LCS Tool: A computational platform for Lagrangian coherent structures

K. Onu, F. Huhn, G. Haller

Institute of Mechanical Systems, Department of Mechanical and Process Engineering, ETH Zurich, Switzerland

ARTICLE INFO

Article history:
Received 13 June 2014
Received in revised form 17 December 2014
Accepted 29 December 2014
Available online 7 January 2015

PACS:
47.10.Fg
47.27.De
47.11.--j
05.45.--a
47.51.+a
47.54.--r
92.10.Hf
93.30.Qn
92.10.ab
92.10.Ty
92.10.ak
92.10.A--

MSC:
37N10
37B25
97N80
37G60
37B55
70H33
76F25

Keywords:
Lagrangian coherent structures
Mathematical software
Non-autonomous dynamical systems
Invariant manifolds
Mixing
Transport barriers
Fluid dynamics
Ocean surface flows

ABSTRACT

We give an algorithmic introduction to Lagrangian coherent structures (LCSs) using a newly developed computational engine, LCS Tool. LCSs are most repelling, attracting and shearing material lines that form the centrepieces of observed tracer patterns in two-dimensional unsteady dynamical systems. LCS Tool implements the latest geodesic theory of LCSs for two-dimensional flows, uncovering key transport barriers in unsteady flow velocity data as explicit solutions of differential equations. LCS Tool makes theoretical results accessible to the fluid mechanics community since implementing these results directly could be time consuming. After a review of the underlying theory, we explain the steps and numerical methods used by LCS Tool, and illustrate its capabilities on three unsteady fluid flow examples.

1. Introduction

Lagrangian coherent structures (LCSs) are evolving organizing centres of trajectory patterns in non-autonomous dynamical systems [1–3]. Applications of LCSs include oceanic and atmospheric flows [4,5], biological transport problems [6–8], aeronautics [9], and ocean surface flows [10].
computational engine called LCS Tool. Haller [13] proposed that ridges of the finite-time Lyapunov exponent (FTLE) detection is heuristic indicators of hyperbolic (i.e., repelling and attracting type) LCSs. A number of examples support this principle [2]. Equating FTLE ridges with LCSs, however, would create theoretical inconsistencies, as well as false positives and negatives in hyperbolic LCS detection [14,15]. In addition, the role of the FTLE field in the accurate detection of elliptic (vortex-type) and parabolic (jet-core type) LCSs has remained an open question (but see [16]).

More recent work has focused on an exact mathematical formulation of the properties defining LCSs [14,17–22]. In two-dimensional flows, hyperbolic and parabolic LCSs turn out to be stationary curves of the averaged material shear [19,20]. These variational formulations lead to explicit solutions for LCSs as null geodesics of appropriate Lorentzian metrics.

We present an algorithmic introduction to geodesic LCS detection. We then review the implementation of this approach in a computational engine called LCS Tool. This engine is a library of MATLAB functions that extract LCSs from two-dimensional unsteady flows. The examples we present form demonstration scripts provided with LCS Tool.

Other software for LCS detection exists. ManGen [24] calculates the FTLE and advects material curves in two-dimensional velocity fields. It includes a graphical user interface and uses the Message Passing Interface standard for parallel calculations. Newman [25] calculates the FTLE using dimension independent code. It assists ridge extraction of FTLE fields and supports analytic and dataset velocity definitions. FlowTK [26] calculates the FTLE in two and three-dimensions on Cartesian and unstructured grids. The NVIDIA CUDA parallel computing platform is used for fast computations and Kitware’s ParaView data analysis and visualization application is used for a user interface.

These packages generate FTLE plots to aid the visual assessment of hyperbolic LCSs. LCS Tool also has functions to generate FTLE plots, but its emphasis is to provide geodesic extraction of LCSs as parametrized material curves, and extend the scope of such extraction to elliptic LCSs.

