm.

First, I give a lower bound on the maximum dot product between an arbitrary unit vector and a set of orthonormal basis vectors. This bound guarantees that there must be at least one basis vector that is within a certain angle (dependent on the dimension of the space) of any arbitrary vector.

Theorem 2. *Given an orthonormal basis $\{\mathbf{e}_i\}$ for $\mathbb{R}^n$ and an arbitrary unit vector $\mathbf{v} \in \mathbb{R}^n$ there is a basis vector $\mathbf{e}^*$ that maximizes $|\mathbf{v} \cdot \mathbf{e}_i|$ and $|\mathbf{v} \cdot \mathbf{e}^*| \geq 1/\sqrt{n}$.*

Proof. Existence of the maximum is implied by the extreme value theorem since the dot product is a continuous function and the set $\{\mathbf{e}_i\}$ is finite and therefore closed and bounded. The lower bound on the maximum is as follows: Let $\mathbf{e}^*$ be the basis vector that maximizes

$|\mathbf{v}\cdot\mathbf{e}_i|$ so $|\mathbf{v}\cdot\mathbf{e}_i| \le |\mathbf{v}\cdot\mathbf{e}^*|$. Assume $|\mathbf{v}\cdot\mathbf{e}^*| < 1/\sqrt{n}$. Then

$$\begin{aligned}
1 = \|\mathbf{v}\| &= \left\|\sum_{i=1}^{n}(\mathbf{e}_i\cdot\mathbf{v})\mathbf{e}_i\right\| \\
&= \sqrt{\left(\sum_{i=1}^{n}(\mathbf{e}_i\cdot\mathbf{v})\mathbf{e}_i\right)\cdot\left(\sum_{i=1}^{n}(\mathbf{e}_i\cdot\mathbf{v})\mathbf{e}_i\right)} \\
&= \sqrt{\sum_{i=1}^{n}(\mathbf{e}_i\cdot\mathbf{v})^2} \\
&\le \sqrt{\sum_{i=1}^{n}(\mathbf{e}^*\cdot\mathbf{v})^2} = \sqrt{n(\mathbf{e}^*\cdot\mathbf{v})^2} \\
&< \sqrt{n(1/\sqrt{n})^2} \\
&< 1
\end{aligned}$$

which is a contradiction so $|\mathbf{v}\cdot\mathbf{e}^*| \ge 1/\sqrt{n}$. □

Next, I prove a bound on the roots of a specific type of polynomial. Given certain bounds on the coefficients, there must be a root within a certain distance from the origin. Intuitively, if the second derivative of a parabola is small compared the ratio of the slope at $x = 0$ and its y-intercept, then the parabola must have a real root within some maximum distance from the origin.

Theorem 3. *Given a quadratic equation $y = ax^2 + bx + c$ with bounds on the real valued constant coefficients*

$$\begin{aligned}
|c| &\le c_1 \\
0 < b_0 < |b| &\le b_1 \\
|a| &\le \frac{b_0^2}{4c_1}
\end{aligned}$$

there is a real root x^* such that $|x^*| < \dfrac{2c_1}{b_0}$.

Proof. If $a = 0$, there is a single root at $x^* = -c/b$ and $|x^*| = |-c/b| < 2c_1/b_0$.

If $a \neq 0$, the roots are given by the alternate form of the quadratic formula

$$x = \frac{2c}{-b \pm \sqrt{b^2 - 4ac}}.$$

The discriminant, $\Delta = b^2 - 4ac > b_0^2 - 4\dfrac{b_0^2}{4c_1}c_1 = 0$ is positive so the quadratic has two real roots. Denote these roots

$$x_1 = \frac{2c}{-b + \sqrt{b^2 - 4ac}}$$
$$x_2 = \frac{2c}{-b - \sqrt{b^2 - 4ac}}.$$

If $b \leq 0$ let $x^* = x_1$, otherwise let $x^* = x_2$. Then

$$|x^*| = \frac{2\,|c|}{|b| - \sqrt{b^2 - 4ac}} < \frac{2c_1}{b_0}.$$

□

Finally, I prove an error bound on the difference between coordinate local maxima and ridges. Since there is always a grid ridge point that converges to the coordinate local maxima, this is also the error of the ridge tracking algorithm as the grid spacing goes to zero.

Theorem 4. *Given a C^3 continuous function F that admits a sufficiently sharp second derivative ridge, for every point $\mathbf{x}_0$ on the ridge, there is a nearby point, $\mathbf{x}^*$, in the ridge normal direction that is a coordinate local maximum. The nearest coordinate local maximum is no further from the ridge than*

$$d = \frac{2\sqrt{n-1}\,\|\mathbf{D}F(\mathbf{x}_0)\|}{|\lambda_1|},$$

as long as

$$\left\|\mathbf{v}_1 \cdot (\mathbf{D}^3 F)\right\| < \frac{1}{2}\frac{\lambda_1^2}{n\sqrt{n-1}\,\|\mathbf{D}F(\mathbf{x}_0)\|}$$

and

$$\lambda_n \le \frac{n}{n-1}\left|\frac{\lambda_1}{n} + \frac{2\sqrt{n-1}}{|\lambda_1|}\right| \|\mathbf{D}F(\mathbf{x}_0)\| \, \|\mathbf{v}_1 \cdot \mathbf{D}F(\xi)\| \, ,$$

where $\mathbf{D}$ *denotes the gradient operator,* λ_1 *(respectively* λ_n*) is the minimum (respectively maximum) eigenvalue of the Hessian,* $\mathbf{D}^2F(\mathbf{x}_0)$*,* $\mathbf{v}_1$ *and* $\mathbf{v}_n$ *are the eigenvectors associated with* λ_1 *and* λ_n*, and* n *is the dimension of the space.*

Although these criteria may seem restrictive, they are typically easily satisfied by well defined ridges as is shown in examples below. λ_1 is typically very large in magnitude (sometimes up to 10^{12}) while $\|\mathbf{D}F\|$ and $|\lambda_n|$ are typically $\mathcal{O}(1)$. If λ_n is negative the last condition is trivially satisfied.

Proof. By assumption F is C^3 continuous and admits a second derivative ridge through point $\mathbf{x}_0$. Denote the eigenvalues and normalized eigenvectors of the Hessian $\mathbf{D}^2F(\mathbf{x}_0)$ as $\lambda_1 \le \lambda_2 \le \cdots \le \lambda_n$ and $\mathbf{v}_1, \mathbf{v}, \ldots, \mathbf{v}_n$. By definition, $\mathbf{v_1}$ is normal to the ridge and $\mathbf{D}F(\mathbf{x}_0)$ is parallel to the ridge. Also, $\{\mathbf{v}_1, \ldots, \mathbf{v}_n\}$ forms an orthonormal basis for the space since the Hessian is a real symmetric matrix. Also, let the coordinate system for the space be defined by the orthonormal basis vectors $\{\mathbf{e}_1, \ldots, \mathbf{e}_n\}$.

I first choose the coordinate direction that is closest to the ridge normal direction. That is, choose $\mathbf{e}^*$ from the set $\{\mathbf{e}_i\}$ such that $|\mathbf{e}^* \cdot \mathbf{v}_1| \ge |\mathbf{e}_i \cdot \mathbf{v}_1| \;\; \forall\, i \in \{1, \ldots, n\}$. Theorem 2 gives that

$$|\mathbf{e}^* \cdot \mathbf{v}_1| \ge 1/\sqrt{n}. \tag{5.2}$$

Using a Taylor series approximation centered at $\mathbf{x}_0$, the gradient of F may be written as

$$\mathbf{D}F(\mathbf{x}) = \mathbf{D}F(\mathbf{x}_0) + ((\mathbf{x}-\mathbf{x}_0)\cdot\mathbf{D})\,\mathbf{D}F(\mathbf{x}_0) + \frac{1}{2!}\left((\mathbf{x}-\mathbf{x}_0)\cdot\mathbf{D}\right)^2\mathbf{D}F(\xi)$$

for some ξ that is a linear combination of $\mathbf{x}$ and $\mathbf{x}_0$. Taking the dot product of $\mathbf{D}F(\mathbf{x})$ with $\mathbf{e}^*$ and considering only points on a line normal to the ridge such that $\mathbf{x} = \mathbf{x}_0 + d\mathbf{v}_1$ gives

$$\mathbf{e}^* \cdot \mathbf{D}F(\mathbf{x}) = \mathbf{e}^* \cdot \mathbf{D}F(\mathbf{x}_0) + d\mathbf{e}^* \cdot \{(\mathbf{v}_1\cdot\mathbf{D})\mathbf{D}F(\mathbf{x}_0)\} + \frac{1}{2!}d^2\mathbf{e}^* \cdot \left\{(\mathbf{v}_1\cdot\mathbf{D})^2\mathbf{D}F(\xi)\right\}.$$

Since $\mathbf{v}_1$ is an eigenvalue of $D^2F(\mathbf{x}_0)$, this can be rewritten as

$$\mathbf{e}^* \cdot \mathbf{D}F(\mathbf{x}) = \mathbf{e}^* \cdot \mathbf{D}F(\mathbf{x}_0) + d\lambda_1 \mathbf{e}^* \cdot \mathbf{v}_1 + \frac{1}{2!} d^2 \mathbf{e}^* \cdot \left(\mathbf{v}_1 \cdot \left[\mathbf{v}_1 \cdot \left\{\mathbf{D}^3 F(\xi)\right\}\right]\right) \tag{5.3}$$

An upper bound on $|\mathbf{e}^* \cdot \mathbf{D}F(\mathbf{x}_0)|$ is established by noting that $\mathbf{v}_1 \cdot \mathbf{D}F(\mathbf{x}_0) = 0$ and expressing $\mathbf{e}^*$ and $\mathbf{D}F(\mathbf{x}_0)$ in terms of the basis $\{\mathbf{v}_i\}$:

$$\begin{aligned}
|\mathbf{e}^* \cdot \mathbf{D}F(\mathbf{x}_0)| &= \left| \left(\sum_{i=1}^{n} (\mathbf{v}_i \cdot \mathbf{e}^*) \mathbf{v}_i \right) \cdot \left(\sum_{i=2}^{n} (\mathbf{v}_i \cdot \mathbf{D}F(\mathbf{x}_0)) \mathbf{v}_i \right) \right| \\
&= \left| \sum_{i=2}^{n} (\mathbf{v}_i \cdot \mathbf{e}^*)(\mathbf{v}_i \cdot \mathbf{D}F(\mathbf{x}_0)) \right| \\
&= \left| \left(\sum_{i=2}^{n} (\mathbf{v}_i \cdot \mathbf{e}^*) \mathbf{v}_i \right) \cdot \left(\sum_{i=2}^{n} (\mathbf{v}_i \cdot \mathbf{D}F(\mathbf{x}_0)) \mathbf{v}_i \right) \right| \\
&\leq \left\| \sum_{i=2}^{n} (\mathbf{v}_i \cdot \mathbf{e}^*) \mathbf{v}_i \right\| \left\| \sum_{i=2}^{n} (\mathbf{v}_i \cdot \mathbf{D}F(\mathbf{x}_0)) \mathbf{v}_i \right\| \\
&\leq \sqrt{\sum_{i=2}^{n} (\mathbf{v}_i \cdot \mathbf{e}^*)^2} \sqrt{\sum_{i=2}^{n} (\mathbf{v}_i \cdot \mathbf{D}F(\mathbf{x}_0))^2} \\
&\leq \sqrt{1 - \frac{1}{n}} \left\| \mathbf{D}F(\mathbf{x}_0) \right\| .
\end{aligned} \tag{5.4}$$

The right hand side of Eq. 5.3 is a quadratic function in d of the form

$$\mathbf{e}^* \cdot \mathbf{D}F(\mathbf{x}) = ad^2 + bd + c$$

where

$$\begin{aligned}
a &= \frac{1}{2} \mathbf{e}^* \cdot (\mathbf{v}_1 \cdot [\mathbf{v}_1 \cdot \mathbf{D}^3 F(\xi)]), \\
b &= \lambda_1 \mathbf{e}^* \cdot \mathbf{v}_1, \\
c &= \mathbf{e}^* \cdot \mathbf{D}F(\mathbf{x}_0).
\end{aligned}$$

Eq. 5.4 establishes a bound on c, b is trivial bounded by the value of λ_1 and the fact that $\mathbf{e}^*$ and $\mathbf{v}_1$ are unit vectors, and a is bounded by the assumption that $\|\mathbf{v}_1 \cdot \mathbf{D}^3 F\|$ is bounded.

The bounds on the coefficients are

$$\begin{aligned} 0 \le |c| \le c_1 &= \sqrt{1-1/n}\,\|\mathbf{D}F(\mathbf{x}_0)\|\,, \\ 0 < b_0 = \frac{|\lambda_1|}{\sqrt{n}} &\le |b| \le b_1 = |\lambda_1|\,, \\ 0 \le |a| &\le \frac{\lambda_1^2}{4n\sqrt{n-1}\,\|\mathbf{D}F(\mathbf{x}_0)\|} < \frac{b_0^2}{4c_1}. \end{aligned}$$

Therefore, by Thm. 3, this function has a root at d^* with

$$|d^*| < \frac{2\sqrt{n-1}\,\|\mathbf{D}F(\mathbf{x}_0)\|}{\lambda_1}.$$

where $\mathbf{e}^* \cdot \mathbf{D}F(\mathbf{x}) = 0$.

We have now shown that the value of $\mathbf{e}_j \cdot \mathbf{D}F(\mathbf{x})$ is zero somewhere on the line normal to the ridge at $\mathbf{x}_0$ at a distance of less than

$$2\sqrt{(1-1/n)}\,\|\mathbf{DF}(\mathbf{x_0})\|\,/\lambda_1.$$

Call the zero point $\mathbf{x}^*$. To complete the proof, we will show that $\mathbf{e}^{*\intercal}(\mathbf{D}^2F)\mathbf{e}^*$ must be negative at this point so it is a coordinate local maximum. The first order Taylor series for the Hessian near $\mathbf{x}_0$ is

$$\mathbf{D}^2F(\mathbf{x}) = \mathbf{D}^2F(\mathbf{x}_0) + ((\mathbf{x}-\mathbf{x}_0)\cdot\mathbf{D})\mathbf{D}^2F(\xi).$$

At $\mathbf{x}^*$ the second derivative of F in the $\mathbf{e}^*$ direction is given by

$$\mathbf{e}^{*\intercal}(\mathbf{D}^2F(\mathbf{x}^*))\mathbf{e}^* = \mathbf{e}^{*\intercal}(\mathbf{D}^2F(\mathbf{x}_0))\mathbf{e}^* + d^*\mathbf{e}^{*\intercal}(\mathbf{v}_1\cdot\mathbf{D}^3F(\xi))\mathbf{e}^*$$

where $d^* < 2\sqrt{n-1}\,\|\mathbf{D}F(\mathbf{x}_0)\|\,/\lambda_1$ is the distance from the ridge of the point where $\mathbf{e}^* \cdot \mathbf{D}F(\mathbf{x}) = 0$. Then

$$\begin{aligned} \mathbf{e}^{*2}\mathbf{D}^2F(\mathbf{x}) &< \frac{\lambda_1}{n} + \left(1-\frac{1}{n}\right)\lambda_n + \frac{2\sqrt{n-1}\,\|\mathbf{D}F(\mathbf{x}_0)\|}{|\lambda_1|}\left\|\mathbf{v}_1\cdot(\mathbf{v}_1\cdot\mathbf{D}^3F(\xi))\right\| \\ &< 0 \end{aligned}$$

where $\|\mathbf{v}_1 \cdot (\mathbf{v}_1 \cdot \mathbf{D}^3 F(\xi))\|$ is understood as the operator norm (induced by the vector norm) of the matrix that results from the tensor dot product $\mathbf{v}_1 \cdot (\mathbf{v}_1 \cdot \mathbf{D}^3 F(\xi))$. Thus, at $\mathbf{x}^*$ the first derivative of F is zero in the $\mathbf{e}^*$ direction and the second derivative is negative. $\mathbf{x}^*$ is a coordinate local maximum.

□

As long the criteria of Theorem 1 are satisfied along a ridge, there is a nearby coordinate local maximum that will be detected by the ridge tracking algorithm. The error of the location of the detected ridges is bounded by

$$d = \frac{2\sqrt{n-1}\,\|\mathbf{D}F(\mathbf{x}_0)\|}{|\lambda_1|}.$$

$\|\mathbf{D}F\|$ is typically $\mathcal{O}\left(1\right)$ (the ridge does not rise and fall very rapidly) since in the limit as $T \to \infty$, the FTLE value is constant along ridges [85]. Therefore, the error is typically $\mathcal{O}\left(1/\left|\lambda_1\right|\right)$.

5.3.1 Ridge examples

In this section, the properties of the ridge tracking algorithm are verified in two two-dimensional examples: an analytically defined surface that admits a ridge and the FTLE field for the time dependent double gyre flow. The surface

$$F(x,y) = \frac{e^{-0.1(x-y)^2}}{\ln((x+y)^2+15)} \tag{5.5}$$

is examined first.

A contour plot of this function is shown in Figure 5.3 and the function admits a ridge along the line $y = x$. This represents a worst case scenario in terms of ridge orientation since the ridge is at a 45° angle to both coordinate directions. The ridge is also much thicker than is typically seen in well defined FTLE ridges. Since this ridge is defined by an analytical function, it is possible to easily compute all the necessary criteria for Theorem 4. In Figure 5.3, the actual ridge has been drawn as a solid black line. The error bound, d is

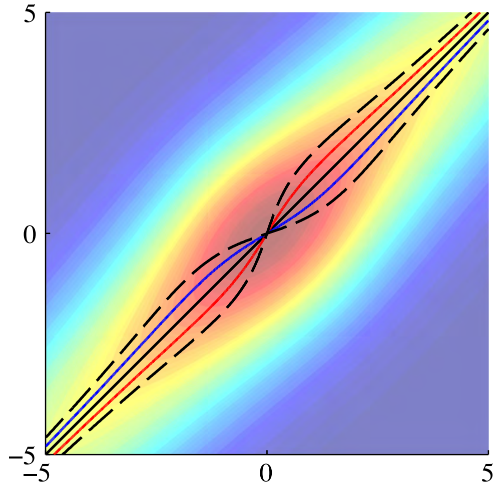


Figure 5.3: The function defined by Equation 5.5. The solid black line denotes the ridge while the dashed black curves show the error bound and the red and blue curves show the coordinate local maxima.

drawn in the figure as a pair of dashed black lines and the the coordinate local maxima have been drawn as red (for $\partial F/\partial x = 0$) and blue (for $\partial F/\partial y = 0$) lines. The other criteria of Theorem 4 are all satisfied despite λ_1 being much larger than is usually seen in FTLE ridges. In this case, $\lambda_1 \in [-0.1477, -0.0843]$, $\lambda_2 \in [-0.0363, 0.0071]$, and $\|\mathbf{D}F\| \leq 0.0356$ on the ridge in the domain $(x, y) \in [-5, 5]^2$.

A ridge in the FTLE field of the time dependent double gyre introduced in Section 4.2.6.1 is presented as a second example to apply Theorem 4. Here, the parameters $A = 0.1$, $\epsilon = 0.25$, and $\omega = 10$ are used for the double gyre and the integration time for computing the FTLE is set at $T = 15$. The forward time FTLE field at time $t = 0$ is shown in Figure 5.4.

The main FTLE ridge in this double gyre flow (shown in Figure 5.4b) was computed with very high precision by iteratively estimating the ridge position and tangent direction and then adjusting the position in the normal direction. This is necessary to accurately compute the FTLE values on the ridge as well as the gradient, and Hessian of the FTLE field to bound the error in the ridge locations. Note that the ridge seen in this FTLE field

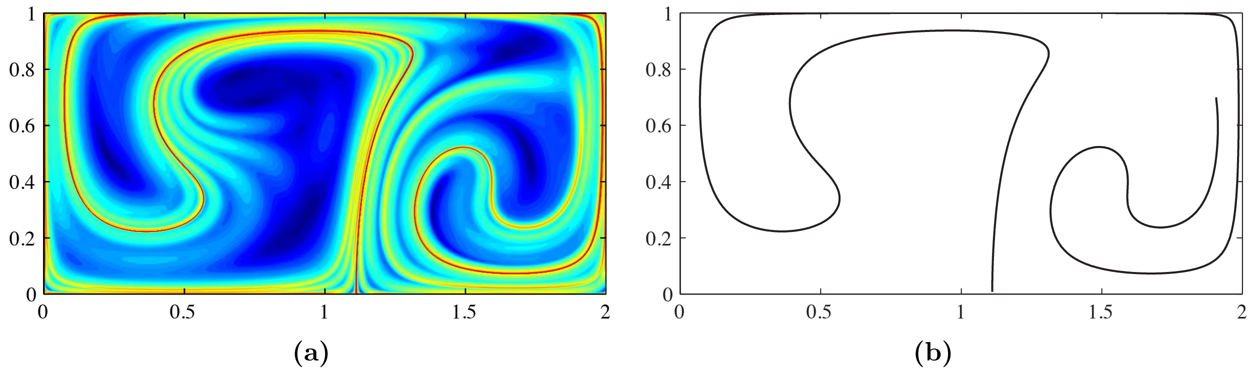


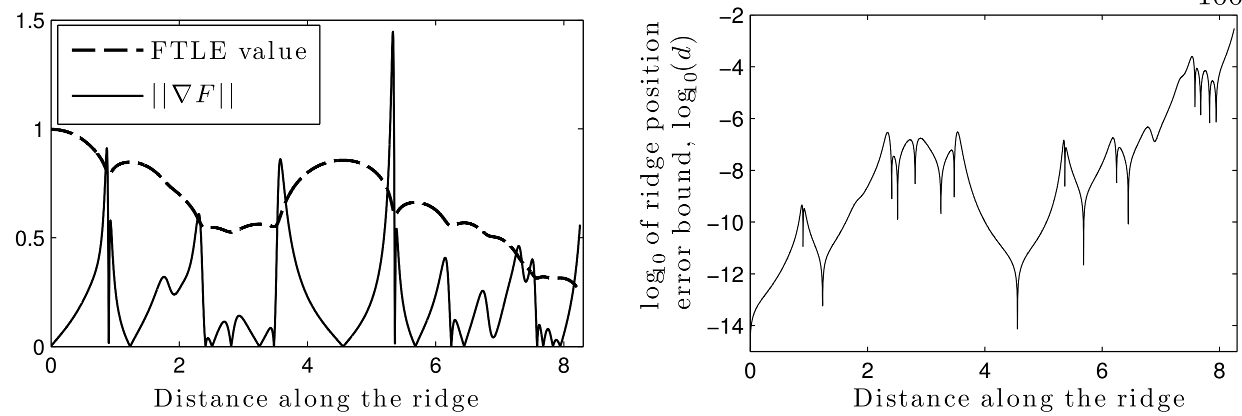
Figure 5.4: (a) The forward FTLE field and (b) the main FTLE ridge (with FTLE value > 0.25) for the time dependent double gyre with $A = 0.1$, $\epsilon = 0.25$, $\omega = 10$, $t = 0$, and $T = 15$.

is much sharper than the analytical ridge surface investigated in the previous example.

