The Backward Phase Flow Method for the Eulerian Finite Time Lyapunov Exponent Computations

Shingyu Leung*

(Dated: December 1, 2013)

We propose a simple Eulerian approach to compute the moderate to long time flow map for approximating the Lyapunov exponent of a (periodic or aperiodic) dynamical system. The idea is to generalize a recently proposed backward phase flow method which is specially designed for long time level set propagation. Unlike the original phase flow method or the backward phase flow method which is applicable only to autonomous systems, the current approach can also be applied to any time-dependent (periodic or aperiod) flows. We will discuss the stability of the proposed method. Numerical examples will be given to demonstrate the effectiveness of the algorithm.

Finite-time Lyapunov exponent (FTLE) is a widely-used quantity for visualizing the Lagrangian coherent structure (LCS) in a complex dynamical system. Unfortunately, typical numerical approaches to FTLE are computationally very expansive. In this work, we further develop an Eulerian approach to this Lagrangian quantity and propose a partial differential equation (PDE) based phase flow approach to speed up the overall computations even for aperiodic flows. This numerical method can efficiently compute the moderate to long time flow map for approximating the Lyapunov exponent of a dynamical system.

I. INTRODUCTION

Lagrangian coherent structures (LCS) are a tool to visualize and to study a complex dynamical system. For example, it has been recently applied to study flow data in oceans [15, 27], hurricane [26], flight data [4, 32], gravity wave propagation [33], some bio-inspired fluid flows [10, 19, 21], and etc. The idea is to partition the space-time domain into different regions according to a Lagrangian quantity advected along with passive tracers. One of many possible Lagrangian quantities is the Lyapunov exponent (LE). Numerically, one has to truncate the infinite time limit by a fixed period of time and to obtain the so-called finite-time Lyapunov exponent (FTLE) [11–13, 16, 27]. This quantity measures the rate of separation between adjacent particles over a finite time interval with an infinitesimal perturbation in the initial location. In practice, the first step to compute the FTLE is to move particles in the flow for a period of time and obtain the flow map which takes the initial particle location to its arrival location. Mathematically, the motion of these particles in the extended phase space satisfies the ordinary differential equation (ODE)

$$\dot{\mathbf{x}}(t) = \mathbf{u}(\mathbf{x}(t), t) \tag{1}$$

with a given Lipschitz velocity field $\mathbf{u} : \mathbb{R}^d \times \mathbb{R} \to \mathbb{R}^d$ and an initial condition $\mathbf{x}(t_0) = \mathbf{x}_0$. We define the flow map $\Phi_{t_0}^T : \mathbb{R}^d \to \mathbb{R}^d$ to be the mapping which takes the point $\mathbf{x}_0$ to the particle location at the final time $t = t_0 + T$, i.e $\Phi_{t_0}^T(\mathbf{x}_0) = \mathbf{x}(t_0 + T)$ with $\mathbf{x}(t)$ satisfies (1). Then the FTLE is computed from the Jacobian of the resulting flow map.

In a recent work [17], we have proposed an Eulerian formulation to compute the FTLE. The idea is to embed the flow map using a Liouville equation (see section IIA below). This converts the Lagrangian formulation into an Eulerian formulation which can be handled using simply a uniform mesh.

In this work, we are interested in developing numerical methods for moderate to long time FTLE computations leading to more efficient extraction of the LCS. We first consider an autonomous or a period flow, and propose a backward phase flow method for constructing the corresponding flow map between two time levels with a very large separation, i.e. $T \gg 1$. The idea is to iterate the flow map using a monotone interpolation scheme (see section IIB below). This gives an efficient method to approximate the FTLE on one single time level. One property of the proposed algorithm is that the computed FTLE on a single time level converges to the LE exponentially fast in the number of flow map interpolations. Stability of this algorithm will also be discussed. Further, we propose an Eulerian method to compute the LE at all time levels based on the property that the Lyapunov exponent is invariant along particle trajectories. Finally, we extend these methods to aperiodic velocity fields. Such approach turns to be useful in computations of high frequency wave propagations.

These methods share some similarities with some other recent fast algorithms. An interesting algorithm [2] has been recently suggested to speed up the time for computing the flow map between two arbitrary time levels. The idea is to decompose the flow map into a composition of maps of smaller time steps and then recycle these maps in computing the FTLE at later times. Our approach applied to autonomous flows can be interpreted as an exten-

* masyleung@ust.hk; Department of Mathematics, Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong.

sion of the latter approach, where, however, we compute only one single flow map and then iterate it to obtain a flow map for two distant time levels. Comparing to the original phase flow method [3], our algorithm solves partial differential equations (PDE) forward in time to construct the backward flow map. The original phase flow method is designed for ODE, while our backward phase flow method is especially designed for Eulerian computations. Comparing to the original backward phase flow method developed in the context of high frequency wave propagation [18], the present method employs the techniques of iterating the flow map in order to reduce the computational time. However, in the current implementation we also incorporate the concept of time-doubling, a property giving rise to an exponentially converging algorithm to the LE. Moreover, the phase flow method [3] and the backward phase flow method [18] where originally developed for autonomous (time-independent) evolutions, since the fundamental idea depends heavily on the phase flow property of an autonomous ordinary differential equation (ODE).

The paper is organized as follows. To make the paper self-contained, in section II A we first summarize and discuss the Eulerian approach in our earlier work [17] to construction the flow map. We refer interested reader to the original paper for a detailed description. Then in section II B we introduce the backward phase flow method for autonomous and periodic evolutions so that we would be able to determine the flow map between two time levels in which the difference T grows exponentially in the number the flow map interpolations. In section II C, we use the long-time flow map for computing the finite time Lyapunov exponent (FTLE). In section III, we discuss several properties of the proposed numerical approach and will give an extension of the method. In section III A we discuss the influence of the boundary condition on the backward flow method. We will study the computational complexity of the algorithm in section III B. For $T \to \infty$, FTLE converges to LE and it becomes invariant along particle characteristics. Using this property, in section III C we discuss an Eulerian approach to recover the FTLE for all intermediate time. We will also propose a generalization of the approach to aperiodic flows in section III D. Finally, in section IV we give numerical examples demonstrating the efficiency of the proposed approach.

II. THE PROPOSED METHOD

A. An Eulerian method for short-time flow maps

We define a vector-valued function $\Psi = (\Psi^1, \Psi^2, \cdots, \Psi^d) : \Omega \times \mathbb{R} \to \mathbb{R}^d$. At $t = 0$, we initialize these functions by