2. Theory

We consider two-dimensional, finite-time, unsteady velocity fields of the form

$$\frac{dx}{dt} = \mathbf{v}(x,t), \quad x \in U \subset \mathbb{R}^2, \quad t \in [t_0, t_f].$$

(Trajectories of Eq. (1) are denoted \(\mathbf{x}(t; x_0, x_0)\), with \(x_0 \in U\) denoting their initial position in the open set \(U\) at an initial time \(t_0 \in [t_0, t_f]\). The flow map is then defined as \(F_{t_0}^t(x_0) = \mathbf{x}(t; x_0, x_0)\), mapping initial positions to current positions at time \(t\). The time interval \([t_0, t_f]\) is part of the definition of the finite-time dynamical system in Eq. (1). This interval may be a time scale of interest or the maximum interval over which velocity data is available from simulations or observations.

The right Cauchy-Green strain tensor associated with the flow map is defined as

$$C_{t_0}^t(x_0) = \left[ \nabla F_{t_0}^t(x_0) \right] \nabla F_{t_0}^t(x_0).$$

measuring Lagrangian strain in the velocity field. This tensor is symmetric and positive definite [27]. We label the eigenvalues and eigenvectors of \(C_{t_0}^t(x_0)\) as follows:

$$\lambda_i = \xi_i \xi_i^T, \quad 0 < \lambda_1 \leq \lambda_2, \quad i = 1, 2;$$

$$\left| \xi_i \right| = 1, \quad \Omega \xi_i = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}. \quad (3)$$

2.1. Elliptic LCSs

We seek positions of closed material lines at time \(t_0\) that prevail as coherent Lagrangian vortex boundaries (or elliptic LCSs) over a time interval \([t_0, t] \subset [t_0, t_f]\). Haller and Beron-Vera [19,20] argue that such initial material line positions are closed stationary curves of the averaged strain functional

$$Q(\gamma) = \frac{1}{\sigma} \int_0^\sigma \frac{\sqrt{\left\langle r'(s), C_{t_0}^t(r(s)) r'(s) \right\rangle}}{\sqrt{\left\langle r'(s), r'(s) \right\rangle}} \, ds,$$

obtained by averaging the tangential strain arising over \([t_0, t]\) along closed material lines parametrized as \(r(s)\) with \(s \in [0, \sigma]\). Solutions to this variational problem turn out to be closed orbits of one of the two parametrized vector-field families

$$\eta^*_s = \frac{\lambda_2 - \lambda^2}{\lambda_2 - \lambda_1} \xi_1 \pm \sqrt{\frac{\lambda_2 - \lambda_1}{\lambda_2 - \lambda_1}} \xi_2,$$

which coincide with null geodesics of the Lorentzian metric family

$$e_s(u, v) = \langle u, D_{t_0}^t r(\sigma) \rangle,$$

for this reason, we refer to the computation of elliptic LCSs as limit cycles of Eq. (5) as geodesic detection of elliptic LCSs.

Any orbit of Eq. (5) turns out to stretch uniformly under the flow map \(F_{t_0}^t\). Specifically, any subset of an orbit of Eq. (5) increases its arc length precisely by a factor of \(\lambda\). For this reason, we refer to trajectories of Eq. (5) as \(\lambda\)-lines. Following Haller and Beron-Vera [19,20], we call the outermost member of a closed family of \(\lambda\)-lines a coherent Lagrangian vortex boundary.

2.2. Hyperbolic LCSs

Next we consider positions of material lines at time \(t_0\) that prevail as most repelling or attracting material lines (or hyperbolic LCSs) over a time interval \([t_0, t] \subset [t_0, t_f]\). Farazmand et al. [23] argue that hyperbolic LCSs are stationary curves of the averaged shear functional

$$Q(\gamma) = \frac{1}{\sigma} \int_0^\sigma \frac{\left\langle r'(s), D_{t_0}^t C_{t_0}^t(r(s)) r'(s) \right\rangle}{\sqrt{\left\langle r'(s), D_{t_0}^t C_{t_0}^t(r(s)) r'(s) \right\rangle}} \, ds,$$

obtained by averaging the Lagrangian shear arising over \([t_0, t]\) along material lines parametrized as \(r(s)\) with \(s \in [0, \sigma]\). More precisely, hyperbolic LCSs are stationary curves of \(Q(\gamma)\) with respect to fixed-endpoint perturbations. We note that parabolic LCSs (Lagrangian jet cores) are also stationary curves of \(Q(\gamma)\), but under variable endpoint perturbations (cf. Farazmand et al. [23]).
Solutions to this variational problem turn out to be curves of the \( \xi_1 \) or \( \xi_2 \) eigenvector field. Repelling LCSs (shrink lines) are obtained as trajectories of the differential equation
\[
r' = \xi_1(r),
\]
and attracting LCSs (stretch lines) are obtained as trajectories of the differential equation
\[
r' = \xi_2(r),
\]
(6) (7)

