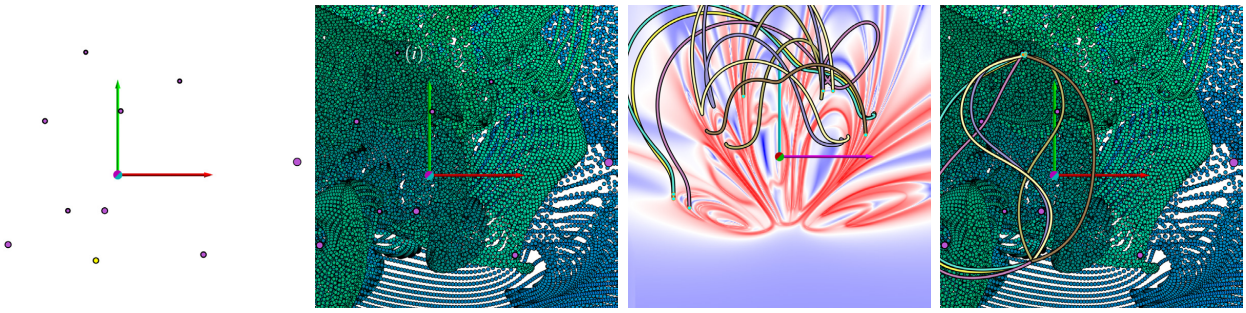
Topological Analysis of Inertial Dynamics

Antoni Sagristà, Stefan Jordan, Andreas Just, Fabio Dias, Luis Gustavo Nonato, and Filip Sadlo



(a) initial position (b) final position (c) initial velocity (d) final position

Fig. 1. Analysis of inertial dynamics at the example of the 2D Nine-Body example. (a) Nine stationary bodies (purple) inducing gravitational field. The initial position of our 400^2 inertial particles has been manually selected (yellow point), whereas their initial velocity is given by a uniform sampling of the initial velocity space (c). (b) Positions of the samples after inertial transport for time T. We select a small region of interest (i) that we would like to reach from the initial position. This provides the initial velocities (cyan dots in (c)) that lead to the desired region of interest. (c) Inertial trajectories in velocity space, together with phase-space finite-time Lyapunov exponent (color-coded), providing the topological structure and thus regions of similar inertial dynamics. (d) The trajectories in position space show how the desired location can be reached.

Abstract—Traditional vector field visualization has a close focus on velocity, and is typically constrained to the dynamics of massless particles. In this paper, we present a novel approach to the analysis of the force-induced dynamics of inertial particles. These forces can arise from acceleration fields such as gravitation, but also be dependent on the particle dynamics itself, as in the case of magnetism. Compared to massless particles, the velocity of an inertial particle is not determined solely by its position and time in a vector field. In contrast, its initial velocity can be arbitrary and impacts the dynamics over its entire lifetime. This leads to a four-dimensional problem for 2D setups, and a six-dimensional problem for the 3D case. Our approach avoids this increase in dimensionality and tackles the visualization by an integrated topological analysis approach. We demonstrate the utility of our approach using a synthetic time-dependent acceleration field, a system of magnetic dipoles, and N-body systems both in 2D and 3D.

Index Terms—Visualization of inertial dynamics, N-body systems, magnetism, acceleration.

1 Introduction

Our everyday life and large parts of the universe are dominated by masses, and forces acting upon them. In the continuous setup, prominent examples are acceleration caused by gravitation between bodies, electrostatics, and magnetism. Some of these accelerations can be represented as time-dependent vector fields. However, most traditional techniques for vector field visualization have either instantaneous (local) scope, or are based on the kinematics of massless particles, and thus cannot provide appropriate insight into the dynamics of inertial particles. Beyond that, phenomena like magnetic interaction cannot be represented by vector fields in terms of acceleration.

Topological analysis of vector fields is motivated by the aim to separate their spatiotemporal domain into regions of qualitatively different behavior. For steady vector fields, this is typically achieved by extraction of streamlines that converge in forward or reverse time direction to isolated zeros of the vector field, known as separatrices and critical points. For time-dependent vector fields, Lagrangian coherent structures, which can be obtained as ridges in the finite-time Lyapunov exponent (FTLE) field, are a counterpart to separatrices, separating regions of qualitatively different behavior for a prescribed time interval.

In this paper, we present a technique for the analysis of the inertial dynamics of point masses. In contrast to traditional massless particles, whose velocity is given by the vector field at the respective position and time, velocity of inertial particles is also part of the underlying initial value problem (IVP). In other words, instead of solving an IVP in space only, it needs to be solved in phase space, which includes the degrees of freedom of velocity. This leads to an increase of dimension by a factor of two, leading to four-dimensional problems for 2D spatial problems, and six-dimensional problems for 3D spatial configurations. We avoid the difficulties with higher-dimensional visualization by separating position and velocity in our approach, which is appropriate for practical applications since position and velocity play different roles.

The contributions of this paper include:

- A counterpart to the finite-time Lyapunov exponent for the analysis of arbitrary inertial dynamics in phase space,
- derived concepts that constrain initial values for analysis,
- decomposition into spread due to velocity and position,
- dimensional stacking for phase-space navigation, and
- a concept to analyze the multiplicity in the underlying maps.

2 Related Work

The most closely related works we are aware of are those by Sapsis and Haller, Peng and Dabiri, and Günther and Theisel, all in the field of inertial particles driven by fluid flow. Studying the flow-induced dynamics of inertial particles is a rather evolved topic with various applications from, e.g., meteorology [23], biology [22], and multi-

- *A. Sagristà, S. Jordan, and F. Sadlo are with Heidelberg University, Germany. A. Just is with ZAH at Heidelberg University, Germany. E-mail: toni.sagrista@iwr.uni-heidelberg.de, {jordan, just}@ari.uni-heidelberg.de, sadlo@uni-heidelberg.de.*
- *F. Dias and L. G. Nonato are with Universidade de São Paulo, São Carlos, Brazil. E-mail: {fabio, gnonato}@icmc.usp.br.*

Manuscript received xx xxx. 201x; accepted xx xxx. 201x. Date of Publication xx xxx. 201x; date of current version xx xxx. 201x. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org. Digital Object Identifier: xx.xxxx/TVCG.201x.xxxxxxx

component or multi-phase flow [19]. Recent works outside visualization include those by Sapsis and Haller [22], and Peng and Dabiri [17]. Sudharsan al. [25] gives a good introduction into this field. In visualization, the initial work of Günther et al. [5] provided techniques and introduced new concepts for integral curves of flow-induced motion of inertial particles. Subsequently, Günther and Theisel extended the concept of vortex core lines to flow-induced motion of inertial particles [6]. More recently, they presented a counterpart [8] to vector field topology for steady-state flow-induced motion of inertial particles, a respective concept [7] for time-dependent flow inspired by the FTLE with focus on the separation of flow-induced inertial trajectories due to mass, and a solution [9] to the demanding source inversion problem in flow-induced transport of inertial particles. Also recently, Garaboa-Paz and Pérez-Muñuzuri [4] extended FTLE for flow-induced inertial trajectories in incompressible flow to phase space, i.e., they investigate the impact of the variation of initial velocity, similar to our approach but as a whole (without separating position and velocity as we do). Although all these works analyze the motion of inertial particles, they are all defined in terms of the dynamics of inertial particles induced by fluid flow. This motion is obtained by an empirical model ([5], equation 1), defining acceleration as a function of the velocity of the particle, flow field velocity, and other accelerations. Thus, the problem of motion of flow-induced inertial particles is a special case of the problems that we address with our technique, i.e., we have no specific constraints on the acceleration of inertial particles. Also, in all these previous works (except for the topology approach [8] and the phase space approach [4]), initial velocity is assumed constant, which does not make it necessary to treat the problem in $2n$-dimensional phase space, in contrast to the problems that we aim at.

Regarding topological analysis of time-dependent vector fields, the flow map-based formulation of the FTLE due to Haller [10] provides a basis for our approach. For further related work in this field, we would like to point the reader to the survey by Pobitzer et al. [18] and to recent advances in streak-based topology [21, 26, 16].

Multidimensional projection (MP) methods [15] have been one of the main alternatives to visualize data residing in spaces with dimension larger than three, and could thus provide solutions to visualize the $2n$-dimensional phase-space finite-time Lyapunov exponent field that we employ in this paper. Projection methods aim to map data to a visual space such that distances are preserved as much as possible, thus making possible the visual analysis of neighborhood structures present in the original data. In fact, in the context of visualization, MP techniques have mainly been used for the visual inspection of clusters and their properties [11], since the analysis of more complex structures is not so straightforward with MP methods. As we will show by an example, however, MP techniques are difficult to apply and even more difficult to interpret in the context of phase space of inertial particles.

3 Method

We would like to refer the reader to the video accompanying this work for a quick overview of the method and its implementation.

