Spectral-clustering approach to Lagrangian vortex detection

Alireza Hadjighasem,[1,*] Daniel Karrasch,[1,†] Hiroshi Teramoto,[2,‡] and George Haller[1,§]
[1]*Department of Mechanical and Process Engineering, Institute of Mechanical Systems, ETH Zürich, Leonhardstrasse 21, 8092 Zürich, Switzerland*
[2]*Molecule & Life Nonlinear Sciences Laboratory, Research Institute for Electronic Science, Hokkaido University, Kita 20 Nishi 10, Kita-ku, Sapporo 001-0020, Japan*

(Received 6 June 2015; revised manuscript received 17 November 2015; published 17 June 2016)

One of the ubiquitous features of real-life turbulent flows is the existence and persistence of coherent vortices. Here we show that such coherent vortices can be extracted as clusters of Lagrangian trajectories. We carry out the clustering on a weighted graph, with the weights measuring pairwise distances of fluid trajectories in the extended phase space of positions and time. We then extract coherent vortices from the graph using tools from spectral graph theory. Our method locates all coherent vortices in the flow simultaneously, thereby showing high potential for automated vortex tracking. We illustrate the performance of this technique by identifying coherent Lagrangian vortices in several two- and three-dimensional flows.

DOI: 10.1103/PhysRevE.93.063107

I. INTRODUCTION

It has long been recognized that even unsteady flows with aperiodic time dependence admit persistent patterns that govern the transport of passive tracers [1–3]. Generally referred to as coherent structures, these patterns are often vortex-type spatial features that remain recognizable over times exceeding typical time scales in the flow. Our goal here is to systematically decompose trajectories in such a general flow into coherent and incoherent families, providing a conceptual simplification of the underlying dynamical system.

The majority of coherent structure identification methods used in fluid dynamics continue to be Eulerian (see, e.g., [4–7] for recent examples), concerned with features of the instantaneous velocity field driving the flow [8,9]. The resulting Eulerian coherent structure criteria have been broadly used in flow structure identification, although none has emerged as a definitive tool of choice. By their focus on the velocity field, these Eulerian criteria inherently depend on the reference frame in which they are applied [10].

By contrast, Lagrangian methods identify vortical flow structures based on the properties of fluid particle trajectories [2,3,11–13]. Several of these methods are frame invariant and hence the structures they locate (or miss) are the same in all frames that translate and rotate relative to each other. This invariance is especially important for geophysical flows, which are invariably defined in the rotating frame of the earth. In such flows, long-lived coherent vortices may transport fluid over great distances, surrounded by strongly mixing background turbulence [2,14].

Lagrangian vortex detection approaches either seek a coherent material boundary to the vortex or aim to identify a coherent interior of a vortex. Coherent material vortex boundaries are special cases of Lagrangian coherent structures (LCSs), the most influential material surfaces in the flow [3]. Within this class, Lagrangian vortex boundaries can be defined either as outermost nonfilamenting closed material surfaces (elliptic LCSs [14,15]) or as outermost closed material surfaces of equal material rotation [16,17]. Other approaches target Lagrangian vortex boundaries as locations of minimal curvature change [18] or as curves that maximize the volume to boundary size ratio throughout advection [19].

Approaches seeking the interior of Lagrangian vortices have mostly been probabilistic in nature. Early techniques relied on the diagnostic use of relative and absolute dispersion [2]. Later mathematical approaches offer a bipartition of phase space into minimally diffusive regions by delineating the density evolution that can be characterized by the Perron-Frobenius or transfer operator [20–22]. Further diagnostic approaches have also been influenced by techniques for ergodic dynamical systems, such as trajectory complexity and long-term averages along trajectories [23–25].

The clustering approach developed here falls in the second category, focusing on the identification of the interiors of coherent Lagrangian vortices. Our method is unconcerned with the deformation of the boundary, requiring only a bulk coherence for the interior of the material vortex instead. We build on techniques developed over the past few decades in computer science for data clustering [26]. While clustering methods have already been used in coherent structure detection in fluid flows [27,28], here we apply spectral clustering to a graph describing the spatiotemporal evolution of a fluid. This approach identifies coherent vortices as clusters of Lagrangian trajectories remaining close over a finite-time interval. As we show, our proposed method detects coherent vortices in two- and three-dimensional flows and can be extended to higher-dimensional problems as well. Its main advantage is that it requires a relatively low number of Lagrangian trajectories as an input, making it suitable for the analysis of low-resolution trajectory data sets (see also [19,28,29] for methods with a similar capability). A recent work [30] applies a similar spectral clustering approach to the transfer-operator framework.

Prior definitions of coherence are tied to specific geometrical requirements such as convexity [17,31], lack of

[*] Corresponding author: alirezah@ethz.ch

[†] Present address: Zentrum Mathematik M3, Technische Universität München, Boltzmannstraße 3, 85748 Garching, Germany; karrasch@ma.tum.de

[‡] teramoto@es.hokudai.ac.jp

[§] georgehaller@ethz.ch

filamentation [14], or shape coherence [18] of the vortex boundary. In contrast, our approach does not pose any geometrical constraint on the vortex boundary, which helps us to identify coherent vortices that may have nonconvex or deformable boundaries. Unlike most other Lagrangian methods [14,18,20,23], which rely only on initial and final positions of particles, our method makes use of intermediate particle location information (as does [28]). Another important feature is the ability to extract the *a priori* unknown number of coherent structures from the trajectory data set together with their simultaneous detection. This is an important prerequisite for automatic vortex tracking in large-scale data sets (see also [32]).

Our approach is based on three basic principles.

Principle 1 (coherence indicator). The dynamical distance between two Lagrangian particles is the distance between their corresponding trajectories in space-time over a finite-time interval $[t_0,T]$ of interest.

Principle 2 (coherent structure). A *coherent structure* is a distinguished set of Lagrangian particles that have mutually short dynamical distances relative to the distances to particles from its complement.

This definition adopts the notion of coherence from spatiotemporal clustering algorithms [33] to coherence in fluid flows, in a fashion similar to [28]. A typical unsteady fluid, however, is not a union of coherent structures. Rather, it is composed of coherent sets and their surrounding incoherent background turbulence [1,2]. Our third principle makes this explicit as follows.

Principle 3 (coherence vs incoherence). Coherent structures are surrounded by an incoherent background of particles.

Principle 3 underlines the impossibility of a simple clustering of a general fluid flow into coherent structures. Instead, we formulate the following main objective.

Problem 1. Given a fluid domain, possibly sampled discretely, and a finite-time interval $[t_0,T]$ of interest, find a partition of the fluid domain into coherent structures surrounded by an incoherent background.

The rest of the paper is organized as follows. Section II presents our method for identifying coherent vortices. Section III describes the relationship of our method with previous methods, namely, the transfer-operator approach [20,21], its hierarchical application [34], the application of the community detection method Infomap to the transfer operator [27], and the direct application of the fuzzy C-means algorithm to trajectory data sets [28]. We demonstrate the applicability and effectiveness of our method through four examples in Sec. IV.

II. METHOD

The general outline of our method is as follows. To solve the physical Problem 1, we start with a discrete sample of the fluid flow and generate an abstract weighted graph, whose nodes correspond to Lagrangian particles and whose edge weights are determined according to Principle 1. Next we apply spectral clustering to this graph, which is particularly suited to detect clusters in the graph according to Principle 2 together with the incoherent background, consistently with Principle 3.

A. Input: A trajectory data set

The essential input for our algorithm is a spatiotemporal trajectory data set, such as particle tracks from a flow experiment, drifter data from the ocean, or data from numerical integration of a differential equation. The trajectory data set may be sparse or spatially nonuniform at the initial time. Specifically, we only assume that in a d-dimensional configuration space, n trajectory positions $\{\mathbf{x}^i(t)\}_{i=1}^n \in \mathbb{R}^d$ are available at m discrete times $t_0 < t_1 < \cdots < t_k < \cdots < t_{m-1} = T$. This information can be stored in an $(n \times m \times d)$-dimensional numerical array, with elements $\mathbf{x}_k^i := \mathbf{x}^i(t_k) \in \mathbb{R}^d$.

From this trajectory data, we define the dynamical distance r_{ij} between Lagrangian particles $\mathbf{x}^i$ and $\mathbf{x}^j$ as

$$r_{ij} := \frac{1}{t_{m-1}-t_0}\sum_{k=0}^{m-2}\frac{t_{k+1}-t_k}{2}\left(\left|\mathbf{x}_{k+1}^i-\mathbf{x}_{k+1}^j\right|+\left|\mathbf{x}_k^i-\mathbf{x}_k^j\right|\right)$$

$$\approx \frac{1}{t_{m-1}-t_0}\int_{t_0}^{t_{m-1}}|\mathbf{x}^i(t)-\mathbf{x}^j(t)|dt.$$

Here $|\cdot|$ denotes the spatial Euclidean norm and hence r_{ij} approximates the L^1 norm of pairwise trajectory distances. Since Euclidean coordinate transformations leave Euclidean distances unchanged, one readily sees that the pairwise distances are objective, i.e., they remain unchanged in coordinate systems rotating and translating relative to each other [35]. Moreover, it is noteworthy that the pairwise distances remain unchanged under refinements of the spatial resolution.

B. Similarity graph construction

Next we convert the spatiotemporal data set with the pairwise distances r_{ij} into a similarity graph $G = (V,E,W)$, which is specified by the set of its nodes $V = \{v_1,\ldots,v_n\}$, the set of edges $E \subseteq V \times V$ between nodes, and a similarity matrix $W \in \mathbb{R}^{n\times n}$ that associates weights w_{ij} to the edge e_{ij} between the nodes v_i and v_j. Specifically, the nodes of G are defined as the Lagrangian particles, i.e., $v_i = \mathbf{x}^i$. The edges between these nodes have the associated weights

$$w_{ij} = 1/r_{ij}, \quad i \neq j, \tag{1}$$

$w_{ij} = 1/r_{ij}$ for $i \neq j$, expressing pairwise similarities between distinct Lagrangian particles. Other definitions of similarity are also possible. In general, converting distance to similarity can be done via any monotonically decreasing function, as long as the distance function r_{ij} is a metric, i.e., it satisfies for all points in the space the metric axioms of identity, non-negativity, symmetry, and triangle inequality. This is according to the intuition that the ordering of graph nodes from most dissimilar to least dissimilar should be preserved through the similarity conversion.

Extending the present similarity definition (1) to the diagonal of W would yield infinitely large quantities. To regularize W, we set the diagonal elements to a large constant $w_{ii} = K \gg 1$, $i = 1,\ldots,n$. As we will see later, the actual value of K is immaterial in our algorithm.

The entries of W characterize the likelihood of nodes v_i and v_j to be in the same coherence cluster. By construction, W is non-negative ($w_{ij} \geqslant 0$) and symmetric ($W = W^\top$, with the superscript $\top$ referring to matrix transposition).

The degree of a node $v_i \in V$ is defined as [36]

$$\deg(v_i) := \sum_{j=1}^{n} w_{ij}.$$

Subsequently, the degree matrix D is defined as the diagonal matrix with the degrees $\deg(v_i)$ on the diagonal. For a subset $A \subset V$ of nodes, we denote its complement in V by $\overline{A}$. We measure the size of A by two different quantities

$$|A| := \text{number of elements in } A,$$
$$\text{vol}(A) := \sum_{i \in A} \deg(v_i).$$

Here $|A|$ measures the size of A by its number of nodes, while vol(A) measures the size of A by summing over the weights of all edges attached to nodes in A.

C. Graph sparsification

For large data sets, storing all entries of the similarity matrix W is prohibitive. For instance, storing $n = 10^6$ elements with double precision requires 8 terabytes of memory, which clearly exceeds the capacity of today's typical personal computers [37].

To address this issue, techniques have been developed to sparsify W by retaining only elements describing strong enough similarity. Two widely used approaches are the k-nearest-neighbor and the ϵ-neighborhood approaches [38]. In the former, w_{ij} is retained if v_j (or v_i) is among the k nearest neighbors of v_i (or v_j), $k \ll n$. In the latter, w_{ij} is retained if it exceeds a specified threshold ϵ. All other w_{ij} entries are set to zero and hence require no storage. Other advanced sparsification approaches include random sampling [39], sampling in proportion to edge connectivities [40], sampling in proportion to the effective resistance of an edge [41], and sampling using relative neighborhood graphs [42–44].

Here we select the ϵ-neighborhood approach because of its low computational cost. For the practical determination of nearest neighbors, a number of efficient packages are available [45,46].

D. Spectral clustering

With the notation developed so far, our original Problem 1 can be reformulated as follows.

Problem 2 (similarity graph clustering). Given a similarity graph, find a partition of the set of its nodes into clusters such that both of the following hold.