Shrink lines and stretch lines coincide with the null geodesics of the Lorentzian metric \( h(u, v) = \langle u, D_v r \rangle \). For this reason, we refer to the computation of hyperbolic LCSs as strongest normally-repelling or normally-attracting curves of Eq. (5) as geodesic detection of hyperbolic LCSs.

To compute the normal repulsion of shrink lines, we note that an infinitesimal normal perturbation to a shrink line \( \gamma \) at its point \( r \) grows under the flow map \( F_t \) by a factor of \( \lambda_2(r) \) in the direction normal to \( F'_t(\gamma) \). Similarly, small normal perturbations to a stretch line decay by a factor \( \lambda_1(r) \) in the direction normal to the evolving stretch line.

3. Numerical methods

This section describes stepwise the numerical implementation of geodesic detection of elliptic and hyperbolic LCSs based on Eqs. 5, 6, 7. Table 1 gives an overview of the steps, functions, and variable names used in LCS Tool to analyse a flow.

3.1. Computing the invariants of the Cauchy-Green strain tensor

The first step in calculating elliptic and hyperbolic LCSs is the computation of the Cauchy-Green strain tensor field \( C_g(x_0) \), as defined in Eq. (2). The function performing this calculation in LCS Tool is \( \text{eig}_{\text{cgStrain}} \). The main steps executed by this function are enumerated in Table 2, while Table 3 summarizes the syntax of \( \text{eig}_{\text{cgStrain}} \).

The Cartesian grid of initial conditions mentioned in Table 2 is rectangular, with user-defined vertical and horizontal ranges and resolutions. The optimal resolution may be determined by a successive doubling of the initial resolution until convergence of the extracted LCSs is observed visually. If the domain of interest comprises only a few expected LCSs, e.g., one vortex, then a resolution of about 500 grid points along the longest axis usually gives good results. Otherwise, a higher resolution must be chosen.

The auxiliary grid (cf. Table 2) comprises four points placed symmetrically around each point of the Cartesian grid (Fig. 1). These points are used to achieve increased accuracy in the finite-difference approximation
\[
\nabla F^r_{t_0}(x_0) \approx \frac{\alpha_{11} \alpha_{12} \alpha_{21} \alpha_{22}}{2/|\Delta x|} \quad \text{of } \nabla F^r_{t_0} \text{ at a point } x_0 \text{ of the Cartesian grid. Here } \delta x \text{ is a vector of length } |\delta x| > 0 \text{ that points from the Cartesian grid-point } x_{0,j} \text{ in the } j\text{th coordinate direction (Fig. 1). Computational improvements arising from the use of the auxiliary grid over simply using the nearest points of the main grid were reported in Farazmand and Haller \[17\]. Experience suggests setting the auxiliary grid spacing to 1–10% of the main grid spacing.}

The function \( \text{eig}_{\text{cgStrain}} \) of LCS Tool provides the option to calculate Cauchy-Green eigenvectors from the auxiliary grid using eigenvalues calculated from the main grid. We have found that for analytic flows, the eigenvalues can be calculated from the main grid. For dataset flows, using the auxiliary grid for eigenvalue calculations gives better results.

As stated above, a typical main grid for the Cauchy-Green strain tensor has 500 × 500 points. This means that after the addition of 4
Results are from the function `ode45` called by the function
`poincare` to perform trajectory integration from all grid
points simultaneously. This calculation typically takes 5–10 min.

A potential drawback of vector form integration is that memory
use associated with vector integration at different resolutions.

