Detecting coherent structures using braids

Michael R. Allshouse*

Department of Mechanical Engineering, MIT, Cambridge, MA 02139, USA

Jean-Luc Thiffeault†

Department of Mathematics, University of Wisconsin, Madison, WI 53706, USA

(Dated: 11 June 2011)

The detection of coherent structures is an important problem in fluid dynamics, particularly in geophysical applications. For instance, knowledge of how regions of fluid are isolated from each other allows prediction of the ultimate fate of oil spills. Existing methods detect Lagrangian coherent structures, which are barriers to transport, by examining the stretching field as given by finite-time Lyapunov exponents. These methods are very effective when the velocity field is well-determined, but in many applications only a small number of flow trajectories are known, for example when dealing with oceanic float data. We introduce a topological method for detecting invariant regions based on a small set of trajectories. In the method we regard the two-dimensional trajectory data as a braid in three dimensions, with time being the third coordinate. Invariant regions then correspond to trajectories that travel together and do not entangle other trajectories. We detect these regions by examining the growth of hypothetical loops surrounding sets of trajectories, and searching for loops that show negligible growth.

Keywords: topological chaos, dynamical systems, Lagrangian coherent structures

I. INTRODUCTION

The ability to accurately identify regions of mixing and barriers to transport in two-dimensional systems has applications in the ocean [1, 2], the atmosphere [3, 4], mantle flows [5], as well as granular flows [6]. Central to this is the concept of Lagrangian coherent structures, which are barriers to transport between different dynamical regions in a flow. Thus, fluid in a region delimited by Lagrangian coherent structures does not mix well with the surrounding fluid. These structures can be impossible to detect in the Eulerian (fixed) frame, since they move around and change their shape, possibly in an irregular manner. Over the last decade, new methods were developed to find Lagrangian coherent structures [2, 7–13]. All of these methods require field data (*e.g.* velocity or vorticity fields) which in many practical situations is not readily accessible. Instead, the data available is often in the form of *tracer trajectories*, either from oceanic floats or atmospheric balloons. Though it may contain the trajectory of dozens of floats, such data is too sparse to reconstruct a velocity field. A method which estimates the location of Lagrangian coherent structures based on trajectory data is thus highly desirable.

* allshouse@mit.edu

† jeanluc@math.wisc.edu

A first step towards this was taken by Thiffeault [14, 15], who regarded a set of two-dimensional float trajectories as a *braid* in three dimensions, where the third dimension is time. This is a well-established technique for studying two-dimensional trajectories [14–24]. Thiffeault [15] applied recently-developed mathematical techniques [25] to the rapid computation of the *topological entropy* of trajectory datasets. The topological entropy is related to the degree of *entanglement* of trajectories: a set of trajectories trapped in a vortex, for instance, would have zero topological entropy. A measure of chaos, and hence of mixing, was thus obtained without requiring the full velocity field. Finn and Thiffeault [26] showed numerically that as the number of trajectories is increased, the topological entropy associated with the trajectories approaches the true topological entropy of the underlying flow [27–30].

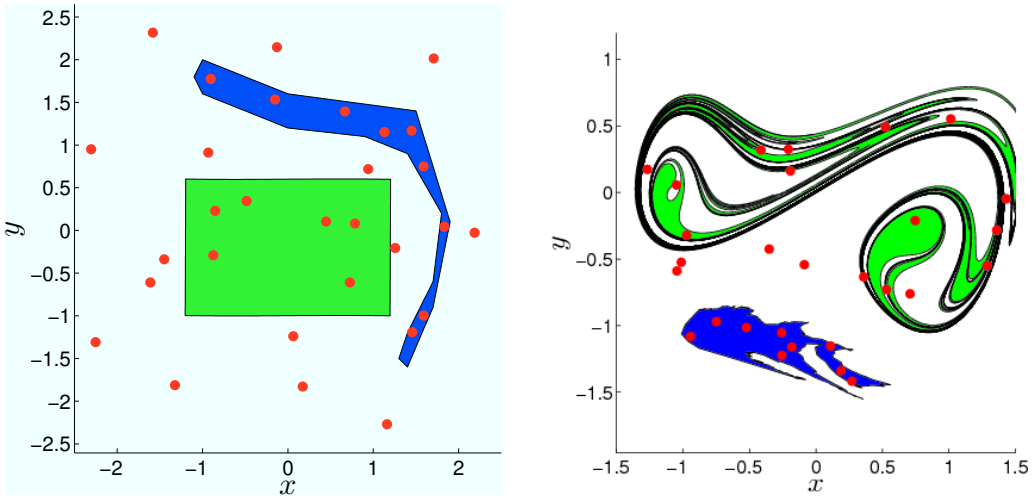
In the present paper, we implement the next step — the detection of invariant regions based on sparse particle trajectory data. These regions contain fluid that possibly mixes well with other fluid within the region, but not with fluid outside. The regions can move and change shape aperiodically, and each region has its own topological entropy.

If a set of trajectories are within such an invariant region, then from the braid perspective they form a 'coherent bundle.' A coherent bundle can be thought of as rope, which is made up of small strands (the individual trajectories). We will detect such bundles in the following manner. Consider an imaginary material loop drawn around a bundle of trajectories. We say that the loop is 'pulled-tight' if it is tightened to its shortest length that still encloses the trajectories (see Fig. 3). If the bundle really is inside an invariant region, then the length of the pulled-tight loop does not grow rapidly in time. On the other hand, if any trajectory is outside of a region, then typically the chaotic action of the flow will cause this loop to grow exponentially, even if it is pulled-tight.

What allows this calculation to be practical is that from the topological perspective we do not actually need to follow true material loops in the fluid. Rather, there is a symbolic description of pulled-tight loops that allows a very rapid computation of their evolution, in particular the growth rate of their length. These are relatively recent topological tools which are only now being applied to physical problems [15, 25].

We must emphasize that the detection method presented here is not perfect: like all finite-time approaches, it is susceptible to the trajectory data being too short, or too sparse — for instance, it is not possible to reliably detect an invariant region based on a single trajectory. But when relatively few trajectories are known, our approach comes close to yielding as much topological information as is theoretically available.

Throughout this paper, we use a modified version of the Duffing oscillator to test and illustrate the features of the method. A typical time-evolution of this model system is depicted in Fig. 1: Fig. 1a shows 30 trajectories as well as two regions of interest (blue and green) in an initial state. These are shown at a later time in Fig. 1b. While the boundaries of the two regions are distorted, the trajectories that start within a region remain in that region for all time. The blue region has changed shape, but its pulled-tight perimeter has remained approximately the same length. This indicates that the boundary of the blue region would be classified as a potential transport barrier (Lagrangian coherent structure). The green region shows substantial folding and stretching of its boundary as trajectories from outside crisscross the region. The boundary of the green region thus tends to grow exponentially, even if it is pulled-tight around the trajectories. The boundary of the green region would then not be considered a transport barrier. In this paper, we will show how to automate the detection of loops which do not grow rapidly, such as the one enclosing the



(a) (b)

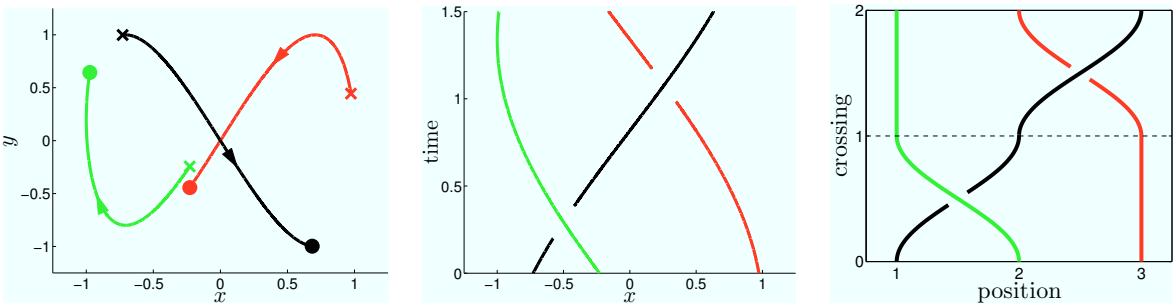
FIG. 1: (a) Initial state and (b) state after evolution by the Duffing system of Section III C. The blue region is surrounded by a transport barrier, while the green region is not. Particle positions are shown as red dots.

blue region.

