# Integrated computation of finite-time Lyapunov exponent fields during direct numerical simulation of unsteady flows

Justin Finn[a] and Sourabh V. Apte[b]
*School of Mechanical, Industrial and Manufacturing Engineering*
*Oregon State University, 204 Rogers Hall, Corvallis, OR 97331*

(Dated: 5 March 2013)

The computation of Lagrangian coherent structures typically involves post-processing of experimentally or numerically obtained fluid velocity fields to obtain the largest finite-time Lyapunov exponent (FTLE) field. However, this procedure can be tedious for large-scale complex flows of general interest. In this work, an alternative approach involving computation of the FTLE on-the-fly during direct numerical simulation of the full three dimensional Navier-Stokes equations is developed. The implementation relies on Lagrangian particle tracking to compose forward time flow maps, and an Eulerian treatment of the backward time flow map [S. Leung, Journal of Computational Physics 230, 2011] coupled with a semi-Lagrangian advection scheme. The flow maps are accurately constructed from a sequence of smaller sub-steps stored on disk [S. Brunton and C. Rowley, Chaos 20, 2010], resulting in low CPU and memory requirements to compute evolving FTLE fields. Several examples are presented to demonstrate the capability and parallel scalability of the approach for a variety of two and three dimensional flows.

PACS numbers: Valid PACS appear here
Keywords: Suggested keywords

**The notion of Lagrangian coherent structures (LCS) as invariant transport barriers in steady and unsteady fluid flows has evolved from dynamical systems theory, and has proven its utility in understanding a number of mixing and transport problems. These structures are most often defined from ridges in the finite-time Lyapunov exponent field. However, their practical use has been limited to this point because computing the FTLE field typically involves expensive post-processing of large fluid velocity datasets generated either from experiments or numerical simulations. Using some recently developed tools[1,2], we have designed and implemented an integrated approach to compute evolving, transient, three dimensional FTLE fields during a CFD simulation. By integrating the computations in this way, tedious post-processing of velocity fields is no longer needed, and larger, more complex problems become accessible to the application of LCS theory.**

## I. INTRODUCTION

That fluid flows are organized by an underlying structure is not a new idea; da Vinci described the similarity of hair-like curls (eddies) generated by a narrow jet issuing into a pool as early as the year 1500, and much of turbulence theory has evolved from notions of organized scales of motion. Modern advances in experimental techniques and the advent of direct numerical simulation (DNS) of the Navier-Stokes equations have provided databases with incredible microscopic detail for a variety of natural and engineered flows. However, a structural description of the coherent motions exhibited by unsteady flows remains elusive in most cases. As available databases continue to grow rapidly in size and complexity, a number of tools have been developed to help extract coherence from chaotic fluid motions. For example, variants of the Eulerian velocity gradient tensor have aided in the detection of instantaneous vortical regions[3], and proper orthogonal decomposition has been helpful in extracting energetic modes of complex turbulent flows[4]. More recently, a class of methods to detect *Lagrangian coherent structures* (LCS) has emerged from dynamical systems theory and the works of Haller[5,6].

LCS are the codimension one manifolds[7] which form the skeleton of tracer trajectories[8] and separate time dependent flows into regions of dynamically distinct behavior[9]. When properly defined, they act as the locally most attracting or repelling material surfaces[10,11] and therefore have important consequences for mixing and transport. In addition, as boundaries of dynamically distinct regions they provide an exceptional tool with which to visualize and understand coherent fluid motion. The identification of LCS in experimentally measured or numerically simulated flows has most often relied on the relation of these special material surfaces to ridges of the finite-time Lyapunov exponent (FTLE) field. Ridges in the forward time FTLE field can be thought of as candidates for *repelling* LCS about which there is large fluid stretching, while ridges in the backward time FTLE field are candidates for *attracting* LCS about which folding occurs[5]. An

[a] Electronic mail: finnj@engr.oregonstate.edu
[b] Electronic mail: sva@engr.oregonstate.edu

impressively broad range of flows have now been studied using the FTLE ridges as a proxy for LCS (see Peacock and Dabiri [12], Samelson [13] for recent reviews). For example, new insight has been provided in areas such as fundamental vortex dynamics[14,15], aerodynamics[16], biological feeding[17], ocean and atmospheric transport[18], and granular flows[19]. Significant work has been undertaken to explore the link between LCS and ridges of the FTLE field from a theoretical standpoint[6,9–11,20,21]. Unfortunately, it appears that there is not necessarily a one to one mapping between FTLE ridges and LCS, and both false LCS positives and negatives can be obtained without sufficient additional restrictions[10,11] on the ridges. Haller and Yuan [5] provided the counter example of two-dimensional shear flow, $u = -x,\ v = y$, where the line $x = 0$ is a positive time FTLE ridge, but is not an unstable material surface. Recently, the problem of defining LCS has been revisited, and a variational theory[10,11] and geodesic theory[21] have been developed which more rigorously distinguish Lagrangian transport barriers. Practical implementation of these more general LCS theories, as done by Farazmand and Haller [22] and Haller and Beron-Vera [21], is presently limited to two dimensional flows, but algorithmically the preliminary steps are similar to a standard FTLE computation (requiring the computation of the Cauchy-Green deformation tensor). In this sense computation of the FTLE, or similar field, seems to be an important step to determining *LCS candidates* in general time dependent flows.

Typically, FTLE fields are computed in a post-processing procedure using either experimentally or numerically obtained velocity fields. First a flow map over the finite-time interval $[t_0,\ t_1]$ is computed by numerical integration of a grid of Lagrangian particles through snapshots of the time dependent velocity field. The flow map Jacobian is then obtained by finite differencing on the initial tracer grid, and used to build the right Cauchy-Green deformation tensor. Finally, the maximum eigenvalue of this tensor is used to compute the maximum FTLE field, which corresponds to the rate of Lagrangian separation of initially adjacent trajectories. As just described, this procedure can be resource intensive for a number of reasons. Most significantly, to obtain sharp, well defined ridges in the FTLE field, the initial grid of tracers must be refined relative to the smallest spatial scales of fluid motion, and the integration time must be long relative to the dominant time scales of the flow. For non-trivial flow fields, especially in three dimensions, this can result in an enormous number of tracer advections. Additionally, for every time the FTLE is needed, for example to visualize its evolution, the tracer advections must be re-computed in forward and backward time. Finally, for acceptable accuracy, the spatial and temporal resolution of the flow field snapshots must be fine enough for accurate interpolation of the velocity field to the Lagrangian tracers. For large datasets, this implies careful memory management during post-processing.

A number of strategies have been employed to address these difficulties. Lipinski and Mohseni [23] noted that the LCS are themselves Lagrangian objects which are advected with the flow and devised an efficient ridge tracking algorithm so that the FTLE field need only be computed in the vicinity of the LCS. A few authors have enlisted unstructured meshes and automated mesh refinement (AMR) schemes to enhance the resolution of FTLE field near the LCS[24,25] while reducing the number of tracer advections far from the LCS. In a similar spirit, Farazmand and Haller [22] used a staggered auxiliary grid of tracers to increase the accuracy of the flow map gradient while maintaining a relatively coarse tracer background grid. Recent developments in computer hardware have also been leveraged to speedup the computation. Due to their streaming capabilities, graphical processing units (GPUs) are naturally suited for the problem of particle advection which is the bottleneck of the FTLE computation and impressive speedups have been obtained relative to standard CPU computations[26,27]. From a theoretical perspective, a recent observation that the largest FTLE in forward time is related to the smallest FTLE in backward time (and vice-versa)[28] could allow for the computation of both attracting and repelling LCS candidates with only a single set of tracer integrations. However, practical implementation issues arise for long integration times and three dimensional flows due to the need to interpolate from a highly deformed grid of particles. Moving forward, it is important to continue to develop efficient computational strategies so that LCS theory can be applied to more complex problems of general interest.

In this paper, we describe an integrated procedure for computing both forward and backward time FTLE fields *on-the-fly* during DNS of the Navier-Stokes equations. Two recent observations have provided the necessary building blocks for our current implementation. The first, by Leung [1], is that the backward time flow map may be treated as a collection of Eulerian scalar fields which can be evolved forward in time by solving three level set equations on a fixed grid. This provides a natural way to compute the backward time flow map during a simulation which evolves only in forward time. The second important observation, made by Brunton and Rowley [2], exploits the property[29] that a time $T$ flow map, where $T = |t_1 - t_0|$, may be constructed from a sequence of $N$ smaller time $h$ flow maps, where $h = T/N$. This removes the need to do redundant tracer integrations when many concurrently evolving FTLE fields must be computed, for example to animate their evolution, meaning the simulation only needs the resources to evolve one forward and one backward flow map at a time.

Integrating the FTLE computations into a CFD simulation has several potential advantages over the post-processing approach. Perhaps the most compelling is that the full temporal and spatial resolution of the simulation become available when computing the flow maps. This is not generally true in the post-processing approach, where the temporal and spatial resolution of the velocity field, and by consequence the flow map uncer-

tainty, could be limited by available hard disk or memory resources. This makes applying LCS theory to study flows with broad length and timescale separations, such as multiphase and turbulent flows, as well as flow in complex geometries, such as packed beds and porous media, much more feasible. Second, by performing the computations during the simulation, we can harness the parallelism of the simulation code as well as tools already built into many flow solvers such as accurate gradient calculations, interpolations, particle tracking schemes, and passive scalar solvers for the advection-diffusion equation. Finally, integration of the FTLE computations directly into a CFD simulation framework makes it possible to extend LCS theory to study a number of new applications. For example, new explorations of active feedback control could be pursued based on characteristics of the LCS, or measurement of Lagrangian length scales could be made using the FTLE field in evolving three dimensional turbulent flows. Some of these benefits have been suggested previously (for example by Leung [1]), but to our knowledge, this work represents the first fully integrated computation.

Our main objective is to describe a relatively simple algorithm which can be used to compute evolving forward and backward time FTLE fields during a computational fluid dynamics (CFD) simulation, with minimal additional overhead. The remainder of the paper is organized as follows. We first give a brief mathematical background for the computation of FTLE fields in Section II. Next, we present numerical details related to the integration of the computations in a direct numerical simulation framework in Section III C. In Section IV, we demonstrate the performance of the implementation and discuss the computational overhead for several two and three dimensional flows. Finally, in Section V we make some general conclusions and proposals for future work.