The Cauchy-Green strain tensor can have point singularities,
generically arises at a finite set of isolated points within the compu-
tation errors. By setting the integration tolerances to smaller values,
the number of such grid points is reduced. Enforcing $\lambda_2 < 1$
elsewhere, however, can incur excessive computational cost. To this
end, the function `eigCGstrain` records the number of points with
$\lambda_2 < 1$, providing a measure for setting feasible integration toler-
ances.

### 3.1.1. Special case: incompressible velocity fields

Incompressible flows (i.e. those satisfying $\nabla \cdot v = 0$) satisfy the relation $\lambda_1 (x_0) \lambda_2 (x_0) = 1$ at all points of the computational domain [29]. Incompressibility can be computationally imposed by first calculating $\lambda_2 (x_0)$, then setting $\lambda_1 (x_0) = 1/\lambda_2 (x_0)$ and calculating the strain eigenvectors $\xi_2$ from $\lambda_2$, then $\xi_1$ from the relationship in Eq. (3). Experience shows that computing $\lambda_1$ in this order gives higher accuracy than in the reverse order [17].

At some grid points, $\lambda_2 < 1$ may occur due to numerical integration errors. By setting the integration tolerances to smaller values, the number of such grid points is reduced. Enforcing $\lambda_2 > 1$ everywhere, however, can incur excessive computational cost. To this end, the function `eigCGstrain` records the number of points with $\lambda_2 < 1$, providing a measure for setting feasible integration tolerances.

### 3.1.2. Special case: dataset velocity fields

Velocity fields defined by datasets require preprocessing before they are used in the numerical integration of Eq. (1). This requires spatial and temporal interpolation that enables the evaluation of the velocity function at arbitrary points in $U$ and at arbitrary times between $t_-$ and $t_+$ in Section 4.3, we present an ocean dataset example with details of possible interpolation functions.

### 3.2. Computing elliptic LCSs

As discussed in Section 2.1, positions of elliptic LCSs at time $t_0$ are found as closed orbits of the $n_x^+$ vector fields defined in Eq. (4). We find such orbits by integrating (5) from points of an appropriately chosen section (Poincare section), and evaluating the first return map (Poincare map) onto this section. A $\lambda$-line returning to its starting point is then an elliptic LCS. The outermost member of a family of closed $\lambda$-lines (obtained by varying $\lambda$) is a coherent Lagrangian vortex boundary [19,20].

The main steps in calculating elliptic LCSs are enumerated in Table 5 and described in further detail below. The syntax of elliptic LCS functions in LCS Tool is shown in Table 6.

The first step is to define the position of Poincare sections in regions where closed $\lambda$-lines are expected based on a visual analysis of the orbit structure of the $n_x^+$ vector field. The Poincare section is to be oriented such that the first endpoint is close to the centre of the
expected Lagrangian vortex, and the second endpoint is outside this vortex. Additionally, the number of lambda-lines launched from the Poincare section, `poincareSection.numPoints`, must be defined. A reasonable default value is 100.

The second step is to integrate the lambda-lines starting from the Poincare section to obtain the corresponding Poincare map. Integration of lambda-lines is performed using the \( n_x^\perp \) vector fields defined in Eq. (4) over the main grid. The underlying eigenvector fields, \( \xi_i(x_0) \), have generic but removable orientation discontinuities, which require monitoring and local reorientation. This process is sketched in Table 7 and illustrated in Fig. 2. Linear interpolation is used in the interpolation of \( n_x^\perp \) within a grid element, since using higher-order interpolation would necessitate verifying that there are no orientation discontinuities beyond the four nearest grid points. We identify orientation discontinuities by checking the inner product of the \( n_x^\perp \) vectors at adjacent grid points. Rotations exceeding 90° between two such neighbouring vectors are classified as orientation discontinuities and are corrected before linear interpolation. When setting the Cauchy-Green strain tensor main grid resolution, one may find it helpful to calculate a histogram of eigenvector field rotations to ensure that all rotations are well below 90° or almost 180°.

An example of a Poincare map produced from integration of the \( n_x^\perp \) field is shown in Fig. 3. Most orbits will return to the Poincare section and their integration will then be stopped using the ordinary differential equation event detection function of MATLAB. Some orbits may, however, deviate far from the Poincare section and do not return for any reasonable integration time. To control this behaviour, we specify a maximum orbit length, `poincareSection.orbitMaxLength`. In practice, viewing the Poincare section as the radius of a circle and setting the maximum lambda-line integration length to twice the circumference gives good results.