Figure 5.5 shows the FTLE values and the norm of the gradient of the FTLE field along the ridge as well as the bound on the ridge location error which is less than 10^{-6} for the majority of the ridge and never rises above 10^{-2}. The norm of the gradient is bounded by $\|\nabla F\| < 1.5$. Additionally the eigenvalues of the Hessian fall in the range $-1.8 \times 10^{12} < \lambda_1 < -367$ and $-6.1 < \lambda_2 < 37.3$ with averages of $\overline{\lambda_1} = -8.1 \times 10^{10}$ and $\overline{\lambda_2} = -0.033$ and medians $\mathrm{med}(\lambda_1) = -5.71 \times 10^7$ and $\mathrm{med}(\lambda_2) = -0.30$. The very large magnitude of λ_1 means that the grid based ridge tracking algorithm will be extremely accurate for this example. For reference, grid spacings of around 10^{-2} or 10^{-3} are typically used when performing LCS computations for this problem so the dominant error is caused by the grid discretization rather than the use of coordinate local maxima rather than true ridge points.

5.4 Algorithm performance

Performance of the ridge tracking algorithm is examined in this section by applying the algorithm to the time dependent double gyre, Arnold-Beltrami-Childress (ABC) flow, and a swimming jellyfish. The expected LCS surfaces are extracted from these flows and the



(a) **(b)**

Figure 5.5: (a) The FTLE values and gradient along the ridge shown in Figure 5.4b and (b) the bound on the ridge position error as given in Theorem 4. The distance along the ridge (x-axis) is computed as the distance along the ridge from the ridge origin at $(1.110, 0)$. Since $\|\nabla F\| = \mathcal{O}(1)$ and $|\lambda_1| \gg 1$, the error bound is very small, typically less than 10^{-6}.

computational order of the surface tracking algorithm is established and compared to the standard LCS algorithm that computes the FTLE field everywhere in the domain.

5.4.1 Time dependent double gyre

The first example presented is the time dependent double gyre used above in Section 5.3.1 and introduced in Section 4.2.6.1. As in the previous chapter, this flow is extended to three dimensions by setting the velocity in the z-direction to $w = 0$. Since there is no z-dependence and no velocity in the z-direction, the LCS will be independent of z as well. The parameters $A = 0.1$, $\epsilon = 0.1$, and $\omega = 2\pi/10$. are used and the integration time is set to $T = \pm 15$. Both the forward and backward LCS are computed. A threshold of 80% of the maximum FTLE value is used to determine the LCS.

The full three-dimensional LCS surfaces are shown in Figure 5.6. This figure also shows the backward FTLE field overlaid with the results of the three-dimensional ridge tracking algorithm. The LCS extracted by the ridge tracking algorithm lie exactly on top of the ridge in the FTLE field. The computational timing results are summarized below in Section 5.4.4.

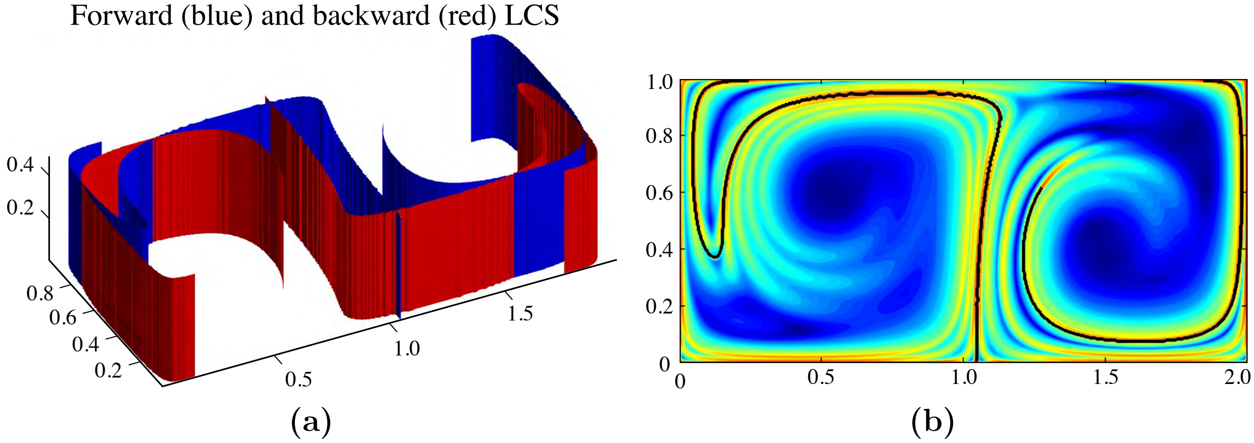


Figure 5.6: Surface tracking results for the double gyre flow. The three dimensional LCS are shown in (a) with forward LCS colored blue and backward LCS colored red. The forward FTLE field is shown in (b) with the ridge tracking results overlaid as the black curve that precisely lines up with the FTLE ridge.

5.4.2 Arnold-Beltrami-Childress flow

Arnold-Beltrami-Childress (ABC) flow is a three-dimensional, 2π-periodic flow that has been previously studied with LCS techniques [34] and is defined in detail in Section 4.3.4.2. To compute the LCS in the ABC flow, an integration time of $T = 10$ and a threshold value of 70% of the maximum FTLE value are used.

The LCS are shown in Figure 5.7 and agree with previously published results [28, 34]. Figure 5.7 also shows the FTLE field and corresponding LCS computed by the ridge tracking algorithm at a height of $z = \pi$. The complex LCS present in the ABC flow are a product of the non-trivial invariant manifolds of this flow. They clearly divide the flow into different regions that appear as tube-like structures through the flow domain. These tubes are dynamically distinct from one another and particles travel within and along the tubes without escaping to other regions of the space.

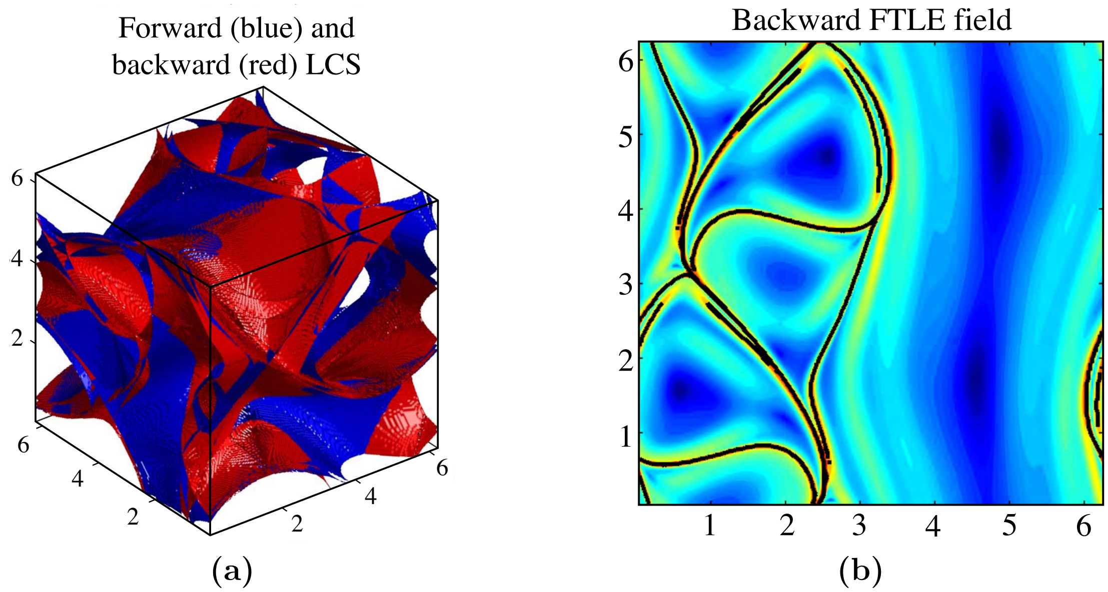


Figure 5.7: Ridge tracking results for ABC flow. The three-dimensional LCS are shown in (a) with forward LCS colored blue and backward LCS colored red. (b) shows a single plane at $z = \pi$ to display the FTLE field (colored) and corresponding LCS (black curves) as computed with the ridge tracking algorithm.

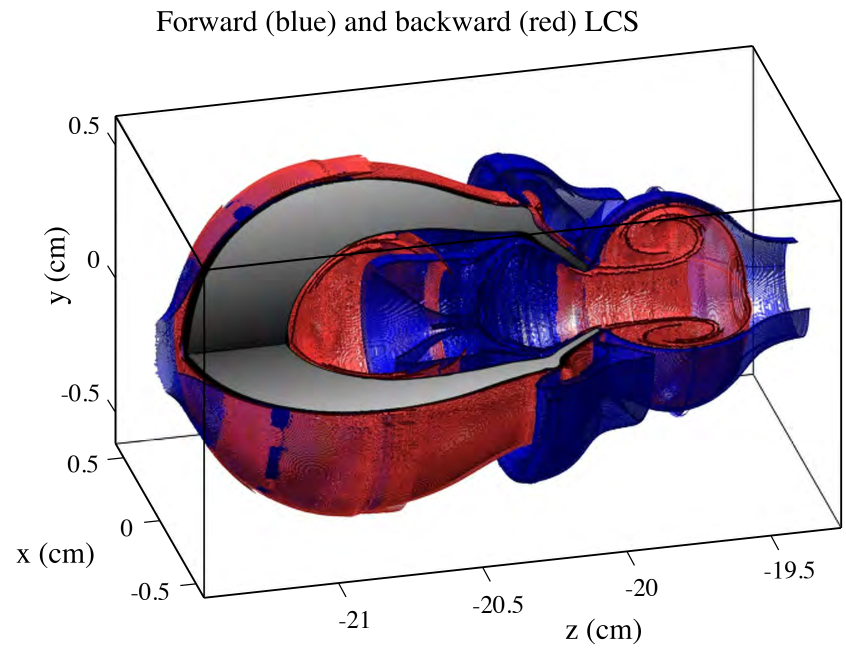
5.4.3 Swimming jellyfish

As a final example, the grid-based ridge tracking algorithm is used to compute the LCS created by a jetting type jellyfish, *Sarsia tubulosa*. This jellyfish is discussed in detail in Chapter 3 and full details of the procedure used to compute the jellyfish velocity field are found in Sahin and Mohseni [80] and Sahin et al. [81].

The axisymmetric velocity field is given on a moving, non-uniform quadrilateral mesh in (r, z)-coordinates. During LCS computations, the (x, y, z) coordinates are converted to (r, z) coordinates to compute the velocity and the the velocity is converted back to Cartesian coordinates for particle advections.

The jellyfish has a swimming period of 1 s between contractions and an integration time of 0.5 s was used to compute the LCS. Figure 5.8 shows the LCS as computed with the ridge tracking algorithm as well as the backward FTLE field computed with the standard LCS algorithm. The backward LCS (Figure 5.8a) clearly show a strong vortex being ejected

(a)



(b)

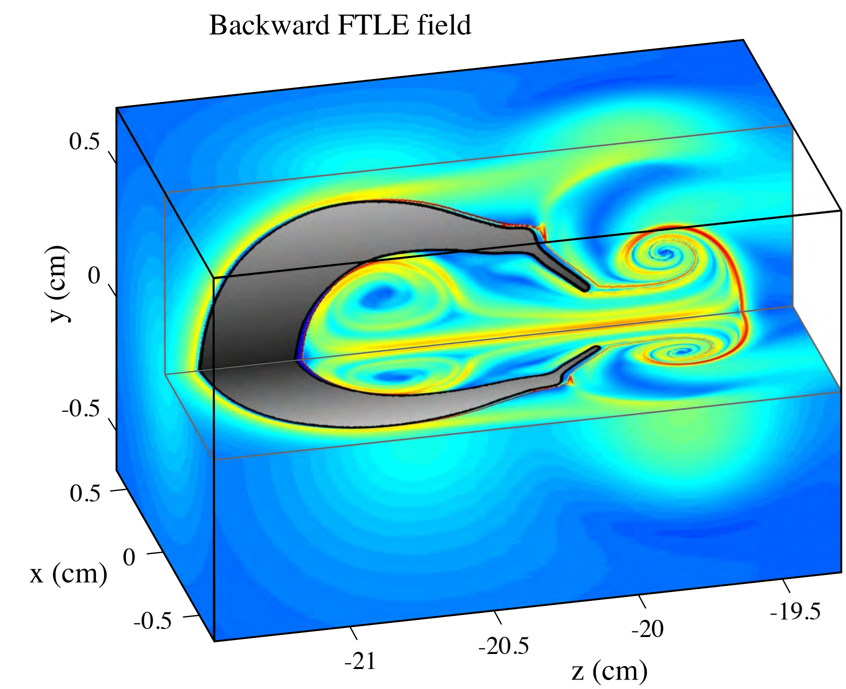


Figure 5.8: (a) The results of the ridge tracking algorithm and (b) the backward FTLE field for the swimming jellyfish. A strong vortex is being ejected near the end of the jellyfish's bell contraction. A cutaway view is shown so that the full LCS structure is visible.

as the jellyfish's bell contraction comes to an end. Additionally, the forward LCS outline fluid ahead of the vortex that will soon be entrained into the vortex ring as well as a region of fluid near the jellyfish bell that will be drawn into the bell during the relaxation phase of jellyfish swimming. These results are in excellent agreement with previously published LCS for this jellyfish [56] and the results presented in Chapter 3. Furthermore, this example clearly demonstrates the visualization advantages offered by the ridge tracking algorithm. Since the LCS surfaces are computed it is much simpler to visualize both the forward and backward LCS simultaneously as well as the three-dimensional nature of the structures.

5.4.4 Timing results

The computational time of the ridge tracking algorithm is expected to be $\mathcal{O}\left(1/\delta x^2\right)$ for a grid of spacing δx since computations are performed only near the two-dimensional ridge surfaces. To establish the computational order, the full FTLE field is computed using the standard algorithm (Algorithm 1) and the LCS surfaces are computed with the grid-based ridge tracking algorithm presented above for the double gyre, ABC, and jellyfish flows. Computations were performed at a variety of grid resolutions as well as on a single computer core and with a parallel code based on the message passing interface (MPI) libraries running on 16 or 48 cores. A least squares best fit was performed on a log-log scale for each case, assuming a fit of

$$t_f = C/(\delta x)^{\alpha}. \tag{5.6}$$

The resulting data points and curve fits are shown in Figure 5.9 and show that the standard algorithm is $\approx \mathcal{O}\left(1/\delta x^{3.0}\right)$ while the ridge tracking algorithm scales approximately as $\mathcal{O}\left(1/\delta x^{2.1}\right)$. This performance is maintained for both the serial and the parallel versions of the code.

It is worth noting that in the jellyfish example, for low resolutions, the time required to read the velocity data files and build the alternating digital trees used for search (see

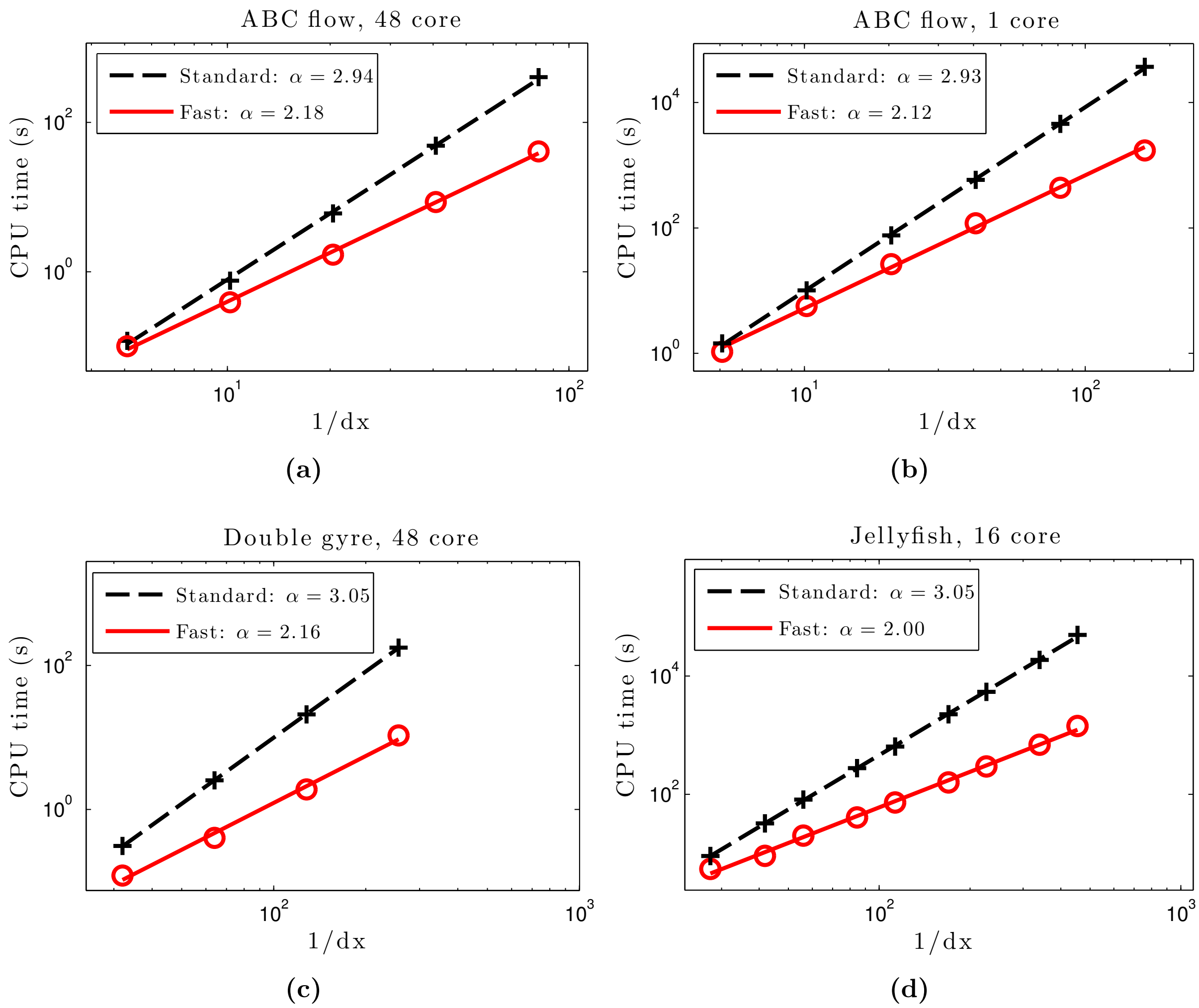


Figure 5.9: Timing results for the standard FTLE algorithm and the ridge tracking algorithm. α is the scaling exponent of the algorithm (CPU time $\approx \mathcal{O}(1/\delta x^{\alpha})$) and appears as the slope of the lines in these log-log plots. The standard algorithm scales as $\mathcal{O}(1/\delta x^{3.0})$ and the ridge tracking algorithm scales as $\mathcal{O}(1/\delta x^{2.1})$.

Section 3.2.2) at each time step is a significant part of the total computational time. Since the velocity read in time does not change with grid resolution, this has the effect of giving artificially low exponents for the algorithm order (both for the standard algorithm and the ridge tracking algorithm). To compensate for this effect and more accurately estimate the asymptotic order, the velocity read in and ADT creation time were subtracted from the total run time before computing the computational order for the jellyfish example. These modified times are reported in Fig. 5.9d. The resulting values for α closely match the values that result from using only the last few data points and represent a closer approximation of the asymptotic values of α.