3.1 Inertial Dynamics

The subject of visualization in our approach is inertial dynamics in terms of accelerations. One source of accelerations are time-dependent acceleration fields

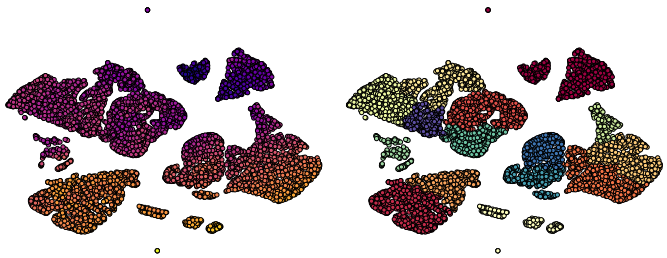
$$\mathbf{a}(\mathbf{x},t) = (a_1(x_1,\dots,x_n,t),\dots,a_n(x_1,\dots,x_n,t))^\top \tag{1}$$

assigning each point $\mathbf{x} := (x_1,\dots,x_n)^\top$ at time t in the domain $\Omega \subseteq \mathbb{R}^n \times \mathbb{R}$ a vector $\mathbf{a} \in \mathbb{R}^n$ with accelerations a_i, $i = 1,\dots,n$ in the respective dimensions. The trajectory $\mathbf{x}(t)$ is fully determined by an IVP starting at time t_0 at initial position $\mathbf{x}_0 := \mathbf{x}(t_0)$ with initial velocity

$$\dot{\mathbf{x}}_0 := \dot{\mathbf{x}}(t_0) = \left(\frac{\mathrm{d}x_1(t)}{\mathrm{d}t},\dots,\frac{\mathrm{d}x_n(t)}{\mathrm{d}t}\right)^\top\Bigg|_{t=t_0}. \tag{2}$$

We first reformulate the IVP in phase space:

$$\boldsymbol{\xi}(t_0) := \begin{pmatrix}\mathbf{x}_0\\ \dot{\mathbf{x}}_0\end{pmatrix}, \quad \frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{\xi}(t) = \begin{pmatrix}\dot{\mathbf{x}}(t)\\ \ddot{\mathbf{x}}(t)\end{pmatrix} = \begin{pmatrix}\dot{\mathbf{x}}(t)\\ \mathbf{a}(\mathbf{x}(t),t)\end{pmatrix}, \tag{3}$$



(a) (b)

Fig. 2. Ridge points mapped from 4D to 2D using t-SNE projection. (a) Neighbor points in 4D are close in the projection. (b) Agglomerative clustering cannot provide a direct notion of the original separating structures in 4D phase space.

with $\boldsymbol{\xi}(t)$ representing the path of a point mass in phase space. The resulting integral formulation

$$\boldsymbol{\xi}(t) = \boldsymbol{\xi}(t_0) + \int_{t_0}^{t} \begin{pmatrix}\dot{\mathbf{x}}(\tau)\\ \mathbf{a}(\mathbf{x}(\tau),\tau)\end{pmatrix} \mathrm{d}\tau \tag{4}$$

represents a coupled system, i.e., one needs to solve

$$\mathbf{x}(t) = \mathbf{x}_0 + \int_{t_0}^{t} \dot{\mathbf{x}}(\tau)\,\mathrm{d}\tau \tag{5}$$

and concurrently

$$\dot{\mathbf{x}}(t) = \dot{\mathbf{x}}_0 + \int_{t_0}^{t} \mathbf{a}(\mathbf{x}(\tau),\tau)\,\mathrm{d}\tau, \tag{6}$$

i.e., Equation 5 is needed in Equation 6, and vice versa. We accomplish this by coupled fourth-order Runge-Kutta (RK4) integration, detailed in the supplemental material. In our experiments, we obtained sufficiently accurate results with fixed step size RK4, although adaptive approaches, such as the Runge-Kutta-Fehlberg method, would be likewise possible.

For phenomena, such as magnetic interaction, where acceleration cannot be represented by a vector field, we replace $\mathbf{a}(\mathbf{x},t)$ with an appropriate function, e.g., $\mathbf{a}(t,\mathbf{x}(t),\dot{\mathbf{x}}(t),\dots)$, in the above formulation.

3.2 Finite-Time Mapping

Now that we can solve for the dynamics of point masses over finite advection times T, we can investigate some of their properties. We assume that the underlying accelerations $\mathbf{a}(\cdot)$ are continuous (which is, e.g., the case for N-body systems, magnetism, and tensor-product linearly interpolated fields) both in space and time. In analogy to streamlines in vector fields, this leads to the fact that the trajectories $\boldsymbol{\xi}(t)$ of point masses cannot intersect in phase space. Thus, trajectories can converge but cannot reach the same point in phase space within finite time intervals. In other words, the phase-space mapping

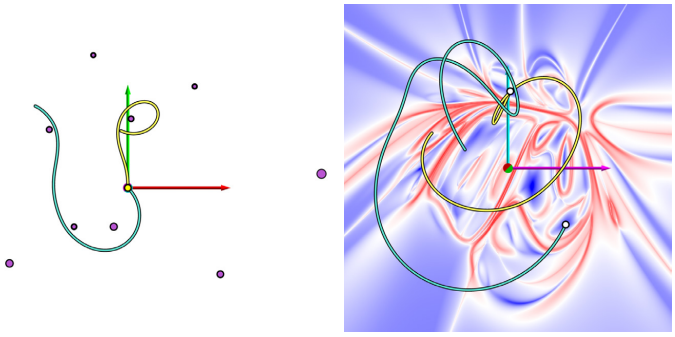
$$\boldsymbol{\Phi}_{t_0}^{T} : \boldsymbol{\xi}(t_0) \mapsto \boldsymbol{\xi}(t_0+T) \tag{7}$$

is bijective. As a consequence, in phase space, manifolds of point masses can only deform due to inertial dynamics, but not self-intersect or tear apart or merge, i.e., they are topologically invariant.

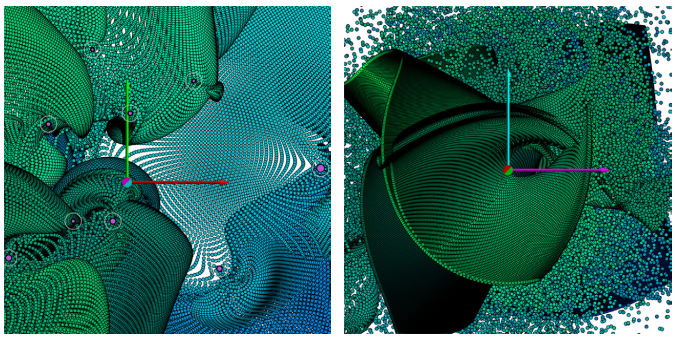
Figure 3 shows an example of a two-manifold of point masses (varying in initial velocity) after transport for a finite time interval. In the spatial projection (Figure 3(c)) and in the velocity projection (Figure 3(d)), one can observe overlaps, but there are no self-intersections in four-dimensional phase space.

3.3 Phase-Space Finite-Time Lyapunov Exponent

Topological analysis aims at providing a partitioning of the domain into parts of qualitatively different behavior. For time-dependent vector fields, where massless particles are assumed to strictly follow the vector field, the finite-time Lyapunov exponent [10] field represents

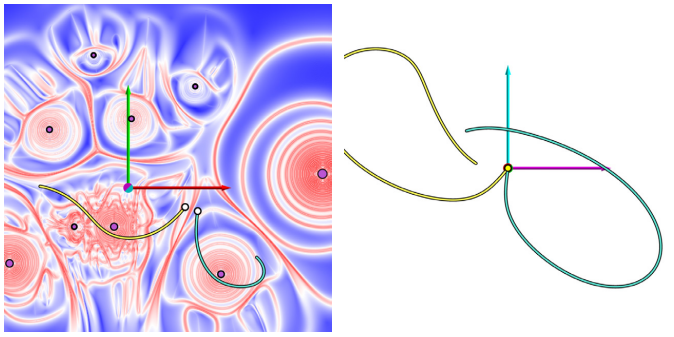


(a) initial position (b) initial velocity

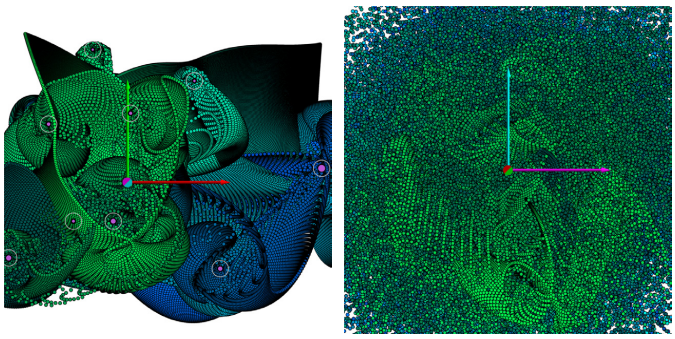


(c) final position (d) final velocity

Fig. 3. Analysis of gravitation-induced inertial dynamics, for the 2D Nine-Body example, with bodies (purple dots, mass-proportional), trajectories (colored), and their seeds (white dots). Views: Initial position (a), initial velocity (b), final position (c), and final velocity (d), with axes denoting x-position (red), y-position (green), x-velocity (magenta), and y-velocity (cyan). Initial position has been constrained to $\mathbf{0}$ (yellow dot in (a)), resulting in degrees of freedom of initial velocity only, which is visualized with PS-FTLE-V (b). All samples of the 600×600 PS-FTLE-V grid after inertial transport for time T shown by dots in (c) and (d) (initial x-velocity mapped to blue channel, initial y-velocity to green channel).



(a) initial position (b) initial velocity



(c) final position (d) final velocity

Fig. 4. Same as Figure 3, but with initial velocity constrained to $\mathbf{0}$ (yellow point in (b)) and thus PS-FTLE-P field in (a).

the maximum spatial spread of massless particles started at time t_0 infinitesimally close to position $\mathbf{x}$, and has proven successful in revealing their topology. The FTLE can be computed as follows:

$$\sigma_{t_0}^{T}(\mathbf{x}) := \frac{1}{|T|} \ln \left\| \nabla \boldsymbol{\phi}_{t_0}^{T}(\mathbf{x}) \right\|_2 , \tag{8}$$

with $\boldsymbol{\phi}_{t_0}^{T}(\mathbf{x})$ representing the flow map which maps massless particles started at $\mathbf{x}$ and time t_0 to their position after advection for time T, and $\|\cdot\|_2$ representing the spectral norm (i.e., for a matrix A the square root of the largest eigenvalue of $A^\top A$). Ridges in this field indicate Lagrangian coherent structures (LCS) [24], which are typically codimension-1 subsets (ridges) of the domain and separate qualitatively different regions for the finite advection time T.