(a) Nodes in the same cluster are similar to each other, which aims to maximize the within-cluster similarities.

(b) Nodes in a cluster are dissimilar from those located in other clusters or those not included in any cluster (incoherent background), which aims to minimize the between-cluster similarities.

These two requirements for clusters implement Principles 2 and 3, respectively. A particularly efficient method to identify clusters is spectral clustering, which we discuss below (see also [38] for a review).

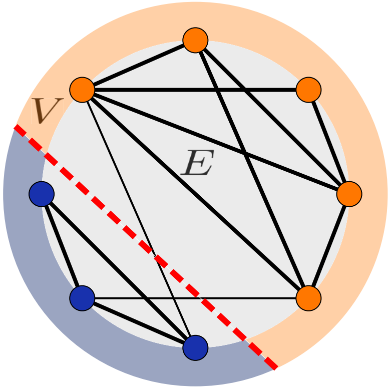


FIG. 1. Undirected graph partitioning. The dashed line shows the solution of the problem of finding a decomposition of the graph into two size-balanced groups with a minimal number of edges connecting nodes from distinct groups.

1. Spectral clustering and optimal graph cuts

Given a similarity graph $G = (V,E,W)$, a graph cut is a partition of the set of nodes V into two (or possibly more) subsets A and B. To such a partition, we assign a weight cut $W(A,B)$ defined as the sum of the edge weights between two sets A and B, i.e.,

$$W(A,B) := \sum_{i \in A, j \in B} w_{ij}.$$

Now consider a subset of graph nodes with very high within-group similarity and with weak connections to its complement, such as the orange set in Fig. 1. A graph cut separating this subset from the rest of the graph (such as the cut indicated by the red dashed line) then yields a much smaller weight cut $W(A,\overline{A})$ than another graph cut through A, which would necessarily cut some of the strong connections within A.

This suggests the following minimization problem, also known as the mincut problem, as a solution of Problem 2: For a given number k of subsets, the mincut problem is to find a partition $A_1, \ldots, A_k$ of V that minimizes

$$\text{cut}(A_1, \ldots, A_k) = \frac{1}{2} \sum_{i=1}^{k} W(A_i,\overline{A}_i). \tag{2}$$

For $k = 2$, the mincut problem can be solved very efficiently (see, e.g., [47]). In practice, however, the solution of the mincut problem often just separates one individual node (the one with weakest connections) from the rest of the graph. One way to circumvent this problem is to penalize the smallness of sets in candidate partitions. The most commonly applied objective functions that implement this idea are the normalized cut [48], or NCut for short, RatioCut [49], MinMaxCut [50], and Cheeger ratio cut [51]. Notably, not all of these graph cut objective functions have solutions that satisfy both conditions in Problem 2 (see [38] for more details).

In this paper we use the NCut objective function, whose (approximate) solutions maximize the within-cluster similarity and minimize the between-cluster similarity:

$$\mathrm{NCut}(A_i,\ldots,A_k)=\frac{1}{2}\sum_{i=1}^{k}\frac{\mathrm{cut}(A_i,\overline{A}_i)}{\mathrm{vol}(A_i)}.$$

Introducing the penalizing balancing conditions, however, turns the originally simple mincut problem into an NP-hard problem [52]. Spectral clustering is a way to solve relaxed versions of balanced graph cut problems.

2. *Graph Laplacian*

Shi and Malik [48] showed that the solution of the Ncut problem can be approximated by solutions of the generalized eigenproblem associated with the (unnormalized) graph Laplacian $L=D-W$, where D is the diagonal degree matrix of node degrees and W is the similarity matrix defined earlier. The generalized eigenvalue problem for the graph Laplacian is then defined as

$$Lu=\lambda Du. \tag{3}$$

We refer to its solutions as generalized eigenvectors for short. Generalized eigenvectors u then offer an alternative representation of the weighted graph data. As we will see in the following sections, this change of representation enhances the cluster properties in the data, so that clusters can be easily detected in the new representation. In particular, the simple K-means clustering algorithm has no difficulties to detect the clusters in this new representation (see Sec. II F regarding K-means clustering).

It is known from spectral graph theory [36] that the eigenvalues solving (3) satisfy $0=\lambda_1\leqslant\cdots\leqslant\lambda_n$. If the underlying graph consists of k disconnected components (clusters with zero between-cluster similarity), then $\lambda=0$ is a generalized eigenvalue of multiplicity k. In that case, the eigenspace corresponding to this eigenvalue is spanned by the indicator vectors of the individual connected components. A perturbation argument implies that if the between-cluster similarities remain small, then the eigenvectors of the first k eigenvalues remain close to indicator type [38]. This enables reconstructing the clusters from the first k eigenvectors obtained from (3). The main challenge, therefore, is to extract a meaningful number of clusters directly from the data, as opposed to postulating its value beforehand.

E. Estimating the number of clusters by eigenspace analysis

For a predetermined number k, the spectral clustering algorithm of Shi and Malik [48] collects the k dominant generalized eigenvectors as cluster indicators in a matrix $U=(u_1,\ldots,u_k)\in\mathbb{R}^{n\times k}$. To retrieve k from the graph data, we adopt here the eigengap heuristic [53] by which

$$k=\arg\min_i[\max(g_i)], \tag{4}$$

where $g_i=\lambda_{i+1}-\lambda_i$ for $i=1,\ldots,n$. In other words, k is simply determined as the number of eigenvalues preceding the largest gap in the eigenvalue sequence. The presence of such a gap enables us to invoke the perturbation argument of the previous section and argue that our graph $G=(V,E,W)$ is a perturbation of one with k disconnected components.

Expression (4) determines the number of coherent clusters satisfying the definition given in Sec. II D. Ultimately, however, we need to partition the graph $G=(V,E,W)$ into $k+1$ clusters to also account for the incoherent cluster surrounding the coherent clusters, as codified in our Principle 3. We refer to the last, $(k+1)$st, cluster arising in this process as the noise cluster or incoherent cluster since it includes nodes that do not belong to any coherent cluster.

Spectral gap arguments were used before in the context of dynamical systems (see [19,25,54–56] for examples). While the number of cluster indicators (leading singular vectors and eigenvectors) in some of these works (i.e., [54,55]) coincides with the number of coherent structures, in others (i.e., [19,25,56]) the number of cluster indicators differs from the number of coherent structures (see [57] and Sec. III A for more details).

Remark 1. As discussed, we identify the number of vortices present in a given domain by locating the largest gap in the eigenvalue sequence. This implies that the number of eigenvalues and eigenvectors to be computed should be greater than the maximum number of vortices expected to be present in a domain. In the absence of intuition for the maximum number of vortices, one needs to conduct a full matrix decomposition instead of a partial decomposition. The computational cost of such a decomposition, however, increases dramatically with respect to the number of eigenvalues to be computed (see [58] for more information).

F. Retrieving clusters from matrix *U* by *K*-means clustering

As the last step we employ K-means clustering to convert relaxed continuous spectral vectors, corresponding to U's k columns, into a discrete cluster indicator vector containing the cluster assignment for each node x^i. Given the spectral vectors $U\in\mathbb{R}^{n\times k}$ and integer K, K-means clustering aims to determine K points in $\mathbb{R}^k$, called centers, so as to minimize the mean-square distance from each node to its nearest center. Lloyd [59] suggested a simple iterative algorithm that efficiently finds a local minimum for this problem. Given any set of K centers, the algorithm proceeds by alternating between the following two steps.

Assignment. Find each node's nearest center and assigns it to the corresponding cluster.

Update. Recalculate cluster centers by measuring the mean of all nodes included in each cluster.

These steps repeat until no node is reassigned. Readers not familiar with K-means can read about this algorithm in numerous textbooks; see, for example, [26]. Throughout the paper, we choose the number of cluster centers K equal to $k+1$, where the last, $(k+1)$st, cluster corresponds to the incoherent or noise cluster discussed in Sec. II E. The K-means algorithm and its probabilistic counterpart (fuzzy C-means) have been used before to extract coherent structures either directly from a trajectory data set [28] or indirectly from cluster indicators resulting from various spectral dimensionality reduction algorithms [25,56].

We summarize our numerical procedure in Sec. II F.

Algorithm 1.

Input: similarity matrix $W\in\mathbb{R}^{n\times n}$ (see Sec. II B).

1. Sparsify W by using the NCut algorithm (see Sec. II C). Remove isolated nodes, i.e., nodes with degree zero, from $G = (V,E,W)$.

2. Compute the graph Laplacian L and solve the generalized eigenvalue problem $Lu = \lambda Du$.

3. Identify the number k of coherent clusters as the number of eigenvalues preceding the largest gap among the increasingly ordered eigenvalues. Select the first k generalized eigenvectors $u_1,\dots,u_k$ as coherent cluster indicators.

4. Assemble the matrix $U = (u_1,\dots,u_k)$. Each row of U corresponds to a graph node (excluding the isolated nodes). Apply K-means to the first k eigenvectors and extract $k+1$ clusters. The last cluster is the incoherent cluster and corresponds to the mixing region filling the space between coherent clusters.

Output: clusters $C_1,\dots,C_{k+1}$.

G. Large-scale spectral clustering

For large data sets, considerable time and memory is required to compute and store the similarity matrix W and the graph Laplacian L. The most commonly used approach to address this issue is graph sparsification, as discussed earlier in Sec. II C. From the sparse similarity matrix W so obtained, one determines the corresponding Laplacian matrix L and applies a sparse eigenvalue solver.

Even after the sparsification of W, however, calculating the generalized eigenvectors of the graph Laplacian L remains challenging with $O(n^3)$ worst-case complexity [37]. Several authors [37] tried to alleviate the problem by adapting standard eigenvalue solvers to distributed architecture. Other approaches are designed to achieve efficiency by finding numerical approximations to eigenfunction problems [60–62].

Here we adopt a low-rank matrix approximation approach. The main idea is to coarse grain the similarity graph $G = (V,E,W)$ while keeping as much information as possible from the original graph and its weights. To this end, we construct a bipartite graph $G_{\mathcal{B}} = (V_{\mathcal{B}},E_{\mathcal{B}},W_{\mathcal{B}})$ from the original similarity graph by uniform spatial sampling of q graph nodes, called supernodes, from n graph nodes, where $q \ll n$ [63,64]. A bipartite graph is a graph whose set of nodes $V_{\mathcal{B}}$ admits a partition into two disjoint sets A and B such that each edge connects a node in A to one in B. As a result, no two nodes within A and within B are connected by an edge. Here we set A as the set of all n original graph nodes and B as its subset of q supernodes, considered as independent copies. The weights are now defined as before, such that the square $(n+q)\times(n+q)$ similarity matrix $W_{\mathcal{B}}$ of the bipartite graph can be written as

$$W_{\mathcal{B}} = \begin{pmatrix} 0 & Z^\top \\ Z & 0 \end{pmatrix}, \tag{5}$$

where $Z \in \mathbb{R}^{q\times n}$ is a tight similarity matrix containing the edge weights between all nodes and supernodes, i.e., between A and B. Now one can pose the Ncut problem to the bipartite graph whose similarity matrix enjoys a simple block structure. As shown by Dhillon [65] and Zha *et al.* [66], this block structure breaks the associated Ncut problem into two parts such that the dominant right singular vectors of the normalized $q\times n$ tight similarity matrix $\hat{Z} = D_2^{-1/2} Z D_1^{-1/2}$ play the role of the generalized eigenvectors of the graph Laplacian in Sec. II D. Here D_1 is an $n\times n$ diagonal matrix whose entries are column sums of Z and D_2 is a $q\times q$ diagonal matrix whose entries are row sums of Z (see Appendix B for more details).

We now summarize our algorithm for large-scale trajectory data sets.

Algorithm 2.

1. Select uniformly q supernodes from n graph nodes.

2. Construct a tight similarity matrix $Z \in \mathbb{R}^{q\times n}$ between all original graph nodes and the supernodes.

3. Given Z, form $\hat{Z} = D_2^{-1/2} Z D_1^{-1/2}$. Compute the singular values and vectors of $\hat{Z}$. Select the first k right singular vectors $u_1,\dots,u_k$ as cluster indicators for the original graph.

4. Assemble the matrix $U = (u_1,\dots,u_k)$. Each row of U corresponds to a graph node. Apply K-means to the first k right singular vectors and extract $k+1$ clusters. The last cluster is the incoherent cluster and corresponds to the mixing region filling the space between coherent clusters.

Output: clusters $C_1,\dots,C_{k+1}$.

III. RELATED PREVIOUS WORK

A. Transfer-operator approach

In the transfer-operator-based approach [20–22] finite-time coherent sets are defined as regions in phase space that minimally diffuse with the surrounding phase space during a finite-time interval. The method builds on the Perron-Frobenius operator or transfer operator, which describes the evolution of material densities under the flow map.

