Prediction, Analysis, and Learning of Advective Transport in Dynamic Fluid Flows

by

Chinmay Sameer Kulkarni

B.Tech, Indian Institute of Technology Bombay (2015)
S.M., Massachusetts Institute of Technology (2017)

Submitted to the Department of Mechanical Engineering and
Center for Computational Science and Engineering
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Mechanical Engineering and Computation

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2021

Author .....
Department of Mechanical Engineering and
Center for Computational Science and Engineering
October 1, 2020

Certified by .....
Pierre F.J. Lermusiaux
Professor, Department of Mechanical Engineering
Thesis Supervisor

Accepted by .....
Youssef Marzouk
Co–Director, Computational Science and Engineering

Accepted by .....
Nicolas Hadjiconstantinou
Chairman, Department Committee on Graduate Theses

Prediction, Analysis, and Learning of Advective Transport in Dynamic Fluid Flows

by

Chinmay Sameer Kulkarni

Submitted to the Department of Mechanical Engineering and
Center for Computational Science and Engineering
on October 1, 2020, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Mechanical Engineering and Computation

Abstract

Transport of any material quantity due to background fields, *i.e.* advective transport, in fluid dynamical systems has been a widely studied problem. It is of crucial importance in classical fluid mechanics, geophysical flows, micro and nanofluidics, and biological flows. Even though mathematical models that thoroughly describe such transport exist, the inherent nonlinearities and the high dimensionality of complex fluid systems make it very challenging to develop the capabilities to accurately compute and characterize advective material transport. We systematically study the problems of predicting, uncovering, and learning the principal features of advective material transport in this work. The specific objectives of this thesis are to: (i) develop and apply new numerical methodologies to compute the solutions of advective transport equations with minimal errors and theoretical guarantees, (ii) propose and theoretically investigate novel criteria to detect sets of fluid parcels that remain the most coherent / incoherent throughout an extended time interval to quantify fluid mixing, and (iii) extend and develop new machine learning methods to infer and predict the transport features, given snapshot data about passive and active material transport.

The first part of this work deals with the development of the PDE-based 'method of flow map composition', which is a novel methodology to compute the solutions of the partial differential equation describing classical advective and advective–diffusive–reactive transport. The method of composition yields solutions almost devoid of numerical errors, and is readily parallelizable. It can compute more accurate solutions in less time than traditional numerical methods. We also complete a comprehensive theoretical analysis and analytically obtain the value of the numerical timestep that minimizes the net error. The method of flow map composition is extensively benchmarked and its applications are demonstrated in several analytical flow fields and realistic data-assimilative ocean plume simulations.

We then utilize the method of flow map composition to analyze Lagrangian material coherence in dynamic open domains. We develop new theory and schemes to

efficiently predict the sets of fluid parcels that either remain the most or the least coherent over an extended amount of time. We also prove that these material sets are the ones to maximally resist advective stretching and diffusive transport. Thus, they are of significant importance in understanding the dynamics of fluid mixing and form the skeleton of material transport in unsteady fluid systems. The developed theory and numerical methods are utilized to analyze Lagrangian coherence in analytical and realistic scenarios. We emphasize realistic marine flows with multiple time-dependent inlets and outlets, and demonstrate applications in diverse dynamical regimes and several open ocean regions.

The final part of this work investigates the machine inference and prediction of the principal transport features from snapshot data about the transport of some material quantity. Our goals include machine learning the underlying advective transport features, coherent / incoherent sets, and attracting and repelling manifolds, given the snapshots of advective and advective–diffusive material fields. We also infer and predict high resolution transport features by optimally combining coarse resolution snapshot data with localized high resolution trajectory data. To achieve these goals, we use and extend recurrent neural networks, including a combination of long short-term memory networks with hypernetworks. We develop methods that leverage our knowledge of the physical system in the design and architecture of the neural network and enforce the known constraints that the results must satisfy (e.g. mass conservation) in the training loss function. This allows us to train the networks only with partial supervision, without samples of the expected output fields, and still infer and predict physically consistent quantities. The developed theory, methods, and computational software are analyzed, validated, and applied to a variety of analytical and realistic fluid flows, including high-resolution ocean transports in the Western Mediterranean Sea.

Thesis Supervisor: Pierre F.J. Lermusiaux
Title: Professor, Department of Mechanical Engineering

Acknowledgments

My research over the past five years would not have been possible without the contributions from a number of people. First and foremost, I would like to express my gratitude towards my advisor, Prof. Pierre Lermusiaux for his guidance and invaluable support during the course of this thesis. Pierre encouraged me to work on the problems that I found interesting and fun to tackle. This intellectual freedom has allowed me to grow as a researcher and as a person. His extraordinary work rate and scrupulous attention to detail has never ceased to amaze me. I have learned a lot from him about the academic areas such as computational science, stochastics, dynamical systems. However, more importantly, he has taught me a lot about understanding the relevance and the broader implications of the research and approaching open scientific problems with a principled and methodical manner. I am sure that these skills will be useful in any discipline. Pierre's sense of humor and quirky jokes have lightened the mood during many long research and group meetings.

I would like to thank the members of my thesis committee - Prof. Thomas Peacock and Prof. Stefanie Jegelka for their helpful suggestions during the committee meetings. Many of the real-time work in this thesis has been in collaboration with Prof. Peacock. His insights about the ocean physics and his knowledge about the deep sea mining processes have been extremely beneficial to this work. Prof. Jegelka's insightful queries and comments especially about the machine learning work have been instrumental in the development and the analysis of the proposed ML algorithms.

I thank Dr. Pat Haley for all his suggestions, guidance, and for patiently answering all my questions over the years. The realistic ocean examples would have been impossible without Pat. His amazing ability of writing robust and bug-free code is something I wish to learn one day. Thanks to Pat's calming smile, the most intense of situations became manageable. Dr. Chris Mirabito has been instrumental in the real-time sea exercises and I have learned a lot from him about handling real data and also about web-related work.

Navigating the administrative life at MIT is impossible without the staff and

administrators who are always willing to help. I thank Leslie Regan and Una Sheehan at the MechE graduate office, and Kate Nelson CCSE office for all their help and support. I thank Marcia Munger and Lisa Mayer for being a calming influence and for taking care of all my administrative issues with remarkable efficiency.

We are grateful to the Office of Naval Research (ONR) for research support under grants N00014-14-1-0476 (Science of Autonomy-LEARNS), N00014-15-1-2616 (DRI-NASCar), N00014-20-1-2023 (MURI ML-SCOPE), N00014-14-1-0725 (Bays-DA), N00014-18-1-2781 (DRI-CALYPSO), N00014-15-1-2626 (DRI-FLEAT), and N00014-19-1-2693 (IN-BDA), to the National Oceanographic Partnership Program (NOPP) for research support under grant N00014-15-1-2597 (Seamless Multiscale Forecasting), and to the National Science Foundation (NSF) for support under grant EAR-1520825 (Hazards SEES âĂŞ ALPHA), each to the Massachusetts Institute of Technology. We also thank the MIT Environmental Solutions Initiative (MIT-ESI) for Seed Grant research support.

Thank you to the MSEAS group for being a family away from home! I am thankful to Tapovan, John, Jing, Deepak, and Sydney for initially helping me settle into the group. Deepak has been a great academic and personal mentor over the years. Abhinav and Manan have been great friends and labmates and I feel fortunate to have had the ability to collaboratively work with them. Abhinav's willingness to always help out cannot be understated. Manan has been instrumental in furthering the path planning work. I will always cherish our discussions about elegant mathematics, puzzles, cricket, and Formula 1! Wael, Corbin, and Jing have been great labmates over the years. The various seminars, birthday parties, and more recently the work on SeaVizKit have all been memories that will stay with me. Many thanks to Manmeet, Akis, Aaron, Mike, and Jacob for the great camaraderie in the lab. Starting my MSEAS tenure alongside Johnathan, Florian, Corbin, and Arko was a great blessing. JVo has been one of my closest friends over the years (and a birthday buddy!). Our trip to India along with Abhinav was very memorable! Saviz and Alexis have almost been our labmates over the years, and I thank them for the many bike rides and the great discussions.

My friends outside the lab have made my MIT life really enjoyable. Shraddha and Ben have been awesome housemates. I will never forget our impromptu dessert / dumpling outings and the long hours spent playing different board games. These two along with Nidhi, Sachin, and Prashanth have been great friends and I will cherish all the fun times we have had. Thanks to Andrew for sharing my enthusiasm to try out new food and for sharing my passion about aviation. I feel lucky to have a close friend in my cousin Nupur. The night rides and the rides to RSC and beyond are some of the most fun memories I have. Yamini has been a great friend since undergrad and it has been amazing to have her around in Cambridge. I will always think fondly of our many long and intense discussions and the never-ending search for good Biryani. I am extremely grateful to my friends from undergrad - Bakshi, Kelkar, Prateek, Jayesh, and Rik for being awesome friends and a source of emotional and mental support during turbulent times.

Finally, I would like to thank all my family for their love, affection and constant support over the years. My family in the US has been a true home away from home, and I can always count on their help for any difficulty. I am grateful to my sister Renu for looking out for me, and being the best sister anyone could have asked for. I thank my girlfriend Jess for her love and kindness, and for always being there – both in times of hardship and happiness. Above all, I am thankful to my parents for everything in life. I cannot even begin to imagine where I would be without their support, guidance, love, and friendship at every step.

THIS PAGE INTENTIONALLY LEFT BLANK

Contents

List of Figures

List of Tables

Chapter 1

Introduction and Motivation

1.1 Introduction and Summary of Contributions

The transport of fluid parcels (and any material that they carry) due to the background flow is called 'advective transport' [15]. Advective fluid transport plays a major role in many disciplines of science and engineering. Often times, it occurs as a part of a larger dynamical context involving advection, diffusion, and/or reaction processes [139]. It is essential in classical fluid mechanics [20], microfluidics [236], and biological flows [219]. However, one of the most important fields where advective fluid and material transport plays a key role is atmospheric and ocean sciences [202].

The transport of anthropogenic and natural material in environmental flows is ubiquitous and profoundly impacts society. Preparedness and effective response can save many lives, untold environmental damage, and enormous financial cost. For example, the potential proliferation of large-scale ocean exploitation such as deep sea mining activities creates major risks to the surrounding marine environment as the generated plumes are advected and dispersed due to the background ocean currents [181]. The ever-increasing size of the great Pacific garbage patch and of the pollution from macro- and micro-plastics and other ocean debris are of great concern due to the harm they cause to the local, regional, and global ecosystems [142]. At the same time, several natural systems rely heavily on flow transport in order to survive and thrive. For example, marine biological ecosystems require advection of nutrients for

plankton and fish growth [252]. Predicting and quantifying the overall impact and risks of such natural events and harmful human activities is essential in deciding the best course of action.

Understanding the dynamics of flow transport and predicting where things go in unsteady and complex environmental flows remains a formidable scientific challenge [101]. Some of the reasons that make this process demanding are the following: (i) unsteady transport processes require precise analysis so that material transport is accurately defined and predicted, (ii) accurate high-resolution numerical modeling of the flow field and advective transport is highly challenging due to nonlinearity, uncertainty, and compounding numerical errors, (iii) data is only partially and sparsely available in the form of snapshot observations of material transport or trajectories of marine floats and drifters, with no access to the actual underlying flow fields.

Addressing several of the above challenges, this thesis develops fundamental theories and rigorous methodologies for predicting, quantifying, and learning key material transport processes and structures in dynamic fluid flows, often observed in marine and geophysical sciences. The research is specifically driven by the following questions: (i) can we develop a highly accurate and efficient numerical method to solve the advective transport equation which ensures that numerical errors are not compounded?, (ii) can we efficiently quantify the sets of passively advected material that remain the most / least coherent over a considered time interval and so characterize fluid mixing and diffusion?, and (iii) can we utilize the recent advances in scientific machine learning to infer and predict the underlying material transport features from snapshot data about the transport of a material quantity?

This thesis may be broadly categorized into three topics: (i) development of a novel highly accurate numerical method to solve the advective transport equation, (ii) Lagrangian coherence analysis of passive advective material transport, and (iii) inference and prediction of transport features and coherent structures from snapshot data about material flow. In the following, we summarize the main contributions of this thesis by highlighting the major research accomplishments.

Advection Through Flow Map Composition

In the first part, we develop theory and advanced numerical tools to accurately predict and quantify material transport in fluid dynamic systems. Specifically, we propose a novel numerical methodology for the computation of advective and advective–diffusive–reactive transport of tracer quantities in fluid flows. Our *'method of flow map composition'* is readily parallelizable, ensures that the numerical errors are not compounded in time resulting in a much higher accuracy, and can be used in conjunction with any of the well-established numerical schemes and partial differential equation (PDE) solvers. We also derive rigorous theoretical expressions for numerical errors and for the optimal composition timestep, *i.e.* the numerical timestep value that results in the minimum total error. We show that the optimal composition timestep minimizes the net numerical error by balancing the errors from the advection PDE solve and the composition-interpolation operation. Our theoretical results are analyzed and illustrated through several idealized and realistic two and three-dimensional examples, including regional ocean plume simulations. We discover that in both hindcast and real-time studies, the composition-based results match very well with the observations ehile preserving the accuracy and ensuring uniform spatial coverage. These examples validate our theoretical developments and showcase the efficiency and the superiority of the method of composition for predicting advective transport when compared to regular numerical advection schemes. The increased accuracy and decreased computational cost along with the ease of implementation make the method of flow map composition an attractive choice for a wide range of flow and material transport applications.

Lagrangian Analysis of Material Transport

We utilize the method of flow map composition to analyze Lagrangian material coherence in dynamic open domains, with special attention to flow fields with multiple time-dependent inlets and outlets. Specifically, we quantify sets of fluid parcels that remain the most (or the least) rigid over an extended time interval of interest, referred to as *'persistently rigid (non-rigid) sets'*. Such sets are of paramount importance in

quantifying fluid mixing, as it can be shown that these material sets that remain rigid are the ones to minimally mix with their complement. This coherence-incoherence information is valuable in determining turbulent, chaotic, and diffusive mixing regions in the flow. Based on the previously theorized 'polar distance', we propose the *'extended polar distance'* criterion, which quantifies the persistent rigidity or coherence (or the lack thereof) of all the possible fluid sets in the spatio-temporal domain of interest. We prove that the persistently rigid material sets determined using the extended polar distance are the ones that maximally resist both chaotic advection and diffusive mixing. Further, by utilizing the method of composition, we develop efficient schemes to compute this parameter. Several analytical flow fields and realistic data-assimilative ocean simulations in diverse dynamical regimes are considered as examples, wherein we demonstrate the capabilities of our criterion to capture the persistently coherent and incoherent sets. We discover that even through two material sets might initially be very close to each other, their behaviors can be very different from each other.

Inference and Prediction of Material Transport Features

Finally, we focus our attention on the machine inference and prediction of the transport features and coherent structures only from the snapshot data about the passive advection and advection–diffusion of some material. Our goals include machine learning and predicting the underlying flow maps, coherent / incoherent material sets, and attracting and repelling manifolds, given a time series of snapshot data about material transport. We utilize and extend the recent developments from machine learning (ML) for this purpose. The fundamental difference in our case from the typical supervised ML applications is that there does not exist any labeled training data of material flow snapshots and the corresponding flow maps / coherent structures. However, we have the knowledge of the physical constraints that the results must satisfy (e.g. conservation laws). We thus develop new theory and algorithms that can leverage our physical knowledge of the system in the design of the network and the constraints on the output fields in the loss function to be minimized. Specifically, we

extend recurrent neural networks, including a combination of long short-term memory networks with hypernetworks, and introduce relaxed weight sharing, while enforcing advective transport, mass conservation, and velocity smoothness in the loss function. The developed theory, methods, and computational architectures are validated and tested on a variety of analytical and realistic fluid flows, including high-resolution ocean transports in the Western Mediterranean Sea. The results highlight our ability to extract the generic transport features from specific material flow data.

1.2 Outline of the Thesis

Flow Maps and Advective Transport: In Chapter 2, we formally introduce the notion of flow maps of a dynamical system, which forms the backbone of this thesis. We describe the mathematical setup and the relevant notation, define the flow maps, discuss their governing equations, and study the relationship between flow maps and passive advective transport.

Advection Through Flow Map Composition - Theory: Chapter 3 discusses the theoretical foundations of the method of flow map composition to numerically solve the advective transport equation. We first describe the proposed method, followed by provisions to account for additional source and diffusion terms as well as the application of various different boundary conditions. This is followed by rigorous error analysis to obtain global error bounds and the optimal composition timestep - a numerical timestep value that yields the least total numerical error. Some results from this chapter have been published in Kulkarni and Lermusiaux [132].

Advection Through Flow Map Composition - Applications: In Chapter 4, we exemplify the method of composition using a variety of analytical and realistic applications. We first benchmark and analyze the developed scheme using advection and advection–diffusion in a reversible swirl flow with a known solution and an advection–reaction in an idealized flow exiting a strait. This is followed by the prediction of simulated

advective transport of sediment plumes in the Bismarck Sea and the real-time prediction of sediment plume transport during the 2018 PLUMEX sea exercise. Results from this chapter have been published in Kulkarni and Lermusiaux [132]; Kulkarni et al. [131]; Munoz-Royo et al. [181]; Coulin et al. [38].

Lagrangian Analysis of Material Transport - Theory: Chapter 5 establishes the role of flow maps as a backbone of Lagrangian material transport and extends the method of composition to compute flow maps in dynamic open domains. This is followed by the development of rigorous theory and criteria to predict sets of fluid parcels that remain the most (or the least) rigid over an extended time interval, along with several remarks that highlight the objectivity, efficient computation, and theoretical connections of the proposed criteria to chaotic advection and diffusive mixing. This work is the basis of Kulkarni and Lermusiaux [136].

Lagrangian Analysis of Material Transport - Applications: In Chapter 6, first we extensively benchmark the computation of flow maps for open domains using a realistic marine flow around the island of Palau in the Southern Pacific Ocean. This is followed by a demonstration of some of the applications of the software toolbox based on the method of composition to analyze Lagrangian coherence in a variety of dynamical regimes and different marine regions around the world. We then focus on the prediction of coherent, persistently coherent, and incoherent sets in the analytical double gyre flow and in realistic data-assimilative ocean simulations around Palau. The results from this chapter appear in Kulkarni and Lermusiaux [136]; Doshi et al. [47]; Lermusiaux et al. [153].

Inference of Material Transport Features - Theory: In Chapter 7, we develop a novel machine learning (ML) based algorithm that can infer and predict the underlying advective transport features and coherent structures using a time series of snapshot data about material transport. We describe the proposed deep recurrent neural network architecture, the loss function, and its extension to learn flow maps from data about

advective–diffusive material flow. Finally, we extend our algorithm to learn flow maps in open domains, simultaneously learn from transport data about multiple quantities, and optimally combine low resolution snapshot data and high resolution Lagrangian trajectory data. This work is a part of Kulkarni and Lermusiaux [133, 134].

Inference of Material Transport Features - Applications: The applications of our ML framework developed in Chapter 7 on a variety of analytical and realistic flows are detailed in Chapter 8. We look at the inference and prediction capability of the ML algorithm through two analytical flow fields: a swirl flow and a double gyre flow. We then study the applications of the developed algorithm to learn the flow maps and coherent structures in the Western Mediterranean (Alboran) Sea using data about sea surface temperature and salinity. Results from this work appear in Kulkarni and Lermusiaux [133, 134].

Chapter 2

Advective Transport and Flow Maps

In this chapter, we formally introduce the notion of flow maps of non-autonomous (*i.e.* time-dependent) dynamical systems. In fluid dynamic systems, flow maps directly govern the advective transport of any passive material, and they can be viewed as the Lagrangian counterpart to the Eulerian velocity fields. In Sec. 2.1, we first introduce the Eulerian and Lagrangian perspectives in fluid mechanics to highlight why one must adopt the Lagrangian perspective while studying material transport. We then look at the associated mathematical setup and notation in Sec. 2.2. The governing equations satisfied by passive advective transport of any material quantity from the Eulerian and the Lagrangian viewpoints are specified in Sec. 2.3. Finally, Sec. 2.4 defines the flow maps, specifies their governing equations, and other important properties. This section also establishes the relationship between advective tracer transport and flow maps, thereby highlighting the role of flow maps as the backbone of passive material transport. The various contributions of this thesis are directly connected to flow maps, and this chapter serves to build the necessary background and technical knowledge.

2.1 Introduction

2.1.1 Eulerian and Lagrangian Descriptions of Fluid Flow

There are two ways to describe the motion of any continuum entity (e.g. fluids). In the first approach, often referred to as the *Eulerian perspective* or *field description of fluid motion*, the properties of the flow field are monitored at fixed locations in space. However, the complimentary viewpoint, called the *Lagrangian perspective* or the *particle description of fluid motion*, follows individual fluid parcels as they move through the flow field. These two perspectives are equivalent and either of them is capable of completely describing any fluid flow and the related processes. However, both have certain strengths and pitfalls, hence an understanding of both descriptions and the ability to switch between these viewpoints is of paramount importance. For example, the acceleration following a fluid parcel is needed for the application of Newton's second law to fluid flows and the motion of individual fluid parcels is necessary to understand the transport of material carried by these parcels. However, observations, measurements, and simulations of fluid flows are commonly made at fixed spatial locations with the fluid moving past the locations or through certain regions.

The Eulerian description focuses on the flow field and its properties at fixed locations or in regions of interest, and typically involves three or four independent variables: the two or three spatial coordinates (represented by the position vector $\boldsymbol{x}$), and time t. Thus, in this field-based Eulerian description of fluid motion, any scalar, vector, or tensor flow field property or a related quantity, denoted by F, depends directly on $\boldsymbol{x}$ and t as $F = F(\boldsymbol{x}, t)$.

The Lagrangian description of fluid flow is the direct extension of single particle kinematics to a whole field of fluid parcels labeled by their initial location, $\boldsymbol{r}_0$, at an initial reference time, $t = t_0$. The subsequent position $\boldsymbol{r}$ of each parcel as a function of time, $\boldsymbol{r}(t; \boldsymbol{r}_0, t_0)\ \forall\ \boldsymbol{r}_0$, implicitly specifies the the flow field. Here, $\boldsymbol{r}_0$ and t_0 are the boundary or initial condition parameters that label each of the fluid parcels, and are not independent variables. The only independent variable is t. Thus, the instantaneous flow field velocity $\boldsymbol{v}$ at the location $\boldsymbol{x}$ at time t is the same as the

instantaneous velocity of the fluid parcel that was located at $\boldsymbol{r}_0$ at time t_0 which now is at $\boldsymbol{r}$ $(= \boldsymbol{x})$ at time t. That is,

$$\frac{d\boldsymbol{x}(t)}{dt} = \boldsymbol{v}(\boldsymbol{x}(t), t) = \boldsymbol{v}(\boldsymbol{r}, t) = \frac{d\boldsymbol{r}(t; \boldsymbol{r}_0, t_0)}{dt} \,. \tag{2.1}$$

In this particle-based Lagrangian description of fluid motion, fluid parcel kinematics are identical to that in ordinary particle mechanics, and any flow field property or a related quantity affected by the flow field F may depend on the path(s) followed of the relevant fluid parcel(s) and time, *i.e.* $F = F(\boldsymbol{r}(t; \boldsymbol{r}_0, t_0), t)$.