A straightforward approach to analyze inertial dynamics is to apply the FTLE concept to the inertial flow map

$$\boldsymbol{\Phi}_{t_0}^{T}(\boldsymbol{\xi}) := \boldsymbol{\xi} + \int_{t_0}^{t_0+T} \begin{pmatrix} \dot{\mathbf{x}}(\tau) \\ \mathbf{a}(\mathbf{x}(\tau), \tau) \end{pmatrix} \mathrm{d}\tau \tag{9}$$

in phase space, leading to the phase-space finite-time Lyapunov exponent (PS-FTLE):

$$\varsigma_{t_0}^{T}(\boldsymbol{\xi}) := \frac{1}{|T|} \ln \left\| \nabla \boldsymbol{\Phi}_{t_0}^{T}(\boldsymbol{\xi}) \right\|_2 . \tag{10}$$

The PS-FTLE is a (time-dependent) scalar field in the $2n$-dimensional phase-space domain, i.e., for 2D spatial problems it is a 4D scalar field, and for 3D cases it is 6D scalar field.

These higher-dimensional fields could be visualized by interactive hyperslicing [28] or similar approaches, but in these cases the overall picture would need to be "constructed" mentally, which would be cumbersome due to the typically very complex structure of (PS-)FTLE fields. Another approach would be to make use of dimensionality reduction techniques, which aim to map points from 4D (or 6D) to a visual space preserving distances as much as possible [15]. Figure 2 shows the point cloud resulting from a projection of "ridge points" in 4D to a two-dimensional visual space using t-SNE [27]. The ridge points approximate the ridges by selecting those nodes of a regular 50^4 sampling grid (resolution limited by memory requirements of t-SNE) with PS-FTLE value larger than 0.85. The t-SNE projection method has been chosen due to its ability to reveal groups of similar instances. Figure 2(a) shows the resulting projected points colored according to their proximity in 4D (close points have similar colors). As one can see, t-SNE is performing well in terms of neighborhood preservation. Figure 2(b) depicts the same point cloud as in Figure 2(a), but colored according to cluster labels so as to facilitate group identification. Clusters were computed in the visual space using agglomerative clustering [30]. Although some clusters clearly show up in the projection, it is difficult to figure out from the projection layout the interplay among the ridges, that is, the resulting clusters do not provide a direct notion of the true separating structures in phase space, nor their properties with respect to inertial dynamics. Moreover, t-SNE took about 36 s to perform the mapping of 5 726 ridge points, hampering interactive applications. Computationally more efficient projection methods could be used, but interactive rates would hardly be reached.

3.4 Constrained PS-FTLE

To avoid the abovementioned difficulties with visualization of higher-dimensional fields, we decouple, to the necessary extent, the analysis of the phase space with respect to position and velocity, which allows us to avoid an increase of dimensionality. Thus, our approach requires only 2D visualization for 2D problems, and 3D visualization for 3D problems. The central idea of our approach is to constrain the degrees of freedom of the initial condition during interactive analysis by selection, but to enable exploration of this choice, supported by a representation that provides overall context and a notion of the impact of the choice. By definition, and also with respect to most research questions, position and velocity play different roles. Thus, in our approach, either initial position or initial velocity are constrained during the in-

teractive process, and the remaining degrees of freedom are those of initial velocity or initial position, respectively.

Assume that we want to shoot a mass ballistically (governed by gravitation only) from a given point. In this case, the initial position is determined by the launching pad (yellow point in Figure 3(a)). Thus, the remaining degrees of freedom are the initial velocity, the starting time, and the duration of flight. Starting time t_0 and duration of flight T are treated as in FTLE-based examinations described above, i.e., they are typically explored by the user or given by the problem under investigation. This means, they are not treated as a degree of freedom during these types of analysis. Thus, initial velocity are the only degrees of freedom that would need to be examined in this example.

In our example, the degrees of freedom of initial velocity are represented by the range of the initial velocity view (Figure 3(b)), and interactive exploration in that view can be accomplished by inertial trajectories starting with the respective velocity and (predetermined initial position) from Figure 3(a). Since the true trajectory is in phase space, it of course also has a velocity component, which represents a respective trajectory in the initial velocity view. Of course, exhaustive exploration by such interactive trajectories would be cumbersome. But by employing the concept of the PS-FTLE with respect to the remaining degrees of freedom (in this example initial velocity), we can provide a concise representation of the structure of the problem, i.e., of the regions with similar inertial dynamics (Figure 3(b)).

We accomplish this by constraining the PS-FTLE to a fixed initial velocity $\dot{\mathbf{x}}_0$, resulting in the PS-FTLE-P:

$$ {}^{\dot{\mathbf{x}}_0}\varsigma^T_{t_0}(\mathbf{x}) := \varsigma^T_{t_0}\begin{pmatrix}\mathbf{x}\\ \dot{\mathbf{x}}_0\end{pmatrix} := \frac{1}{|T|}\ln\left\|\left(\frac{\partial}{\partial x_1},\ldots,\frac{\partial}{\partial x_n}\right)\boldsymbol{\Phi}^T_{t_0}\begin{pmatrix}\mathbf{x}\\ \dot{\mathbf{x}}_0\end{pmatrix}\right\|_2 \tag{11}$$

or by constraining the PS-FTLE to a fixed initial position $\mathbf{x}_0$, resulting in the PS-FTLE-V:

$$ {}^{\mathbf{x}_0}\varsigma^T_{t_0}(\dot{\mathbf{x}}) := \varsigma^T_{t_0}\begin{pmatrix}\mathbf{x}_0\\ \dot{\mathbf{x}}\end{pmatrix} := \frac{1}{|T|}\ln\left\|\left(\frac{\partial}{\partial \dot{x}_1},\ldots,\frac{\partial}{\partial \dot{x}_n}\right)\boldsymbol{\Phi}^T_{t_0}\begin{pmatrix}\mathbf{x}_0\\ \dot{\mathbf{x}}\end{pmatrix}\right\|_2. \tag{12}$$

Note that

$$\left(\frac{\partial}{\partial x_1},\ldots,\frac{\partial}{\partial x_n}\right)\boldsymbol{\Phi}^T_{t_0}\begin{pmatrix}\mathbf{x}\\ \dot{\mathbf{x}}_0\end{pmatrix} \quad\text{and}\quad \left(\frac{\partial}{\partial \dot{x}_1},\ldots,\frac{\partial}{\partial \dot{x}_n}\right)\boldsymbol{\Phi}^T_{t_0}\begin{pmatrix}\mathbf{x}_0\\ \dot{\mathbf{x}}\end{pmatrix} \tag{13}$$

are $2n \times n$ matrices both leading to an $n \times n$ matrix and thus n eigenvalues during spectral norm computation. Note also that both variants capture spread in phase space, since in general, $\boldsymbol{\Phi}^T_{t_0}(\boldsymbol{\xi})$ maps to varying position and varying velocity, irrespective if initial position or initial velocity are kept constant or not. We denote ${}^{\dot{\mathbf{x}}_0}\varsigma^T_{t_0}(\mathbf{x})$ PS-FTLE-P because it captures phase-space spread in terms of varying initial position, and ${}^{\mathbf{x}_0}\varsigma^T_{t_0}(\dot{\mathbf{x}})$ PS-FTLE-V since it represents phase-space spread due to varying initial velocity. Please see Figure 4(a) for an example of PS-FTLE-P, and Figure 3(b) for a respective example of PS-FTLE-V. We omit a color legend, because for topological analysis based on FTLE variants, only a qualitative view is needed. We employ a colormap that maps low values to blue, medium to white, and large to red. One can nicely see in these illustrative examples how the constrained PS-FTLE captures the topological structure of inertial dynamics, i.e., how ridges in these fields separate regions of qualitatively different inertial dynamics (exemplified with selected inertial trajectories).

3.5 Decomposition of PS-FTLE

The ridges in the PS-FTLE-P and the PS-FTLE-V separate qualitatively different regions in initial position or initial velocity, respectively. By this, they provide a concise representation of inertial dynamics with respect to initial position and velocity. Nevertheless, these fields (and thus also their ridges) represent combined spread only: it is not possible to tell to what extent the detected spread is due to varying final position, and to what extent it is due to varying final velocity.

This motivates, as a complementary visualization component, to decompose PS-FTLE into a part that represents position spread, and another part that represents velocity spread. Due to the abovementioned difficulties with higher-dimensional visualization, we do not provide examples for decomposing the PS-FTLE itself. Instead, we decompose both the PS-FTLE-P and PS-FTLE-V into two fields each. To this end, we first decompose the inertial flow map $\boldsymbol{\Phi}^T_{t_0}(\boldsymbol{\xi})$ into a part $\tilde{\boldsymbol{\Phi}}^T_{t_0}(\boldsymbol{\xi})$ that maps to position, and a part $\tilde{\boldsymbol{\Phi}}^T_{t_0}(\boldsymbol{\xi})$ that maps to velocity:

$$\boldsymbol{\Phi}^T_{t_0}(\boldsymbol{\xi}) =: \begin{pmatrix}\tilde{\boldsymbol{\Phi}}^T_{t_0}(\boldsymbol{\xi})\\ \tilde{\boldsymbol{\Phi}}^T_{t_0}(\boldsymbol{\xi})\end{pmatrix}. \tag{14}$$

This allows us to decompose PS-FTLE-P into position separation:

$$ {}^{\dot{\mathbf{x}}_0}\tilde{\varsigma}^T_{t_0}(\mathbf{x}) := \frac{1}{|T|}\ln\left\|\left(\frac{\partial}{\partial x_1},\ldots,\frac{\partial}{\partial x_n}\right)\tilde{\boldsymbol{\Phi}}^T_{t_0}\begin{pmatrix}\mathbf{x}\\ \dot{\mathbf{x}}_0\end{pmatrix}\right\|_2 \tag{15}$$

(being identical to IFTLE [17]), and velocity separation:

$$ {}^{\dot{\mathbf{x}}_0}\tilde{\varsigma}^T_{t_0}(\mathbf{x}) := \frac{1}{|T|}\ln\left\|\left(\frac{\partial}{\partial x_1},\ldots,\frac{\partial}{\partial x_n}\right)\tilde{\boldsymbol{\Phi}}^T_{t_0}\begin{pmatrix}\mathbf{x}\\ \dot{\mathbf{x}}_0\end{pmatrix}\right\|_2. \tag{16}$$