The paper is divided as follows. In Section II, we introduce the necessary tools from braid theory and topological surface dynamics. We then demonstrate how to use these tools to detect invariant regions in Section III, using a direct approach of searching over a large set of loops. In Section IV, we refine the method to make it much quicker, by focusing on loops enclosing pairs of trajectories. Both the direct and the improved method are tested on the modified Duffing oscillator. We test the refined approach on a two-dimensional rod-stirring device in Section V, and offer some conclusions in Section VI. Because this paper contains some terminology that may not be familiar to many readers, we have included a Glossary in an appendix.

II. BRAID THEORY AND TOPOLOGICAL SURFACE DYNAMICS

In this section, we present a number of definitions and tools from braid theory and topological dynamics. These will be kept to a minimum, and we give references containing a more complete treatment. Specifically, we focus on applying the theory to a set of trajectories evolving in time on a two-dimensional plane. As a visual example, a sample set of three particle trajectories is shown in Fig. 2a. The trajectories are constrained to two dimensions in physical space. The particles are often referred to as *punctures*, since they are regarded as topological obstacles analogous to small holes in the plane. For example, there are three punctures in Fig. 2a which are represented by dots. The initial positions of the punctures are shown as crosses.



(a) (b) (c)

FIG. 2: (a) Three sample trajectories in the physical plane. A cross represents the initial position of the trajectory and a dot the final position. (b) The projection of the trajectories onto the x-axis as a function of time. (c) The braid in (b) shown as a standard braid diagram.

A. Physical braids

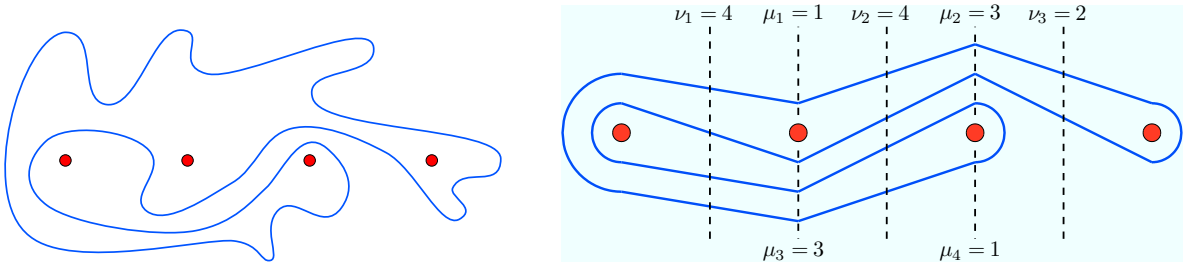
The particle trajectories can be lifted from a two-dimensional domain to a three-dimensional space where the vertical axis is time. These three-dimensional space-time trajectories are called *strands*. Due to the monotonic nature of time, a strand can only move upwards, since it cannot travel back in time. If these three-dimensional strands are projected onto the plane containing the x-axis and time then Fig. 2a becomes Fig. 2b. Note that the crossing information was preserved in making the projection, which records which strand was above another in the y coordinate. The collection of strands defines a *physical braid*. Two physical braids are considered equivalent if they can be continuously deformed into one another without the strands crossing each other or boundaries. Figure 2c is the *standard braid diagram* corresponding to Fig. 2b, in which the strands were displaced into standardized 'elementary crossings,' with only one crossing occurring at a time. On the horizontal axis we record the left-to-right position of each particle or puncture.

B. Generators

Figure 2 shows that the topological information contained in a set of trajectories — who passes in front of who, and when — can be reduced to a fairly simple picture (Fig. 2c). From there it is straightforward to pass from this geometric description to an *algebraic* one, where crossings are written as a sequence of symbols. For instance, the braid in Fig. 2c is written algebraically as

$$\sigma_1 \, \sigma_2^{-1} \, . \tag{1}$$

The symbol σ_1 denotes the clockwise interchange of the first and second particles (when viewed from above), and the symbol σ_2^{-1} the counterclockwise interchange of the second and third particles. Note that first, second, etc. refer here to the position of a particle from left to right along the x-axis at any point in time. The individual symbols in (1) are ordered from left to right in time. In general, σ_i denotes the clockwise interchange of the ith and $(i+1)$th particles, with i ranging from 1 to $n-1$ for n particles. The symbols σ_i are called *generators of the braid group on n strands*, though we also refer to the σ_i^{-1} as generators. This is indeed



(a) (b)

FIG. 3: (a) Physical loop around four punctures. (b) Simplified ('pulled-tight') representation of the same loop, called a topological loop. The crossing numbers μ_i and ν_i are also shown; the corresponding Dynnikov coordinates are $a = [1\ \ -1]$ and $b = [0\ 1]$.

a group, more precisely a finitely-generated group with an infinite number of elements. The group-theoretic description of braids is due to Artin [31]; see the book by Birman [32] or chapters in many books on knot theory [33, 34].

Thus, the topological information in a set of n trajectories can be written as a sequence of $n-1$ generators and their inverses. These generators are obtained by following the trajectories, projecting along the x-axis, and recording when two particles exchange positions along the x-axis. We call this exchange a crossing. To determine the sign of the generator (i.e., whether it is σ_i or σ_i^{-1}), we look at the y coordinate of the two particles at the time of crossing. If the y coordinate of the ith particle is greater than that of the $(i+1)$th particle the generator is σ_i; otherwise it is σ_i^{-1}. For a more complete description of the process of extracting generators from trajectories, see Thiffeault [14, 15].

C. Loops

So far we have shown how a set of particle trajectories can be converted to a physical braid, and that physical braid can then in turn be converted to an algebraic sequence of generators while preserving the essential topological information. Now we will pursue an analogous path for *material loops*. A sample loop is drawn in Fig. 3a, encapsulating four punctures (or trajectories) depicted as disks. We will only deal with loops that are *non-self-intersecting*, i.e., which do not cross themselves. This is a natural property of material loops in fluids, since by determinism two initially-distinct fluid particles can never simultaneously occupy the same point in space. We are concerned here with *topological loops* with respect to the particle trajectories. That is, two loops are considered topologically identical if they can be deformed into each other without crossing any punctures. The process of passing from a physical material loop to a simplified topological loop is analogous to passing from Fig. 2b to Fig. 2c for braids. Figure 3b shows the simplified topological loop corresponding to Fig. 3a, obtained by pulling-tight on the loop without crossing the punctures.

Now that we have passed from physical material loops to simplified topological loops, we can go a step further and provide a symbolic description of loops. The crucial fact is that any non-self-intersecting closed topological loop, winding around punctures, can be reconstructed merely by counting the number of intersections with fixed reference lines, such as the seven dashed lines in Fig. 3b. The variables μ_i and ν_i count the number of

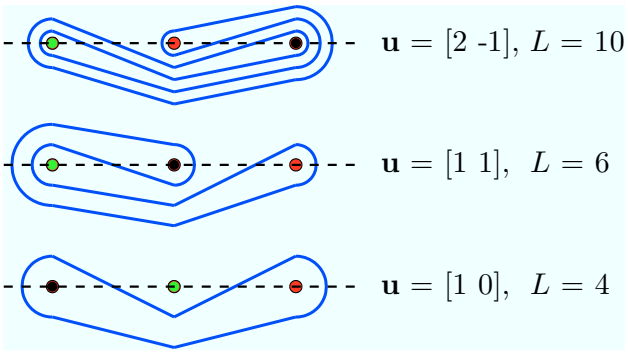


FIG. 4: How an initial loop (bottom) around $n = 3$ punctures is affected by the generators for the trajectories in Fig. 2. For each loop, the Dynnikov coordinate vector $\boldsymbol{u}$ and number of intersections with the dashed line $L(\boldsymbol{u})$ are also given.

intersections with each line, with ν_i corresponding to lines between the punctures, and μ_i lines above and below the punctures. (We do not need to know the number of crossings above and below the first and last punctures, as they can be deduced from the other crossing information.)