In Fig. 3, circle markers indicate fixed points of the Poincare section and their integration will then be stopped using the ordinary differential equation event detection function of MATLAB. Some orbits may, however, deviate far from the Poincare section and do not return for any reasonable integration time. To control this behaviour, we specify a maximum orbit length, `poincareSection.orbitMaxLength`. In practice, viewing the Poincare section as the radius of a circle and setting the maximum lambda-line integration length to twice the circumference gives good results.

As discussed in Section 2.2, hyperbolic LCS positions at time \( t_0 \) are found as the strongest repelling curves of the vector field Eq. (6) (repelling LCSs), and strongest attracting curves of the vector field Eq. (7) (attracting LCSs). By repulsion and attraction we mean a property of the LCS (as an evolving material line) under the flow map \( F_{t_0} \). We identify the strongest repelling shrink lines as those crossing a local maximum of the \( \lambda_2(x_0) \) field. Similarly, we identify the strongest attracting stretch lines as those crossing a local minimum of the \( \lambda_1(x_0) \) field. These local maxima and minima of the appropriate \( \lambda_2(x_0) \) or \( \lambda_1(x_0) \) can be thought of as the extensions of the concept of saddle points to the present finite-time, temporally aperiodic flow setting.

The main steps of hyperbolic LCS discovery are enumerated in Table 8. The core function to compute hyperbolic LCSs is `seed_curves_from_lambda_max` and its syntax is given in Table 9.

### 3.3. Computing hyperbolic LCSs

As discussed in Section 2.2, hyperbolic LCS positions at time \( t_0 \) are found as the strongest repelling curves of the vector field Eq. (6) (repelling LCSs), and strongest attracting curves of the vector field Eq. (7) (attracting LCSs). By repulsion and attraction we mean a property of the LCS (as an evolving material line) under the flow map \( F_{t_0} \). We identify the strongest repelling shrink lines as those crossing a local maximum of the \( \lambda_2(x_0) \) field. Similarly, we identify the strongest attracting stretch lines as those crossing a local minimum of the \( \lambda_1(x_0) \) field. These local maxima and minima of the appropriate \( \lambda_2(x_0) \) or \( \lambda_1(x_0) \) eigenvalue field can be thought of as the extensions of the concept of saddle points to the present finite-time, temporally aperiodic flow setting.

The main steps of hyperbolic LCS detection are enumerated in Table 8. The core function to compute hyperbolic LCSs is `seed_curves_from_lambda_max` and its syntax is given in Table 9.

### 4. Examples

This section presents the use of LCS Tool in three examples: a double gyre, a jet, and an oceanic geostrophic flow. The examples are available as scripts in the demo folder of LCS Tool. Executing these scripts demonstrates how to call LCS Tool functions. These functions are written to achieve balance between ease of use and performance. Examples of demo runtimes are given in Table 10.

#### 4.1. Double gyre

The double gyre is a model for a time-dependent two gyre system observed in geophysical flows [30]. The model consists of two counter rotating sinusoidal vortices with a harmonically

---

**Table 7**

<table>
<thead>
<tr>
<th>Algorithm used for variable time step integration of ( \lambda )-lines.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Linearly interpolate vector field orientation at initial position.</td>
</tr>
<tr>
<td>2. At next position, check whether vector field has rotated by over 90°, if yes, flip the vector field orientation by 180°.</td>
</tr>
<tr>
<td>3. Stop integration when ( \lambda )-line returns to Poincare section, ( \lambda )-line reaches the domain boundary, or maximum integration length has been reached.</td>
</tr>
</tbody>
</table>
Table 8
Algorithm to calculate initial positions of repelling LCSs at time $t_0$. The algorithm for attracting LCSs is similar.