In practice, the infinite-dimensional transfer operator needs to be approximated by a finite-dimensional matrix, the transition matrix P, which is most commonly obtained from a partition of the flow domain $(B_i)_i$ and the flow image $(C_j)_j$ into distinct boxes, and subsequent computation of discrete transition probabilities: The transition matrix entry P_{ij} is computed as the number of particles transported from B_i to C_j, normalized by the total number of particles released from B_i (see Fig. 2). This box partitioning is also referred to as Ulam's method and introduces (numerical) diffusion at the implementation level [20].

In our context, the transition matrix P can be interpreted as the tight similarity matrix Z of a bipartite graph $G_{\mathcal{B}}$ as follows: Define the first set of nodes A as the collection of initial boxes B_i, the second set of nodes B as the collection of final boxes C_j, and the edge weights as $Z_{ij} = P_{ij}$ (see Fig. 2). A similar connection to spectral clustering and graph cuts has been worked out earlier in [67]. Our presentation here, however, differs from [67] in that we interpret the graph as a bipartite graph and relate it to the original references [65,66].

Remark 2. The size of the resulting weight matrix depends on the size of the B_i's and C_j's as well as on the underlying dynamics of the system. For instance, in the presence of chaotic dynamics, particles released at the initial time can scatter in a large domain. This, in return, may require a large number of boxes C_j to cover the final domain and results in a large number of columns in the subsequent transition matrix. In contrast, the size of the weight matrix of Algorithms 1 and 2 depends on the number of tracked particles.

With this bipartite graph construction, the optimization problem that is underlying the definition of a coherent set in

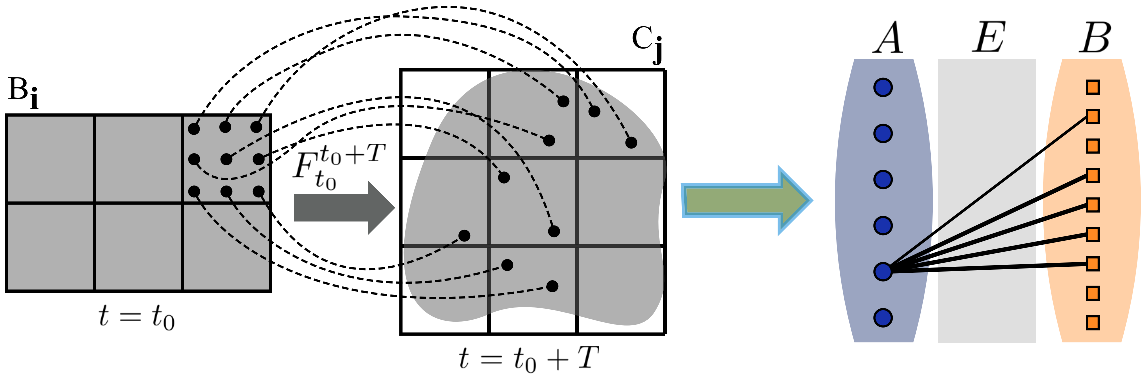


FIG. 2. Interpretation of the transition matrix constructed from tracer advection as a tight similarity matrix Z of a bipartite graph.

the transfer-operator setting can be reformulated as a clustering problem. In a (bipartite) graph cut, such as the one shown in Fig. 3, the weight of the cut can be interpreted as the mass leakage of one set with its complement.

As discussed in Appendix A, minimizing the normalized cut for a binary cluster indicator is NP-hard. Relaxation of the binary cluster indicator in the real value domain yields the eigenvector corresponding to the second smallest eigenvalue of L as an approximate cluster indicator [48]. However, in order to obtain a partition of the graph, we need to retransform the real-valued cluster indicator vector of the relaxed problem into a discrete indicator vector. The simplest way to do this is to use the sign of the eigenvector as a discrete cluster indicator function [48]. Alternatively, one can search for a splitting point such that the resulting partition has the best NCut$(A,\bar{A})$ value [48] or apply the line-search algorithm of [20]. Viewing the transfer-operator approach [20] as a bipartite spectral graph partitioning [65,66], one can similarly recover discrete cluster indicator vectors from real-valued singular vectors.

Figure 4(a) shows the second largest singular vector of the normalized transition matrix for the Bickley jet model discussed in Sec. IV B. We obtain the corresponding binary cluster indicator by searching through all possible Ncuts [48]. As shown in Fig. 4(b), the binary cluster indicator so obtained highlights two coherent sets in which coherent vortices still remain hidden. Therefore, we step in hierarchy of increasingly ordered singular vectors and search for these vortices in the third singular vector, shown in Fig. 4(c). Similarly, we extract the corresponding discrete-valued indicator vector by examining all possible Ncuts [see Fig. 4(d)]. Figure 4(d) reveals that the yellow set, as a single entity, is composed of two vortices. The two vortices forming the yellow set have overall small mass exchange with the blue set, implying that the objective function is minimized. The spatial connectedness, however, appears to be missing in this solution as the yellow set is composed of two distant vortices. In other words, a search for sets with minimal mass exchange, without enforcing spatial connectedness, can lead to a union of coherent structures as a solution (see also [27,68] for similar observations). In this case, applying K-means clustering to the collection of leading singular vectors may resolve the issue. However, the number of coherent structures that needs to be extracted may not be detectable anymore with the eigengap heuristic (see Sec. II E), as each singular vector highlights a combination of coherent structures. In fact, the number of coherent structures generally does not coincide with the number of singular values or eigenvalues preceding the largest eigengap (see [56]) and therefore needs to be guessed or to be known *a priori*.

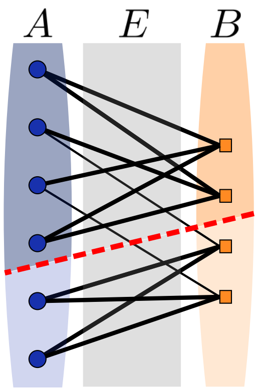


FIG. 3. Partitioning of a bipartite graph $G_{\mathcal{B}} = (V_{\mathcal{B}}, E_{\mathcal{B}}, W_{\mathcal{B}})$ whose set of nodes $V_{\mathcal{B}}$ is divided into two disjoint sets A and B such that $V_{\mathcal{B}} = A \cup B$. The dashed line shows the solution of normalized graph cut yielding a simultaneous decomposition of A and B.

The graph cut approach, in general, does not guarantee that the resulting cluster will form a connected region in the physical space. The connectedness constraint, however, can be enforced indirectly during the construction of the similarity graph. Our method specifically enforces this constraint by measuring and penalizing distances in the spatiotemporal domain. As a result, unlike for the transfer-operator method, a union of coherent structures is not a solution for our method. In Sec. IV B, we will apply our Algorithm 1 to the same Bickley jet flow considered earlier in Fig. 4.

B. Hierarchical partitioning of the transfer operator

In the spectral clustering community, one distinguishes between two approaches to detect a specified number of clusters in a given similarity graph using the graph cut procedure [38,48,65,66]: two-way clustering and multiway clustering. Our methodology presented in Sec. II follows (up to the introduction of the incoherent cluster) the multiway clustering approach, in which k clusters are retrieved from the k dominant eigenvectors at once.

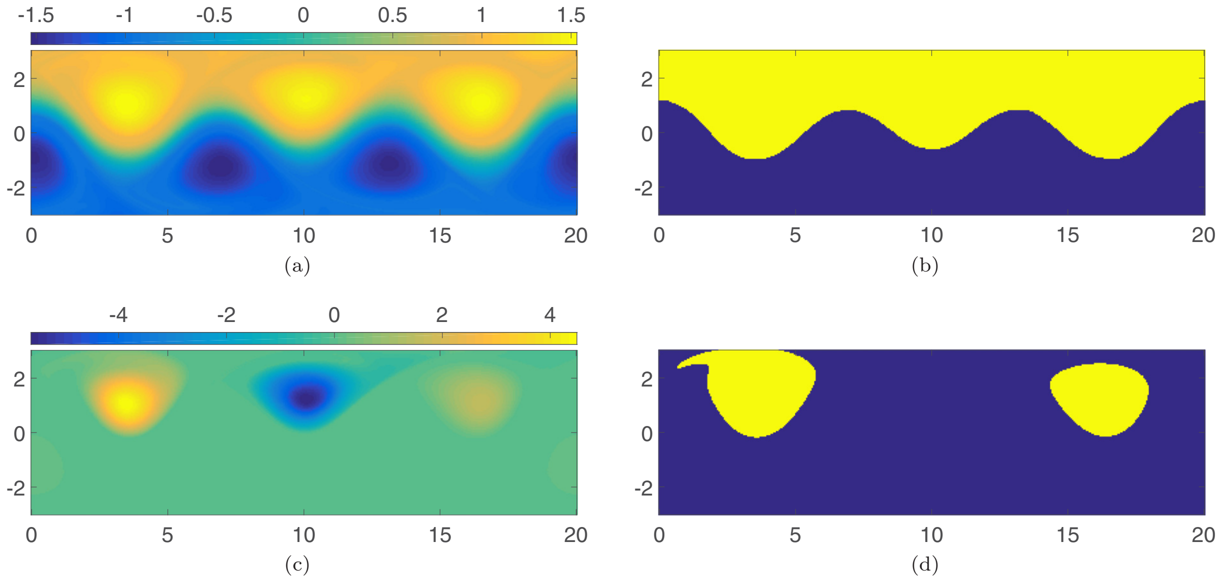


FIG. 4. (a) Second and (c) third largest (left) singular vectors of the normalized transition matrix for the Bickley jet flow. (b) and (d) Corresponding coherent sets that are obtained by searching through all possible cuts [48]. To compute the transition matrix, we subdivided the domain into a grid of 400 × 120 identical boxes and released 400 particles in each box. We then advected particles from $t_0 = 0$ to $t = 40$ days.

In two-way clustering, the following procedure is applied recursively to generate multiple clusters: (i) Compute the top generalized eigenvector of the unnormalized graph Laplacian and (ii) bisect the graph into two subgraphs. In the transfer-operator context, this procedure has been put forward in [34] and is stopped when the obtained partitions no longer satisfy a prespecified coherence ratio (see [34] for details). In the clustering analysis community, two-way clustering is also found to be inefficient due to the fact that separate eigenvalue problems need to be solved repeatedly [48,69,70].

C. Application of fuzzy clustering to a trajectory data set

Recently, Froyland and Padberg-Gehle [28] proposed a method based on traditional fuzzy C-means clustering [71,72] to identify regions of phase space that remain compact over a finite-time interval. Specifically, they first build a trajectory data set $X \in \mathbb{R}^{n\times dm}$ whose rows are vectors $(X_i)_{i=1,\dots,n}$ containing concatenated positions of Lagrangian particles in time. Second, they apply the C-means algorithm, with a prespecified number of clusters K and a set of K initial starting points in $\mathbb{R}^{dm}$, to the trajectory data set. The result is a membership value describing the likelihood that a trajectory belongs to a cluster. Thus, each trajectory carries K membership values, showing the degree of belonging to each of the K clusters. Finally, each trajectory is assigned to only one cluster based on the maximum membership value it carries. Those trajectories carrying low membership values for all clusters are occasionally considered to be noncoherent (see [28] for more details).

Compared with the fuzzy C-means clustering used in [28], the spectral clustering technique considers the connectedness of the data, whereas the C-means clustering method considers the compactness of the data. The fuzzy C-means algorithm optimizes cluster compactness by assessing the proximity between the uncertain data points assigned to the cluster and the corresponding cluster center. We note that cluster centers are not true trajectories of a dynamical system, although they are in the trajectory space [28]. In contrast, our spectral clustering technique maximizes connectedness inside clusters and disconnectedness between clusters at the same time by measuring pairwise distances between trajectories.

As opposed to centroid-based clustering algorithms such as K-means or C-means, where the resulting clusters tend to be convex sets [28,73,74], spectral clustering can find any cluster shape, because it has no preference for the shape of the cluster. This is important as we will show in Sec. IV C that vortices with nonconvex shapes are the rule rather than the exception considering the known vortex stirring in geophysical flows [75].

Most clustering methods including centroid-based methods are plagued with the problem of noisy data, i.e., identifying good clusters among noise points that just do not belong to any cluster [76]. In some cases, even a few noisy points or outliers may bias the final output of the algorithm [76]. In our specific context, the noise corresponds to the incoherent or turbulence region itself, where particles do not remain compact. This implies that the turbulence region is not residing in a hypersphere and consequently cannot be captured by adding an extra cluster to C-means or K-means algorithms (see [76] for more details).