Kinematic relationships between the two descriptions can be determined by requiring equality of the flow field properties when $\boldsymbol{x}$ and $\boldsymbol{r}$ define the same point in space, both are resolved in the same coordinate system, and a common clock is used to determine the time t:

$$F(\boldsymbol{r}(t; \boldsymbol{r}_0, t_0), t) = F(\boldsymbol{x}, t) \quad \text{when} \quad \boldsymbol{x} = \boldsymbol{r}(t; \boldsymbol{r}_0, t_0) \,. \tag{2.2}$$

Here the condition $\boldsymbol{x} = \boldsymbol{r}(t; \boldsymbol{r}_0, t_0)$ specifies the trajectory followed by a fluid parcel. Taking the total time derivative of Eq. (2.2) yields Eq. (2.3):

$$\frac{dF(\boldsymbol{x}, t)}{dt} = \frac{dF(\boldsymbol{r}(t; \boldsymbol{r}_0, t_0), t)}{dt} = \frac{\partial F}{\partial t} + \frac{d\boldsymbol{r}}{dt} \cdot \nabla_{\boldsymbol{r}} F \,. \tag{2.3}$$

However, we have that $\frac{d\boldsymbol{r}}{dt} = \boldsymbol{v}$ from Eq. (2.1), and as $\boldsymbol{x} = \boldsymbol{r}(t; \boldsymbol{r}_0, t_0)$, we have that $\nabla_{\boldsymbol{r}} = \nabla_{\boldsymbol{x}} = \nabla$. Thus we define the *total derivative* or the *material derivative* of any quantity F as:

$$\frac{DF(\boldsymbol{x}, t)}{Dt} \triangleq \frac{\partial F(\boldsymbol{x}, t)}{\partial t} + \boldsymbol{v}(\boldsymbol{x}, t) \cdot \nabla F(\boldsymbol{x}, t) = \frac{dF(\boldsymbol{r}(t; \boldsymbol{r}_0, t_0), t)}{dt} \quad \text{where} \quad \boldsymbol{x} = \boldsymbol{r}(t; \boldsymbol{r}_0, t_0) \,. \tag{2.4}$$

Material derivative $\frac{D\bullet}{Dt}$ in the Eulerian description is equivalent to the total time derivative $\frac{d\bullet}{dt}$ in the Lagrangian formulation. It is also called *particle derivative*, signifying that it provides Eulerian time derivative information following a fluid parcel.

Material derivative is the key component in transitioning between the Eulerian and

the Lagrangian perspectives and is composed of unsteady and advective acceleration terms. The unsteady part of $\frac{DF}{Dt}$, *i.e.* $\frac{\partial F}{\partial t}$, is the local temporal rate of change of F at the location $\boldsymbol{x}$. The advective (or convective) part of $\frac{DF}{Dt}$, $\boldsymbol{v} \cdot \nabla F$, is the rate of change of F that occurs as fluid parcels move from one location to another due to the velocity field $\boldsymbol{v}$. It is zero either where F is spatially uniform, where the fluid is not moving, or where $\boldsymbol{v}$ and ∇F are orthogonal. For clarity and consistency, the movement of fluid parcels from place to place is referred to as *advection* throughout this work, and we avoid using the term convection as it is typically used to specifically describe heat transport through fluid movement.

Eulerian description of a fluid flow can often be visualized through *streamlines*, which are curves that are instantaneously tangent to the fluid velocity throughout the flow field. Lagrangian description of fluid flow is formulated through *pathlines*. A pathline is the trajectory (through time) of a fluid parcel of fixed identity. Eulerian description of the flow field through streamlines is defined over the entire domain of interest, but is only defined at an instance in time. That is, a streamline description of the fluid flow can be obtained at every time instant and the streamline plots at two different time instants are in general independent of each other. However, the Lagrangian description of the fluid flow using pathlines describes an integrated effect through the entire time interval of interest, but the pathlines are independent for two different fluid parcels. Streamlines and pathlines coincide in steady flows but have no direct relation in unsteady flows.

We refer the readers to the classic texts of Batchelor [15]; Kundu et al. [139] for an extensive introduction as well as a detailed review of Eulerian and Lagrangian viewpoints in fluid mechanics.

2.1.2 Advective Material Transport in Fluid Flows

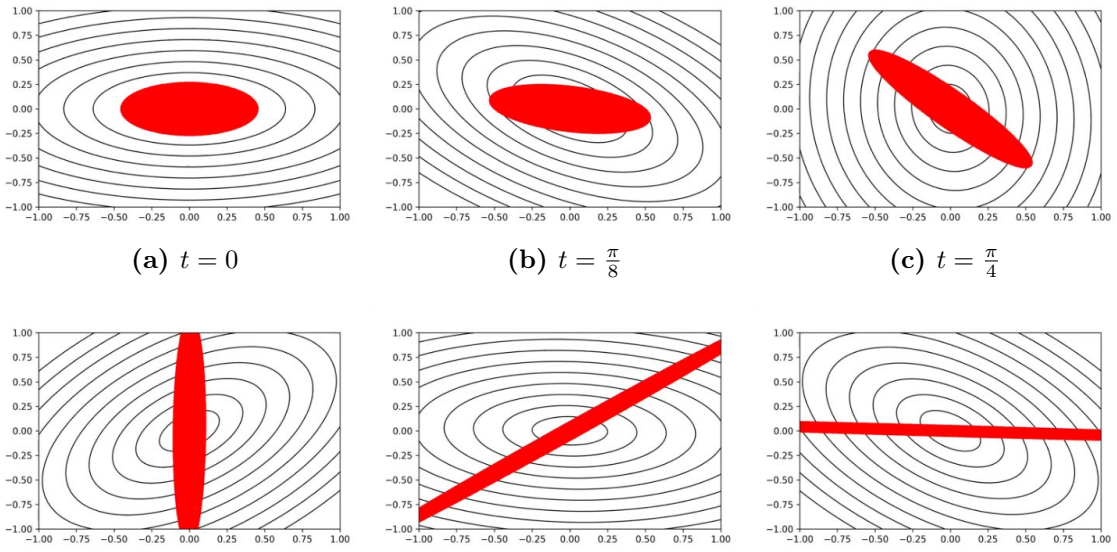
It is of paramount importance to understand the transport of 'tracers' in any fluid flow. A *tracer* is defined as any scalar or vector material, property, or quantity that is transported with the fluid parcels due to the motion of these parcels. The transport can be active (*i.e.* active tracers) or passive (*i.e.* passive tracers), depending on if

changes in the tracer values do, or do not, affect the fluid motions [139; 42]. Common examples of tracers in geophysical environments include temperature, salinity, dissolved Oxygen, plankton etc [42]. Further, external materials in the ocean such as oil (from oil spills), marine plastics, debris from airplanes or ships often behave as passive tracers and are advected with the background flow. Thus, predicting, quantifying, and understanding advective tracer transport is extremely valuable.

Typically, tracers are expressed in terms of their concentration, which is an Eulerian field, and the movement of tracer is reflected as the change in the concentration field. In this thesis, we will refer to the tracer concentration field as the *tracer field* for convenience. As mentioned before, passive tracers travel with the fluid parcels without affecting the dynamics of the fluid parcels. Thus, the path followed by a tracer is exactly the same as that of the fluid parcel that this tracer belongs to. Further, every tracer is transported along the trajectory originating at the initial location of the tracer at the initial time. This inherently implies that one must adopt the Lagrangian view while studying the transport of these tracers (that are contained within the fluid parcels).

However, many researchers have attempted to utilize Eulerian information such as the streamlines of the flow field, the vorticity field etc. to predict and quantify material transport. Criteria such as the Okubo-Weiss parameter [186; 256] which identifies regions where vorticity dominates strain have been used to detect coherent vortices. However, these formulations do not account for the temporally integrated effect of the flow field on tracer transport. Further, they all lack objectivity (observer independence), as well as a clear mathematical connection to sustained material coherence. As a consequence, vortex boundaries suggested by instantaneous Eulerian diagnostics tend to lose their coherence rapidly under advection in unsteady flows, rendering these metrics insufficient [99; 120; 61; 21].

The incapability of Eulerian diagnostics to quantify material transport and coherence is illustrated through the following example [101]. Consider the velocity field



(a) $t = 0$ **(b)** $t = \frac{\pi}{8}$ **(c)** $t = \frac{\pi}{4}$

(d) $t = \frac{3\pi}{8}$ **(e)** $t = \frac{\pi}{2}$ **(f)** $t = \frac{5\pi}{8}$

Figure 2-1: Streamlines and advective tracer transport in the velocity field specified by Eq. (2.5). The streamlines of the flow field at every instant are denoted by solid black lines and are concentric ellipses. The red regions in each panel denote the tracer field. Note that even though all the streamlines of the flow field are closed at all times and the tracer is initiated within one such elliptical 'vortex', it exponentially diverges away from the center.

given by Eq. (2.5):

$$\dot{\boldsymbol{x}} = \boldsymbol{v} = \begin{bmatrix} \sin 4t & 2 + \cos 4t \\ -2 + \cos 4t & -\sin 4t \end{bmatrix} \boldsymbol{x} \,. \tag{2.5}$$

The streamlines of this field are plotted as black solid lines in Fig. (2-1) for various time instants. Once can clearly observe that all the streamlines are closed and maintain concentric elliptical shapes while these ellipses rotate about the origin with a uniform speed and a period of $\pi/2$. Due to the closed nature of the streamlines, most Eulerian diagnostics (e.g. the Okubo-Weiss parameter) detect this as a vortex.

To actually verify this claim, Haller [98] initiate a tracer field, denoted by red in Fig. (2-1), in a region contained within one of the closed streamlines at the initial time. This tracer is passively advected under the action of the velocity field $\boldsymbol{v}$ from Eq. (2.5), essentially by solving Eq. (2.1) for all the tracer start locations. Contrary to the predictions of the Eulerian diagnostics, we observe that instead of being contained within the area inside the original streamline, the tracer diverges exponentially fast

away from the origin and ends up forming a thin filament. This example clearly highlights the fact that even in simple periodic flows, the behavior of passive tracer transport is vastly different from that predicted by the Eulerian diagnostics or that obtained by simply looking at the snapshots of the velocity field. Thus there is no basis to expect that such Eulerian criteria and analysis would yield reliable and correct predictions of advective material transport in complex aperiodic flows, and hence one must resort to the Lagrangian or particle tracking approach to predict, analyze, and quantify advective transport in fluid flows.

Given the initial tracer field, its evolution can be obtained analytically for a few known velocity fields [158]. In most cases however, the dynamic velocity fields $\boldsymbol{v}(\boldsymbol{x}, t)$ are derived as the output of some computational fluid dynamics (CFD) simulation or are observed quantities. In such cases, one must numerically compute the evolution of the tracer concentration field. It must be made clear that throughout this thesis, we assume that the tracer advected with the fluid does not alter the velocity of the fluid in any way. Further, it is assumed that the velocity fields $\boldsymbol{v}(\boldsymbol{x}, t)$ used in this work are either obtained through simulations, measurements, or observations and are known as a function of space and time. This work does not deal with computing these velocity fields. Our focus in the first two parts of this thesis is the prediction and analysis of advective material transport in these known dynamic velocity fields. In the last part, we do not have any information about the velocity fields. Rather we are given data about the transport of a particular tracer, and our objective is to infer the flow maps and coherent structures (not the velocity fields) from this information.

2.2 Setup and Notation

In this section, we define our problem setup and the relevant notation that would be used throughout this thesis. Let the spatial domain of interest be denoted by $\Omega \subset \mathbb{R}^n$, and its boundary by $\partial\Omega$. n is the spatial dimension of the problem (often, $n = 2$ or 3). The outward normal over $\partial\Omega$ is defined by $\hat{\mathbf{n}}_{\partial\Omega}$. The (unsteady) velocity field is given by $\boldsymbol{v} : \Omega \times [0, T] \to \mathbb{R}^n$, where $[0, T]$ is the fixed time interval of interest. Further,

we assume that the velocity $\boldsymbol{v}$ is Lipschitz continuous in the spatial coordinate with Lipschitz constant $\mathcal{L}_{\boldsymbol{v}}$. That is, $|\boldsymbol{v}(\boldsymbol{x},t) - \boldsymbol{v}(\boldsymbol{y},t)| \leq \mathcal{L}_{\boldsymbol{v}} |\boldsymbol{x} - \boldsymbol{y}|$ for all $\boldsymbol{x}, \boldsymbol{y} \in \Omega$ and $0 \leq t \leq T$. $\boldsymbol{x}$ indicates a general spatial coordinate in Ω and t indicates a general temporal coordinate within $[0, T]$. The density of the fluid flow at $(\boldsymbol{x}, t)$ is $\rho(\boldsymbol{x}, t)$.

The tracer, whose advection is to be studied is denoted by α. Although many physical tracers tend to be scalar, we do not limit ourselves to scalar tracer concentrations. We assume that in general, the tracer concentration α can be vector valued. Thus, $\alpha : \Omega \times [0, T] \to \mathcal{R}_\alpha$, where $\mathcal{R}_\alpha$ is the range of values of α. Often times, $\mathcal{R}_\alpha = \mathbb{R}^+ \cup \{0\}$ for physical scalar tracers (e.g. salinity), but $\mathcal{R}_\alpha = \Omega$ if studying the advective transport of coordinate positions. The initial tracer concentration over the entire domain is assumed to be known and denoted by $\alpha_0 : \Omega \to \mathcal{R}_\alpha$. We assume that α_0 is also Lipschitz continuous, with Lipschitz constant $\mathcal{L}^0_\alpha$, *i.e.* $|\alpha_0(\boldsymbol{x}) - \alpha_0(\boldsymbol{y})| \leq \mathcal{L}^0_\alpha |\boldsymbol{x} - \boldsymbol{y}|$ for all $\boldsymbol{x}, \boldsymbol{y} \in \Omega$.

To perform numerical analysis and computation, we assume that the spatio-temporal domain of interest is discretized. Specifically, we assume that the temporal direction is discretized into N_t discrete times with a timestep of Δt. The spatial domain is discretized into N_x, N_y, and N_z nodes along the X, Y, and Z directions respectively, with a corresponding grid spacing of Δx, Δy, and Δz.

The transpose of a matrix $\bullet$ is denoted by $\bullet^*$ and its determinant by $\det(\bullet)$. A well-defined tensor norm of a tensor $\bullet$ is denoted by $|\bullet|$. Finally, the numerically computed value of a quantity $\bullet$ is denoted by $\tilde{\bullet}$. Table 2.1 summarizes the relevant notion that is used throughout this thesis.

2.3 Advective Tracer Transport

We now look at the fundamental governing equations for the evolution of the tracer field α from the Eulerian and Lagrangian perspectives.

The evolution of the tracer $\alpha(\boldsymbol{x}, t)$ is governed by the classic advective transport

Table 2.1: Notation and Symbols used in this thesis.

Symbol	Description
$\bullet^*$	Transpose of a matrix $\bullet$
$\det(\bullet)$	Determinant of a matrix $\bullet$
$\lvert\bullet\rvert$	Norm of a tensor $\bullet$
$\tilde{\bullet}$	Numerically computed value of $\bullet$
n	Spatial dimension
$\mathbb{I}_d$	Identity matrix of size $d \times d$
$\Omega \in \mathbb{R}^n$	Spatial domain
$\partial\Omega$	Boundary of Ω
$\hat{\mathbf{n}}_{\partial\Omega}$	Outward normal over $\partial\Omega$
$\boldsymbol{x} \in \Omega$	Spatial position
$[0, T]$	Time interval of interest
$t \in [0, T]$	temporal coordinate
$\boldsymbol{v} : \Omega \times [0, T] \to \mathbb{R}^n$	Unsteady velocity field
$\alpha : \Omega \times [0, T] \to \mathcal{R}_\alpha$	Passive tracer
$\alpha_0 : \Omega \to \mathcal{R}_\alpha$	Initial concentration of α
$\mathcal{L}_{\boldsymbol{v}} \in \mathbb{R}^+$	Lipschitz constant of the velocity field $\boldsymbol{v}$
$\mathcal{L}^0_\alpha \in \mathbb{R}^+$	Lipschitz constant of the initial tracer field α_0
$\rho : \Omega \times [0, T] \to \mathbb{R}^+ \cup \{0\}$	Density of the fluid
$\boldsymbol{\phi}^t_s : \Omega \to \Omega$	Flow map between times s and t
$C^t_s(\boldsymbol{x}) = [\nabla \boldsymbol{\phi}^t_s(\boldsymbol{x})]^* [\nabla \boldsymbol{\phi}^t_s(\boldsymbol{x})]$	Right Cauchy-Green (CG) strain tensor
$\lambda_i^{(s,t)}(\boldsymbol{x}) \quad 1 \leq i \leq n$	Eigenvalues of $C^t_s(\boldsymbol{x})$ indexed in decreasing order
$\xi_i^{(s,t)}(\boldsymbol{x})$	Eigenvectors corresponding to $\lambda_i^{(s,t)}(\boldsymbol{x})$
N_x, N_y, N_z	Number of grid points in the spatial dimension(s)
$N = N_x \times N_y \times N_z$	Number of total grid points in the domain
$\Delta x, \Delta y, \Delta z$	Grid spacing in the spatial dimension(s)
N_t	Number of intervals in the temporal dimension
Δt	Numerical timestep

Eq. (2.6), subject to appropriate boundary conditions [15; 67]:

$$\begin{aligned} \frac{\partial \rho(\boldsymbol{x}, t)\alpha(\boldsymbol{x}, t)}{\partial t} + \nabla \cdot (\rho(\boldsymbol{x}, t)\boldsymbol{v}(\boldsymbol{x}, t)\alpha(\boldsymbol{x}, t)) = 0\,, \\ \alpha(\boldsymbol{x}, 0) = \alpha_0(\boldsymbol{x})\,. \end{aligned} \tag{2.6}$$

This equation models the transport of a passive tracer α defined per unit mass of the fluid, under the velocity field $\boldsymbol{v}$. Further, from mass conservation, we have that,

$$\frac{\partial \rho(\boldsymbol{x}, t)}{\partial t} + \nabla \cdot (\rho(\boldsymbol{x}, t)\boldsymbol{v}(\boldsymbol{x}, t)) = 0\,, \tag{2.7}$$

Eq. (2.6) is simplified into Eq. (2.8) using Eq. (2.7), without further assumptions:

$$\begin{aligned} &\frac{\partial \alpha(\boldsymbol{x},t)}{\partial t} + \boldsymbol{v}(\boldsymbol{x},t)\cdot\nabla\alpha(\boldsymbol{x},t) = 0\,, \\ &\alpha(\boldsymbol{x},0) = \alpha_0(\boldsymbol{x})\,. \end{aligned} \tag{2.8}$$

If the tracer concentration was defined per unit volume, the equation equivalent to Eq. (2.6) is given by Eq. (2.9),

$$\begin{aligned} &\frac{\partial \alpha(\boldsymbol{x},t)}{\partial t} + \nabla\cdot(\boldsymbol{v}(\boldsymbol{x},t)\alpha(\boldsymbol{x},t)) = 0\,, \\ &\alpha(\boldsymbol{x},0) = \alpha_0(\boldsymbol{x})\,. \end{aligned} \tag{2.9}$$

Eq. (2.9) simplifies to Eq. (2.8) if one assumes an incompressible velocity field, *i.e.* $\nabla\cdot\boldsymbol{v} = 0$. In this work, we assume that the fluid is incompressible and thus tracer transport is governed by Eq. (2.8). As Eq. (2.8) specifies the evolution of the tracer field over a fixed domain through time, it is considered an Eulerian description of α.

One can equivalently consider the Lagrangian description of tracer transport corresponding to the Eulerian Eq. (2.8). Comparing Eq. (2.8) with Eq. (2.4), we get:

$$\begin{aligned} &\frac{\partial \alpha(\boldsymbol{x},t)}{\partial t} + \boldsymbol{v}(\boldsymbol{x},t)\cdot\nabla\alpha(\boldsymbol{x},t) = 0 \quad \text{with} \quad \alpha(\boldsymbol{x},0) = \alpha_0(\boldsymbol{x})\,, \\ \implies &\frac{D\alpha(\boldsymbol{x}(t),t)}{Dt} = 0 \quad \text{with} \quad \alpha(\boldsymbol{x},0) = \alpha_0(\boldsymbol{x})\,. \end{aligned} \tag{2.10}$$

Eq. (2.10) follows the motion of individual tracers through time and hence is the Lagrangian description of tracer transport [43; 159].

2.4 Flow Maps

2.4.1 Introduction and Definitions

The two principal questions in advective tracer transport, given a fixed time interval, are: (i) where does a tracer that started at a particular location at the initial time end up at the final time? and (ii) where did a tracer come from at the initial time

that ends up at a particular location at the final time? Since any advective tracer passively travels with the fluid parcels, the mapping between the initial and the final concentrations of a tracer can be determined through the mapping between the initial and the final positions of the fluid parcels in the concerned domain. The *forward and backward flow maps* corresponding to the velocity field $\boldsymbol{v}$ quantify this mapping [237].

The *forward flow map* describes the mapping of the positions in Ω at the initial time to their respective locations at the final time under the advective action of the velocity field $\boldsymbol{v}$, given by Eq. (2.11):

$$\boldsymbol{\phi}_0^T(\boldsymbol{x}_0) = \boldsymbol{x}(T) \;\; \text{where} \;\; \frac{d\boldsymbol{x}}{dt} = \boldsymbol{v}(\boldsymbol{x}(t), t);\; \boldsymbol{x}(0) = \boldsymbol{x}_0 \in \Omega \;\; \text{s.t.} \;\; \boldsymbol{x}(T) \in \Omega\,. \tag{2.11}$$

Similarly, the *backward flow map* describes the mapping of the positions in Ω at the final time to their respective locations at the initial time, and is the inverse of the forward flow map:

$$\boldsymbol{\phi}_T^0(\boldsymbol{x}_T) = \boldsymbol{x}(0) \;\; \text{where} \;\; \frac{d\boldsymbol{x}}{dt} = \boldsymbol{v}(\boldsymbol{x}(t), t);\; \boldsymbol{x}(T) = \boldsymbol{x}_T \in \Omega \;\; \text{s.t.} \;\; \boldsymbol{x}(0) \in \Omega\,. \tag{2.12}$$

The ODE in Eq. (2.12) is solved backwards in time with the specified terminal condition starting from $t = T$ ending at $t = 0$.