It is also expected that the computational time should be directly proportional to the surface area of the LCS in the domain. This is tested by generating an artificial FTLE field that has ridges along pre-selected planes. The planes are defined by $z = 0.1(x + y) + z_i$ and the artificial FTLE field is given by

$$\sigma(x, y, z) = \sum_i \exp[-2000(z - 0.1(x + y) - z_i)^2]. \tag{5.7}$$

Each plane that defines an FTLE ridge is tilted slightly so that it is out of line with the grid to make detecting and tracking the ridge slightly more realistic. In the domain $[0, 1]^3$ each ridge has an area of $A_{\text{LCS}} = \sqrt{1.02}$ for $z_i \in (0, 0.8)$. Six different cases are tested, corresponding to 1-6 ridges in the domain. Particles are advected using the double gyre velocity field listed above to give realistic particle advection times, but instead of returning the true FTLE field for the double gyre flow, the artificial field of Equation 5.7 is returned. The results are listed in Table 5.1, which shows the computational time according to surface area for four different values of δx.

All four values of δx show a linear relationship between LCS surface area and CPU time and least squares fits result in the following regression coefficients for the fit $t_{\text{CPU}} = C_1 + C_2 A_{\text{LCS}}$ where A_{LCS} is the surface area of the LCS and t_{CPU} is the required CPU time:

CPU time (s)		Surface area					
		1.01	**2.02**	**3.03**	**4.04**	**5.05**	**6.06**
Grid Spacing	**1/64**	0.533	0.689	0.844	0.987	1.162	1.333
	1/128	2.713	3.306	3.913	4.625	5.385	6.121
	1/256	19.54	22.12	25.10	27.68	32.79	34.80
	1/512	192.8	204.5	215.7	235.7	244.7	259.9

Table 5.1: CPU time (in seconds) to compute the artificial FTLE ridges of Equation 5.7 for various surface areas and grid spacings.

δx	C_1	C_2
1/64	0.368	0.157
1/128	1.94	0.679
1/256	15.9	3.14
1/512	177.9	13.5

The relatively large values of C_1 for all these cases means that there is some initial cost regardless of the amount of LCS surface area. This is due to the cost associated with initializing data structures and performing the initial ridge detection step as described in Section 5.2.1. Additionally, C_1 appears to scale roughly as $\mathcal{O}\left(1/\delta x^3\right)$. Since the current implementation of the ridge tracking code uses and allocates full three-dimensional arrays rather than using sparse data structures, it is reasonable to expect this relationship. On the other hand, C_2 scales roughly as $\mathcal{O}(1/\delta x^2)$. This accounts for the ridge tracking part of the algorithm.

The $\mathcal{O}\left(1/\delta x^{3.0}\right)$ scaling of C_1 may explain why the overall computational order of the algorithm is $\mathcal{O}\left(1/\delta x^{2.1}\right)$ rather than $\mathcal{O}\left(1/\delta x^2\right)$. As resolution is increased beyond current capabilities, the initialization cost of may be expected to completely overwhelm the ridge tracking cost due to this difference in order. The implementation of sparse data structures would likely help solve this problem.

5.5 Grid-based ridge tracking conclusions

The ridge tracking algorithm presented in this chapter has been shown to reduce the order of LCS computations from $\mathcal{O}\left(1/\delta x^{3.0}\right)$ to about $\mathcal{O}\left(1/\delta x^{2.1}\right)$ for three-dimensional flows. This reduction in order allows potentially tremendous savings in computational time as the required LCS resolution is increased.

The effectiveness and algorithm properties have been demonstrated by several examples, including the analytically defined double gyre and ABC flow and the swimming jellyfish that is defined by velocity stored in data files. On single processor as well as multicore machines, the ridge tracking algorithm shows the expected change in computational order and provides large speed ups for all tested cases.

Mathematical proof has been given that although the ridge tracking algorithm detects *coordinate local maxima* rather than the actual ridges, for well defined ridges the associated error is small. The distance between a ridge and the surfaces detected by this algorithm is $\mathcal{O}\left(\|\mathbf{D}F\| / |\lambda_1|\right)$ where $\mathbf{D}F$ is the gradient along the ridge and $|\lambda_1|$ is the second derivative normal to the ridge (or, equivalently the smallest eigenvalue of the Hessian of F). In typical examples this error is smaller than the grid spacing and for well defined ridges it may be several orders of magnitude smaller than the grid spacing.

The general framework of this algorithm could easily be adapted to other LCS definitions or techniques. For example, the finite size Lyapunov exponent (FSLE) could easily be used in lieu of the FTLE. A two-dimensional version of this algorithm could also be modified to use the variational formulation for LCS which was recently proposed by Haller [26, 35].

Chapter 6

Applications of the 3D ridge tracking algorithm

6.1 Introduction

In this chapter the ridge tracking algorithm of Chapter 5 is applied to find the LCS in two geophysical vortices: a warm-core ring in the Gulf of Mexico and a numerical simulation of hurricane Rita (2005). Vortical structures such as these are some of the most impactful geophysical flows in terms of weather and climate. Hurricanes and tornadoes are the most prominent examples of such vortex flows, both have the potential for massive impacts on human activities and infrastructure. The importance of understanding these extreme weather events has lead to decades of study in an effort to better model, predict, and mitigate their impacts. However, for hurricanes in particular, there is still a lack of knowledge about the processes that govern energy and momentum exchange in the boundary layer [89]. The importance of the boundary layer physics for hurricane intensity and development is well known [3]. However, the energy influx from the ocean is a complex process depending on many factors such as wind speed, wave height and direction, fetch length, sea surface and air temperatures, and the specific transport structure of the ocean and atmosphere.

Further complicating hurricane dynamics are the coupled interactions with the underlying ocean. In particular, hurricane intensification is closely related to heat content of the upper layers of the ocean, which depends on the depth of the thermocline and the warm surface water [87]. Intensification often coincides with a hurricane passing over a warm current or a warm-core ring (WCR) where the warmer surface waters extend to a greater depth than in the surrounding ocean. In 2005, hurricane Katrina intensified from a category 3 to a category 5 storm within 12 hours as it passed over the warm waters of the loop current and a WCR in the Gulf of Mexico [2, 83]. WCRs in the Gulf of Mexico periodically separate from the loop current in the eastern gulf [90] and migrate westward under the planetary beta effect [41] to eventually dissipate near the Mexico-Texas border. They play an important role in the transport and mixing of the Gulf [93].

Despite the obvious differences in hurricanes and ocean eddies (e.g. Reynolds number, destructive force, time scales involved, etc.), both share some characteristic features. Large parts of both flows are quasi-geostrophic where flow speeds are determined by a balance between Coriolis and pressure gradient forces. Additionally, both flows have complex boundary layer interactions at the ocean-atmosphere interface where frictional effects are important and energy exchange takes place. Finally, both have strongly hyperbolic regions in the flow where the flow separates from the boundary, giving rise to highly complex LCS corresponding to the intense stretching and mixing behavior of these regions.

In this chapter, I use the three-dimensional, grid-based ridge tracking algorithm to study the LCS in the boundary layers of hurricanes and ocean eddies. I have observed a characteristic *checkerboard pattern* formed by the attracting and repelling LCS in hurricane boundary layers and the mixed layer of warm-core rings in the Gulf of Mexico. Evidence of a similar checkerboard pattern exists in the LCS reported by other researchers [77], but these structures were not directly addressed there or in other previous work. In the following sections I propose a simple mechanism that generates checkerboard LCS patterns and present the checkerboard LCS in two simple analytical flows with the necessary criteria for checkerboard LCS. Similar structures are observed in the LCS in a warm-core ring in the Gulf of Mexico and in a simulation of Hurricane Rita. Finally, I discuss the implications of this checkerboard pattern for transport and mixing, in particular, the possibility for greatly enhanced diffusive mixing due to the creation of smaller length scales in regions influenced by checkerboard LCS.

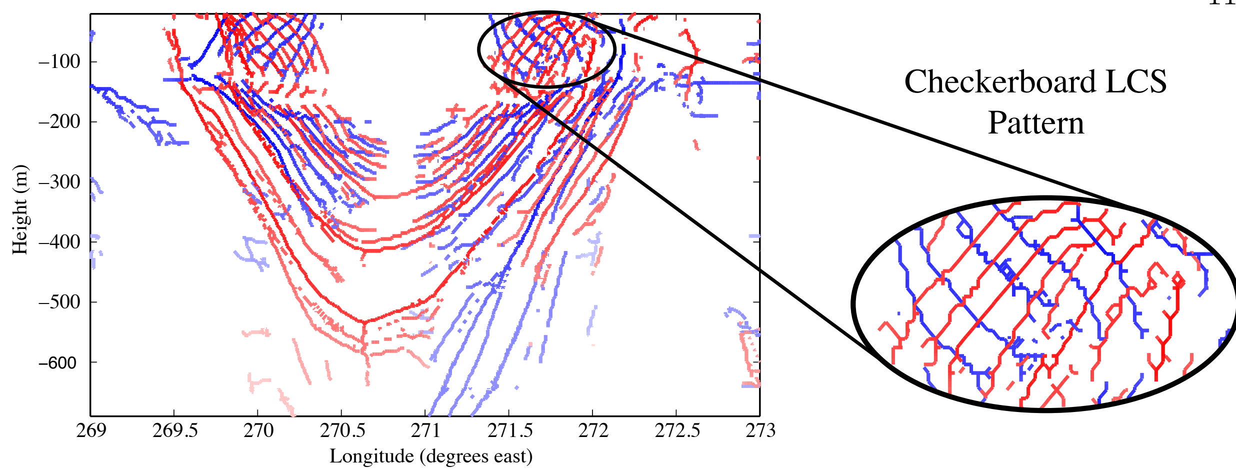


Figure 6.1: A vertical cross section of the LCS in a warm-core ring in the Gulf of Mexico as computed from the publicly available ECCO2 dataset. The inset shows the characteristic "checkerboard" pattern formed by the interaction of the attracting (red) and repelling (blue) LCS.

6.2 Checkerboard patterns in LCS

While investigating the transport structure of hurricanes and ocean eddies, one of the most striking features that consistently appears in the LCS of these geophysical vortices is a "checkerboard" pattern formed by the interaction of the attracting and repelling LCS in the boundary layer. This is the most complicated region of the flow and there is large shear and an expectation of chaotic mixing behavior. However, I have observed these checkerboard features in both warm-core rings in the Gulf of Mexico in the publicly available ECCO2 dataset (available at `http://ecco2.org/`) and a simulation of Hurricane Rita from researchers at the National Center for Atmospheric Research (NCAR). Additionally, I found evidence of similar structures in the previously published LCS of an axisymmetric hurricane simulation, but that publication did not note or address these structures [77]. Each of these flows has a separation region where flow separates from the boundary (updrafts in the hurricane, downwelling in the eddy) near the region of highest azimuthal velocity which is coupled with shear in the azimuthal velocity and periodic perturbations to produce the checkerboard pattern.

A typical example of the LCS checkerboard pattern is shown in Figure 6.1. This figure shows a vertical slice through the three-dimensional LCS of a WCR in the Gulf of Mexico. The pattern is characterized by a grid of quadrilateral regions with alternating sides defined by attracting and repelling LCS. Since this is only a single, two-dimensional slice of the three-dimensional LCS, it does not reveal the full structure of these lobes. In fact, these quadrilateral lobes form tube-like structures that wrap around the eddy in the azimuthal direction.

The angle between the LCS in the checkerboard pattern gives an indication of the nature of the stretching caused by the flow in this region. Fluid parcels are drawn to the attracting LCS (red curves in Figure 6.1) and pushed away from the repelling LCS (blue curves in Figure 6.1), creating a hyperbolic stretching effect. Since this pattern occupies a fairly large region (over 25 km in the radial direction), I postulate that the flow in this entire region experiences similar hyperbolic stretching, but perturbations to the system create surfaces of maximal attraction or repulsion that are revealed by the LCS.

6.2.1 Linear saddle

To investigate the dynamics responsible for the checkerboard pattern LCS, I begin by analyzing simple analytical velocity fields designed to mimic some of the key aspects of the hurricane and eddy flows in these checkerboard regions. I will use a three-dimensional Cartesian velocity field that is periodic in the z-direction. This is analogous to a cylindrical coordinate system centered on a vortex (where the θ-direction is periodic).

The first flow is a simple linear saddle and an z-velocity that is a linear function of x and y:

$$\begin{aligned} u &= -x, \\ v &= y, \\ w &= A + B(x + y). \end{aligned} \tag{6.1}$$

The x-y velocity components of this flow are shown in Figure 6.2. This linear system may be integrated directly to find trajectories

$$\begin{aligned} x(t) &= x_0 e^{-t}, \\ y(t) &= y_0 e^{t}, \\ z(t) &= At + Bx_0(1 - e^{-t}) + By_0(e^{t} - 1) + z_0. \end{aligned} \tag{6.2}$$

Taking the gradient of the flow map defined by Equation 6.2 gives a result that is independent of (x_0, y_0, z_0) and therefore admits no FTLE ridges and no LCS. Put another way, the stretching is uniform over the entire domain in this example. This linear saddle may be an approximation to the velocity field near a separation point in a realistic flows, however, to create any interesting dynamics the system must be perturbed in a meaningful way.

The hyperbolic stretching, combined with the shear in the z-velocity determines the effect of perturbations to the system. In forward time, fluid parcels in this system are compressed in the x-direction and stretched in the y-direction while being sheared by the z-velocity. This z-velocity accentuates the hyperbolic stretching by creating velocity differences proportional to the x-y distance between particles. As fluid parcels are stretched, they are also rotated by the shear so that initial displacements in the y-direction become large displacements in the z-direction while initial displacements in the x- and z-directions remain relatively small.

Once a fluid parcel has been stretched in such a way, perturbations with limited z extent will effect parts of the fluid parcel that correspond to a large range of y_0 and z_0 coordinates but only a very small range of x_0 coordinates. The above flow (Equation 6.1) is linear and generates no LCS. I perturb the system to be

$$\begin{aligned} u &= -x(1 + \varepsilon \sin(z)), \\ v &= y(1 + \varepsilon \sin(z)), \\ w &= A + B(x + y), \end{aligned} \tag{6.3}$$

and generate localized regions of higher and lower stretching. This leads to enhanced stretching about certain values of x_0 and repelling LCS that align with $x_0 =$ constant. In backward time, the hyperbolic stretching has the opposite effect in the x- and y-directions while the shear in the z-velocity still acts to accentuate the hyperbolic stretching. In this case, perturbations create attracting LCS aligned with $y_0 =$ constant. Finally, if a series of perturbations is encountered in the z-direction, this will create a series of parallel LCS and result in a checkerboard pattern. This is the case in vortical flows where the azimuthal direction is analogous to the z-direction and perturbations will be encountered periodically as fluid recirculates around the vortex multiple times.

Here, the perturbations act on the x- and y-velocity and depend only on the z-position. This is motivated by the physical systems of interest. In vortical flows, the largest velocity component is in the azimuthal direction, but many of the interesting dynamics are determined by the axial and radial flow components. Perturbations to these velocity terms may arise from non-circular vortices, asymmetric forcing or varying terrain or bathymetry. The periodic perturbation in z keeps the flow incompressible and creates variations in the stretching experienced by different regions of the flow since the z-velocity is dependent on the x- and y-positions. The magnitude of ε does not qualitatively effect the results, but only changes the prominence of the resulting ridges in the FTLE field.

Figure 6.2 shows the velocity and LCS resulting from Equation 6.3 with $A = 10$, $B = 1$, $\varepsilon = 1$ and $T = 4$. Although the perturbation to the system is only dependent on z-position, it creates LCS that are aligned with the principle directions of stretching in the original linear saddle.

This simplified example has many unphysical aspects, but captures the key components that lead to the checkerboard LCS pattern in several geophysical vortices:

- hyperbolic stretching in the r-z plane
- azimuthal flow with large shear
- perturbations to create LCS with locally maximum stretching

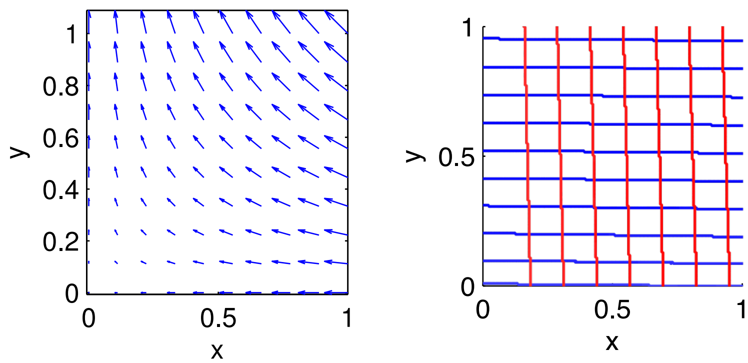


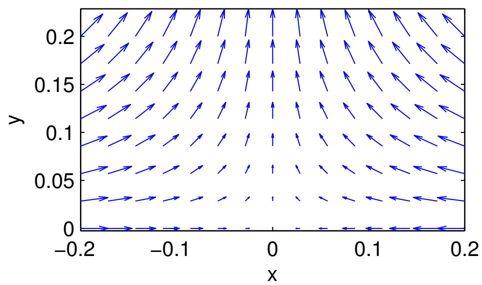
Figure 6.2: Left: the x-y components (at $z = 0$) of the linear saddle velocity field given by Equation 6.3. Right: the LCS resulting from Equation 6.3. The perturbation is periodic in the z-direction, but causes nearly perpendicular LCS aligned in the x- and y-directions. The red vertical curves are attracting LCS and the blue horizontal curves are repelling LCS.

In this example, the perturbation was a periodic change to the x- and y-velocity based on the z-position, but it could also be due to asymmetry or time dependence in more realistic flows.

6.2.2 Nonlinear saddle

Although the velocity field used in the previous section is the simplest velocity that demonstrates all the characteristics necessary for formation of the checkerboard LCS pattern, it is quite unrealistic in two obvious ways: the velocity is unbounded for large values of x and y and the hyperbolic component of the flow stretching is exactly constant. This may be a valid approximation near hyperbolic points, but cannot be valid over large flow domains due to the unbounded velocity. A slightly more complicated nonlinear saddle flow is given by

$$\begin{aligned} u &= -A\sin(\pi x)\cos(\pi y)(1 + 10\sin(\pi z)), \\ v &= A\cos(\pi x)\sin(\pi y)(1 + 10\sin(\pi z)), \\ w &= B(x + y + 1), \end{aligned} \tag{6.4}$$



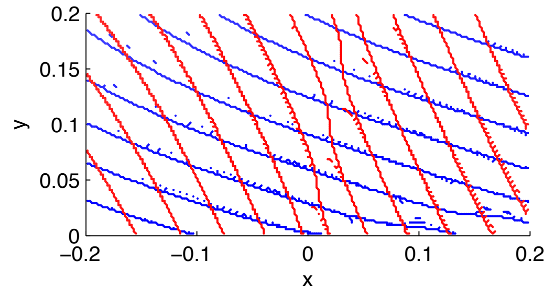


Figure 6.3: Left: the x-y velocity at $z = 0$ for the nonlinear saddle flow of Eq. 6.4. Right: the LCS resulting from Eq. 6.4. The red, more vertical curves are attracting LCS and the blue horizontal curves are repelling LCS.

and has finite velocity u-v velocity and non-uniform hyperbolic stretching. Parameters are set to $A = \pi/10$ and $B = 35$ and an integration time of $T = 1$. For this integration time, the $x - y$ coordinates of particle trajectories remain small and the w velocity is bounded. The flow is incompressible and the x-y velocity gives a hyperbolic point at the origin with stretching in the y-direction and compression in the x-direction. The dynamics of fluid parcels are very similar to those of the linear saddle above. The hyperbolic stretching near the origin causes the shear in the z-velocity to preferentially stretch fluid parcels initially aligned along the x- or y-direction resulting in the nearly perpendicular LCS shown in Figure 6.3. The checkerboard LCS structure seen in Figure 6.3 closely resembles the structures seen in the WCRs (Figure 6.1), indicating the nonlinear nature of the flow in physically relevant situations.

In the previous section involving the linear saddle, the hyperbolic stretching was constant throughout the domain so any perturbation to this stretching might be expected to create local maxima in the FTLE field. However, in the nonlinear saddle of Equation 6.4, the hyperbolic stretching is larger near the origin than at the edges of the domain. This does not disrupt the dynamics that create the checkerboard pattern.

6.3 Checkerboard LCS in geophysical vortex flows

The previous section presented two analytical flows that produce checkerboard LCS. The same flow characteristics that cause checkerboard LCS in those examples also appear in the boundary layers of geophysical vortices. These regions have large shear (due to friction at the boundary) and hyperbolic stretching due to separation from the boundary. Below, the LCS for two such geophysical vortices are presented: WCRs in the Gulf of Mexico and a simulation of hurricane Rita.

6.3.1 Warm-core rings in the Gulf of Mexico

Warm-core rings are one of the most striking and dynamically important flow features in the Gulf of Mexico [8, 24, 25]. In the eastern Gulf, the Loop Current enters through the Yucatan straits and loops north into the Gulf before exiting through the Florida Straits and becoming the Gulf Stream. This current carries warm, salty water from the Caribbean through the eastern Gulf and out into the Atlantic. Occasionally, the Loop Current doubles back on itself and pinches off a closed ring of this warm, salty water: a warm-core ring (WCR). Ring pinch off occurs at irregular intervals typically ranging from 7-13 months [90]. The rings have an elevated sea surface height and rotate anti-cyclonically. They maintain coherence for a year or more as they drift westward under the influence of the planetary beta effect before eventually dissipating in the western gulf near the border of Texas and Mexico [41].