Accordingly, we also decompose PS-FTLE-V into position separation:

$$ {}^{\mathbf{x}_0}\tilde{\varsigma}^T_{t_0}(\dot{\mathbf{x}}) := \frac{1}{|T|}\ln\left\|\left(\frac{\partial}{\partial \dot{x}_1},\ldots,\frac{\partial}{\partial \dot{x}_n}\right)\tilde{\boldsymbol{\Phi}}^T_{t_0}\begin{pmatrix}\mathbf{x}_0\\ \dot{\mathbf{x}}\end{pmatrix}\right\|_2 \tag{17}$$

and velocity separation:

$$ {}^{\mathbf{x}_0}\tilde{\varsigma}^T_{t_0}(\dot{\mathbf{x}}) := \frac{1}{|T|}\ln\left\|\left(\frac{\partial}{\partial \dot{x}_1},\ldots,\frac{\partial}{\partial \dot{x}_n}\right)\tilde{\boldsymbol{\Phi}}^T_{t_0}\begin{pmatrix}\mathbf{x}_0\\ \dot{\mathbf{x}}\end{pmatrix}\right\|_2. \tag{18}$$

Figure 5 exemplifies the utility of these decompositions for the case of the 2D Nine-Body example. Note that these measures have different

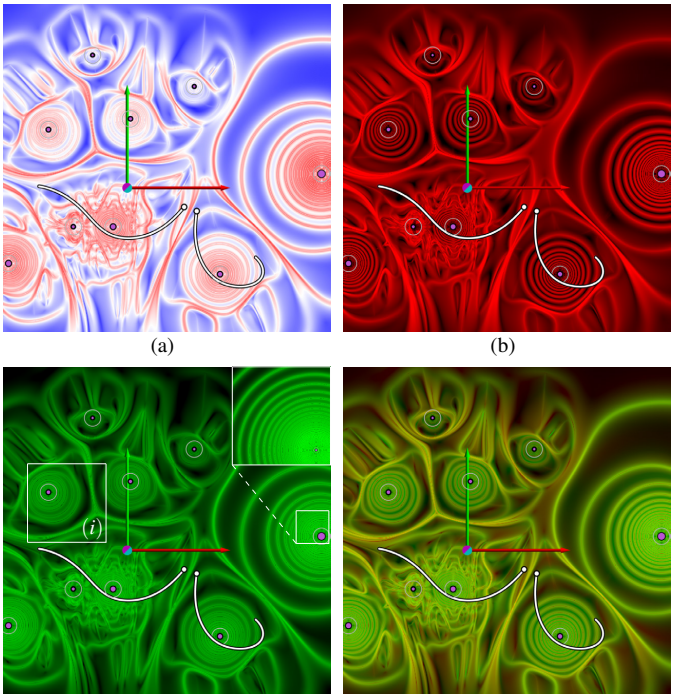
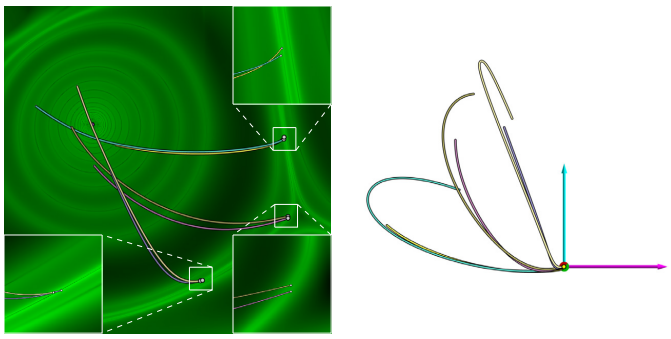
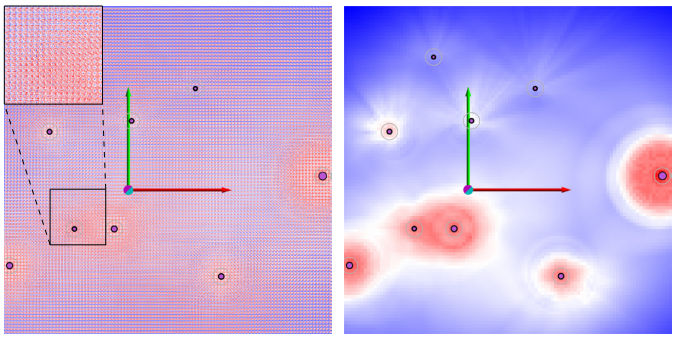


Fig. 5. Decomposition of PS-FTLE-P ((a) from Figure 4(a)) into position separation ${}^{\dot{\mathbf{x}}_0}\tilde{\varsigma}^T_{t_0}(\mathbf{x})$ (b) (mapped to red channel), and velocity separation ${}^{\dot{\mathbf{x}}_0}\tilde{\varsigma}^T_{t_0}(\mathbf{x})$ (c) (mapped to green channel). (d) Combination of both channels reveals relative contribution of position spread and velocity spread. Observe that the ridge where the two trajectories are seeded is more yellowish, and thus position spread is there stronger in relation to velocity spread. This is consistent with the distance of their endpoints in position space (e.g., (d)) and velocity space (Figure 4(b)). Note that both position spread and velocity spread have been normalized because they have different units (1 for ${}^{\dot{\mathbf{x}}_0}\tilde{\varsigma}^T_{t_0}(\mathbf{x})$, and s^{-1} for ${}^{\dot{\mathbf{x}}_0}\tilde{\varsigma}^T_{t_0}(\mathbf{x})$). The very thin valleys within the ridges in region (i) are examined in detail in Figure 6.



(a) (b)

Fig. 6. (a) Valleys within ridges of velocity separation (green), within range (i) of Figure 5(c), with trajectories (colored lines). (b) Same trajectories in velocity space. One can see (zoomed regions) that the valleys separate trajectories with increasing velocity magnitude from those where velocity magnitude decreases again ("returning" velocity curves). The point of maximum velocity magnitude (farthest from origin in (b)) corresponds to shortest distance from mass body (purple point in (a)).



(a) initial position (b) initial position

Fig. 7. Stacked PS-FTLE at the example of the 2D Nine-Body example. (a) Stacked PS-FTLE with zoomed region (17×17 cells of the stacked PS-FTLE grid). Observe that each cell contains the respective PS-FTLE-V field (e.g., Figure 3(b)). (b) Discretized version of (a), using the average operator, for better context at medium zoom levels. In our interactive implementation, there is a transition from (b) to (a) as the user zooms into the stacked PS-FTLE representation.

units and thus their combined visualization cannot provide a quantitative view: ${}^{\mathbf{x}_0}\tilde{\varsigma}^T_{t_0}(\mathbf{x})$ has unit 1, ${}^{\dot{\mathbf{x}}_0}\tilde{\varsigma}^T_{t_0}(\mathbf{x})$ has unit s^{-1}, ${}^{\mathbf{x}_0}\tilde{\varsigma}^T_{t_0}(\dot{\mathbf{x}})$ has unit s, and ${}^{\mathbf{x}_0}\tilde{\varsigma}^T_{t_0}(\dot{\mathbf{x}})$ has unit 1. Their combination to color channels (red and green in our implementation) can only provide qualitative insights and relations. We therefore normalize them prior to mapping to color. For Figure 5, the maximum of position spread of 264.478 has been mapped to maximum red level, and the maximum of velocity spread of $342.153\ \mathrm{s}^{-1}$ to maximum green level. Observe that the nested ring structure around the bodies is more clear in Figure 5(b) than in Figure 5(c), revealing that different types of orbits (different periodicities and thus different Kepler orbital times) are more pronounced with respect to position than velocity. On the other hand, close inspection of Figure 5(c) reveals that the respective ridges enclosing the bodies reveal a very thin valley line at their center. We examined this case, but zooming in did not increase the gap. Nevertheless, Figure 6 shows that the valleys are caused by trajectories "returning" in velocity space, caused by deceleration of particles as they pass a near mass.

3.6 Stacked PS-FTLE

So far, we have shown how phase space can be analyzed using PS-FTLE-P and PS-FTLE-V, i.e., by constraining the underlying IVP either by selecting an initial position or by selecting an initial velocity. A limitation with this approach alone would be, however, that this selection would not be supported, i.e., that the user would need to explore this choice "blindly" without guidance.

This motivates our stacked PS-FTLE (SPS-FTLE) approach. The SPS-FTLE is inspired by dimensional stacking [14, 29] of discrete data in information visualization. Assume we are in the configuration where PS-FTLE-V is used, i.e., where initial position is constrained and the remaining degrees of freedom are initial velocity (Figure 3). In this setup, PS-FTLE-V is displayed in the initial velocity view, and the initial position view contained so far only a point representing the selected initial position. To support this selection, we provide context in the respective view (in this setup the initial position view) by presenting there the respective stacked PS-FTLE field.

For the described configuration, the SPS-FTLE representation in the initial position view consists of a grid that discretizes the initial position range, and within each cell of this grid, the PS-FTLE-V field is represented that results if the center of the respective cell is used as initial position (see Figure 7(a)). To avoid unnecessarily long computation times, the resolution of the PS-FTLE-V fields within the cells is kept low (100×100 in this case). For the opposite configuration, i.e., where initial velocity is constrained, the SPS-FTLE field discretizes the range of initial velocity into cells instead, and each cell contains the respective PS-FTLE-P field.