On the other hand, the high dimensionality of the trajectory data set poses a considerable challenge to K-means or C-means clustering approaches. First, the curse of dimensionality can cause slow convergence for these traditional algorithms and, second, the existence of redundant subspaces may not allow for the identification of the underlying structure in the data (see [77] and [78], p. 10).

Similar to many clustering methods, the K-means or C-means algorithms assume that the number of clusters K in the data set is known beforehand, which is not necessarily true in real-world applications. In contrast, the spectral clustering can detect the right number of clusters automatically using techniques such as the eigengap heuristic (see Sec. II E).

Finally, the result of K-means or C-means clustering depends on the initial guess for the cluster centers [73,74] and can reach a local minimum of the objective function instead of the desired global minimum [74,79]. Often one restarts the procedure a number of times to mitigate the problem. However, when the number of clusters K is large, the number of times to restart K-means or C-means to reach an optimum can be prohibitively high and lead to a substantial increase in run-time (see [79]).

IV. RESULTS

We demonstrate the implementation of Algorithms 1 and 2 on four examples to detect coherent Lagrangian vortices. In the first example, we consider a periodically forced pendulum for which we can explicitly confirm our results using an appropriately defined Poincaré map. Our second example is one whose temporal complexity is one level higher: the Bickley jet with quasiperiodic time dependence [80,81]. In the third example, we detect coherent Lagrangian vortices in a quasigeostrophic ocean surface flow derived from satellite-based sea-surface height observations [82]. Our last example is a three-dimensional velocity field, the Arnold-Beltrami-Childress (ABC) flow, which is an exact solution of Euler's equation [83]. This is our computationally most demanding example, where we deploy Algorithm 2 to reduce the graph size and the associated computational cost. For the rest of the examples, we use Algorithm 1 with the ϵ-neighborhood graph sparsification approach described in Sec. II C. We notice that the coherent structures in our first and last examples remain invariant in the phase space and hence they are in principle detectable by other spectral methods developed specifically for steady flows and maps (see, e.g., [25,54–56,84]).

To implement Algorithms 1 and 2 in the forthcoming examples, we use a variable-order Adams-Bashforth-Moulton solver (ODE113 in MATLAB) to solve the differential equations. The absolute and relative tolerances of the ordinary differential equation (ODE) solver are chosen as 10^{-6}. In Sec. IV C, we obtain the velocity field at any given point by interpolating the velocity data set using bilinear interpolation.

The dynamic distances r_{ij} can be computed using two approaches that differ in terms of memory consumption, suitability for parallel computation and accuracy. In the first approach, one builds a spatiotemporal trajectory data set by saving trajectory positions over m intermediate times. One then measures pairwise distances using the trapezoidal rule and sparsifies them simultaneously. This can be done effectively using the ExhaustiveSearcher model object in MATLAB or other packages, such as those in [45,46]. This approach is memory consuming but highly parallelizable.

In the second approach, one constructs the similarity matrix without building any spatiotemporal trajectory data set. To this end, one measures pairwise distances concurrent with the advection of particles. Specifically, one defines an extra output argument inside the ODE function that measures and cumulates the pairwise distances over a given time interval.

Compared with the first approach, the second approach is more accurate and more memory efficient. However, its parallel implementation requires communication between processors, which may make the computation prohibitively slow. For this reason, we only employ the second approach in our last example, the ABC flow, and use the first approach otherwise.

A. Periodically forced pendulum

Consider the periodically forced pendulum

$$\begin{aligned}\dot{x}_1 &= x_2,\\ \dot{x}_2 &= -\sin(x_1) + \varepsilon\cos(t).\end{aligned}$$

For $\varepsilon = 0$, the system is integrable with hyperbolic fixed points at $(0,(2m-1)\pi)$ and elliptic fixed points at $(0,2m\pi)$, where $m \in \mathbb{Z}$. As is well known, there are two heteroclinic orbits connecting each successive pair of hyperbolic fixed points, enclosing an elliptic fixed point, which is in turn surrounded by periodic orbits. These periodic orbits appear as closed invariant curves for the Poincaré map $\mathcal{P} := F_0^{2\pi}$. The fixed points of the flow are also fixed points of $\mathcal{P}$.

Kolmogorov-Arnold-Moser (KAM) theory [85] guarantees the survival of most closed invariant sets for $\mathcal{P}$ and $0 < \varepsilon \ll 1$. Increasing the perturbation strength ε further leads to the appearance of resonance islands [86,87] and to the coexistence of regular and chaotic particle trajectories, as one would expect in a turbulent fluid flow containing coherent structures.

Figure 5(b) shows these surviving invariant sets (KAM tori and resonance islands) of the Poincaré map $\mathcal{P}$ obtained for $\varepsilon = 0.4$, obtained from 800 iterations of $\mathcal{P}$. This many iterations are required to obtain continuous-looking boundaries of the various coherent regions. We would like to capture the surviving KAM regions as coherent clusters using Algorithm 1.

To construct the pairwise dynamic distances r_{ij} and subsequent similarity matrix W, we advect 90 000 particles, distributed initially over a uniform grid of 300 × 300 points, from $t_0 = 0$ to $t_1 = 800 \times 2\pi$. The spatial domain ranges from -2.6 to -0.3 in the x_1 direction and from -1.2 to 1.2 in the x_2 direction. We output the trajectory data with 3600 intermediate points, evenly spaced in time. Moreover, we sparsify edges from the complete graph representing a distance greater than $\epsilon = 0.45$.

Figure 6(a) shows the degree of connectivity of graph nodes $\deg(v_i)$ as a scalar field. We refer to this scalar field here and in our later examples as connectivity field. This field looks generally smoother than other diagnostic fields, such as the finite-time Lyapunov exponent [88,89] or finite-size Lyapunov exponent [90,91] fields (see Fig. 6). The smoothness of the connectivity field is the result of two averaging processes that attenuate computational and *in situ* measurement noises. The first averaging process happens as we integrate Euclidean distances between graph nodes over time. The second averaging takes place once we compute d_i, i.e., when summing the edge weights connected to a node v_i.

Figure 7(a) shows the first 20 generalized eigenvalues as a function of their indices. We can see that the first nine eigenvalues are very close to 1, while the tenth has an appreciable difference, creating the largest gap in the eigenvalue

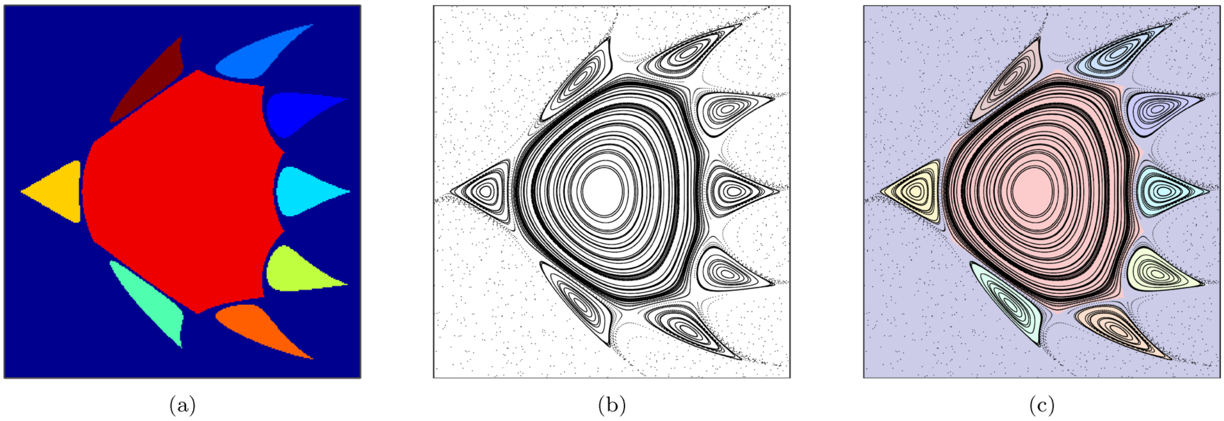


FIG. 5. (a) Ten clusters extracted by K-means clustering from the first nine generalized eigenvectors of the graph Laplacian L for the periodically forced pendulum. The tenth cluster corresponds to the chaotic sea filling the space between elliptic regions. (b) Eight hundred iterations of the Poincaré map for the periodically forced pendulum. (c) Computed clusters, compared with the Poincaré map computed for the same integration time (800 iterates).

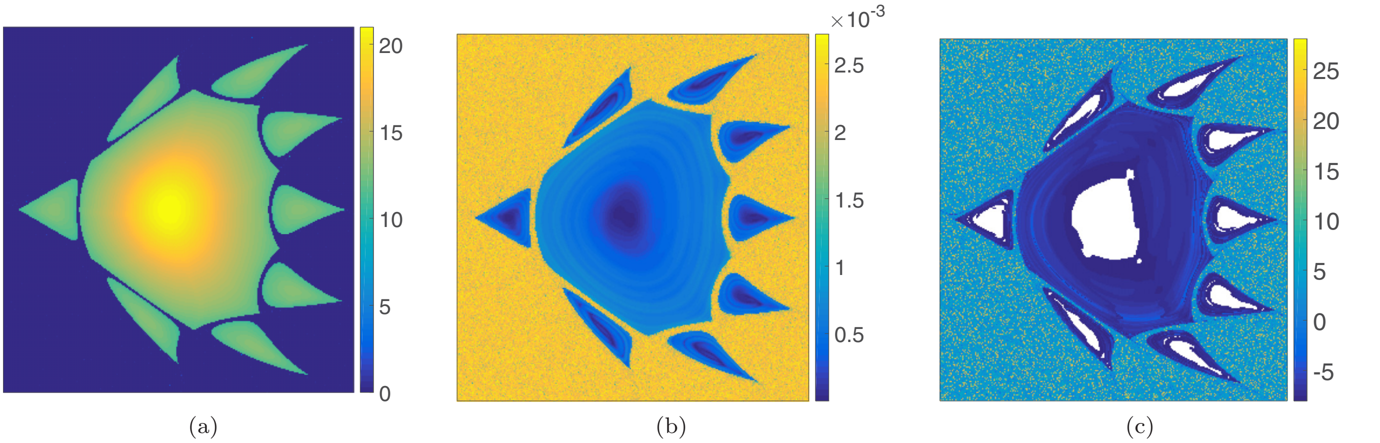


FIG. 6. Comparison of three different diagnostic fields for the periodically forced pendulum. The scalar fields are constructed for the same integration time $T = 800 \times 2\pi$. (a) Forward-time connectivity field. (b) Forward-time FTLE field. (c) Forward-time FSLE field.

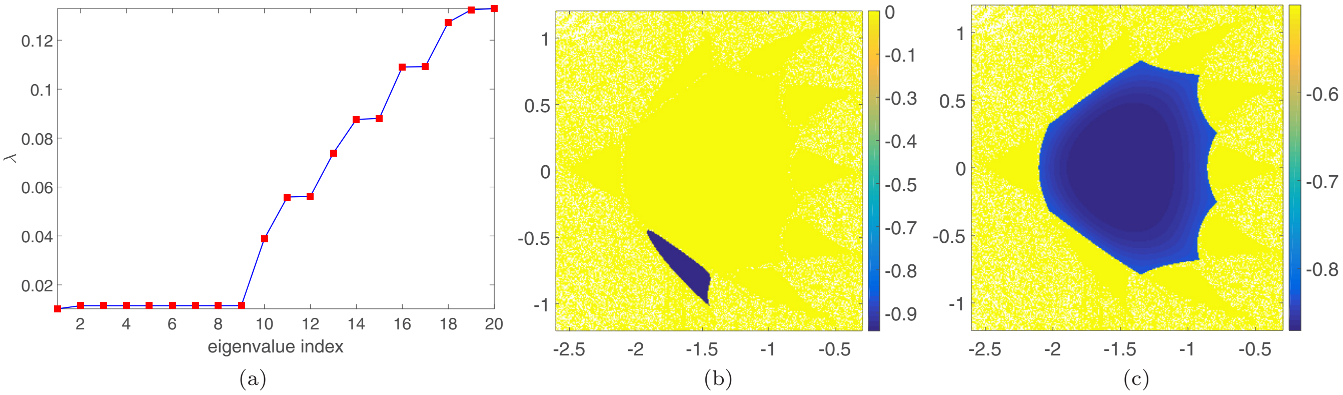
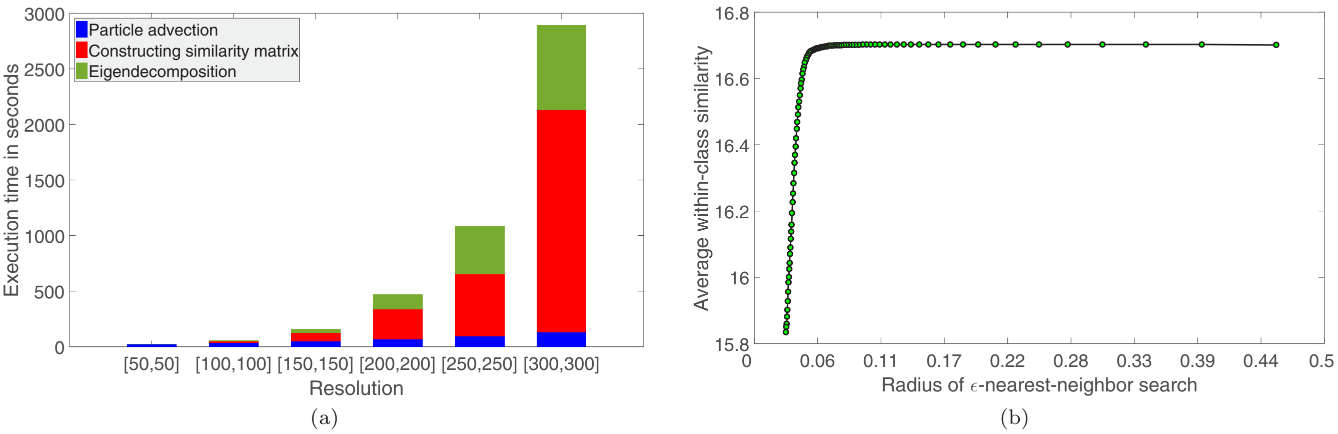


FIG. 7. (a) Sorted generalized eigenvalues for the graph Laplacian L for the periodically forced pendulum. (b) First and (c) ninth generalized eigenvectors of the graph Laplacian L. Isolated points resulting from the graph sparsification are shown in white.