$$\Psi(\mathbf{x}, 0) = \mathbf{x} = (x^1, x^2, \cdots, x^d)\,. \tag{2}$$

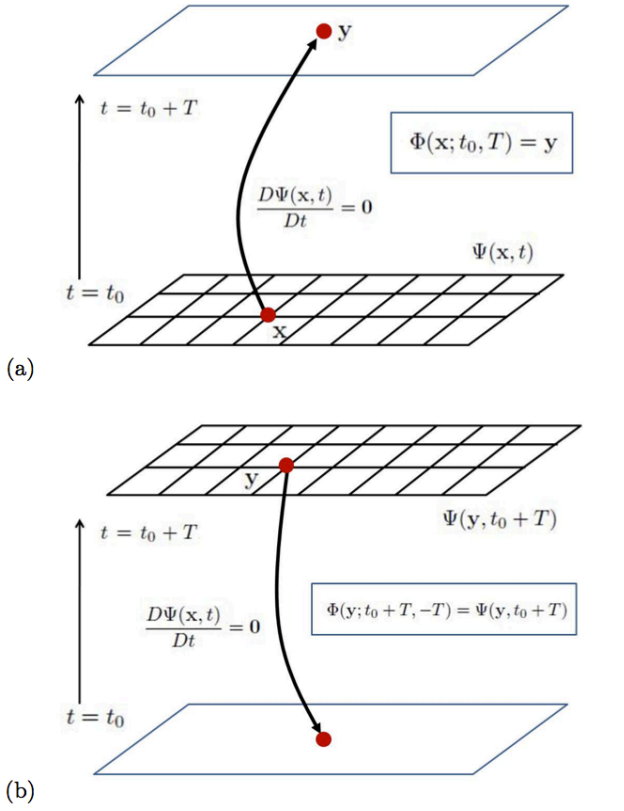

FIG. 1. Lagrangian and Eulerian interpretations of the function Ψ. (a) Lagrangian ray tracing from a given grid location $\mathbf{x}$ at $t = 0$. Note that $\mathbf{y}$ might be a non-grid point. (b) Eulerian values of Ψ at a given grid location $\mathbf{y}$ at $t = T$ gives the corresponding take-off location at $t = 0$. Note the take-off location might not be a mesh point.

These functions provide a labeling for any particle in the phase space at $t = 0$. In particular, any particle initially located at $(\mathbf{x}_0, 0) = (x_0^1, x_0^2, \cdots, x_0^d, 0)$ in the extended phase space can be **implicitly** represented by the intersection of d codimension-1 surfaces represented by $\cap_{i=1}^{d}\{\Psi^i(\mathbf{x}, 0) = x_0^i\}$ in $\mathbb{R}^d$. Following the particle trajectory with $\mathbf{x} = \mathbf{x}_0$ as the initial condition in a given velocity field, any particle identity should be preserved in the Lagrangian framework and this implies that the material derivative of these level set functions is zero, i.e.

$$\frac{D\Psi(\mathbf{x}, t)}{Dt} = \mathbf{0}\,.$$

This implies the following level set equations, or the Liouville equations,

$$\frac{\partial \Psi(\mathbf{x}, t)}{\partial t} + (\mathbf{u} \cdot \nabla)\Psi(\mathbf{x}, t) = \mathbf{0} \tag{3}$$

with the initial condition (2).

The above **implicit** representation embeds all path lines in the extended phase space. For instance, the trajectory of a particle initially located at $(\mathbf{x}_0, 0)$ can be

found by determining the intersection of d codimension-1 surfaces represented by $\cap_{i=1}^{d}\{\Psi^i(\mathbf{x},t)=x_0^i\}$ in the extended phase space. Furthermore, the forward flow map at a grid location $\mathbf{x}=\mathbf{x}_0$ from $t=0$ to $t=T$ is given by $\Phi_0^T(\mathbf{x}_0)=\mathbf{y}$ where $\mathbf{y}$ satisfies $\Psi(\mathbf{y},0+T)=\Psi(\mathbf{x}_0,0)\equiv\mathbf{x}_0$. Note that, in general, $\mathbf{y}$ is a non-mesh location. The typical two dimensional scenario is illustrated in figure 1 (a).

The solution to (3) contains much more information than what was referred to above. Consider a given mesh location $\mathbf{y}$ in the phase space at the time $t=T$, as shown in figure 1 (b), i.e. $(\mathbf{y},T)$ in the extended phase space. As discussed in our previous work, these level set functions $\Psi(\mathbf{y},T)$ defined on a uniform Cartesian mesh in fact give the backward flow map from $t=T$ to $t=0$, i.e. $\Phi_T^0(\mathbf{y})=\Psi(\mathbf{y},T)$. Moreover, the solution to the level set equations (3) for $t\in(0,T)$ provides also backward flow maps for all intermediate times, i.e. $\Phi_t^0(\mathbf{y})=\Psi(\mathbf{y},t)$.

In order, now, to produce a forward flow map, we simply reverse the above process by initializing the level set functions at $t=T$ by $\Psi(\mathbf{x},T)=\mathbf{x}$ and solving the corresponding level set equations (3) backward in time.

Algorithm 1: Computing the forward flow map $\Phi_0^T(\mathbf{x})$:

1. Discretize the computational domain to get x_i, y_j, t_k.
2. Initialize the level set functions on the last time level $t=t_k$

$$\Psi^1(x_i,y_j,t_k)=x_i$$
$$\Psi^2(x_i,y_j,t_k)=y_j\,.$$

3. Solve the Liouville equations for each individual level set function $l=1,2$

$$\frac{\partial\Psi^l}{\partial t}+(\mathbf{u}\cdot\nabla)\Psi^l=0$$

from $t=t_{k-1}$ down to $t=0$ using any well-developed high order numerical methods like WENO5-TVDRK2 [9, 20, 28] with the boundary conditions

$$\Psi(\mathbf{x},t)|_{\mathbf{x}\in\partial\Omega}=\mathbf{x}\ \text{ if } \mathbf{n}\cdot\mathbf{u}<0 \tag{4}$$
$$\mathbf{n}\cdot\nabla\Psi^l(\mathbf{x},t)|_{\mathbf{x}\in\partial\Omega}=0\ \text{ if } \mathbf{n}\cdot\mathbf{u}>0 \tag{5}$$

where $\mathbf{n}$ is the outward normal of the boundary.

4. Assign $\Phi_0^T(x_i,y_j)=\Psi(x_i,y_j,0)$.

B. A backward phase flow method for long-time flow maps

For a given $t=0$, a Lagrangian method has to solve the system of ODEs for many different initial conditions in order to compute the flow map. For computing the FTLE at the next time step $t=t_1=\Delta t$ however, those rays obtained on the previous time step $t=0$ are all discarded. This process actually throws away much useful information on the flow map [2]. To improve the computational efficiency, one interesting method is to recycle part of the information obtained in earlier steps [2]. The idea is to decompose the flow map Φ_0^T into a composition of maps $\Phi_0^T=\Phi_{t_{N-1}}^{t_N}\circ\cdots\circ\Phi_{t_1}^{t_2}\circ\Phi_0^{t_1}$ where $t_k=kT/N=k\,\Delta t$ for $k=0,1,\cdots,N$. Numerically, since each map $\Phi_{t_k}^{t_{k+1}}$ is still computed using Lagrangian ray tracing, interpolation has to be done on each step to match ray trajectories at different time level. Defining the interpolation operator by $\mathcal{I}$, we have $\Phi_0^T=\Phi_{t_{N-1}}^{t_N}\circ\cdots\circ\mathcal{I}\Phi_{t_1}^{t_2}\circ\mathcal{I}\Phi_0^{t_1}$. This approach is computational efficient since the flow map $\Phi_{t_1}^{t_1+T}$ can be decomposed into $\Phi_{t_1}^{t_1+T}=\mathcal{I}\Phi_{t_N}^{t_{N+1}}\circ\cdots\circ\mathcal{I}\Phi_{t_2}^{t_3}\circ\Phi_{t_1}^{t_2}$. If the maps $\Phi_{t_k}^{t_{k+1}}$ for $k=1,\cdots,N$ are all stored once they are computed, we can form $\Phi_{t_1}^{t_1+T}$ by determining only $\Phi_{t_N}^{t_{N+1}}$. However, the memory requirement for such implementation could be extremely large if $N\gg1$. This could be problematic in high dimensional high resolution simulations.