## II. THE FINITE TIME LYAPUNOV EXPONENT

We now briefly develop the theory and notation related to the FTLE calculation. More thorough expositions describing the FTLE and its relation to LCS are available[9,20]. Assume that some velocity field, $\mathbf{u}(\mathbf{x},t)$, is defined on the three dimensional domain, $\mathbf{x} \subseteq \mathbb{R}^3$, over the time interval, $t \in (t_0, t_1)$. The flow map, $\mathbf{\Phi}_{t_0}^{t_1}(\mathbf{x}_0, t_0)$ integrates passive tracers from their initial position, $\mathbf{x}_0$, at time $t_0$ along pathlines to their "advected" position, $\mathbf{x}$, at time $t_1$,

$$\mathbf{\Phi}_{t_0}^{t_1}(\mathbf{x}_0, t_0) = \mathbf{x}_0 + \int_{t_0}^{t_1} \mathbf{u}(\mathbf{x}(\tau), \tau) d\tau. \tag{1}$$

The flow map may be computed in *forward time* ($t_1 > t_0$), or *backward time* ($t_1 < t_0$). Two tracers initially displaced by a small perturbation, $\delta\mathbf{x}_0$, will find themselves separated by distance $\delta\mathbf{x}$ at time $t_1$. To a leading order, this separation can be written in terms of the flow map as[29]

$$\delta\mathbf{x}(t_1) = \mathbf{\Phi}_{t_0}^{t_1}(\mathbf{x}_0 + \delta\mathbf{x}_0, t_0) - \mathbf{\Phi}_{t_0}^{t_1}(\mathbf{x}_0, t_0) \tag{2}$$

$$= \mathbf{D}\mathbf{\Phi}_{t_0}^{t_1}(\mathbf{x}_0, t_0)\delta\mathbf{x}_0 + \text{H.O.T.}. \tag{3}$$

Where $\mathbf{D}\mathbf{\Phi}_{t_0}^{t_1}(\mathbf{x}_0, t_0)$ is the Jacobian of the flow map evaluated at the initial tracer coordinate, $\mathbf{x}_0$. The magnitude of the separation, $||\delta\mathbf{x}(t_1)||$, is found from the $L^2$ matrix norm,

$$||\delta\mathbf{x}(t_1)|| = \sqrt{\left\langle \delta\mathbf{x}_0, \left[\mathbf{D}\mathbf{\Phi}_{t_0}^{t_1}(\mathbf{x}_0, t_0)\right]^* \left[\mathbf{D}\mathbf{\Phi}_{t_0}^{t_1}(\mathbf{x}_0, t_0)\right] \delta\mathbf{x}_0 \right\rangle}, \tag{4}$$

where, $\langle\cdot,\cdot\rangle$ denotes the Euclidean inner product, and $^*$ denotes transposition. We introduce the right Cauchy-Green deformation tensor,

$$\mathbf{C}_{t_0}^{t_1}(\mathbf{x}_0, t_0) = \left[\mathbf{D}\mathbf{\Phi}_{t_0}^{t_1}(\mathbf{x}_0, t_0)\right]^* \left[\mathbf{D}\mathbf{\Phi}_{t_0}^{t_1}(\mathbf{x}_0, t_0)\right]. \tag{5}$$

From equation 4, the maximum stretching over the interval $(t_0, t_1)$ occurs when $\delta\mathbf{x}_0$ aligns with the eigenvector associated with the maximum eigenvalue, $\lambda_{max}$, of $\mathbf{C}_{t_0}^{t_1}(\mathbf{x}_0, t_0)$. This leads to the definition of the finite-time Lyapunov exponent, which characterizes the maximal rate of stretching over the finite-time interval $(t_0, t_1)$.

$$\sigma_{t_0}^{t_1}(\mathbf{x}_0, t_0) = \frac{1}{|t_1 - t_0|} \log \sqrt{\lambda_{max}(\mathbf{C}_{t_0}^{t_1}(\mathbf{x}_0, t_0))} \tag{6}$$

For simplicity, the standard convention is to notationally drop the dependence of the flow map, deformation tensor, and FTLE fields on $\mathbf{x}_0$ and $t_0$, and to introduce the integration time, $T = |t_1 - t_0|$. This way for example, the forward and backward time $T$ flow maps can be written in shorthand as $\mathbf{\Phi}_{t_0}^{t_0+T}$, and $\mathbf{\Phi}_{t_0}^{t_0-T}$ respectively.

Under sufficient restrictions[10,11], ridges in the FTLE field can mark attracting ($t_1 < t_0$) or repelling ($t_1 > t_0$) LCS. However, as mentioned earlier, it is possible to identify false positive and negative LCS using primitive ridge definitions alone. At the present, we choose to use FTLE ridges as indicators of *LCS candidates*, knowing that more rigorous analysis is needed to truly confirm them as having all the properties of LCS.

## III. NUMERICAL IMPLEMENTATION

In this section, we discuss the tools needed to integrate the FTLE computations within the framework of a CFD solver. We will discuss the details relevant to our present implementation, however, it is intended that the general algorithm which is presented could be followed by anyone wishing to add a similar capability to their own CFD code. The present solver was originally developed to perform large-eddy and direct numerical simulations of flows in complex geometries on potentially very large grids[30]. The code solves the Navier-Stokes equations on arbitrary shaped, unstructured grids using a co-located finite volume discretization, a fractional step method for

time advancement, and an algebraic multigrid (AMG) solver for the pressure Poisson equation[31]. The code is parallelized using Message Passing Interface (MPI), allowing for larger scale simulations by distributing the required memory over many processors. The flow solver has a number of extended capabilities particularly relevant to multiphase flow simulations including Euler-Lagrange models for a subgrid scale dispersed phase[32], and a fictitious domain approach for fully resolved simulation of immersed rigid bodies[33].

### A. Lagrangian and Eulerian flow map computations

In the post-processing approach to computing the FTLE fields, the forward and backward time flow maps are typically composed by seeding the flow with Lagrangian tracers at time $t = t_0$ and advecting them in positive and negative time over the intervals $[t_0 + T]$ or $[t_0 - T]$ respectively. This is the methodology which we employ for the calculation of the forward time flow maps. The three important steps of this process are illustrated schematically in Figure 1a. First, at time $t_0$, we *launch* a Lagrangian tracer from each cell, by setting the initial tracer coordinates, $\mathbf{x}_0 = \mathbf{x}_{cv}$, where the subscript $cv$ denotes the cell center coordinates of each control volume. The tracers are then advected passively with the flow by integrating the equation,

$$\frac{d\mathbf{x}}{dt} = \mathbf{u}(\mathbf{x}, t) \tag{7}$$

To do this, the position of the Lagrangian tracers is updated after each time step of the flow solver using a simple, explicit, trapezoidal approximation of equation 7.

$$\mathbf{x}^{n+1} = \mathbf{x}^n + \mathbf{u}(\mathbf{x}^n, t^{n+1/2})\Delta t \tag{8}$$

Where the time step, $\Delta t$, is the time step used by the flow solver. The fluid velocity used for the tracer position update, $\mathbf{u}^{n+1/2}$, is staggered in time with respect to the fluid velocity known on the fixed grid. Since we perform the tracer update after the fluid solve, this is approximated as simply, $\mathbf{u}^{n+1/2} = 0.5\left(\mathbf{u}^n + \mathbf{u}^{n+1}\right)$. Higher order tracer integrations such as RK4 are possible. However, since our time step is small to assure good accuracy of the flow solver, we have found that time advancement according to equation 8 performs well and is computationally efficient. During the integration, the tracers will cross over cell and processor boundaries. To handle this in our unstructured solver we use a particle in cell approach and known vicinity search algorithms[32] to efficiently locate the tracers as they evolve with the flow field. At time $t_0 + T$, all processors gather the Lagrangian tracers currently located on their own grid partition. Each tracer carries two integer tags corresponding to the processor and $cv$ index that it was launched from. Each processor sorts the tracers currently belonging to it by these tags and communicates their current position, equal to the forward time flow map $\mathbf{\Phi}_{t_0}^{t_0+T}$, back to its launch processor and $cv$.

During the simulation, if a tracer crosses over a solid boundary at any point during the integration, it is relocated to the cell center of the last control volume which contained it. In practice, this is a rare occurrence because the no-slip condition at solid boundaries and low simulation timesteps prevent these types of erroneous tracer trajectories. If a tracer crosses over an outlet boundary, its position is fixed at the boundary for the remainder of the simulation. This condition is somewhat unphysical, but is not an issue if the simulation outlet is placed significantly downstream of the region of FTLE interest.

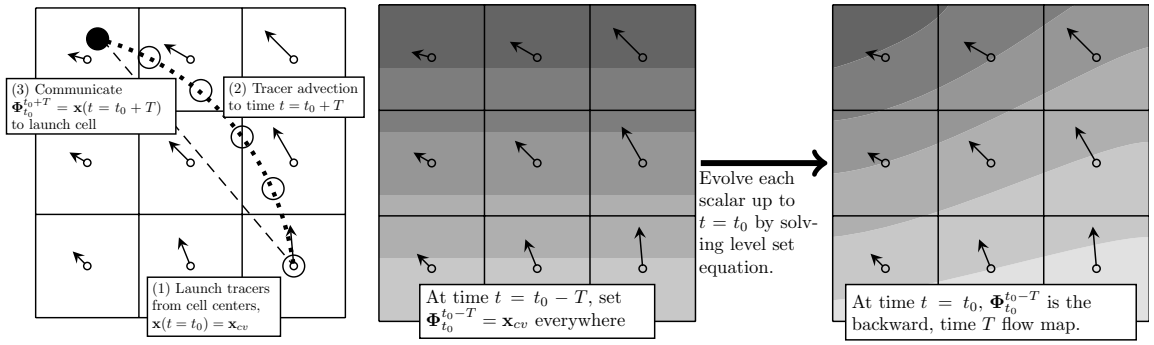
Unfortunately, there is not a natural way to compose the backward time flow map during the CFD simulation with the Lagrangian approach as just described. Doing so would involve saving several snapshots of the velocity field between $t_0$ and $t_0 + T$, either in memory or hard disk, then subsequently using them to integrate tracers backward in time once the simulation reached $t_0 + T$. This is both cumbersome and resource intensive. A clever alternative, proposed by Leung [1], which integrates almost seamlessly with a forward evolving CFD simulation is to treat each component ($x$, $y$, and $z$) of the backward time flow map as a scalar field. These scalar fields represent the *"takeoff coordinates"* of tracers located on the fixed Eulerian grid, and are evolved forward in time by solving three level set equations at each timestep:

$$\frac{\partial \Phi_{t_0,i}^{t_0-T}}{\partial t} + (\mathbf{u} \cdot \nabla)\Phi_{t_0,i}^{t_0-T} = 0 \tag{9}$$

This equation states that the $i^{th}$ component of $\mathbf{\Phi}_{t_0}^{t_0-T}$ remains constant along tracer trajectories, just as it would in the Lagrangian approach if we integrated backward in time. To embed the *takeoff coordinates* at time $t_0 - T$ in the Eulerian representation, we simply initialize the backward flow map at the cell centers to the grid coordinates, $\mathbf{\Phi}_{t_0}^{t_0-T}(\mathbf{x}_0, t_0 - T) = \mathbf{x}_{cv}$. The three scalars are then evolved according to equation 9 up to time $t_0$, at which point their value on the fixed grid is equivalent to the $i^{th}$ component of the backward flow map for integration time $T$. This process is shown for the $y$ component of the backward flow map being evolved by a simple flow field in Figure 1b.