These warm rings are typically hundreds of kilometers in diameter and contain massive amounts of warm, salty water [53]. Additionally, they generate some of the strongest currents seen in the central Gulf of Mexico with flow speeds around 1 m/s [9]. The dynamics of WCRs have important effects on the heat and salt balance in the Gulf of Mexico and therefore indirectly impact the climate for much of the eastern United States [24]. More directly, these WCRs have a depressed thermocline and carry a much larger depth of warm water

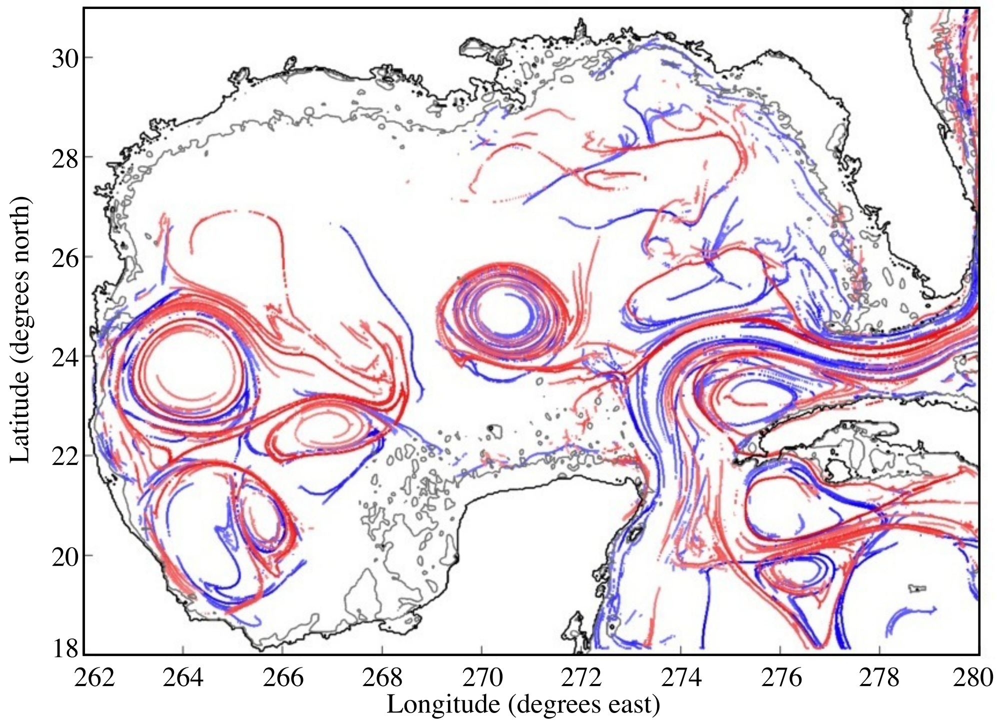


Figure 6.4: Surface LCS for the Gulf of Mexico, attracting LCS are red, repelling are blue. A large warm-core ring is located in the center of the gulf and an older ring is in the western gulf.

than the surrounding ocean [87]. This results in an increased heat content for columns of water in a warm ring compared to those outside. This has very important effects on hurricane intensity. Since there are much greater heat reserves in a WCR, passing hurricanes have been known to rapidly intensify, leading to more dangerous storms [2, 40, 83].

While studying WCRs and other vortices, it is convenient to adopt a cylindrical coordinate system, (r, θ, z), rather than the Cartesian coordinates used above. The r-z plane in the WCRs is analogous to the x-y plane in the above flows and the azimuthal (θ) direction is periodic and analogous to the z direction above. In certain regions of the flow, the WCRs have the characteristics of the linear and nonlinear saddle flows discussed above. Although much of the flow near these WCRs is in nearly geostrophic balance, friction in the mixed layer has the effect of disrupting the geostrophic balance and creating radial outflow near the ocean surface due to the elevated pressure at the ring center. This flow recirculates downward at the outer margins of the ring, creating a hyperbolic separation region where the outflow turns downward. In this same region, there is a large azimuthal velocity component due to the vortex circulation that is analogous to the z-velocity above.

A vertical cross section of the LCS in one of these WCRs can be seen above in Figure 6.1. This particular ring has a diameter of over 300 km and a depth of almost 600 m. It is closed at the bottom by an LCS surface that wraps around the ring and extends upward to the ocean surface. The depth of the WCR is well determined by the LCS that close the bottom of the ring. In fact, this depth is closely related to the geostrophic balance and depends primarily on the density of the eddy water and the surrounding ocean, the sea surface height anomaly and the depth of the mixed layer. The LCS allow for an unambiguous definition of the WCR depth and the verification of models for this depth that can be used to estimate the volume and transport properties of the ring. Such a model is presented in Appendix C.

The WCR core consists of relatively placid flow that is in near-solid body rotation and does not contain any LCS surfaces. In the lower depths of the ring, there are layers of attracting and repelling LCS that are nearly parallel to one another. This is characteristic of shear flows, where separation occurs along regions of maximum shear in the same way for both forward and backward time advection. However, a dramatic change occurs in the LCS at a depth of around 150 m, the bottom of the mixed layer. In the mixed layer, the checkerboard pattern LCS appear. Although large shear exists in this region, the importance of friction effects and the separation of flow from the ocean surface create the hyperbolic stretching that is responsible for the transverse intersection of the LCS that form the checkerboard pattern.

Colored drifter particles were placed in some of the closed "lobes" created by the checkerboard LCS to further investigate the transport in this region. The initial configuration of the drifters can be seen in Figure 6.5. Advecting the particles backwards in time reveals the trajectory histories. The initial configuration (Figure 6.5) is for February 1, 2010. Figure 6.6 shows the particle positions two and four weeks before this date while Figure 6.7 shows the

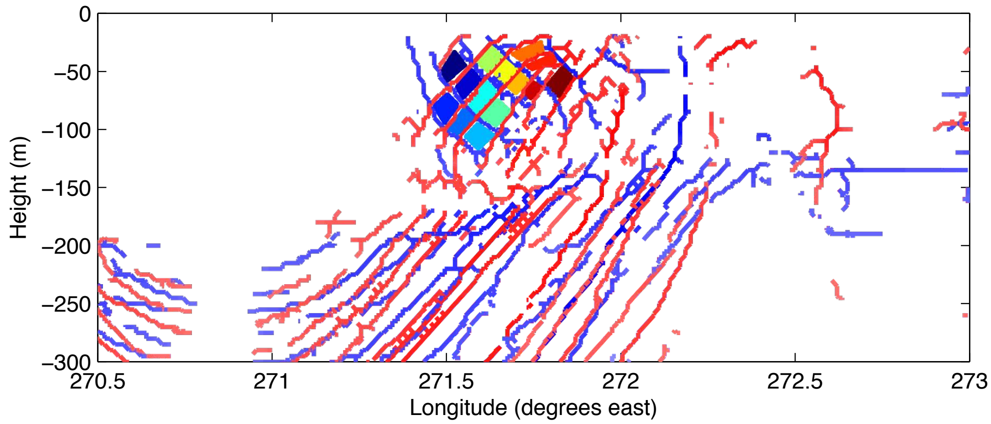


Figure 6.5: Passive drifters placed in a warm-core ring. The red and blue curves are attracting and repelling LCS and the drifters are colored based on their initial lobe position.

positions two and four weeks after. Note that in backward time (Figure 6.6), the drifter groups divided by the attracting (red) LCS in Figure 6.5 separate most rapidly from one another while in forward time (Figure 6.7) the groups separated by repelling LCS separate from one another. Additionally, the drifter groups that are initially released at the same azimuthal position are quickly stretched and rotated by the shear in the azimuthal velocity until they are stretched around the entire perimeter of the warm ring.

6.3.2 Hurricane Rita simulation

Hurricanes are one of the most destructive natural phenomena on the planet. The 2005 season alone caused over $100 billion in property damages in the United States [2]. One of the remaining difficulties in hurricane forecasting is intensity prediction. Hurricanes often undergo rapid intensification, especially when passing over warm water currents or eddies [87]. Additionally, boundary layer processes are critical to the hurricane's development and intensity, but are still not fully understood [3].

Here, the LCS for a hurricane in the Gulf of Mexico are presented. The data used is from a weather research and forecasting (WRF) hurricane simulation of hurricane Rita (2005) [96]. This high resolution model has a finest grid with 1.3 km resolution in the

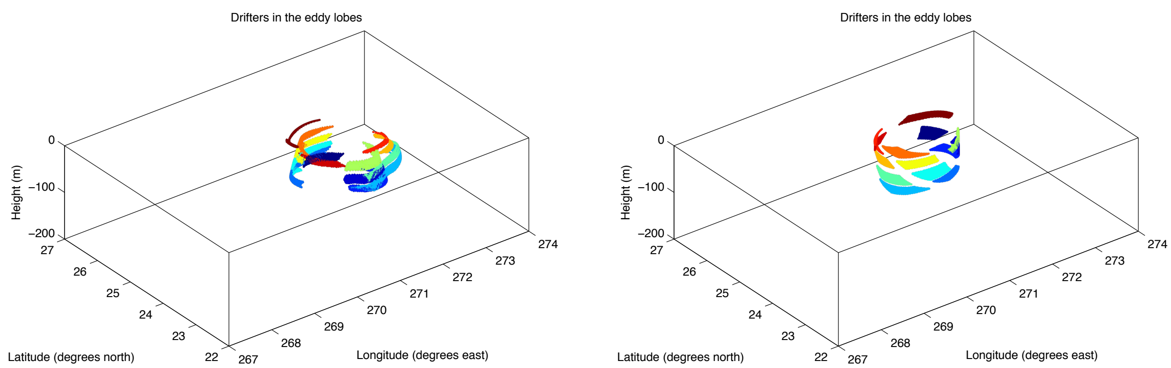


Figure 6.6: Previous positions of the drifters shown in Figure 6.5. The drifters are colored based on their initial configuration. Left: $t = -4$ weeks. Right: $t = -2$ weeks.

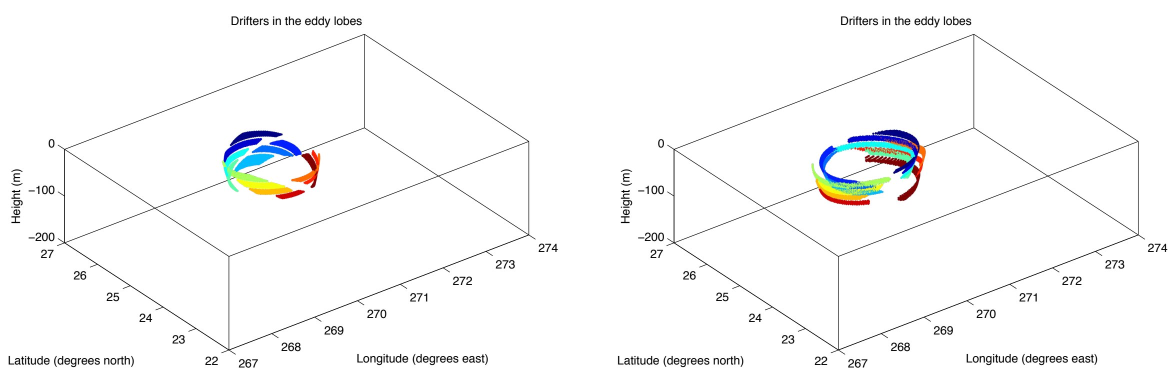


Figure 6.7: Future positions of the drifters shown in Figure 6.5. The drifters are colored based on their initial configuration. Left: $t = 4$ weeks. Right: $t = 2$ weeks.

horizontal direction. An integration time of $T = 1$ hr is used to compute the FTLE values and the three-dimensional ridge tracking algorithm is used to speed computations.

The hurricane has all the criteria listed above to create checkerboard LCS in the boundary layer. Near the hurricane eye, the inflow is turned upward by convection and converging flow at the eyewall, creating hyperbolic stretching in the r-z plane. At the same time, there is a very large velocity in the azimuthal direction with tangential wind speeds of more than 70 m/s, an order of magnitude larger than the radial and vertical velocities. There is also large shear due to the frictional interaction with the sea or land surface.

Figure 6.8 shows a constant latitude slice of the LCS for the hurricane simulation. The top part of the figure show the entire height of the hurricane, revealing the complex boundary layer inflow, a calm eye bounded by LCS in the eyewall, and upper level outflow also bounded by LCS. The bottom part of this figure focuses on the lowest 3000 m of the hurricane, containing the boundary layer. The flow to the west of the eye is dominated by the checkerboard LCS that are the subject of this section. In this region, fluid parcels experience hyperbolic stretching in the r-z plane while being stretched and wrapped around the hurricane in the azimuthal direction in the same way the particles are wrapped around the eddy of Section 6.3.1. This stretching is expected to enhance mixing in the hurricane boundary layer.

The rest of the hurricane dynamics revealed by the LCS merit further study, but one notable feature is the occasional appearance of asymmetries near the low level eyewall. Such an asymmetry is presented in Appendix D where it is found to rapidly eject fluid from the eye into the upper atmosphere.

6.4 Checkerboard LCS and the horseshoe map

Due to the periodic nature of the flows discussed above, in the steady flow case, and if an appropriate reference frame and time period is chosen, these flows exhibit some similarities to the horseshoe map of classical dynamical systems [88]. However, as discussed in this

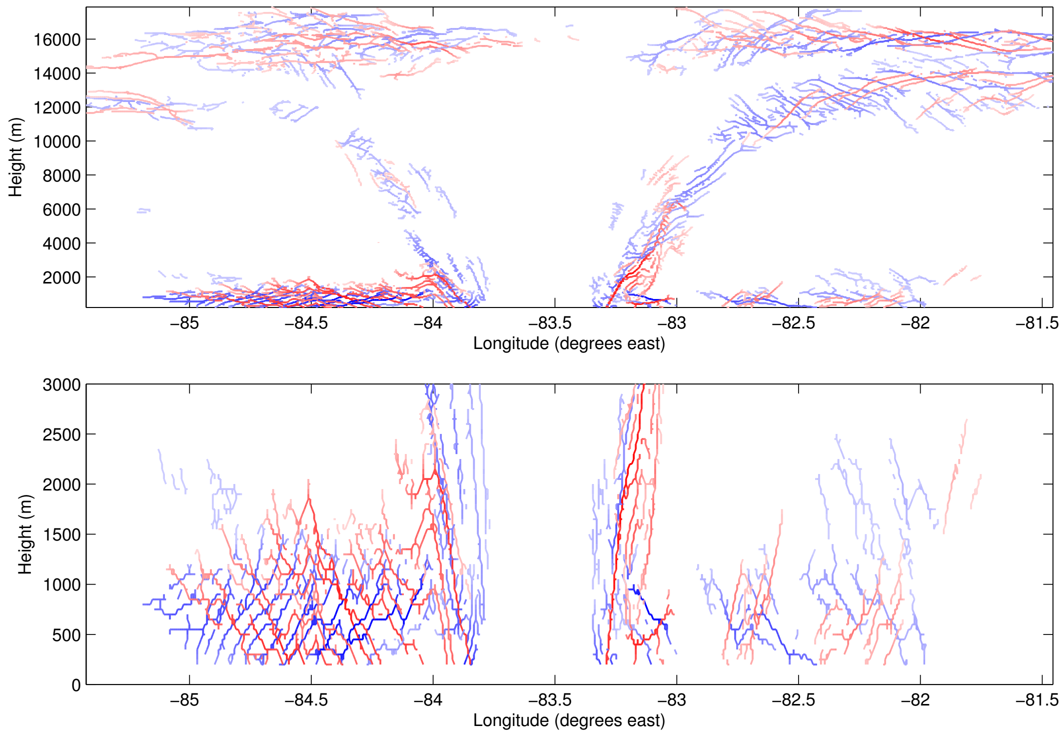


Figure 6.8: A constant latitude slice of the LCS in the hurricane simulation. Red LCS are attracting structures, blue are repelling. The color intensity is proportional to the FTLE magnitude. There are more LCS to the west of the hurricane due to asymmetry in the hurricane wind field (stronger winds in the west) at this time. TOP: the full height of the hurricane showing a complex inflow/boundary layer region, a calm eye and an upper atmosphere outflow region. BOTTOM: A zoomed view of the lower 3000 m of the hurricane containing the boundary layer. The checkerboard LCS pattern dominates the LCS to the west of the eye.

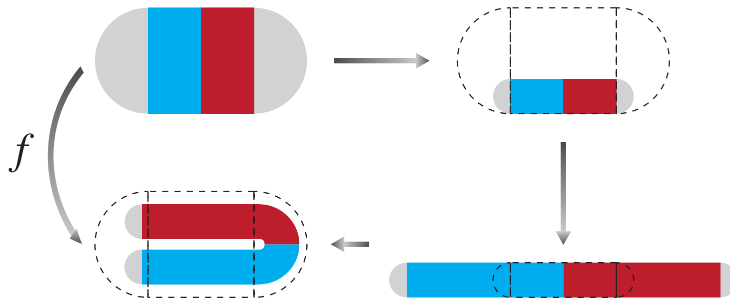


Figure 6.9: The classical horseshoe map. The unit square is linearly compressed in one direction and stretched in the other before being folded back on itself.

section, there are several key distinctions. The classical horseshoe map is a diffeomorphism of the unit square as shown in Figure 6.9 and the stretching and folding seen in the horseshoe map is similar to the types of chaotic advection seen in many flows [70, 71]. The horseshoe map also has an infinite number of periodic orbits as well as orbits of any given period that form invariant sets in the unit square.

At first, the flows discussed here seem to share the key characteristics of the horseshoe map. There is hyperbolic stretching in the flow that creates compression in one direction and expansion in another as well as a type of folding where the flow doubles back on itself due to periodicity. In the horseshoe map, the folding step is typically caused by some process that causes the domain to bend in half. However, folding may also be caused by the geometry of the problem. In the saddle flows above, the WCR and hurricane, folding behavior is created by the periodicity of the domain; in the z-direction for the analytical examples and in the azimuthal direction for the vortical flows. While this is more naturally discussed in terms of fluid wrapping around a vortex than as folding, it is topologically similar to a fold since points on opposite sides of the domain are mapped to positions near one another.

In general though, the examples discussed above do not meet all the criteria to be a horseshoe map. Let us consider the linear saddle of Section 6.2.1 without the periodic perturbation. Let the periodic flow be defined on the domain $\mathbb{R}^2 \times [0, 2\pi)$ and consider the map of particles from time 0 to time τ. In this case, the map is given by

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix}^{i+1} = \begin{pmatrix} e^{-\tau} & 0 & 0 \\ 0 & e^{\tau} & 0 \\ B(1-e^{-\tau}) & B(e^{\tau}-1) & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix}^{i} + \begin{pmatrix} 0 \\ 0 \\ A\tau \end{pmatrix} \tag{6.5}$$

Since this map is linear, the only possible fixed points lie on the line $x = y = 0$. If A is nonzero, τ must be chosen as $2\pi/A$ to take advantage of the periodicity to admit even these fixed points. Additionally, it is trivial to show that for $\tau > 0$ and $\forall x^i, y^i \neq 0$, $|x^{i+1}| < |x^i|$ and $|y^{i+1}| > |y^i|$. Therefore, in contrast to the horseshoe map, there are typically *no* periodic orbits in the checkerboard map. Additionally, the traditional horseshoe map is a contraction on the space shown in Figure 6.9 (the unit square with added "end caps"), but the saddle flows and the WCR and hurricane are all volume preserving (i.e. incompressible) and therefore cannot be contraction mappings. The area preserving horseshoe map may be more similar to the realistic flows presented above.

Despite qualitative similarities (stretching and folding), flows creating the checkerboard LCS do not typically qualify as horseshoe maps, even in the absence of aperiodic time dependence or perturbations. However, the checkerboard LCS are indicative of extreme stretching that can significantly enhance mixing in a flow. The flow map that characterizes the examples seen above is shown in Figure 6.10 and will be referred to as the "checkerboard map". This map is different than the horseshoe map in several ways. Step 1 (possible rotation and shearing in the plane) is not present in the horseshoe map. Step 2 of the checkerboard map combines the first two steps of the horseshoe map. Step 3 (shearing out of the plane) is also absent from the horseshoe map. Finally, step 4 (wrapping around the vortex or periodicity of the domain) is equivalent to the folding step of the horseshoe map. Note that the checkerboard creates stretching in the third dimension via large shear, a step that is not present in the horseshoe map. Additionally, the stretching and folding of the checkerboard map necessarily wraps fluid from the surrounding regions into the initial domain. Since the checkerboard map is not a contraction on any part of the domain (because these flows are volume preserving), it cannot map all points in an initial domain back into the same domain. This is in contrast to the horseshoe map, which is a contraction on the the initial domain.

There are many ways to define mixing. In the context of LCS, mixing typically refers to topological mixing. In the context of fluid dynamics, topological mixing occurs when fluid from one region moves into another region. Often, this type of mixing is revealed by lobe dynamics techniques that show the fluid exchanged between regions through the interaction of stable and unstable manifolds of hyperbolic fixed points [76]. Analogous behavior has been shown in LCS, most notably in the exchange of fluid between a vortex ring and the surrounding fluid [84]. In such topological mixing situations, distinct lobes in the domain are bounded by a single stable manifold and a single unstable manifold and act to entrain and detrain fluid from a region. In the checkerboard LCS, the checkerboard squares are bounded by four separate LCS, two attracting and two repelling. For this reason, these LCS structures are not expected to reveal lobe based topological mixing. However, the stretching and folding of the checkerboard map may lead to other types of mixing.