To any loop corresponds a unique set of crossing numbers, and from a set of valid crossing numbers we can reconstruct a loop. However, not all sets of crossing numbers are valid: if we pick some random non-negative integers for the μ_i and ν_i, then most likely they will not correspond to a non-self-intersecting closed loop. To refine the description, we define *Dynnikov coordinates* [35] by taking the difference between adjacent μ_i and ν_i:

$$a_i = \tfrac{1}{2}(\mu_{2i} - \mu_{2i-1}), \qquad b_i = \tfrac{1}{2}(\nu_i - \nu_{i+1}),$$

for $i = 1, \ldots, n-2$. These coordinates are signed integers; if we define the vector $\boldsymbol{u} \in \mathbb{Z}^{2n-4}$

$$\boldsymbol{u} = (a_1, \ldots, a_{n-2}, b_1, \ldots, b_{n-2}) \tag{2}$$

then we have a bijection between $\mathbb{Z}^{2n-4}$ and the loops. This fact is far from obvious: we refer the reader to the literature for more details [15, 25, 35, 36], and for the exact prescription of how to reconstruct a loop from the coordinates. Figure 5 shows some examples of the loops corresponding to Dynnikov coordinates.

The striking fact about the coordinates is that the loop they represent can be immensely complicated (as tends to happen in chaotic dynamical systems), and yet only a small number of integers is needed to represent it. These integers can, however, be very large. Thus the coordinates sidestep one of the biggest computational problems when dealing with material loops: the memory required to store their configuration.

D. The action of generators on loops

We now come to the final piece of our topological description. We have a simple algebraic coding of a set of particle trajectories (Section II B) and a symbolic representation for topological loops (Section II C). In this section, we combine the two by defining an *action of generators on loops*. Figure 4 illustrates the idea: we start in the bottom frame with three punctures, surrounded by a topological loop. Assume now that the punctures are particle trajectories that move as in Fig. 2a. The braid representation of their motion is $\sigma_1\sigma_2^{-1}$,

which means the clockwise interchange of particles 1 and 2, followed by the counterclockwise interchange of particles 2 and 3. This is represented in Fig. 4, reading from the bottom up to parallel Fig. 2c. The loop around the punctures is forced to grow as depicted.

However, we can sidestep having to draw any pictures by using the *action of generators on loops*. This action is a set of algebraic update rules that describes how the Dynnikov coordinates for a loop are changed by a generator σ_i or σ_i^{-1}. These update rules are somewhat bulky and have been presented elsewhere [25, 35, 36], so we do not reproduce them here (see also Thiffeault [15] which follows the present notation more closely). The update rules are piecewise-linear in the coordinates (a_i, b_i).

The consequence of these update rules is that given a set of trajectories we can very rapidly compute the growth of arbitrary loops enclosing them. This uses only integer arithmetic. Contrast this to evolving material loops: not only does this require a detailed knowledge of the velocity field, it also requires refinement of the points making up the exponentially-growing loop, which quickly leads to memory overflow. It is this rapid computation of the growth of loops that allows our detection method to work.

We will associate invariant regions, and thus transport barriers, with loops having negligible growth. Hence, we need a way to measure the length of a topological loop. Moussafir [25] demonstrated that the length of a loop is proportional to the number of times the loop crosses an imagined line passing through all the punctures (the line eventually connects to the boundary of the domain). This line is shown dashed in Fig. 4; the final number of intersections is 10 (top frame). The number of intersections $L(\boldsymbol{u})$ for a given loop is given by [15, 25, 36]

$$L(\boldsymbol{u}) = |a_1| + |a_{n-2}| + \sum_{i=1}^{n-3} |a_{i+1} - a_i| + \sum_{i=0}^{n-1} |b_i| \tag{3}$$

where

$$b_0 = -\max_{1\le i\le n-2} \left(|a_i| + \max(b_i, 0) + \sum_{j=1}^{i-1} b_j \right),$$
$$b_{n-1} = -b_0 - \sum_{i=1}^{n-2} b_i \,.$$

This formula for 'length' $L(\boldsymbol{u})$ will be used to determine how fast a loop is growing with time, and thus whether it is a candidate transport barrier.

Finally, the set of punctures enclosed by a loop that shows negligible growth, or non-growing loop, will be referred to as an *invariant puncture set* (IPS). If the flow inside an invariant region is chaotic, then any set containing more than two punctures will not be an invariant puncture set. Hence, the invariant puncture set will automatically be 'maximal' and delimit the invariant region, that is, the only way for a loop not to grow is for it to contain all the punctures in the region. If the flow inside an invariant region is not chaotic (for instance, a large vortex), then proper subsets of an invariant puncture set may also be slow. In that case we can look for the largest invariant puncture set (the one containing the most punctures).

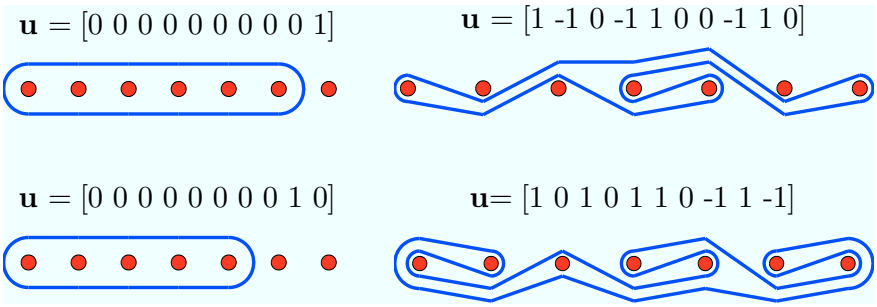


FIG. 5: Four simple loops with Dynnikov coordinates between −1 and 1.

E. How the topological tools fit together

With the tools and definitions developed in the previous sections, it is now possible to give an overview of the application of braid theory to the problem of finding two-dimensional transport barriers. We start with a set of n two-dimensional particle trajectories on some domain, obtained from numerical simulations or experimental data. We then extract a sequence of braid group generators corresponding to these trajectories. (This may require some interpolation, but is usually fairly insensitive to how this is done [15].)

The detection of invariant regions is then reduced to searching for loops that have negligible growth. There are two ways of doing this. The first method is a methodical search for non-growing loops by sequentially testing a large number of simple loops, using their Dynnikov coordinates and the action of generators on these coordinates; this is presented in Section III. This direct approach is, however, prohibitively slow for more than about eleven trajectories. We improve upon it by a refined searching technique, the pair-loop method, described in Section IV. We test both methods using a model system (Section III C).

III. SEARCH FOR NON-GROWING LOOPS: DIRECT METHOD

The basis of the direct search method is that if there are loops that have negligible growth, which correspond to transport barriers, they will emerge by examining the set of all simple loops. Here we define what we mean by a 'simple' loop, give an outline of the algorithm to analyze the loops, and apply this to a model system.

A. Simple loops

We assume that the invariant regions present in our flow are described by relatively simple loops. All loops can be described by their Dynnikov coordinates, and the complexity of a loop is typically reflected in the magnitude of the coordinates. In fact, as a loop becomes more convoluted its coordinate values grow larger. For this reason, and to make the problem computationally tractable, we limit our search to loops with initial values of the Dynnikov coordinates between −1 and 1. Four such loops are shown in Fig. 5, demonstrating the range in complexity captured by this limited set of Dynnikov coordinates. As mentioned in Section II C, a set of n trajectories has loops represented by $2n-4$ coordinates. This results in $3^{2n-4}-1$ possible loops with coordinates between −1 and 1. (The null loop with all coordinates equal to zero is not considered.) Many of these loops are 'multi-loops' (see

Fig. 8c) which are redundant, but it is not easy to eliminate those *a priori* from the Dynnikov coordinates, so we test those as well.

B. Algorithm outline

In order to find the non-growing loops, we developed an algorithm in Matlab. The objective of this algorithm is to take in a set of trajectories and output the non-growing loops surrounding punctures. The first step is to calculate the generator sequence which defines the braid of the trajectories. We discuss this in Section II B, and we refer the reader to Thiffeault [14, 15] for more details and Matlab codes.

Next, for each Dynnikov coordinate vector with entries between -1 and 1 inclusively, we apply the entire generator sequence to each loop and compute the intersection number $L(\boldsymbol{u})$, which is proportional to the loop's length (Section II C–II D). We then identify the loops with small final number of intersections as non-growing loops. As we shall see below, 'small' typically means tens of intersections, compared to most loops which grow to have 10^{10} or more intersections. The punctures enclosed by a non-growing loop form an invariant puncture set (see Section II D).

C. Modified Duffing oscillator

In order to test the direct method, we use a model velocity field with transport barriers. This velocity field will be used to calculate trajectories for analysis. While this is not the target application of the algorithm, it is useful for testing to have a system where we can determine *a priori* the location of transport barriers.