Equation 9 can be solved in a number of ways, and Leung [1] chose a higher order weighted essentially non-oscillatory (WENO) scheme. In our present implementation, we have chosen a semi-Lagrangian approach. Compared to many Eulerian solvers, these schemes are simple, have relatively low overhead, and are easy to implement. They have gained popularity in atmospheric and multiphase flow simulations for their efficiency and stability with large timesteps[34]. The idea of semi-Lagrangian advection is to approximately integrate equation 9 along tracer trajectories:

$$\frac{\Phi_{t_0,i}^{t_0-T}(\mathbf{x}, t^{n+1}) - \Phi_{t_0,i}^{t_0-T}(\mathbf{x} - 2\mathbf{m}, t^n)}{\Delta t} = 0 \tag{10}$$




(a)Forward time (b)Backward time

FIG. 1. Lagrangian and Eulerian computations of the flow maps for time $t = t_0$. (a) The Lagrangian computation of the forward time flow map, $\mathbf{\Phi}_{t_0}^{t_0+T}$. (b) The Eulerian computation of the backward time flow map, $\mathbf{\Phi}_{t_0}^{t_0-T}$. The scalar field shows the evolution of the $y$ component of the backward time flow map in a simple velocity field.

Here, the vector $\mathbf{m}$ is the distance traveled by a tracer in the time $\Delta t/2$, ie. the midpoint of the trajectory between its takeoff point at $t^n$ and the fixed grid point $\mathbf{x}$. In order to solve this equation, we use the second order, two time level scheme described by Staniforth and Côté [35].

1. The vector $\mathbf{m}$ may be approximated to the second order through the implicit relation, $\mathbf{m} = \mathbf{u}(\mathbf{x} - \mathbf{m}, t^{n+1/2})\frac{\Delta t}{2}$, which can be solved iteratively, provided some initial guess, $\mathbf{m}_0$

$$\mathbf{m}_{k+1} = \mathbf{u}(\mathbf{x} - \mathbf{m}_k, t^{n+1/2})\frac{\Delta t}{2}, \tag{11}$$

   where subscript $k$ denotes the iteration index. On the first time step, we initialize $\mathbf{m}_0$ to zero everywhere. On subsequent timesteps, we simply use the $\mathbf{m}$ from the previous time step. Because major temporal changes to the flow field occur on much longer timescales than the simulation time step, we observe that typically very few iterations are required after the initial time step.

2. Evaluate $\Phi_{t_0,i}^{t_0-T}(\mathbf{x} - 2\mathbf{m}, t^n)$. This value is then projected forward in time along the trajectory according to equation 10.

At the boundaries of the simulation domain, we enforce the boundary conditions for $\mathbf{\Phi}_{t_0}^{t_0-T}$ suggested by Leung [1]. At inflow, and no-slip boundaries, we enforce $\mathbf{\Phi}_{t_0}^{t_0-T} = \mathbf{x}|_{boundary}$. At all other boundaries, we enforce $\mathbf{n} \cdot \nabla \Phi_{t_0,i}^{t_0-T} = 0$, where $\mathbf{n}$ is the outward normal vector associated with the boundary.

Our use of a semi-Lagrangian scheme to solve the level set equation is somewhat different from most multiphase flow problems where tracking of an interface is required. In such applications, particle level set methods[36] which employ tracer particles that carry a marker function, can be used to accurately capture the evolution of a deforming interface. As the interface deforms, so does the distribution of its marker particles, and in practice the tracer grid must be *re-meshed* or *reinitialized* to obtain a high quality representation of the evolving interface. Similar issues arise in our problem, where the goal is to track all values of the scalar fields (backward flow map), everywhere on our grid. As the scalar fields evolve from their initial states (eg. Figure 1b), stretching and folding over long times will result in large gradients that could lead to loss of accuracy or instability. One way to handle this would be to employ the forward time Lagrangian tracers in a way similar to the particle level set approach; Every few timesteps, the tracer *takeoff coordinates* could be interpolated fixed grid to correct the backward flow map representation. However, in many of our cases the initial tracers distribution is non-uniform (because it is associated with the fixed grid, see for example section IV B), meaning such an interpolation could be of dubious quality. The approach we pursue instead, is to only compute the flow maps for short integration times over which the scalar field does not distort significantly. The long integration time flow maps are then composed approximately using the method of Brunton and Rowley [2], explained in more detail in section III B.

The discrete equations for evolving both the forward and backward time flow maps involve variables which may be located at non-mesh locations in space. In general, this requires an interpolation of some variable, $\phi$, known at the cell centers of the fixed grid to these points. In an orthogonal, structured computational grid, some form of trilinear interpolation[37,38] can be applied to interpolate the values to the non-mesh points. Since our flow solver is more general and designed to handle complex geometries and unstructured grids, we use an interpolation, based on a two term Taylor series about the cell center which contains the non-mesh point. Let the non-mesh point, $\mathbf{x}_p$ be contained in the control volume with cell center coordinate, $\mathbf{x}_{\text{cv of p}}$, where $\phi|_{\text{cv of p}}$ and $\nabla\phi|_{\text{cv of p}}$ are known. The interpolant, $\mathcal{I}\phi$ provides an estimate for $\phi|_{\mathbf{x}_p}$.

$$\phi|_{\mathbf{x}_p} = \mathcal{I}\phi = \phi|_{\mathbf{x}_{\text{cv of p}}} + \nabla\phi|_{\mathbf{x}_{\text{cv of p}}}(\mathbf{x}_p - \mathbf{x}_{\text{cv of p}}) \tag{12}$$

To compute $\nabla\phi|_{\mathbf{x}_{\text{cv of p}}}$, a least squares gradient estima-

tion for unstructured meshes[39,40] is used which is robust in the presence of skewed mesh elements, and reduces to second order central differencing for regular Cartesian grids.

### B. Efficient composition of flow maps

In practice, the desired temporal resolution of the FTLE fields is typically much finer than the integration time, $T$, over which the flow maps are computed. If this is the case, then several sets of tracer particles and scalar fields need to be evolved concurrently in the simulation in order to compute the exact forward and backward time flow maps. This problem has been addressed by Brunton and Rowley [2], who used the property[29] that a time $T$ flow map can be decomposed into a sequence of $N$ sub-steps of length, $h = T/N$. Adopting their notation, we can write

$$\mathbf{\Phi}_{t_0}^{t_0+T} = \mathbf{\Phi}_{t_0+(N-1)h}^{t_0+Nh} \circ \cdots \circ \mathbf{\Phi}_{t_0+h}^{t_0+2h} \circ \mathbf{\Phi}_{t_0}^{t_0+h} \tag{13}$$

Because the flow maps are obtained discretely, the flow map from one sub-step will not necessarily point to the fixed grid points of the next sub-step, and implementing this reconstruction involves interpolation between sub-steps. Using the interpolation operator, $\mathcal{I}$, we have

$$\mathbf{\Phi}_{t_0}^{t_0+T} = \mathcal{I}\mathbf{\Phi}_{t_0+(N-1)h}^{t_0+Nh} \circ \cdots \circ \mathcal{I}\mathbf{\Phi}_{t_0+h}^{t_0+2h} \circ \mathbf{\Phi}_{t_0}^{t_0+h} \tag{14}$$

If the sub-step, $h$, corresponds to the frequency that FTLE fields are to be computed, then redundant tracer integrations and scalar evolutions can be eliminated. This means only one set of Lagrangian tracers and one set of Eulerian scalars, corresponding to the forward and backward time $h$ flow maps, need to be evolved at any time during the simulation to compute the time $T$ flow maps at all times of interest. The integer $N$ was referred to as the "speedup factor" by Brunton and Rowley [2] because it represented the potential speedup of their post-processing approach relative to computing the "exact" time $T$ flow map every time the FTLE field is needed.

A schematic of this flow map composition scheme, which corresponds to the "single-tiered unidirectional" approach of Brunton and Rowley [2], and the way that it fits into the CFD simulation is shown in Figure 2. The simulation begins at $t = t_0$ and always advances in forward time. To compute the first time $T$ flow maps, $\mathbf{\Phi}_{t_0}^{t_0+T}$ and $\mathbf{\Phi}_{t_0+T}^{t_0}$, exactly we could evolve a single set of Lagrangian tracers and Eulerian scalars over the entire interval $[t_0, t_0+T]$, but this becomes computationally inefficient if we need to compute the FTLE field at intervals smaller than $T$. Instead the total integration time, $T$ is broken up into the $N$ equal sub-steps of length $h = T/N$ shown in Figure 2. As the simulation proceeds, time $h$ flow maps are recorded at these regular intervals and then re-initialized. Rather than storing all time $h$ flow maps in memory, we write them to structured binary files on the hard disk. If sufficient memory is available to the computation, it is possible to avoid this step. However, we have found that performing parallel binary read/write operations accounts for a very small amount of the total simulation time (see section IV F), and hard disk space is more readily available than memory on most computing platforms. Once the simulation reaches time $t_0 + T$, the time $T$ flow maps, $\mathbf{\Phi}_{t_0}^{t_0+T}(t_0)$ and $\mathbf{\Phi}_{t_0+T}^{t_0}(t_0+T)$, are constructed using equation 14 and the time $h$ flow maps which have been stored on disk. Because the flow solver uses unstructured grids and distributed memory, efficient interpolation of one flow map to the next requires careful memory and parallel task management. Details on the parallelization of this step are given in the Appendix. At subsequent simulation times, $t$, which are multiples of the flow map sub-step, $h$, the approximate reconstruction procedure is repeated to construct the forward time flow map, $\mathbf{\Phi}_{t-T}^{t}(t-T)$, and the backward time flow map, $\mathbf{\Phi}_{t}^{t-T}(t)$. For example, when the simulation time reaches $t = t_0+2h+T$, the time $h$ flow maps, $\mathbf{\Phi}_{t_0+h+T}^{t_0+2h+T}(t_0+h+T)$ and $\mathbf{\Phi}_{t_0+2h+T}^{t_0+h+T}(t_0+2h+T)$ are first written to disk. Then, the forward time $T$ flow map, $\mathbf{\Phi}_{t_0+2h}^{t_0+2h+T}(t_0+2h)$, and the backward time $T$ flow map,$\mathbf{\Phi}_{t_0+2h+T}^{t_0+2h}(t_0+2h+T)$, are constructed as shown in Figure 2.

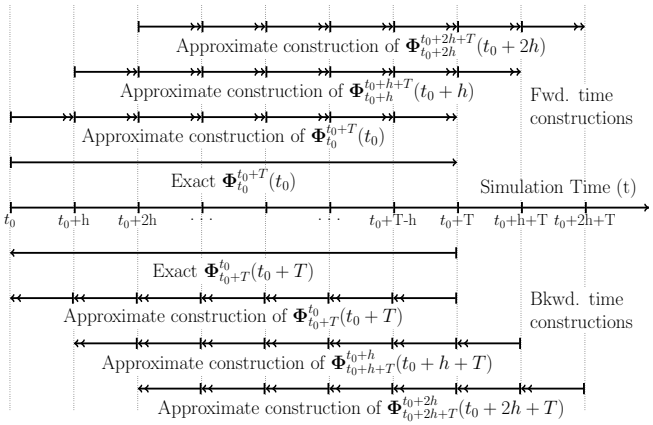



FIG. 2. Integration of the flow map composition scheme with the flow solver forward time advancement. At any simulation time $t$ which is a multiple of $h$, the forward time flow map, $\mathbf{\Phi}_{t-T}^{t}(t-T)$, and the backward time flow map, $\mathbf{\Phi}_{t}^{t-T}(t)$, are constructed from the $N$ time $h$ sub-steps.