Note that in general, the flow maps are not defined at all locations in Ω. Specifically, the forward flow map is undefined at locations that exit Ω during $[0, T]$ and the backward flow map is undefined at locations that enter the domain during $[0, T]$. This implies that the domains of definitions of the forward and the backward flow map are different from each other and are a function of the end time T. Further, the forward flow map over a time interval is defined at the start time of this interval, whereas the backward flow map over a time interval is defined at the end time of this interval. This together implies that the forward and the backward flow maps are not directly comparable with each other.

One may look at flow maps from two complementary perspectives: (i) as a diffeomorphism or (ii) as a functional operator. The first perspective considers flow maps as a vector-valued Eulerian fields $\boldsymbol{x} \to \boldsymbol{\phi}_0^T(\boldsymbol{x})$ and $\boldsymbol{x} \to \boldsymbol{\phi}_T^0(\boldsymbol{x})$ that describe the

motion of individual positions $\boldsymbol{x} \in \Omega$ forward and backward in time respectively. In the second perspective, flow maps are viewed as functional operators $\mu \rightarrow \mu \circ \boldsymbol{\phi}_T^0$ and $\mu \rightarrow \mu \circ \boldsymbol{\phi}_T^0$ that map all density functions μ on Ω to their corresponding solutions from the advection equation (Eq. (2.8)) forward and backward in time respectively. These perspectives can be shown to be equivalent by considering the action of the flow map operator on the density function $\mu = \boldsymbol{x}$. Further details about the equivalence of diffeomorphism and operator based approaches in dynamical systems can be found in [6; 44]. This work mainly deals with understanding and utilizing the role of the Eulerian flow map fields in the transport of tracers. We thus refer to the flow map fields (*i.e.* $\boldsymbol{\phi}_0^T(\boldsymbol{x})$ and $\boldsymbol{\phi}_T^0(\boldsymbol{x})$) as 'flow maps' for convenience henceforth. The operator theoretic notion of the flow map (*i.e.* $\mu \rightarrow \mu \circ \boldsymbol{\phi}_T^0$) is referred to as the the 'flow map operator' when required.

2.4.2 Governing Equations

The instantaneous evolution of the flow map $\boldsymbol{\phi}_0^t(\boldsymbol{x})$ can be quantified in terms of the velocity field. For an infinitesimally small time interval δt, we can write that:

$$\lim_{\delta t \rightarrow 0} \boldsymbol{\phi}_0^{t+\delta t}(\boldsymbol{x}) = \lim_{\delta t \rightarrow 0} \left(\boldsymbol{\phi}_0^t(\boldsymbol{x}) + \boldsymbol{v}(\boldsymbol{\phi}_0^t(\boldsymbol{x}), t)\delta t \right) . \tag{2.13}$$

Rearranging and replacing the differential limit with the derivative, we obtain:

$$\frac{d\boldsymbol{\phi}_0^t(\boldsymbol{x})}{dt} = \lim_{\delta t \rightarrow 0} \frac{\boldsymbol{\phi}_0^{t+\delta t}(\boldsymbol{x}) - \boldsymbol{\phi}_0^t(\boldsymbol{x})}{\delta t} = \boldsymbol{v}(\boldsymbol{\phi}_0^t(\boldsymbol{x}), t) . \tag{2.14}$$

To obtain the governing equations for the gradients of the forward flow maps, we differentiate Eq. (2.14) with respect to $\boldsymbol{x}$:

$$\frac{\partial}{\partial \boldsymbol{x}} \frac{d\boldsymbol{\phi}_0^t(\boldsymbol{x})}{dt} = \frac{\partial}{\partial \boldsymbol{x}} \left(\boldsymbol{v}(\boldsymbol{\phi}_0^t(\boldsymbol{x}), t) \right) . \tag{2.15}$$

Denoting $\frac{\partial \boldsymbol{\phi}_0^t(\boldsymbol{x})}{\partial \boldsymbol{x}}$ by $\nabla \boldsymbol{\phi}_0^t(\boldsymbol{x})$ and using chain rule, we write Eq. (2.15) as:

$$\frac{d\nabla \boldsymbol{\phi}_0^t(\boldsymbol{x})}{dt} = \left(\frac{\partial \boldsymbol{v}(\boldsymbol{\phi}_0^t(\boldsymbol{x}),t)}{\partial \boldsymbol{\phi}_0^t(\boldsymbol{x})}\right)\left(\nabla \boldsymbol{\phi}_0^t(\boldsymbol{x})\right) = \left(\nabla \boldsymbol{v}(\boldsymbol{\phi}_0^t(\boldsymbol{x}),t)\right)\left(\nabla \boldsymbol{\phi}_0^t(\boldsymbol{x})\right) . \tag{2.16}$$

Similarly, the evolution of the backward flow map is governed by Eq. (2.17)

$$\frac{d}{dt}\nabla \boldsymbol{\phi}_t^0(\boldsymbol{x}) = -\left(\nabla \boldsymbol{v}(\boldsymbol{\phi}_t^0(\boldsymbol{x}),t)\right)\left(\nabla \boldsymbol{\phi}_t^0(\boldsymbol{x})\right) . \tag{2.17}$$

It can be shown that the determinant of $\nabla \boldsymbol{\phi}_0^t(\boldsymbol{x})$ satisfies the following equation [176]:

$$\frac{d}{dt}\det(\nabla \boldsymbol{\phi}_0^t(\boldsymbol{x})) = \det(\nabla \boldsymbol{\phi}_0^t(\boldsymbol{x}))\left(\nabla \cdot \boldsymbol{v}(\boldsymbol{\phi}_0^t(\boldsymbol{x}),t)\right) . \tag{2.18}$$

For an incompressible flow we have that $\nabla \cdot \boldsymbol{v}(\boldsymbol{x},t) = 0 \;\forall\; \boldsymbol{x} \in \Omega,\; \forall\; t \in [0,T]$. This yields $\frac{d}{dt}\det(\nabla \boldsymbol{\phi}_0^t(\boldsymbol{x})) = 0$, implying that $\det(\nabla \boldsymbol{\phi}_0^t(\boldsymbol{x}))$ is a constant for all t. However, at $t = 0$, we have that $\boldsymbol{\phi}_0^0(\boldsymbol{x}) = \boldsymbol{x}$. Thus we get that:

$$\nabla \cdot \boldsymbol{v}(\boldsymbol{x},t) = 0 \implies \det(\nabla \boldsymbol{\phi}_0^t(\boldsymbol{x})) = \det(\nabla \boldsymbol{\phi}_0^0(\boldsymbol{x})) = \det(\mathbb{I}_n) = 1 . \tag{2.19}$$

Analogous results can also be obtained for the backward flow map. Overal, for incompressible flow fields, we obtain that for all times t:

$$\det(\nabla \boldsymbol{\phi}_0^t(\boldsymbol{x})) = \det(\nabla \boldsymbol{\phi}_t^0(\boldsymbol{x})) = 1 . \tag{2.20}$$

These constraints are utilized in various capacities in Chapter 5 and Chapter 7.

2.4.3 Flow Map Composition

We now show that the forward and the backward flow map computations can be decoupled in time. Consider an arbitrary time instant $\boldsymbol{\tau} \in [0,T]$. The forward flow

maps over $[0, \tau]$ and $[\tau, T]$ are defined by Eq. (2.21) and Eq. (2.22):

$$\boldsymbol{\phi}_0^{\tau}(\boldsymbol{x}_0) = \boldsymbol{x}(\tau) \;\; \text{where} \;\; \frac{d\boldsymbol{x}}{dt} = \boldsymbol{v}(\boldsymbol{x}(t), t); \;\; \boldsymbol{x}(0) = \boldsymbol{x}_0 \in \Omega \;\; \text{s.t.} \;\; \boldsymbol{x}(\tau) \in \Omega \,, \tag{2.21}$$

$$\boldsymbol{\phi}_{\tau}^{T}(\boldsymbol{x}_{\tau}) = \boldsymbol{x}(T) \;\; \text{where} \;\; \frac{d\boldsymbol{x}}{dt} = \boldsymbol{v}(\boldsymbol{x}(t), t); \;\; \boldsymbol{x}(\tau) = \boldsymbol{x}_{\tau} \in \Omega \;\; \text{s.t.} \;\; \boldsymbol{x}(T) \in \Omega \,. \tag{2.22}$$

Substituting $\boldsymbol{x}(\tau) = \boldsymbol{x}_{\tau} = \boldsymbol{\phi}_0^{\tau}(\boldsymbol{x}_0)$ from Eq. (2.21) in Eq. (2.22), we get that:

$$\boldsymbol{\phi}_{\tau}^{T}\left(\boldsymbol{\phi}_0^{\tau}(\boldsymbol{x}_0)\right) = \boldsymbol{x}(T) \;\; \text{where} \;\; \frac{d\boldsymbol{x}}{dt} = \boldsymbol{v}(\boldsymbol{x}(t), t); \;\; \boldsymbol{x}(0) = \boldsymbol{x}_0 \in \Omega \;\; \text{s.t.} \;\; \boldsymbol{x}(T) \in \Omega \,. \tag{2.23}$$

Comparing Eq. (2.23) to Eq. (2.11), one can clearly see that:

$$\boldsymbol{\phi}_0^{T}(\boldsymbol{x}_0) = \boldsymbol{\phi}_{\tau}^{T}\left(\boldsymbol{\phi}_0^{\tau}(\boldsymbol{x}_0)\right) = \boldsymbol{x}_T \;\; \text{for all} \;\; \boldsymbol{x}_0 \in \Omega \;\; \text{such that} \;\; \boldsymbol{x}_T \in \Omega \,. \tag{2.24}$$

The analogous expression for backward flow map is given by Eq. (2.25):

$$\boldsymbol{\phi}_T^{0}(\boldsymbol{x}_T) = \boldsymbol{\phi}_{\tau}^{0}\left(\boldsymbol{\phi}_T^{\tau}(\boldsymbol{x}_T)\right) = \boldsymbol{x}_0 \;\; \text{for all} \;\; \boldsymbol{x}_T \in \Omega \;\; \text{such that} \;\; \boldsymbol{x}_0 \in \Omega \,. \tag{2.25}$$

In terms of the flow map operators, we get that the flow map operators $\boldsymbol{\phi}_0^T$, $\boldsymbol{\phi}_0^{\tau}$, and $\boldsymbol{\phi}_{\tau}^T$ (or $\boldsymbol{\phi}_T^0$, $\boldsymbol{\phi}_{\tau}^0$, and $\boldsymbol{\phi}_T^{\tau}$) are connected through operator composition ($\circ$) as:

$$\boldsymbol{\phi}_0^T = \boldsymbol{\phi}_{\tau}^T \circ \boldsymbol{\phi}_0^{\tau} \quad \text{and} \quad \boldsymbol{\phi}_T^0 = \boldsymbol{\phi}_{\tau}^0 \circ \boldsymbol{\phi}_T^{\tau} \tag{2.26}$$

This implies that flow maps over a larger duration can be obtained by composing flow maps over shorter and *mutually independent* time intervals [132; 24; 266].

Specifically, when the considered time interval $[0, T]$ is divided into N_t smaller intervals $[t_0 = 0, t_1), [t_1, t_2), \ldots, [t_{N_t-1}, t_{N_t} = T]$, one can recursively apply Eq. (2.24) and Eq. (2.25) to obtain that,

$$\boldsymbol{\phi}_0^T(\boldsymbol{x}) = \boldsymbol{\phi}_{t_0}^{t_{N_t}}(\boldsymbol{x}) = \boldsymbol{\phi}_{t_{N_t-1}}^{t_{N_t}} \circ \boldsymbol{\phi}_{t_{N_t-2}}^{t_{N_t-1}} \circ \ldots \circ \boldsymbol{\phi}_{t_0}^{t_1}(\boldsymbol{x}) = \boldsymbol{\phi}_{t_{N_t-1}}^{t_{N_t}}\left(\ldots\left(\boldsymbol{\phi}_{t_0}^{t_1}(\boldsymbol{x})\right)\right) \,, \tag{2.27}$$

$$\boldsymbol{\phi}_T^0(\boldsymbol{x}) = \boldsymbol{\phi}_{t_{N_t}}^{t_0}(\boldsymbol{x}) = \boldsymbol{\phi}_{t_1}^{t_0} \circ \ldots \circ \boldsymbol{\phi}_{t_{N_t-1}}^{t_{N_t-2}} \circ \boldsymbol{\phi}_{t_{N_t}}^{t_{N_t-1}} = \boldsymbol{\phi}_{t_1}^{t_0}\left(\ldots\left(\boldsymbol{\phi}_{t_{N_t}}^{t_{N_t-1}}(\boldsymbol{x})\right)\right) \,. \tag{2.28}$$

Thus, Eq. (2.27)) and Eq. (2.28) suggest that the desired flow maps $\boldsymbol{\phi}_0^T(\boldsymbol{x})$ and $\boldsymbol{\phi}_T^0(\boldsymbol{x})$ can be recursively written as a composition of elementary functions (flow maps) over independent time intervals. This important observation allows us to decouple flow map computations in a mutually independent way.

2.4.4 Relationship Between Advective Transport and Flow Maps

The Lagrangian interpretation of advective material transport, *i.e.* Eq. (2.10) implies that the tracer value $\alpha_0(\boldsymbol{x}_0)$, initially located at $\boldsymbol{x}_0$, travels with the fluid parcel at $\boldsymbol{x}_0$ along the pathline (trajectory) originating at $\boldsymbol{x}_0$ to end up at the same final location as this fluid parcel. However, from Eq. (2.11), we know that the fluid parcel originating at $\boldsymbol{x}_0$ ends up at $\boldsymbol{x} = \boldsymbol{\phi}_0^T(\boldsymbol{x}_0)$ at the end time $t = T$. Thus, we have that:

$$\alpha(\boldsymbol{\phi}_0^T(\boldsymbol{x}_0), T) = \alpha(\boldsymbol{x}_0, 0) = \alpha_0(\boldsymbol{x}_0)\,, \tag{2.29}$$

$$\implies \alpha_0(\boldsymbol{x}) = \alpha(\boldsymbol{\phi}_0^T(\boldsymbol{x}), T)\,. \tag{2.30}$$

However, as the forward and the backward flow maps are inverses of each other, we can represent $\boldsymbol{x}_0$ as $\boldsymbol{x}_0 = \boldsymbol{\phi}_T^0(\boldsymbol{x})$ in Eq. (2.29). This yields:

$$\alpha(\boldsymbol{x}, T) = \alpha_0(\boldsymbol{\phi}_T^0(\boldsymbol{x}))\,. \tag{2.31}$$

Eq. (2.30) states that the initial tracer value at any position $\boldsymbol{x}$ is the composition of the final tracer field at time T with the forward flow map between times 0 and T, evaluated at $\boldsymbol{x}$. Eq. (2.31) states that the tracer value at any position $\boldsymbol{x}$ at time T is the composition of the initial tracer field with the backward flow map between times 0 and T, evaluated at $\boldsymbol{x}$. This implies that the backward (or forward) flow map completely determines the advected tracer field, given its initial (final) condition. That is, flow maps capture and quantify the underlying features of passive tracer advection independent of the actual tracer considered and thus are the generic advective transport maps over the considered time interval. Predicting and analyzing the transport features of any passive tracer is tantamount to predicting and quantifying the excep-

tional features of the flow maps. Hence, flow maps serve as the backbone of this work on predicting, analyzing, and inferring advective transport.

2.5 Summary

In this chapter, we introduce the two principal perspectives in fluid mechanics and justify why the Lagrangian perspective must be adopted to study advective transport in fluid flows. We then specify the mathematical setup and the notation used throughout this thesis. This is followed by the classic PDEs governing the advective transport of passive tracers. Finally, we look at the formal definition of the flow map along with its governing equations and delineate the connections between the flow maps and passive advective transport. This chapter builds fundamental knowledge about the notion of flow maps, and the contributions from the succeeding chapters utilize and build up on the concepts introduced in this chapter.

Chapter 3

Advection Through Flow Map Composition - Theory

In this chapter, we propose and theoretically investigate the foundations and capabilities of the *method of flow map composition*, a novel numerical methodology to compute the advection and advection–diffusion–reaction of tracer quantities. The method of composition accurately solves Eq. (2.8) to determine the evolution of the tracer field $\alpha(\boldsymbol{x}, t)$, given a (possibly unsteady) velocity field. Tracer advection occurs through flow map composition and is super-accurate, yielding numerical solutions almost devoid of compounding numerical errors, while allowing for direct parallelization in time. It is computed by implicitly solving the characteristic evolution through a modified transport partial differential equation and domain decomposition in the temporal direction, followed by composition with the known initial condition. This advection scheme allows a rigorous computation of the spatial and temporal error bounds, yields an accuracy comparable to that of Lagrangian methods, and maintains the advantages of Eulerian schemes. We further show that there exists an optimal value of the composition timestep that yields the minimum total numerical error in the computations, and derive the expression for this value. We prove that the optimal composition timestep minimizes the net numerical error by balancing the errors from the advection PDE solve and the composition-interpolation operation. The results from this chapter form the theoretical basis of Kulkarni and Lermusiaux

[132].

We first provide a brief introduction and a literature review of the existing methods in Sec. 3.1. The problem statement is specified in Sec. 3.2. The theoretical description of the composition-based advection is in Sec. 3.3; emphasizing the fundamentals behind the new methodology, the addition of tracer diffusion and source terms, and the implementation of boundary conditions. Sec. 3.4 derives specific numerical properties of the new composition-based numerical advection. We first analyze the error estimates and then derive the expression for the *optimal composition timestep* - the numerical timestep value that results in the minimum total numerical error. Finally, in Sec. 3.5, we discuss the development of a software toolbox that relies on the method of composition to analyze advective material transport. We detail the architecture of the solver, the flexibility afforded in the computation, and its performance metrics.

3.1 Introduction and Literature Review

The pure hyperbolic nature of Eq. (2.8) presents significant numerical challenges and most schemes are either susceptible to a high degree of unwanted numerical diffusion or exhibit excessive nonphysical oscillations [159]. Although the use of extremely fine mesh is an option for high fidelity solutions, the computational cost quickly becomes prohibitive and often rules out this possibility for large, realistic problems [158].

Computing the accurate solution of Eq. (2.8) is not trivial and there are several classes of numerical methods for such solutions. First, the more traditional 'Eulerian methods' or 'optimal spatial methods' [28] compute the numerical solution using finite differences [39; 196], finite volumes [159; 116; 172], finite elements [115; 69; 112], or other numerical discretization techniques on a spatial grid. It can be proven that central difference approximations for the temporal and spatial derivatives commonly yields excessive oscillations and unstable solutions [159]. Considering an upstream bias ('upwinding') for the spatial gradients, where the upwind direction is decided by the local velocity fields, removes these spurious oscillations to a great extent. Some of the benefits of these methods include rigorous theoretical bounds on the spatial

and temporal errors and straightforward use of well-established robust and high-order accurate numerical methods [179; 203; 243; 230; 191; 189], many of which are implemented in numerical toolboxes [7; 53; 138] However, they tend to exhibit significant numerical diffusion when the temporal truncation error dominates [159]. Such spurious diffusion effects can be large enough when they compound to overshadow the physical tracer dispersion [28], and hence produce inaccurate solutions. In order to minimize such errors, one can use high-order accurate schemes that rely on wide computational stencils increasing the cost and posing challenges near boundaries. The use of hp-adaptivity [126; 34] reduces the local truncation errors but does not eliminate their compounding over time. Further, explicit numerical schemes often pose restrictions on the Courant number whereas implicit methods involve a large matrix–vector or a nonlinear system solve. For such Eulerian methods, the computational cost can thus become high and there are no direct avenues for temporal parallelization.

The second type, often referred to as 'Lagrangian methods' leverage the hyperbolic nature of the system to compute tracer transport by utilizing the characteristic lines of Eq. (2.8). The various techniques developed under the umbrella of Lagrangian methods involve the method of characteristics [201], the modified method of characteristics [48; 57; 214], and transport-diffusion methods [108; 114] amongst others. Eulerian-Lagrangian methods (ELM) [12; 184; 188] and their extensions such as localized adjoint-based methods (ELLAM) [28; 255] are used for advection-diffusion equations where the advective transport is computed by a Lagrangian method, the diffusive component is treated on an Eulerian grid, and finally both solutions are combined through operator-splitting methods [258]. Lagrangian methods significantly reduce the temporal truncation error, alleviate the Courant number restrictions often found in Eulerian methods, and can be readily parallelized [171]. They pose problems however in the rigorous treatment of boundary fluxes, especially in domains with multiple inlets and outlets, and in maintaining mass conservation. Further, Lagrangian methods involve some form of particle tracking, which implies that the solutions are seldom computed on uniform spatial grids leading to a loss of resolution in some regions of the domain [156; 64; 65]. Further, the Lagrangian methods require the velocity field

defined not only at the (known) grid points, but also at arbitrary locations in the domain. Although high-order reconstruction or interpolation schemes can be used, they increase the overall computational cost. This also implies that rigorous estimates for the spatial error bounds are often not available.

Another class of advection schemes, called 'jet schemes' [220; 183] or 'characteristic mapping methods' [177] propose a semi-Lagrangian 'advect-and-project' approach to numerically solve Eq. (2.8). These schemes use Hermite polynomials to achieve local high-order accuracy, and the advection process is simulated by Lagrangian particle tracking. For classical examples of level set function evolutions [190], it is observed that these methods can produce solutions with accuracy comparable to discontinuous Galerkin methods. These methods may be combined with appropriate parallel-in-time methods [79] to achieve some speedup.

It would be ideal to obtain numerical diffusion- and dispersion-free accurate solutions comparable to those of Lagrangian methods, while maintaining the theoretical consistency, soundness and ease of use of Eulerian methods (*i.e.* on a spatially fixed grid). In this work, we develop a new method to numerically solve Eq. (2.8), referred to as the *method of flow map composition* or simply the *method of composition*, and extend it to advection-diffusion-reaction problems. It generalizes prior results in the area of flow map computation for dynamical systems [265; 266; 156]. As will be shown, this is a special case of the method of composition, *i.e.* solving Eq. (2.8) but for very specific initial and boundary conditions. A main novelty of the present methodology is its capability to handle the advection of any passive tracer while respecting space and time-dependent boundary conditions and to also account for any classic diffusion and reaction operators. We further prove that the order of accuracy of the method is at least equal to the minimum amongst the orders of accuracy of the constituent operations, which serves as a generalization of Theorem 1 from You and Leung [265]. We also prove that there exists a particular value of the composition timestep that minimizes the total numerical error. Finally, this method is parallelizable in time, which leads to a much lower computational time, given enough resources.