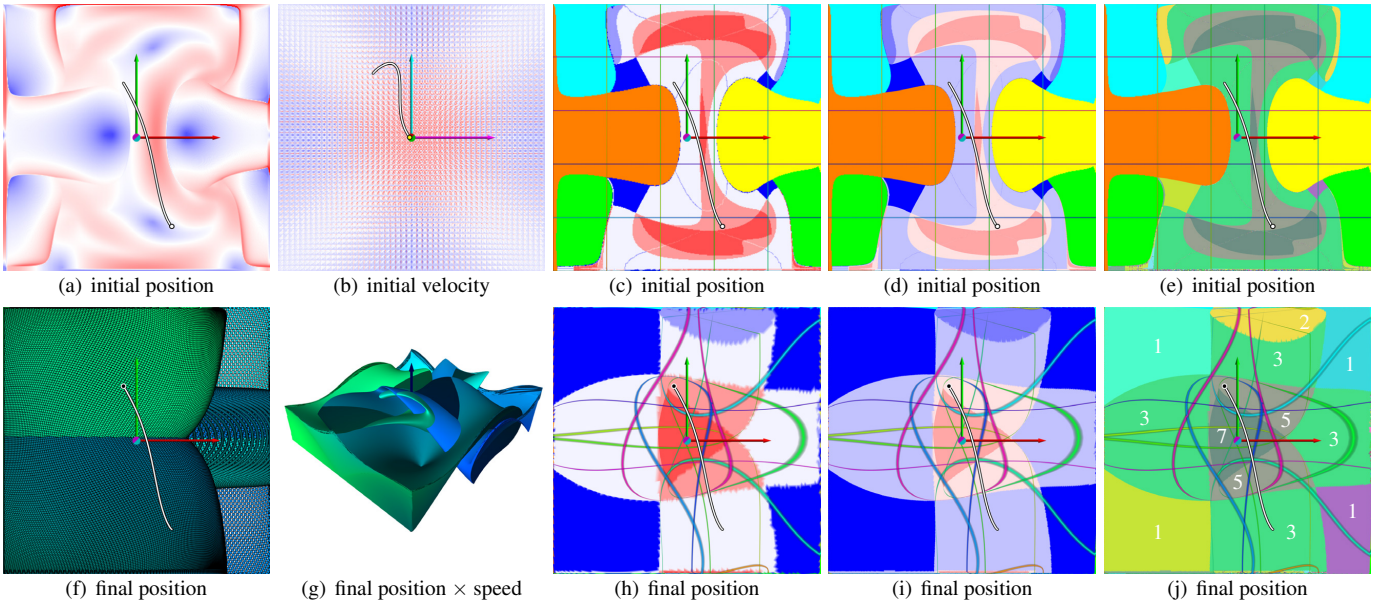
Because the large-scale structure of the SPS-FTLE representation is hard to perceive at medium zoom levels (i.e., one has to zoom out rather far to see the overall structure), we additionally generate a discretized version of the SPS-FTLE representation. This representation is obtained by "merging" the PS-FTLE field within each cell "into a single pixel" which is then shown at the respective resolution, i.e., each cell (or "pixel") of the discretized version represents the respective PS-FTLE-P or PS-FTLE-V field (see Figure 7(b)). In our implementation, the field within a cell is "merged" to a single value using the maximum or average operator. Whereas taking the maximum provides a more conservative view, i.e., high values of the resulting field indicate the maximum separation that appears in the PS-FTLE and thus regions with low values are likely to be of inferior interest, the average operator provides a more quantitative view and, in our experiments, it provided more specific guidance.

3.7 Multiplicity Maps

So far, we focused on the views of initial position (e.g., Figure 3(a)) and initial velocity (e.g., Figure 3(b)). There, the constrained PS-FTLE, its decomposition, and its stacking enable the analysis of inertial dynamics with respect to the initial values of the IVP. As exemplified by the trajectories in Figures 3 and 4, such analysis can answer various research questions (see also Section 5). This is similar to traditional (advection-based) FTLE visualization, which is considering the initial view only, i.e., traditional FTLE "resides" at time t_0; the state at time t_0+T, that the flow map maps to, is typically not analyzed. One reason why this final state is typically not analyzed is that the traditional flow map $\boldsymbol{\phi}^T_{t_0}(\mathbf{x})$ only represents a deformation without overlap (which follows from an argumentation similar to that in Section 3.2), i.e., it is a continuous bijective mapping from the seeds $\mathbf{x}(t_0)$ to the end points $\mathbf{x}(t_0+T)$ of the respective trajectories.

As discussed in Section 3.2, the inertial flow map $\boldsymbol{\Phi}^T_{t_0}(\boldsymbol{\xi})$ represents a bijective mapping in *phase space*. However, as discussed there too, this mapping is typically not bijective anymore in the respective spatial projections (e.g., Figures 3(c) and 4(c)) and velocity projections (e.g., Figures 3(d) and 4(d)): the n-dimensional manifold defined by the degrees of freedom of initial velocity or initial position, respectively, typically exhibits overlap, i.e., folds. In other words, a point in the final position view or final velocity view is often reached by more than one IVP. As we will see, analyzing these properties provides answers to relevant research questions and thus motivates our final component for analysis of inertial dynamics: multiplicity maps.

So far, we visualized in the final position view (e.g., Figure 3(c)) and final velocity view (e.g., Figure 3(d)) the spatial and velocity projections of the sample points of the constrained PS-FTLE. Since constrained PS-FTLE is sampled in the domain of initial position or initial velocity, it represents an n-manifold with a parametrization induced by these degrees of freedom. Figures 3(c) and 3(d) already provide an impression how the n-manifold resides when projected to the space or



(a) initial position (b) initial velocity (c) initial position (d) initial position (e) initial position

(f) final position (g) final position × speed (h) final position (i) final position (j) final position

Fig. 8. PS-FTLE and multiplicity for in Quad-Gyre example. Seed of inertial trajectory by white dot, final state by black dot. (a) PS-FTLE-P. (b) Selected initial velocity (yellow dot) with stacked PS-FTLE-P for context. (f) Final position of phase-space samples. (g) 3D mesh representation with final position × final speed (velocity magnitude, blue axis). (h) Position multiplicity map ${}^{\dot{\mathbf{x}}_0}\mu(\mathbf{x})_{t_0}^T$ induced by PS-FTLE-P shows how many IVPs reach a given final position. (c) Multiplicities from (h) mapped to respective initial position view show how many other IVPs reach the same final position (e.g., dark blue (1) means that no other IVP reaches the same final state). Boundaries are mapped: left→orange, right→yellow, top→cyan, bottom→green. (d),(i) Same as (c),(h) but with two levels of adaptive refinement. See how adaptive refinement improves the map. Observe also that some edges were already sharp in (h) because they originate from silhouettes of the folded 2-manifold. The color mapping (but not the multiplicities) typically changes during refinement because silhouettes representing almost "oblique" manifold regions are subject to "discretization noise" (e.g., at bottom boundary), typically caused by particles that are stopped at the domain boundary. (j) Connected components in (i), with multiplicity numbers. Note that in our prototype, these numbers are provided by hovering the mouse over the respective region. (e) Labels from (j) mapped to corresponding initial position. Grid lines (regular in initial space (c), (d), and (e)) provide visual cue to the mapping.

velocity domain, since the x-component of the initial value is mapped to the blue channel and the y-component to green. Nevertheless, due to the discrete sampling and due to occlusion of the points, this kind of visualization does not provide the multiplicity of the mapping, i.e., it does not show how many IVPs reach a given point in the projections of the final state.

The position multiplicity map

$$\mu(\mathbf{x})_{t_0}^T := \left| \left\{ \boldsymbol{\xi} \mid \tilde{\boldsymbol{\Phi}}_{t_0}^T(\boldsymbol{\xi}) = \mathbf{x} \right\} \right| \tag{19}$$

represents how many IVPs starting at time t_0 reach a given position $\mathbf{x}$ after time T. Its counterpart is the velocity multiplicity map

$$\mu(\dot{\mathbf{x}})_{t_0}^T := \left| \left\{ \boldsymbol{\xi} \mid \tilde{\boldsymbol{\Phi}}_{t_0}^T(\boldsymbol{\xi}) = \dot{\mathbf{x}} \right\} \right| \tag{20}$$

which counts how many IVPs starting at time t_0 reach a given velocity $\dot{\mathbf{x}}$ after inertial transport for time T.

In the context of constrained PS-FTLE, both the PS-FTLE-P and the PS-FTLE-V lead each to a respective position multiplicity map and a velocity multiplicity map. The PS-FTLE-P leads to the position multiplicity map

$${}^{\dot{\mathbf{x}}_0}\mu(\mathbf{x})_{t_0}^T := \left| \left\{ \mathbf{y} \mid \tilde{\boldsymbol{\Phi}}_{t_0}^T \begin{pmatrix} \mathbf{y} \\ \dot{\mathbf{x}}_0 \end{pmatrix} = \mathbf{x} \right\} \right| \tag{21}$$

and velocity multiplicity map

$${}^{\dot{\mathbf{x}}_0}\mu(\dot{\mathbf{x}})_{t_0}^T := \left| \left\{ \mathbf{y} \mid \tilde{\boldsymbol{\Phi}}_{t_0}^T \begin{pmatrix} \mathbf{y} \\ \dot{\mathbf{x}}_0 \end{pmatrix} = \dot{\mathbf{x}} \right\} \right|, \tag{22}$$

whereas PS-FTLE-V leads to the position multiplicity map

$${}^{\mathbf{x}_0}\mu(\mathbf{x})_{t_0}^T := \left| \left\{ \dot{\mathbf{y}} \mid \tilde{\boldsymbol{\Phi}}_{t_0}^T \begin{pmatrix} \mathbf{x}_0 \\ \dot{\mathbf{y}} \end{pmatrix} = \mathbf{x} \right\} \right| \tag{23}$$

and velocity multiplicity map

$${}^{\mathbf{x}_0}\mu(\dot{\mathbf{x}})_{t_0}^T := \left| \left\{ \dot{\mathbf{y}} \mid \tilde{\boldsymbol{\Phi}}_{t_0}^T \begin{pmatrix} \mathbf{x}_0 \\ \dot{\mathbf{y}} \end{pmatrix} = \dot{\mathbf{x}} \right\} \right|. \tag{24}$$

Figure 8(i) shows a position multiplicity map ${}^{\dot{\mathbf{x}}_0}\mu(\mathbf{x})_{t_0}^T$ for the Quad-Gyre example, induced by PS-FTLE-P. For better display, we apply a logarithmic scaling and use the same colormap as for the constrained PS-FTLE results. Such representations of the multiplicity map provide insight into how many IVPs reach a respective point or velocity, but they do not reveal which IVPs reach which region.