Now, if the flow is autonomous, we note that $\Phi_{t_k}^{t_{k+1}}=\Phi_{t_0}^{t_1}$ for $k=1,\cdots,N$ and so the method can be interpreted as a simple case of the phase flow method [3]. The original phase flow method was developed in the context of obtaining geometrical optics approximation to the wave equation in the Lagrangian ODE formulation. The idea is to first construct the phase flow map for a fixed, small step size Δt and then apply it iteratively by virtue of the group property of the phase flow. For instance, one obtains the map from $t=0$ to t_1, $\Phi_0^{\Delta t}$. Thus, the value of the phase flow map (the arrival location of a bicharacteristic) at $t_2=2\Delta t$ can be computed by

$$\Phi_0^{2\Delta t}=\Phi_{\Delta t}^{2\Delta t}\circ\Phi_0^{\Delta t}=\Phi_0^{\Delta t}\circ\Phi_0^{\Delta t}=\left(\Phi_0^{\Delta t}\right)^2\,.$$

Indeed, the phase flow method is very efficient. One can obtain the solution for large t by first constructing the flow map for a small Δt (as an overhead) and then marching forward by interpolation since the flow map is defined on an invariant manifold. Most of the computational overhead occurs during the pre-processing step. The interpolation can be easily done. To make the propagation even faster, one can use the fact that

$$\Phi_0^{2^k\Delta t}=\left(\Phi_0^{2^{k-1}\Delta t}\right)^2=\left(\left(\Phi_0^{2^{k-2}\Delta t}\right)^2\right)^2=\cdots$$

to make long time computation feasible. A detailed error analysis of the method can be found in the original paper of the phase flow method [3] and we refer interested readers to the paper for a complete description of the original algorithm.

The original idea in the phase flow method [3] is based on the Lagrangian ODE ray tracing which might be

problematic for flows in a bounded domain [36]. In a recent paper [18], we have proposed a backward phase flow method for Eulerian flow map construction which is based on an Eulerian PDE formulation. This formulation provides a more natural way to handle the boundary condition. Solving the level set function Ψ **forward** in time, we obtain the backward flow map. Here, we develop an Eulerian method for the forward flow map construction by applying the phase flow method backward in time. Mathematically, we obtain the backward flow map from $t = T$ to $T - \Delta t$, $\Phi_T^{T-\Delta t}$, for some $\Delta t > 0$ by solving the Liouville equation for Ψ forward in time from $t = T - \Delta t$ to $t = T$. Thus, the value of the flow map (the take off location of a bicharacteristic) at $t = T - 2\Delta t$ with a terminate condition $\mathbf{x} = \mathbf{x}_i$ at $t = T$ can be computed by

$$\begin{aligned}\Phi_T^{T-2\Delta t} &= \Phi_{T-\Delta t}^{T-2\Delta t} \circ \Phi_T^{T-\Delta t} \\ &= \Phi_T^{T-\Delta t} \circ \Phi_T^{T-\Delta t} = \left(\Phi_T^{T-\Delta t}\right)^2 .\end{aligned}$$

Since that particular application [17] requires to recompute other physical quantities along with the backward flow map, we have not studied in details the map size doubling technique to speedup the long time flow map computations. Therefore, we have implemented only the simple version of the backward flow map method which requires $\Phi_T^{T-k\Delta t} = \left(\Phi_T^{T-\Delta t}\right)^k$. Also, these two methods are proposed for autonomous systems in which the velocity model is independent of time. Therefore, Δt can be chosen arbitrarily.

We first extend the above approaches from autonomous to periodic flows. The idea is to develop a map doubling phase flow method for long time flow map computations. We first construct the solution $\Psi(\mathbf{x}, T^*)$ by solving the Liouville equation (3) forward in time from $t = 0$ to $t = T^*$ for some time difference T^*. For autonomous flows, one can simply pick T^* to be a small time step size Δt. For periodic flows, instead, T^* is chosen to be one period of the flow. Now, to determine $\Psi(\mathbf{x}, 2T^*)$, we use the phase flow property and obtain $\Psi(\mathbf{x}, 2T^*) = \Psi(\Psi(\mathbf{x}, T^*), T^*) = \Psi \circ \Psi(\mathbf{x}, T^*)$. In general, once we have obtained the solution $\Psi(\mathbf{x}, 2^{k-1}T^*)$, we can obtain $\Psi(\mathbf{x}, 2^k T^*) = \Psi(\Psi(\mathbf{x}, 2^{k-1}T^*), 2^{k-1}T^*)$. Finally, if we take $T = 2^m T^*$, the backward flow map $\Phi_{2^m T^*}^0(\mathbf{x})$ is given by $\Phi_{2^m T^*}^0(\mathbf{x}) = \Psi(\mathbf{x}, T)$.

The idea to compute the forward flow map is simple. We just need to reverse the time direction in the previous algorithm. For example, to compute the flow map from 0 to T, we solve the Liouville equation backward in time from $t = T$ to $t = T - T^*$. Then we iterate the map k-times to get the overall flow map forward in time from $t = 0$ to $t = T = T^* \cdot 2^k$. Here, we summarize the backward phase flow method for forward flow map computation.

Algorithm 2: Computing the forward flow map $\Phi_0^{2^m T^*}(\mathbf{x})$

for a T^*-periodic flow with $T = 2^m T^*$:

1. Construct $\Psi(\mathbf{x}, (2^m - 1)T^*)$ as in Algorithm 1, which gives $\Phi_{(2^m-1)T^*}^{T^*}(\mathbf{x})$.
2. For $k = 1, \cdots, m$, interpolate

$$\begin{aligned}\mathbf{y} &= \Psi(\mathbf{x}, (2^m - 2^{k-1})T^*) \\ \Psi(\mathbf{x}, (2^m - 2^k)T^*) &= \Psi(\mathbf{y}, (2^m - 2^{k-1})T^*)\end{aligned}$$

   using any monotone interpolating scheme [6, 14, 24, 31].
3. Set $\Phi_0^{2^m T^*}(\mathbf{x}) = \Psi(\mathbf{x}, 0)$.

C. The finite time Lyapunov exponent (FTLE)

With the backward flow map defined on a uniform mesh at $t = T$, we first compute the deformation tensor

$$\Delta_T^0(\mathbf{x}) = [\mathcal{D}\Phi_T^0(\mathbf{x})]^* \mathcal{D}\Phi_T^0(\mathbf{x}) ,$$

where $\mathcal{D}\Phi$ is the Jacobian of the flow map Φ and $[\cdot]^*$ denotes the transpose of the matrix. Then the backward FTLE $\sigma^{-T}(\mathbf{x}, T)$ is computed according to

$$\sigma^{-T}(\mathbf{x}, T) = \frac{1}{|T|} \ln \sqrt{\lambda_{\max}[\Delta_T^0(\mathbf{x})]} .$$

Numerically, the derivatives in the Jacobian are approximated using simple finite differences.

Similarly, for the forward FTLE at $t = 0$, we first determine the Jacobian of the resulting flow map and then compute the largest eigenvalue of the deformation tensor $\Delta_0^T(\mathbf{x})$. Then the forward FTLE is formed using a similar expression.