To my knowledge, LCS applications have previously considered only advective transport of either passive drifters or inertial particles and have not considered the possible impacts of diffusive mixing. In many situations the diffusion lengths are very small on the time scales of interest. However, diffusion may play a key role when advective transport is small, large time scales are involved, or small length scales are generated. Diffusion is particularly important when large stretching exists in the flow (creating smaller length scales in the direction normal to the stretching) or when long times are considered. The diffusion length, a measure of the distance over which diffusion is effective, is defined as $L_D = 2\sqrt{Dt}$ where D is the diffusion coefficient and t is the time scale involved.

A typical WCR may exist for around 12 months and has a radius of around 150 km. Two important (in terms of their impact on ocean physics) diffusive quantities are salinity and heat content which have diffusion coefficients of $D_s \approx 10^{-9}$ m^2/s and $D_h \approx 10^{-7}$ m^2/s respectively [42, 75]. This gives diffusion lengths of around 1 m for salt and around 10 m

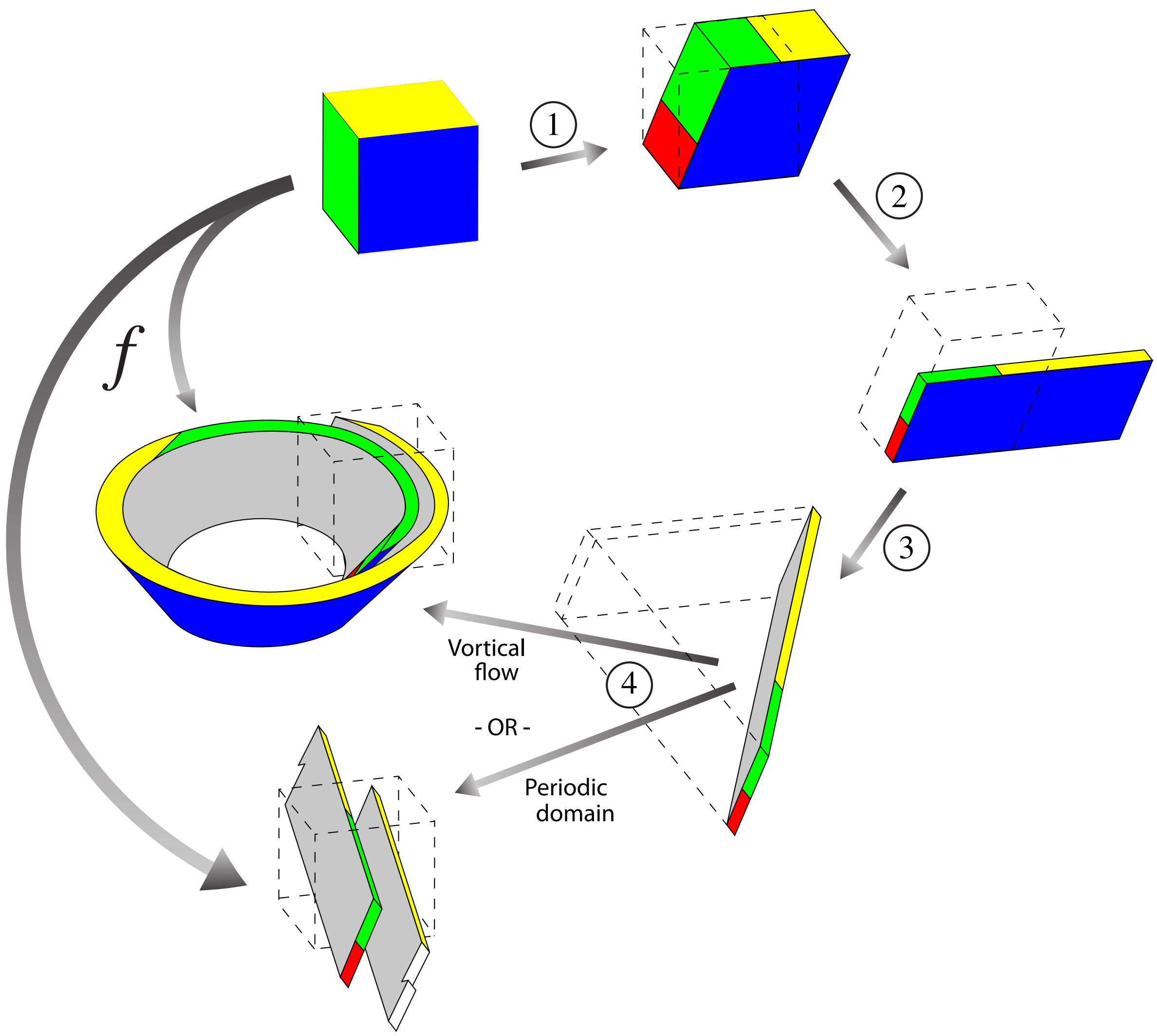
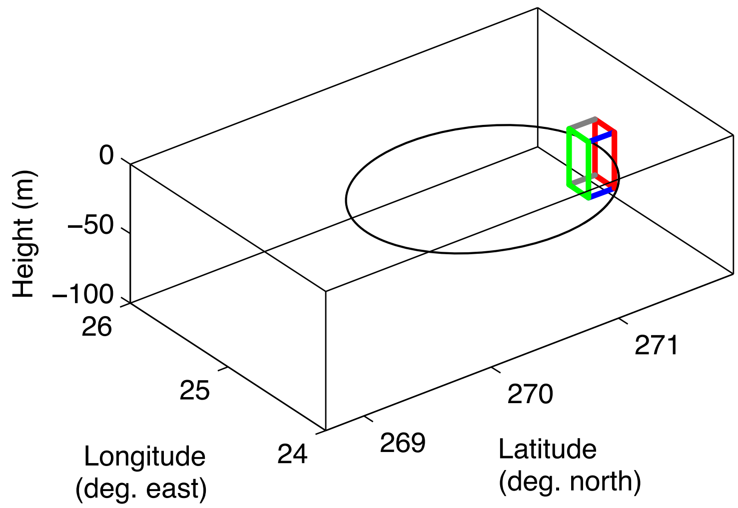


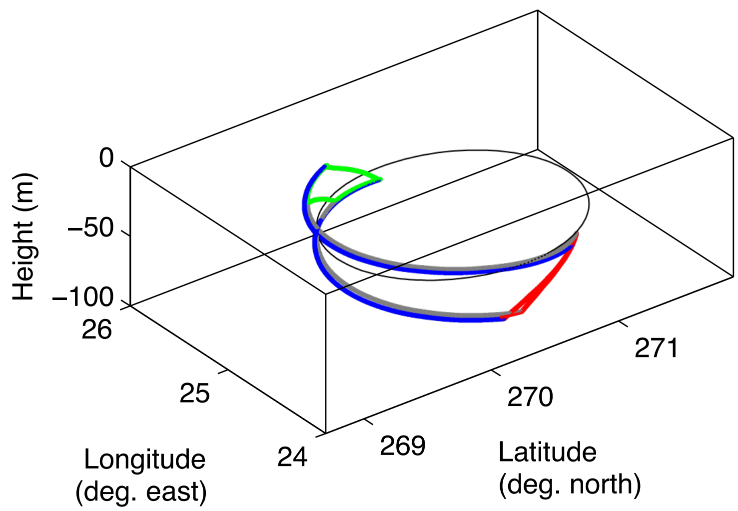
Figure 6.10: The flow map for the checkerboard flows, note the similarities with Figure 6.11. The flow map can be broken into 4 steps: (1) Possible shear and rotation in the plane of the blue (front) face, (2) Stretching in at least one dimension and compression in at least one dimension, (3) Shear out of the plane of the blue (front) face, (4) Wrapping (in the case of vortical flows) or periodicity (in the case of the periodic domain used in Section 6.2).

for heat over the 12 month eddy life. In the context of the eddy size, these diffusion lengths are insignificant, but when the stretching and compression caused by the flow in the region of the checkerboard LCS is considered, these effects may become important. To quantify this effect, a box shaped configuration of drifters is placed into the checkerboard LCS region of the WCR and advected with the flow using the data from the ECCO2 simulation. The resulting drifter motions are used to estimate the stretching over this region and are shown in Figure 6.11. Note that this is *not* an approximation of the stretching over an infinitesimal distance. Rather, it this is the stretching averaged over a large region that contains the checkerboard LCS.

The length and thickness of the box is approximated (the depth range stays fairly constant) and the results are shown in Figure 6.12. This figure shows that the length of the box grows linearly while the thickness of the box shrinks exponentially. The shrinking thickness results in initially disparate regions coming very close to each other, creating smaller length scales in the flow. Additionally, the initial length of the box was over 22 km. This is not a local effect but acts over large distances and a time period of almost a month. Using a least squares fit on the box thickness plot in Figure 6.12 gives the thickness $\delta = \delta_0 \exp(-0.11t)$. Using this rate of thickness decrease, it is simple to estimate how length scales may be compressed over time. For example, in this 25 day period, a 160 m length may be compressed to only 10 m, the heat diffusion length scale. Furthermore, applying this compression rate over 90 days results in length scales of 210 km being compressed to 10 m (the heat diffusion length) or 21 km being compressed to 1 m (the salt diffusion length). The exponential compression of the box width results in large compression ratios for even fractions of the eddy life cycle, generating smaller length scales that potentially enable diffusion of heat and salt to become an important effect.

Finally, note that in Figure 6.11 it is the blue and grey curves that approach each other as the box width is compressed. These curves were initially separated in the azimuthal direction, meaning that length scales in the azimuthal direction are potentially compressed





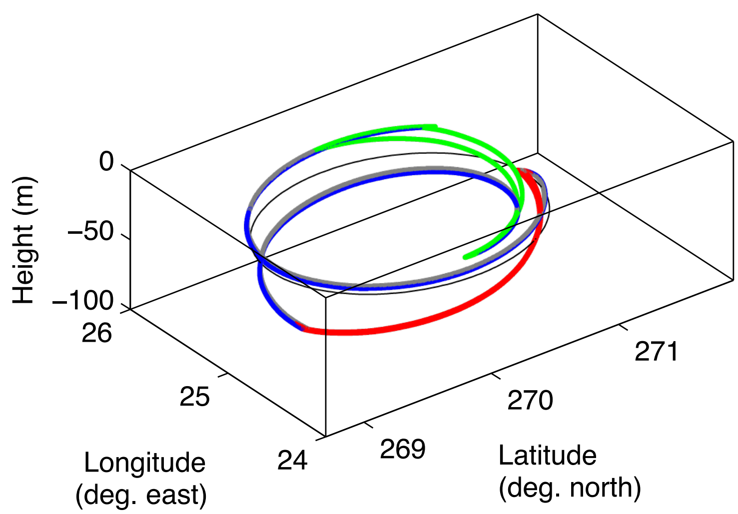


Figure 6.11: A box of drifters placed in the checkerboard LCS of an ocean eddy. The drifter positions are shown at times of zero, 12.5, and 25 days. The black circle shows the position of the warm-core ring.

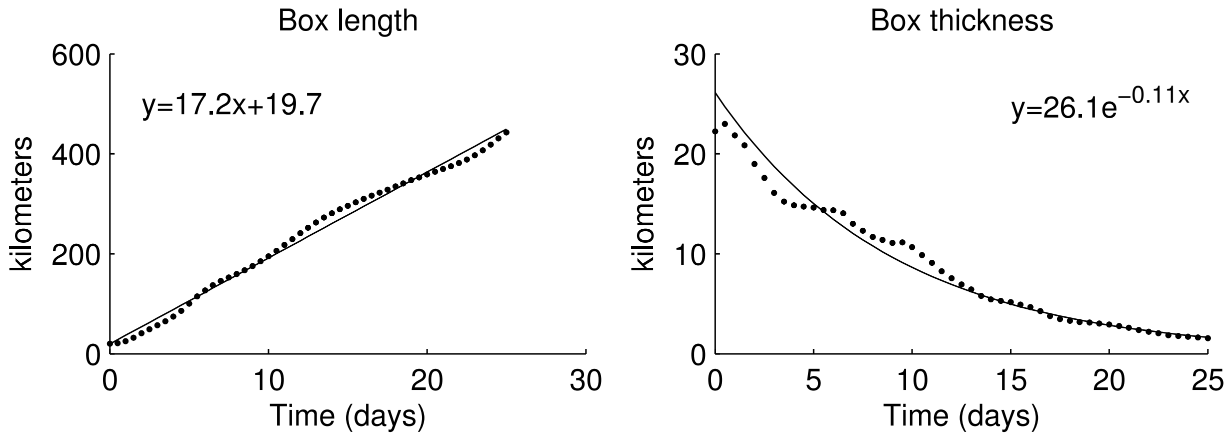


Figure 6.12: The deformed length and thickness of the the drifter box shown in Figure 6.11. The length is the direction that is stretched around the ring and the thickness is the radial thickness of the box. The length grows linearly at a rate of about 17.2 km day^{-1} while the thickness decays exponentially. A least squares best fit was used to determine the regression curves.

to a level that enables diffusion to play a role. Given this, one would expect the fluid properties of the warm core ring to be homogenized along the azimuthal direction due to this enhanced diffusion while gradients in temperature and salinity persist in the radial and depth directions. In fact, this is precisely what is observed in the warm core rings; the rings are largely axisymmetric with temperature and salinity that depend on the radius and depth.

6.5 Applications conclusions

A checkerboard map was identified in several simple dynamical systems meeting specific criteria. The velocity field of these systems must create hyperbolic stretching in two directions and contain large shear in the third dimension. If the hyperbolic stretching is in the x-y plane, shear in the z-velocity acts to accentuate the hyperbolic stretching. Fluid parcels in such a flow become stretched out over a large range of z-values. Then, perturbations to the x-y velocity appear as Lagrangian coherent structures (LCS) aligned with the principle directions of the hyperbolic stretching, creating the checkerboard pattern in the LCS.

The simplest demonstration of the these characteristics is an analytically defined linear saddle in a shear flow with a periodic perturbation. This flow has several unphysical aspects including unbounded velocity and a constant FTLE field that admits no LCS in the absence of perturbations. However, the LCS produced by the perturbed system form a very clear checkerboard pattern as seen in Figure 6.2.

The nonlinear saddle of Section 6.2.2 presents a more realistic situation. The FTLE field is not constant even in the absence of perturbations and the velocity is bounded. In this case, the resulting LCS shown in Figure 6.3 are slightly deformed from the perpendicular alignment of the linear saddle, but the checkerboard pattern is very clear. The nonlinearity of the system produces a checkerboard pattern that is qualitatively similar to that seen in more realistic flows.

The motivation for these analytical examples is the observation of the checkerboard pattern in the boundary layer of two geophysical vortices. Warm-core rings (WCRs) in the Gulf of Mexico show a very clear checkerboard pattern in the boundary layer LCS (Figure 6.1). Passive particles placed in this flow are stretched and wrapped around the eddy, generating smaller length scales in the azimuthal direction and potentially enhancing diffusive effects. Similar behavior is seen in the boundary layer of the hurricane. Although the flow structure of the hurricane is much more complicated and turbulent, the checkerboard pattern manifests in the hurricane inflow region near the ocean surface.

Finally, I have shown that the flow in the checkerboard region of the WCR creates exponential compression over long time scales and large distances by studying the behavior of a box-shaped fluid parcel. Particles in this example were set adrift for 25 days over an initial domain spanning 22 km. The domain was eventually stretched around the WCR to a length of over 400 km. The stretching that typifies this flow must be balance by compression in another direction and the box experiences exponential compression in the radial thickness to a final thickness of just 1.5 km. This exponential compression acts to bring entire fluid regions together even over large distances. In principle, this creates the potential for diffusive mixing to play an important role in the heat and salt distribution of the WCR, despite the very small diffusion coefficients of these properties.

Chapter 7

Conclusions

This dissertation set out to support the thesis that Lagrangian coherent structures provide important insight and opportunities for analysis of realistic, three-dimensional fluid flows and a ridge tracking algorithm can accurately compute Lagrangian coherent structures in complex flow while improving the computational order and simplifying LCS visualization and analysis. The LCS for swimming jellyfish, ocean currents, a hurricane, and various analytically defined flows including the time-dependent double gyre have all been presented above. Two closely related ridge tracking algorithms have also been developed, a gridless algorithm that uses ridge aligned local coordinates and a grid based algorithm that relies on a predetermined grid.

The LCS for the swimming jellyfish revealed several new structures in the flow as well as confirmed previously discovered behavior while offering further qualitative understanding of the transport behavior that is important to jellyfish swimming and feeding. Two types of jellyfish were studied, a prolate paddling jellyfish of the *Aequorea victoria* species and a jetting jellyfish of the *Sarsia tubulosa* species. These two types of swimming produce very different flow structures as observed in the LCS. The jetting jellyfish produces a strong vortex along the axis of symmetry that quickly moves away from the jellyfish. The small Strouhal number of $St = 0.1$ reflects the rapid rate of vortex separation. *S. tubulosa* retracts its tentacles while swimming and feeds only while drifting passively. Additionally, the LCS computations revealed new structures within the bell of *S. tubulosa* during swimming. A vortex with opposite rotation from the ejected vortex forms in the subumbrellar cavity during each jetting cycle. It is likely that this vortex has not been previously observed due to the difficulty of imaging this cavity during swimming, but the numerical simulations used for this study suffer no such constraints.

The paddling jellyfish, *A. victoria*, uses a very different swimming stroke. Paddling or rowing swimmers shed vortices from the bell margins and swim with a higher Strouhal number of about $St = 1.1$. These vortices are less energetic and move slowly away from the jellyfish while lingering in the tentacle region. *A. victoria* and other paddling jellyfish are known to feed while swimming and this slow vortex motion allows the jelly to capture prey that may have been entrained into the vortex. Additionally, an LCS lobe forms along the outer bell surface and draws fluid into the forming vortex during each swimming stroke. This lobe is likely to contain small prey that have impacted the jellyfish's outer bell surface during swimming and provides an excellent feeding opportunity.

Although the results of the jellyfish LCS study provided new insights into the flow behavior for these swimmers, the required computations were quite intensive. To analyze three-dimensional flow problems, more efficient algorithms are needed. A ridge tracking algorithm was developed to improve the computational efficiency of finding LCS. A gridless algorithm was first developed to detect and track the LCS ridges through space. In two-dimensional flows, the LCS are one-dimensional curves and can be effectively and accurately computed with the gridless algorithm, provided speed ups of up to 80 times. However, three-dimensional flows require generating a surface mesh to represent the two-dimensional LCS surfaces. Generating and checking this surface mesh without the use of a fixed grid proved to be computationally expensive, resulting in no net speed up for the gridless ridge tracking algorithm in three-dimensional test cases.

To overcome the limitations of expensive surface meshing, a grid-based ridge tracking algorithm was introduced. Technically, this algorithm tracks surfaces that are local maxima in the grid directions. Such points are called coordinate local maxima and are generally not the same as ridge points. However, given a sufficiently well defined ridge, the error caused by this inaccuracy is typically small. A mathematical proof was given that the error is $\mathcal{O}\left(\|\mathbf{D}F\| / |\lambda_1|\right)$ where $\mathbf{D}F$ is the gradient along the ridge and λ_1 is the second derivative normal to the ridge. In many cases this error is smaller than typical grid spacings that can be used to compute the LCS and for the main LCS ridge in the double gyre flow the error bound is as small as 10^{-14} and is typically less than 10^{-6}.

To overcome the mesh generation difficulties of the gridless algorithm, the grid-based algorithm uses a lookup table to generate surface meshes very efficiently. Each cubic element in the domain is asigned an 8-bit number based on whether each of the eight vertices is determined to be a ridge point. This 8-bit number is then immediately translated to the appropriate one of 256 possible surface triangulations. Although there is some ambiguity in the correct surface triangulation for some cases, this has not been observed to cause difficulty in interpreting the final results.

While the ridge tracking algorithm is shown to be accurate, it is also very efficient. Since the algorithm performs computations only near the ridges in the FTLE field, which are codimension-one manifolds, it reduces the computational order of the algorithm from $\mathcal{O}\left(1/\delta x^n\right)$ to $\mathcal{O}\left(1/\delta x^{n-1}\right)$ for n-dimensional computations. This was verified for several three-dimensional test cases and the algorithm was seen to reduce computations from $\mathcal{O}\left(1/\delta x^{3.0}\right)$ to $\mathcal{O}\left(1/\delta x^{2.1}\right)$.

Finally, the grid-based ridge tracking algorithm was applied to two realistic three-dimensional flows, numerical simulations of the ocean and of hurricane Rita (2005). In the ocean flow, the LCS were computed for a warm-core ring in the Gulf of Mexico. In the surface boundary layer of this ring as well as in the hurricane boundary layer a new LCS structure was consistently observed. These structures were dubbed “checkerboard LCS” due to the transversally intersecting attracting and repelling LCS structures that resemble a checkerboard. These structures appear in regions with hyperbolic stretching as well as large shear in a three-dimensional flow. The checkerboard-like nature of the structures is generated by perturbations to the flow as it wraps around the hurricane or warm ring vortex. This behavior was verified in two simple analytical flows that produced the same checkerboard LCS pattern.