The new method of composition in a nutshell involves three components. We first

divide the temporal domain into smaller intervals and compute a modified flow map by solving Eq. (2.8) taking into account the boundary conditions over each of these intervals. These flow maps are mutually independent and hence can be computed in parallel. Second, we compose these independent (modified) flow maps together through appropriate composition (interpolation) operators to yield the flow map over the entire time interval, similar to the work of [24; 220; 177]. Last, we compose this modified flow map with the initial condition for the tracer field to yield the advected tracer field. There are multiple advantages to this method: (i) it can be readily parallelized in the temporal direction as the flow map computations are mutually independent, (ii) as we simply solve Eq. (2.8), the computation can be performed on any spatial discretization with any existing solvers or numerical toolboxes, (iii) due to the Eulerian nature of the computations, we can easily obtain error estimates, and (iv) as the flow maps are independently computed and then composed, the truncation errors are not compounded in time and we obtain an accuracy comparable to that of the particle methods.

3.2 Problem Statement

We now formally define the problem at hand using the notation from Table 2.1. Our goal is to numerically compute the tracer field $\alpha(\boldsymbol{x}, t)$ for all $0 \leq t \leq T$ and $\boldsymbol{x} \in \Omega$, with minimum practical numerical error, by solving Eq. (2.8) over a discretized spatio-temporal domain. For error analysis, we assume the domain to be homogeneous in all spatial directions, and discretize it into $N_{\boldsymbol{x}}$ distinct nodes, with a spacing of Δx along each direction without loss of generality.

3.3 Method of Composition for Tracer Advection

This section derives and discusses the method of composition to compute tracer advection, defined by Eq. (2.8), as well as the handling of tracer diffusion, reaction, and source terms, and the imposition of generic boundary conditions.

3.3.1 PDE-Based Flow Map Computation

Instead of devising numerical schemes to solve for Eq. (2.8), we take advantage of the flow map. Utilizing the Lagrangian interpretation of tracer transport from Sec. 2.4.4, we know that a fluid parcel starting from the location $\boldsymbol{x}_0$ carries with it the tracer value $\alpha_0(\boldsymbol{x}_0)$. The position of this fluid parcel (and thus the tracer value $\alpha_0(\boldsymbol{x}_0)$) at time t is given by $\boldsymbol{x} = \boldsymbol{\phi}_0^t(\boldsymbol{x}_0)$. From this, we get Eq. (3.1) (same as Eq. (2.31); restated for convenience):

$$\alpha(\boldsymbol{x}, t) = \alpha(\boldsymbol{\phi}_0^t(\boldsymbol{x}_0), t) = \alpha_0(\boldsymbol{x}_0) = \alpha_0\left(\boldsymbol{\phi}_t^0(\boldsymbol{x})\right) \;\; \forall \; t \in [0, T]\,. \tag{3.1}$$

However, note that, this is valid for any Lipschitz continuous initial condition field α_0. Specifically, we choose the initial condition given by $\alpha_0(\boldsymbol{x}) = \boldsymbol{x} \;\; \forall \; \boldsymbol{x} \in \Omega$. This then implies that:

$$\alpha(\boldsymbol{x}, t) = \alpha_0\left(\boldsymbol{\phi}_t^0(\boldsymbol{x})\right) = \boldsymbol{\phi}_t^0(\boldsymbol{x}) \;\; \forall \; \boldsymbol{x} \in \Omega \text{ and } \forall \; t \in [0, T]\,. \tag{3.2}$$

Thus, the backward flow map of of the underlying velocity field over the time interval $[0, T]$ can be obtained by solving the following PDE system forward in time with the given initial condition [156; 266]:

$$\begin{aligned} &\frac{\partial \alpha(\boldsymbol{x}, t)}{\partial t} + \boldsymbol{v}(\boldsymbol{x}, t) \cdot \nabla \alpha(\boldsymbol{x}, t) = 0\,, \\ &\alpha(\boldsymbol{x}, 0) = \boldsymbol{x}\,, \\ \text{then} \;\; &\alpha(\boldsymbol{x}, T) = \boldsymbol{\phi}_T^0(\boldsymbol{x})\,. \end{aligned} \tag{3.3}$$

Similarly, the forward flow map ($\boldsymbol{\phi}_0^T(\boldsymbol{x})$) can be obtained by solving the following PDE backward in time, with the stated terminal condition:

$$\begin{aligned} &\frac{\partial \alpha(\boldsymbol{x}, t)}{\partial t} + \boldsymbol{v}(\boldsymbol{x}, t) \cdot \nabla \alpha(\boldsymbol{x}, t) = 0\,, \\ &\alpha(\boldsymbol{x}, T) = \boldsymbol{x}\,, \\ \text{then} \; &\alpha(\boldsymbol{x}, 0) = \boldsymbol{\phi}_0^T(\boldsymbol{x})\,. \end{aligned} \tag{3.4}$$

Thus, by simply solving Eq. (2.8) with the initial condition $\alpha_0(\boldsymbol{x}) = \boldsymbol{x}$, we can compute the backward flow map $\boldsymbol{\phi}_T^0$ (Eq. (3.3)). Similarly, by solving Eq. (2.8) backward in time with the terminal condition given by $\alpha(\boldsymbol{x}, T) = \boldsymbol{x}$, we can compute the forward flow map $\boldsymbol{\phi}_0^T$. We refer to this as the *'PDE-based flow map computation'* [156; 157]. Without loss of generality, we only deal with the specifics of the backward flow map in this chapter. The corresponding conclusions hold true also for the forward flow map, and can be derived by simply flipping the temporal index.

It must be noted that our definition of the flow map is a slight extension of the classical flow maps from dynamical systems theory (Chapter 2). This is because we look at the flow maps as solutions of advective transport PDEs with specific initial conditions, and hence have the ability to impose specific boundary conditions. However, we will still use the term 'flow map' to refer to such flow maps that include the imposition of boundary conditions.

3.3.2 PDE-Based Flow Map Composition

Computing flow maps over longer time intervals may be expensive, especially without the possibility of parallelization. Further as the governing equation is hyperbolic, the flow map computation is susceptible to compounding diffusive and dispersive numerical errors. To alleviate these problems, we utilize and build upon the idea first proposed by [24] and later used by [266] to efficiently compute the flow map, as described in Sec. 2.4.3. Instead of computing the flow map over the entire duration, the interval is broken up into multiple smaller intervals of the order of a numerical timestep, and the individual flow maps (over these smaller intervals) are computed independently of each other. Finally, flow maps are composed to obtain the flow map over the larger interval.

Let us assume that the time interval $[0, T]$ is broken up into N_c distinct flow map intervals (of length $\Delta t_c = T/N_c$ each), and the flow map is independently computed over each of these intervals. Further, Δt_c is assumed an integral multiple of the discrete timestep Δt, *i.e.* $\Delta t_c = M\Delta t$. Then, the flow map over the entire duration

is given by,

$$\boldsymbol{\phi}_{N_c}^0(\boldsymbol{x}) = \boldsymbol{\phi}_1^0 \circ \boldsymbol{\phi}_2^1 \circ \ldots \circ \boldsymbol{\phi}_{N_c-1}^{N_c-2} \circ \boldsymbol{\phi}_{N_c}^{N_c-1}(\boldsymbol{x})\,, \tag{3.5}$$

where $\boldsymbol{\phi}_{i+1}^i = \boldsymbol{\phi}_{(i+1)\Delta t_c}^{(i)\Delta t_c}$. Each of these individual flow maps is computed by using Eq. (3.3). Numerically, once the flow maps over the individual timesteps are computed, interpolation is required to compose the new flow map at the positions given by the previous (total) flow map. We denote the interpolation operator associated with the flow map $\boldsymbol{\phi}_{i+1}^i$ by $\mathcal{I}_{i+1}^i$. Note that this interpolation operator is specific to the flow map, and does not change with the argument of the flow map: hence it needs to be computed only once (per flow map), see also Sec. 3.4.3. With the inclusion of this interpolation operator, Eq. (3.5) is transformed into Eq. (3.6):

$$\boldsymbol{\phi}_{N_c}^0(\boldsymbol{x}) = \mathcal{I}_1^0\boldsymbol{\phi}_1^0 \circ \mathcal{I}_2^1\boldsymbol{\phi}_2^1 \circ \ldots \circ \mathcal{I}_{N_c-1}^{N_c-2}\boldsymbol{\phi}_{N_c-1}^{N_c-2} \circ \boldsymbol{\phi}_{N_c}^{N_c-1}(\boldsymbol{x})\,. \tag{3.6}$$

While computing the flow maps using this method, each individual computation can be readily parallelized. Further, the method also allows the very efficient computation of $\boldsymbol{\phi}_{t+\tau}^0$ for any τ given $\boldsymbol{\phi}_T^0$, as we can use

$$\boldsymbol{\phi}_{t+\tau}^0(\boldsymbol{x}) = \mathcal{I}_t^0\boldsymbol{\phi}_t^0 \circ \boldsymbol{\phi}_{t+\tau}^t(\boldsymbol{x}) \tag{3.7}$$

Finally, the application of general boundary conditions is straightforward, as will be discussed in the following section.

To summarize, the solution of Eq. (2.8) is computed by solving Eq. (3.8),

$$\alpha(\boldsymbol{x}, T) = \alpha(\boldsymbol{x}, N_c\Delta t_c) = \mathcal{I}_\alpha\alpha_0 \circ \mathcal{I}_1^0\boldsymbol{\phi}_1^0 \circ \mathcal{I}_2^1\boldsymbol{\phi}_2^1 \circ \ldots \circ \mathcal{I}_{N_c-1}^{N_c-2}\boldsymbol{\phi}_{N_c-1}^{N_c-2} \circ \boldsymbol{\phi}_{N_c}^{N_c-1}(\boldsymbol{x})\,, \tag{3.8}$$

where $\mathcal{I}_\alpha$ is the interpolation operator for the initial tracer field $\alpha_0(\boldsymbol{x})$. The individual flow maps are computed according to Eq. (3.3), using a numerical timestep of Δt and appropriate boundary conditions. The methodology is schematized in Fig. (3-1).

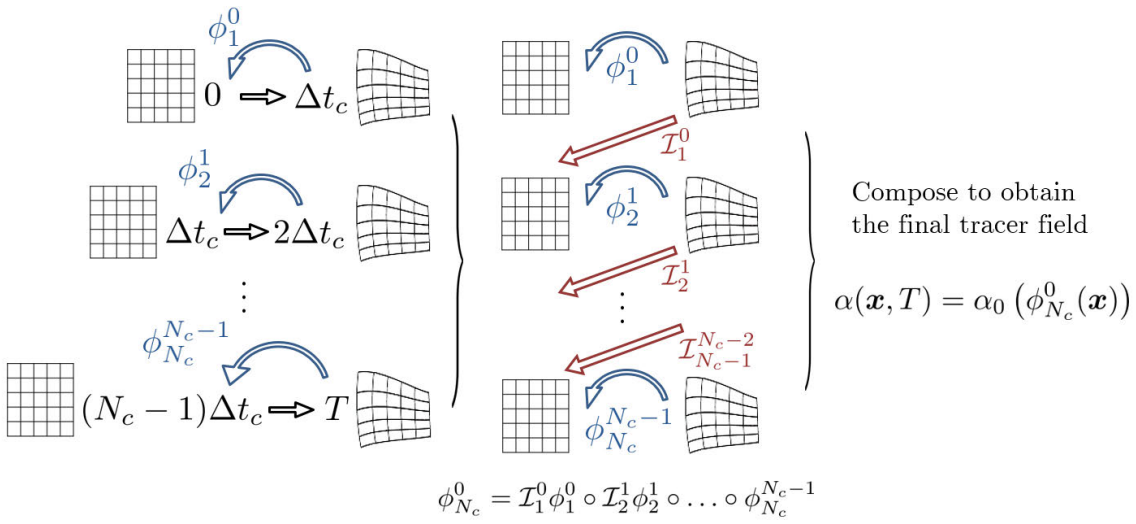


Figure 3-1: Composition-based advection methodology. The first step involves computing the individual flow maps (while accounting for boundary conditions), which can be done in parallel. The second step then involves computing the cumulative flow map over the entire time interval by sequentially interpolating and composing the individual flow maps. The final step involves composing this cumulative flow map with the initial tracer field to obtain the final tracer field. We note that the methodology is schematized for $\mathbb{R}^2$ but it generalizes to $\mathbb{R}^n$ as it is agnostic to the spatial dimension.

3.3.3 Inclusion of Diffusion, Sources, and Reactions Terms

We now incorporate the effects of tracer diffusion, sources, sinks, and reactions while computing the advective contribution through the method of composition. We utilize the commonly used technique of operator splitting. Splitting methods include Lie splitting [170] and Strang splitting [235], which we shall discuss. As will be seen, the Lie splitting introduces a first-order splitting error while Strang splitting introduces a second-order splitting error. However, in certain situations, these operator splitting methods can either yield a higher order of accuracy or even be exact and introduce no splitting error [140].

Let us consider the general form of advection-diffusion-reaction Eq. (3.9) which is an extension of Eq. (2.8),

$$\frac{\partial \alpha(\boldsymbol{x},t)}{\partial t} + \boldsymbol{v}\cdot\nabla\alpha(\boldsymbol{x},t) = \nabla\cdot(\kappa(\boldsymbol{x},t)\nabla\alpha(\boldsymbol{x},t)) + S_\alpha(\alpha(\boldsymbol{x},t),\boldsymbol{x},t) \quad \text{with} \quad \alpha(\boldsymbol{x},0) = \alpha_0(\boldsymbol{x})\,. \tag{3.9}$$

where $\kappa(\boldsymbol{x},t)$ is the diffusivity of the tracer and $S_\alpha(\alpha(x,t),\boldsymbol{x},t)$ serves to model the

reaction terms (if any) and the source/sink contributions (if any). We shall refer to $S_\alpha(\alpha(x,t),\boldsymbol{x},t)$ as the 'source term', keeping in mind the fact that it may very well represent any of the aforementioned phenomena.

In our methodology, we solve for the advection contribution using flow map composition. The source contribution is computed explicitly or implicitly for a first-order scheme, or integrated over one time stage or timestep for a higher-order time marching scheme. This is followed by solving for the diffusion implicitly. An implicit solve for the diffusion operator is often preferred as it is stiff and imposes a strict CFL condition. We will denote this resulting implicit diffusion operator by $\mathcal{D}$.

If using the Lie operator splitting, the split equations are given by Eq. (3.10):

$$\begin{aligned} \alpha_* &= \alpha(\boldsymbol{\phi}^t_{t+\Delta t}(\boldsymbol{x}),t)\,, \\ \alpha_{**} &= S_\alpha(\alpha_*,\boldsymbol{x},t)\Delta t\,, \\ \alpha(\boldsymbol{x},t+\Delta t) &= \mathcal{D}\left(\alpha_* + \alpha_{**}\right)\,. \end{aligned} \tag{3.10}$$

The BCs are incorporated in the advection contribution and the diffusion contribution, according to the corresponding fluxes (see Sec. 3.3.4 next). The possible ill-posedness of the diffusion operator (under zero Neumann boundary conditions everywhere, for example) can be eliminated by any of the established singularity removal techniques [168; 247]. It can further be shown that the leading order truncation error term in the splitting error is given by Eq. (3.11) [234]:

$$\mathcal{E}_{\text{lie}} = \frac{\Delta t}{2}\left(\nabla\cdot\boldsymbol{v}\left(S_\alpha - \frac{\partial S_\alpha}{\partial\alpha}\alpha\right) + \nabla\cdot(\kappa\nabla S_\alpha) - \frac{\partial S_\alpha}{\partial\alpha}\left(\nabla\cdot(\kappa\nabla\alpha)\right)\right) \tag{3.11}$$

Thus, the Lie splitting is second-order accurate if the (i) source term is absent (*i.e.* an advection–diffusion equation), (ii) diffusion is absent (*i.e.* an advection–reaction equation) and the velocity field $\boldsymbol{v}$ is divergence free or the source term is linear in α.

One can also use the second-order Strang splitting method given by Eq. (3.12),

$$
\begin{aligned}
\alpha_* &= \alpha(\boldsymbol{\phi}^t_{t+\frac{\Delta t}{2}}(\boldsymbol{x}), t)\,, \\
\alpha_{**} &= S_\alpha(\alpha_*, \boldsymbol{x}, t)\frac{\Delta t}{2}\,, \\
\alpha' &= \mathcal{D}\left(\alpha_* + \alpha_{**}\right)\,, \\
\alpha'_* &= \alpha'(\boldsymbol{\phi}^{t+\frac{\Delta t}{2}}_{t+\Delta t}(\boldsymbol{x}))\,, \\
\alpha &= S_\alpha\left(\alpha'_*, \boldsymbol{x}, t + \frac{\Delta t}{2}\right)\frac{\Delta t}{2}\,.
\end{aligned}
\tag{3.12}
$$

Similar to the Lie splitting, the boundary conditions are accounted separately in the advection and the diffusion computation. Note that we chose the order of operations in the splitting to ensure that the implicit diffusion solve is only carried out once for computational efficiency. However, this sequence is not unique, and maybe altered depending on the specific application. For specific forms of the source term, it is also possible to absorb it in the advection / diffusion operators and in such cases the computation would be appropriately altered.

Even though the Strang splitting is second-order accurate in general, there are specific cases under which it is exact and no splitting error is introduced. We borrow the results from Lanser and Verwer [140] who detail the sufficient conditions for the absence of the splitting error. They prove that for Eq. (3.9) with Strang splitting, no splitting error exists if (i) S_α is at most linear in α and independent of $\boldsymbol{x}$, and (ii) $\boldsymbol{v}$ and κ are independent of $\boldsymbol{x}$. Note that this is rarely possible in realistic cases, and hence there will always be a splitting error introduced. Lanser and Verwer [140] detail several ingenious approaches to split Eq. (3.9) based on the specific nature of various terms involved to minimize the splitting error. Although higher-order splitting methods are possible [121], they are rarely used in practice due to the computational overhead and complexity of implementation. Further, it is typically observed that the temporal / spatial errors in the computation of the individual terms (*i.e.* advection, diffusion, and reaction) often dominate the splitting errors, which further curbs the need to go to higher-order accurate splitting methods [249].

In our examples, we solve an advection–diffusion equation in Sec. 4.2.1 and an advection–reaction equation in Sec. 4.2.2. We use the Lie splitting method for both. For the first case, the source term is absent, and, for the second, diffusion is absent and the velocity field is divergence free (as it is obtained from an incompressible flow simulation). Thus Lie splitting is second-order accurate in both cases.

3.3.4 Implementation of Boundary Conditions

As Eq. (2.8) and Eq. (3.9) are PDEs, boundary conditions (BCs) are typically required to solve them, especially when inlets and/or outlets are present.

Previous works that compute PDE-based flow maps [156; 157] solve Eq. (3.3) which essentially amounts to Eq. (2.8) for a very specific initial condition, *i.e.* $\alpha_0(\boldsymbol{x}) = \boldsymbol{x}$. The examples studied either involve closed domains (such as the analytical double gyre flow) or open domains with a very specific set of BCs, stating that either the flow map field ($\boldsymbol{\phi}$) is free to leave the domain or, when new flow map positions are to enter, they bear the value equal to the value at the position at the initial time. These BCs, well-posed for the flow map problem, are summarized in Eq. (3.13),

$$\frac{\partial \boldsymbol{\phi}(\boldsymbol{x},t)}{\partial \hat{\mathbf{n}}_{\partial\Omega}} = 0 \quad \text{if} \quad \hat{\mathbf{n}}_{\partial\Omega} \cdot \boldsymbol{v} > 0 \quad \text{and} \quad \boldsymbol{\phi}(\boldsymbol{x},t) = \boldsymbol{\phi}(\boldsymbol{x},0) \quad \text{if} \quad \hat{\mathbf{n}}_{\partial\Omega} \cdot \boldsymbol{v} < 0\,. \tag{3.13}$$

For Eq. (3.9) and general tracer advection-diffusion-reaction problems, these BCs are however not sufficient as one may have time-dependent inlets, outlets with varying strengths, specified tracer advection-diffusion fluxes, or more complex conditions. The nature of the BCs may itself also locally change in time. None of these possibilities are accounted for in Eq. (3.13). More general BCs are thus needed.

For tracer advection-reaction without diffusion, Eq. (3.9) becomes hyperbolic as Eq. (2.8) and Eq. (3.3). Either a single Dirichlet or Neumann BC is then commonly imposed at boundaries. For open boundaries, due to the hyperbolic nature, if the velocity vector at the boundary is outwards then no BCs are necessary, as the tracer then is simply advected outside the domain. However, BCs are commonly provided when the velocity vector points into the domain. In some cases, BCs may be irrelevant

even for open domains and the equations can be exactly solved without BCs [141; 67]. This occurs for example when the whole domain of computation, e.g. the whole fluid, is deformed according to the advection process itself.

Next, we discuss the generic implementation of Dirichlet and Neumann type BCs, noting that results extend to other well-posed conditions. We discuss these BCs when the velocity vector points inwards, but the treatment extends to the outward BCs when they are needed.

Dirichlet Boundary Condition

Let us assume that the BC for Eq. (2.8) consists of provided tracer values, given by Eq. (3.14),

$$\alpha(\boldsymbol{x}, t) = \alpha_{BC}(\boldsymbol{x}, t) \quad \text{for } \boldsymbol{x} \in \partial\Omega \text{ such that } \boldsymbol{v} \cdot \hat{\mathbf{n}}_{\partial\Omega} < 0\,. \tag{3.14}$$

For regular PDE-based advection, such BCs are typically imposed by using ghost cells, *i.e.* numerical cells at the domain boundaries with the tracer value set equal to $\alpha_{BC}(\boldsymbol{x}, t)$ [159]. Such a setup makes the resulting discretized equation well-posed. The addition of the diffusion operator can also be handled in a similar way. Through operator splitting, we split the advection-diffusion equation into a hyperbolic contribution (advection part) and a parabolic contribution (diffusion part). BCs for the hyperbolic part are handled as described before. As diffusion is a stiff operator, we solve the parabolic part implicitly. Dirichlet boundary conditions are readily incorporated in the RHS of the resulting linear system by again considering ghost cells at the boundary with known tracer value $\alpha_{BC}(\boldsymbol{x}, t)$ [141].