III. SOME PROPERTIES AND FURTHER EXTENSIONS

A. Boundary condition and the interpolation scheme

As in the previous paper [17], we impose the boundary condition (4) on the in-flow boundary condition to the Liouville equation for computing the backward flow map by solving the time-dependent PDE forward in time, i.e. we fix the inflow boundary condition on $\partial\Omega_i = \{\mathbf{x} \in \partial\Omega : \mathbf{n} \cdot \mathbf{u}(\mathbf{x}) < 0\}$ where $\mathbf{n}$ is the outward normal of the domain Ω. With this treatment, we have

Lemma III.1 *With the boundary condition (4) on the inflow boundary, the exact solution $\Psi = (\Psi^1, \Psi^2)$ satisfies*

$$\begin{aligned}\Psi^1(x, y, t) &\in [x_{\min}, x_{\max}] \\ \Psi^2(x, y, t) &\in [y_{\min}, y_{\max}]\end{aligned} \tag{6}$$

for all $(x, y) \in \Omega$ and $t \in [0, T]$.

This lemma can be easily proven because both the initial condition and the inflow boundary condition satisfy (6) with the property that the solution to the linear advection equation is constant along any characteristics.

Numerically, step 2 in Algorithm 2 has to be done with care. If the underlying flow map function is not smooth enough (for example if the derivative of the underlying function exhibits jumps, i.e. the function exhibits kinks), the widely used bicubic spline interpolation schemes might generate spurious oscillations. In the application to geometrical optics or high frequency wave propagations, such overshooting or undershooting in the interpolant will create artificial caustics since the ordering of rays cannot be preserved after implementing the interpolation step. This effect is not so important in applications, because such perturbation occurs only in small, not easily recognizable, regions. In the current application, however, these oscillations could lead to a serious problem in the implementation. In particular, such overshooting or undershooting may violate Lemma III.1 and therefore one might not be able to evaluate $\Psi(\Psi(\mathbf{x}, T-T^*), T-T^*)$.

To obtain a stable evolution in the flow map constructions, we require the interpolation scheme to be monotone, i.e. the interpolation scheme should preserve the monotonicity of the given data points. Note that this monotonicity preserving condition is imposed on the interpolation scheme, rather than on the numerical scheme for solving the PDE for stability consideration. Mathematically, we call an interpolation scheme monotone if the constructed interpolating polynomial $p(x)$ is monotone on $x \in [x_1, x_N]$ for any monotonic input data $\{(x_i, p_i), i = 1, 2, \cdots, N\}$. For example, usual cubic spline or high order polynomial interpolation are unfortunately not monotone. The simplest monotone interpolation scheme is bilinear interpolation. However, such simple scheme might give low order accurate flow map solution which will smooth out the resulting Lyapunov exponent after several flow map iterations. In practice, there are many possible choices of high order monotone interpolating schemes [6, 14, 24, 31]. In this paper, we use the piecewise bicubic Hermite interpolation method [7] implemented using the MATLAB function pchip. With the monotonicity constraint imposed on the interpolation scheme, we have the following important property, guaranteeing the availability of sufficient data of the interpolation which would lead to the robustness in the phase flow method.

Lemma III.2 *If the flow map* $\Psi = (\Psi^1, \Psi^2)$ *constructed in* step 1 *of* Algorithm 2 *satisfies*

$$x_{\min} \le \Psi^1(x_i, y_j, (2^m - 1)T^*) \le x_{\max}$$
$$y_{\min} \le \Psi^2(x_i, y_j, (2^m - 1)T^*) \le y_{\max},$$

and if the interpolation scheme is monotone, we have

$$x_{\min} \le \Psi^1(x_i, y_j, (2^m - 2^k)T^*) \le x_{\max}$$
$$y_{\min} \le \Psi^2(x_i, y_j, (2^m - 2^k)T^*) \le y_{\max}$$

for all $i = 1, 2, \cdots, I$, $j = 1, 2, \cdots, J$ *and* $k = 1, 2, \cdots, m$.

B. Computational complexity

In this subsection, we discuss the computational complexity of our algorithm as regards the determination of the FTLE for a two dimensional periodic flow. Generalization to higher dimensions is straight-forward. As discussed in the previous paper [17], the computational complexity for step 1 in Algorithm 2 is $O(N^2 M)$ where N is the number of mesh points in each spatial dimension and M is the number of mesh points in the time dimension. Because the Eulerian approach requires solving hyperbolic equations, the CFL stability condition implies $M = O(N)$, i.e. step 1 requires $O(N^3)$ operations for a fixed T^*.

Now, in step 2 of Algorithm 2, we perform two-dimensional interpolations. A one-dimensional Hermite interpolation requires $O(N)$ operations to obtain all piecewise polynomials. For bi-dimensional interpolations at a given point (x^*, y^*), we first compute all one-dimensional piecewise polynomials. This requires $O(N^2)$ operations. Note that since these polynomials can be re-used for all interpolation locations, we store all these values in the memory, in order to reduce the overall computational complexity. Now, the first step is to evaluate these polynomials at $x = x^*$ along each $y = y_j$ for $j = 1, 2, \cdots, J = O(N)$. This needs $O(N)$ operations. Once we obtain the values at (x^*, y_j), we perform another one-dimensional Hermite interpolation, which requires additional $O(N)$ operations. Therefore, the total number of operations in step 2 is $O(N^2) + O(N^2) \cdot [O(N) + O(N)] = O(N^3)$.

Let $T^* = M\Delta t$ be the period of the flow, Δt is the step size in the time-marching and $T = T^* \cdot 2^k$. To summarize, the overall computational complexity of a typical Lagrangian approach and the original Eulerian approach [17] are $O(N^2 M 2^k)$ and $O(N^3 2^k)$, repectively, while the current proposed algorithm has the complexity $O(N^3) + k \cdot O(N^3) = O(kN^3)$ which is significantly smaller.

C. Time evolution of the FTLE

Following a similar approach as in our previous work [17], we now propose a way to compute the FTLE at various time levels. Since the Lyapunov exponent σ is constant along particle trajectories, it satisfies the following equation along the trajectory $\frac{D}{Dt}\sigma(\mathbf{x}, t) = 0$ or the following partial differential equation in the Eulerian framework

$$\frac{\partial \sigma(\mathbf{x}, t)}{\partial t} + \mathbf{u} \cdot \nabla \sigma(\mathbf{x}, t) = 0\,. \tag{7}$$

In our previous approach [17], we have proposed to use (7) in order to approximate the finite time Lyapunov exponent $\sigma^T(\mathbf{x}, T+S)$ for some $S>0$ using $\sigma^T(\mathbf{x},T)$ with an error growing linearly in S. In this paper, since k iterations of the flow map already gives the flow map for a period of time $t=T=T^*\cdot 2^k$ (which grows exponentially in k), we are using the solution of the equation (7) to approximate directly the Lyapunov exponent. Since the error is of order $O(1/T)$ [27], our solution converges to the Lyapunov exponent by an error $O(2^{-k})$, i.e., exponentially fast in the number of flow map iterations.

D. Extension to aperiodic flows

If the flow field depends on time and is not periodic, one cannot directly apply the approach in Section II B to iteratively interpolate the obtained flow map for long time propagation because in general $\Phi_{t_{k-1}}^{t_k} \neq \Phi_{t_k}^{t_{k+1}}$. Here, we propose a simple way to extend the phase flow method to aperiodic flows. For simplicity, we will consider the Eulerian formulation for the backward FTLE construction so that we solve the Liouville equation forward in time. The method can be easily generalized to compute the forward FTLE by reversing the time direction.