To this end, we perform a connected component labeling of the multiplicity maps, i.e., we detect connected components in the map that exhibit the same multiplicity, i.e., that are reached by the same number of IVPs. Figure 8(j) shows the connected component labeling for the multiplicity map from Figure 8(i). To reveal the correspondences between the initial values and the connected components in the projection of the final state, we map the connected component labels back to the initial state. We achieve this by looking up for each sample in its final state in phase space the connected component label from the multiplicity field at its projection, and then apply the label to the respective initial value, resulting in a corresponding labeling in the initial position space or velocity space (Figure 8(e)). Together with Figure 8(j)), this reveals the correspondences of the inertial IVPs, i.e., one can see which connected components in the multiplicity map are caused by which regions of initial values.

Nevertheless, since both the connected components in the multiplicity map and the corresponding initial regions can be comparably large, this visualization does not provide information on exactly which initial values map to which final state. We address this issue by two complementary approaches. First, we add grid lines that are also mapped to the corresponding view. Second, and more specific, we allow the user to interactively select regions in the final state and display the corresponding initial values (sample points) that reach those regions (Figure 1(c) and 1(b)). This way, the labels and grid lines provide context

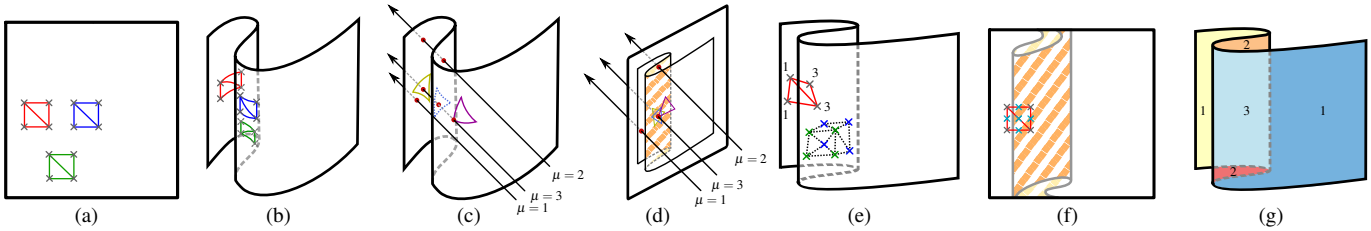


Fig. 9. Multiplicity map computation. (a) Regular discretization and triangulation of initial space, yielding a flat manifold. (b) Manifold is transformed after transport, third dimension here by mapping to a quantity, e.g., velocity magnitude. (c) Vertex multiplicities represent number of ray intersections with triangles. (d) Intersections are determined in projected space. (e) Refinement (red) is performed for quads with edges that join vertices with different multiplicities (numbers). Projected vertices (from green and blue quad) are triangulated (dotted) to compute connected components in final space. (f) The quad to be refined is subdivided into four subcells by adding five new samples in the initial space. The process (triangulation, multiplicity computation) is then repeated with the new samples. (g) Connected components labeling by traversal in final space, with connectivity defined by equality of multiplicity.

information, and the interactive selection of samples provides explicit information on which IVPs reach which state.

3.7.1 Computation of Multiplicity Maps

Figure 9 illustrates the computation of multiplicity maps and the connected components therein. As depicted in Figure 9(a), the n degrees of freedom (i.e., initial position or velocity range) of the constrained PS-FTLE field are sampled on a uniform grid and a mesh is obtained using Delaunay triangulation. We then transport each sample by the inertial IVP to its final state in phase space (Figure 9(b)). For a position multiplicity map, these points are then projected to position space, otherwise they are projected to velocity space (in both cases, the n-manifold is projected from $2n$-dimensional phase space to nD). Let us assume, without loss of generality, that we construct a position multiplicity map. A ray is shot through each of the projected samples and the intersections between the ray and the triangles of the projected mesh are counted, providing the multiplicity count for each sample (Figures 9(c) and 9(d)). This count can then be directly visualized in the initial space (e.g., Figure 8(d)). However, visualization of this map in the final state (e.g., Figure 8(i)) is nontrivial because the n-manifold typically exhibits overlap, and rendering, e.g., using blending, would result in artifacts due to strongly stretched triangles which are generated due to the typically very strong distortions due to finite-time dynamics. Therefore, we instead employ Delaunay triangulation of the projected samples after transport (Figure 9(e), dotted edges). Then, the planar mesh containing multiplicity values defined at the samples can be rendered (based on barycentric interpolation). The connected component labeling (Figure 9(g)) is achieved by identifying the connected subgraphs where all the vertices have the same multiplicity count. After labeling, the connected components can be visualized in the final state (Figure 8(j)), or mapped back to the initial value space (Figure 8(e)). The optional step of refinement can be processed once the sample multiplicity counts are in place. In that case, we identify the edges that join samples with different multiplicity and subdivide the quads in the initial uniform grid into four subcells, and then repeat the process until convergence or a maximum depth is reached (Figures 9(e) and 9(f), red quad and green/blue samples).

4 Implementation Details

Our implementation uses CUDA (through PyCuda bindings [13]), when possible, for the processing, and OpenGL (through VisPy gloo bindings) for the rendering. We use Python as a general-purpose scripting language to launch the CUDA kernels, fetch the data and pass them to OpenGL for rendering, and to construct and manage the GUI. Some functions cannot be fully implemented in the parallel paradigm that CUDA offers. In such cases we rely on the CPU.

A special use case is the processing of the PS-FTLE field, which is computed in each time step. In this case, a CPU synchronization step between the flow map and the PS-FTLE kernels is needed. Some operations are fully implemented in CUDA (the flow map computation, all variants of the PS-FTLE, and the interactive trajectories), while others are implemented on the CPU and partially accelerated using CUDA kernels (the multiplicity maps, the adaptive sampling, and the stacked PS-FTLE). Finally, we have a few operations fully implemented on the CPU, such as the Delaunay triangulation (which uses the Qhull [1] library through SciPy [12]).

In order to access the vector field or N-body simulation data on the GPU, we use textures. These textures consist of a $T \times N \times M$ matrix in the case of 2D vector fields, and a $T \times N \times M \times L$ matrix in the case of 3D vector fields, where T is the number of time steps and N, M, L are the dimensions of the vector field. In the case of N-body systems, the textures consist of a $T \times N$ matrix, where T is the number of time steps in the original N-body simulation and N is the number of bodies.

Due to the flexibility and symmetry of our approach, we found writing a single CUDA kernel for each operation would render the system unmaintainable. In order to tackle this issue, we set up a templating structure which generates and compiles all possible combinations of kernel files. These kernel files are then accessed and run through indexed tables which are queried using the application's current internal state model. So, for example, the kernel which computes the force acting on a test particle is used to compute the trajectories but also the flow map, and the code is different depending on the input field in use (N-body simulation or vector field).

We have a total of 30 template files (2 046 code lines) which produce, after templating, 40 compilable kernel files. The number of code lines of all final kernel files combined after templating is 9 107.

5 Results

We exemplify our approach using one analytic example (Section 5.1), a 2D N-body system (Section 5.2), a 2D magnetic system (Section 5.3), and a 3D N-body system (Section 5.4).

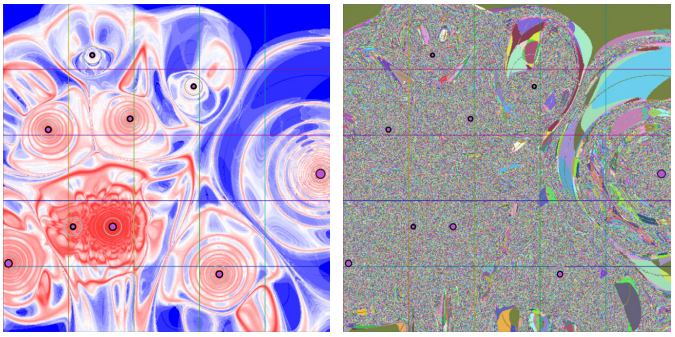
5.1 Quad-Gyre

Our first example has been presented by Shadden et al. [24] for the analysis of Lagrangian coherent structures. It represents a time-dependent vector field given in analytic representation:

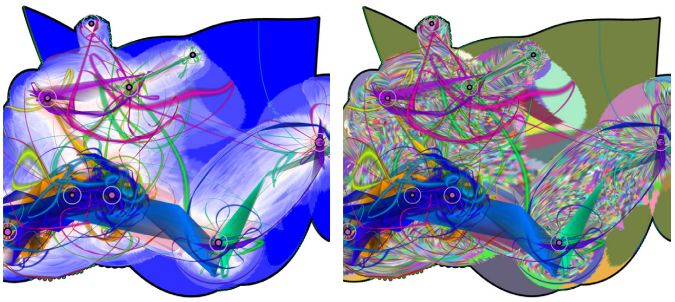
$$\mathbf{a}(\mathbf{x},t) = \begin{pmatrix} -\pi A \sin(\pi f(x,t)) \cos(\pi y) \\ \pi A \cos(\pi f(x,t)) \sin(\pi y) \frac{\mathrm{d}f}{\mathrm{d}x} \end{pmatrix} \tag{25}$$

with $f(x,t) = a(t)x^2 + b(t)x$, $a(t) = \varepsilon \sin(\omega t)$, $b(t) = 1 - 2\varepsilon \sin(\omega t)$, $\mathbf{x} = (x,y)^\top$, $\varepsilon = 0.25$, $\omega = \pi/5$, and $A = 0.1$. We sampled this field at a resolution of 50×50 nodes and 100 time steps in the spatial range $\mathbf{x} \in [-1,1] \times [-1,1]$ and temporal range $[0,20]$. Note that we use this y-range instead of the often employed y-range $[0,1]$, leading to a y-symmetric field with four time-dependent vortices instead of two.