For the method of composition, we solve the transport PDE to compute $\boldsymbol{\phi}(\boldsymbol{x}, t)$, not $\alpha(\boldsymbol{x}, t)$. Thus, we need to reformulate the BCs given by Eq. (3.14) into BCs that make Eq. (3.3) well-posed. This is achieved by the using *marker positions*. The marker positions are defined as fictitious position values that do not lie in the domain, and are exclusively utilized to impose boundary values on the flow map computation, *i.e.* on Eq. (3.3). These marker positions simply keep track of where a particular

position entered the domain from, and what its corresponding tracer value was. Thus, the total number of distinct marker positions required is equal to the number of distinct discrete Dirichlet BC values throughout the entire spatio-temporal domain of interest. Let us denote the marker position corresponding to a location $\boldsymbol{x}$ on $\partial\Omega$ at time t by $\boldsymbol{x}_{BC}(t)$. Note that $\boldsymbol{x}_{BC}(t) \notin \Omega$. We also define composition operations over these marker positions through Eq. (3.15),

$$\alpha_0\left(\boldsymbol{x}_{BC}(t)\right) \triangleq \alpha_{BC}(\boldsymbol{x}, t) \quad \text{and} \quad \boldsymbol{\phi}^i_{i+1}\left(\boldsymbol{x}_{BC}(t)\right) \triangleq \boldsymbol{x}_{BC}(t)\,. \tag{3.15}$$

The numerical solution of Eq. (2.8) with Dirichlet BCs starts by solving the following system of PDEs,

$$\begin{aligned} &\frac{\partial \boldsymbol{\phi}^i_{i+1}(\boldsymbol{x}, t)}{\partial t} + \boldsymbol{v}(\boldsymbol{x}, t) \cdot \nabla \boldsymbol{\phi}^i_{i+1}(\boldsymbol{x}, t) = 0\,, \\ &\boldsymbol{\phi}^i_{i+1}(\boldsymbol{x}, t_i) = \boldsymbol{x} \quad \text{and} \quad \boldsymbol{\phi}^i_{i+1}(\boldsymbol{x}, t) = \boldsymbol{x}_{BC}(t) \;\forall\; \boldsymbol{x} \in \partial\Omega\,, \end{aligned} \tag{3.16}$$

which provides the set of individual flow maps. The addition of the diffusion operator is handled similarly, as described above, by reformulating the resulting linear system to include these BCs in the RHS by using ghost cells. The composition step Eq. (3.8) is then completed, utilizing the composition operation over the marker positions $\boldsymbol{x}_{BC}$, defined by Eq. (3.15).

Neumann Boundary Condition

We now discuss the implementation of the Neumann BC, given by Eq. (3.17),

$$\frac{\partial \alpha(\boldsymbol{x}, t)}{\partial \hat{\mathbf{n}}_{\partial\Omega}} = k_\alpha(\boldsymbol{x}, t) \quad \text{for } \boldsymbol{x} \in \partial\Omega \text{ such that } \boldsymbol{v} \cdot \hat{\mathbf{n}}_{\partial\Omega} < 0\,. \tag{3.17}$$

Although we discuss BCs on the first derivative, results directly extend to higher-order BCs and Robin BCs. Neumann BCs are implemented by suitably modifying the problem such that they can be reduced to special Dirichlet BCs for the individual

flow maps. For the advection PDE, we write Eq. (2.8) along the boundary as follows:

$$\frac{\partial \alpha(\boldsymbol{x},t)}{\partial t} + (\boldsymbol{v}_{\hat{\mathbf{n}}}(\boldsymbol{x},t), \boldsymbol{v}_{\perp}(\boldsymbol{x},t)) \cdot (\nabla \alpha_{\hat{\mathbf{n}}}(\boldsymbol{x},t), \nabla \alpha_{\perp}(\boldsymbol{x},t)) = 0\,, \tag{3.18}$$

where the subscript $\hat{\mathbf{n}}$ indicates gradients along the normal direction to the boundary and the subscript $\perp$ gradients along the basis components of the hyperplane orthogonal to $\hat{\mathbf{n}}$. Note that $\hat{\mathbf{n}}$ and the basis of its orthogonal hyperplane can always be uniquely expressed in terms of the grid coordinates. Substituting Eq. (3.17) in Eq. (3.18), we obtain:

$$\frac{\partial \alpha(\boldsymbol{x},t)}{\partial t} + \boldsymbol{v}(\boldsymbol{x},t)_{\perp} \cdot \nabla \alpha(\boldsymbol{x},t)_{\perp} = -\boldsymbol{v}_{\hat{\mathbf{n}}}(\boldsymbol{x},t) k_{\alpha}(\boldsymbol{x},t)\,. \tag{3.19}$$

Note that Eq. (3.19) is similar to Eq. (3.9) with $\kappa = 0$ (see Sec. 3.3.3), simply one dimension lower, with Dirichlet BCs over the co-dimension one hyperplane locally orthogonal to the boundary. Thus, Eq. (3.19) can be solved by the combination of schemes already presented in Sec. 3.3.3 and Sec. 3.3.4.

For the advection-diffusion PDE, the BC is implemented similarly to Sec. 3.3.4. We first split the PDE into a hyperbolic part (solved explicitly) and a parabolic part (solved implicitly). Neumann BCs can be applied as described above to the hyperbolic part. For the parabolic part, we can accommodate the BCs again through ghost cells, where the ghost cell values are implicitly incorporated by adding local numerical approximations of the derivative to the linear system.

Computationally, another approach to impose Neumann BC is as follows. First, according to the numerical stencil used to discretize $\frac{\partial \alpha(\boldsymbol{x},t)}{\partial \hat{\mathbf{n}}_{\partial\Omega}}$, the boundary values of α can be explicitly computed in terms of the interior values by solving a local linear system resulting from the BC discretization. Once the boundary values to be advected into the domain are computed, they can be imposed as Dirichlet BCs, as described above. Such numerical Neumann BCs is then completed for each of the individual time duration independently.

Imposition of Neumann BCs for the individual flow maps require special attention at the composition/interpolation step. Specifically, the order of the numerical inter-

polation scheme must be equal to or higher than the order of the derivative involved in the BC. This ensures that the error introduced in the composition of the BC is at least one order higher than the error in the discretization of the BC itself, and the dominant order of accuracy thus remains unchanged.

3.4 Numerical Properties

This section derives numerical properties of the computation of advective transport, *i.e.* Eq. (2.8), using the method of flow map composition. First, total error bounds are derived. This is followed by a discussion on the existence and derivation of the 'optimal' composition timestep value, for which the net error is minimized.

3.4.1 Error Estimates

We derive error bounds for the computation of the flow map using the method of composition in terms of the spatial discretization ($\Delta \boldsymbol{x}$) and temporal discretization (Δt). We assume that the accuracy of the velocity gradient computation is $\mathcal{O}(\Delta \boldsymbol{x}^\beta)$ and that of all interpolation operations are $\mathcal{O}(\Delta \boldsymbol{x}^\gamma)$. The time marching scheme for the advection computation is assumed to have a local truncation error of order $\mathcal{O}(\Delta t^\theta)$. The numerically computed value of a quantity $\bullet$ is denoted by $\tilde{\bullet}$.

The proof extends the results of You and Leung [265] who show that the composition of the flow map for a dynamical system is second-order accurate, as their advection scheme, if the interpolation is at least second-order accurate. Inspired by classic initial value problem error analyses [241; 30], we generalize this result to the advection of any passive tracer and prove that the advection computation has a temporal accuracy equal to that of the time marching scheme used, and has a spatial accuracy at least equal to the minimum amongst the orders of accuracies of the advection and interpolation operations, regardless of the specific numerical schemes and orders of integration employed.

The exact value of the tracer field at time T is given by $\alpha(\boldsymbol{x}, T) = \alpha_0(\boldsymbol{\phi}^0_{N_c}(\boldsymbol{x}))$, whereas the computed solution is given by $\mathcal{I}_\alpha \alpha_0(\tilde{\boldsymbol{\phi}}^0_{N_c}(\boldsymbol{x}))$. The total global error

(omitting the $\boldsymbol{x}$ dependency for brevity) is then:

$$E = \left|\mathcal{I}_\alpha \alpha_0(\tilde{\boldsymbol{\phi}}^0_{N_c}) - \alpha_0\left(\boldsymbol{\phi}^0_{N_c}\right)\right| = \left|\mathcal{I}_\alpha \alpha_0\left(\mathcal{I}^0_1 \tilde{\boldsymbol{\phi}}^0_1\left(\tilde{\boldsymbol{\phi}}^1_{N_c}\right)\right) - \alpha_0\left(\boldsymbol{\phi}^0_1\left(\boldsymbol{\phi}^1_{N_c}\right)\right)\right| . \tag{3.20}$$

Adding and subtracting $\alpha_0\left(\mathcal{I}^0_1 \tilde{\boldsymbol{\phi}}^0_1\left(\tilde{\boldsymbol{\phi}}^1_{N_c}\right)\right)$ and using the triangle inequality, we get:

$$\begin{aligned} E \leq & \left|\mathcal{I}_\alpha \alpha_0\left(\mathcal{I}^0_1 \tilde{\boldsymbol{\phi}}^0_1\left(\tilde{\boldsymbol{\phi}}^1_{N_c}\right)\right) - \alpha_0\left(\mathcal{I}^0_1 \tilde{\boldsymbol{\phi}}^0_1\left(\tilde{\boldsymbol{\phi}}^1_{N_c}\right)\right)\right| \\ & + \left|\alpha_0\left(\mathcal{I}^0_1 \tilde{\boldsymbol{\phi}}^0_1\left(\tilde{\boldsymbol{\phi}}^1_{N_c}\right)\right) - \alpha_0\left(\boldsymbol{\phi}^0_1\left(\boldsymbol{\phi}^1_{N_c}\right)\right)\right| . \end{aligned} \tag{3.21}$$

Further simplifying the notation, we henceforth omit the temporal indices of interpolation operators (since each interpolation corresponds to the flow map following it). We denote the exact tracer advection over $[0, t_i]$ by $\alpha_i(\bullet) = \alpha_0\left(\boldsymbol{\phi}^0_i(\bullet)\right)$ and the exact backward propagation to $t = 0$ of errors made beyond $t = t_i$ by $\mathcal{E}_i = \left|\alpha_i\left(\tilde{\bullet}\right) - \alpha_i(\bullet)\right|$. The Eq. (3.21) can then be rewritten,

$$E \leq \left|\mathcal{I}_\alpha \alpha_0\left(\mathcal{I}\tilde{\boldsymbol{\phi}}^0_1\left(\tilde{\boldsymbol{\phi}}^1_{N_c}\right)\right) - \alpha_0\left(\mathcal{I}\tilde{\boldsymbol{\phi}}^0_1\left(\tilde{\boldsymbol{\phi}}^1_{N_c}\right)\right)\right| + \mathcal{E}_0 , \tag{3.22}$$

where the first norm is the error due to the interpolation of the tracer field at $t = 0$ and the second is the net error $\mathcal{E}_0$ due to the numerical backward flow map integration over $[T, 0]$. As the interpolations are of order γ, we can further bound E by:

$$E \leq C_{I_\alpha} \Delta x^\gamma + \mathcal{E}_0 , \tag{3.23}$$

where C_{I_α} is a constant independent of Δx and Δt. Next, we expand $\mathcal{E}_0$ to link $\mathcal{E}_0$ to $\mathcal{E}_1$, using the fact that, $\forall i = 1, \cdots, N_c$, $\alpha_i(\bullet) = \alpha_0\left(\boldsymbol{\phi}^0_i(\bullet)\right) = \alpha_0\left(\boldsymbol{\phi}^0_{i-1}\left(\boldsymbol{\phi}^{i-1}_i(\bullet)\right)\right) = \alpha_{i-1}\left(\boldsymbol{\phi}^{i-1}_i(\bullet)\right)$, thus, $\mathcal{E}_i = \left|\alpha_{i-1}\left(\boldsymbol{\phi}^{i-1}_i(\tilde{\bullet})\right) - \alpha_{i-1}\left(\boldsymbol{\phi}^{i-1}_i(\bullet)\right)\right|$. We obtain Eq. (3.24),

$$\mathcal{E}_0 \leq \left|\alpha_0\left(\mathcal{I}\tilde{\boldsymbol{\phi}}^0_1\left(\tilde{\boldsymbol{\phi}}^1_{N_c}\right)\right) - \alpha_0\left(\mathcal{I}\boldsymbol{\phi}^0_1\left(\tilde{\boldsymbol{\phi}}^1_{N_c}\right)\right)\right| + \left|\alpha_0\left(\mathcal{I}\boldsymbol{\phi}^0_1\left(\tilde{\boldsymbol{\phi}}^1_{N_c}\right)\right) - \alpha_0\left(\boldsymbol{\phi}^0_1\left(\tilde{\boldsymbol{\phi}}^1_{N_c}\right)\right)\right| + \mathcal{E}_1 , \tag{3.24}$$

where the first norm is an error due to the numerical flow map integration over $[0, t_1]$, the second is an interpolation error for the flow map over $[0, t_1]$, and the third is the

net error $\mathcal{E}_1$ due to the numerical backward flow map integration over $[T, t_1]$. Since α_0 is Lipschitz continuous of constant $\mathcal{L}_\alpha^0$ (see Sec. 3.2),

$$\mathcal{E}_0 \leq \mathcal{L}_\alpha^0 \left| \mathcal{I}\tilde{\boldsymbol{\phi}}_1^0 \left(\tilde{\boldsymbol{\phi}}_{N_c}^1 \right) - \mathcal{I}\boldsymbol{\phi}_1^0 \left(\tilde{\boldsymbol{\phi}}_{N_c}^1 \right) \right| + \mathcal{L}_\alpha^0 \left| \mathcal{I}\boldsymbol{\phi}_1^0 \left(\tilde{\boldsymbol{\phi}}_{N_c}^1 \right) - \boldsymbol{\phi}_1^0 \left(\tilde{\boldsymbol{\phi}}_{N_c}^1 \right) \right| + \mathcal{E}_1 \,. \tag{3.25}$$

Let us assume that the norm of each discrete interpolation operator, $|\mathcal{I}|$, is bounded by C_I independent of Δx. The dominant numerical advection error in the individual flow map computation can be asymptotically written as $M\left(C_{AX}\Delta x^\beta + C_{AT}\Delta t^\theta\right)$, where C_{AX} and C_{AT} are non-negative constants independent of Δx and Δt, and of C_I. As a result, we obtain for the first term in Eq. (3.25),

$$\left| \mathcal{I}\tilde{\boldsymbol{\phi}}_1^0 \left(\tilde{\boldsymbol{\phi}}_{N_c}^1 \right) - \mathcal{I}\boldsymbol{\phi}_1^0 \left(\tilde{\boldsymbol{\phi}}_{N_c}^1 \right) \right| \leq C_I \max \left| \tilde{\boldsymbol{\phi}}_1^0 - \boldsymbol{\phi}_1^0 \right| \leq C_I M \left(C_{AX}\Delta x^\beta + C_{AT}\Delta t^\theta \right) \,. \tag{3.26}$$

For the second term in Eq. (3.25), as the interpolation is of order γ, we obtain:

$$\left| \mathcal{I}\boldsymbol{\phi}_1^0 \left(\tilde{\boldsymbol{\phi}}_{N_c}^1 \right) - \boldsymbol{\phi}_1^0 \left(\tilde{\boldsymbol{\phi}}_{N_c}^1 \right) \right| \leq \Delta x^\gamma \max \left| \partial^\gamma \boldsymbol{\phi}_1^0 \right| \leq C_{I_{\boldsymbol{\phi}}} \Delta t_c \Delta x^\gamma e^{(2\gamma-1)\mathcal{L}_{\boldsymbol{v}}\Delta t_c} \,, \tag{3.27}$$

where the latter step is obtained by using *Lemma 5* from [265], and $C_{I_{\boldsymbol{\phi}}}$ is a constant independent of Δx and Δt. Finally, substituting Eq. (3.26) and Eq. (3.27) in Eq. (3.25) results in Eq. (3.28),

$$\mathcal{E}_0 \leq \mathcal{L}_\alpha^0 \left(C_I M \left(C_{AX}\Delta x^\beta + C_{AT}\Delta t^\theta \right) + C_{I_{\boldsymbol{\phi}}} \Delta t_c \Delta x^\gamma e^{(2\gamma-1)\mathcal{L}_{\boldsymbol{v}}\Delta t_c} \right) + \mathcal{E}_1 = \mathcal{L}_\alpha^0 \mathcal{E} + \mathcal{E}_1 \,, \tag{3.28}$$

where $\mathcal{E} = C_I M \left(C_{AX}\Delta x^\beta + C_{AT}\Delta t^\theta \right) + C_{I_{\boldsymbol{\phi}}} \Delta t_c \Delta x^\gamma e^{(2\gamma-1)\mathcal{L}_{\boldsymbol{v}}\Delta t_c}$. Finally, to obtain a recursive relation to compute $\mathcal{E}_0$, we need to compute the Lipschitz constant of α_i (denoted by $\mathcal{L}_\alpha^i$) in terms of the known quantities. *Lemma 4* from [265] provides a way to relate these quantities. It states that if the velocity field $\boldsymbol{v}$ has a Lipschitz constant $\mathcal{L}_{\boldsymbol{v}}$, then we have

$$\left| \boldsymbol{\phi}_N^0(\boldsymbol{x}_1) - \boldsymbol{\phi}_N^0(\boldsymbol{x}_2) \right| \leq e^{\mathcal{L}_{\boldsymbol{v}} N \Delta t} \left| \boldsymbol{x}_1 - \boldsymbol{x}_2 \right| \,. \tag{3.29}$$

The theorem stated in [265] deals with forward flow maps, however the analogous results hold true for backward flow maps as well. As a result, we have:

$$|\alpha_i(\boldsymbol{x}_1) - \alpha_i(\boldsymbol{x}_2)| \leq \mathcal{L}_\alpha^0 \left|\boldsymbol{\phi}_i^0(\boldsymbol{x}_1) - \boldsymbol{\phi}_i^0(\boldsymbol{x}_2)\right| \leq \mathcal{L}_\alpha^0 e^{\mathcal{L}_{\boldsymbol{v}} i\Delta t_c} |\boldsymbol{x}_1 - \boldsymbol{x}_2| \,. \tag{3.30}$$

Eq. (3.30) implies that the Lipschitz constant of α_i is $\mathcal{L}_\alpha^i = \mathcal{L}_\alpha^0 e^{\mathcal{L}_{\boldsymbol{v}} i\Delta t_c}$. Using Eq. (3.28), recursively substituting for $\mathcal{E}_i$ and using $\mathcal{L}_\alpha^i$, we obtain Eq. (3.31),

$$\mathcal{E}_0 \leq \mathcal{L}_\alpha^0 \mathcal{E} + \mathcal{E}_1 \leq \mathcal{E}\left(\mathcal{L}_\alpha^0 + \ldots + \mathcal{L}_\alpha^{N_c}\right) + \mathcal{E}_{N_c} \,. \tag{3.31}$$

However, $\mathcal{E}_{N_c} = \left|\alpha_0\left(\boldsymbol{\phi}_{N_c}^0\right) - \alpha_0\left(\boldsymbol{\phi}_{N_c}^0\right)\right| = 0$. Substituting for $\mathcal{L}_\alpha^i$ in Eq. (3.31):

$$\mathcal{E}_0 \leq \mathcal{E}\mathcal{L}_\alpha^0 \left(1 + \ldots + e^{\mathcal{L}_{\boldsymbol{v}} i\Delta t_c} + \ldots + e^{\mathcal{L}_{\boldsymbol{v}} N_c\Delta t_c}\right) \leq \mathcal{E}\mathcal{L}_\alpha^0 \frac{e^{\mathcal{L}_{\boldsymbol{v}} T} - 1}{e^{\mathcal{L}_{\boldsymbol{v}}\Delta t_c} - 1} \,. \tag{3.32}$$

Substituting Eq. (3.32) in Eq. (3.23), we obtain:

$$E \leq \mathcal{L}_\alpha^0 \frac{e^{\mathcal{L}_{\boldsymbol{v}} T} - 1}{e^{\mathcal{L}_{\boldsymbol{v}}\Delta t_c} - 1}\left(C_I M\left(C_{AX}\Delta x^\beta + C_{AT}\Delta t^\theta\right) + C_{I_{\boldsymbol{\phi}}}\Delta t_c \Delta x^\gamma e^{(2\gamma-1)\mathcal{L}_{\boldsymbol{v}}\Delta t_c}\right) + C_{I_\alpha}\Delta x^\gamma \,. \tag{3.33}$$

Note that, even though the first term on the right hand side has a non-polynomial dependence on Δt, the following relation always holds:

$$1 + \ldots + e^{\mathcal{L}_{\boldsymbol{v}} i\Delta t_c} + \ldots + e^{\mathcal{L}_{\boldsymbol{v}} N_c\Delta t_c} \leq N_c \max\left(1, e^{\mathcal{L}_{\boldsymbol{v}} T}\right) \,. \tag{3.34}$$

Substituting this in Eq. (3.33), we finally have:

$$\begin{aligned} E \leq \mathcal{L}_\alpha^0 \max\left(1, e^{\mathcal{L}_{\boldsymbol{v}} T}\right) \Bigg(& C_I\left(N_c M C_{AX}\Delta x^\beta + C_{AT} T \Delta t^{\theta-1}\right) \\ & + C_{I_{\boldsymbol{\phi}}} T \Delta x^\gamma e^{(2\gamma-1)\mathcal{L}_{\boldsymbol{v}}\Delta t_c}\Bigg) + C_{I_\alpha}\Delta x^\gamma \,. \end{aligned} \tag{3.35}$$

We note that this is the global error summed over all timesteps and not the instantaneous local error [46; 241]. At worst, there is an addition of errors. There is indeed no multiplicative compounding of errors. This leads to *'super-accurate'* properties

discussed in Section 3.4.3.

3.4.2 Optimal Composition Timestep

We note that the method of composition involves two basic operations, namely advection and interpolation. We now present an analysis that provides an optimal composition timestep, yielding minimum (or near minimum) total error. As the available velocity field is gridded, we have an already established Δx, and hence we are only concerned with finding an optimal Δt_c, henceforth denoted by Δt_{opt}.

The intuitive argument for the existence of the optimal composition timestep is as follows. First, each individual flow map advection (over M timesteps) introduces errors that depend on both Δx and Δt. Second, when composing this flow map with the flow map for the previous time interval, the interpolation operation introduces an error that depends only on Δx, as this operation is localized in time. Hence for fixed start and end times, as Δt and thus $\Delta t_c = M\Delta t$ decrease, the advection error decreases. However with a decreasing Δt_c more interpolations are performed, and the total interpolation error increases. Conversely, for a large Δt_c, the total advection error is high, but the interpolation error is low (as less number of interpolations are performed). Such an interplay between the advection and interpolation errors suggests the existence of an optimal Δt_c that minimizes the total error.