For this test, we use a slightly modified form of the Duffing oscillator. One change is made in order to create two relatively-complex invariant regions, and the other change adds solid body rotation to further obscure the dynamics. The basic system without rotation is defined as

$$\dot{x} = y + \alpha\cos\omega t, \tag{4a}$$

$$\dot{y} = x(1-x^2) + \gamma\cos\omega t - \delta y\,, \tag{4b}$$

where we will use $\alpha = .1$, $\gamma = .14$, $\delta = .08$, and $\omega = 1$. To add rotation to the system, and thus further 'hide' the transport barriers, we make the transformation

$$\begin{pmatrix}\tilde{x}\\ \tilde{y}\end{pmatrix} = \begin{pmatrix}\cos\Omega t & \sin\Omega t\\ -\sin\Omega t & \cos\Omega t\end{pmatrix}\begin{pmatrix}x\\ y\end{pmatrix}.$$

Note that with $\delta \neq 0$ this velocity field is *not* incompressible. From a topological viewpoint, this is irrelevant to the issue of invariant regions, and shows the wide applicability of the method. More importantly, the transport barriers have a more complex shape with these parameter values. A more realistic, incompressible flow will be used in Section V.

The key feature of the Duffing system with these parameter values is that there are two separate regions, plotted in Fig. 6 at $t = 0$. There is a transport barrier between the gray and the white regions, that is, all points initially in the gray region stay in their own region, and vice-versa. Thus, a set of trajectories entirely within one region should have a non-growing loop surrounding them.

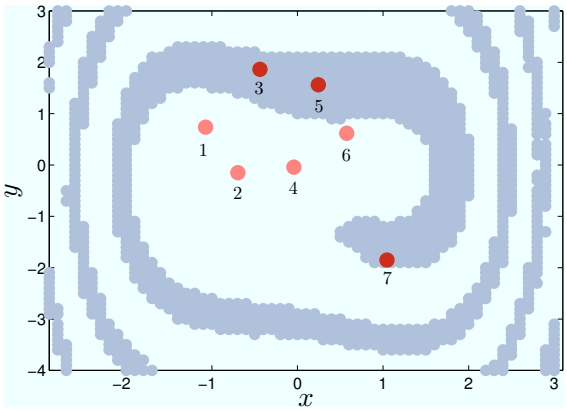
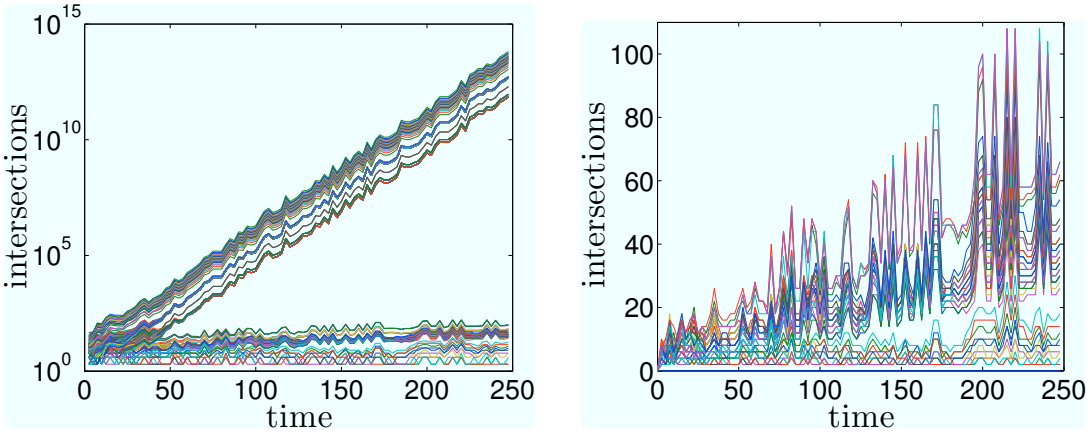


FIG. 6: The two types of initial conditions for the modified Duffing oscillator system (4) with rotation. Orbits in the white region are attracted to a chaotic region at the center. Orbits in the gray region approach a limit cycle around the white region. The dots (numbered from left to right) are the initial conditions for the trajectories discussed in Section III D.



(a) (b)

FIG. 7: (a) Intersections with the x-axis as a function of time for loops with Dynnikov coordinate values ranging from -1 to 1. (b) Zoom on a linear scale of the loops in (a) that have sub-exponential growth.

D. Results of the direct method

To test the direct method we used a variety of sets of initial conditions. A single representative example is presented here. This system consists of seven trajectories with initial conditions shown in Fig. 6. Based on the initial conditions, four of the trajectories are contained in the white region (chaotic), and the other three trajectories are in the gray region (limit cycle). From these trajectories, we compute the corresponding braid group generators (Section II B) and examine the growth of 3^{10} loops via the update rules (Section II D).

We plot the number of intersections with the x-axis as a function of time for a representative set of loops in Fig. 7a. (Recall from Section II C that these intersections are a proxy for length.) This plot indicates that there are two regimes of growth: loops growing

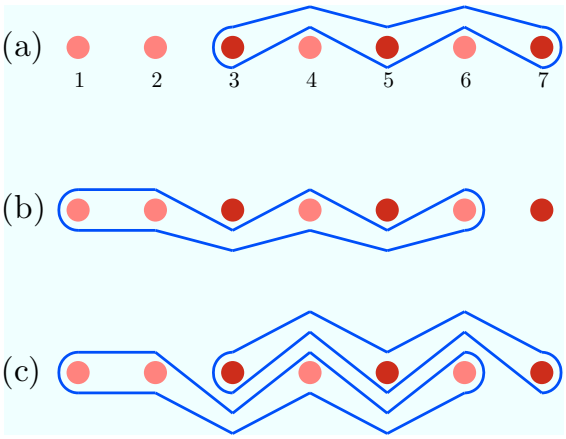


FIG. 8: Simplified loop diagrams of the loops which do not show steady growth. (a) Loop containing the limit-cycle trajectories (red punctures — 3, 5, and 7). (b) Loop containing the inner chaotic trajectories (pink punctures — 1, 2, 4, and 6). (c) Double-loop consisting of the (a) and (b) loops.

exponentially (linearly increasing in the semi-log plot) and loops that show sub-exponential growth (approximately flat in the plot). The loops growing exponentially do not correspond to loops surrounding a region, so they can be discarded. It should be noted that all the discarded loops enclose a proper subset of the four trajectories in the central chaotic region.

A closer view for the loops with non-exponentially-growing intersections is shown in Fig. 7b. This plot is on a linear scale and shows that some of the loops grow approximately linearly while others don't grow at all. The slowly-growing loops that grow linearly contain all four trajectories in the chaotic region and a proper subset of trajectories from the region with a limit cycle. The non-growing loops, shown in Fig. 6, correspond to transport barriers. Changes in the length of the loops do occur due to motion of the particles, but there is no steady growth of the initial loop.

Thus, the direct method found the expected 'topological transport boundaries' by sifting through an enormous number of loops, and finding the very few that do not grow. But a great drawback of this method is how poorly it scales as the number of trajectories is increased: the number of loops examined is $3^{2n-4} \sim 9^n$. For this algorithm, the bottleneck in the calculation is the time required to apply the generators to all the loops, which scales with the number of loops. The direct method is rendered useless when dealing with more than eleven trajectories or so, which leads us to improve the detection method in Section IV.

IV. SEARCH FOR NON-GROWING LOOPS: PAIR-LOOP METHOD

While the methodical search through simple loops produces the non-growing and slowly-growing loops, it is not practical for more than a dozen punctures. In real applications we might have access to hundreds of trajectories, so we need the capacity to easily analyze many more than a dozen trajectories.

One major drawback of the previous method is that no information is gained from a loop other than its growth rate, so every loop has to be tested. The refined algorithm tests a specific set of loops and analyzes their state after the generators have been applied. Specifically, the loops analyzed are the ones that connect two punctures, which we call

pair-loops. Their final state gives an indication of which punctures cause the loop to grow and which punctures become *entangled* by a given loop (we will define entanglement more precisely in Section IV B). This is the basis for the pair-loop algorithm.

The objective of the pair-loop method is the same as the direct method of Section III: to take a set of trajectories and find non-growing loops. The steps of the pair-loop method are as follows:

1. We calculate the generator sequence corresponding to the trajectories. We then apply this sequence to pair-loops via their Dynnikov coordinates (Section IV A).

2. We then analyze the resulting final loops to determine which punctures are entangled by each loop, yielding a collection of entangled puncture sets (Section IV B).