We interpret this vector field as an acceleration field and employ our analysis technique. Initial time $t_0 = 0$ s and transport time $T = 3.046$ s, with zero initial velocity, resulting in a PS-FTLE-P visualization (Figure 8(a)). One can see from the inertial trajectory that mass released at the respective position reaches the upper half, which would not be possible by advection. Looking at the multiplicity map in initial position space (Figure 8(d)), one can see that many of the ridges in the PS-FTLE-P are caused by the domain boundaries (orange, yellow, cyan,

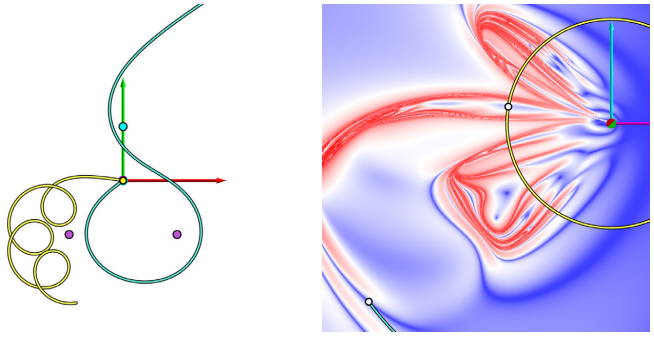


(a) initial position (b) initial position



(c) final position (d) final position

Fig. 10. Position multiplicity map in 2D Nine-Body example. (c) Position multiplicity map ${}^{\mathbf{x}_0}\mu(\mathbf{x})_{t_0}^{T}$, induced by PS-FTLE-P. (a) Multiplicities from (c) mapped to corresponding initial values. (d) Connected components in (c). (b) Component labels from (d) at respective initial positions.



(a) initial position (b) initial velocity

Fig. 11. Magnetic Dipoles example. (a) Initial position set to $\mathbf{0}$ (yellow dot), with magnetic dipoles (magnetic moment by dot color) and trajectories (colored lines). (b) PS-FTLE-V visualizing regions of qualitatively different dynamics induced by Lorenz force, with seeds (white dots).

or green color) of our sampled vector field. The reason for this is that inertial particles reaching the domain boundary are stopped and accumulate there. The shown trajectory is seeded within one of the regions with low PS-FTLE-P value, representing trajectories that do not reach the boundary. The refined multiplicity map (Figure 8(i)) and in particular its connected component labeling with multiplicity numbers (Figure 8(j)) provide insight into the folding of the manifold at final position and thus inertial dynamics, which would not be possible from direct visualization of the final positions by points (Figure 8(f)).

5.2 2D Nine-Body System

A traditional problem that involves inertial dynamics is the motion of masses in gravitation fields. To this end, we integrated a N-body simulation in our interactive implementation. The 2D Nine-Body example consists of nine bodies with different masses at fixed positions. To avoid singularities at the body centers, a softening length [20, 2] is employed and visualized with gray circles. Our approach is used to analyze the gravitational field induced by the bodies. Please see the accompanying video for further examples, where the bodies also undergo inertial dynamics. In all these cases, the initial value problems (point masses) underlying our technique are influenced only by the gravitational field of the bodies, i.e., the test particles do not influence each other. Although this could be accomplished at some additional computational cost, it would for most research questions be inappropriate, because our approach, in accordance with traditional FTLE, aims at analyzing the choice of an IVP, not the analysis of mutually dependent particles.

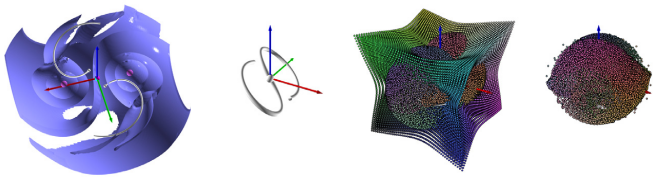
Figures 1, 3, and 4 provide an analysis using PS-FTLE-V and PS-FTLE-P fields. It can be nicely seen how ridges in the PS-FTLE-V field separate different inertial dynamics with respect to initial velocity, whereas the PS-FTLE-P shows regions of different inertial dynamics due to varying initial position. Since these constrained PS-FTLE fields represent a superposition of position spread and velocity spread, we investigated these spreads individually by PS-FTLE decomposition (Figure 5). This analysis provided a surprising result: position spread captures the variation in dynamics around the gravitational bodies better than velocity spread. On the other hand, the large-scale ridges in velocity spread exhibit very thin valley lines (discussed in Section 3.5 and Figure 6). Figure 7(a) shows stacked PS-FTLE-V that supports the choice of an initial position, i.e., helps navigate the $2n$-dimensional phase space. Finally, we investigated multiplicity maps (Figure 10). We have chosen for this the same transport time $T = 4.246$ s, as for the other result images. Figure 10(b) reveals the chaotic inertial dynamics of this example at this transport time: one can observe "islands of stability in a sea of chaos". Due to the extremely strong stretching (see stretched grid lines in Figures 10(c) and 10(d)) and mixing of trajectories (i.e., chaotic transport), the connected components of the position multiplicity map are disrupted in chaotic "noise", too (Figure 10(d)), and the grid is distorted to an extent that would require extremely high refinement. Nevertheless, as can be seen in Figure 10(a), the multiplicity-labeled initial position representation is still able to provide a good notion of IVP multiplicity: it shows for each initial value the count of other IVPs reaching the same final state (observe the quantization of this image which represents these counts).

5.3 Magnetic Dipoles

Our approach lends itself for the analysis of any type of inertial dynamics. We exemplify this with a special case that cannot be represented by an acceleration (vector) field: the motion of charged particles due to Lorenz forces, i.e., the motion of such particles in magnetic fields. Such dynamics exhibits "helical" motion which not only depends on the strength of the magnetic field, but also on the velocity of the charged particle. For clarity and ease of representation, we have chosen a configuration that results in purely 2D dynamics. Nevertheless, it would be straightforward to adapt the next example (Section 5.4), which examines 3D N-body systems, accordingly.

Our example follows the 2D Nine-Body example in that it exhibits discrete objects (in this case magnetic dipoles) with varying properties (in this case different magnetic moment), and it also employs a softening length to avoid numerical issues due to singularities at the dipole centers. To obtain purely 2D inertial dynamics, all dipoles are located on, and oriented perpendicular to, the xy-plane, and we examine only initial xy-positions and initial xy-velocities. Note that, as in the case of the 2D Nine-Body example, this simulation is integrated in our interactive visualization system and thus does not involve artificial domain boundaries and is fully parametrizable.

Because initial velocity plays the more important role, we fix initial position to $\mathbf{0}$ and investigate the problem using PS-FTLE-V (Figure 11(b)). We chose two inertial trajectories by selecting different initial velocities (see also their spatial representation in Figure 11(a)) to exemplify the different dynamics of the regions in the PS-FTLE-V field, which are separated by respective ridges. We do not provide here the point-based visualization of final position and final velocity, because they are both heavily convoluted.



(a) initial position (b) initial velocity (c) final position (d) final velocity

Fig. 12. 3D Two-Body example. (a) Height ridge surfaces of PS-FTLE-P nicely separate different types (periodicities) of orbits (bodies by purple spheres, trajectories by white tubes, seeds by white spheres). (b) Respective trajectories in velocity space (note that we did not introduce a new color scheme for 3D velocity axes). (c) Final positions of PS-FTLE-P samples. (d) Final velocities.

5.4 3D Two-Body System

As our last example, we exemplify that our approach lends itself equally well for the visualization of 3D problems. To this end, we set up a 3D variant of the Nine-Body example. In this case, we have two bodies that are not fixed but move according to an N-body system in 3D, i.e., they orbit each other. Figure 12 shows our results for an initial velocity of $\mathbf{0}$, transport time $T = 12$ s, with a PS-FTLE-P sampling grid of 120^3, running at about 3.1 FPS, without including the N-body simulation, which is precomputed. Instead of direct visualization of the constrained PS-FTLE field, we extracted height ridge surfaces [3].

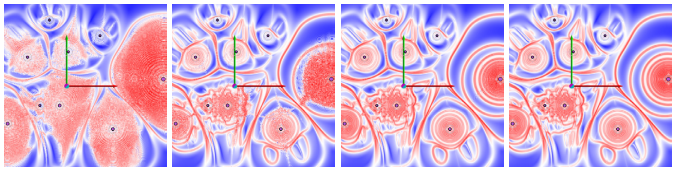
5.5 Performance

We have run two series of performance tests to analyze how the number of time steps and the grid resolution affect the compute time. To run the tests, we have used the 2D Nine-Body example with a softening length $\epsilon^2 = 0.05$. All tests have been run on a system with Manjaro Linux 15.12, kernel 4.4.5.1, 16 GB of RAM memory, an Intel Core i7-4790K CPU at 4.4 GHz, and a GeForce GTX 970 4 GB graphics card. For the performance analysis, if not stated differently, we have used the 2D Nine-Body example with a transport time $T = 4.0$ s, a time step $\Delta t = 0.001$ s, and a grid of 400×400 particles. We have found that the computing time scales linearly with the number of steps, see Table 1 for the performance results. To analyze the impact of grid resolution, we have chosen four representative resolutions, 200×200, 400×400, 600×600, and 800×800. The results show (Table 1) that the computing time also scales linearly with the number of samples.