For the following analysis, we assume that $M = 1$ (*i.e.* $\Delta t_c = \Delta t$, $N_c = N_t$, and $\Delta t = T/N_t$). That is, the number of composition intervals is the same as the number of timesteps. Eq. (3.3) is thus solved for each timestep independently, and the resulting timestep flow maps composed with each other. The extension of this result is straightforward for $M > 1$.

We could start from the bounds provided by Eq. (3.26) and Eq. (3.27). However, we can obtain an estimate of the optimal timestep in closed form from a simpler analysis. We consider the errors in the computation of the tracer field during a single timestep. They are due to the numerical advection and interpolation errors, but for $\Delta t_c = \Delta t$, without compounding of errors within the flow-map computation. We assume that we are given an exact backward flow map $\boldsymbol{\phi}_{N_t}^1$ and consider the remaining

timestep $[0, t_1]$. The error then is:

$$E_{\Delta t} = \left| \mathcal{I}_\alpha \alpha_0 \left(\mathcal{I} \tilde{\boldsymbol{\phi}}_1^0 \left(\boldsymbol{\phi}_{N_t}^1 \right) \right) - \alpha_0 \left(\boldsymbol{\phi}_1^0 \left(\boldsymbol{\phi}_{N_t}^1 \right) \right) \right| . \tag{3.36}$$

Separating the interpolation error from the flow-map error and using triangle inequality leads to the bound:

$$E_{\Delta t} \leq \left| \mathcal{I}_\alpha \alpha_0 \left(\mathcal{I} \tilde{\boldsymbol{\phi}}_1^0 \left(\boldsymbol{\phi}_{N_t}^1 \right) \right) - \alpha_0 \left(\mathcal{I} \tilde{\boldsymbol{\phi}}_1^0 \left(\boldsymbol{\phi}_{N_t}^1 \right) \right) \right| + \left| \alpha_0 \left(\mathcal{I} \tilde{\boldsymbol{\phi}}_1^0 \left(\boldsymbol{\phi}_{N_t}^1 \right) \right) - \alpha_0 \left(\boldsymbol{\phi}_1^0 \left(\boldsymbol{\phi}_{N_t}^1 \right) \right) \right| .$$

As the order of interpolation is of order γ and α_0 is Lipschitz continuous, we obtain,

$$E_{\Delta t} \leq C_{I_\alpha} \Delta x^\gamma + \mathcal{L}_\alpha^0 \left| \mathcal{I} \tilde{\boldsymbol{\phi}}_1^0 \left(\boldsymbol{\phi}_{N_t}^1 \right) - \boldsymbol{\phi}_1^0 \left(\boldsymbol{\phi}_{N_t}^1 \right) \right| .$$

Next, as for Eq. (3.26), since the numerical advection is of order β in space and θ in time, the error in the computation of $\mathcal{I} \tilde{\boldsymbol{\phi}}_1^0 \left(\boldsymbol{\phi}_{N_t}^1 \right)$ can be written asymptotically as Δx and $\Delta t \to 0$,

$$\left| \mathcal{I} \tilde{\boldsymbol{\phi}}_1^0 \left(\boldsymbol{\phi}_{N_t}^1 \right) - \boldsymbol{\phi}_1^0 \left(\boldsymbol{\phi}_{N_t}^1 \right) \right| \approx \left| \mathcal{I} \tilde{\boldsymbol{\phi}}_1^0 \left(\boldsymbol{\phi}_{N_t}^1 \right) - \mathcal{I} \boldsymbol{\phi}_1^0 \left(\boldsymbol{\phi}_{N_t}^1 \right) \right| \approx C_{AX} \Delta x^\beta + C_{AT} \Delta t^\theta .$$

Hence, if we now only retain the lowest-order terms for Δx and Δt in the estimate for E, we obtain,

$$E_{\Delta t} \approx C_X \Delta x^\lambda + \mathcal{L}_\alpha^0 C_{AT} \Delta t^\theta . \tag{3.37}$$

where $\lambda = \min\{\gamma, \beta\}$ and $C_X = C_{I_\alpha X}$ if $\gamma < \beta$, $C_X = \mathcal{L}_\alpha^0 C_{AX}$ if $\gamma > \beta$, and $C_X = C_{I_\alpha X} + \mathcal{L}_\alpha^0 C_{AX}$ if $\gamma = \beta$. Eq. (3.37) yields the error bound for the computation of a *single timestep*, independent of the errors committed in the other timesteps. With the present method of composition, the order of the largest possible error is only the sum of these errors from the individual timesteps (individual composition steps when $\Delta t_c > \Delta t$), unlike the faster compounding of errors seen in common time integration schemes of ODEs and PDEs (e.g. exponential or polynomial growth). Hence, using

Eq. (3.37), we write the net error as:

$$E = N_t E_{\Delta t} \approx N_t \left(C_X \Delta x^\lambda + \mathcal{L}_\alpha C_{AT} \Delta t^\theta \right) ,$$

$$\Leftrightarrow \quad E \approx N_t \left(C_X \Delta x^\lambda + \frac{\mathcal{L}_\alpha^0 C_{AT} T^\theta}{N_t^\theta} \right) .$$

Denoting $C_T = \mathcal{L}_\alpha^0 C_{AT} T^\theta$, we then have:

$$E \approx C_X \Delta x^\lambda N_t + \frac{C_T}{N_t^{\theta-1}} . \tag{3.38}$$

Our goal is to compute the optimal timestep, *i.e.* optimal N_t, given the time interval. Hence we enforce:

$$\left. \frac{\partial E}{\partial N_t} \right|_{N_t^{opt}} = 0 .$$

We then obtain:

$$C_X \Delta x^\gamma - \frac{(\theta - 1) C_T}{\left(N_t^{opt} \right)^\theta} = 0 ,$$

$$\Leftrightarrow \quad N_t^{opt} = \left(\frac{(\theta - 1) C_T}{C_X \Delta x^\gamma} \right)^{\frac{1}{\theta}} .$$

Therefore, the optimal timestep $\Delta t = T / N_t^{opt}$ is given by Eq. (3.39):

$$\Delta t^{opt} = T \left(\frac{C_X \Delta x^\lambda}{(\theta - 1) C_T} \right)^{\frac{1}{\theta}} . \tag{3.39}$$

The computation of the optimal timestep requires prior knowledge of C_X and C_T, which is typically well-known for low-order schemes. However, for higher-order schemes, these estimates may not be available in closed form, or the bounds may be coarse. In such cases, the optimal timestep may still be computed numerically. As the existence of the optimal timestep is known, we compute the total error while decreasing the timestep. The value of the timestep at which the total error just starts increasing is the optimal timestep.

Another possible approach is as follows. The value of $\mathcal{L}_\alpha^0$ can be easily computed as the initial tracer field is known. For example, if the initial tracer field is differentiable,

$\mathcal{L}_\alpha^0 = \max_{x\in\Omega} \frac{\partial \alpha_0}{\partial x}$. To estimate the constants C_{IX}, C_{AX}, and C_{AT}, we complete a set of simulations with different Δx and Δt. The Y intercepts of the appropriate $\log - \log$ plots yield the asymptotic values of these constants [151]. Using these values along with Eq. (3.39), we can compute the optimal timestep value for higher-order schemes, as will be illustrated in Chapter 4

3.4.3 Remarks

Super-accuracy.

If the numerical PDE advection involves explicit schemes, stability (e.g. CFL) conditions would be commonly imposed. However, each composed advection is here carried out separately (if $M = 1$, for each Δt), and are thus independent and decoupled from the other advections, without multiplication of errors. This is because the numerical advections of past composition timesteps are not directly utilized in the new numerical advection. Within $\Delta t_c = M\,\Delta t$, past numerical advections are only used $M - 1$ times, with $M = 1$ or small. After Δt_c, there is thus no multiplicative compounding of past errors, neither of spatial and temporal truncation errors, nor of round-off errors. For M small, there is thus no possibility for numerical instability, even for classically unstable advection schemes. Without multiplicative error growth, spurious numerical diffusion or dispersion are also not possible. Hence, in all of our simulations (not all shown), we observed neither growth nor propagation of numerical errors and instabilities, even for larger timesteps. Combining all of these properties together, the result is what we refer to in short as 'super-accuracy'.

To ensure stability in the case of M large and large optimal composition timestep $\Delta t_c = \Delta t_{opt}$, one needs to ensure that Δt is small enough. This is because each individual flow map (of duration Δt_{opt}) is computed by solving Eq. (3.3) using the small timestep Δt, ensuring that the advection scheme is stable.

Of course, in all cases, the global truncation error is in general larger if Δt is larger. The super-accuracy however ensures that this global error is not affected by compounding of the local truncation errors.

Optimal grid spacing.

If the velocity field is analytically known or if spatial interpolations can be performed, then the optimal grid spacing may also be computed, by using a desired CourantâĂŞFriedrichsâĂŞLewy (CFL) number or desired local accuracy. Considering the former, let us assume a characteristic velocity $\mathcal{U}$ and a desired CFL value of $\mathcal{C}$. Then we have:

$$\mathcal{C} = \mathcal{U}\frac{\Delta t}{\Delta x}\,. \tag{3.40}$$

Inserting the optimal timestep of Eq. (3.39), we obtain,

$$\mathcal{C} = \mathcal{U}T\left(\frac{C_X \Delta x^{\lambda-\theta}}{(\theta-1)\,C_T}\right)^{\frac{1}{\theta}}\,. \tag{3.41}$$

As a result, for the desired $\mathcal{C}$, the optimal grid spacing is given by Eq. (3.42):

$$\Delta x_{opt} = \left(\frac{(\theta-1)\,C_T}{C_X}\right)^{\frac{1}{\lambda-\theta}}\left(\frac{\mathcal{C}}{\mathcal{U}T}\right)^{\frac{\theta}{\lambda-\theta}}\,. \tag{3.42}$$

The corresponding optimal timestep is given by Eq. (3.43):

$$\Delta t_{opt} = \frac{\mathcal{C}}{\mathcal{U}}\left(\frac{\mathcal{C}}{\mathcal{U}T}\right)^{\frac{\theta}{\lambda-\theta}}\left(\frac{(\theta-1)\,C_T}{C_X}\right)^{\frac{1}{\lambda-\theta}}\,. \tag{3.43}$$

Parallelization and memory.

It is clear from Sec. 3.3.2 that the individual flow map computations and the corresponding interpolation operator computations are completely independent and hence can be parallelized. Thus, if one has access to large number of processors (larger than the number of flow map computations *i.e.* N_c), the entire computation would simply require a computational time comparable to the time needed for a single advection timestep computation. However, where we gain in computational time, we lose in storage. The more the parallelization, the more the storage (if all the N_c processes are run in parallel, we need to store N_c fields instead of just one). Often, one has access to only a small number of processors. In such cases, efficient par-

allelization strategies can be designed to minimize the net computational time and memory requirement. For example, if we have access to four processors, than an efficient strategy can be to compute the flow map $\boldsymbol{\phi}_{0.25N_c}^{0}$ on processor 1, $\boldsymbol{\phi}_{0.5N_c}^{0.25N_c}$ on processor 2, $\boldsymbol{\phi}_{0.75N_c}^{0.5N_c}$ on processor 3, and $\boldsymbol{\phi}_{N_c}^{0.75N_c}$ on processor 4 in parallel. Each of these four computations would still use composition of constituent flow maps (each over a single optimal composition timestep duration), but now without storing the intermediate fields. Once these computations are complete, they can be composed together and with the tracer initial condition to yield the final tracer field. In general, if N_{proc} processors are available, then the computational time is reduced by a factor of N_{proc} while the memory usage is increased by the same factor (as compared to a completely serial solve). Depending on the specifications of the system (FLOP count, I/O speeds etc.) an optimal choice about the degree of parallelization can be made. Note that this discussion is only pertinent to the case of the advection equation, and if one is to solve for a case with diffusion and/or sources, no time parallelization is possible. Of course, space parallelization can still be used.

Storage vs. re-computation of the flow map.

For multiple tracers, it may be desirable to store the computed intermediate flow maps (and the associated interpolation operators) in memory if the storage access is efficient, or they can be recomputed whenever required if the machine has a better FLOP count. In general, there is no preferred choice, and the decision may vary from case-to-case. If the flow map computation is completed online, just after the new advection velocity is itself computed, the storage needs are limited. These needs then depend on the ratio of the optimal flow map composition timestep to the velocity advection timestep.

Unavailability of the velocity field at all intermediate times.

It is common for velocity fields to be available as snapshots only at certain fixed times, and not at all the required times. This occurs when one is trying to study the advection of a tracer field in experimentally observed velocity fields, either in

a lab or in nature (ocean, atmosphere, etc.). In such cases, the snapshots of the velocity field need to be interpolated at the required times. This introduces further errors in the numerical advection, however it does not affect the interpolation as this operation simply requires only the computed flow maps. Even though a rigorous treatment as in Sec. 3.4.2 is possible, here we present a qualitative description of the effect of velocity interpolations. As the timestep size decreases, the number of advection computations increase. Even though the truncation error in each of the advection computations is small, the error due to the interpolation of the velocity fields required for these computations is larger (along with the interpolation error in the flow maps). However, for larger timesteps, even though the (truncation) error from the individual advection computations is large, the error due to the interpolation of the velocity fields at intermediate times is low (as the velocity field is needed at less intermediate times). This implies that knowing the velocity field only as snapshots in time has the same effect on the optimal composition timestep size as the flow map interpolation. Thus, if this fact is not taken into account while predicting the optimal composition timestep, the computed optimal timestep value would be smaller than the actual optimal timestep (*i.e.* when the velocity interpolation is accounted for).

Effect of round-off error on the optimal composition timestep.

The current analysis to compute the optimal numerical composition timestep only accounts for the truncation errors in the flow map computation, interpolation operation, and the propagated error. However, during the actual implementation, one might also want to consider the effect of round-off errors due to finite precision arithmetic. Specifically, let us assume that round-off errors in a single advection computation (to compute $\boldsymbol{\phi}_{i+1}^{i}$) is $\mathscr{E}_A$ and that in the interpolation computation is $\mathscr{E}_I$. The value of $\mathscr{E}_A$ would not be significantly affected by the size of the timestep (*i.e.* even though the truncation error will be large for large timesteps, the magnitude of the round-off errors may not change much). Assuming that N_t is large enough, and as all the individual advection computations are decoupled, we can assume that the total round-off error in the advection computations is of the order of the maximum amongst the individual

$\mathscr{E}_A$'s. We can further assume that the round-off error in the interpolation operations ($\mathscr{E}_I$'s) would all be of the same order. Thus, similar to $\mathscr{E}_A$, the net round-off error in interpolation would be of the order of the maximum amongst the individual interpolation round-off errors. Once these are estimated, we can account for these in the error balance equation (Eq. (3.38)) that now yields the modified optimal composition timestep. Often times, the round-off errors will be negligibly small as compared to the truncation errors and can be neglected. However, for machines with lower precision, and stiff flow fields, the round-off errors may be significant.

3.5 Software Development

To practically employ the method of composition to predict tracer advection in various analytical and realistic flow fields, we have developed an efficient and easy-to-use computational framework. Our software framework has been extensively tested and benchmarked, and has been utilized to study advective transport in simulated marine flow fields as well as to predict transport features and the associated coherent structures in real-time sea experiments [131; 47; 87; 88; 180; 181; 51; 148; 3; 38; 132].

Our toolbox is programmed in MATLAB with extensions to read data from the native MATLAB data structures as well as from *netCDF* files commonly used in ocean modeling. The entire setup is object oriented and modular wherein a user can easily set up new simulations, create complex domains, and implement new numerical methods. The involved difference equations are solved on a regular Cartesian grid using the finite volume method with upto 5^{th}-order accurate spatial and upto 3^{rd}-order accurate temporal numerical schemes. The numerical timestep is either set explicitly, governed implicitly through a maximum permissible CFL number, or chosen to be the optimal composition timestep. The solver is capable of handling both two-dimensional (2D) and three-dimensional (3D) domains and makes use of the internal MATLAB parallelization for efficient matrix-vector solves. Further, flow maps over different time intervals are computed independently and in parallel. The amount of parallelization is intelligently chosen based on the availability of processors and free

memory (see Sec. 3.4.3).

To make conclusive decisions from the predicted data, visualization of the results must be in intuitively interpretable. To better visualize our advective transport predictions and the other Lagrangian fields such as the flow maps and coherent structures, we have incorporated a direct visualization interface through *SeaVizKit* - a novel browser-based tool used for visualizing various ocean products [3].

Our software toolbox has been used to generate all relevant results in this thesis and the applications of our software toolbox coupled with the SeaVizKit visualization framework are showcased in Sec. 6.1.3 for various different marine flow fields around the world for various different purposes.

3.6 Summary

In this chapter, a novel numerical methodology was derived for the numerical computation of advective transport and diffusion–reaction of tracer quantities through flow map composition. The method of composition for advection is 'super-accurate', yielding numerical solutions almost devoid of compounding numerical errors but with the advantage of Eulerian resolution in space. We develop schemes for the addition of tracer diffusion, reaction, and source terms, and the implementation of various boundary conditions. We derive rigorous error bounds and prove that the method of composition yields additive (and not compounding) errors. We further show that there exists an optimal value of the composition timestep that yields the minimum total numerical error by balancing the advection and composition errors, and derive the expression for this value. We conclude with some remarks and a discussion about the development of the software toolbox based on the method of composition. The applications of the developed numerical method are showcased in Chapter 4 for various analytical flow fields and realistic data-assimilative plume simulations.

Chapter 4

Advection Through Flow Map Composition - Applications

In this chapter we illustrate the properties and capabilities of the new method of composition, introduced in Chapter 3, and compare and contrast the results with those from classic high and low-order schemes. We first consider a forward-backward advection in a swirl flow with an analytically known solution in Sec. 4.1. This allows us to accurately capture and illustrate the error trends and the effectiveness of the optimal composition timestep value for schemes of varied orders of accuracy. We then add a diffusion term in Sec. 4.2.1 and create a new benchmark test case for advection–diffusion in the swirl flow. We show that the net spurious numerical diffusion is minimized for the method of composition. The second example (Sec. 4.2.2) considers an idealized version of a fluid flow exiting a narrow constriction that creates unsteady eddies and meanders downstream. We study the advection–reaction of a tracer to show that composition-based advection can be easily combined with source / reaction terms as well as with different kinds of tracer boundary conditions. Sec. 4.3.1 considers the simulated advection of sediment plumes resulting from deep sea mining operations around a possible mining site in the Bismarck Sea. After 5 days of advection by the underlying dynamic ocean currents, we again see that the composition-based advection leads to a more accurate sediment field evolution than regular PDE advection schemes. Finally, after analyzing these various simulated flow

fields, we look at the performance of the method of composition to provide real-time predictions during the 2018 PLUMEX sea exercise conducted in the Southern California Bight (Sec. 4.3.2). We compare the predicted sediment plume transport in the dynamic ocean in 3D with the corresponding real-time observations to find a satisfactory match between the predicted and the observed plumes for up to 6 hours after the initial release. This real-time application highlights the usability and the accuracy of the method of composition to support and analyze real-time sea experiments and data. Results from this chapter have been published in Kulkarni and Lermusiaux [132]; Kulkarni et al. [131]; Munoz-Royo et al. [181]; Coulin et al. [38].

4.1 Benchmarking and Optimal Timestep Results

To compare the resulting numerical errors accurately, we require a test case where the analytical solution is exactly known. We therefore choose the classic test case of an analytical swirl flow in a closed square domain $(x, y) \in [0, 1] \times [0, 1]$ and $t \in [0, 1]$ [49; 161], where the non-dimensional velocity is given by Eq. (4.1),

$$\boldsymbol{v}(x, y, t) = \left(C(t) \sin^2(\pi x) \sin(2\pi y), -C(t) \sin^2(\pi y) \cdot \sin(2\pi x)\right) . \tag{4.1}$$

Here $0 \leq t < 1$, and $C(t) = 1 \; \forall \; t < 0.5$ and $C(t) = -1 \; \forall \; t \geq 0.5$. The direction of the flow thus reverses at $t = 0.5$. Due to symmetry, the final tracer field is exactly equal to the initial condition. This allows computing the errors incurred in the various methods to a high degree of precision. The non-dimensional initial condition for the tracer field is given by Eq. (4.2),

$$\alpha_0(x, y) = \exp\left(-100\left((x - 0.25)^2 + (y - 0.25)^2\right)\right) . \tag{4.2}$$

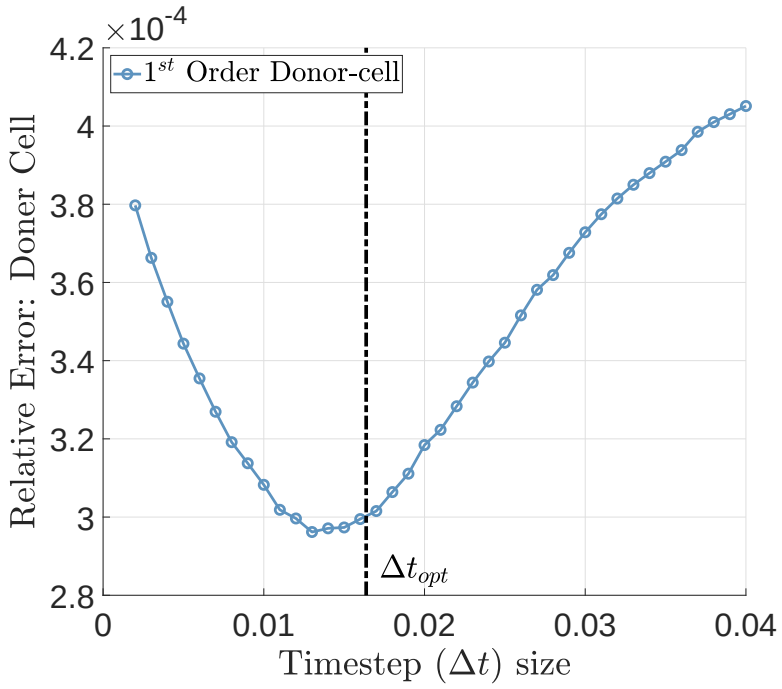
Optimal Composition Timestep

To illustrate the results related to the optimal composition timestep, we use the first-order upwind scheme (also called the donor-cell method) for the advection flux,

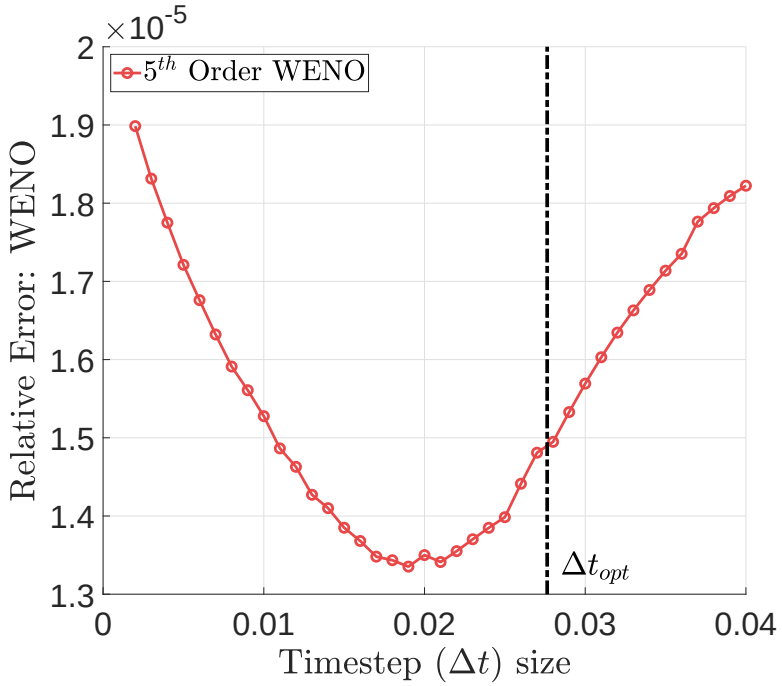
forward Euler time marching, and bilinear interpolation. The composition operation is here performed after every timestep, *i.e.* $M = 1$. The domain is discretized in 256×256 control volumes. As explicit error bounds are available for these schemes, the optimal timestep Δt_{opt} is evaluated using Eq. (3.39). Fig. (4-1a) shows this optimal timestep value and compares it to the actual total error (numerically computed) for a wide range of timestep sizes. This actual total error is observed to be minimum at a timestep very close to the optimal one.