3. With the information on entangled puncture sets, we determine which punctures are contained in invariant regions (Section IV C).

4. We construct the loop surrounding each invariant region (Section IV D).

After describing the steps of the improved algorithm in Section IV A–IV D, we apply it to the modified Duffing oscillator in Section IV E.

A. Step 1: Application of generators to pair-loops

As mentioned in the description of the algorithm, in the pair-loop method we start with loops that connect pairs of punctures. While there are an infinite number of ways for a loop to connect punctures, the main concern here is how simple loops entangle the other punctures. Given this consideration, punctures are connected as shown in Fig. 9a. Adjacent punctures are connected as for loop $(1, 2)$ in that figure. For punctures that are not adjacent, two loops are created where one loop passes above intermediate punctures (e.g. loop $(1, 3)$ in Fig. 9a), and the other passes below (loop $(5, 2)$ in the figure). The notation for each initial pair-loop is (p, q), where if $p < q$ (resp., $p > q$) a loop connects puncture p to q and passes above (resp., below) intermediate punctures.

The first step of the method, shown diagrammatically in Fig. 10, takes the trajectories as input. As in the direct method, the generator sequence is calculated from the trajectories. Then we determine the pair-loops to be tested. For a system of n trajectories, there are $n-1$ loops connecting adjacent punctures and $(n-2)(n-1)$ loops connecting non-adjacent punctures above and below, for a total of $(n-1)^2$ loops to analyze. The generator sequence is then applied to the pair-loops and the resulting Dynnikov coordinates are the output of this step. A sample set of pair-loops before and after the application of the generator sequence is shown in Fig. 9.

At this point, a marked improvement has already been made. In the direct method, the bottleneck was the application of the generators to $3^{2n-4} - 1$ loops, causing the run time to scale exponentially as $t \sim O(9^n)$. With the pair-loop approach, the bottleneck of the algorithm still involves the number of loops analyzed, but the number of loops tested now scales algebraically as $O(n^2)$.

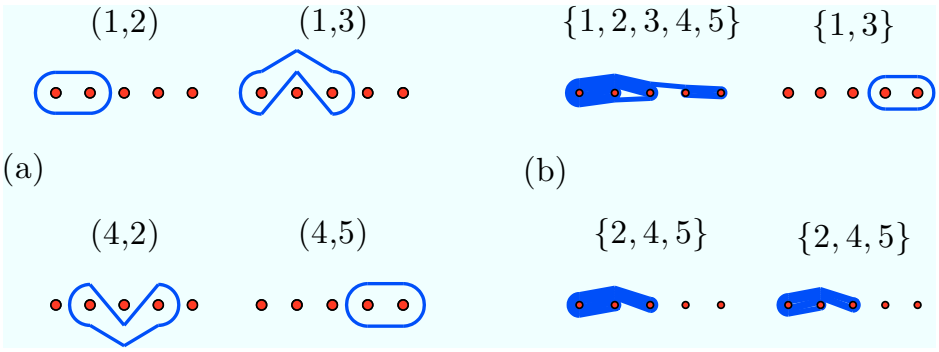


FIG. 9: (a) A representative set of the initial pair-loops for a five-puncture system. (b) The loops that result after applying a generator sequence and the corresponding entangled puncture set. The loops (1,3), (4,2), and (4,5) fail to entangle all punctures after applying the generators. This group of trajectories form the two invariant puncture sets $\{1, 3\}$ and $\{2, 4, 5\}$.

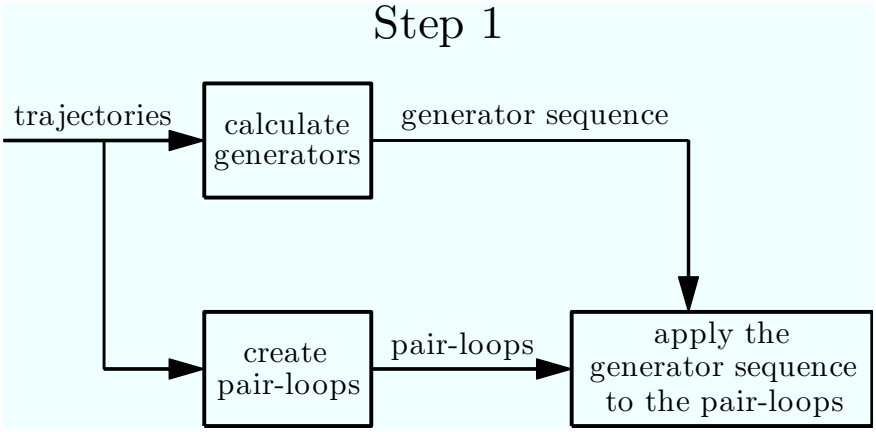


FIG. 10: Step 1 takes the trajectories as the input and outputs a set of Dynnikov coordinates representing the pair-loops after the application of the generator sequence.

B. Step 2: Determine the entangled puncture sets

After the generator sequence has been applied to the pair-loops, we need to determine which punctures are entangled by the loops. In this step, presented in Fig. 11, we go through each set of Dynnikov coordinates, change to crossing number coordinates, apply the entangled puncture criterion, and determine which punctures are entangled. We say that a puncture is *entangled* by a loop if the loop passes both above and below the puncture, with punctures ordered from left to right as usual. To determine this, the Dynnikov coordinates of a loop are converted back to the μ, ν representation (see Hall and Yurttaş [36] for the inversion formula). The odd-indexed crossing numbers μ_{2i-1} count the number of times the loop passes above a puncture, and the even-indexed crossing numbers μ_{2i} count the number of times the loop passes below the puncture (see Fig. 3).

With this μ, ν coordinate representation, it is straightforward to determine if a puncture is entangled or not: the ith puncture is entangled if both μ_{2i-1} and μ_{2i} are non-zero for the loop. For each pair-loop, the set of all entangled punctures is the *entangled puncture set* (EPS) of that loop. After finding the entangled puncture set for all loops, we remove

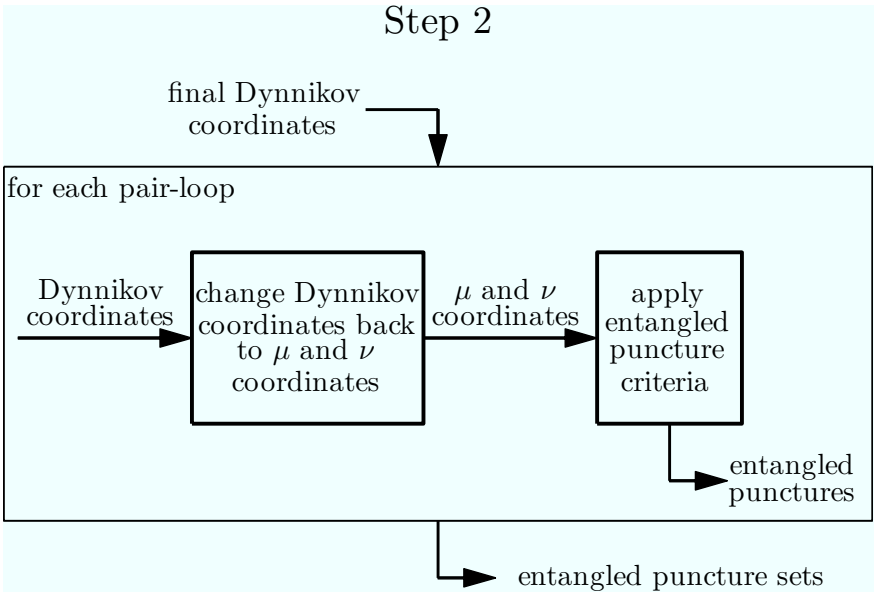


FIG. 11: Step 2 takes the pair-loops and finds the entangled puncture sets.

duplicates and only consider entangled sets which are a proper subset of the full set of punctures. (That is, we discard entangled puncture sets that contain all the punctures.)

For the example presented in Fig. 9, the generators applied to the loops in Fig. 9a produce the loops in Fig. 9b. Of the four loops shown, one entangles all punctures, so this entangled puncture set does not provide any information about the invariant regions and is discarded. The loop $(1, 3)$ entangles only punctures 1 and 3, so its entangled puncture set is $\{1, 3\}$. The loops $(4, 2)$ and $(4, 5)$ entangle only the punctures 2, 4, and 5, so their entangled puncture set is $\{2, 4, 5\}$. The example results in two entangled sets which also happen to be the punctures in the two invariant regions for this system. There is a non-growing loop around punctures 1 and 3 and one around 2, 4, and 5. Thus, in this case entangled puncture sets are also invariant puncture sets.