To apply the backward phase flow method to an aperiodic flow, we convert the flow field into an autonomous field by introducing a new variable s so that the ODE system becomes

$$\begin{aligned}\dot{\mathbf{x}}(t) &= \mathbf{u}(\mathbf{x}, s)\\ \dot{s}(t) &= 1\,.\end{aligned}$$

Now, in the ODE framework one can directly apply the phase flow method to the $\mathbf{x}-s$ space (which is one dimension higher than the original physical space). In this paper, we do not discuss this Lagrangian approach in details but will concentrate on it's Eulerian counterpart. We consider the corresponding Liouville equations

$$\frac{\partial \Psi(\mathbf{x},s,t)}{\partial t} + \left(\mathbf{u}\cdot\nabla + \frac{\partial}{\partial s}\right)\Psi(\mathbf{x},s,t) = \mathbf{0}$$

in the computational domain $[\mathbf{x},s,t] \in \Omega\times[0,T]\times[0,T]$ with the initial condition $\Psi^1(\mathbf{x},s,0)=\mathbf{x}^1$, $\cdots$, $\Psi^d(\mathbf{x},s,0)=\mathbf{x}^d$ and $\Psi^{d+1}(\mathbf{x},s,0)=s$, where Ψ^k is the k-the component of the vector function Ψ and $\mathbf{x}^k$ is the k-th dimension of $\mathbf{x}$. Since the velocity field is now independent of time, we can solve for $\Psi(\mathbf{x},s,T^*)$ for an arbitrary $T^*>0$ to construct the initial flow map $\Phi_{T^*}^0 : (\mathbf{x}(T^*), s(T^*)) \to (\mathbf{x}(0), s(0))$. Then, following an algorithm similar to Algorithm 2, we obtain the long time flow map $\Phi_T^0 : (\mathbf{x}(T), s(T)) \to (\mathbf{x}(0), s(0))$ by interpolation

$$\Phi_T^0 = \Phi_{2^kT^*}^0 = \Phi_{2^{k-1}T^*}^0 \circ \Phi_{2^kT^*}^{2^{k-1}T^*} = \left(\Phi_{2^kT^*}^{2^{k-1}T^*}\right)^2 .$$

With the backward flow map Φ_T^0, we get the backward FTLE $\sigma^{-T}(\mathbf{x},T)$.

Finally, we point out that all intermediate solutions $\Psi(\mathbf{x},s,t)$ also contain useful information. Mathematically, these functions provide **all** flow maps $\Phi_s^{s-t}(\mathbf{x})$ which yield the initial location of the particles at the take-off time $(s-t)$ when the particle arrives at the physical location x at time s. In view of this, the current approach can also be regarded as a systematic Eulerian generalization of the Lagrangian method proposed earlier [2].

IV. EXAMPLES

In this section, we consider two examples to demonstrate the effectiveness of the algorithm. Since some cases of the following systems are integrable, the FTLE tends to zero as $T\to\infty$. Therefore, instead of showing the FTLE directly, we have scaled the quantity by T (i.e. we are showing $T\,\sigma^T$ or $T\,\sigma^{-T}$) so that all computed solutions can be depicted in a suitable color scale.

A. Double-gyre flow

The example is based on the flow taken from Shadden et al. [27] to describe a periodically varying double-gyre. The flow is modeled by the following stream-function $\psi(x,y,t) = A\sin[\pi f(x,t)]\sin(\pi y)$, where $f(x,t) = a(t)x^2+b(t)x$, $a(t)=\epsilon\sin(\omega t)$, and $b(t)=1-2\epsilon\sin(\omega t)$. The velocity field can be obtained by $u(x,y)=-\partial\psi/\partial y$ and $v(x,y)=\partial\psi/\partial x$. In this example, we follow the original paper [27] and use $A=0.1$ and $\omega=2\pi/10$. The parameter ϵ models the magnitude of the periodic perturbation.

We will consider two different cases for this flow. The first case is an autonomous flow where we turn off the periodic perturbation in the x-direction by setting $\epsilon=0$, i.e. $\psi(x,y)=A\sin(\pi x)\sin(\pi y)$. Therefore, we are allowed to apply the backward phase flow method for long term computation using any T^* in constructing the initial flow map. In this case, we choose $T^*=0.1$ and iterate the map for 15 times, which gives the flow map for time period $t=T=T^*\cdot 2^{15}=3276.8$. For comparison, since $u_yv_x-u_xv_y=\frac{1}{2}A^2\pi^4(\cos 2\pi x+\cos 2\pi y)$, the condition for absolute stability using RK4 is approximately bounded by $3/(A\pi^2)\simeq 3$, which implies the number of time levels needed for a Lagrangian method is over 1000. For the proposed Eulerian method with $\max\sqrt{u^2+v^2}=\pi A$ using the CFL number 0.5, the number of time levels is only approximately $8+15=23$ for the case with $\Delta x=1/128$, and $16+15=31$ for the case with $\Delta x=1/256$. Figure 2 (a-c) shows the FTLE obtained using a computational mesh of $\Delta x=\Delta y=1/256$. Once we have obtained the forward flow map from $t=0$ to $t=T$, we construct the corresponding forward FTLE on $t=0$ and then propagate it to $t=T$. The solutions are shown in figure 2 (d).

We have also plotted several ray trajectories corresponding to figure 2 (a) in figure 3. Near the region

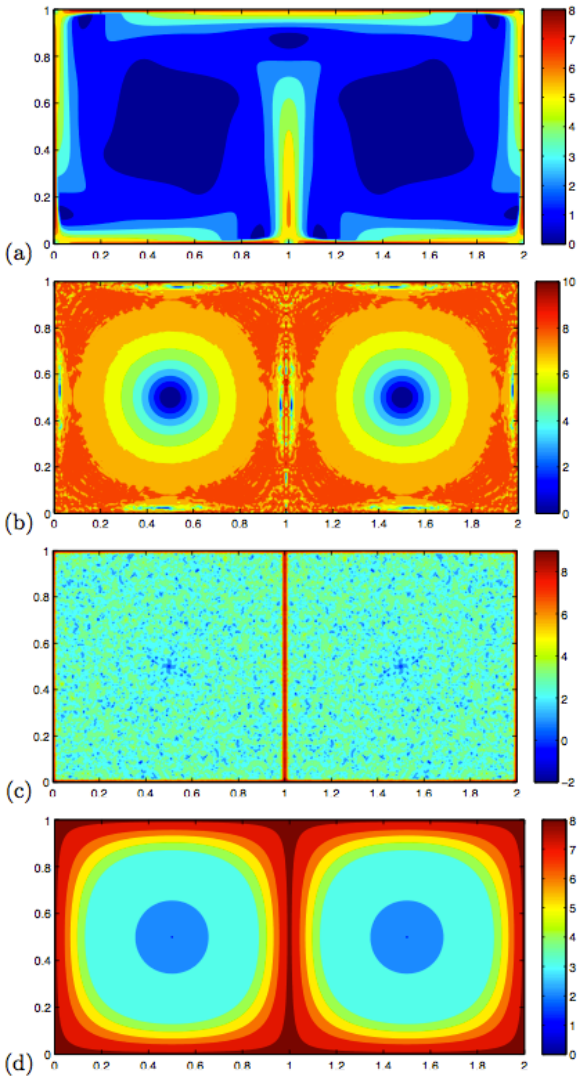

FIG. 2. (Section IV A with $\epsilon = 0.0$) The scaled forward FTLE using the proposed backward phase flow method, i.e. $T\sigma^T$: $\Delta x = 1/256$, $T^* = 0.1$. We apply the backward phase flow method to obtain the scaled FTLE at (a-c) $t = T^* \cdot (2^{15} - 2^{5k})$ for $k = 1, 2, 3$ with $T = T^* 2^{5k}$. (d) Once we have obtained $3276.8\,\sigma^{3276.8}(\mathbf{x}, 0)$, we propagate the solution to obtain $3276.8\,\sigma^{3276.8}(\mathbf{x}, 3276.8)$.