Fig. (4-1b) shows the actual total error against the size of the timestep for the same test case, but when the 5^{th}-order accurate WENO scheme (WENO5) [191; 122] along with third-order accurate TVD Runge Kutta (TVD-RK3) time marching [85] are used for advection and a WENO-based interpolation scheme (also 5^{th}-order accuracy) is used for interpolation [216]. It is observed that the actual total error is minimum for a certain timestep value. It can be challenging to compute the optimal timestep value accurately for higher-order schemes, as the analytical error bounds may not be available and/or may be loose. To estimate the optimal timestep in this case, we follow the approach outlined in Sec. 3.4. We know the values of γ, β and θ through our choices of numerical schemes, along with the value of $\mathcal{L}_\alpha^0$. We estimate the constants C_{IX}, C_{AX}, and C_{AT} through multiple simulations with different Δx and Δt. Once the values for these constants are obtained, we compute the optimal timestep value for higher-order schemes using Eq. (3.39). Fig. (4-1) confirms that this optimal value is close to the observed minimum of the actual total error. Lastly, as the following results describe, when using the method of composition for advection, higher-order schemes are typically not needed, as lower-order schemes already provide accurate solutions without compounding errors. Hence, even though the analytical optimal timestep may not be not available, or might be difficult to estimate for certain higher-order schemes, it is of little impact in practice.

In Fig. (4-1), we note that the two predicted optimal timesteps are very close to, but slightly larger than, the two observed minima. Overall, a larger analytical optimal timestep will be observed when interpolation errors are overestimated or PDE errors are underestimated. This can occur for several reasons. First, interpolation errors are



(a) Optimal timestep for first-order donor-cell scheme



(b) Optimal timestep for 5^{th}-order WENO scheme

Figure 4-1: Relative error in the final tracer field against the timestep size for the donor-cell - forward-Euler and WENO5 - TVD-RK3 combinations, respectively. For the donor-cell - forward-Euler (Fig. (4-1a)), the analytically evaluated 'optimal timestep' is $\Delta t_{opt} = 0.01637$ (dash-dotted line), which is close to the observed minimum error timestep. For the WENO5 - TVD-RK3 (Fig. (4-1b)), the predicted optimal timestep is at $\Delta t_{opt} = 0.027633$ (dash-dotted line), which is also close to its observed minimum error timestep at $\Delta t = 0.019$.

estimated using the global bound through $\partial^\gamma \boldsymbol{\phi}$. Since this value varies in space, the expected average interpolation error could be low but the maximum could be high (in which case we overestimate the true averaged error). Second, the PDE error can be underestimated since we only consider the dominant error component and truncate the rest of the error expression. This truncated expression can thus underestimate the net error in the PDE solve. We note that the magnitude of the truncated terms relative to the actual total error will vary with the order of the schemes, and so the relative accuracy of the optimal timestep will also vary, as shown in Fig. (4-1). Finally, in deriving Eq. (3.39), the integrated effects of the interpolation and advection errors over one timestep were separated into to a sum of errors.

Advection by the Reversible Swirl Flow

Building on the above validation of the optimal timestep derivations, in all the studies henceforth, we utilize $\Delta t = \Delta t_{opt}$ and select $M = 1$. Fig. (4-2a) shows the relative total error in the final tracer field at $t = 1$ against the grid spacing. The relative error for the method of composition is found to be much smaller than that for the regular PDE-based approach even if the order of convergence of the latter is significantly higher. One observes that even a first-order composition-based method yields smaller errors than a 5^{th}-order regular PDE-based method, for up to about 600×600 grid points. This is mainly because the composition-based method computes an independent flow map for each timestep, thus the diffusive or dispersive errors in the numerical scheme for advection are not compounded in time. The computational time required for a specific accuracy is orders of magnitude lower for the method of composition. This is clearly observed in Fig. (4-2b), which plots the relative error in the solution against the computational time (in seconds). The spatial grid resolutions considered in this study are the same as those from the study of relative error against grid spacing, and the computational time increases with decreasing grid spacing (increased resolution). These simulations were performed in MATLAB on a single quad-core Intel i7-4790 CPU clocked at 3.60 GHz. It was ensured that no parallelization was used between the different flow map computations for the composition-based

advection to maintain a fair comparison between the methods. However, MATLAB's internal vectorization/parallelization for matrix-vector operations was not disabled. As both the approaches used the same internal routines and sub-routines, these internal operations impacts both the methods similarly. These results suggest that even as low as first-order accurate composition-based advection should yield good accuracy, and practically there is little need to go to higher-order schemes.

Finally, Fig. (4-3) shows the initial tracer field followed by the tracer fields at $t = 0.5$ and at $t = 1$. As mentioned above, ideally the tracer field at $t = 1$ should be identical to the initial condition. We find that regular advection with low-order numerical schemes (Fig. (4-3a)) suffers heavily due to numerical diffusion, whereas regular advection with high-order numerical schemes (Fig. (4-3b)) is minimally prone to such errors, and recovers the initial tracer field well. However, even when using low-order numerical schemes, the method of composition is able to almost exactly recover the initial tracer field (Fig. (4-3c)) due to the lack of compounding numerical errors. This further reinforces the result that low-order composition-based advection produces results comparable to much higher-order regular advection schemes.

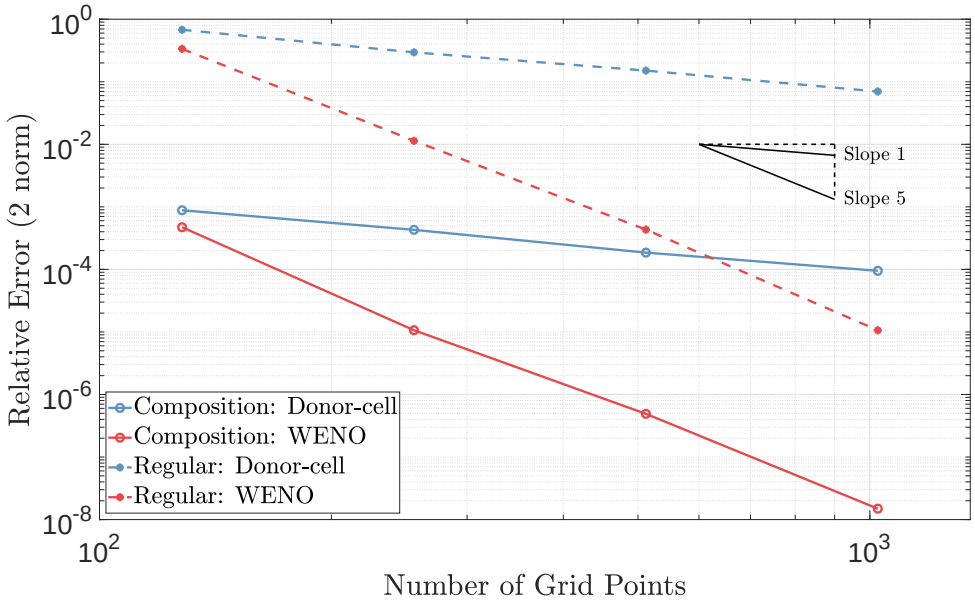
4.2 Analytical Applications

4.2.1 Advection-Diffusion in a Reversible Analytical Swirl Flow

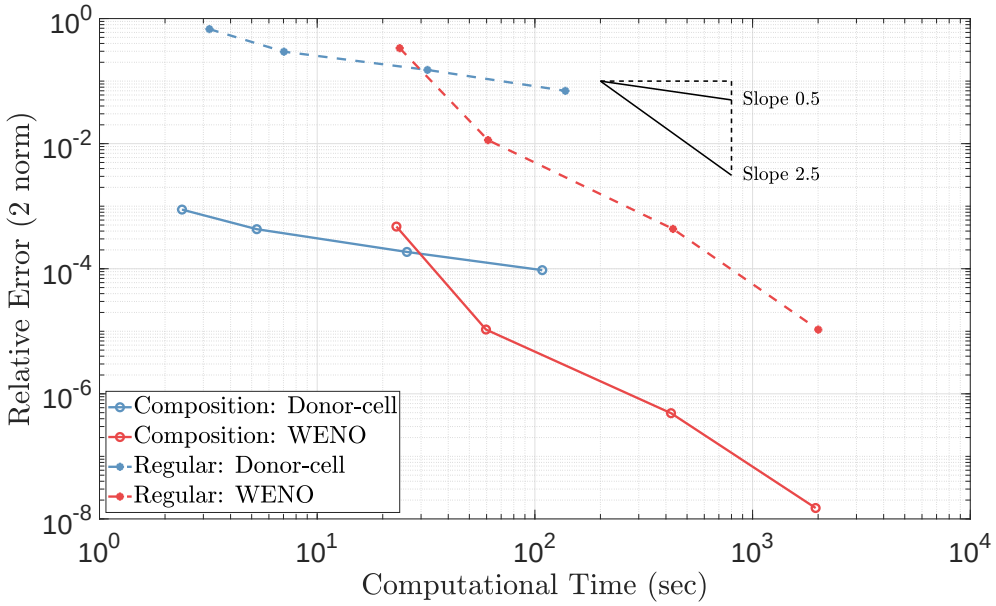
To showcase the developments of Sec. 3.3.3, we now add tracer diffusion, specifically a non-dimensional diffusivity $\kappa = 3.510^{-4}$ in Eq. (3.9). We create a new benchmark problem that extends the classic swirl flow advection test to a new advection-diffusion test. The underlying velocity field is still given by Eq. (4.1), but the tracer is now advected and diffused, with an initial condition given by Eq. (4.3),

$$\alpha_0(x, y) = \mathbb{I}\left(0.185 \leq \sqrt{(x - 0.25)^2 + (y - 0.25)^2} \leq 0.20\right), \tag{4.3}$$

where $\mathbb{I}$ is the indicator function, that is equal to unity if the condition in the parentheses is satisfied, and is zero otherwise. That is, the initial tracer is unity inside

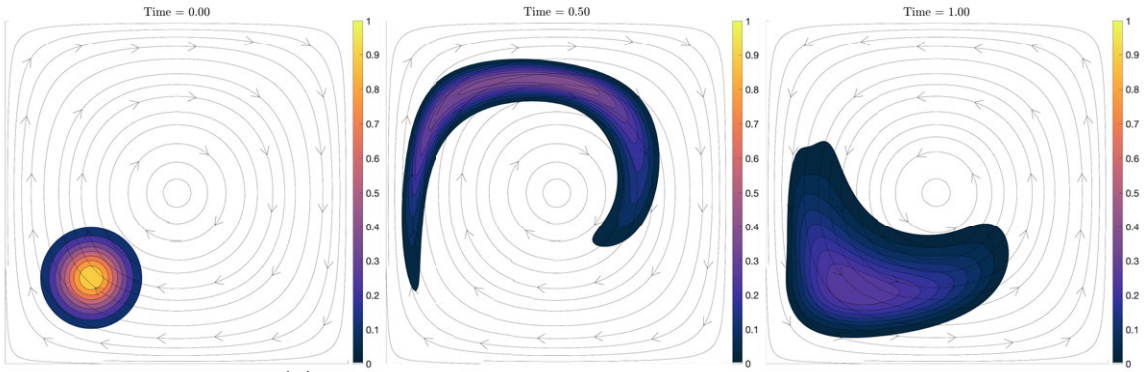


(a) Relative error in the tracer field against the number of grid points

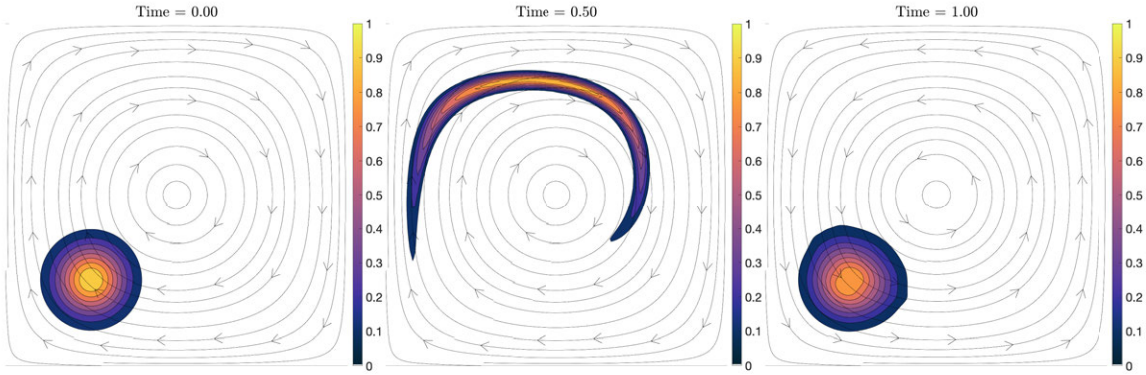


(b) Relative error in the tracer field against computational time

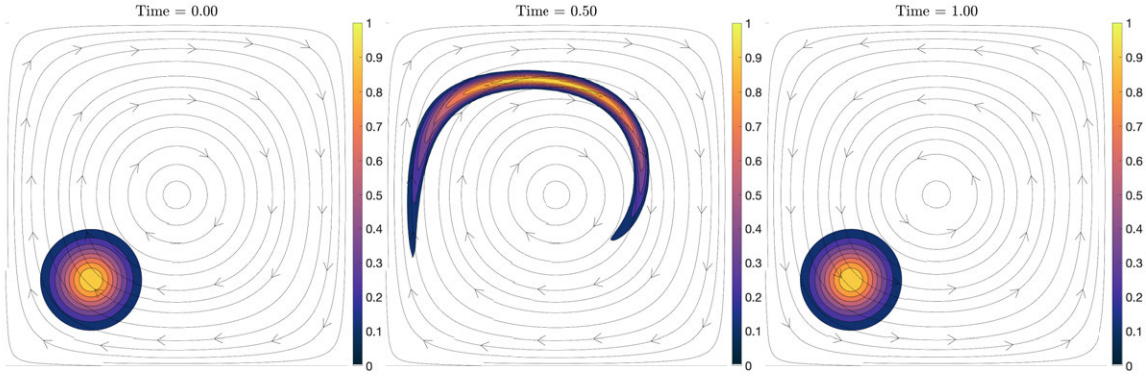
Figure 4-2: Relative error in the tracer field at final time $t = 1$ for the forward-backward advection in a reversible analytical swirl flow. Fig. (4-2a) shows the relative error against the number of grid points, where we observe the expected orders of convergence for the two numerical schemes used. Fig. (4-2b) plots the relative error against the computational time, for the same spatial grid resolutions as Fig. (4-2a). It can be seen that composition-based advection requires an order of magnitude lower computational time for the same accuracy.



(a) Regular advection: Donor-cell, forward Euler



(b) Regular advection: WENO5, TVD-RK3



(c) Composition-based advection: donor-cell, forward Euler

Figure 4-3: Forward-backward advection in the analytical swirl flow: the three panels in each panel show the initial tracer condition, the tracer field after the forward advection is complete, and the tracer field after the backward advection is complete, respectively. Regular advection with low-order numerical schemes (Fig. (4-3a)) suffers heavily from spurious numerical diffusion, whereas the method of composition can almost exactly recover the initial tracer field (Fig. (4-3c)), due to the lack of compounding numerical errors.

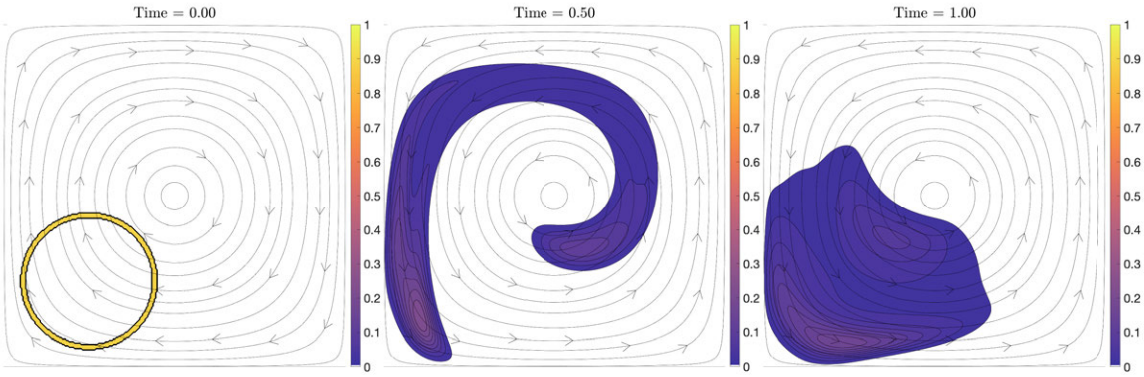
a ring of outer radius 0.2 and thickness 0.015, centered at $(0.25, 0.25)$. This initial condition provides high tracer gradients along the inner and outer circumferences of the ring for diffusion to work, during both the forward and backward swirl advections. The results thus highlight the effects of spurious numerical diffusion and dispersion.

We solve this benchmark using Lie operator splitting (see Sec. 3.3.3) with, for all advection schemes, a second-order accurate implicit diffusion solve. We compare and contrast classic advections (first-order donor-cell and 5^{th}-order WENO) with the composition-based advection (first-order donor-cell) using the optimal numerical timestep of Eq. (3.39). The grid resolution is 256×256.

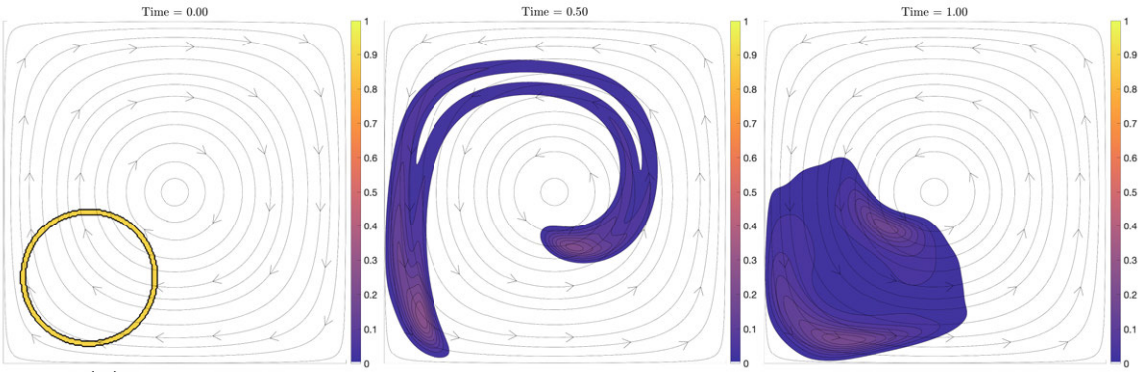
Fig. (4-4) shows the initial tracer field followed by the tracer fields at $t = 0.5$ and $t = 1$ for the regular advection–diffusion solve (with low-order and high-order advection schemes) and the composition-based advection–diffusion solve. Once can clearly see that the numerical diffusion in Fig. (4-4a) causes the inner hole in the ring to close up by $t = 0.5$, whereas the hole is not closed up for the higher-order regular advection and composition-based advection (even though this composition scheme is only first-order). Further, after backward advection and diffusion ($t = 1$), contrasting the last panels of Fig. (4-4b) and Fig. (4-4c), we find that the hole is closed up for the WENO-based regular advection, but still persists for composition-based advection, indicating that the former has larger spurious numerical diffusion than the latter. This fact is corroborated by the relative error in these fields (where the exact solution is assumed to be that on a 1024×1024 grid, using 5^{th}-order WENO regular advection), as illustrated in Table 4.1. This showcases the usability and the superiority of composition-based advection in solving advection-diffusion PDEs.

Table 4.1: Relative errors in solving the advection-diffusion equation for the reversible swirl flow (all methods use Lie operator splitting with a second-order implicit solve for diffusion).