1. Define a local maximization distance.
2. Find all points of the main grid that are local maxima of $\lambda_2^{(x)}$ within a circle whose radius is the local maximization distance.
3. Define a maximum shrink line length.
4. Integrate a shrink line forward and backward according to Eq. (6) and using the largest $\lambda_2$ local maximum as the initial position. Integrate until the shrink line has attained the maximum shrink line length, or until it has reached the domain boundary.
5. Flag any remaining local maxima of $\lambda_2^{(x)}$ within the maximization distance of the shrink line as ineligible initial positions for subsequent shrink lines.
6. Continue integrating shrink lines using local maxima of $\lambda_2^{(x)}$ as initial positions until no eligible local maxima of $\lambda_2^{(x)}$ remain.
7. Remove all shrink line segments within elliptic LCSs.

Table 9
Syntax of the function seed_curves_from_lambda_max.

```
[curvePosition, curveInitialPosition] = seed_curves_from_lambda_max(distance, cgEigenvalue, cgEigenvector, flowDomain, flowResolution)
```

distance threshold distance for placement of $\lambda_2^{(x)}$ maxima
cgEigenvalue array of Cauchy-Green strain eigenvalues
cgEigenvector array of Cauchy-Green strain eigenvectors
flowDomain $2 \times 2$ array to define flow domain
flowResolution $1 \times 2$ array to define Cauchy-Green strain main grid resolution
periodicBc optional $1 \times 2$ logical array to specify periodic boundary conditions. Default: [false, false].
numel(cgEigenvalue).

odeSolverOptions optional odeoptions structure to specify flow map integration parameters

Table 10
Runtime of some demo scripts. Results obtained with an Intel Core i3-4130 3.4 GHz processor with 8 GB of memory.

<table>
<thead>
<tr>
<th>Demo</th>
<th>Runtime (min)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Double gyre</td>
<td>50</td>
</tr>
<tr>
<td>Bickley jet</td>
<td>25</td>
</tr>
<tr>
<td>Ocean dataset</td>
<td>127</td>
</tr>
</tbody>
</table>

Listing 1
Double gyre derivative function corresponding to Eq. (8).

```
function derivative = derivative(t,x,e,A,omega)

dx/dt = -\pi A \sin[\pi f(x,t)] \cos(\pi y),
\text{Eq. (8)}

dy/dt = \pi A \cos[\pi f(x,t)] \sin(\pi y) \frac{df(x,t)}{dx},

f(x,t) = e \sin(\omega t)x^2 + [1 - 2e \sin(\omega t)]x.
```

The MATLAB function describing this velocity field is given in Listing 1, specifying the right hand side of the particle ordinary differential equation in a way that supports vectorized integration.

In what follows, the parameter values are: $A = 0.1, e = 0.1, \omega = \pi/5$. The flow timespan is $t \in [0, 10]$ and the domain is $x \in [0, 2], y \in [0, 1]$. By examining the FTLE field (which LCS Tool can provide), we position Poincare sections to capture elliptic LCSs. A script to perform this operation is given in Listing 2 where two Poincare sections are defined (lines 15 and 16). The free stretching parameter $\lambda$ (Eq. (5)) is varied over the range [0.93, 1.07] with increments of 0.01 (line 18). Closed orbits for all predefined $\lambda$ values are computed and the outermost closed orbit is kept as the Lagrangian vortex boundary.
Fig. 4. Convergence of closed $\lambda$-lines for increasing main-grid resolution in the double-gyre. The outermost closed $\lambda$-line (bold green line) is the vortex boundary. Top: $500 \times 250$, middle: $750 \times 375$, bottom: $1000 \times 500$. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

In Fig. 4, the resolution of the Cauchy–Green strain tensor is varied from $500 \times 250$ to $1000 \times 500$. The location of the outermost closed $\lambda$-line changes insignificantly, demonstrating convergence. For all tested resolutions, the $\lambda$ values of the outermost closed orbits match. This also suggests that the lowest tested resolution, $500 \times 250$, is sufficient to identify elliptic LCSs.

Fig. 5 shows elliptic and hyperbolic LCSs for the double gyre at this resolution. Listing 3 gives the LCS Tool commands to compute hyperbolic LCSs. The maximum length of shrink lines and stretch lines, shrinkLineMaxLength and stretchLineMaxLength, is set to 20, a multiple of the domain size since the hyperbolic LCSs may wind around vortices several times.