### C. Integrated Algorithm

Before discussing how the complete algorithm has been integrated into the flow solver, it is beneficial to review the relevant temporal scales, summarized in table I, and how they are chosen in our integrated approach. In each problem considered in section IV, we can identify a reference time, $t_{ref}$, which characterizes the dominant hydrodynamic timescale of the flow. This could be an eddy turnover time, oscillation period, or any other meaningful reference time. To retain good ac-

curacy of our flow solver, the simulation time, $t$, is advanced using the timestep, $\Delta t$. The latter is set so that the maximum Courant-Friedrich-Lewy number is roughly $CFL = u_{cv}\Delta t/\Delta x = 0.3$, where $u_{cv}$ is the velocity in a control volume with characteristic dimension $\Delta x$. In general, this results in a simulation timestep several orders of magnitude smaller than $t_{ref}$. The FTLE integration time, $T$, should be chosen to be long enough that all LCS candidates are detected. In practice, $T$ is typically chosen to be larger than the longest hydrodynamic time scale of the flow. The final timescale, $h$, is the flow map sub-step. This is chosen to match the desired temporal resolution of FTLE fields. Typically, to create an animation of an FTLE field's evolution, one would desire $h$ to be significantly smaller than $t_{ref}$, and therefore significantly smaller than $T$. Our experience with the several different flows presented in section IV suggests that high quality animations with sharp FTLE features, and good temporal resolution between frames can be obtained with $h$ between $T/20$ and $T/10$.

TABLE I. Timescales involved in the integrated computations

| Timescale | Meaning |
|---|---|
| $t_{ref}$ | Hydrodynamic reference time |
| $t$ | Current simulation time |
| $\Delta t$ | Simulation timestep |
| $T$ | FTLE integration time |
| $h$ | Flow map sub-step |

Figure 3 is a block diagram which shows how the combined simulation and FTLE computation proceeds. After the flow solver advances the velocity field per the usual procedure from $t^n$ to $t^{n+1}$, the intermediate velocity field, $\mathbf{u}^{n+1/2} = 0.5\left(\mathbf{u}^n + \mathbf{u}^{n+1}\right)$ is used to update the forward and backward time $h$ flow maps. The explicit trapezoidal scheme (equation 8) is used to update the Lagrangian tracer positions, and the two time level semi-Lagrangian scheme is used to update the value of the current time $h$ backward flow maps on the fixed grid (according to equation 10). If Lagrangian tracers cross boundaries during the integration step, they are relocated back into the in-bounds cell, except in the case of an outflow, to which they remain fixed for the duration of their integration. The backward time flow map boundary conditions suggested by Leung [1] and described earlier are imposed at the boundary cells of the fixed grid.

At this point, if the current simulation time is not a multiple of the flow map sub-step ( $\mod(t^{n+1}, h) \neq 0$), we return to the main flow solver loop. If $\mod(t^{n+1}, h) = 0$ is satisfied, then the following steps are initiated to compose the time $T$ flow maps and compute the FTLE fields. First, the current position of all Lagrangian tracers is communicated back to the processor and cell which launched them time $h$ ago, giving the time forward time $h$ flow map, $\mathbf{\Phi}_{t-h}^{t}(t-h)$, on the fixed grid. The backward time $h$ flow map, $\mathbf{\Phi}_{t}^{t-h}(t)$, is known

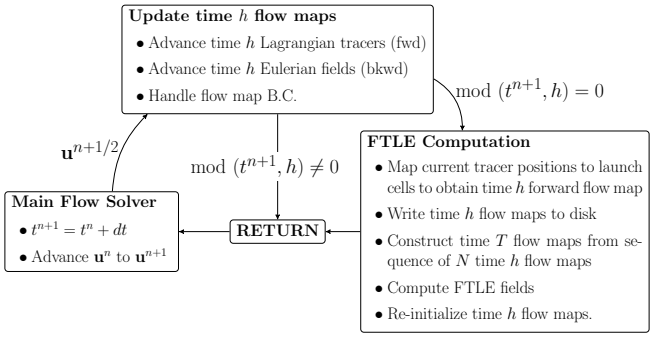



FIG. 3. Block diagram showing the flow of the CFD simulation with integrated FTLE computations.

because it has been evolved up to time $t$ in forward time on the fixed grid. Next, these time $h$ forward and backward flow maps are written to binary files on the hard disk. Then, the time $T$ flow maps are constructed according to equation 14 from the sequence of time $h$ flow maps stored on the hard disk. The Jacobian of each time $T$ flow map is computed at the cell centers of the fixed grid using the least squares gradient operator, and the right Cauchy-Green deformation tensors are computed according to equation 5. Finally, the largest eigenvalue of the forward and backward time tensors are determined, and the corresponding FTLE fields are computed using equation 6. As a last step before returning to the flow solver, the time $h$ flow maps are re-initialized on the fixed grid. This amounts to positioning a Lagrangian tracer at each cell center, and setting value of the backward time flow map to the cell center coordinates.

## IV. NUMERICAL TEST CASES

We now present several examples designed to show the performance and capability of the integrated computations.

### A. Time Dependent Double Gyre

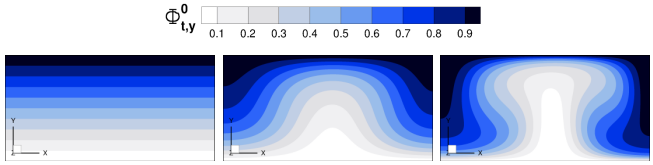
As the first test, we examine the analytic model of time periodic Raleigh-Bénard convection of Solomon and Gollub [41]. This two dimensional stream-function model, sometimes referred to as "double gyre flow," consists of two counter rotating vortices which expand and contract in tandem. The time dependent velocity in the $xy$ plane is described by

$$\begin{aligned} u_x &= -\pi A \sin(\pi f(x,t)) \cos(\pi y) \\ u_y &= \pi A \cos(\pi f(x,t)) \sin(\pi y) \frac{\partial f}{\partial y}. \end{aligned} \tag{15}$$

where,

$$f(x,t) = a(t)x^2 + b(t)x \tag{16}$$
$$a(t) = \epsilon \sin(\omega t) \tag{17}$$
$$b(t) = 1 - 2\epsilon \sin(\omega t) \tag{18}$$

To facilitate comparison with other previous studies of the FTLE field in this system[1,23], we choose parameter values of $A = 0.1$, $\epsilon = 0.1$, $\omega = 2\pi/10$. The domain is discretized with a uniform, $512 \times 256$ Cartesian mesh in the region $[0, 2] \times [0, 1]$. The "simulation" is started at at time $t = 0$, and at advanced to $t = 1.5t_{ref}$, where the reference time is $t_{ref} = 2\pi/\omega$. A constant time step of $\Delta t = 2.5 \times 10^{-4}$ is used. The computational pipeline shown in Figure 3 is followed exactly, with the exception that in the main flow solver we force the velocity field to obey system (15) at every time step.



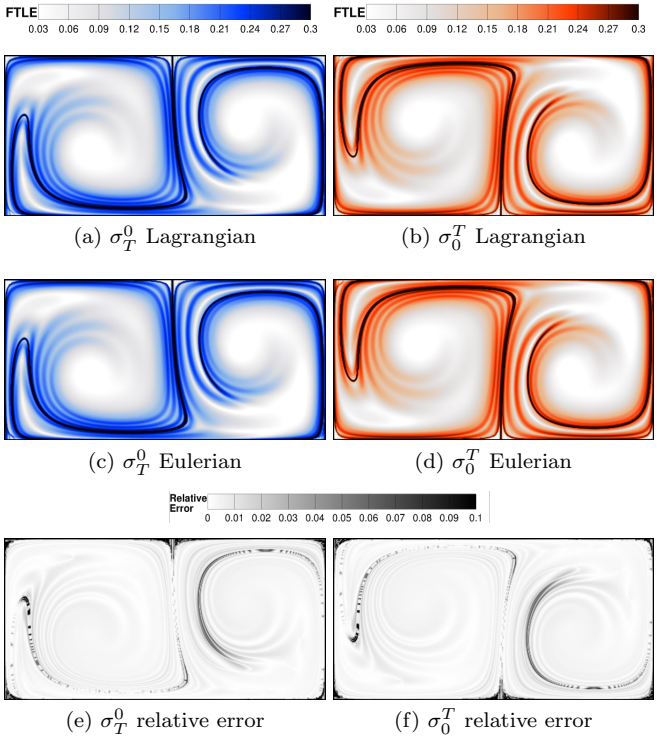

(a) $t/t_{ref} = 0$ (b) $t/t_{ref} = 0.125$ (c) $t/t_{ref} = 0.25$

(d) $t/t_{ref} = 0.5$ (e) $t/t_{ref} = 1.0$ (f) $t/t_{ref} = 1.5$

FIG. 4. Evolution of the Eulerian representation of the backward time flow map on the fixed grid during simulation of the double gyre flow. The scalar field, $\Phi^0_{t,y}$, represents the $y$ coordinate of the Lagrangian particle at time, $t = 0$, which arrives at the fixed grid points at each later time.

To illustrate the Eulerian treatment of the backward time flow map, we plot the evolution of its $y$ component, $\Phi^0_{t,y}$, at several times in Figure 4. The Eulerian field is initialized to $\Phi^0_{t,y} = y_{cv}$ at $t = 0$. These takeoff coordinates are evolved in with flow in forward time using the semi-Lagrangian scheme. The flow maps are reinitialized at regular intervals, $h = 1.25t_{ref}$, and the longer time flow maps are constructed using the composition scheme described in section III B. For short times ($t/t_{ref} \lesssim 0.5$), it is evident that the flow transports the low $\Phi^0_{t,y}$ values in the positive $y$ direction in the center of the cell, and transports the high $\Phi^0_{t,y}$ in the negative $y$ directions along the left and right edges. At later times, the $x$ direction oscillation results in stretching and folding of fluid transported between the two gyres, and a significantly more complex flow map. Despite the Eulerian treatment of $\Phi^0_{t,y}$, its sharp gradients are well preserved by the semi-Lagrangian advection scheme.

We can also use this flow to show that the Lagrangian treatment of the forward time flow map is equivalent to the Eulerian treatment of the backward time flow map.