C. Step 3: Find the invariant puncture sets

While the example presented in Sections IV A and IV B produces the invariant puncture sets directly from the entangled puncture sets, this is not always the case. For a pair-loop, the two punctures are either within the same invariant region or not. If a pair-loop connects points within an invariant region, it can either entangle all the punctures within the region, a subset of those punctures, or a subset as well as punctures from another invariant region. In the simple example, loops connecting punctures within a region always only entangled with all the punctures within the same region. Alternatively, if a loop connects punctures from different invariant regions, they will entangle punctures from at least those two regions. This step of the method, presented in Fig. 12, takes in the entangled puncture sets and deciphers which punctures form an invariant region.

To create the invariant puncture sets, the code runs through the entangled sets and allocates a region index to each puncture in the system. The entangled sets are read in from largest to smallest (in terms of the number of punctures they contain), so the first and largest set (if there are multiple largest sets the starting set is chosen arbitrarily) has all its

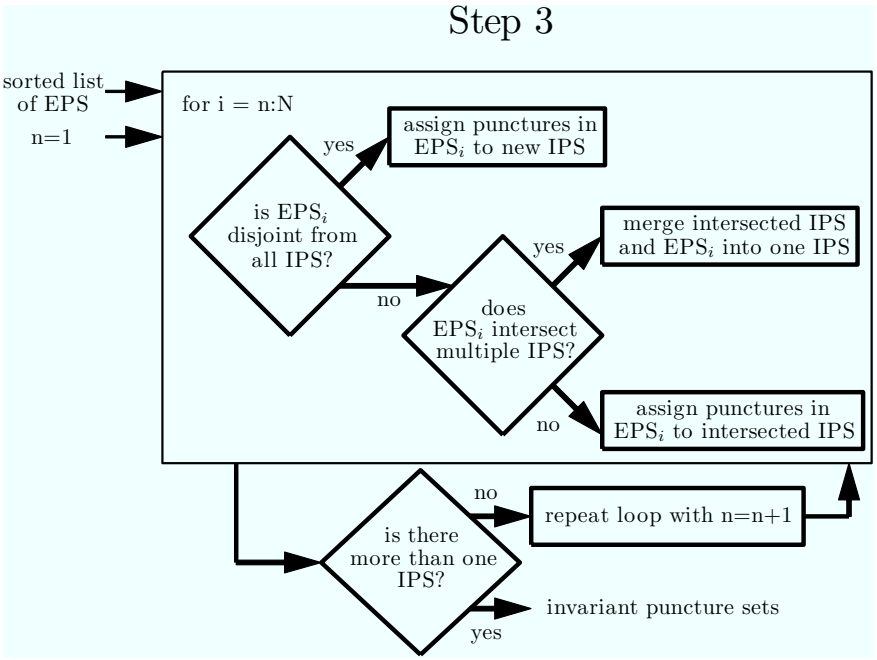


FIG. 12: Step 3 takes the entangled puncture sets and distributes them into the invariant puncture sets.

punctures designated as region 1, and then the next entangled set is analyzed. If the second set intersects the first, then all points in the second set are labeled with the same region index. Otherwise, the second set is disjoint from the first region, and it is given a new region index. This process is repeated for each of the remaining entangled puncture sets, merging the sets if they share points, or otherwise creating a new region index. It is possible for an entangled set to intersect multiple regions, in which case the intersected regions as well as the punctures in the entangled set are all merged into one region.

At the end of this process, we obtain a list of disjoint puncture sets. If this list has only one member (containing all the punctures), then we have failed to find any invariant regions, and we restart the process of finding the invariant puncture sets. But now instead of starting with the largest entangled set, we start with the next largest entangled set as our first region, completely discounting the previous largest one. We repeat until we obtain a list of several disjoint invariant puncture sets. If we fail to obtain such a list even after using all the entangled sets as a starting point, then we conclude that there are no non-growing loops present.

This method of comparing entangled sets to discern invariant regions is susceptible to the unintentional merging of multiple regions into one larger region. To combat this issue, the entire pair-loop method can be run on each invariant puncture set produced, to determine if it can be broken down into smaller invariant regions.

For clarity, the following is an example where steps 1 and 2 have produced the following entangled sets:

$$\{1,2,3,4\}, \quad \{3,4,5,6\}, \quad \{1,2\}, \quad \{3,4\}, \quad \{5,6\}. \tag{5}$$

The search for the invariant puncture sets begins by assigning the punctures in the first entangled set the region index 1. Next we determine that the second entangled set, $\{3,4,5,6\}$, intersects region 1, so all of the punctures in the second entangled set are also assigned the

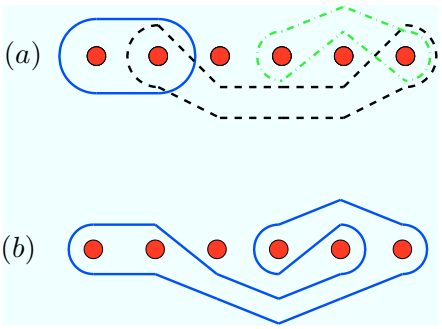


FIG. 13: Punctures 1, 2, 4, and 6 are in an invariant region. (a) The link-loops shown here connect the punctures. The solid line connects 1 to 2, dashed connects 2 to 6, and dot-dashed connects 6 to 4. (b) The connecting link-loops can be merged to form a single loop around the punctures.

region index 1. This means that all punctures are in the same region, indicating that the search failed and we must start again with the next largest set. This time we start with the entangled set $\{3, 4, 5, 6\}$, assign it the region index 1, and move on to the entangled set $\{1, 2\}$. This set is disjoint from all the assigned regions, so it is given the region index 2. The code then compares the two remaining sets, but both of these are subsets of region 1. This time the code successfully finds multiple disjoint invariant puncture sets, and moves on to the next step of the algorithm. It may be the case that $\{3, 4\}$ and $\{5, 6\}$ are actually disjoint invariant puncture sets; this is discovered by running the entire pair-loop method solely on the $\{3, 4, 5, 6\}$ trajectories.

D. Step 4: Creation of slow-growing loops

At this point in the procedure, we have successfully identified the punctures that make up the invariant puncture sets. Because the regions are assumed to be relatively simple, it is reasonable to presume that the least-complicated loop enclosing the punctures in the invariant puncture sets will be the non-growing loop. However, there are an infinite number of ways to connect the punctures, but only one way in which the loop will not grow. This step determines the loop by creating link-loops, which will be described next. Because of its complexity, this step is only necessary if the "obvious" loop surrounding the region turns out to be a growing loop. For this reason, many readers may wish to skip the rest of this section.

A loop connecting the invariant puncture set can be viewed as the union of the pair-loops connecting punctures in a region. These specific pair-loops are referred to as *link-loops*. An example is presented in Fig. 13, where three link-loops connect the punctures in a region, then are merged to form one loop. These link-loops are the key to finding the non-growing loop surrounding the entire region.

A key characteristic of the non-growing loop is that punctures from outside the region do not cause the loop to grow, nor do they cause the link-loops to grow. However, the other punctures in the invariant puncture set, but outside a given link-loop, will cause that link-loop to grow. This property is the basis for the procedure to find the correct link-loops and to build the non-growing loop.

This step takes as input the invariant puncture sets and finds the link-loops connecting

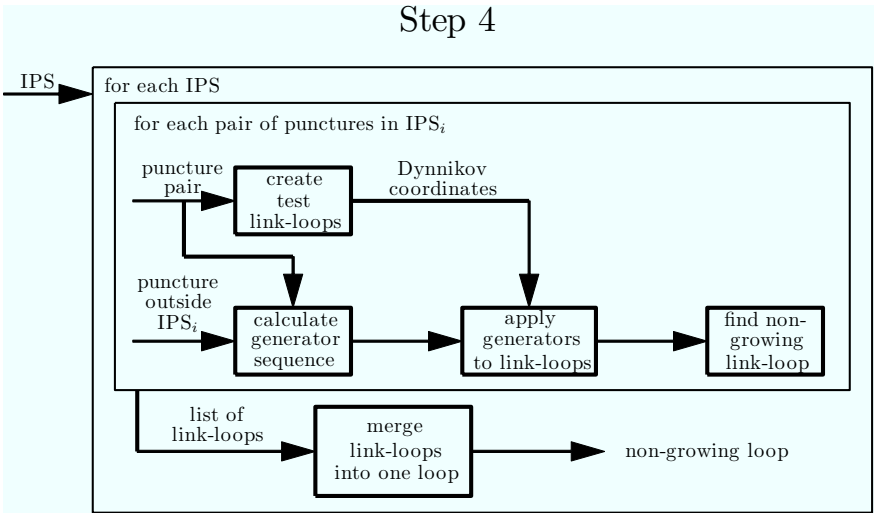


FIG. 14: Step 4 takes the invariant puncture sets and finds the corresponding non-growing loops.