Finally, the behavior of a large region of drifters place in the checkerboard LCS of the warm-core ring was analyzed. This initial box-shaped region of drifters experience large stretching to eventually wrap around the entire ring. The length of the box increased linearly from about 22 km to over 400 km in the course of 25 days. At the same time, the box experienced exponential compression in the radial thickness, decreasing from 22 km to about 1.5 km. This stretching and compressing behavior acts over a very large area and time range and generates small length scales in the direction of compression. When combined with wrapping around the vortex, this flow behavior initially looks similar to the stretching and folding caused by the horseshoe map. However, this map is not a contraction and in general creates no fixed points or periodic orbits such as those caused by the horseshoe map. Nevertheless, the generation of small length scales may act to enhance the impact of diffusive transport in this flow, particularly for heat and salt, leading to homogenization of the flow properties in the azimuthal direction.

The research presented in the previous chapters shows both the usefulness of LCS for analyzing realistic fluid flows and the success of the ridge tracking algorithms. In two-dimensional flows, the gridless algorithm provides excellent speed ups while the grid-based algorithm works well in three-dimensions. Perhaps in the future, a more efficient mesh generation technique can be implemented to improve the three-dimensional gridless algorithm, removing the error associated with the grid-based computations. In the mean time, the gridless algorithm has already been applied to several flows and provided insight into some of the boundary layer structures seen in two-geophysical vortices. Furthermore, the smaller output files and resulting decrease in processing time makes final analysis much faster and easier.

Bibliography

[1] D. Beigie, A. Leonard, and S. Wiggins. Invariant manifold templates for chaotic advection. Chaos, Solitons & Fractals, 4(6):749–868, 1994.

[2] J.L. Beven, L.A. Avila, E.S. Blake, D.P. Brown, J.L. Franklin, R.D. Knabb, R.J. Pasch, J.R. Rhome, and S.R. Stewart. Atlantic hurricane season of 2005. Monthly Weather Review, 136(3):1109–1173, 2008.

[3] M. Bister and K.A. Emanuel. Dissipative heating and hurricane intensity. Meteorology and Atmospheric Physics, 65(3):233–240, 1998.

[4] J. Bonet and J. Peraire. An alternating digital tree (ADT) algorithm for 3D geometric searching and intersection problems. Int. J. for Numerical Methods in Engineering, 31(1):1–17, 1991.

[5] S.L. Brunton and C.W. Rowley. Fast computation of finite-time Lyapunov exponent fields for unsteady flows. Chaos, 20:017503 (12pp.), 2010. doi:10.1063/1.3270044.

[6] S.P. Colin and J.H. Costello. Morphology, swimming performance and propulsive mode of six co-occuring hyromedusae. J. Exp. Biol., 205:427–437, 2002.

[7] S.P. Colin, J.H. Costello, and E. Klos. *In situ* swimming and feeding behavior of eight co-occurring hydromedusae. Mar Ecol Prog Ser, 253:305–309, 2003.

[8] C. Cooper, G.Z. Forristall, and T.M. Joyce. Velocity and hydrographic structure of two Gulf of Mexico warm-core rings. Journal of Geophysical Research, 95(C2):1663–1679, 1990.

[9] P. Cornillon and K.A. Park. Warm core ring velocities inferred from NSCAT. Geophys. Res. Lett, 28(4):575–578, 2001.

[10] J.H. Costello. Foraging mode and energetics of hydrozoan medusae. Sci. Mar., 56:185–191, 1992.

[11] J.H. Costello and S.P. Colin abd J.O. Dabiri. Medusan morphospace: phylogenetic constraints, biomechanical solutions, and ecological consequences. Invertebrate Biology, 127(3):265–290, 2008.

[12] J.H. Costello and S.P. Colin. Morphology, fluid motion and predation by the scyphomedusae *Aurelia aurita*. Mar. Biol., 121:327–334, 1994.

[13] J.H. Costello and S.P. Colin. Flow and feeding by swimming scyphomedusae. Mar. Biol., 124:399–406, 1995.

[14] J.H. Costello and S.P. Colin. Prey resource use by coexistent hydromedusae from Friday Harbor, Washington. Limnol. Oceanogr., 47(4):934–942, 2002.

[15] J.O. Dabiri, S.P. Colin, and J.H. Costello. Fast swimming hydromedusae exploit velar kinematics to form an optimal vortex wake. J. Exp. Biol., 209:2025–2033, 2006.

[16] J.O. Dabiri, S.P. Colin, and J.H. Costello. Morphological diversity of medusan lineages constrained by animal-fluid interactions. J. Exp. Biol., 210:1868–1873, 2007.

[17] J.O. Dabiri and M. Gharib. Starting flow through nozzles with temporally variable exit diameter. J. Fluid Mech, 538:111–136, 2005.

[18] T.L. Daniel. Mechanics and energetics of medusan jet propulsion. Canadian J. of Zoology, 61(6):1406–20, 1983.

[19] T.L. Daniel. Cost of locomotion: Unsteady medusan swimming. J. Exp. Biol., 119(1):149–164, 1985.

[20] J.M. Delery. Robert Legendre and Henri Werle: Toward the elucidation of three-dimensional separation. Ann. Rev. Fluid Mech., 33:129–154, 2001.

[21] M. Dellnitz and O. Junge. An adaptive subdivision technique for the approximation of attractors and invariant measures. Computing and Visualization in Science, 1(2):63–68, April 1998.

[22] T. Dombre, U. Frisch, J.M. Greene, M. Henon, A. Mehr, and A.M. Soward. Chaotic streamlines in the ABC flows. J. Fluid Mech, 167:353–391, 1986.

[23] R.P. Dwight. Robust mesh deformation using the linear elasticity equations. In H. Deconinck and E. Dick, editors, Computational Fluid Dynamics. Springer, 2006.

[24] B.A. Elliott. Anticyclonic rings and the energetics of the circulation of the Gulf of Mexico. PhD thesis, Texas A and M Univ., College Station, TX, January 1979.

[25] B.A. Elliott. Anticyclonic rings in the Gulf of Mexico. Journal of Physical Oceanography, 12:1292–1309, 1982.

[26] M. Farazmand and G. Haller. Computing Lagrangian coherent structures from variational LCS theory. Chaos, 22:013128, 2012.

[27] M.D. Ford, J.H. Costello, K. B. Heidelberg, and J. E. Purcell. Swimming and feeding by the scyphomedusa *Chrysaora quinquecirrha*. Mar. Biol., 129:355–362, 1997.

[28] G. Froyland and K. Padberg. Almost-invariant sets and invariant manifolds – connecting probabilistic and geometric descriptions of coherent structures in flows. Physica D, 238:1507–1523, 2009.

[29] G. Froyland and K. Padberg. Almost invariant sets and invariant manifolds - connecting probabilistic and geometric descriptions of coherent structures in flows. Physica D, 238:1507–1523, 2009.

[30] G. Froyland, K. Padberg, M.H. England, and A.M. Treguier. Detection of coherent oceanic structures via transfer operators. Phys. Rev. Lett., 98(22):224503, May 2007.

[31] C. Garth, F. Gerhardt, X. Tricoche, and H. Hagen. Efficient computation and visualization of coherent structures in fluid flow applications. IEEE Transactions on Visualization and Computer Graphics, 13(6):1464, 2007.

[32] M. Gharib, E. Rambod, and K. Shariff. A universal time scale for vortex ring formation. J. Fluid Mech, 360:121–140, 1998.

[33] M.A. Green, C.W. Rowley, and G. Haller. Detection of Lagrangian coherent structures in 3d turbulence. J. Fluid Mech, 572:111–120, February 2007.

[34] G. Haller. Distinguished material surfaces and coherent structures in three-dimensional fluid flows. Physica D, 149:248–277, 2001.

[35] G. Haller. A variational theory of hyperbolic Lagrangian coherent structures. Physica D, 240:574–598, 2011.

[36] G. Haller and T. Sapsis. Where do inertial particles go in fluid flows. Physica D, 237:573–583, 2008.

[37] G. Haller and G. Yuan. Lagrangian coherent structures and mixing in two-dimensional turbulence. Physica D, 147:352–370, 2000.

[38] M.J.M. Hill. On a spherical vortex. Phil. Trans. Roy. Soc. A, 185:213–245, 1894.

[39] C.W. Hirt, A.A. Amsden, and J.L. Cook. An arbitrary Lagrangian-Eulerian computing method for all flow speeds. J. Comp. Physics, 14:227–253, 1974.

[40] X. Hong, S.W. Chang, S. Raman, L.K. Shay, and R. Hodur. The interaction between Hurricane Opal (1995) and a warm core ring in the Gulf of Mexico. Monthly Weather Review, 128(5):1347–1365, 2000.

[41] H.E. Hurlburt and J.D. Thompson. A numerical study of Loop Current intrusions and eddy shedding. Journal of Physical Oceanography, 10(10):1611–1651, 1980.

[42] D.W. James. The thermal diffusivity of ice and water between -40 and +60 C. Journal of Materials Science, 3(5):540–543, 1968.

[43] A. Johnson and T. Tezduyar. Mesh update strategies in parallel finite element computations of flow problems with moving boundaries and interfaces. Comput. Methods App. Mech. Engrg., 119:73–94, 1994.

[44] B. Krauskopf and H. Osinga. Two-dimensional global manifolds of vector fields. Chaos, 9(3):768–774, 1999.

[45] M. Krieg and K. Mohseni. Thrust characterization of a bio-inspired vortex ring generator for locomotion of underwater robots. IEEE J. Oceanic Engineering, 33(2):doi 10.1109/JOE.2008.920212 (8pp), 2008.

[46] P.S. Krueger and M. Gharib. The significance of vortex ring formation to the impulse and thrust of a starting jet. Phys. Fluids, 15(5):1271–1281, 2003.

[47] P.A. Lagerstrom. Laminar Flow Theory. Princeton University Press, Princeton, New Jersey, 1964.

[48] H. Lamb. Hydrodynamics. Dover, New York, 1945.

[49] F. Lekien. Time-dependent dynamical systems and geophysical flows. PhD thesis, California Institute of Technology, Pasadena, California, 2003.

[50] F. Lekien and S.D. Ross. The computation of finite-time Lyapunov exponents on unstructured meshes and for non-Euclidean manifolds. Chaos, 20:017504, 2010.

[51] F. Lekien, S.C. Shadden, and J.E. Marsden. Lagrangian coherent structures in n-dimensional systems. Journal of Mathematical Physics, 48(065404):1–19, 2007.

[52] S. Leung. An Eulerian approach for computing the finite time Lyapunov exponent. J. Comp. Physics, 230(9):3500 – 3524, 2011.

[53] J.K. Lewis, A.D. Kirwan Jr., and G.Z. Forristall. Evolution of a warm-core ring in the Gulf of Mexico: Lagrangian observations. Journal of Geophysical Research, 94(C6):8163–8178, 1989.

[54] D. Lipinski, B. Cardwell, and K. Mohseni. A Lagrangian analysis of a two-dimensional airfoil with vortex shedding. J. Phys. A, 41:344011, 2008.

[55] D. Lipinski and K. Mohseni. Flow structures and fluid transport for the hydromedusa *Sarsia tubulosa*. AIAA paper 2009-3974, 39^{th} AIAA Fluid Dynamics Conference, San Antonio, Texas, June 22-25 2009.

[56] D. Lipinski and K. Mohseni. Flow structures and fluid transport for the hydromedusae *Sarsia tubulosa* and *Aequorea victoria*. J. Exp. Biol., 212:2436–2447, 2009.

[57] D. Lipinski and K. Mohseni. A ridge tracking algorithm and error estimate for efficient computation of Lagrangian coherent structures. Chaos, 20:017504 (9pp.), 2010. doi:10.1063/1.3270049.

[58] D. Lipinski and K. Mohseni. A 3D fast algorithm for computing Lagrangian coherent structures via ridge tracking. arXiv preprint arXiv:1202.5236, 2012.

[59] W.E. Lorensen and H.E. Cline. Marching cubes: A high resolution 3D surface construction algorithm. SIGGRAPH Comput. Graph., 21(4):163–169, 1987.

[60] E.N. Lorenz. Deterministic nonperiodic flow. Journal of the Atmospheric Sciences, 20(2):130–141, 1963.

[61] RS MacKay, JD Meiss, and IC Percival. Transport in hamiltonian systems. Physica D: Nonlinear Phenomena, 13(1-2):55–81, 1984.

[62] L.P. Madin. Feeding behavior of tentaculate predators: in situ observations and a conceptual model. Bull. Mar. Sci., 43:413–429, 1988.

[63] J.A. Madrid and A.M. Mancho. Distinguished trajectories in time dependent vector fields. Chaos, 19:013111, 2009.

[64] J.D. Meiss. Differential Dynamical Systems (Monographs on Mathematical Modeling and Computation). Society for Industrial and Applied Mathematics, 2007.

[65] K. Mohseni. A: Universality in Vortex Formation B: Evaluation of Mach Wave Radiation in a Supersonic Jet. PhD thesis, California Institute of Technology, April 2000.

[66] K. Mohseni. Pulsatile jets for unmanned underwater maneuvering. AIAA paper 2004-6386, Chicago, Illinois, 20-23 September 2004. 3rd AIAA Unmanned Unlimited Technical Conference, Workshop and Exhibit.

[67] K. Mohseni. Pulsatile vortex generators for low-speed maneuvering of small underwater vehicles. Ocean Engineering, 33(16):2209–2223, 2006.

[68] K. Mohseni, H. Ran, and T. Colonius. Numerical experiments on vortex ring formation. J. Fluid Mech, 430:267–282, 2001.

[69] R.K. O'Dor. Limitations on locomotor performance in squid. J. Appl. Physiol., 64(1):128–134, 1988.

[70] E. Ott. Chaos in dynamical systems. Cambridge Univ Pr, 2002.

[71] J.M. Ottino. The kinematics of mixing: stretching, chaos, and transport, volume 3. Cambridge Univ Pr, 1989.

[72] K. Padberg. Numerical Analysis of Chaotic Transport in Dynamical Systems. PhD thesis, University of Paderborn, 2005.

[73] J. Peng and J.O. Dabiri. Transport of inertial particles by Lagrangian coherent structures: application to predator-prey interaction in jellyfish feeding. J. Fluid Mech., 623:75–84, 2009.

[74] L. Piegl. On NURBS, a survey. IEEE Computer Graphics and Applications, 11(1):55–77, 1991.

[75] A. Poisson and A. Papaud. Diffusion coefficients of major ions in seawater. Marine Chemistry, 13(4):265–280, 1983.

[76] V. Rom-Kedar and S. Wiggins. Transport in two-dimensional maps. Archive for Rational Mechanics and Analysis, 109(3):239–298, 1990.

[77] B. Rutherford, G. Dangelmayr, J. Persing, M. Kirby, and M.T. Montgomery. Lagrangian mixing in an axisymmetric hurricane model. Atmos. Chem. Phys, 10:6777–6791, 2010.

[78] Y. Saad and M. H. Schultz. GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems. SIAM J. Sci. Statist. Comput., 7(3):856–869, 1986.

[79] F. Sadlo and R. Peikert. Efficient visualization of Lagrangian coherent structures by filtered AMR ridge extraction. IEEE Transactions on Visualization and Computer Graphics, 13(6):1456, 2007.

[80] M. Sahin and K. Mohseni. An arbitrary Lagrangian-Eulerian formulation for the numerical simulation of flow patterns generated by the hydromedusa *Aequorea victoria*. J. Comp. Physics, 228:4588–4605, 2009.

[81] M. Sahin, K. Mohseni, and S. Colins. The numerical comparison of flow patterns and propulsive performances for the hydromedusae *Sarsia tubulosa* and *Aequorea victoria*. J. Exp. Biol., 212:2656–2667, 2009.

[82] T. Sapsis and G. Haller. Inertial particle dynamics in a hurricane. J. Atmosph. Sci., 66:2481–2492, 2009.

[83] R. Scharroo, W.H.F. Smith, and J.L. Lillibridge. Satellite altimetry and the intensification of Hurricane Katrina. Eos, Transactions American Geophysical Union, 86(40):366, 2005.

[84] S.C. Shadden, J.O. Dabiri, and J.E. Marsden. Lagrangian analysis of fluid transport in empirical vortex ring flows. Physics of Fluids, 18:7105–7120, 2006.

[85] S.C. Shadden, F. Lekien, and J.E. Marsden. Definition and properties of Lagrangian coherent structures from finite time Lyapunov exponents in two-dimensional aperiodic flows. Physica D, 212:271–304, 2005.

[86] S.C. Shadden, F. Lekien, J.D. Paduan, F.P. Chavez, and J.E. Marsden. The correlation between surface drifters and coherent structures based on high-frequency radar data in Monterey Bay. Deep-Sea Research II, 56:161–172, 2009.

[87] L.K. Shay, G.J. Goni, and P.G. Black. Effects of a warm oceanic feature on Hurricane Opal. Monthly Weather Review, 128(5):1366–1383, 2000.

[88] S. Smale. Differentiable dynamical systems. Bulletin of the American Mathematical Society, 73(6):747–817, 1967.

[89] R.K. Smith and M.T. Montgomery. Hurricane boundary-layer theory. Quarterly Journal of the Royal Meteorological Society, 136(652):1665–1670, 2010.

[90] W. Sturges and R. Leben. Frequency of ring separations from the Loop Current in the Gulf of Mexico: A revised estimate. Journal of Physical Oceanography, 30(7):1814–1819, 2000.

[91] Unidata. NetCDF (network Common Data Form). `http://www.unidata.ucar.edu/software/netcdf/`, September 2009.

[92] B. Van der Pol. On relaxation-oscillations. The London, Edinburgh, and Dublin Phil. Mag. & J. of Sci., 2(11):978–992, 1926.

[93] S.E. Welsh and M. Inoue. Loop Current rings and the deep circulation in the Gulf of Mexico. Journal of Geophysical Research, 105:16, 2000.

[94] J. Weston, S.P. Colin, J.H. Costello, and E. Abbott. Changing form and function during development in rowing hydromedusae. Mar. Ecol. Prog. Ser., 374:127–134, 2009.

[95] M. Wilson, J. Peng, J.O. Dabiri, and J.D. Eldredge. Lagrangian coherent structures in low Reynolds number swimming. J. Physics: Condensed Matter, 21:204105, 2009.

[96] Q. Xiao, X. Zhang, C. Davis, J. Tuttle, G. Holland, and P.J. Fitzpatrick. Experiments of hurricane initialization with airborne doppler radar data for the advanced research hurricane wrf (ahw) model. Monthly Weather Review, 137(9):2758–2777, 2009.

Appendix A

Proof that the FTLE is non-negative for incompressible flows

For incompressible flows, contraction of a fluid in one direction must be balanced by expansion in another direction. In terms of LCS, this translates to the following theorem:

Theorem 5. *For incompressible flows, the FTLE field is non-negative.*

$$\nabla \cdot \mathbf{u} = 0 \quad \Rightarrow \quad \sigma_{t_0}^{T}(\mathbf{x}) \geq 0 \quad \forall\, \mathbf{x},\ t_0,\ T$$

The gradient of the flow field, commonly denoted $\frac{d\Phi}{d\mathbf{x}}$ is the same as the Jacobi matrix, $\mathbf{J}$, which appears in many Lagrangian fluid dynamics texts. It is a well known result that if the flow is incompressible the determinant of the Jacobi matrix is 1: $\det(J) = 1$ (for example, see Meiss, p. 63 [64]).

Also, for general matrices A and B,

$$\begin{aligned} \det(AB) &= \det(A)\det(B), \\ \det(A^{\mathsf{T}}) &= \det(A), \\ \det(A) &= \prod_{i=1}^{d} \lambda_i \ . \end{aligned}$$

Then, using the definition of the FTLE field,

$$\begin{aligned}
\Delta &= J^\intercal J \\
\det(\Delta) &= \det(J^\intercal)\det(J) \\
&= 1 \\
&= \prod_{i=1}^{d} \lambda_i(\Delta)
\end{aligned}$$

Δ is positive definite since

$$\begin{aligned}
\langle \mathbf{x}, \Delta\mathbf{x} \rangle &= \langle \mathbf{x}, J^\intercal J\mathbf{x} \rangle \\
&= \langle J\mathbf{x}, J\mathbf{x} \rangle \\
&= ||J\mathbf{x}||^2 \geq 0
\end{aligned}$$

and

$$\det(\Delta) = 1$$

So $\lambda_i(\Delta) > 0$ and

$$\lambda_{\max}(\Delta) = \max_i (\lambda_i(\Delta)) \geq 1$$

and

$$\sigma = \frac{1}{|T|} \ln \sqrt{\lambda_{\max}(\Delta)} \geq \frac{1}{|T|} \ln \sqrt{1} = 0$$

$\square$

Appendix B

Approximating the stable and unstable manifolds of the Lorenz system

The Lorenz system was developed by Edward Lorenz as a model for atmospheric convection [60]. This system generates the famous “butterfly attractor” that is the unstable manifold of the hyperbolic fixed point of the origin. Much work has been done to accurately compute the stable and unstable manifolds of this (and other) system. In particular, the work of Krauskopf and Osinga focused on computing and visualizing the manifolds of the Lorenz system [44]. The Lorenz system is defined by

$$\begin{aligned}
\dot{x} &= \sigma(y - x),\\
\dot{y} &= x(\rho - z) - y,\\
\dot{z} &= xy - \beta z,
\end{aligned}$$

where $\sigma = 10$ $\beta = 8/3$, and $\rho = 28$ in this appendix.

To visualize the stable and unstable manifolds of the Lorenz system, I use a method similar to that of Krauskopf and Osinga [44]. This method is easily implemented in MATLAB and allows for quick computations of one- or two-dimensional manifolds in three-dimensional systems. One-dimensional manifolds may be easily approximated as a series of points while two-dimensional manifolds are represented as a series of topological circles connected by triangles (giving a surface triangulation).