(a) $\sigma^0_T$ Lagrangian (b) $\sigma^T_0$ Lagrangian

(c) $\sigma^0_T$ Eulerian (d) $\sigma^T_0$ Eulerian

(e) $\sigma^0_T$ relative error (f) $\sigma^T_0$ relative error

FIG. 5. Comparison of the Lagrangian and Eulerian approaches to calculating the FTLE field. (a) through (d) show the FTLE fields computed in (a) backward time with the Lagrangian approach, (b) forward time with the Lagrangian approach, (c) backward time with the Eulerian approach (d) forward time with the Eulerian approach. (e) and (f) show the relative error between the Lagrangian and Eulerian approaches for backward and forward time respectively.

Because the system is symmetric in time about $t = 0$, forward trajectories in system (15) are equivalent to backward trajectories in the negative of system (15). This allows us to make direct comparisons of FTLE fields computed using the Lagrangian and Eulerian representations. To do this, we first apply the flow field of system (15) and simulate the flow for an integration time of $T = 1.5t_{ref}$. We then simulate the negative of system (15) up to the same time. In Figure 5a-d the forward and backward time FTLE fields are shown. It is nearly impossible to distinguish any differences by eye in the FTLE obtained by the Lagrangian and Eulerian representations of the flow map. To obtain a quantitative measure of the difference, we also plot the normalized relative error between the Lagrangian and Eulerian results, Relative Error $= |FTLE^L - FTLE^E|/max(FTLE^L)$ for forward and backward time in Figure 5e-f. Because this metric is very sensitive to slight shifts in ridges of the FTLE field, we are generally satisfied that for most of the domain, the difference between the Lagrangian and Eulerian FTLE field is less than 10% of the max FTLE. Qualitatively, the FTLE fields for the model parameters chosen are in good agreement with prior studies[1,23] with the same parameters. This gives us confidence that our

forward and backward time computations are consistent and accurate.

### B. Flow Over a Fixed Cylinder at Re = 300

We now turn to flow over a fixed cylinder, a typical test case for an unsteady flow solver. A body fitted C-grid containing a total of 1.3 million hexahedral control volumes is used to discretize the computational domain, shown in Figure 6. At the no-slip cylinder boundary, the grid spacing is $D/100$, where $D$ is the cylinder diameter. In the wake region the grid spacing is coarsened to $D/40$. A slip condition is imposed on the top and bottom boundaries in the $y$ direction, and a convective outlet condition ($\partial\mathbf{u}/\partial\mathbf{n} = 0$) is set $25D$ downstream of the cylinder. The grid is two cells thick in the $z$ direction, where periodicity is imposed. At the C boundary $20D$ upstream of the cylinder, a uniform inflow velocity, $U$, is specified so that the Reynolds number, $Re = UD/\nu$, is equal to 300 where $\nu$ is the kinematic viscosity.

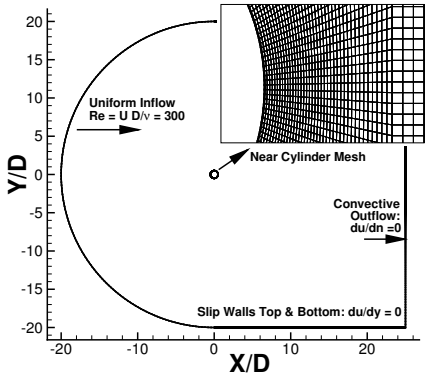
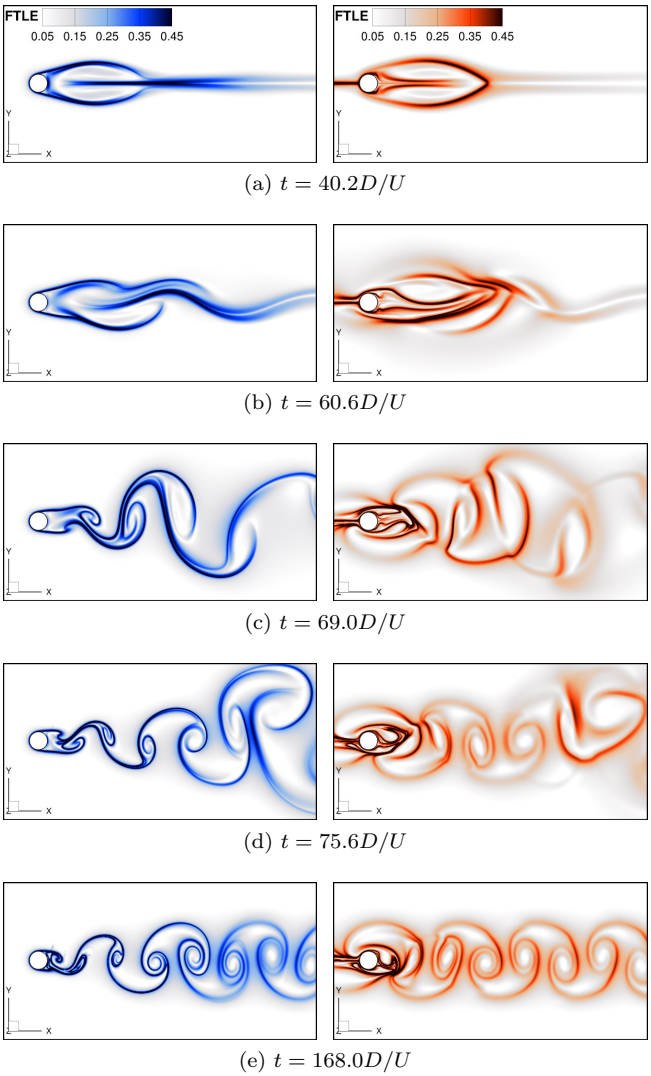



FIG. 6. Domain and mesh used for simulating flow over a fixed cylinder.

The flow is started from rest and simulated for $t = 200t_{ref}$, where $t_{ref} = D/U$, using a constant timestep $\Delta t = 1.7 \times 10^{-3} t_{ref}$. The FTLE fields are computed for an integration time of $T = 7.8t_{ref}$, and with a flow map sub-step, $h = T/13 = 0.6t_{ref}$. When updating the backward time flow map, the boundary conditions discussed earlier are used. That is, on the solid cylinder boundary and at the inflow we impose $\Phi_{t,i}^{t-T} = x_i$ , while at the slip boundaries and outflow, we impose $\mathbf{n} \cdot \nabla \Phi_{t,i}^{t-T} = 0$. The influence of the non-physical fixing of Lagrangian particles which encounter the outflow does not travel upstream far enough to affect the forward time FTLE fields in the near wake. At this Reynolds number, the cylinder wake becomes unsteady, and develops into an unsteady Kármán vortex street. This phenomena is nicely captured by the FTLE fields shown for several instants in time in Figure 7. For $t \lesssim 50D/U$, the boundaries of the symmetric recirculation bubble are clearly marked as LCS candidates by the FTLE ridges. The bubble becomes unstable and the FTLE ridges begin to wander from the line of symmetry at $t = 60D/U$. By $t = 75D/U$, alternating vortex shedding has set in, and the FTLE ridges show the skeleton of the periodic vortex chain in the wake of the cylinder.




(a) $t = 40.2D/U$

(b) $t = 60.6D/U$

(c) $t = 69.0D/U$

(d) $t = 75.6D/U$

(e) $t = 168.0D/U$

FIG. 7. Snapshot of FTLE fields for flow over a cylinder at $Re = 300$ for $T = 7.8D/U$. Left column shows backward time FTLE, right column shows forward time FTLE.

### C. In-Line Oscillation of a Circular Cylinder

In this example, we show that our integrated approach to computing FTLE fields can be easily applied to simulations of flows with *moving boundaries*, which are encountered in multiphase flows, and fluid-structure interaction problems. Our flow solver has the extended capability to simulate resolved rigid particle-flow interactions on non-body conformal Cartesian grids using a fictitious domain approach[33], and we demonstrate the simultaneous FTLE computations here with a simple test case. For

our example, we consider the flow generated by a circular cylinder performing linear oscillations. This problem can be described in terms of the maximum cylinder Reynolds number and the Keulegan-Carpenter number which characterizes the oscillation frequency of the cylinder:

$$Re = \frac{U_m D}{\nu} \tag{19}$$

$$KC = \frac{U_m}{fD} = \frac{2\pi A}{D} \tag{20}$$

where, $U_m$ is the maximum cylinder velocity, $D$ is the cylinder diameter, $\nu$ is the kinematic viscosity of the fluid, $f$ is the frequency of cylinder oscillations, and $A$ is the oscillation amplitude. The cylinder and fluid both start at rest. At $t > 0$, the cylinder position and velocity are described by the sinusoidal functions:

$$x_c(t) = -A\sin(2\pi f t) \tag{21}$$

$$U_c(t) = -2\pi A f \cos(2\pi f t) \tag{22}$$

The cylinder diameter, viscosity, maximum velocity, and oscillation amplitude and are chosen so that $Re = 100$, and $KC = 5$, corresponding with the experimental study of Dütsch *et al.*[42]. A block-type Cartesian grid is used, with a uniform patch in the region of cylinder motion, and periodicity assumed in the spanwise direction. The domain size is $50D \times 50D$ in the $x$ and $y$ directions. Near the cylinder, a uniform grid spacing of $D/100$ is used. The grid uses $750 \times 600 \times 2$ ($9 \times 10^5$ total) cells in the $x$, $y$, and $z$ directions.

The flow is allowed to develop for 9 complete oscillations before computing any FTLE fields. The integration time, $T = 1.5t_{ref}$, where $t_{ref} = 1/f$ is the oscillation period. The FTLE fields are constructed at a flow map sub-step, $h = T/30$. Figure 8 shows the forward and backward FTLE field at four different phase angles, $\theta = 2\pi f t$, of the oscillation. At this Reynolds number and KC number, the flow is characterized by a pair of counter rotating vortices being shed from the top and bottom of the cylinder every half cycle. Upon reversing direction at $\theta = 90^o$ and $180^o$, the cylinder destroys the previously formed pair while creating a new pair in its wake. The attracting LCS candidates compare very well with experimental dye visualizations of the same flow (see FIG 5 of Dütsch *et al.*[42]). For this relatively long integration time the LCS candidates, which can be visually identified as ridges of the FTLE field, show how the cylinder entrains fluid into its wake, then stretches and folds the fluid as it oscillates.

During this simulation, the fluid inside the cylinder, and as a result the flow map, are constrained to the rigid body motion specified by equations 21 and 22 at all times. This results in a sharp FTLE ridge along the solid-fluid interface as would be expected. Despite the immersed representation of the moving rigid body on the fixed grid, the FTLE fields obtained contain sharp well defined features, which provides confidence that the integrated computations are easily extensible to problems with moving boundaries.