# of trajectories	5	6	7	8	9	10	11	20
direct method	0.33	0.46	0.70	6.0	53	462	3445	N/A
pair-loop method	6.7	9.5	11.6	12.3	13	15	20	128

TABLE I: Run times (in seconds) for the modified Duffing oscillator using the direct and pair-loop methods. The run with 20 trajectories had over four times as many generators than previous cases, which contributes to the run time being longer than the trend would suggest.

punctures within the region. This search is done by considering the topological sub-braid resulting from the two selected punctures within the region and all punctures outside of the sub-braid. If a link-loop is present between the two punctures, there will be a loop connecting the two punctures that does not grow. Once link-loops are found connecting all the punctures in the invariant puncture sets, they are combined to form the non-growing loop. This step is outlined in Fig. 14.

E. Application to the modified Duffing system

The pair-loop method is much faster at finding non-growing loops, making it practical to use braid theory to detect transport barriers. The run times for the direct method (Table I) significantly limit its applicability, but by improving the scaling from $O(9^n)$ to $O(n^2)$ the pair-loop method is a viable alternative. Table I compares the run times of both methods when applied to the modified Duffing oscillator of Section III C, for increasing number of punctures n. Because of the extra overhead of the pair-loop algorithm, it is slower until about 9 punctures, after which it scales much better that the direct method.

While the run times for the direct method clearly scale with the predicted $t \sim O(9^n)$, the pair-loop algorithm does not demonstrate a specific trend in growth time. This is due to the fact that we are not yet in the regime where the number of loops analyzed is the

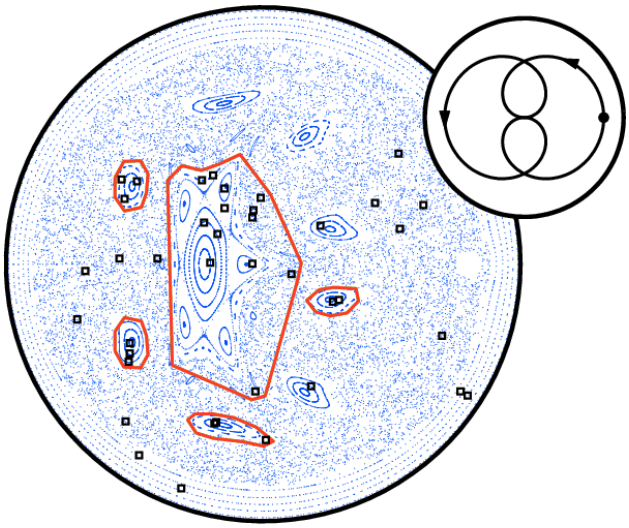


FIG. 15: Poincaré section (stroboscopic map) for a vat of viscous fluid stirred by a rod (see inset for the rod's path). A chaotic sea and several islands are visible, as well as regular orbits near the wall. The squares indicate the initial position of the 40 trajectories, with non-growing loops detected by the pair-loop method drawn in.

limiting factor. For systems where there $n > 50$, the number of pair-loops analyzed is the bottleneck.

V. FINDING NON-GROWING LOOPS IN A ROD-STIRRING DEVICE

We finally apply the method to an actual fluid system. We use the translating-rotating mixer (TRM) system described by Finn and Cox [37]. Their two-dimensional stirring mechanism features a circular domain of viscous fluid and a circular stirring rod, which can translate and rotate about its center. The authors derive an analytic complex-variable solution for the instantaneous velocity field of the fluid in the zero-Reynolds-number limit, which allows an accurate calculation of fluid trajectories.

For many parameter values, islands of poor mixing are present for periodic rod motions. Each island is surrounded by a transport boundary. These islands are undesirable, since the goal of stirring is to homogenize the entire domain. Since the rod motion is periodic, the islands can easily be identified using a Poincaré section (stroboscopic map). Our objective is to determine if the pair-loop method can also find these islands, given relatively few trajectories. We note that the pair-loop method would still apply if the system was not time-periodic, but the Poincaré section would be useless.

We specify an epitrochoidal trajectory of the rod as shown in the inset to Fig. 15. A set of 40 sample initial conditions for the trajectories is also shown in the figure as small squares: half of the initial conditions are distributed in seven islands, and the other half throughout the well-mixed chaotic region. The rod's position is also taken as a trajectory, bringing the total to forty-one strands in the braid. The trajectories are tracked for five periods of the rod's motion, and produce a sequence of seven thousand generators. The application of the pair-loop algorithm takes a few minutes and produces loops roughly drawn in Fig. 15.

The first observation is that not all of the islands have a non-growing loop around them.

The reason for this is that there are two islands which do not contain any trajectories, and two with only one trajectory. The algorithm relies on multiple trajectories being inside a transport boundary in order to detect it. If there are no trajectories, then the method has no information about the region, and a loop surrounding a single trajectory likewise provides no information (and cannot even be encoded in Dynnikov coordinates).

The other observation is that some of the loops include nearby trajectories in the chaotic sea. The largest loop and the one at the bottom of Fig. 15 are both examples of non-growing loops that extend beyond the boundaries of the island. The explanation for this is that in the period of time analyzed none of the punctures in the chaotic sea, the other islands, or the rod passes between the island and its neighbors, which then end up being included in the region. If longer or more numerous trajectories were used, then the probability that a puncture passes between the island and the nearby puncture becomes higher, increasing the likelihood of detection by the pair-loop method.

VI. CONCLUSIONS

In this paper we used tools from braid theory to find transport boundaries using trajectory data in two-dimensional systems. We described how topological loops typically grow as a result of the chaotic motion of the trajectories, and how this growth rate can be related to a topological entropy. The trajectory data is encoded as a sequence of braid group generators, and the loops are represented by Dynnikov coordinates. We then described a method for detecting non-growing loops, which enclose invariant regions. In this *direct method* we simply try a very large number of loops and look for the ones that grow slowly.

The direct method scales very poorly with the number of trajectories, so we described a more complex approach, the *pair-loop method*, where we examine a much smaller number of loops that connect pairs of punctures. Though algorithmically more involved, this method achieves the same thing as the direct method, but at a much-reduced computational cost. The pair-loop method then allowed us to detect regular islands in a simple rod-stirring device, using a set of 40 trajectories as well as the trajectory of the rod itself.

We discussed some of the limitations of the method in Section V, but let us reiterate them here. For the detection method to be successful we need enough trajectories, but also we need them to be long enough in time. The number of trajectories determines how well we can delimit invariant regions, since we need at least two trajectories in a region to have a chance at detecting it. The length in time of the trajectories determines whether loops have grown enough to be in their asymptotic growth regime, so that we can distinguish them according to growth rate.

In practical applications Lagrangian coherent structure often do not completely surround an invariant region. Thus, many regions tend to be quasi-invariant, but for long times trajectories can escape and enter the region. In our method this would lead to an inexact determination of the structures, or perhaps cause us to miss a quasi-invariant region altogether. A possible solution is to run trajectories for shorter times, which will then make quasi-invariant regions show up as invariant regions. For example, in Fig. 15 some chaotic trajectories were ascribed to the central region, even though they will eventually leak out.

Another potential issue is that the puncture (float) trajectories have to be relatively accurate, especially if they happen to pass near each other. It is necessary to have the correct generator sequence in order to find loops that do not grow. If the position data does not have a high enough resolution, it is possible that a crossing will be missed, leading

to inaccurate regions and false non-growing loops. While it is straightforward to refine a trajectory from a velocity field, this may not be an option for trajectory data taken from a float in the ocean.

There are possible applications of this method beyond fluid dynamics. For instance, one can search for clustering in granular flow data, where the trajectories are now individual grains, with no fluid between them. This may allow a better understanding of transitions in the state of a granular medium [38–40]. In future work we will apply the topological detection method to this and other physical problems.