The first step is to compute the eigenvectors of a saddle point in the system to determine the tangent space of the manifold and initialize manifold extraction. Let the dynamical system be

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x})$$

with a hyperbolic point at $\mathbf{x}_0$ such that $\mathbf{f}(\mathbf{x}_0) = \mathbf{0}$. Then assume $\nabla \mathbf{f}(\mathbf{x}_0)$ has eigenvectors $\{\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3\}$ and eigenvalues $\lambda_1 \leq \lambda_2 < 0$, $\lambda_3 > 0$. If there are two positive eigenvalues, the dimensions of the stable and unstable manifolds (the methods used to approximate them) are simply switched.

By noting that $\mathbf{f}$ is always tangent to the manifolds, the stable and unstable manifold may be simply computed. The one-dimensional unstable manifold $\dot{\mathbf{X}}_s$ is estimated by solving

$$\dot{\mathbf{X}}_s(s) = \mathbf{f}(\mathbf{X}_s(s)) / \left\| \mathbf{f}(\mathbf{X}_s(s)) \right\|$$

with initial conditions

$$\mathbf{X}_s(0) = \mathbf{x}_0,$$
$$\dot{\mathbf{X}}_s(0) = \pm \mathbf{v}_3,$$

where s is the geodesic distance from the fixed point along the manifold. This system may be solved using many numerical methods including Euler or Runge-Kutta methods and the resulting manifold is shown in Figure B.1.

Computing two-dimensional manifolds is somewhat more complicated. I initialize the manifold with six points on a circle in the tangent plane defined by the two vectors $\{\mathbf{v}_1, \mathbf{v}_2\}$ and connected to the origin in six triangles. Each step forward along the manifold is completed by taking a step of length dx in the direction $-\mathbf{f}(\mathbf{x}_i)$ for all nodes along the edge of the currently detected manifold, adding a topological circle of new points. Typically the circumference of this topological circle grows with each step, necessitating the addition and redistribution of points along it. This is accomplished by defining a periodic spline through the new nodes and then evenly distributing points along the spline to obtain a spacing of approximately dx. The new points are then connected to the previous topological circle by projecting this one parameter curve onto the line $y = 1$ and the previous curve onto the line

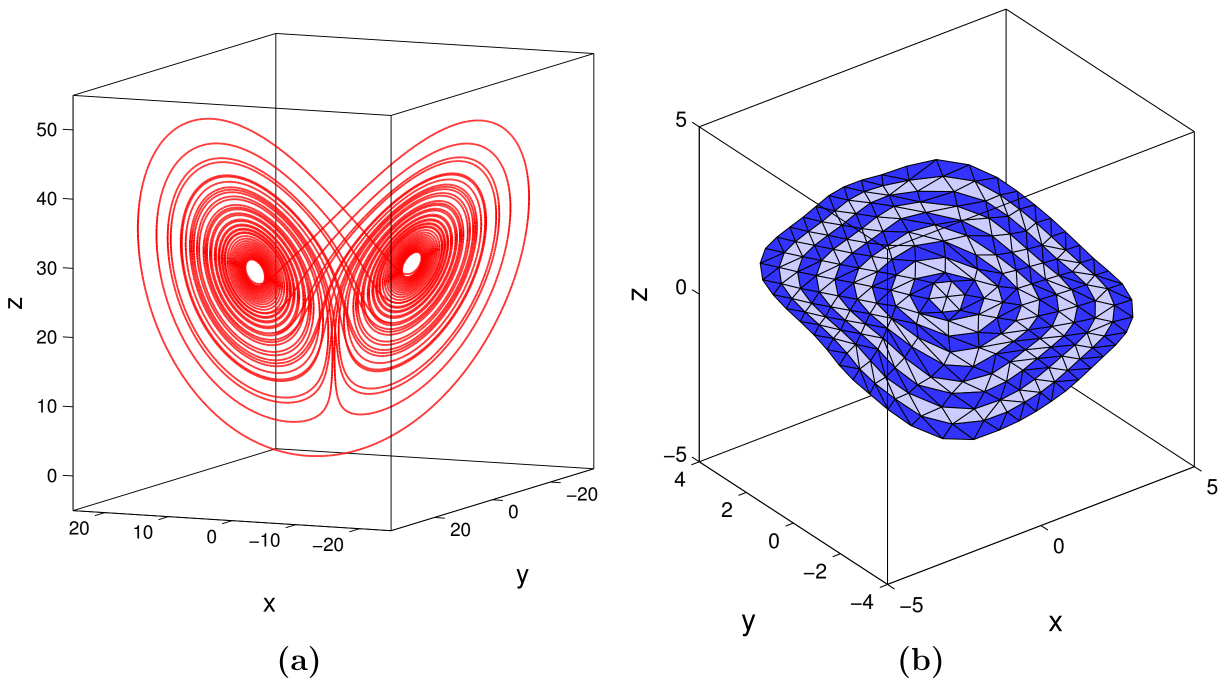


Figure B.1: a) The unstable manifold of the Lorenz system for $s \leq 1000$. b) The stable manifold of the Lorenz system. 10 steps were taken along the manifold with $dx = 0.5$.

$y = 0$ in the $x - y$ plane. The nodes on these two lines are then connected using a Delaunay triangulation which then determines the surface triangulation in 3-space. The results of the first 10 steps along the manifold with spacing $dx = 0.5$ are shown in Figure B.1. The two manifolds (stable and unstable) are shown together in Figure B.2.

If higher accuracy is desired for this method, a Runge-Kutta type method could be used at each step. Additionally, no attempt was made here to see that the successive topological circles are each separated by a geodesic distance of dx. Instead, each step of dx may have a significant component parallel to the previous edge, resulting in smaller steps forward along the manifold. A slightly more complex procedure could be developed to ensure that the steps along the manifold are normal to the edge of the currently detected manifold.

Although the accuracy of this method has not been vetted, extracting the manifolds in this way is a stable process and gives results that closely resemble those of Krauskopf and Osinga [44]. This method may be useful in other systems when a simple and easily implementable method is needed to visualize the stable and unstable manifolds of hyperbolic fixed points. For reference, the MATLAB code used to perform these computations is included below. Note that this code requires the curve fitting toolbox to run.

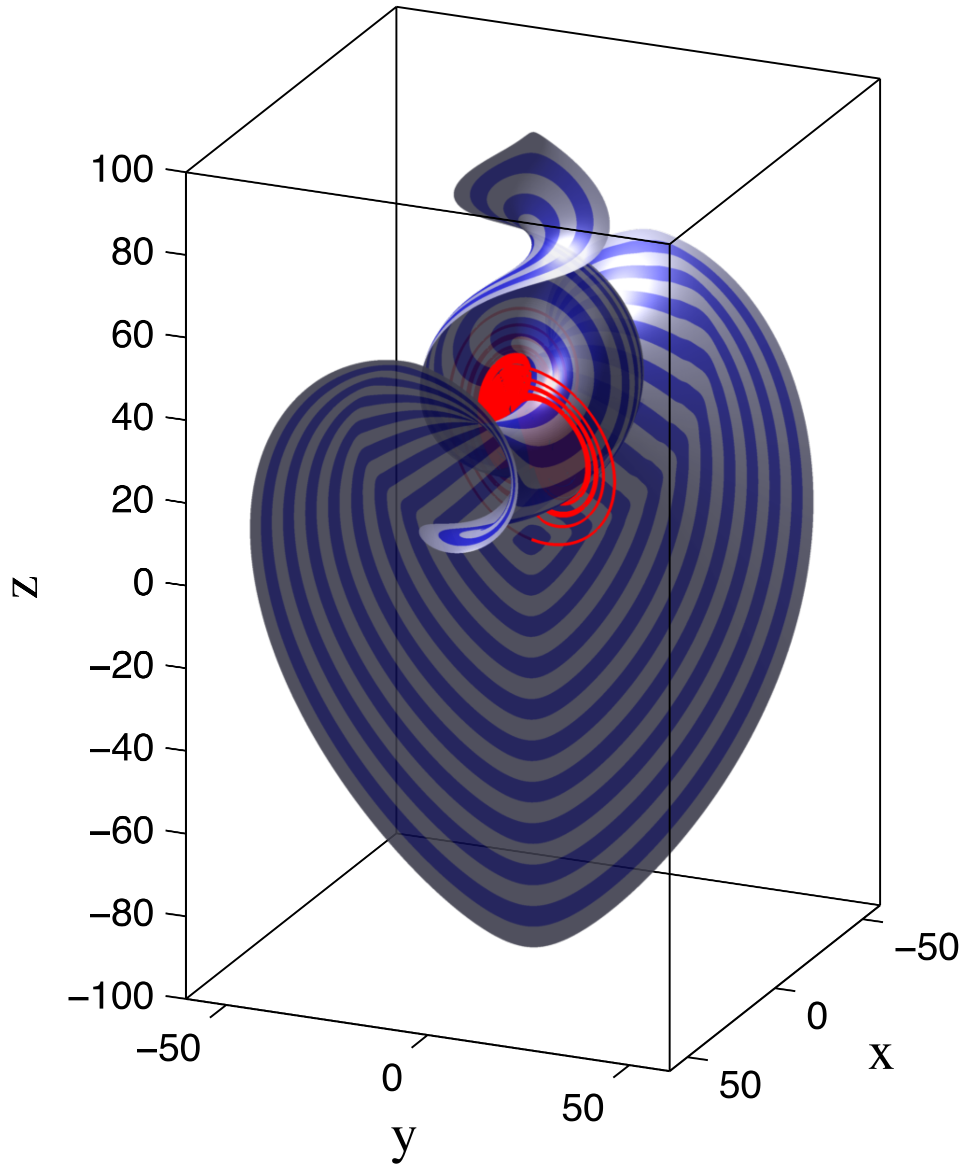


Figure B.2: A part of the stable and unstable manifolds of the Lorenz system. The unstable manifold (red curve) is shown for $s \leq 2000$ while the stable manifold (blue and gray surface) is shown were 200 steps were taken along the manifold with $dx = 0.5$.

MATLAB code used for Lorenz system manifold computations.

File: `velocity.m`

```
function [u v w] = velocity(x,y,z)
% function to return the right hand side of the Lorenz equations
% input: (x,y,z) is a point in phase space
% output: (u,v,w) the velocity at the input point

% system parameters
sigma=10;
rho=28;
beta=8/3;

u=sigma*(y-x);
v=rho*x-y-x.*z;
w=x.*y-beta*z;

return;
```

File: `main.m`

```
clear
clc

%find the eigenvalues/vectors of grad(f) at the origin
A=[-10 10 0; ...
    28 -1 0; ...
    0 0 -8/3]; %this is grad(f)(0) for the Lorenz system
[V D]=eig(A); %V:= eigenvectors, D:=eigenvalues

%%%%%%%%%%%%%%%%  find the unstable manifold  %%%%%%%%%%%%%%%%
dx=.05; %spatial step size
L=2000; %length of the manifold to compute

ii=find([D(1,1) D(2,2) D(3,3)]>0); %find the positive eigenvalue

%initialize the unstable manifold in both directions
x_u=[0 0; dx -dx]*V(1,ii);
y_u=[0 0; dx -dx]*V(2,ii);
z_u=[0 0; dx -dx]*V(3,ii);

for i=1:2 %for each direction
for j=1:L/dx %take enough steps to compute a length of L

    %compute the velocity
    [u v w]=velocity(x_u(j+1,i),y_u(j+1,i),z_u(j+1,i));
    a=norm([u v w]); %magnitude of velocity

    %use a fwd Euler step in the direction of the normalize velocity
    %this could be replaced with a Runge-Kutta method for better accuracy
    x_u(j+2,i)=x_u(j+1,i)+dx*u/a;
```

```
    y_u(j+2,i)=y_u(j+1,i)+dx*v/a;
    z_u(j+2,i)=z_u(j+1,i)+dx*w/a;

end
end

%plot the unstable manifold
figure(1)
clf
hold on
plot3(x_u(:,1),y_u(:,1),z_u(:,1),'r','linewidth',2)
plot3(x_u(:,2),y_u(:,2),z_u(:,2),'r','linewidth',2)
daspect([1 1 1])
box on
view(3)
drawnow


%%%%%%%%%%%%%%%%% find the stable manifold %%%%%%%%%%%%%%%%%
dx=.5; %spatial step size
L=100; %length to step along manifold


k=6; %number of points to use on initial circle
theta = pi/4+linspace(0,2*pi-2*pi/k,k)';

n_s=[1 k]; %number of points on each topological circle (1 then 6 then...)

%step from origin along manifold tangent plane
x=dx*V(1,1)*cos(theta)+dx*V(1,3)*sin(theta);
y=dx*V(2,1)*cos(theta)+dx*V(2,3)*sin(theta);
z=dx*V(3,1)*cos(theta)+dx*V(3,3)*sin(theta);

%add these points to the stable manifold
x_s=[0; x];
y_s=[0; y];
z_s=[0; z];

%define first k surface triangles
stable.faces = [];
for i=1:k
    stable.faces=[stable.faces; 1 2+mod(i,6) 2+mod(i+1,6)];
end
ntri = k;

s_old=0:dx:6*dx; %length along the previous topological circle
for i=1:L/dx %take steps to a total length of L

    %compute and normalize the velocity
    [u v w]=velocity(x,y,z);
    a = sqrt(u.^2 + v.^2 + w.^2);
    u=u./a;
    v=v./a;
```

```
    w=w./a;

    %step along manifold (i.e. in the -f direction for stable manifold)
    x=x-dx*u;
    y=y-dx*v;
    z=z-dx*w;

    %create a periodic spline for the new points (x,y,z) to
    %represent the new topological circle
    xyz=cscvn([x' x(1); y' y(1); z' z(1);]);

    nt=ceil(xyz.breaks(end)/sqrt(dx)+1); %number of new points to use
    t=linspace(0,xyz.breaks(end),nt); %evenly distributed points
                                      %  for spline parameter, t
    s=cumsum([0 diff(t).^2]); %distance parameter, s

    %evaluate the spline at these equal intervals
    z=fnval(xyz,t);
    x=z(1,:)'; y=z(2,:)'; z=z(3,:)';

    %number of points on this new topological circle
    n_s(i+2)=length(x)-1;

    %remove the last point (it's redundant with the first)
    x=x(1:end-1); y=y(1:end-1); z=z(1:end-1);

    %add the new points to the stable manifold
    x_s=[x_s; x];
    y_s=[y_s; y];
    z_s=[z_s; z];

    %Delaunay triangulation to generate surface mesh

    %project previous and current topological circles
    %                      onto the lines y=0 and y=1
    xx=[s_old/max(s_old) s/max(s)];
    yy=[zeros(size(s_old)) ones(size(s))];

    tri=delaunay(xx,yy); %generate Delaunay triangulation

    %add another triangle due to the periodicity of the circle
    ii=find(tri==length(s_old));
    tri(ii)=1;
    ii=find(tri==length([s_old s]));
    tri(ii)=length(s_old)+1;
    ii=find(tri>length(s_old));
    tri(ii)=tri(ii)-1;

    %add a constant to the triangle indices to account for the
    %         previously computed points on the stable manifold
    tri=tri+sum(n_s(1:i));

    %add the the triangulation to the struct holding the surface info
```

```
    stable.faces = [stable.faces; tri];
    ntri = [ntri length(tri)]; %number of triangles added at each step

    s_old=s;
end

stable.vertices=[x_s y_s z_s]; %add the vertices to the surface struct

%optional: reduce the number of triangles in the
% surface representation for faster plotting
% stable=reducepatch(stable,.2);

figure(2)
clf
hold on
patch(stable,'EdgeColor','none','FaceColor',[.5 .5 1])
daspect([1 1 1])
box on
view(3)
camlight
lighting p
drawnow
```

Appendix C

A model for eddy depth in warm-core rings

This appendix discusses a model for eddy depth based on the warm-core rings (WCRs) in the Gulf of Mexico (GoM) mentioned in Chapter 6. The rings are known to influence the total heat balance in the GoM by largely governing the influx of warm Caribbean water [24]. The total heat balance (and correspondingly the surface temperatures) of the GoM is of great importance to the weather and climate of the midwest and eastern parts of the United States. Additionally, the increased mixed layer depth found in warm rings has important effects on hurricane intensity. In several recent cases, hurricanes have rapidly intensified while passing over a warm ring in the GoM due to the increased water temperature and the decreased effect of cold water upwelling associated with hurricanes [40].

In many oceanic (and atmospheric) flows, the flow is largely in geostrophic balance. The full Navier-Stokes equations in a rotating reference frame attached to the earth are given by

$$\frac{D\mathbf{u}}{Dt} = \frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\,\mathbf{u} = \mathbf{g} - \frac{1}{\rho}\nabla p - 2\boldsymbol{\Omega} \times \mathbf{u} + \nu \nabla^2 \mathbf{u} \tag{C.1}$$

where $\mathbf{u}$ is the fluid velocity, $\mathbf{g}$ is a modified gravity term that also includes centrifugal forces, ρ is the density, p is the pressure, Ω is the angular rotation rate of the frame, and ν is the kinematic viscosity of the fluid. Under the assumptions of negligible viscosity (valid outside boundary layers) and hydrostatic balance (a very good approximation), the horizontal motion of the fluid is given by

$$\frac{Du}{Dt} = -\frac{1}{\rho}\frac{\partial p}{\partial x} + fv \tag{C.2}$$

$$\frac{Dv}{Dt} = -\frac{1}{\rho}\frac{\partial p}{\partial y} - fu \tag{C.3}$$

where f is the Coriolis parameter given by $f = 2\Omega \sin\phi$ at latitude ϕ. If the acceleration terms are small, the flow is said to be in geostrophic balance:

$$u = -\frac{1}{fp}\frac{\partial p}{\partial y}, \tag{C.4}$$

$$v = \frac{1}{fp}\frac{\partial p}{\partial x}, \tag{C.5}$$

a balance between the Coriolis force and the horizontal pressure gradients. This is accurate for small Rossby number flows where the Rossby number, $Ro = U/Lf$, is the ratio of inertial to Coriolis effects. The geostrophic balance gives good approximations of many oceanic flows and is particularly useful for estimating surface currents based on satellite altimetry which can be used to directly measure the sea surface height (SSH) anomaly.

In this appendix, the geostrophic balance is used to predict horizontal ocean currents at depth and estimate the depth of WCRs in the Gulf of Mexico. These rings have a positive SSH anomaly and a lower average density when compared to the surrounding ocean. The SSH anomaly creates a near surface high pressure zone that manifests as anticyclonic eddies. However, since the eddies contain less dense water than the surrounding ocean, the pressure anomaly decreases with depth. At the bottom of the eddy, the horizontal pressure gradient becomes negligible. Basic models of the pressure difference between the eddy core and the surrounding ocean versus depth greatly overestimate the eddy depth. Accurate estimates of eddy depth must account for the increased mixed layer depth in these warm-core rings and the corresponding depression of the pycnocline which greatly limits the depth penetration of the eddies.

C.1 Methods

Lagrangian coherent structures are used to accurately determine the boundaries of the WCRs. This allows for a precise determination of the ring boundaries without the need for an arbitrary threshold based on velocity, vorticity, or other parameters. The LCS are computed using the three-dimensional grid-based ridge tracking algorithm. For the velocity data, the ECCO2 global ocean model data is used from the year 2010 (see `http://ecco2.org/`). The data is available at three day intervals on a 0.25° latitude-longitude grid.

As mentioned above, the warm core rings have a positive SSH anomaly, but a lower density than the surrounding ocean. The positive SSH anomaly creates a near surface high pressure region in the eddy that creates the anticyclonic flow. However, the lower density in the eddy core means that the pressure increases more slowly with depth than the pressure in the surrounding ocean. At some point at depth, the pressure difference is expected to become negligible and the geostrophic wind will vanish.

Assuming hydrostatic balance enables estimation of the pressure at depth by integrating the density field in the depth direction. That is,

$$p(z) = g\rho_0\zeta + g\int_0^z \rho(z')dz' \tag{C.6}$$

for SSH ζ, surface density ρ_0, and depth z. Since horizontal pressure gradients determine the geostrophic wind, the pressure difference between the eddy and the surrounding ocean is considered:

$$\delta p(z) = p_{\mathrm{e}}(z) - p_{\mathrm{o}}(z) = g\left(\rho_{0,\mathrm{e}}\zeta_{\mathrm{e}} - \rho_{0,\mathrm{o}}\zeta_{\mathrm{o}}\right) + g\int_0^z \rho_{\mathrm{e}}(z') - \rho_{\mathrm{o}}(z')dz' \tag{C.7}$$

where subscript e and o indicate quantities for the eddy and the surrounding ocean respectively.