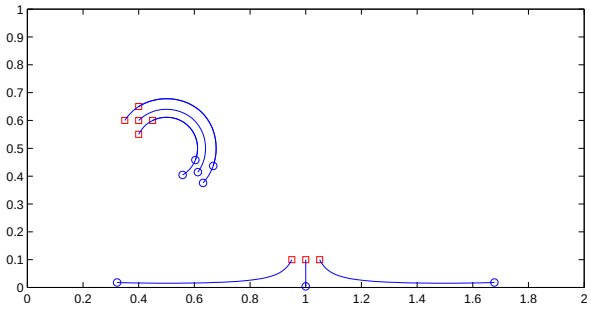

FIG. 3. (Section IV A with $\epsilon = 0.0$) Trajectories of several particles corresponding to figure 2 (a). Initial locations are plotted in (red) square and final locations are plotted in (blue) circle.

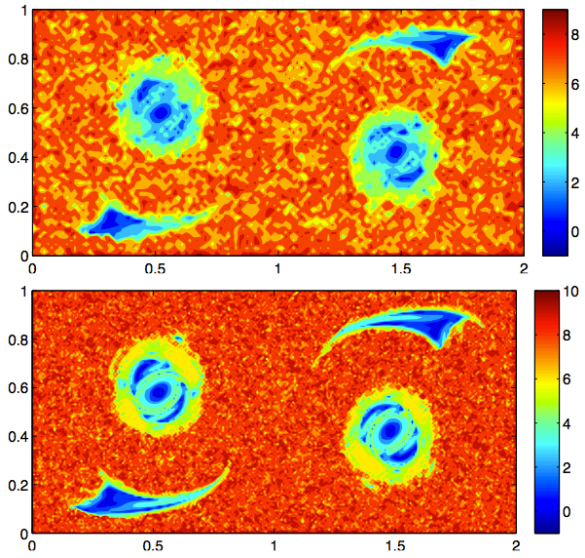

FIG. 4. (Section IV A with $\epsilon = 0.1$) The scaled forward FTLE at $t = 0$ using the **Lagrangian** approach, i.e. $T\sigma^T$: $T = 320$, $\Delta x = 1/64$ and $1/128$, respectively.

$x = 1$ and y close to zero, The computed FTLE is relatively large and this implies that a small perturbation in the initial condition leads to a relatively large change in the final location at $t = 2^5 T^* = 3.2$. In figure 3, we have plotted three initial points at $(0.95, 0.1)$, $(1, 0.1)$ and $(1.05, 1)$ in red squares and we can clearly see that their paths are very different. On the other hand, the FTLE near the region $(0.5, 0.5)$ is close to zero and therefore, as expected, particles started in that region (the remaining five points in figure 3) tend to travel together as a patch.

The second case is a flow with periodic perturbation, i.e. $\epsilon = 0.1$. The x-directional perturbation has a period $2\pi/\omega = 10$. Figure 4 shows the solution obtained by a typical Lagrangian method where the ODE system is solved using a fixed-step fourth-order Runge-Kutta method (RK4) with spatial resolution $1/64$ and $1/128$. The total computational time for obtaining these solutions are approximately 131 minutes and 504 minutes, which approximately follows the computational complexity of $O(N^2 M)$ where N is the number of mesh points in each spatial direction and M is the total number of time steps in the ODE integrator. In this computation, we simply pick a fix $\Delta t = 0.1$.

To construct the initial flow map at $T^* = 10$ using our proposed backward phase flow method, we solve the Liouville equation up to $T^* = 10$. Then we apply the backward phase flow idea to get the flow map at time $T^* \cdot 2^5 = 320$. In figure 5, we discretize the domain $[0, 2] \times [0, 1]$ using $\Delta x = \Delta y = 1/256$ and we show the

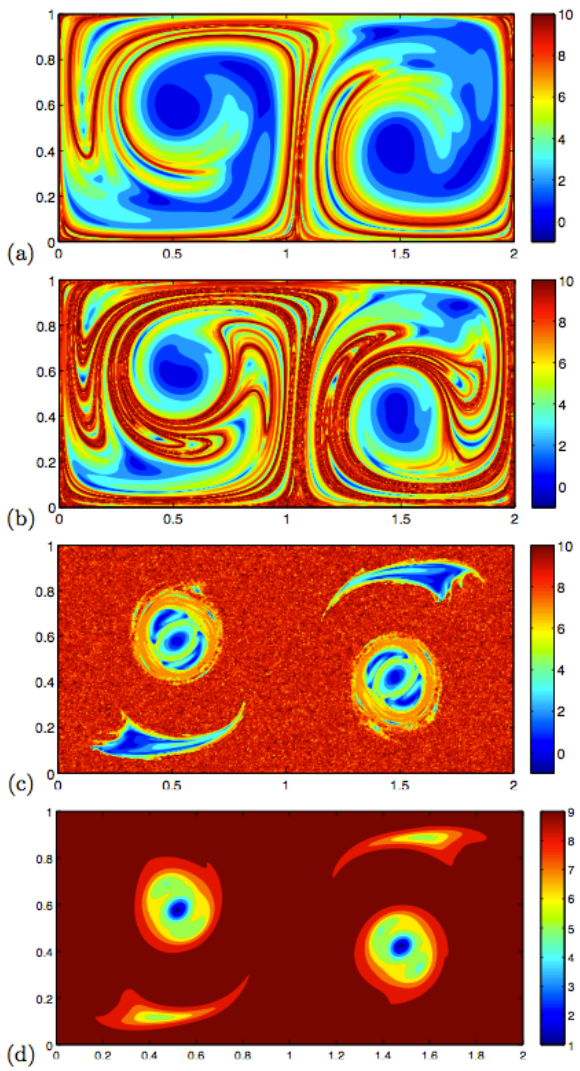

FIG. 5. (Section IV A with $\epsilon = 0.1$) The scaled forward FTLE using the proposed backward phase flow method, i.e. $T\sigma^{T}$: $\Delta x = 1/256$, $T^* = 10$. (a-c) We apply the backward phase flow method to obtain the scaled FTLE's $t_k\,\sigma^{t_k}(\mathbf{x}, 0)$ for $t_k = T^* \cdot 2^5 \cdot k/5$ and $k = 1, 2$ and 5. (d) Once we have obtained $320\,\sigma^{320}(\mathbf{x}, 0)$, we propagate it to obtain $320\,\sigma^{320}(\mathbf{x}, 320)$.