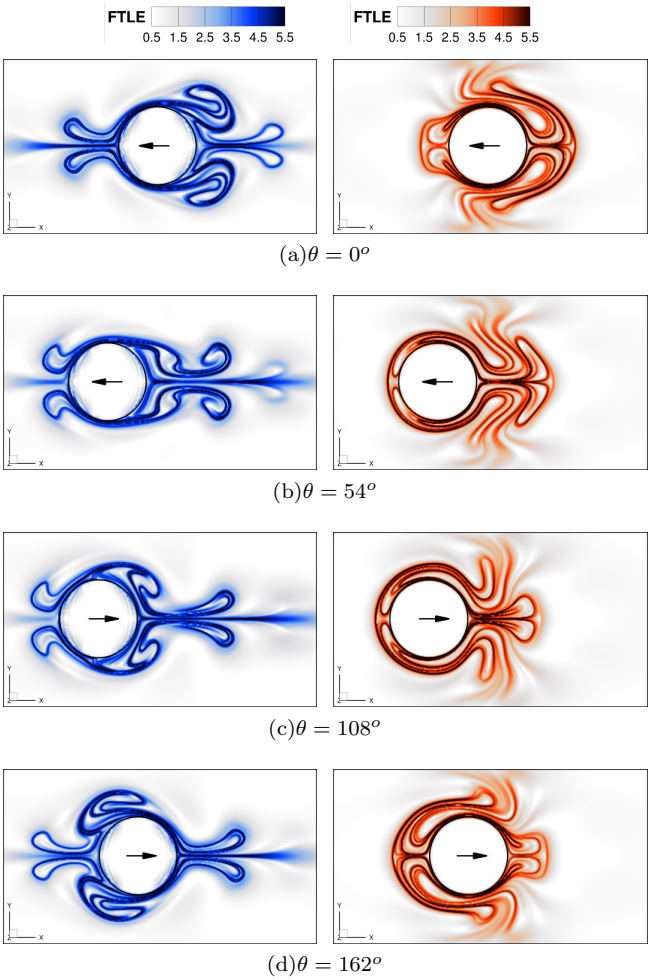
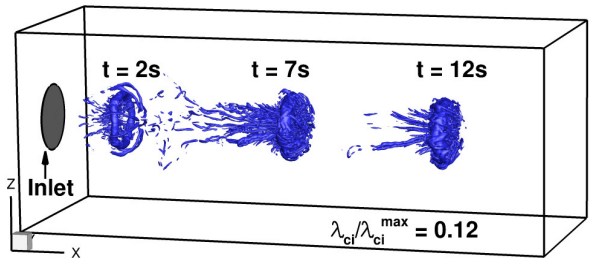



FIG. 8. FTLE fields for selected phase angles in one half cycle of the oscillating cylinder flow. The arrow denotes the direction of cylinder motion. Left column shows backward time FTLE, right column shows forward time FTLE.

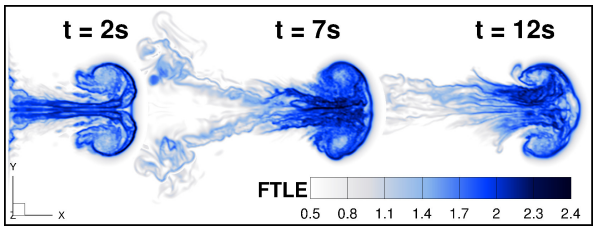
### D. Three Dimensional Turbulent Vortex Ring

To show the capability of the proposed approach for three dimensional flows, we now examine the case of a traveling vortex ring. LCS theory has previously been applied to vortex ring flows as a way to to understand the lobe dynamics of entrainment, detrainment, and transport within the ring[15], as well as the identification of vortex pinch-off during formation of the flow structure[14]. This vortex ring is generated by a quick pulse of fluid from a circular inlet located at the $x = 0$ wall of an initially stagnant rectangular domain. The domain geometry, shown in Figure 9a, is $30cm \times 30cm$ in the $y$ and $z$ directions, and extends $80cm$ in the $x$ direction. The velocity of the inflow jet as a function of time is similar to the studies of bubble vortex interaction by Sridhar and Katz[43] and Finn, Shams, and Apte[44], but is scaled so that the initial circulation is equal to $79.5cm^2/s$. The shear layer of the jet rolls up into a vortex ring and

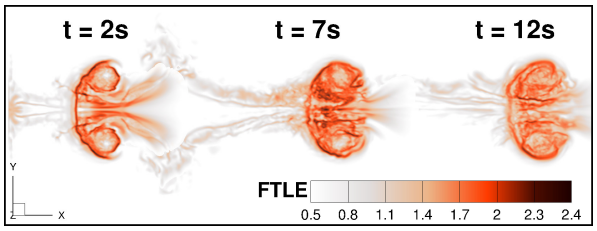
travels downstream at a speed of roughly 14% of the maximum jet velocity. Also in Figure 9a, we visualize the three dimensional ring structure by showing an isosurface of swirling strength, $\lambda_{ci}$. This Eulerian vortex detection criteria corresponds to the imaginary part of the complex eigenvalue of the instantaneous velocity gradient tensor[45]. The isosurface shows the turbulent nature of the ring structure, particularly in its wake as it travels downstream. The domain is meshed with a $640 \times 241 \times 241$ ($37 \times 10^6$ total points) Cartesian grid. The cells are coarsened in the corners of the tank, away from the evolution of the vortex ring.




(a) Eulerian swirling strength




(b) Backward time FLTE




(c) Forward time FTLE

FIG. 9. Snapshots of the three dimensional vortex ring at $t = 2s$ $7s$, $12s$. (a) Isosurface of swirling strength, $\lambda_{ci}/\lambda_{ci}^{max} = 0.12$. (b-c) FTLE fields for the $z = 0$ cross sectional slice through the center of the 3D vortex ring

The forward and backward time FTLE fields are computed during the simulation for $T = 1.5s$, about the time it takes the ring to travel $10cm$ downstream. A flow map sub-step of $h = T/15 = 0.1s$ is used to compose the time $T$ flow maps. Two dimensional slices in the $z = 0$ plane of the three dimensional FTLE fields are shown in Figure 9b and c for $t = 2s$, $t = 7s$, and $t = 12s$. Despite the three dimensionality of the flow, FTLE ridges can be clearly seen in the $xy$ plane, particularly at $t = 2s$ when the candidate LCS surfaces lie perpendicular to the $z = 0$ plane. At later times, as the ring travels downstream and begins to break down, the strong FTLE ridges are less evident in this plane, implying the LCS candidate surfaces are no longer perpendicular to this slice. Three dimensional ridge extraction for cases such as this is an important but difficult task (see Garth *et al.*[26] and Mathur *et al.*[8] for some unique approaches relevant to LCS) and is reserved for a future work.

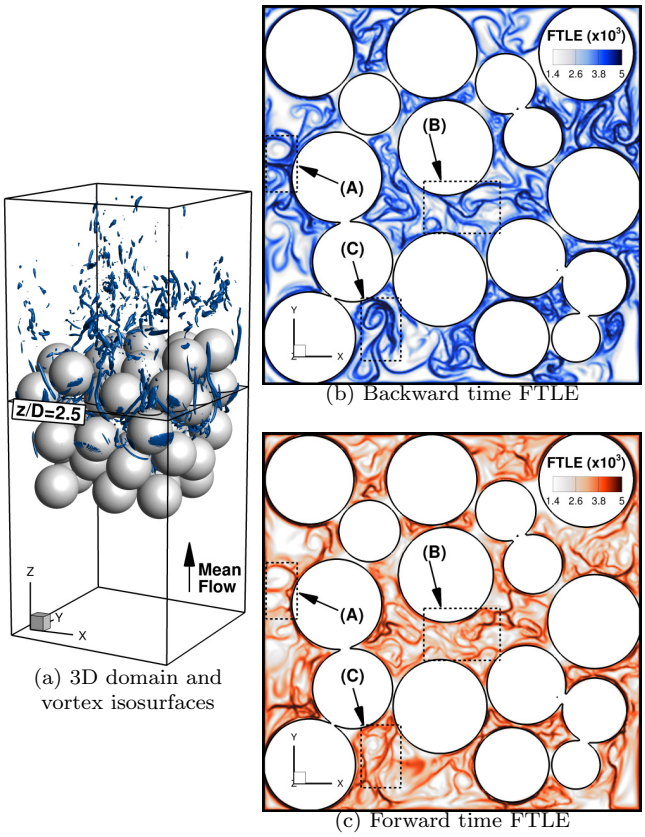
### E. Flow through a random sphere pack

The final case studied, unsteady flow through a random sphere pack, is designed to showcase the benefits of integrating the FTLE computations with the CFD simulation. At moderate Reynolds numbers, flow through packed beds of spheres can develop a variety of coherent porescale flow features including jets and helical vortices, which operate over a wide range of characteristic space and times scales. For packed bed reactors and other complex flow configurations, LCS theory could provide crucial new understanding of how geometric and hydrodynamic features affect transport and mixing, and how slight geometric changes can produce desirable or undesirable to be valuable design tools. However, the geometric scale and complexity of these types of configurations precludes computing the FTLE using the typical post-processing approach.

The geometry which we consider is a channel packed with 51 spheres with constant diameter $D$, shown in Figure 10a. A *ballistic deposition* algorithm, similar to the method employed by Atmakidis and Kenig[46], was used to pack the spheres into a $4D \times 4D \times 4D$ box resulting in a mean void fraction, $\epsilon = 0.58$. The channel was then formed by extending the box boundaries 3D upstream and downstream of the spheres. The flow is driven in the positive $z$ direction by a constant inflow velocity at the upstream boundary so that the pore Reynolds number for the flow is $Re = \frac{UD}{\epsilon\nu} = 600$, where $U$ is the assigned inflow velocity and $\nu$ is the viscosity. A convective outlet condition is located $3D$ downstream of the packing.

The same fictitious domain approach used for the oscillating cylinder case is used to represent both the fixed solid and fluid region on a non body conformal Cartesian grid. This grid has a uniform spacing of $D/80$ everywhere in the porespace, and is stretched toward the inflow and outflow boundaries. In total, it contains $47 \times 10^6$ control volumes. Validation of the fictitious domain approach for flow through packed beds of spheres , including justification for the chosen grid spacing in this case may be found elsewhere[47]. The flow was started from rest and allowed to develop to for $t = 60t_{ref}$ where $t_{ref} = D/U$. The flow becomes unsteady right away, and contains a number of interesting porescale features. The most notable of which is helical vortices, elongated in the mean flow direction. These are shown in Figure 10a by plotting isosurfaces of the Eulerian swirling strength field, $\lambda_{ci}/\lambda_{ci}^{max} = 0.25$.