As usual, the local maximization distance is set larger for stretch lines than for shrink lines (cf. line 3 and line 7). The purpose of the maximization distance is to obtain spatially separated LCSs and to avoid a dense tangle of lines that basically indicates the same hyperbolic LCS (cf. Table 8, item 5). Setting the local maximization distance for stretch lines larger than for shrink lines allows obtaining a comparable number of stretch lines and shrink lines overall in the flow domain (recall that hyperbolic LCS seed points are discarded if they are within the local maximization distance of an existing hyperbolic LCS). Shrink lines are locally tangent to ridges of $\lambda_2$ maxima, whereas stretch lines are locally normal to these ridges. Setting the local maximization distance of stretch lines and shrink lines equal would therefore produce a greater number of stretch lines than shrink lines.

4.2. Bickley jet

The Bickley jet models a meandering zonal jet flanked above and below by counter rotating vortices. This is an idealized model of geophysical flows such as the Gulf Stream and the polar night jet perturbed by a Rossby wave [31,16].
The velocity is given by \( \mathbf{v}(x,y,t) = (\partial_x \psi, \partial_y \psi) \) where
\[
\psi(x,y,t) = \psi_0(x,y) + \psi_1(x,y,t),
\]
\[
\psi_0(x,y) = c_3 y - U_L \tanh \left( \frac{y}{L_y} \right) + \varepsilon_2 U_L y \sech^2 \left( \frac{y}{L_y} \right) \cos 3x,
\]
\[
\psi_1(x,y,t) = U_L y \sech^2 \left( \frac{y}{L_y} \right) \sum_{n=1}^{2} e^{i \Omega n t} \cos \omega_n t.
\]

As a forcing function, we choose a solution running on the chaotic attractor of the damped and forced Duffing oscillator, specifically
\[
\frac{d\phi_1}{dt} = \phi_2,
\]
\[
\frac{d\phi_2}{dt} = -0.1 \phi_2 - \phi_1^3 + 11 \cos(t),
\]
\[
f_{1,2}(t) = 2.625 \times 10^{-2} \phi_1(t/6.238 \times 10^5)
\]

The parameter values we use are: \( U = 62.66, c_2 = 0.205U, c_3 = 0.461U, L_y = 1.77 \times 10^6, \varepsilon_1 = 0.0075, \varepsilon_2 = 0.04, \varepsilon_3 = 0.3, L_x = 6.371 \times 10^8 \pi, \kappa_n = 2n \pi / L_x, \sigma_1 = 0.5k_2(c_2 - c_3), \sigma_2 = 2.2k_2. \)

The integration time is \( T = 4\lambda/\omega \), an integer multiple of the eddy turnover time. Listing 4 shows the LCS Tool commands in which the chaotically perturbed velocity is defined (line 9), periodic boundary conditions are imposed in the x-direction (line 10), and five Poincare sections are defined where we expect coherent vortices (lines 26–30). \( \lambda \)-values for closed orbit detection are varied over the range \([0.80, 1.20]\) with a step of 0.01.

Fig. 6 shows elliptic and hyperbolic LCSs of the Bickley jet with the FTLE in the background.

### 4.3. Ocean velocity data from satellite altimetry

The final example demonstrates the use of LCS Tool on velocity data derived from satellite-observed sea-surface heights under the geostrophic approximation. In contrast to the previous two analytic examples, the velocity field is available only with discrete temporal and spatial resolution. Our region of interest is a small domain in the South Atlantic Ocean, where exceptionally coherent eddies, Agulhas rings, were recently found by Haller and Beron-Vera [19,20] using the theory we surveyed in Section 2.1.