(a)

(b)

FIG. 8. (a) Run times of Algorithm 1 for six different resolutions for the periodically forced pendulum. The run times represent the average CPU times for 300 processors used in parallel in these computations. The computations are performed on a supercomputer with 2.7-GHz Intel Xeon CPUs. (b) Clustering sensitivity with respect to the sparsification radius. The plot shows the average within-class similarities (for nine coherent sets) relative to the ϵ-nearest-neighbor radius used to sparsify the pairwise distances r_{ij}.

plot. This eigengap implies that the first nine eigenvectors are cluster indicators from which coherent structures should be extracted. For example, Figs. 7(b) and 7(c) show the first and ninth generalized eigenvectors of the graph Laplacian L.

Finally, Fig. 5(a) shows the ten clusters extracted by the K-means algorithm from the first nine generalized eigenvectors of graph Laplacian L. The tenth cluster corresponds to the chaotic background filling the space between the coherent clusters. In Fig. 5(c), the extracted clusters are superimposed on the Poincaré map, showing close agreement with the Lagrangian vortices of this example, i.e., the elliptic islands.

Figure 8(a) shows the execution times for three major steps of Algorithm 1 as a function of increasing spatial resolution of the graph nodes. The main computational bottleneck, as shown in the figure, is computing the pairwise distances and subsequently the similarity matrix W. For this purpose, we utilized parallel computing techniques with 300 CPUs, with each processor just computing a few rows and columns of the sparse similarity matrix. Figure 8(a) shows the averaged CPU times spent on each processor on carrying out the particle advection, sparse similarity matrix construction, and eigendecomposition.

Figure 8(b) shows the sensitivity of the clustering results to the choice of the neighborhood radius used to sparsify the pairwise distances r_{ij}. In particular, the figure shows how the averaged within-class similarities of coherent sets change with respect to the choice of neighborhood radius. Figure 8(b) suggests the existence of a critical radius below which the size and shape of clusters can change. This critical radius simply corresponds to a distance where even strong edges within coherent sets are affected by graph sparsification. It is important to choose the sparsification radius such that strong edges will be maintained. As a rule of thumb we set the sparsification radius such that only 5%–10% of the elements in the similarity matrix W will be kept. To estimate such a radius, one can compute the pairwise distances for a subsample of the original graph (e.g., 40 nodes) and choose the sparsification radius accordingly.

B. Quasiperiodic Bickley jet

Next we consider the Bickley jet, an idealized model of a meandering zonal jet flanked above and below by counterrotating vortices [80,81]. This model consists of a steady background flow subject to a time-dependent perturbation. The time-dependent Hamiltonian for this model reads

$$\psi(x,y,t) = \psi_0(y) + \psi_1(x,y,t),$$
$$\psi_0(y) = -U_0 L_0 \tanh\left(\frac{y}{L_0}\right),$$
$$\psi_1(x,y,t) = U_0 L_0 \,\mathrm{sech}^2\left(\frac{y}{L_0}\right) \mathrm{Re}\left[\sum_{n=1}^{3} f_n(t)\exp(ik_n x)\right],$$

where ψ_0 is the steady background flow and ψ_1 is the perturbation. The constants U_0 and L_0 are the characteristic velocity and characteristic length scale, respectively. For the following analysis, we apply the set of parameters used in [80],

$$U_0 = 62.66\ \mathrm{ms}^{-1}, \quad L_0 = 1770\ \mathrm{km}, \quad k_n = 2n/r_0,$$

where $r_0 = 6371$ km is the mean radius of the earth.

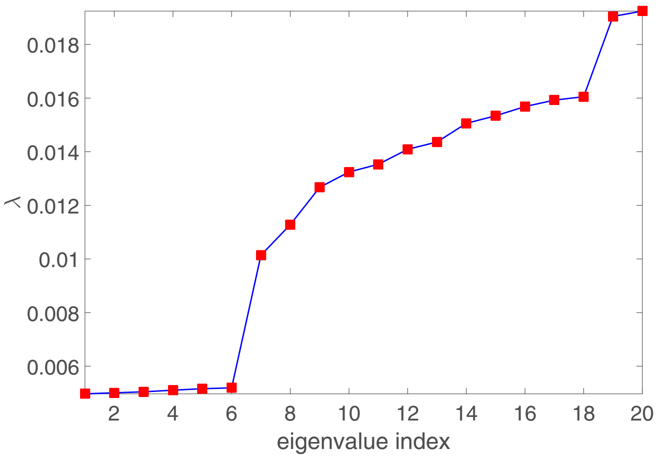
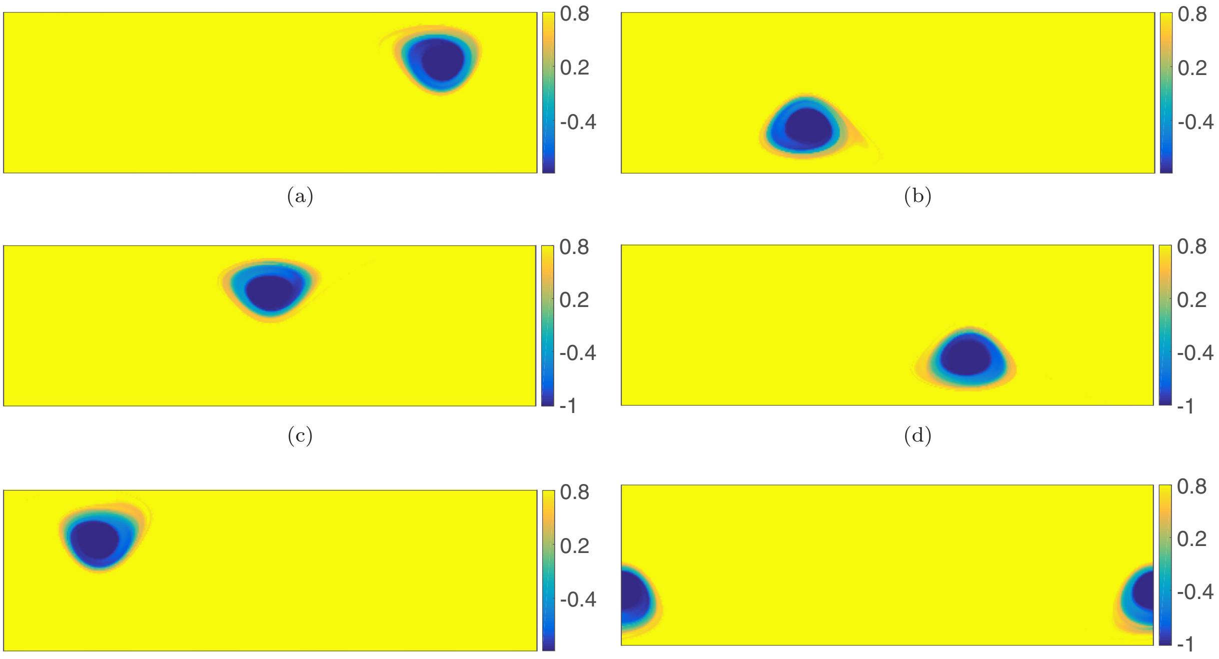


FIG. 9. Sorted generalized eigenvalues for the graph Laplacian L for the quasiperiodic Bickley jet flow.



(a) (b)

(c) (d)

(e) (f)

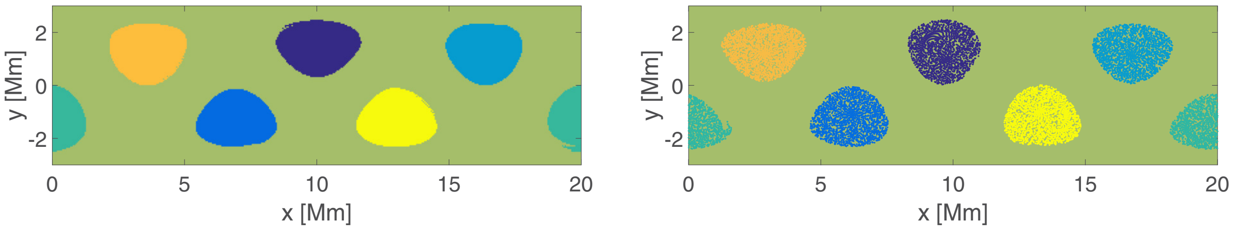
FIG. 10. Leading generalized eigenvectors of the graph Laplacian L for the Bickley jet flow.

For $f_n(t) = \varepsilon_n \exp(-ik_n c_n t)$, the time-dependent part of the Hamiltonian consists of three Rossby waves with wave numbers k_n traveling at speeds c_n. The amplitude of each Rossby wave is determined by the parameters ε_n. Specifically, the parameter values used are $c_1 = 0.1446U_0$, $c_2 = 0.205U_0$, $c_3 = 0.461U_0, l_y = 1.77 \times 10^6, \varepsilon_1 = 0.0075, \varepsilon_2 = 0.15, \varepsilon_3 = 0.3, l_x = 6.371 \times 10^6\pi$, and $k_n = 2n\pi/l_x$.

To construct the dynamic distances r_{ij} and the similarity matrix W, we advect 48 000 particles, distributed initially over a uniform grid of 400 × 120 points, from $t_0 = 0$ to $t = 40$ days. The spatial domain U ranges from 0 to 20 in the x direction and from -3 to 3 in the y direction. We output the trajectory data with 600 intermediate points, evenly spaced in time. Moreover, we sparsify edges from the complete graph representing a distance greater than $\epsilon = 3$.

In Fig. 9 we show the first 20 generalized eigenvalues of the graph Laplacian L with respect to their indices. We can observe that the largest eigengap is between the sixth and seventh generalized eigenvalues, signaling the presence of six coherent clusters in the domain. Hence, we extract seven clusters from the first six generalized eigenvectors shown in Figs. 10(a)–10(f). The last cluster, as described earlier in Sec. II E, corresponds to the incoherent region filling the space between the coherent vortices. The observed fuzziness of the vortex boundary region is due to the fact that coherent and incoherent motion is, on the chosen time interval, not as distinguished as in the forced pendulum example considered in the previous section. After all, this distinction is retrieved from the trajectory data, as opposed to being imposed externally through some threshold, for instance. Interestingly, this dynamic distinction is very clear in the ocean example considered in the next section, which results in very pronounced cluster indicators.

Figure 11(a) shows the identified clusters at the initial time and Fig. 11(b) shows them at the final time, confirming the coherence of extracted vortices over the 40-day period. The complete advection sequence over 40 days is available in the Supplemental Material, movie M1 [92].



(a) (b)

FIG. 11. (a) Seven clusters extracted by K-means clustering from the first six generalized eigenvectors of the graph Laplacian L at initial time $t_0 = 0$. The seventh cluster corresponds to the mixing region filling the space between the coherent clusters. (b) Same clusters advected passively to the final time $t = 40$ days. The complete advection sequence over 40 days is illustrated in the Supplemental Material, movie M1 [92].