The FTLE fields are computed for an integration time of $T = t_{ref}$ using a flow map sub-step of $h = T/7$. Because of the complex geometry, the flow is multiscale




(a) 3D domain and vortex isosurfaces

(b) Backward time FTLE

(c) Forward time FTLE

FIG. 10. Flow through the random sphere pack. (a) Shows the simulation domain and instantaneous isosurfaces of $\lambda_{ci}/\lambda_{ci}^{max} = 0.25$, and the $z/D = 2.5$ plane used for visualization of the FTLE fields. (b) and (c) show a snapshot of the FTLE fields for at the $z/D = 2.5$ cross stream slice. The LCS candidates in pores A, B, and C demonstrate distinct, time dependent behavior.

in nature. After some experimentation, these values of $T$ and $h$ were determined to provide sharp FTLE features (long enough $T$) and good temporal resolution of all timescales (small enough $h$). The fields are visualized in Figure 10b and c on the two dimensional $z/D = 2.5$ slice. This slice is oriented perpendicular to the mean flow, at roughly two thirds of the way through the packed bed. The fine grid resolution used for the simulation provides excellent resolution in the FTLE field, and a number of porescale LCS candidates can be easily located with the eye. We will discuss a few of the features in the plane here. (A) is a thin region sandwiched between a sphere and the $x = 0$ wall occupied by a pair of counter-rotating helical (vorticity aligned with the $z$ axis, in and out of the page) vortices. The boundaries of the two vortex cores are well defined by the forward and backward FTLE fields. In time, these features wander slightly along the wall, but remain distinct throughout the simulation. (B) is a large open pore in the center of the packing, where several streams of fluid converge. The LCS candidates, which seem to distinguish the heads of jets as well as the boundaries of vortex cores, are highly transient and move throughout the pore as the simulation proceeds. Pore (C) is a large open region near the wall of the packing which is also highly transient. There is evidence of vortex shedding as the tail of the ridge feature inside the boxed region flaps periodically into the open area of the pore. These results give us confidence that the integrated approach can capture FTLE features in large scale complex configurations.

### F. Computational Expense

By incorporating the FTLE calculation into our flow solver we have removed the need to post-process a sequence of velocity fields. However, in many ways, we have just shifted the computational overhead to the simulation, and it is important to quantify how much this will cost in terms of additional simulation time. If the cost of the FTLE computation were to become significantly large relative to the standard flow solver, the appeal of the integration we have described would diminish. In Figure 11 the expense of FTLE computations is plotted in terms of the percentage of the total simulation time that they consume for each case presented in this section. The expense for each case is further broken into contributions from (i) the Lagrangian update of the forward time flow map tracers, (ii) the semi-Lagrangian update of the backward time flow map scalar fields, (iii) the reconstruction of time $T$ flow maps from the time $h$ sub-steps including reading back the sub-steps from hard disk, and (iv) all other FTLE related computations including writing to the hard disk, composing the strain rate tensor, computing its max eigenvalue and enforcement of flow map boundary conditions. The double gyre case has been omitted, since the velocity field in that case is enforced artificially and the flow solver itself does very little work. To make the comparison as equal as possible for all cases, the time step, $\Delta t$ was chosen so that the maximum CFL number was 0.3, the integration time was set to $T = 2400\Delta t$, and the flow map substep was set to $h = T/12 = 200\Delta t$. All cases were run on the Lonestar super computer at the Texas Advanced Computing Center (TACC), which consists of 1888 nodes of 3.33GHz Intel 2 Hex-core (12 processing cores/node) Xeon 5680® processors connected with a 40Gbit/s Inifiniband Mellanox Switch. Each node of 12 cores shares 24GB of memory. The grids for each case were partitioned with between $38k$ and $56k$ control volumes per partition, resulting in parallel simulations using between 24 and 1200 processors.

It is encouraging to see that the computational time dedicated all FTLE related computations varies between a low of 14% (the vortex ring), and a high of 26.5% (the fixed cylinder) of the total simulation time. Variations between the cases have to do with differences in the load balancing of the parallel partition and the amount of iterations required by the flow solver for convergence, among

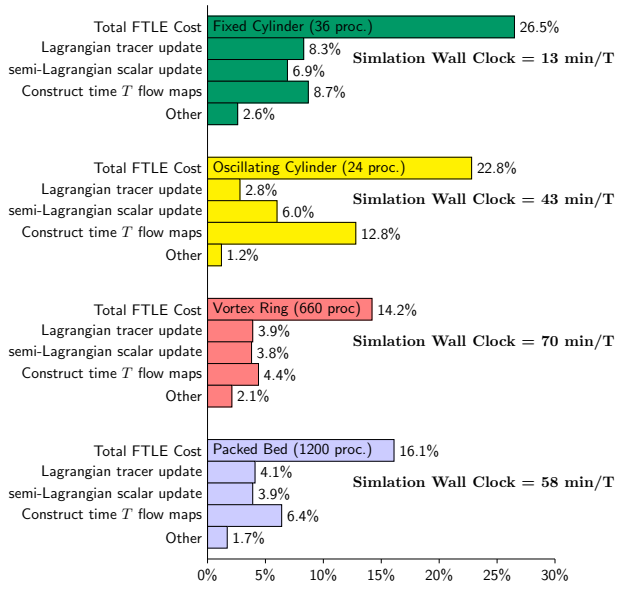



FIG. 11. Computational cost of embedding the FTLE computations into the direct numeric simulation. Bars indicate percentage of the total simulation time dedicated to certain parts of the FTLE computations for each case studied.

other factors. It is evident that the implementation scales effectively to the larger, three dimensional simulations distributed in parallel on over 1,000 processors. It is important to note that the time required to construct the time $T$ flow maps from the time $h$ sub-steps scales almost linearly with the speedup factor, $N$. In this study we have chosen $N = 12$, as we have found it to be sufficient to resolve most FTLE features of interest in these flows.

Memory and hard disk usage is also of concern when performing either distributed memory or shared memory flow simulations. In our current implementation, the simulation memory increase is small because all flow map sub-steps are stored in temporary files on the hard disk. To do this requires

$$N_{cv} \times \frac{T}{h} \times \frac{6 \text{ doubles}}{cv} \times \frac{8 \text{ bytes/double}}{1024^3 \text{ bytes/GB}} \qquad (23)$$

of temporary hard disk space. Thus, a 1 million $cv$ simulation with $N = 12$ will require $0.54GB$. Our largest simulation, the 46 million $cv$ simulation of flow through the packed bed with $T/h = 7$ required $22GB$ of temporary disk space, an amount readily available on most modern computing platforms.

To compare the computational overhead of our integrated approach with the post processing approach, 48 equally spaced velocity fields were output from the fixed cylinder simulation over an interval corresponding to the FTLE integration time, $T = 2400\Delta t$. The velocity fields were then interpolated to a uniform $1750 \times 750$ grid ($1.3 \times 10^6$ total points) covering a subset of the total domain near the cylinder and its wake ($-10 \leq x/D \leq 25$, $-7.5 \leq y/D \leq 7.5$). Using a Lagrangian tracer grid of the same size, a single, backward time FTLE field was computed for integration time $T$ using the freely available *LCS Matlab Toolkit*[17,48]. The computation took 17 minutes using a single core of a 2.83GHz Intel Core 2 Quad® processor, and consumed 0.47GB of memory. In comparison, a total of 24 FTLE fields (12 forward and 12 backward) were computed for the same number of cells during the simulation using the integrated approach. This required 13min of simulation time and 0.7GB of hard disk space to store the intermediate flow maps. The FTLE related computations accounted for 26.5% of the simulation time, meaning each FTLE field took about 9 seconds to compute. This represents a roughly 2 order of magnitude speedup over the post processing approach, and is due to both the parallelism of the integrated approach as well as the efficient flow map composition technique.

## V. CONCLUSIONS

In this paper, we have demonstrated that both the forward and backward time FTLE fields may be computed, on-the-fly, during direct numerical simulation of the Navier-Stokes equations. Our implementation into a parallel, distributed memory, unstructured grid flow solver utilizes Lagrangian particle tracking to compose the forward time flow maps, and an Eulerian treatment of the backward time flow map along with a simple semi-Lagrangian advection scheme. To avoid redundant tracer integrations and scalar advections, time $T$ flow maps are composed from a sequence of time $h$ sub-steps, corresponding to the unidirectional single-tiered method of Brunton and Rowley[2]. This enables us to visualize the evolution of the FTLE fields with high temporal resolution, without burdening the simulation excessively.

The implementation has been tested for several canonical and new flows where LCS theory has not yet been applied. We first showed that the Lagrangian and Eulerian treatments of the forward and backward time flow maps produce high quality and similar results using the analytic double gyre test case. We next showed the applicability to unstructured grids and bluff body flows (the fixed cylinder), rigid body motion and multiphase flow problems (the oscillating cylinder), three dimensional flows (the traveling vortex ring), and large scale simulations with complex boundaries (the packed bed). In all of these cases, the overhead related to the FTLE computations is small relative to the total simulation time. The implementation also scales well for simulations utilizing over 1000 processors.

The integration of these computations directly in the simulation eliminates the need for expensive post-processing of large data sets, but also has a number of additional benefits. Because the velocity field is available at the native space and time resolution of the simulation, integration errors in the flow maps are minimized. Hav-

ing the FTLE fields available to the simulation makes exploring active control of the FTLE fields and their associated LCS much more accessible, and could prove important to a number of applications. Our future work will pursue the possibility of this. We also plan to integrate the extraction of the LCS based on new, more rigorous LCS theory[10,11,21,22] as it continues to evolve. Additional exploration of LCS-like flow features such as inertial LCS which are the attracting and repelling structures for inertial particles[17,49], and burning invariant manifolds (BIMs) which are invariant barriers to reaction fronts[50] are also possible within a similar framework.

## VI. ACKNOWLEDGMENTS

This work was supported by the National Science Foundation (Project #0933857: *Inertial Effects in Flow Through Porous Media*). We gratefully acknowledge the grants for computing time on the Lonestar supercomputer at the Texas Advanced Computing Center (TACC). We thank Andrew Cihonski of Los Alamos National Laboratory for many helpful discussions his help in setting up the vortex ring case. We thank Ivan Christov of Princeton University for a number of thoughtful suggestions.

### Appendix A: Parallelization of the flow map construction scheme

The "single tiered unidirectional" reconstruction procedure proposed by Brunton and Rowley [2] is used to compose the time $T$ flow maps from the sequence $N$ time $h$ flow map sub-steps. Here, we describe the parallelization of this step in our distributed memory, unstructured grid flow solver. Let $t$ be the current simulation time which is a multiple of the flow map sub-step, $h$ ( $\mod(t,h)=0$), and assume that we are interested in constructing the time $T$ forward flow map, $\mathbf{\Phi}_{t-T}^{t}$ from the sub-steps which have already been written to the hard disk. Let $t_C$ be the current *construction time*. We initialize $t_C = t - T$, so that our construction begins at the beginning of our forward time interval, $[t-T,\ t]$, and initialize a temporary construction of the flow map, $\mathbf{\Phi}_{t-T}^{t_C} = \mathbf{x}_{cv}$ everywhere. We then perform the following steps in a loop from $n=1$ to $n=N$:

- Each processor reads its own time $h$ flow map, $\mathbf{\Phi}_{t_C}^{t_C+h} = \mathbf{\Phi}_{t-T+(n-1)h}^{t-T+nh}$ into memory from the binary file located on the hard disk.
- The goal is to update the flow map construction, $\mathbf{\Phi}_{t-T}^{t_C}$, to account for the current sub-step. This involves interpolating the values of $\mathbf{\Phi}_{t_C}^{t_C+h}$ which is known at the grid points, to the non-mesh points of the current flow map construction, $\mathbf{\Phi}_{t-T}^{t_C}$

$$\mathbf{\Phi}_{t-T}^{t_C+h} = \mathcal{I}\mathbf{\Phi}_{t_C}^{t_C+h} \circ \mathbf{\Phi}_{t-T}^{t_C} \tag{A1}$$

  It is likely that the non-mesh points of the current construction will lie across processor boundaries, meaning a coordinated search and retrieve operation is required by all processors to perform this update. To do this, each processor **sends** a query to all processors which could potentially contain each non-mesh point, $\mathbf{\Phi}_{t-T}^{t_C}$. This is determined simply by comparing each non-mesh point to the minimum bounding box of the grid partition belonging to all other processors.