ACKNOWLEDGMENTS

The authors thank Matthew D. Finn for his help and insights, and for the numerical simulation of the rod-stirring flow. The authors also thank George Haller and Tom Peacock for many helpful comments. This work originated at the Summer Program in Geophysical Fluid Dynamics at the Woods Hole Oceanographic Institution, supported by NSF under grant OCE-0824636. MRA was supported by NSF under grant OCE-0645529, J-LT under grant DMS-0806821.

APPENDIX: GLOSSARY OF TERMS USED

braid, algebraic: an encoding of a physical braid in terms of braid group generators. The generators are ordered temporally from left to right.

braid, physical: a set of two-dimensional trajectories plotted in three dimensions, where the third dimension is time.

braid group: the infinite group composed of all possible products of generators.

braiding factor: sometimes used as a synonym for L [see Eq. (3)], though originally it was defined in terms of the largest eigenvalue of the Burau representation [14], whose growth rate is only a lower bound for the growth of L [41, 42].

crossing number: the minimal number of times a topological loop intersects a given line.

Dynnikov coordinates: a set of signed integers, obtained from crossing numbers, uniquely describing a topological loop. For n punctures, $2n-4$ integers are needed.

entangled: a puncture is entangled by a given loop if the loop passes both above and below the puncture, with the canonical ordering of punctures from left to right.

entangled puncture set: the set of punctures that are each entangled by a given loop. These are used in the pair-loop method to detect invariant puncture sets.

entangled set: short for entangled puncture set.

generators: after projecting a physical braid, each crossing is assigned a braid group generator, denoted σ_i or σ_i^{-1}, depending on whether the crossing is over-under or under-over. For n punctures, $n-1$ generators are needed.

invariant region: an ergodic component, in the language of dynamical systems. Also called a dynamically-distinct region. Trajectories that initially start within such a region stay within the region, though the region itself may move and deform with time. An invariant region may be regular or chaotic.

Lagrangian coherent structure: a structure that can separate dynamically-distinct invariant regions. (See **invariant region**.)

link-loops: pair-loops connecting punctures within a given region.

loop: a closed curve in the domain that encloses at least two punctures and does not intersect itself. For our purposes, all loops are material loops, which means they move with the underlying flow.

non-growing loop: a topological loop that surrounds an invariant puncture setand has negligible growth when acted on by a sequence of generators.

pair-loop: a loop or topological loop that encloses exactly two punctures.

particle: an point in the domain, whose time-evolution gives a particle trajectory. In our setting, they are the same as punctures.

particle trajectory: see **trajectory**.

puncture: a point in the domain that is a topological obstruction to loops. See **trajectory**.

region: see **invariant region**.

slowly-growing loop: a topological loop that grows sub-exponentially when acted on by a sequence of generators.

invariant puncture set: the set of punctures that are enclosed by a non-growing loop.

strand: a single trajectory in a physical braid.

topological entropy (of a braid): for a set of trajectories, the topological entropy can be thought of as the growth rate of a 'rubber band' caught on the trajectories. It is a measure of entanglement, as the length of the rubber band will grow faster if the trajectories frequently crisscross each other. As the number of trajectories increases, the topological entropy of the braid converges to the **topological entropy of the flow**. (See **braid, physical**.)

topological entropy (of a flow): this is the usual topological entropy from dynamical systems theory. It measures the rate of loss of information about the precise identity of a trajectory.

topological loop: same as a loop, but topological loops are equivalent if they can be deformed into each other continuously, without crossing the punctures. We usually drop the word topological when the context makes it clear. (In topology, these are known as equivalence classes of homotopic loops.)

trajectory: a continuous path in the domain, parametrized by time. At a given time, a trajectory is a puncture in the domain. (See **puncture**.)

[1] F. J. Beron-Vera, M. J. Olascoaga, and G. J. Goni, Geophys. Res. Lett. **35**, L12603 (2008).
[2] I. Mezić, S. Loire, V. A. Fonoberov, and P. Hogan, Science **330**, 486 (2010).
[3] N. Nakamura, J. Atmos. Sci. **53**, 1524 (1996).
[4] E. Shuckburgh and P. Haynes, Phys. Fluids **15**, 3342 (2003).
[5] C. G. Farnetani and H. Samuel, Earth Planet. Sc. Lett. **206**, 335 (2003).
[6] T. G. Drake, J. Geophys. Res. **95**, 8681 (1990).
[7] G. Haller and G. Yuan, Physica D **147**, 352 (2000).
[8] G. Haller, Physica D **149**, 248 (2001).
[9] G. Haller, Phys. Fluids **13**, 3365 (2001).
[10] G. Haller, Phys. Fluids **14**, 1851 (2002).
[11] M. Mathur, G. Haller, T. Peacock, J. Ruppert-Felsot, and H. L. Swinney, Phys. Rev. Lett. **98**, 144502 (2007).
[12] W. Tang, P. W. Chan, and G. Haller, Chaos **20**, 017502 (2010).
[13] G. Haller, Physica D **240**, 574 (2011).
[14] J.-L. Thiffeault, Phys. Rev. Lett. **94**, 084502 (2005).
[15] J.-L. Thiffeault, Chaos **20**, 017516 (2010).
[16] J.-M. Gambaudo and E. E. Pécou, Ergod. Th. Dynam. Sys. **19**, 627 (1999).
[17] P. L. Boyland, H. Aref, and M. A. Stremler, J. Fluid Mech. **403**, 277 (2000).
[18] P. L. Boyland, M. A. Stremler, and H. Aref, Physica D **175**, 69 (2003).
[19] A. Vikhansky, Phys. Fluids **15**, 1830 (2003).
[20] J.-L. Thiffeault and M. D. Finn, Phil. Trans. R. Soc. Lond. A **364**, 3251 (2006).
[21] E. Kin and T. Sakajo, Chaos **15**, 023111 (2005).
[22] E. Gouillart, M. D. Finn, and J.-L. Thiffeault, Phys. Rev. E **73**, 036311 (2006).
[23] S. K. Nechaev, *Statistics of Knots and Entangled Random Walks* (World Scientific, Singapore; London, 1996).
[24] M. R. Turner and M. A. Berger, Fluid Dyn. Res. **43**, 035501 (2011).
[25] J.-O. Moussafir, Func. Anal. and Other Math. **1**, 37 (2006).
[26] M. D. Finn and J.-L. Thiffeault, SIAM J. Appl. Dyn. Sys. **6**, 79 (2007).
[27] A. Fathi, F. Laundenbach, and V. Poénaru, Astérisque **66-67**, 1 (1979).
[28] W. P. Thurston, Bull. Am. Math. Soc. **19**, 417 (1988).
[29] S. E. Newhouse and T. Pignataro, J. Stat. Phys. **72**, 1331 (1993).
[30] P. L. Boyland, Topology Appl. **58**, 223 (1994).
[31] E. Artin, Ann. Math. **48**, 101 (1947).
[32] J. S. Birman, *Braids, Links, and Mapping Class Groups*, Annals of Mathematics Studies (Princeton University Press, Princeton, NJ, 1975).
[33] C. C. Adams, *The knot book* (American Mathematical Society, Providence, R.I., 2004).
[34] K. Murasugi, *Knot theory and its applications* (Birkhäuser, Boston, 1996).
[35] I. A. Dynnikov, Russian Math. Surveys **57**, 592 (2002).
[36] T. Hall and S. Ö. Yurttaş, Topology Appl. **156**, 1554 (2009).
[37] M. D. Finn and S. M. Cox, J. Eng. Math. **41**, 75 (2001).
[38] D. Bonamy, F. Daviaud, L. Laurent, M. Bonetti, and J. P. Bouchaud, Phys. Rev. Lett. **89**, 034301 (2002).

[39] J. G. Puckett, F. Lechenault, and K. E. Daniels, in *Powders and Grains 2009, Proceedings of the 6th International Conference on Micromechanics of Granular Media*, edited by M. Nakagawa and S. Luding (American Institute of Physics, New York, 2009) pp. 675–678.
[40] F. Lechenault and K. E. Daniels, Soft Matter **6**, 3074 (2010).
[41] D. Fried, Topology **25**, 455 (1986).
[42] B. Kolev, C. R. Acad. Sci. Sér. I **309**, 835 (1989), english translation at `http://arxiv.org/abs/math.DS/0304105`.