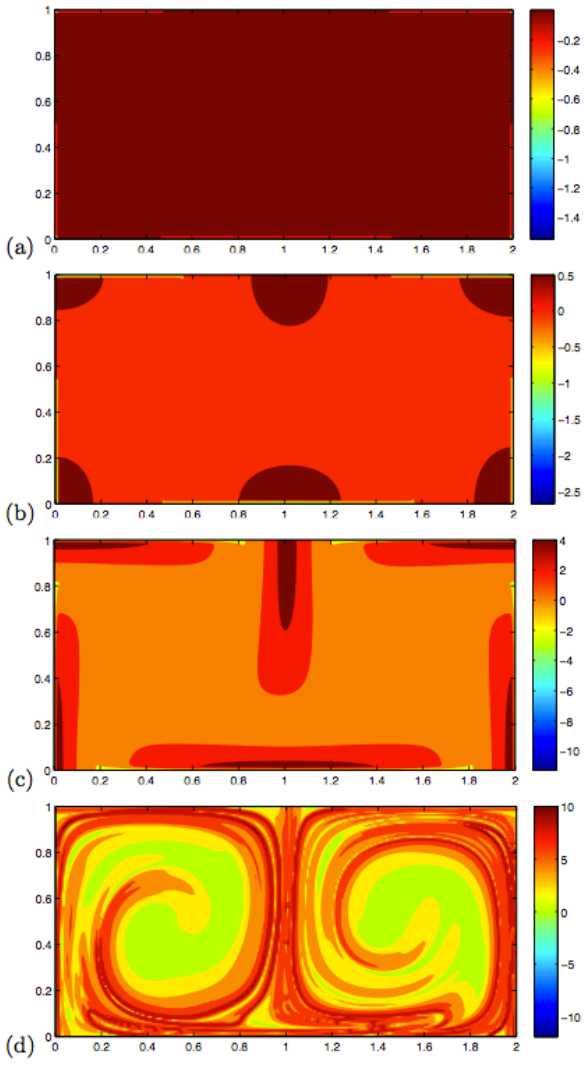

FIG. 6. (Section IV A with an aperiodic perturbation (8)) The scale backward FTLE for an aperiodic flow using the proposed backward phase flow method, i.e. $T\sigma^{-T}$: $\Delta x = \Delta y = 1/256$, $\Delta z = 10/128$, $T^* = 10/2^{12}$. We iterate the obtained flow map for (a) 3, (b) 6, (c) 9 and (d) 12 times to obtain the scaled backward FTLE (a) $2^3T^*\,\sigma^{-2^3T^*}(\mathbf{x}, 10)$, (b) $2^6T^*\,\sigma^{-2^6T^*}(\mathbf{x}, 10)$, (c) $2^9T^*\,\sigma^{-2^9T^*}(\mathbf{x}, 10)$ and (d) $2^{12}T^*\,\sigma^{-2^{12}T^*}(\mathbf{x}, 10)$.

forward FTLE at $t = 0$ using $\Delta x = \Delta y = 1/256$. After 5 flow map iterations, we obtain the forward FTLE from $t = 0$ to $t = t_f = 10 \cdot 2^5 = 320$, as shown in figure 5 (c). Once we have obtained this solution, we propagate it forward in time to get an approximation to the Lyapunov exponent at various time levels, as shown in figure 5 (d). The total computational time needed to obtain these results is 47 minutes. The corresponding computational time for some coarser meshes $\Delta x = \Delta y = 1/64$ and $\Delta x = \Delta y = 1/128$ are approximately 1 minute and 7 minutes, respectively. Since the backward FTLE is no more difficult to compute than the forward FTLE, we have skipped the numerical solutions here.

Next, we compute the backward FTLE of an aperiodic dynamical system. We modify the stream-function by replacing the periodic functions $a(t)$ and $b(t)$ by

$$\begin{aligned} a(t) &= \epsilon \sin[\omega t(1 + \sin t/2)]\,, \\ b(t) &= 1 - 2\epsilon \sin(\omega t(1 + \sin t/2)] \end{aligned} \tag{8}$$

with the same parameters $\epsilon = 0.1$, $A = 0.1$ and $\omega = 2\pi/10$.

Figure 6 shows our computed results of various backward FTLE at $t = 10$ using a mesh of $257 \times 129 \times 129$ in $[0, 2] \times [0, 1] \times [0, 10]$. We use the backward phase flow method by first constructing a flow map of size

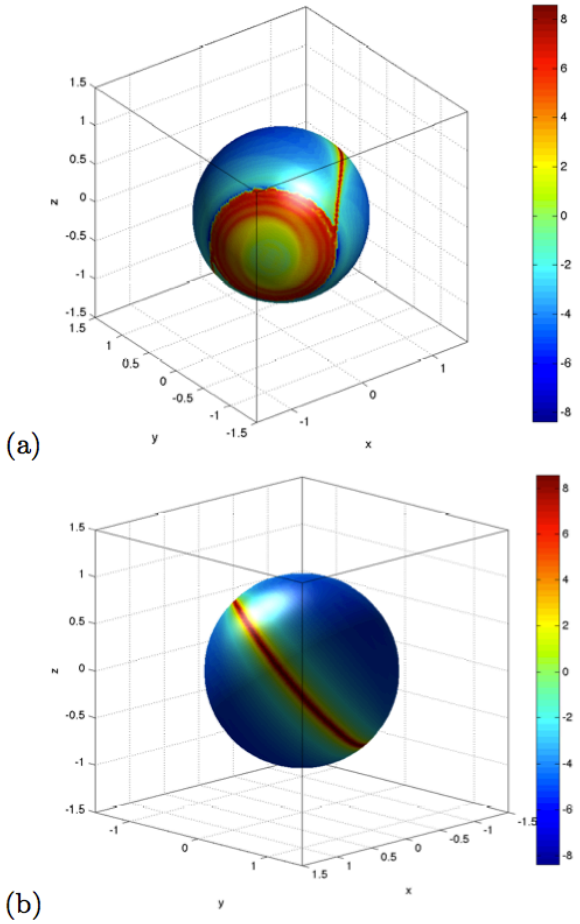

FIG. 7. (Section IV B) The scaled backward FTLE using the proposed backward phase flow method, i.e. $T\sigma^{-T}$: $\Delta x = 3/128$, $T^* = 0.0625$. We apply the backward phase flow method to obtain the scaled FTLE's $64\,\sigma^{-64}(\mathbf{x}, 64)$.

$T^* = 10/2^{12}$. Then we iterate the map for 12 times to obtain the backward FTLE $\sigma^{-2^{12}T^*}(\mathbf{x}, 10) = \sigma^{-10}(\mathbf{x}, 10)$, as shown in figure 6 (d). Note that all intermediate solutions $\Psi(x_i, y_j, s_k, T^*)$ define the flow maps $(x_i, y_j, s_k) \to (x(s_k - T^*), y(s_k - T^*), s_k - T^*)$ and so we can use them to compute the backward FTLE $\sigma^{-T^*}(x_i, y_j, s_k)$. Figure 6 shows the solutions at $s_k = 10$ for $T^* = 2^p T^*$ for $p = 3, 6, 9$ and 12.

B. Point vertex flow on a sphere

In this last example, we consider an advection motion in a field of two point vortices on a sphere of radius 1 centered at the origin [23]. The velocity of a particle on the manifold satisfies the motion

$$\mathbf{x}' = \frac{1}{2\pi} \sum_{i=1}^{2} \frac{\mathbf{x}_i \times \mathbf{x}}{2(1 - \mathbf{x}_i \cdot \mathbf{x})}$$

with two point vortices placed at $(-1, 0, 0)$ and $(0, -1, 0)$. We follow the same Eulerian idea proposed earlier [17] to represent the sphere in $\mathbb{R}^3$ implicitly using a level set function. Then the flow map is computed on the uniform Cartesian mesh without explicitly triangulating the surface. Figure 7 shows the backward FTLE solutions computed using the proposed backward phase flow method at $t = 2$ and $t = 64$ on an underlying uniform mesh of size $\Delta x = \Delta y = \Delta z = 3/128$. We first compute the flow map at $t = T^* = 2^{-4}$ and then iterate the map 10 times to obtain the final flow map at $t = T^* \cdot 2^{10} = 64$. The LCS is clearly observed on the back of the sphere in (b).