The stacked PS-FTLE example in Figure 7 took 2 682.239 s to compute, of which 2 653.038 s where spent running CUDA kernels. The rest was invested in the CPU loop. The multiplicity maps without adaptive sampling with the Quad-Gyre dataset shown in Figures 8(h) and 8(c) took 177.121 s to compute, of which 1.584 s were spent computing vertex multiplicity, 149.454 s in Delaunay triangulation, and 0.733 s in the final color mapping. The two levels of adaptive sampling used to produce Figures 8(d), 8(i), 8(e), and 8(j) took 6 866.671 s to compute. The multiplicity maps without connectivity of this same example, after the adaptive sampling, took 202.827 s, of which 6.394 s were for computing vertex multiplicity, 96.855 s for Delaunay triangulation, and 1.460 s for the final color mapping. Finally, the version with connected components of the same example took 245.657 s to compute, of which 6.375 s was needed for computing vertex multiplicity, 94.764 s for Delaunay triangulation, and 48.478 s were spent in connected component labeling. Figure 13 shows the impact of the variation of the step size, as used for the measurements. Figure 14 shows the resolutions used for the measurements.

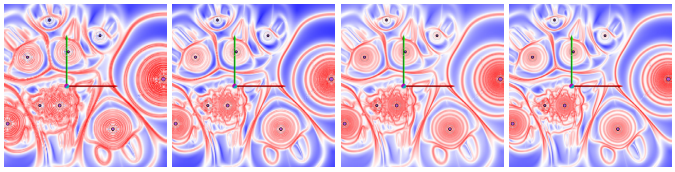
Table 1. Performance results in frames per second for different grid resolutions and time steps. $T = 4.0$ s in all tests.

Test	Fig.	Grid	Δt	Steps	Samples	FPS
Grid	14(a)	200×200	0.001	4 000	40 000	1.0080
	14(b)	400×400			160 000	0.2899
	14(c)	600×600			360 000	0.1322
	14(d)	800×800			640 000	0.0748
Δt	13(a)	400×400	0.1	40	160 000	77.230
	13(b)		0.01	400		10.120
	13(c)		0.001	4 000		1.008
	13(d)		0.0001	40 000		0.107



(a) $\Delta t = 0.1$ s (b) $\Delta t = 0.01$ s (c) $\Delta t = 0.001$ s (d) $\Delta t = 0.0001$ s

Fig. 13. PS-FTLE-P with different step sizes using the Nine-Body example (grid 400×400, $T = 4.0$ s). (a) Integration with $\Delta t = 0.1$ s results in severe artifacts, whereas with (d), no artifacts are perceivable. In the accompanying video, we set Δt between 0.001 s and 0.02 s depending on the experiment. See Table 1 for respective performance measurements.



(a) 200×200 (b) 400×400 (c) 600×600 (d) 800×800

Fig. 14. PS-FTLE-P with different grid resolutions using the Nine-Body example ($T = 4.0$ s, $\Delta t = 0.001$ s). (a) With a grid resolution of 200×200, samples can clearly be seen and fine structures cannot be resolved, whereas with (d), very fine structures can be resolved. In the accompanying video, we used a grid resolution of 200×200 in the introduction and multiplicity maps sections, and a grid resolution of 100×100 in the stacked PS-FTLE section.

6 Conclusion

We presented a novel approach for the analysis of inertial dynamics in terms of initial value problems in n-dimensional space. For this, we extended the concept of the finite-time Lyapunov exponent (FTLE) to $2n$-dimensional phase space, leading to phase-space FTLE (PS-FTLE). By introducing constrained PS-FTLE, we are able to avoid direct visualization of this higher-dimensional space, leading to visualization in nD. To enable the analysis of the contribution of position spread versus velocity spread, we introduced decomposition of the PS-FTLE. To provide guidance for the exploration of the $2n$-dimensional space of initial values, we presented stacked PS-FTLE. Finally, for the analysis of the interrelation of initial value problems in phase space, we presented multiplicity maps, their effective refinement, computation of connected components therein, and complemented this approach with interactive selection of initial values with respect to final states.

As future work, we plan to apply our approach to various fields in science and engineering, and investigate novel approaches for analysis of systems with dimension larger than three, including time.

Acknowledgments

The authors thank Prashant Jalan for the initial prototype. The research leading to these results has been done within the subproject A7 of the Transregional Collaborative Research Center SFB / TRR 165 “Waves to Weather” funded by the German Science Foundation (DFG). It also used computational resources from CeMEAI- ICMC/FAPESP and was partially funded by grants 13/07375-0, 11/22749-8, 14/12815-1 São Paulo Research Foundation (FAPESP). The views expressed are those of the authors and do not reflect the official policy or position of the São Paulo Research Foundation.

REFERENCES

[1] C. B. Barber, D. P. Dobkin, and H. Huhdanpaa. The quickhull algorithm for convex hulls. *ACM Transactions on Mathematical Software*, 22(4):469–483, 1996.

[2] W. Dehnen. Towards optimal softening in three-dimensional N-body codes – I. minimizing the force error. *Monthly Notices of the Royal Astronomical Society*, 324(2):273–291, 2001.

[3] D. Eberly. *Ridges in Image and Data Analysis. Computational Imaging and Vision.* Kluwer Academic Publishers, 1996.

[4] D. Garaboa-Paz and V. Pérez-Muñuzuri. A method to calculate finite-time Lyapunov exponents for inertial particles in incompressible flows. *Nonlinear Processes in Geophysics*, 22(5):571–577, 2015.

[5] T. Günther, A. Kuhn, B. Kutz, and H. Theisel. Mass-dependent integral curves in unsteady vector fields. *Computer Graphics Forum*, 32(3):211–220, 2013.

[6] T. Günther and H. Theisel. Vortex cores of inertial particles. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):2535–2544, 2014.

[7] T. Günther and H. Theisel. Finite-time mass separation for comparative visualizations of inertial particles. *Computer Graphics Forum*, 34(3):471–480, 2015.

[8] T. Günther and H. Theisel. Inertial steady 2D vector field topology. *Computer Graphics Forum*, 35(2):455–466, 2016.

[9] T. Günther and H. Theisel. Source inversion by forward integration in inertial flows. *Computer Graphics Forum*, 35(3):371–380, 2016.

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

[11] P. Joia, F. Petronetto, and L. G. Nonato. Uncovering representative groups in multidimensional projections. *Computer Graphics Forum*, 34(3):281–290, 2015.

[12] E. Jones, T. Oliphant, P. Peterson, et al. SciPy: Open source scientific tools for Python, 2001–. [Online; accessed 2016-03-15].

[13] A. Klöckner, N. Pinto, Y. Lee, B. Catanzaro, P. Ivanov, and A. Fasih. PyCUDA and PyOpenCL: A scripting-based approach to GPU run-time code generation. *Parallel Computing*, 38(3):157–174, 2012.

[14] J. LeBlanc, M. O. Ward, and N. Wittels. Exploring n-dimensional databases. In *Proceedings of the First IEEE Conference on Visualization*, pages 230–237, 1990.

[15] J. A. Lee and M. Verleysen. *Nonlinear dimensionality reduction*. Springer Science & Business Media, 2010.

[16] G. M. Machado, S. Boblest, T. Ertl, and F. Sadlo. Space-time bifurcation lines for extraction of 2D Lagrangian coherent structures. *Computer Graphics Forum*, 35(3):91–100, 2016.

[17] J. Peng and J. O. Dabiri. Transport of inertial particles by Lagrangian coherent structures: application to predatorprey interaction in jellyfish feeding. *Journal of Fluid Mechanics*, 623:75–84, 2009.

[18] A. Pobitzer, R. Peikert, R. Fuchs, B. Schindler, A. Kuhn, H. Theisel, K. Matković, and H. Hauser. The state of the art in topology-based visualization of unsteady flow. *Computer Graphics Forum*, 30(6):1789–1811, 2011.

[19] S. G. Raben, S. D. Ross, and P. P. Vlachos. Experimental determination of three-dimensional finite-time Lyapunov exponents in multi-component flows. *Experiments in Fluids*, 55(10):1–6, 2014.

[20] S. A. Rodionov and N. Y. Sotnikova. Optimal choice of the softening length and time step in N-body simulations. *Astronomy Reports*, 49(6):470–476.

[21] F. Sadlo and D. Weiskopf. Time-dependent 2-D vector field topology: An approach inspired by Lagrangian coherent structures. *Computer Graphics Forum*, 29(1):88–100, 2010.

[22] T. Sapsis and G. Haller. Inertial particle dynamics in a hurricane. *Journal of the Atmospheric Sciences*, 66(8):2481–2492, 2009.

[23] T. Sapsis, J. Peng, and G. Haller. Instabilities on prey dynamics in jellyfish feeding. *Bulletin of Mathematical Biology*, 73(8):1841–1856, 2011.

[24] S. Shadden, F. Lekien, and J. Marsden. Definition and properties of Lagrangian coherent structures from finite-time Lyapunov exponents in two-dimensional aperiodic flows. *Physica D: Nonlinear Phenomena*, 212(3-4):271–304, 2005.

[25] M. Sudharsan, S. L. Brunton, and J. J. Riley. Lagrangian coherent structures and inertial particle dynamics. *Physical Review E*, 93:033108, 2016.

[26] M. Üffinger, F. Sadlo, and T. Ertl. A time-dependent vector field topology based on streak surfaces. *IEEE Transactions on Visualization and Computer Graphics*, 19(3):379–392, 2013.

[27] L. Van der Maaten and G. Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9(2579–2605):85, 2008.

[28] J. J. van Wijk and R. van Liere. HyperSlice: Visualization of scalar functions of many variables. In *Proceedings of the 4th Conference on Visualization*, pages 119–125, 1993.

[29] M. O. Ward, J. T. Blanc, and R. Tipnis. N-land: a graphical tool for exploring n-dimensional data. In *Insight Through Computer Graphics: Proceedings of the Computer Graphics International 1994*, pages 130–141. 1997.

[30] R. Xu and D. Wunsch. Survey of clustering algorithms. *IEEE Transactions on Neural Networks*, 16(3):645–678, 2005.