In the geostrophic approximation, sea surface height (\( \eta \)) serves as a stream-function for surface velocities. In a longitude-latitude (\( \phi, \theta \)) coordinate system, the evolution of a fluid particle is given by
\[
\frac{d\eta(\phi, \theta, t)}{dt} = -\frac{g}{R^2 f(\theta) \cos \theta} \frac{\partial \eta(\phi, \theta, t)}{\partial \phi}
\]
\[
\frac{d\phi(\phi, \theta, t)}{dt} = \frac{g}{R^2 f(\theta) \cos \theta} \frac{\partial \eta(\phi, \theta, t)}{\partial \theta}
\]

where \( g \) is the constant of gravity, \( R \) is the mean radius of the Earth, and \( f(\theta) = 2\Omega \sin \theta \) is the Coriolis parameter, with \( \Omega \) denoting the Earth’s mean angular velocity.
The data is given at a spatial resolution of 1/4° and a temporal resolution of 7 days. Due to the discrete data, defining the right hand side of Eqs. (9) and (10) involves spline interpolation in space and time. An interpolant is generated first, then the function `flowdata_derivative` evaluates the interpolants for the zonal and meridional velocity at the needed coordinates. Listing 5 shows the relevant part of the code in LCS Tool’s ocean demo file `demo/ocean_dataset/elliptic_hyperbolic_lcs.m`. The commands for the interpolation of the velocity data are given in lines 9–12.

We choose the integration time as $T = 30$ days (listing 5, line 4), which is larger than the eddy turnover time in this region. The resolution of the main computational grid for initial conditions is set to 400 × 400 (line 3). This corresponds to a resolution of roughly 0.015°. With this choice, the resolution of the tracer grid is 15 times higher than the resolution of the velocity field. The flow is integrated and the Cauchy-Green strain tensor is computed by the function `eig_cgsStrain` (line 35). The auxiliary grid distance is set to 1% of the main grid distance (line 17), and eigenvalues are computed from the auxiliary grid (line 16). Elliptic LCSs are computed in line 37, after the Poincaré sections have been set (line 21). $\lambda$-values are varied over a range of [0.90, 1.10] with a step of 0.02 (line 23). Hyperbolic LCSs are computed in lines 40 and 43.

Fig. 7 shows elliptic and hyperbolic LCSs on 22 November 2006, the same time as analysed in Haller and Beron-Vera [19,20], Beron-Vera et al. [4]. In those references, the integration time is 90 days. We use a shorter time to avoid tangling hyperbolic LCSs. Our analysis via LCS Tool reveals five coherent eddies. The largest, at $(3, -32)$, has a non-stretching boundary, i.e., $\lambda = 1$. It corresponds to eddy #2 in figure 3 of Beron-Vera et al. [4]. Four additional smaller coherent eddies are found. They do not stay coherent over 90 days. The hyperbolic LCSs determine the deformation of the fluid between coherent eddies.

5. Conclusions

We have described a computational toolbox, LCS Tool, that implements recent variational results for Lagrangian coherent...
structures (LCSs) in two-dimensional unsteady flows. We have also demonstrated the performance of LCS Tool on two analytic flow models and a geophysical velocity dataset. The publicly available software library producing these results enables the exploration of variational LCS methods without assuming a detailed knowledge of geodesic LCS theory.

LCS Tool leverages the capabilities of MATLAB. For FTE based extraction of LCSs, computational performance has received considerable attention [32,33]. We think LCS Tool can facilitate similar computational advances for variational LCS methods. Optimizing computational performance will aid applications to large-scale forecasting applications, such as the tracking of environmental contaminants [34].

We think LCS Tool can serve as a foundation for the numerical implementation of recent theoretical advances. These include the geodesic theory of parabolic LCSs (jet cores) [23] and the variational theory of hyperbolic and elliptic LCSs for three-dimensional flows [22].

Acknowledgements

The altimeter products used in this work are produced by SSALTO/DUACS and distributed by AVISO, with support from CNES, http://www.aviso.oceanobs.com/duacs.

References

Florian Huhn was a postdoc researcher at the Institute of Mechanical Systems at ETH Zurich, Switzerland. He is now at the Department of Experimental Methods, Institute of Aerodynamics and Flow Technology, German Aerospace Center (DLR), Göttingen, Germany. His main research interests are experimental fluid dynamics, reaction-diffusion advection systems, and transport in geophysical flows.

George Haller is Professor of Nonlinear Dynamics at ETH Zürich. He develops dynamical systems methods to handle complex problems in mechanics and fluid dynamics.