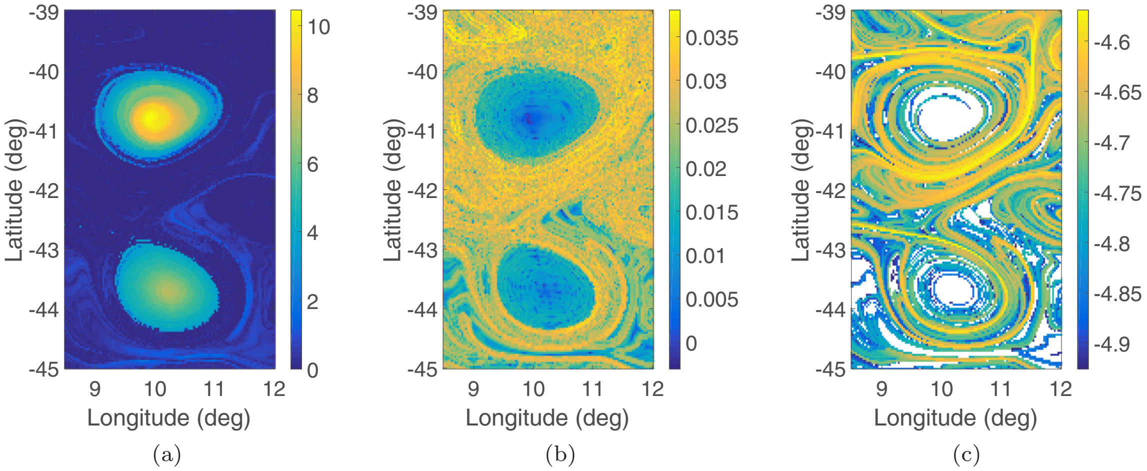


FIG. 12. Comparison of three different diagnostic fields for the ocean data set. The scalar fields are constructed for the same integration time $T = 168$ days. (a) Forward-time connectivity field. (b) Forward-time FTLE field. (c) Forward-time FSLE field.

C. Ocean surface data set

Next, we apply Algorithm 1 to a two-dimensional unsteady velocity data set obtained from satellite altimetry measurements produced by SSALTO/DUACS and distributed by AVISO [93]. The domain of the data set is the Agulhas leakage in the Southern Ocean, characterized by large coherent eddies that pinch off from the Agulhas current of the Indian Ocean.

Here we show how our coherent Lagrangian vortex detection principle uncovers the material eddies over integration time of 168 days, ranging from $t_0 = 11$ January 2006 to $t = 28$ June 2006. The South Atlantic ocean region in question is bounded by longitudes [8.5°E, 12°E] and latitudes [45°S, 39°S]. The region in question is chosen away from the coast so that particle positions will be available for the entire integration time. Otherwise, one has to discard those particles hitting obstacles or the coast at some intermediate times from the computation. We compute the pairwise accumulative distances over a uniform grid of 120×180 points using a trajectory data set composed of 600 evenly spaced intermediate times. We sparsify edges from the complete graph representing a distance greater than $\epsilon = 1$.

Figure 12 compares> the connectivity field with the finite-time Lyapunov exponent (FTLE) and finite-size Lyapunov exponent (FSLE) fields. Note that we view the connectivity field as a simple visualization tool from which one may diagnose the existence of coherent structures before taking the eigendecomposition step.

In Fig. 13(a) we show the first 20 generalized eigenvalues of the graph Laplacian L. We can observe that the largest eigengap exists between the second and third generalized eigenvalues, signaling the presence of two coherent clusters in the domain, which are indicated by the corresponding generalized eigenvectors [see Figs. 13(b) and 13(c)].

Figure 14(a) show the coherent vortices extracted from the first two generalized eigenvectors of the graph Laplacian L at initial time $t_0 = 11$ January 2006 and final time $t = 28$ June 2006, respectively. In Fig. 14(b) we confirm the coherence of extracted vortices by advecting them to the final time $t = 28$ June 2006.

Interestingly, the coherent cluster shown in blue contains isolated points located far away from the cluster core [see Fig. 14(c)]. The presence of isolated points in a given cluster, however, seems to be unphysical due to the continuity of fluid flows. To investigate the true nature of these isolated

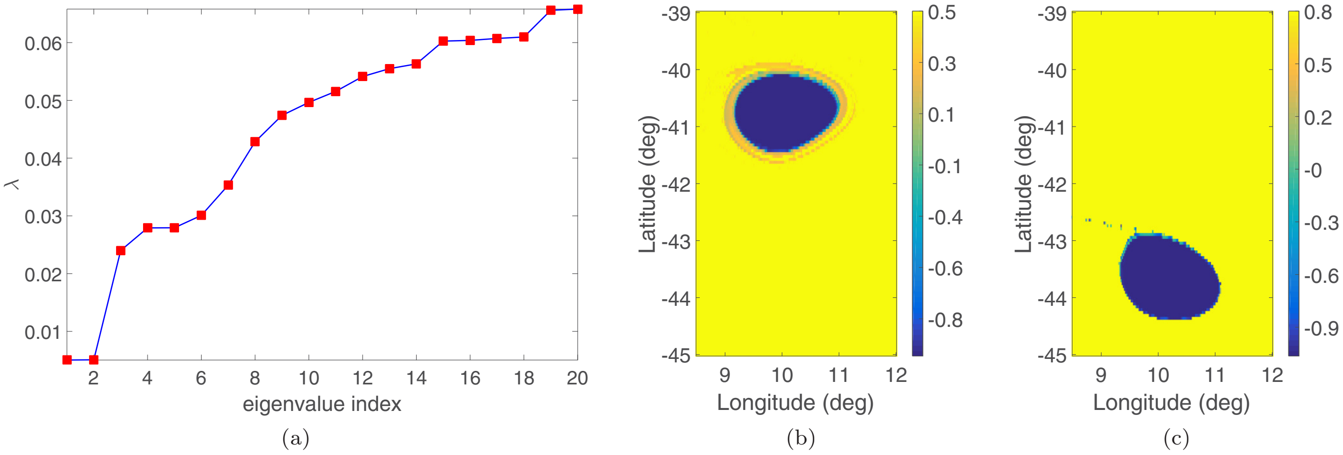


FIG. 13. (a) Sorted generalized eigenvalues for the graph Laplacian L for the ocean data set. (b) and (c) First two generalized eigenvectors.

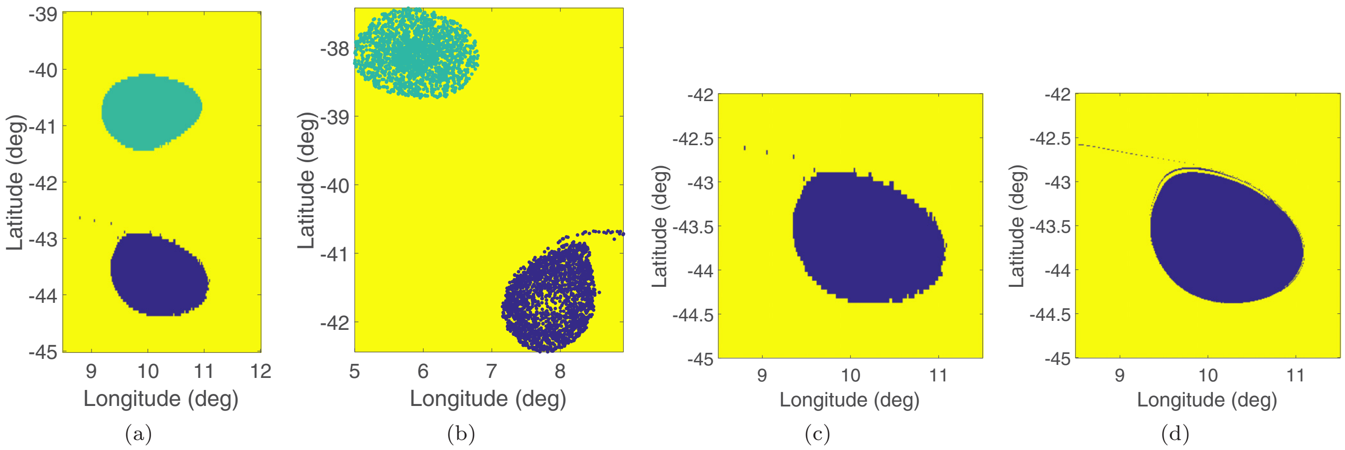


FIG. 14. (a) Coherent vortices at initial time $t_0 = 11$ January 2006. (b) Advected image of the vortices at the final time $t = 28$ June 2006. (c) Magnification of the blue cluster shown in the first panel. The figure shows some isolated points located far from the cluster core. (d) Corresponding cluster obtained from a higher resolution computation, revealing that the previously detected isolated points are part of a narrow fingering emanating from the cluster core. The complete advection sequence over 168 days is illustrated in the Supplemental Material, movies M2 and M3 [92].

points, we repeat our computation with a higher resolution, over a uniform grid of 300 × 300 points, in the range of [8.5°E,12°E] in longitude and [45°S,39°S] in latitude [see Fig. 14(d)]. The higher-resolution computation reveals that the previously detected isolated points are part of a narrow fingering emanating from the core of the blue cluster. This is in line with the known vortex stirring reported by several authors (see [75], for example).

Despite the strange fingering-type appearance, the cluster remains highly coherent over the extraction period of 168 days. The complete advection sequence over 168 days is illustrated in the Supplemental Material, movies M2 and M3 [92].

This example underlines that a Lagrangian vortical region can have an instantaneously nonconvex geometry. It may also, over time, absorb an initial finger-type protrusion and form a convex circular boundary in the end. This illustrates that while requiring convexity [17,31], lack of filamentation [14], or shape coherence [18] of the vortex boundary may yield boundaries meeting high coherence requirements, they will not necessarily identify the largest set of trajectories forming a coherent cluster.

Finally, we repeat our computation with a sparse trajectory data set, composed of 57 particles distributed nonuniformly on an unstructured grid. Here we select the number of intermediate times m and sparsification distance ϵ similar to our earlier computation. Figure 15 shows the clustering result, with Fig. 14(a) shown in the background for comparison.

D. The ABC flow

As a last example we consider the steady ABC flow [83]

$$\dot{x} = A \sin z + C \cos y,$$
$$\dot{y} = B \sin x + A \cos z,$$
$$\dot{z} = C \sin y + B \cos x,$$

an exact solution of Euler's equation. We select the parameter values $A = \sqrt{3}$, $B = \sqrt{2}$, and $C = 1$. This well-studied set of parameter values [15,25,84,94] yields six coherent vortices. We construct a high-resolution graph by selecting a uniform grid of 120 × 120 × 120 points over the spatial domain ranging from 0 to 2π in the x, y, and z directions.

Next we subsample the phase space uniformly on a coarser grid by selecting $q = 1000$ supernodes out of the 120^3 nodes of the original graph and construct the tight similarity matrix $Z \in \mathbb{R}^{q\times n}$, expressing similarity between the q supernodes and the n nodes of the original graph. To construct the tight similarity matrix Z, we measure the dynamic distances in the lifted system, where trajectories can flow out of the 2π cube. Having the similarity matrix Z in hand, we compute the dominant singular values and singular vectors of $\hat{Z} = D_2^{-1/2} Z D_1^{-1/2}$. The left singular eigenvectors are cluster indicators for the reduced graph built upon q supernodes, while the right singular vectors are cluster indicators for the original graph.

As the last step we retrieve seven clusters from six cluster indicators using the K-means algorithm. The last cluster, as

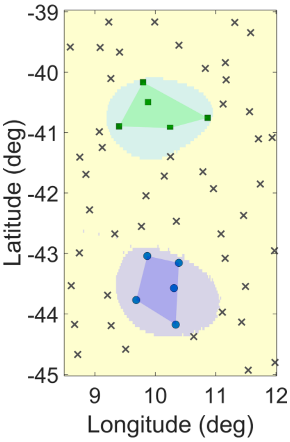
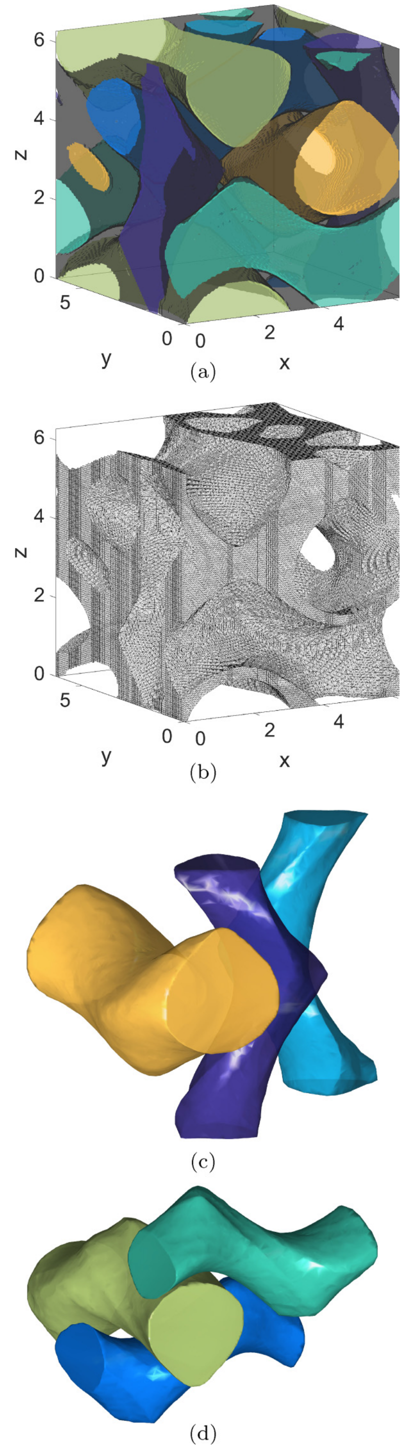


FIG. 15. Coherent vortices are captured at initial time $t_0 = 11$ January 2006, with a sparse trajectory data set. Figure 14(a) is shown in the background for comparison.