- Each processor **receives** a list of non-mesh points, which may lie within its grid partition, to search for. These are the points where the value of the current flow map sub-step is needed by another processor. A bounding box Oct-Tree search algorithm is used to efficiently determine the cell (if any) which contains each of these points. If found, the value of $\mathbf{\Phi}_{t_C}^{t_C+h}$ is interpolated to the non-mesh point using the interpolation kernel described in section III A. These values of are sorted and **sent** back to the processor which requested each search.

- Each processor **receives** values of the updated flow map $\mathbf{\Phi}_{t-T}^{t_C+h}$ from potentially all other processors.

- The construction time is advanced to $t_C = t_C + h$.

At the end of this loop, $\mathbf{\Phi}_{t-T}^{t_C}$ is an approximation of time $T$ flow map $\mathbf{\Phi}_{t-T}^{t}$. The procedure can be repeated to compute the backward time flow map, $\mathbf{\Phi}_{t}^{t-T}$, by starting at $t_C = t$ and advancing the construction time in the negative time direction.

As the number of sub-steps, $N$, increases this portion of the algorithm can become computationally expensive due to the relatively naive search procedure for non-mesh points in our unstructured grids. In the case of a shared memory structured grid flow solver, these steps could be simplified significantly. Nonetheless, we find that the total expense of this step remains small (see section IV F) relative to the simulation time for moderate values of $N$.

[1] S. Leung, "An Eulerian approach for computing the finite time Lyapunov exponent," Journal of Computational Physics **230**, 3500–3524 (2011).
[2] S. Brunton and C. Rowley, "Fast computation of finite-time Lyapunov exponent fields for unsteady flows," Chaos **20**, 017503 (2010).
[3] P. Chakraborty, S. Balachandar, and R. Adrian, "On the relationships between local vortex identification schemes," Journal of Fluid Mechanics **535**, 189–214 (2005).
[4] G. Berkooz, P. Holmes, and J. Lumley, "The proper orthogonal decomposition in the analysis of turbulent flows," Annual Review of Fluid Mechanics **25**, 539–575 (1993).
[5] G. Haller and G. Yuan, "Lagrangian coherent structures and mixing in two-dimensional turbulence," Physica D: Nonlinear Phenomena **147**, 352–370 (2000).
[6] G. Haller, "Distinguished material surfaces and coherent structures in three-dimensional fluid flows," Physica D: Nonlinear Phenomena **149**, 248–277 (2001).
[7] This implies that LCS are curves in two dimensional flows and surfaces in three dimensional flows..

[8]M. Mathur, G. Haller, T. Peacock, J. Ruppert-Felsot, and H. Swinney, "Uncovering the Lagrangian skeleton of turbulence," Physical Review Letters **98**, 144502 (2007).
[9]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**, 271–304 (2005).
[10]G. Haller, "A variational theory of hyperbolic Lagrangian coherent structures," Physica. D **240**, 574–598 (2011).
[11]M. Farazmand and G. Haller, "Erratum and addendum to "A variational theory of hyperbolic Lagrangian coherent structures" [Physica D 240 (2011) 574–598]," Physica D: Nonlinear Phenomena (2011).
[12]T. Peacock and J. Dabiri, "Introduction to focus issue: Lagrangian coherent structures," Chaos **20**, 017501 (2010).
[13]R. Samelson, "Lagrangian motion, coherent structures, and lines of persistent material strain," Annual Review of Marine Science **5** (2012).
[14]C. O'Farrell and J. Dabiri, "A Lagrangian approach to identifying vortex pinch-off," Chaos **20**, 017513–017513 (2010).
[15]S. Shadden, J. Dabiri, and J. Marsden, "Lagrangian analysis of fluid transport in empirical vortex ring flows," Physics of Fluids **18**, 047105 (2006).
[16]M. Green, C. Rowley, and A. Smits, "The unsteady three-dimensional wake produced by a trapezoidal pitching panel," Journal of Fluid Mechanics **685**, 117–145 (2011).
[17]J. Peng and J. Dabiri, "Transport of inertial particles by Lagrangian coherent structures: application to predator-prey interaction in jellyfish feeding," Journal of Fluid Mechanics **623**, 75–84 (2009).
[18]F. Beron-Vera, M. Olascoaga, and G. Goni, "Oceanic mesoscale eddies as revealed by Lagrangian coherent structures," Geophysical Research Letters **35**, L12603 (2008).
[19]I. Christov, J. Ottino, and R. Lueptow, "From streamline jumping to strange eigenmodes: Bridging the Lagrangian and Eulerian pictures of the kinematics of mixing in granular flows," Physics of Fluids **23**, 103302 (2011).
[20]F. Lekien, S. Shadden, and J. Marsden, "Lagrangian coherent structures in n-dimensional systems," Journal of Mathematical Physics **48**, 065404 (2007).
[21]G. Haller and F. Beron-Vera, "Geodesic theory of transport barriers in two-dimensional flows," Physica D Nonlinear Phenomena **241**, 1680–1702 (2012).
[22]M. Farazmand and G. Haller, "Computing Lagrangian coherent structures from their variational theory," Chaos: An Interdisciplinary Journal of Nonlinear Science **22**, 013128–013128 (2012).
[23]D. Lipinski and K. Mohseni, "A ridge tracking algorithm and error estimate for efficient computation of Lagrangian coherent structures," Chaos **20**, 017504–1 (2010).
[24]F. Lekien and S. Ross, "The computation of finite-time Lyapunov exponents on unstructured meshes and for non-Euclidean manifolds," Chaos **20**, 017505–1 (2010).
[25]P. Miron, J. Vétel, A. Garon, M. Delfour, and M. Hassan, "Anisotropic mesh adaptation on Lagrangian coherent structures," Journal of Computational Physics (2012).
[26]C. Garth, F. Gerhardt, X. Tricoche, and H. Hagen, "Efficient computation and visualization of coherent structures in fluid flow applications," Visualization and Computer Graphics, IEEE Transactions on **13**, 1464–1471 (2007).
[27]C. Conti, D. Rossinelli, and P. Koumoutsakos, "GPU and APU computations of finite time Lyapunov exponent fields," Journal of Computational Physics **231**, 2229–2244 (2012).
[28]G. Haller and T. Sapsis, "Lagrangian coherent structures and the smallest finite-time Lyapunov exponent," Chaos **21**, 023115 (2011).
[29]J. Ottino, *The kinematics of mixing: stretching, chaos, and transport* (Cambridge University Press, 1989).
[30]P. Moin and S. Apte, "Large-eddy simulation of realistic gas turbine-combustors," AIAA Journal **44**, 698–708 (2006).
[31]R. Falgout and U. Yang, "hypre: A library of high performance preconditioners," Computational Science ICCS 2002 , 632–641 (2002).
[32]E. Shams, J. Finn, and S. Apte, "A numerical scheme for Euler-Lagrange simulation of bubbly flows in complex systems," International Journal for Numerical Methods in Fluids **67**, 1865–1898 (2010).
[33]S. Apte, M. Martin, and N. Patankar, "A numerical method for fully resolved simulation (FRS) of rigid particle-flow interactions in complex flows," Journal of Computational Physics **228**, 2712–2738 (2009).
[34]J. Strain, "Semi-Lagrangian methods for level set equations," Journal of Computational Physics **151**, 498–533 (1999).
[35]A. Staniforth and J. Côté, "Semi-Lagrangian integration schemes for atmospheric models- a review," Monthly Weather Review **119**, 2206–2223 (1991).
[36]S. Hieber and P. Koumoutsakos, "A Lagrangian particle level set method," Journal of Computational Physics **210**, 342–367 (2005).
[37]F. Lekien and J. Marsden, "Tricubic interpolation in three dimensions," International Journal for Numerical Methods in Engineering **63**, 455–471 (2005).
[38]R. McDermott and S. Pope, "The parabolic edge reconstruction method (PERM) for Lagrangian particle advection," Journal of Computational Physics **227**, 5447–5491 (2008).
[39]K. Mahesh, G. Constantinescu, and P. Moin, "A numerical method for large-eddy simulation in complex geometries," Journal of Computational Physics **197**, 215–240 (2004).
[40]N. Park and K. Mahesh, "Numerical and modeling issues in LES of compressible turbulence on unstructured grids," AIAA Paper **722**, 85 (2007).
[41]T. Solomon and J. Gollub, "Chaotic particle transport in time-dependent Rayleigh-Bénard convection," Physical Review A **38**, 6280 (1988).
[42]H. Dütsch, F. Durst, S. Becker, and H. Lienhart, "Low-Reynolds-number flow around an oscillating circular cylinder at low Keulegan–Carpenter numbers," Journal of Fluid Mechanics **360**, 249–271 (1998).
[43]G. Sridhar and J. Katz, "Effect of entrained bubbles on the structure of vortex rings," Journal of Fluid Mechanics **397**, 171–202 (1999).
[44]J. Finn, E. Shams, and S. Apte, "Modeling and simulation of multiple bubble entrainment and interactions with two dimensional vortical flows," Physics of Fluids **23**, 3301 (2011).
[45]J. Zhou, R. Adrian, S. Balachandar, and T. Kendall, "Mechanisms for generating coherent packets of hairpin vortices in channel flow," Journal of Fluid Mechanics **387**, 353–396 (1999).
[46]T. Atmakidis and E. Kenig, "CFD-based analysis of the wall effect on the pressure drop in packed beds with moderate tube/particle diameter ratios in the laminar flow regime," Chemical Engineering Journal **155**, 404–410 (2009).
[47]J. Finn and S. Apte, "Relative performance of body fitted and fictitious domain simulations of flow through packed beds," Submitted (2012).
[48]LCS MATLAB Kit Version 2.3 available from www.dabiri.caltech.edu/software.html.
[49]T. Sapsis, G. Haller, *et al.*, "Clustering criterion for inertial particles in two-dimensional time-periodic and three-dimensional steady flows," Chaos **20**, 7515 (2010).
[50]J. Mahoney, D. Bargteil, M. Kingsbury, K. Mitchell, and T. Solomon, "Invariant barriers to reactive front propagation in fluid flows," EPL (Europhysics Letters) **98**, 44005 (2012).