V. ADVANTAGES, LIMITATIONS AND FUTURE WORK

Based on the backward phase flow method, we develop an efficient numerical approach to compute the long-time flow map in aperiodic flows. The algorithm can be used to approximate directly the Lyapunov exponent of autonomous, periodic or even aperiodic dynamical systems. Because of the time doubling property of the method, the FTLE converges to the LE exponentially fast in the number of flow map iterations. Another advantage of the proposed Eulerian approach is that we require the velocity field to be defined only on the underlying mesh. No interpolation of the velocity field is necessary. Also, the Eulerian approach gives a more natural way to impose the boundary condition in the bounded computational domain.

For aperiodic flows, computation is performed in the $\mathbf{x} - s - t$ space which is one dimension higher than that of the physical space. Therefore, one limitation of the proposed approach is the extra requirement in memory. This extra dimension poses severe barrier for long time FTLE computations such as $\sigma^{-320}(\mathbf{x}, 320)$ in figure 6.

Besides the FTLE, there are other important quantities used in the study of the chaotic behavior of dynamical sytems[22, 29]. For example, other possible indicators include the invariant spectra [34, 35], the finite size Lyapunov exponent (FSLE) [1], the mean exponential growth factor of nearby orbits (MEGNO) [5], the fast Lyapunov indicator [8], the relative Lyapunov indicator [25], the generalized alignment index (GALI) method [30], and etc. We propose, as future works, to extend our current method in order to incorporate some of these indicators of chaos detection.

ACKNOWLEDGEMENT

The author would like to thank the anonymous reviewers for their extremely helpful and constructive comments and suggestions that greatly improve the quality of the manuscript. The work was supported in part by the Hong Kong RGC under Grant 602210 and 605612.

[1] E. Aurell, G. Boffetta, A. Crisanti, G. Paladin, and A. Vulpiani. Predictability in the large: an extension of the concept of lyapunov exponent. *J. Phys. A: Math. Gen.*, 30:1–26, 1997.

[2] S. Brunton and C. Rowley. Fast computation of finite-time Lyapunov exponent fields for unsteady flows. *Chaos*, 20:017503, 2010.

[3] E. Candès and L. Ying. Fast geodesics computation with the phase flow method. *J. Comput. Phys.*, 220:6–18, 2006.

[4] B. Cardwell and K. Mohseni. Vortex shedding over two-dimensional airfoil: Where do the particles come from? *AIAA J.*, 46:545–547, 2008.

[5] P. Cincotta and C. Simo. Simple tools to study global dynamics in non-axisymmetrix galactic potentials - i. *Astron. Astrophys. Suppl. Ser.*, 147:205–228, 2000.

[6] C. de Boor and B. Swartz. Piecewise monotone interpolation. *Journal of Approximation Theory*, 21:411–416, 1977.

[7] F. Fritsch and R. Carlson. Monotone piecewise cubic interpolation. *SIAM J. Numerical Analysis*, 17:238–246, 1980.

[8] C. Froeschle and E. Lega. On the structure of symplectic mapplings. the fast lyapunov indicator: a very sensitive tool. *Celestial Mechanics and Dynamical Astronomy*, 78:167–195, 2000.

[9] S. Gottlieb and C.-W. Shu. Total variation diminishing Runge-Kutta schemes. *Mathematics of Computation*, 67:73–85, 1998.

[10] M. Green, C. Rowley, and A. Smiths. Using hyperbolic Lagrangian coherent structures to investigate vortices in biospired fluid flows. *Chaos*, 20:017510, 2010.

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

[12] G. Haller. Lagrangian structures and the rate of strain in a partition of two-dimensional turbulence. *Phys. Fluids A*, 13:3368–3385, 2001.

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

[14] H. Huynh. Accurate monotone cubic interpolation. *NASA Technical Memorandum 103789*, 1991.

[15] F. Lekien and N. Leonard. Dynamically consistent Lagrangian coherent structures. *Experimental Chaos: 8-th Experimental Chaos Conference*, pages 132–139, 2004.

[16] F. Lekien, S. Shadden, and J. Marsden. Lagrangian coherent structures in n-dimensional systems. *Journal of Mathematical Physics*, 48:065404, 2007.

[17] S. Leung. Eulerian approach for computing the finite time Lyapunov exponent. *J. Comput. Phys.*, 230:3500–3524, 2011.

[18] S. Leung and J. Qian. The backward phase flow and FBI-transform-based Eulerian Gaussian beams for the Schrödinger equation. *J. Comput. Phys.*, 229:8888–8917, 2010.

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

[20] X. D. Liu, S. J. Osher, and T. Chan. Weighted Essentially NonOscillatory schemes. *J. Comput. Phys.*, 115:200–212, 1994.

[21] S. Lukens, X. Yang, and L. Fauci. Using Lagrangian coherent structures to analyze fluid mixing by cillia. *Chaos*, 20:017511, 2010.

[22] N. Maffione, L. Darriba, P. Cincotta, and C. Giorgano. A comparison of different indicators of chaos based on the deviation vectors: application to sympletic mappings. *Celestial Mechanics and Dynamical Astronomy*, 111:285–307, 2011.

[23] P. Newton. *The N-Vortex Problem: Analytical Techniques*. Springer-Verlag, 2001.

[24] E. Passow. Piecewise monotone spline interpolation. *Journal of Approximation Theory*, 12:240–241, 1974.

[25] Z. Sandor, B. Erdi, A. Szell, and B. Funk. The relative lyapunov indicator: an efficient method of chaos detection. *Celestial Mechanics and Dynamical Astronomy*, 90:127–138, 2004.

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

[27] 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*, 212:271–304, 2005.

[28] C. W. Shu. Essentially non-oscillatory and weighted essentially non-oscillatory schemes for hyperbolic conservation laws. In B. Cockburn, C. Johnson, C. Shu, and E. Tadmor, editors, *Advanced Numerical Approximation of Nonlinear Hyperbolic Equations*, volume 1697, pages 325–432. Springer, 1998. Lecture Notes in Mathematics.

[29] C. Skokos. The lyapunov characteristic exponents and their computation. *Lect. Notes Phys.*, 790:63–135, 2010.

[30] C. Skokos, T. Bountis, and C. Antonopoulos. Geometrical properties of local dynamics in hamiltonian systems: The generalized alignment index (gali) method. *Physica D*, 231:30–54, 2007.

[31] P. Smolarkiewicz and G. Grell. A class of monotone interpolation schemes. *J. Comput. Phys.*, 101:431–440, 1992.

[32] W. Tang, P. Chan, and G. Haller. Accurate extraction of Lagrangian coherent structures over finite domains with application to flight data analysis over Hong Kong international airport. *Chaos*, 20:017502, 2010.

[33] W. Tang and T. Peacock. Lagrangian coherent structures and internal wave attractors. *Chaos*, 20:017508, 2010.

[34] N. Voglis and G. Contopoulos. Invariant spectra of orbits in dynamical systems. *J. Phys. A: Math. Gen.*, 27:4899–4909, 1994.

[35] N. Voglis and C. Efthymiopoulos. Angular dynamical spectra. a new method for determining frequencies, weak chaos and cantori. *J. Phys. A: Math. Gen.*, 31:2913–2928, 1998.

[36] H. Wu and X. Yang. A hybrid phase flow method for solving the liouville equation in a bounded domain. *SIAM J. Numer. Anal.*, 49:733–754, 2011.