(a)

(b)

(c)

(d)

FIG. 16. (a) Seven clusters extracted by K-means clustering ($k = 7$) from the first six eigenvectors of L. The first six clusters correspond to six coherent vortices that were identified earlier in [94]. The chaotic sea between coherent vortices is the seventh cluster and appears as the void between them. (b) Seventh cluster that appears as the chaotic sea between coherent vortices. (c) and (d) Three-dimensional vortices are reconstructed by putting together the coherent cluster pieces.

before, shows the incoherent region filling the space between the coherent clusters or vortices. Figure 16(a) shows the six coherent clusters that are separated by the incoherent cluster. The six clusters capture the six known coherent structures of the ABC flow identified earlier in [94].

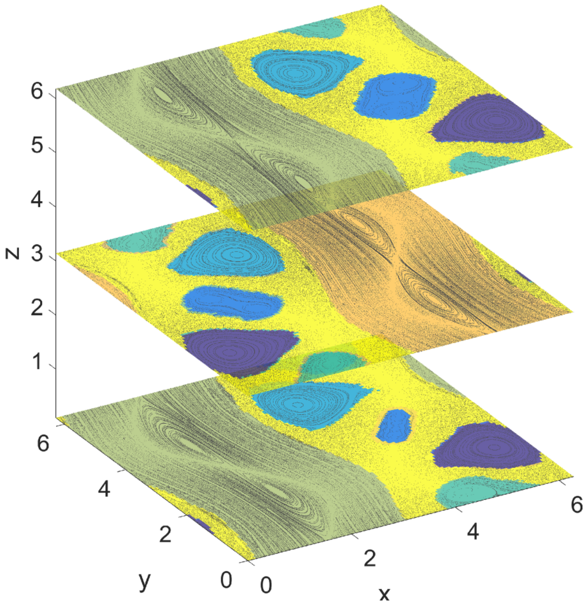


FIG. 17. Coherent vortices extracted by Algorithm 2 are compared with the Poincaré map constructed for integration time $T = 3000$.

Due to the existence of the spatial periodic boundary condition, the coherent vortices are broken into pieces in the initial cubic domain. In Figs. 16(c) and 16(d) we put together the pieces of six coherent vortices and show their full cylindrical geometry. The colors used in Figs. 16(c) and 16(d) are consistent with those in Fig. 16(a). In Fig. 17 the clusters are superimposed on the Poincaré map, showing close agreement between the results of the two approaches.

V. CONCLUSION

We have developed here an approach to locate coherent structures based on spectral graph theory. To identify coherent structures, we measured the pairwise Euclidean distance between Lagrangian trajectories and constructed an undirected weighted graph describing the spatiotemporal evolution of fluid flows. We then identified coherent vortices as clusters of Lagrangian particles remaining close under the flow using two different algorithms. In the first algorithm, we used Shi and Malik's [48] normalized cut to identify coherent vortices whose nodes on graph have large internal (external) coherence (incoherence). We demonstrated the effectiveness of the corresponding Algorithm 1 to detect Lagrangian coherent vortices in periodic, quasiperiodic, and unsteady two-dimensional flows. This includes the determination of the *a priori* unknown number of present vortices in a given domain using the eigengap heuristic.

In Algorithm 2 we applied a recently developed graph subsampling technique [63,64] to handle the memory bottleneck associated with large-scale graphs. We applied Algorithm 2 in our last example, the three-dimensional steady ABC flow, where we succeeded in combining high sampling resolution with computational efficiency.

An advantage of our approach is that it requires a relatively low number of Lagrangian trajectories as input, making it suitable for the analysis of low-resolution trajectory data sets (see also [19,28,29] for similar approaches designed for low numbers of Lagrangian trajectories). Moreover, our method takes advantage of trajectories' intermediate positions, i.e., information that comes in most cases without additional computational cost, e.g., in time-resolved trajectory data sets or numerical integration of velocity data sets or vector fields (see also [28]).

Moreover, we argue that in fluidlike flows coherence-related phenomena can only be conceived in the presence of an incoherent background, which prohibits the partitioning of the fluid domain into purely coherent sets or regions. Here we introduced the definition of incoherent cluster and partitioned the fluid domain into coherent and incoherent clusters, an idea that appears to be missing in other similar approaches [18,27,28].

Finally, we chose spectral clustering as a tool of choice due to its solid mathematical foundation and its performance. However, other clustering algorithms such as density-based clustering approaches [95] that can incorporate the definition of noise or incoherent cluster may be used alternatively. Incorporating other clustering algorithms and comparing their performance for the purpose of Lagrangian coherent vortex identification remains a viable future research direction. Moreover, further work is needed to connect graph properties with physical or mechanical quantities characterizing the fluid motion, beyond the heuristic and numerical arguments given in Secs. I and IV.

APPENDIX A: APPROXIMATING NCut

In this appendix we recall how the NCut problem can be solved for the case $k = 2$, which partitions the graph into two disjoint sets. We follow closely the arguments of [38,48].

Our goal is to solve the optimization problem

$$\min_{A\in V} \mathrm{NCut}(A,\bar{A}). \tag{A1}$$

First, we rewrite the problem in a more convenient form. Given a subset $A \subset V$ we define the cluster indicator vector $f = (f_1,\dots,f_n)^\top \in \mathbb{R}^n$ with entries

$$f_i = \begin{cases} \sqrt{\frac{\mathrm{vol}(\bar{A})}{\mathrm{vol}(A)}} & \text{if } v_i \in A \\ -\sqrt{\frac{\mathrm{vol}(A)}{\mathrm{vol}(\bar{A})}} & \text{if } v_i \in \bar{A}. \end{cases} \tag{A2}$$

Now Eq. (A1) can be conveniently rewritten using the graph Laplacian L as

$$\min_{A} f^\top L f \quad \text{subject to } f \text{ as in } (A2),\ Df \perp 1,\ f^\top D f = \mathrm{vol}(V).$$

This is a Rayleigh quotient, and minimizing it is of NP-hard complexity, since we have constrained f to take on only discrete values as described in (A2). We relax the problem by allowing f to take arbitrary real values (l_2 relaxation) to obtain

$$\min_{f\in R^n} f^\top L f \quad \text{subject to } Df \perp 1,\ f^\top D f = \mathrm{vol}(V).$$

After substitution of $g := D^{1/2} f$, the problem converts to

$$\min_{g\in R^n} g^\top D^{-1/2} L D^{-1/2} g$$
$$\text{subject to } g \perp D^{1/2} 1,\ \|g\|^2 = \mathrm{vol}(V),$$

to which the standard Rayleigh-Ritz theorem applies such that its solution g is given by the second eigenvector of $D^{-1/2} L D^{-1/2}$. Substituting again $f = D^{-1/2} g$, we see that f is the second generalized eigenvector of $Lu = \lambda Du$.

Similarly, we can decompose the graph into k partitions by using cluster indicator vectors $h_j = (h_{1,j},\dots,h_{n,j})^\top$,

$$h_{i,j} = \begin{cases} \frac{1}{\sqrt{\mathrm{vol}(A_j)}} & \text{if } v_i \in A_j \quad \text{for } i = 1,\dots,n;\ j = 1,\dots,k \\ 0 & \text{otherwise.} \end{cases} \tag{A3}$$

Then we set the matrix $H \in R^{n\times k}$ as the matrix containing those k cluster indicator vectors as columns. Observe that the columns in H are orthonormal to each other, that is, $H^\top H = I$, and $h_i^\top L h_i = \mathrm{cut}(A_i,\bar{A}_i)/\mathrm{vol}(A_i)$. So we can write the problem of minimizing NCut as

$$\min_{A_1,\dots,A_k} \mathrm{Tr}(H^\top L H) \quad \text{subject to } H^\top D H = I,\ H \text{ as in } (A3).$$

Relaxing the discreteness condition and substituting $T = D^{1/2} H$, we obtain the relaxed problem

$$\min_{T\in\mathbb{R}^{n\times k}} \mathrm{Tr}(T^\top D^{-1/2} L D^{-1/2} T) \quad \text{subject to } T^\top T = I.$$

Again, this is the standard trace minimization problem, which is solved by the matrix T composed of the first k eigenvectors of $D^{-1/2} L D^{-1/2}$ as columns. Substituting again $H = D^{-1/2} T$, we see that the solution H consists of the first k generalized eigenvectors of $Lu = \lambda Du$. This yields the normalized spectral clustering algorithm according to [48].

APPENDIX B: BIPARTITE SPECTRAL GRAPH PARTITIONING

In this appendix we briefly recall how spectral clustering is applied to bipartite graphs. This specification is also referred to as spectral coclustering [65,66] and is presented here in the subsampling terminology introduced in Sec. II G It applies, however, verbatim to the bipartite transfer-operator graph.

Let $Z \in \mathbb{R}^{q\times n}$ be a tight similarity matrix between the n graph nodes and the q supernodes. To explicitly capture the node-supernode relationship, we consider a bipartite graph $G_{\mathcal{B}} = (V_{\mathcal{B}}, E_{\mathcal{B}}, W_{\mathcal{B}})$ whose nodes can be divided into two disjoint sets A and B such that internal edges all have zero weights, i.e., $w_{ij}^{\mathcal{B}} = 0$ if $v_i^{\mathcal{B}}, v_j^{\mathcal{B}} \in A$ or $v_i^{\mathcal{B}}, v_j^{\mathcal{B}} \in B$. The similarity matrix of the whole bipartite graph $W_{\mathcal{B}}$ then reads

$$W_{\mathcal{B}} = \begin{pmatrix} 0 & Z^\top \\ Z & 0 \end{pmatrix}. \tag{B1}$$

To partition the bipartite graph, the optimization task can be formalized as a generalized eigenvalue problem with suitable relaxation (see Sec. V)

$$L_{\mathcal{B}} q = (D_{\mathcal{B}} - W_{\mathcal{B}}) q = \lambda D_{\mathcal{B}} q, \tag{B2}$$

where $D_{\mathcal{B}}$ is the degree matrix of $W_{\mathcal{B}}$.

Substituting (B1) in (B2), we get

$$\begin{pmatrix} 0 & Z^\top \\ Z & 0 \end{pmatrix}\begin{pmatrix} q_1 \\ q_2 \end{pmatrix} = (1-\lambda)\begin{pmatrix} D_1 & 0 \\ 0 & D_2 \end{pmatrix}\begin{pmatrix} q_1 \\ q_2 \end{pmatrix}, \tag{B3}$$

where D_1 is an $n \times n$ diagonal matrix whose entries are column sums of Z and D_2 is an $q \times q$ diagonal matrix whose entries are row sums of Z. Breaking the block matrix form into parts, Eq. (B3) can be rewritten as

$$Z^\top q_2 = (1-\lambda)D_1 q_1,$$
$$Z q_1 = (1-\lambda)D_2 q_2.$$

Let $b = D_1^{1/2} q_1$ and $a = D_2^{1/2} q_2$ and after variable substitution we have

$$D_1^{-1/2} Z^\top D_2^{-1/2} a = (1-\lambda) b,$$
$$D_2^{-1/2} Z D_1^{-1/2} b = (1-\lambda) a.$$

These equations define the singular value decomposition of the normalized matrix $\hat{Z} = D_2^{-1/2} Z D_1^{-1/2}$. In particular, a and b are the left and right singular vectors, respectively, and $1-\lambda$ is the corresponding singular value [66].

[1] J. C. McWilliams, J. Fluid Mech. **146**, 21 (1984).