A model for the density as a function of depth is needed to compute δp. Seawater density is a complicated function of salinity, temperature and pressure so a simple model for the density is used as well as the actual density field from the ECCO2 simulations to

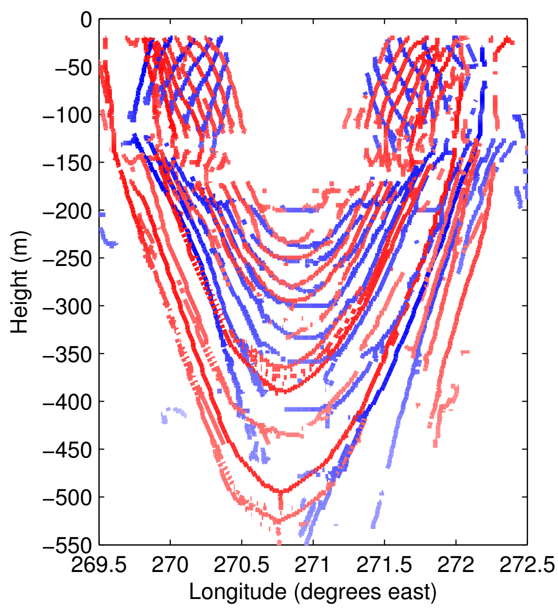


Figure C.1: A vertical slice of warm-core ring. The depth and extents of the eddy are clearly and unambiguously defined by the LCS.

compute δp. All cases use the sea surface height and surface density at the eddy center from the ECCO2 model to define ζ_{e} and $\rho_{0,\mathrm{e}}$. ζ_{o} and $\rho_{0,\mathrm{o}}$ are determined based on the average values of ζ and ρ_0 around the eddy at a radius of 3° latitude/longitude from the eddy center.

A simple piecewise linear model is first used to approximate $\rho(z)$. This model matches the densities at the bottom of the eddy and at all points below this level. The model is given by

$$\rho_{\mathrm{o}}(z) = \rho_{0,\mathrm{o}} + m_1 z, \tag{C.8}$$

$$\rho_{\mathrm{e}}(z) = \begin{cases} \rho_{0,\mathrm{e}} + m_2 z & \text{if } z < d \\ \rho_{0,\mathrm{o}} + m_1 z & \text{if } z \geq d \end{cases}. \tag{C.9}$$

Requiring continuity of $\rho(z)$ gives $m_2 = m_1 + (\rho_{0,\mathrm{o}} - \rho_{0,\mathrm{e}})/d$. Substituting into Equation C.7 results in the m_1 terms canceling and gives an eddy depth of

$$d = -2\frac{\rho_{0,\mathrm{o}}\zeta_{\mathrm{o}} - \rho_{0,\mathrm{e}}\zeta_{\mathrm{e}}}{\rho_{0,\mathrm{o}} - \rho_{0,\mathrm{e}}}. \tag{C.10}$$

For comparison, direct integration of the density at the eddy center and the surrounding ocean as found by the ECCO2 simulation is used to determine the pressure difference. These values will all be compared to the depth as determined by the LCS as shown in Figure C.1. The pressure difference is shown in Figure C.2. In reality the eddy will not extend all the way to the point where $\delta p = 0$ since ageostrophic effects will have some effect and there is generally some rotating flow beyond the transport boundaries of eddies and vortices as revealed by LCS. A threshold of 10% of the surface pressure difference is used to approximate the eddy depth. The horizontal dotted line in Figure C.2 marks this 10% cutoff and gives eddy depths of 550 m for the ECCO2 data and 1750 m for the linear model. The density model greatly overestimates the eddy depth. This is due to the specific shape of the density profile and the increased mixed layer depth in eddy cores. A sample density profile is shown in Figure C.3.

The density profile is approximately linear in the mixed layer and in the deep ocean. In the pycnocline the density changes rapidly due to the corresponding changes in temperature

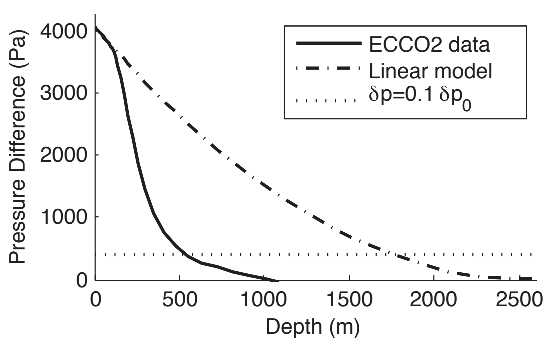


Figure C.2: The pressure difference between the eddy core and the surrounding ocean as measured from the ECCO2 simulation and approximated by the constant and linear density models.

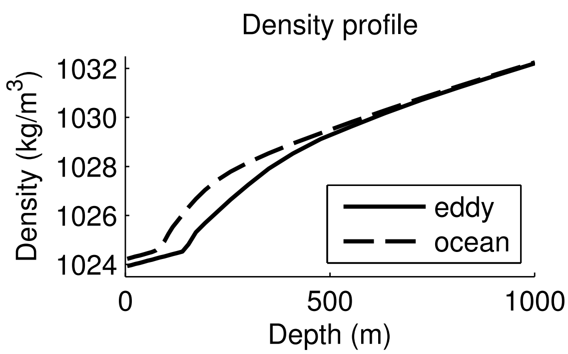


Figure C.3: The density profile in an eddy and the surrounding ocean. The profile is linear in the mixed layer and at depth, the area between the curves accounts for the changing horizontal pressure gradient.

and salinity before approaching the linear deep ocean profile. The area between the curves in Figure C.3 accounts for the change in the horizontal pressure gradients with increasing depth. Since the eddy mixed layer depth is larger than the surrounding ocean, this area is greatly increased, with the result that the eddy depth is less than that predicted by the model discussed above.

To capture this nonlinear behavior I propose an exponential model of the form

$$\rho(z) = \begin{cases} \rho_0 + mz & \text{if } z < D \\ C_0 + mz + (\rho_0 - C_0)\exp\left(-A(z-D)^{\alpha}\right) & \text{if } z \geq D \end{cases} \tag{C.11}$$

where z is the depth, ρ_0 is the surface density, and D is the mixed layer depth. The model also contains four parameters (m, C_0, A, and α), but these parameters are closely tied to the seawater properties.

The slope of the linear parts of Equation C.11 is determined by m and can be estimated based on the bulk modulus of seawater. Within the mixed layer and also in the deep ocean, the variations in salinity and temperature are relatively small so most of the density variation is due to the increasing pressure. This compressibility is described by

$$\frac{\partial \rho}{\partial p} = \frac{\rho}{E} \tag{C.12}$$

where E is the bulk modulus of the seawater. Again using the hydrostatic balance, $\partial p/\partial z = \rho g$. Using the chain rule and Equation C.12 gives

$$\frac{\partial \rho}{\partial z} = \frac{\rho^2 g}{E} \tag{C.13}$$

so $m = \rho^2 g/E \approx 4.4 \times 10^{-3}$ kg/m^4. Additionally, C_0 is the density deep water would have if it were brought to the surface isothermally and expanded according to Equation C.13. For the GoM data in this simulation, $C_0 \approx 1028$ kg/m^3.

Finally, the parameters A and α determine the shape of the exponential part of Equation C.11. These parameters are closely tied to one another and are related to the thickness

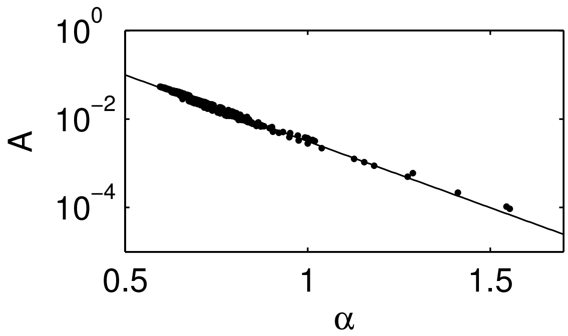


Figure C.4: The parameters A and α as determined from best fits of eq. C.11 to the ECCO2 data at many time steps.

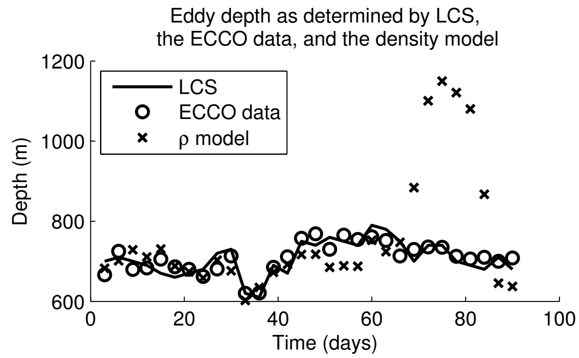


Figure C.5: The eddy depth as determined by the LCS, direct integration of the ECCO2 density field and integration of the exponential model of the density profile. Note that the exponential model fails late in the time period due to the development of a second pycnocline.

of the pycnocline and how quickly the temperature and salinity approach the deep water profile. Using the data from the ECCO2 model, I perform a least squares best fit between Equation C.11 and the density profile to determine A and α. This is done for many different time steps both inside an eddy and in the surrounding ocean. The resulting values of A and α are plotted in Figure C.4. The points all align along the curve $\ln(A) = c_1\alpha + c_2$. Fitting this curve to the values of A and α gives $c_1 \approx -6.914$ and $c_2 \approx 1.148$. Using this relationship and the values of m and C_0 discussed above reduces the four parameter model to a single parameter related to the how the density approaches the deep water values beneath the pycnocline.

The results of the Eddy depth prediction are presented in Figure C.5. As a base case, the "true" eddy depth is determined based on the deepest extent of the LCS surrounding the eddy. This LCS surface represents the separation between water that circulates with the eddy and the surrounding ocean. Also plotted is the eddy depth estimated directly from the ECCO2 data and the depth estimated using the exponential model of the density profile. These three values are in good agreement except for six outliers in the model estimates from days 70-85. The model fails at these points due to the development of a secondary pycnocline that cannot be captured by the model. This secondary pycnocline occurs during the spring months that correspond to increased rainfall and river runoff that create a less salty surface layer in the northern GoM.

These results are based on using a best fit of Equation C.11 to the data at each time step to determine α, enabling computation A based on the results seen in Figure C.4. The other parameters, m and C_0 use the values of 4.4×10^{-3} and 1028 as discussed above.

C.2 Conclusions

The depressed mixed layer present in WCRs acts to limit their depth by increasing the pressure difference between the eddy and the surrounding ocean. This leads to shallower eddies as the geostrophic wind decreases more quickly with depth. Linear models of the seawater density are not capable of capturing the pycnocline at the bottom of the mixed layer and therefore tend to overestimate the total eddy depth. An exponential model accurately predicts the eddy depth when only a single pycnocline is present. This model describes the seawater density as a function of depth and is linear in the mixed layer and for large depths. The pycnocline is captured by an exponential decay from the mixed layer profile to the profile at depth. The model depends on the sea surface height and mixed layer depth as well as four parameters. Two of the parameters can be estimated a priori and the remaining two have been shown to have a simple functional relationship leaving only one free parameter.

Appendix D

Hurricane asymmetry and transport

This appendix presents the effects of a large asymmetry in the low level eyewall of a hurricane simulation. The simulation uses the Weather Research and Forecasting (WRF) model to simulate hurricane Rita (2005). The three-dimensional ridge tracking algorithm is used to extract the resulting LCS surfaces from the flow. Two-dimensional slices of the LCS at constant height reveal a large asymmetry in the hurricane eyewall, corresponding to strong updrafts and fluid being detrained from the eye and ejected upwards and outwards near the hurricane's upper limits.

D.1 Methods

Since the LCS surfaces are parameterized by a large number of triangles, computing a two-dimensional slice amounts to computing the intersections of triangles with a plane and then plotting all the resulting line segments. First, the slice plane is defined by

$$\hat{\mathbf{n}} \cdot (\mathbf{x} - \mathbf{x}_0) = 0 \tag{D.1}$$

where $\hat{\mathbf{n}}$ is a normal vector to the plane and $\mathbf{x}_0$ is a point in the plane. Then, given an edge of the triangle with vertices $\mathbf{a}$ and $\mathbf{b}$, define a line segment

$$\mathbf{x} = \mathbf{a} + s(\mathbf{b} - \mathbf{a}), \quad 0 \le s \le 1. \tag{D.2}$$

By substitution, one arrives at

$$\hat{\mathbf{n}} \cdot (\mathbf{a} + s(\mathbf{b} - \mathbf{a}) - \mathbf{x}_0) = 0, \quad 0 \le s \le 1 \tag{D.3}$$

which gives

$$s\,\mathbf{n}\cdot(\mathbf{b}-\mathbf{a}) = \mathbf{n}\cdot(\mathbf{x}_0-\mathbf{a}). \tag{D.4}$$

If this system has no solution for $s \in [0,1]$, then the edge of the triangle does not intersect the slice plane. If the system has infinite solutions (i.e. $\hat{\mathbf{n}}\cdot(\mathbf{b}-\mathbf{a}) = \hat{\mathbf{n}}\cdot(\mathbf{x}_0-\mathbf{a}) = 0$) then the edge lies in the plane. If there is a single solution for $s \in [0,1]$, the intersection point is found directly from Equation D.2 and another intersection point will be found on one of the other triangle edges, making a line segment. These line segments may be colored according to the FTLE values and plotted to display the LCSs in the chosen two-dimensional slice.

Once a two-dimensional slice has been computed, passive drifters are inserted into the flow at key locations based on the information gained from the LCS. In two-dimensional flows, observing the time dependent LCS is often enough to gain a fairly complete understanding of the transport, but in highly three-dimensional flows, the third spacial dimension makes such visualizations less informative. If particles are transported out of the visualization plane, their behavior is no longer governed by the motion of the LCS being viewed.

D.2 Results

A two-dimensional horizontal slice of the LCS surfaces at a height of 400 m is shown in Figure D.1, along with three groups of drifters. This particular time step was chosen because of the the shark fin shaped structure north of the hurricane eye which contains the cyan and magenta drifters. This structure is the result of an asymmetry in the eye over the previous several hours which rotated around the hurricane and eventually developed the strong forward (blue) LCS separating it from the rest of the eye. This particular LCS is seen as the blue curve that separates the green drifters from the magenta.

The drifter groups seen in Figure D.1 were chosen in a combination of ways. First, a rectangular grid of drifters was inserted over the entire pictured domain. Then, the drifters were advected backward in time to see where they came from. Only the drifters that began

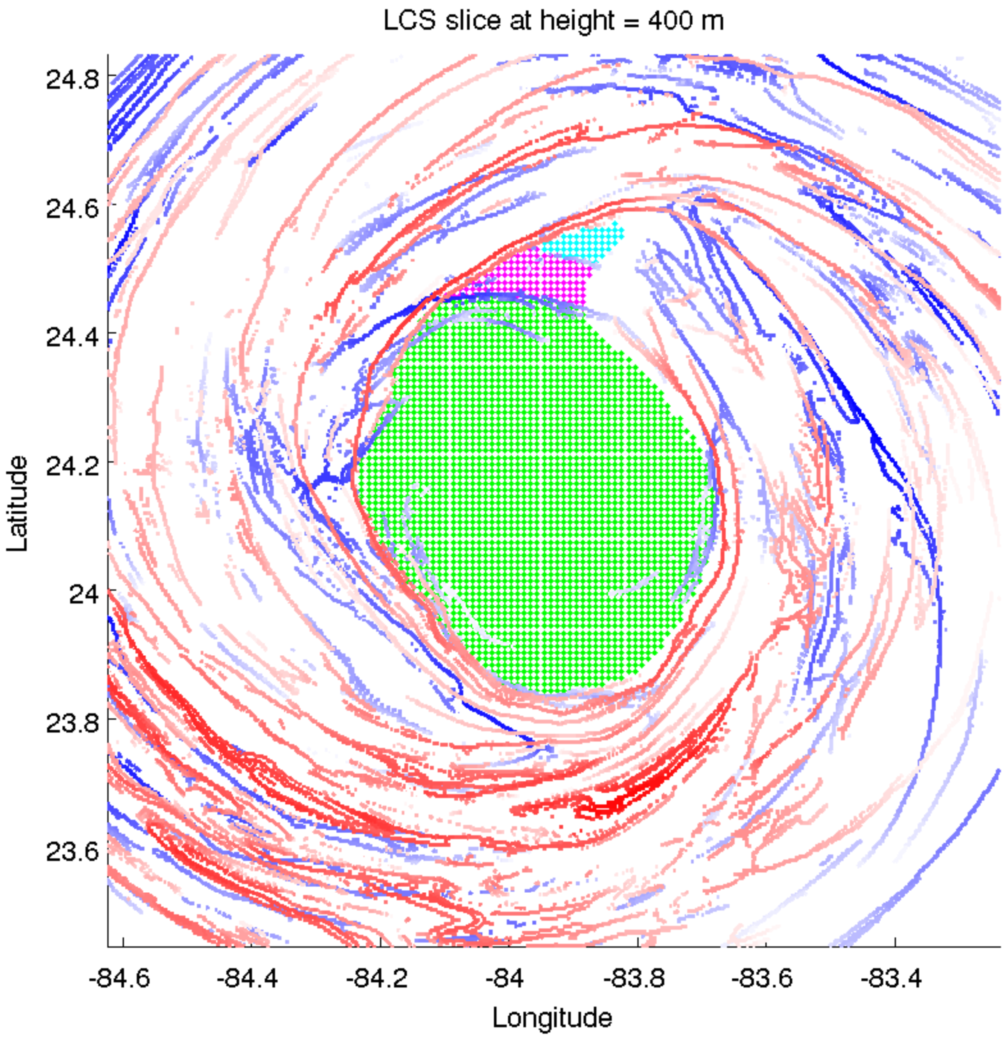


Figure D.1: A horizontal two-dimensional cross section of the LCS surfaces at a height of 400 m. Three groups of drifters are also shown in green, magenta, and cyan.

in the hurricane eye (6 hours previously) were selected (see Figure D.2, top). This group consists of all the drifters (green, magenta, and cyan) shown in Figure D.1. This group was then further divided into the three groups that are shown. The green group and the magenta group are separated by the strong forward LCS (blue) seen in the Figure. The magenta group and the cyan groups were then separated based on their future trajectories. The magenta group rises very quickly to the top of the hurricane and within 6 hours all magenta drifters lie at heights greater than 8000 m while all cyan drifters lie below 8000 m (see Figure D.2, bottom).

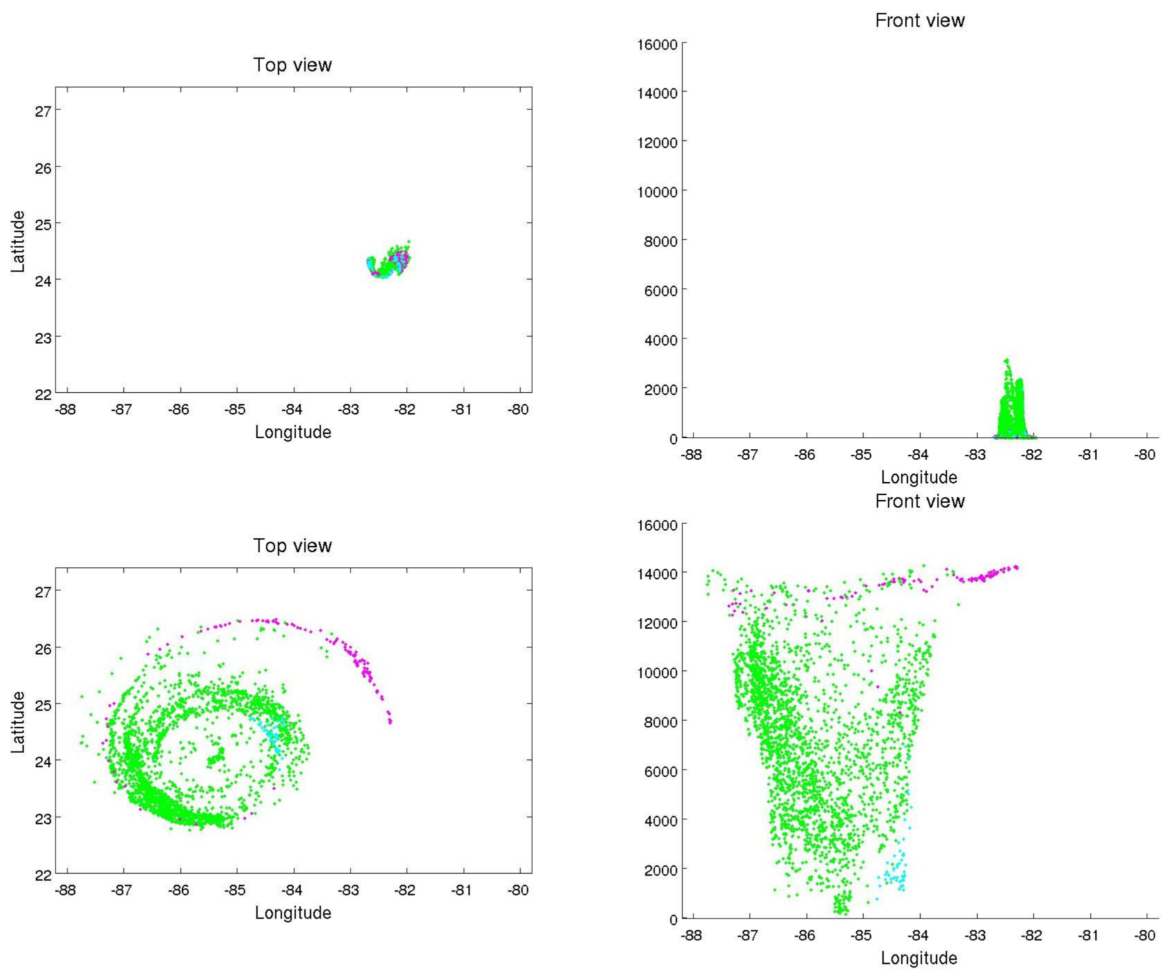


Figure D.2: The past ($t_0 - 6$ hrs, shown at top) and future ($t_0 + 6$ hrs, shown at bottom) positions of the drifters shown in Figure D.1. The LCS for these times are not shown so the drifters may be clearly visualized.