[2] A. Provenzale, Annu. Rev. Fluid Mech. **31**, 55 (1999).

[3] G. Haller, Annu. Rev. Fluid Mech. **47**, 137 (2015).

[4] Z. Zhang, W. Wang, and B. Qiu, Science **345**, 322 (2014).

[5] C. Dong, J. C. McWilliams, Y. Liu, and D. Chen, Nat. Commun. **5**, 3294 (2014).

[6] M. Wilczek, J. Fluid Mech. **784**, 1 (2015).

[7] M. E. O'Neill, K. A. Emanuel, and G. R. Flierl, Nat. Geosci. **8**, 523 (2015).

[8] J. C. R. Hunt, A. A. Wray, and P. Moin, in *Proceedings of the 1988 Summer Program* (1988), pp. 193–208, http://adsabs.harvard.edu/abs/1988stun.proc..193H.

[9] J. Zhou, R. J. Adrian, S. Balachandar, and T. Kendall, J. Fluid Mech. **387**, 353 (1999).

[10] G. Haller, J. Fluid Mech. **525**, 1 (2005).

[11] G. Boffetta, G. Lacorata, G. Radaelli, and A. Vulpiani, Physica D **159**, 58 (2001).

[12] J. B. Weiss and A. Provenzale, *Transport and Mixing in Geophysical Flows* (Springer, Berlin, 2008).

[13] T. Peacock and J. Dabiri, Chaos **20**, 017501 (2010).

[14] G. Haller and F. J. Beron-Vera, J. Fluid Mech. **731**, R4 (2013).

[15] D. Blazevski and G. Haller, Physica D **273–274**, 46 (2014).

[16] M. Farazmand and G. Haller, Physica D **315**, 1 (2016).

[17] G. Haller, A. Hadjighasem, M. Farazmand, and F. Huhn, J. Fluid Mech. **795**, 136 (2016).

[18] T. Ma and E. M. Bollt, SIAM J. Appl. Dyn. Syst. **13**, 1106 (2014).

[19] G. Froyland and O. Junge, Chaos **25**, 087409 (2015).

[20] G. Froyland, N. Santitissadeekorn, and A. Monahan, Chaos **20**, 043116 (2010).

[21] G. Froyland, Physica D **250**, 1 (2013).

[22] G. Froyland and K. Padberg-Gehle, Ergod. Theor. Open Dyn. Coherent Struct. **70**, 171 (2014).

[23] I. Mezić, S. Loire, V. A. Fonoberov, and P. Hogan, Science **330**, 486 (2010).

[24] I. I. Rypina, S. E. Scott, L. J. Pratt, and M. G. Brown, Nonlinear Process. Geophys. **18**, 977 (2011).

[25] M. Budišić and I. Mezić, Physica D **241**, 1255 (2012).

[26] B. S. Everitt, S. Landau, M. Leese, and D. Stahl, *Cluster Analysis* (Wiley, New York, 2011).

[27] E. Ser-Giacomi, V. Rossi, C. López, and E. Hernández-García, Chaos **25**, 036404 (2015).

[28] G. Froyland and K. Padberg-Gehle, Chaos **25**, 087406 (2015).

[29] M. O. Williams, I. I. Rypina, and C. W. Rowley, Chaos **25**, 087408 (2015).

[30] R. Banisch and P. Koltai, arXiv:1603.04709.

[31] J. Pratt, A. Busse, W. Mueller, S. Chapman, and N. Watkins, arXiv:1408.5706.

[32] D. Karrasch, F. Huhn, and G. Haller, Proc. R. Soc. A **471**, 20140639 (2014).

[33] S. Kisilevich, F. Mansmann, M. Nanni, and S. Rinzivillo, *Data Mining and Knowledge Discovery Handbook* (Springer, Boston, 2010), p. 855.

[34] T. Ma and E. M. Bollt, Int. J. Bifurcat. Chaos **23**, 1330026 (2013).

[35] C. Truesdell and W. Noll, in *The Non-Linear Field Theories of Mechanics*, edited by S. S. Antman (Springer, Berlin, 2004), p. 1.

[36] F. Chung, *Spectral Graph Theory* (American Mathematical Society, Providence, 1997), Vol. 92.

[37] W. Y. Chen, Y. Q. Song, H. J. Bai, C. J. Lin, and E. Y. Chang, IEEE Trans. Pattern Anal. Mach. Intell. **33**, 568 (2011).

[38] U. von Luxburg, Stat. Comput. **17**, 395 (2007).

[39] D. R. Karger, Math. Operat. Res. **24**, 383 (1999).

[40] A. A. Benczúr and D. R. Karger, in *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing, STOC '96* (ACM New York, NY, USA, 1996), pp. 47–55.

[41] D. A. Spielman and N. Srivastava, SIAM J. Comput. **40**, 1913 (2011).

[42] J. W. Jaromczyk and G. T. Toussaint, Proc. IEEE **80**, 1502 (1992).

[43] N. Ahuja, IEEE Trans. Pattern Anal. Mach. Intell. **PAMI-4**, 336 (1982).

[44] K. R. Gabriel and R. R. Sokal, Syst. Biol. **18**, 259 (1969).

[45] V. Garcia, E. Debreuve, and M. Barlaud, in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops* (IEEE, Piscataway, 2008), p. 1.

[46] M. Muja and D. G. Lowe, in *Proceedings of the Fourth International Conference on Computer Vision Theory and Applications, Visapp 2009* (VISAPP, Setúbal, 2009), Vol. 1, pp. 331–340.

[47] M. Stoer and F. Wagner, J. ACM **44**, 585 (1997).

[48] J. B. Shi and J. Malik, IEEE Trans. Pattern Anal. Mach. Intell. **22**, 888 (2000).

[49] L. Hagen and A. B. Kahng, IEEE Trans. Comput.-Aid. Design Integr. Circ. Syst. **11**, 1074 (1992).

[50] C. H. Q. Ding, X. F. He, H. Y. Zha, M. Gu, and H. D. Simon, in *Proceedings of the IEEE International Conference on Data Mining* (IEEE, Piscataway, 2001), p. 107.

[51] J. Cheeger, Prob. Anal. **625**, 195 (1970).

[52] D. Wagner and F. Wagner, in *Proceedings of the 18th International Symposium on Mathematical Foundations of Computer Science* (Springer, Berlin, 1993), p. 744.

[53] R. Bhatia, *Matrix Analysis* (Springer, Berlin, 1997), Vol. 169.

[54] M. Dellnitz and O. Junge, SIAM J. Numer. Anal. **36**, 491 (1999).

[55] P. Deuflhard, W. Huisinga, A. Fischer, and C. H. Schütte, Linear Algebra Appl. **315**, 39 (2000).

[56] G. Froyland and M. Dellnitz, SIAM J. Sci. Comput. **24**, 1839 (2003).

[57] S. Sarkar and A. Dong, Phys. Rev. E **83**, 046114 (2011).

[58] J. Cai and S. Osher, UCLA Report No. **5**, 2010 (unpublished).

[59] S. P. Lloyd, IEEE Trans. Inf. Theory **28**, 129 (1982).

[60] C. Fowlkes, S. Belongie, F. Chung, and J. Malik, IEEE Trans. Pattern Anal. Mach. Intell. **26**, 214 (2004).

[61] B. Chen, B. Gao, T. Y. Liu, Y. F. Chen, and W. Y. Ma, in *Proceedings of the 17th European Conference on Machine Learning* (Springer, Berlin, 2006), p. 590.

[62] T.-Y. Liu, H.-Y. Yang, X. Zheng, T. Qin, and W.-Y. Ma, in *Advances in Information Retrieval*, Lecture Notes in Computer Science, edited by G. Amati, C. Carpineto, and G. Romano (Springer, Berlin, Heidelberg, 2007), Vol. 4425, pp. 319–330.

[63] D. Cai and X. Chen, IEEE Trans. Cybern. **45**, 1669 (2015).

[64] J. Liu, C. Wang, M. Danilevsky, and J. Han, in *Proceedings of the 23rd International Joint Conference on Artificial Intelligence* (AAAI, Menlo Park, 2013), p. 1486.

[65] I. S. Dhillon, in *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (ACM, New York, 2001), p. 269.

[66] H. Zha, X. He, C. Ding, H. Simon, and M. Gu, in *Proceedings of the Tenth International Conference on Information and Knowledge Management* (ACM, New York, 2001), p. 25.

[67] E. M. Bollt and N. Santitissadeekorn, *Applied and Computational Measurable Dynamics* (Society for Industrial and Applied Mathematics, Philadelphia, 2013).

[68] P. Deuflhard and M. Weber, Linear Algebra Appl. **398**, 161 (2005).

[69] P. K. Chan, M. D. F. Schlag, and J. Y. Zien, IEEE Trans. Comput.-Aid. Design Integr. Circ. Syst. **13**, 1088 (1994).

[70] A. Y. Ng, M. I. Jordan, and Y. Weiss, in *Advances in Neural Information Processing Systems*, edited by T. G. Dietterich, S. Becker, and Z. Ghahramani (MIT Press, Cambridge, MA, 2002), Vol. 14, pp. 849–856.

[71] J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms* (Kluwer, Norwell, 1981).

[72] J. C. Dunn, J. Cybernetics **3**, 32 (1973).

[73] O. Maimon and L. Rokach, *Data Mining and Knowledge Discovery Handbook* (Springer, Berlin, 2005), Vol. 2.

[74] A. K. Jain, M. N. Murty, and P. J. Flynn, ACM Comput. Surv. **31**, 264 (1999).

[75] H. Aref, J. Fluid Mech. **143**, 1 (1984).

[76] R. N. Dave, Pattern Recogn. Lett. **12**, 657 (1991).

[77] D. Arthur and S. Vassilvitskii, in *Proceedings of the 22nd Annual Symposium on Computational Geometry* (ACM, New York, 2006), p. 144.

[78] R. L. F. Cordeiro, C. Faloutsos, and C. T. Júnior, *Data Mining in Large Sets of Complex Data* (Springer, London, 2013).

[79] R. Winkler, F. Klawonn, and R. Kruse, in *Challenges at the Interface of Data Analysis, Computer Science, and Optimization: Proceedings of the 34th Annual Conference of the Gesellschaft für Klassifikation e. V., Karlsruhe, 2010*, edited by W. Gaul, A. Geyer-Schulz, L. Schmidt-Thieme, and J. Kunze (Springer, Berlin, 2012), p. 79.

[80] I. I. Rypina, M. G. Brown, F. J. Beron-Vera, H. Kocak, M. J. Olascoaga, and I. A. Udovydchenkov, J. Atmos. Sci. **64**, 3595 (2007).

[81] D. Del-Castillo-Negrete and P. J. Morrison, Phys. Fluids A **5**, 948 (1993).

[82] L. L. Fu, D. B. Chelton, P. Y. Le Traon, and R. Morrow, Oceanography **23**, 14 (2010).

[83] V. I. Arnold, J. Appl. Math. Mech. **30**, 223 (1966).

[84] G. Froyland and K. Padberg, Physica D **238**, 1507 (2009).

[85] V. I. Arnold, *Mathematical Methods of Classical Mechanics* (Springer, Berlin, 1989).

[86] V. I. Arnold, V. V. Kozlov, and A. I. Neishtadt, *Mathematical Aspects of Classical and Celestial Mechanics* (Springer Science & Business Media, Berlin, 2007), Vol. 3.

[87] G. D. Birkhoff, Trans. Am. Math. Soc. **14**, 14 (1913).

[88] G. Haller and G. Yuan, Physica D **147**, 352 (2000).

[89] G. Haller, Physica D **149**, 248 (2001).

[90] E. Aurell, G. Boffetta, A. Crisanti, G. Paladin, and A. Vulpiani, J. Phys. A: Math. Gen. **30**, 1 (1997).

[91] V. Artale, G. Boffetta, A. Celani, M. Cencini, and A. Vulpiani, Phys. Fluids **9**, 3162 (1997).

[92] See Supplemental Material at http://link.aps.org/supplemental/10.1103/PhysRevE.93.063107 for the advection of Lagrangian vortices extracted for the Bickley jet model and for the ocean surface data set (low- and high-resolution computations).

[93] P.-Y. Le Traon, F. Nadal, and N. Ducet, J. Atmos. Ocean. Technol. **15**, 522 (1998).

[94] T. Dombre, U. Frisch, J. M. Greene, M. Henon, A. Mehr, and A. M. Soward, J. Fluid Mech. **167**, 353 (1986).

[95] M. Ester, H. P. Kriegel, J. Sander, and X. Xiaowei, in *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, KDD-96* (AAAI, Menlo Park, CA, 1996), pp. 226–231.