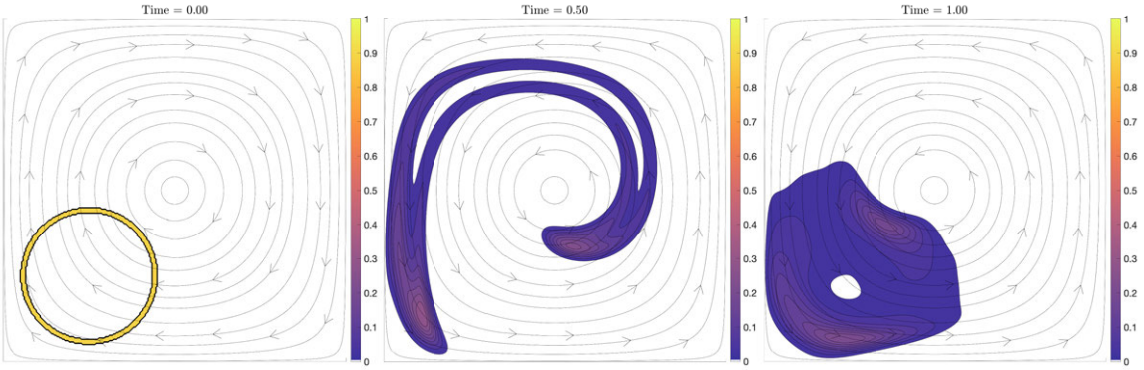
Advection Method	Relative Error
Regular advection: Donor-cell, forward Euler	7.32%
Regular advection: WENO5, TVD-RK3	1.02%
Composition-based advection: Donor-cell, forward Euler	0.76%



(a) Regular advection: Donor-cell, forward Euler; diffusion: second-order implicit solve



(b) Regular advection: WENO5, TVD-RK3; diffusion: second-order implicit solve



(c) Composition-based advection: donor-cell, forward Euler; diffusion: second-order implicit solve

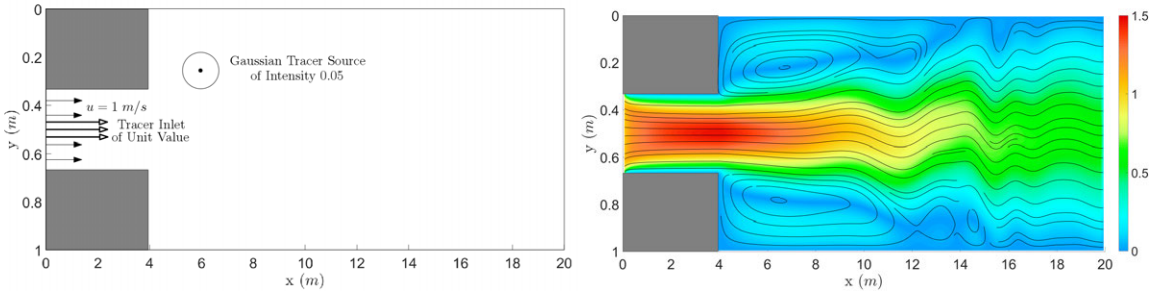
Figure 4-4: Forward-backward advection and diffusion in the analytical swirl flow: the three panels in each panel show the initial tracer condition, the tracer field after the forward advection and diffusion part is complete (*i.e.* at $t = 0.5$), and the tracer field after the backward advection and diffusion is complete (*i.e.* at $t = 1$), respectively.

4.2.2 Advection-Reaction in an Idealized Flow Exiting a Strait

We now examine tracer advection in flows encountered in the coastal oceans and in urban environments, and include a source term and variable boundary conditions. A barotropic jet exits a strait or an estuary into a wider channel and creates unsteady eddies and meanders downstream [163]. In urban settings, wind blowing through narrow constrictions between buildings into an open area also creates similar dynamics. Such flows can be idealized as sudden expansion flows that have been studied extensively [32; 50; 63].

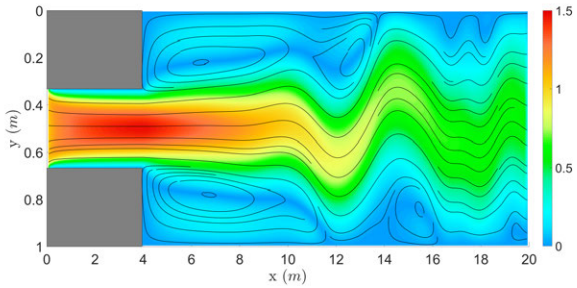
Fig. (4-5a) shows the schematic of the idealized 2D test case. We consider a 20 $m \times 1$ m channel with an inlet narrow constriction of width $1/3$ m and length 4 m. A uniform jet with a velocity 1 m/s enters the inlet constriction at $x = 0$. The other boundary conditions for the velocity are no slip at all the walls and an open (radiation) boundary condition at the $x = 20$ outlet. The velocity field is first developed for 500 s. Recirculation zones and breaks form, either to the north or south of the centerline, depending on the initial uncertain perturbations. The Navier–Stokes equations are solved using a finite volume framework with second-order spatial and temporal schemes and a projection method for pressure-velocity coupling [247]. Fig. (4-5) shows the advection flow field, at three discrete times $t = 10$, $t = 30$, and $t = 50$ s, after the initial development period of 500 s.

For the non-dimensional tracer field, we assume that a unit value enters the domain at the $x = 0$ inlet along the centerline, with a width of 0.1 m, and that there exists another Gaussian-shaped tracer source of intensity $0.05\,\mathrm{s}^{-1}$ centered at $x = 2$ m, $y = 0.25$ m. The boundary conditions are Dirichlet at the inlet, no flux at all the walls, and open (radiation) at the outlet. Further, we assume that there is no tracer diffusion, *i.e.* $\kappa = 0$. The tracer advection-reaction Eq. (3.9) is solved for $t = 0, \cdots, 50$ s after the initial velocity development period. The tracer computational setup involves a 160×120 grid, and both the method of composition and the regular advection computations utilize the WENO5 scheme for the spatial gradients and TVD-RK3 for time marching. The timestep value is the optimal timestep for the

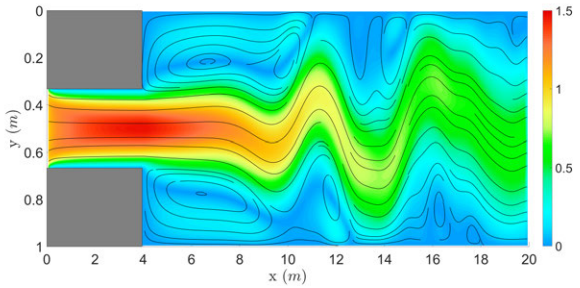


(a) Schematic of the test case setup

(b) Velocity field at 10 seconds



(c) Velocity field at 30 seconds

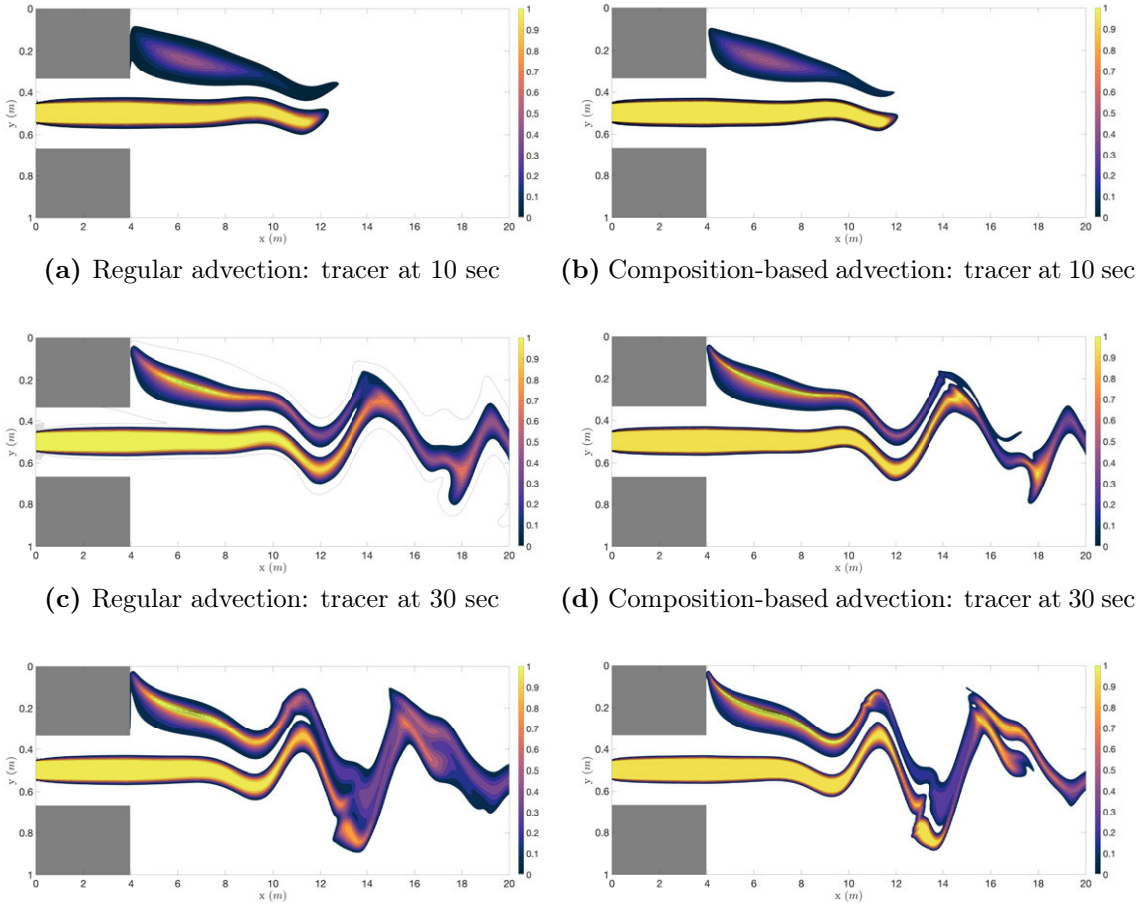


(d) Velocity field at 50 seconds

Figure 4-5: Flow exiting a strait: Test case setup and the velocity fields. Panel (a) shows the setup and the latter three panels show the velocity field at the specified times after the development period. The velocity field is computed by solving the Navier–Stokes equations in a finite volume framework with second-order spatial and temporal schemes and a projection method for pressure-velocity coupling [247].

method of composition. The contributions from the source and advective transport are computed independently of each other using Lie splitting, as outlined in Sec. 3.3.3. This example showcases the ability of composition-based advection to correctly handle tracer sources and multiple types of boundary conditions, as the inlet has a Dirichlet boundary condition whereas all the walls have a Neumann boundary condition.

Fig. (4-6) shows the tracer fields at $t = 10$, $t = 30$, and $t = 50$ s after the initial development period. It is clear that the method of composition results in minimal numerical diffusion in the advection process, whereas the regular advection results in a much higher amount of numerical errors. To compute the accuracy of our solution, we compare both the results with a more refined simulation (of grid size 480×360). We obtain that the relative error in the method of composition is 1.27%, whereas that in the regular advection is 4.18%. This showcases the advantages of the method of composition to simulate advection-reaction problems.



(a) Regular advection: tracer at 10 sec

(b) Composition-based advection: tracer at 10 sec

(c) Regular advection: tracer at 30 sec

(d) Composition-based advection: tracer at 30 sec

(e) Regular advection: tracer at 50 sec

(f) Composition-based advection: tracer at 50 sec

Figure 4-6: Flow exiting a strait: tracer advection-reaction. Panels on the left-hand-side show results using regular tracer advection (WENO5 in space and TVD-RK3 in time), specifically the tracer field at 10, 30 and 50 s (after the flow development period). Panels on the right-hand-side show the tracer fields at the same times but computed using the method of composition (WENO5 in space and TVD-RK3 in time, along with flow map composition). It is clear, especially at 30 and 50 s, that the method of composition is less prone to spurious numerical errors.

4.3 Realistic Applications

4.3.1 Sediment Plumes in the Bismarck Sea

Now moving on to realistic ocean examples, we first simulate the advection of passive sediment plumes in the Bismarck Sea. This example is inspired by the need to study the impacts of proposed activities to mine the seabed for rare metal resources [110; 35]. Extracting these metal ores from the seabed can create a plume of fine particles.

These sediment plumes may be released at various depths within the ocean, and they can prove to be extremely harmful to the local marine ecosystems and its components [251]. Further, they are advected by the dynamic ocean currents and hence can end up far from the original release location. It is thus of utmost importance to quantify and mitigate the impact of such activities on the surroundings. High-accuracy sediment transport forecasts without compounding numerical errors are thus needed.

To advect the sediment plumes, realistic dynamic ocean fields (currents, temperature, salinity, and ocean free-surface) are simulated by our MIT Multidisciplinary Simulation, Estimation and Assimilation System (MSEAS). MSEAS has been utilized for varied data-assimilative transports studies, including the Prestige oil spill [146], biogeochemical transports [17; 37], advection of drifting objects [144; 148], and tracer plumes in the Bismark Sea [38] and Southern California Bight [131]. The MSEAS numerical schemes include second-order finite-volumes with leapfrog time-stepping and implicit two-way nesting for multi-resolution hydrostatic primitive equation dynamics with a nonlinear free-surface [e.g., 96; 155; 95], as well as a high-order finite element code on unstructured grids for non-hydrostatic processes [246; 248; 68]. The MSEAS software has provided realistic data-assimilative simulations and predictions in varied regions of the world's ocean and for varied purposes [e.g., 154; 262; 94; 152; 80; 208; 147; 36; 128; 238; 240; 149].

The present MSEAS simulation domain covers a 855.36 $km \times 650.43$ km region around the Bismarck Sea. The initial ocean conditions were downscaled from the coarse analysis fields of 15 January 2016 of the $1/12^{\circ}$ HYCOM (Hybrid Coordinate Ocean Model) [41]. The higher-resolution MSEAS simulations were then ran up to 31 January 2016, with full tidal forcing [160]. For the dynamical and numerical parameters, as well as the validation of the MSEAS simulations, we refer to [38].

Using these MSEAS four-dimensional dynamic ocean currents, we simulate the advection of passive plumes from two sites in the Bismarck Sea. One site is the Solwara 1 proposed mining site [35]. The other is to the southeast, inside the St. Georges channel. Both the release sites are at a depth of 1275 m below the sea level. For the present example, we neglect the vertical movements of the material, and consider only

its advection in the horizontal plane at 1275 m depth. We assume that the material is released in a square box around the two sites (shown in panel (a) of Fig. (4-7), Fig. (4-8), and Fig. (4-9)), on 22 January 2016, 00Z. The sediment plumes are then advected by the dynamic ocean currents for 5 days. Without loss of generality for the present numerical study, it is assumed that there is no sediment diffusion. For the sediment advection, the horizontal computational domain is discretized in rotated Cartesian grid of size 1152×876, hence each control volume is of size 742 $m \times 742$ m. The timestep is chosen to be the optimal timestep for the method of composition (see Sec. 3.4.2), and the resulting mean CFL number is 0.8. The net duration of advection is 5 days. For the method of composition, we use a first-order donor-cell (upwind) scheme for the spatial gradients and forward Euler time marching. For the regular advection (*i.e.* without composition), we use the 5^{th}-order WENO scheme for the spatial gradients and TVD-RK3 time marching. Hence, we compare the first-order composition scheme to a much higher-order regular advection scheme.

For contrast, we also advect the passive tracer plumes using the method of characteristics, *i.e.* by seeding Lagrangian particles in the regions of interest and advecting them under the influence of the unsteady velocity field. This allows the comparison between the method of composition and the method of characteristics / Lagrangian advection. For this purpose, we seed each of the sites with 900 uniformly spaced particles. These particles are advected using 4^{th}-order Runge-Kutta time marching.

Fig. (4-7), Fig. (4-8), and Fig. (4-9) show the advection of the two initial sediment plumes under the effect of the MSEAS dynamic ocean currents over 5 days. We clearly observe that for the method of composition, even though it is only first-order, the numerical diffusion is minimal and the hyperbolic nature of the advection process is well maintained. However, even when using much higher 5^{th}-order schemes, classic advection suffers from large numerical diffusion due to the compounding of errors.

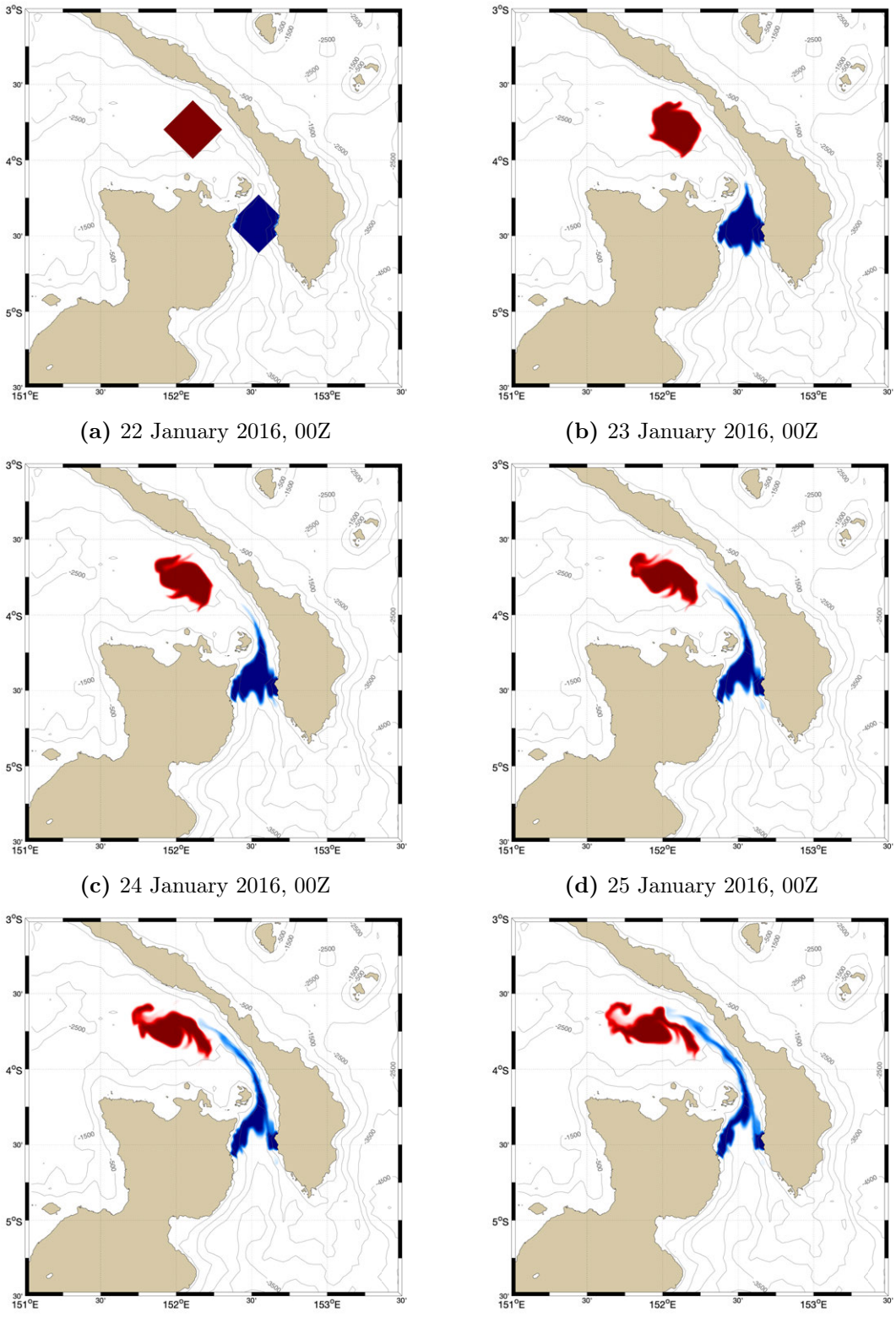
In the method of characteristics, as the particles are advected individually, no diffusion is expected, which is clearly seen in Fig. (4-9). Comparing this with Fig. (4-8), as shown in Fig. (4-10), confirms the lack of numerical diffusion in the method of composition, even though we solve a PDE in an Eulerian setting. The drawback of

using the method of characteristics when compared to the method of composition is however the loss of spatial accuracy in certain regions of the domain as computations are carried out in a Lagrangian frame. This is especially clear from the zoomed plots in Fig. (4-10), where one can observe this loss of spatial accuracy and coverage in certain regions, e.g. near the submesoscale eddies in the bottom left plot and near the mouth of St. Georges channel in the top right plot.

For a more quantitative comparison of the three used methods (regular advection, the method of composition, and the method of characteristics), we compute their relative errors with respect to a field advected using the method of characteristics but using 16 times as many particles. The results are summarized in Table 4.2, where we see that the method of composition yields more accurate solution than the method of characteristics. If the norm focuses only on the regions where final errors are largest (e.g. where the particle coverage is low), the comparison is much more favorable for the composition.

Further, as the composition-based computation uses simple first-order schemes, it was about 30 times faster than the regular high-order advection. The general advection patterns for the two schemes are logically similar (the velocity fields used are identical), but the method of composition maintains sharp gradients and detailed features (effects of submesoscale eddies, fronts, etc.), whereas sharp edges are smeared out by the regular advection. The method of characteristics takes the least amount of time (half that of the method of composition) as its computations were parallelized over the individual particles. However, the Lagrangian particles scheme looses accuracy in regions with low particle density (e.g. repulsive regions), as shown above.

Finally, in operational simulations, turbulent sediment plume diffusion would be modeled, but for accurate sediment forecasts, it would remain crucial for the spurious compounded numerical tracer diffusion to be eliminated. The present composition-based advection is an efficient solution.



(a) 22 January 2016, 00Z

(b) 23 January 2016, 00Z

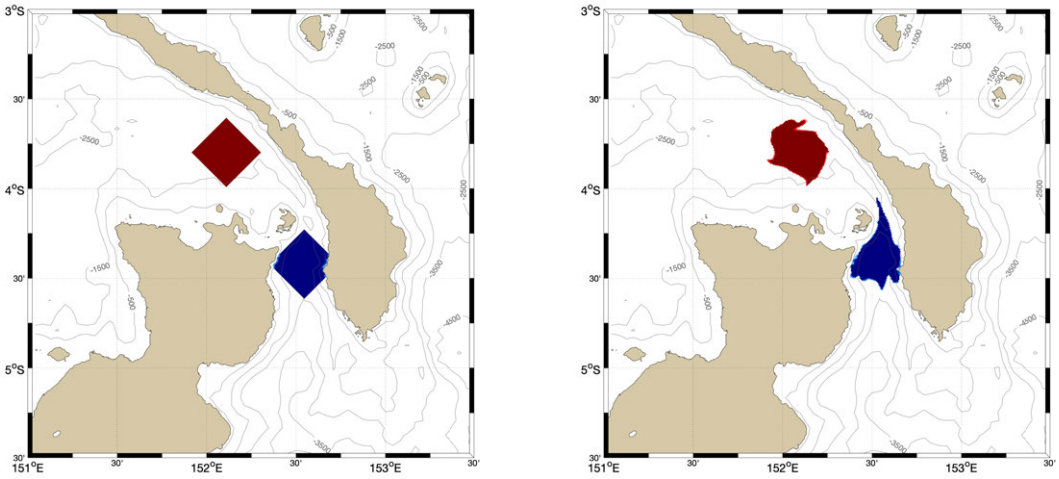
(c) 24 January 2016, 00Z

(d) 25 January 2016, 00Z

(e) 26 January 2016, 00Z

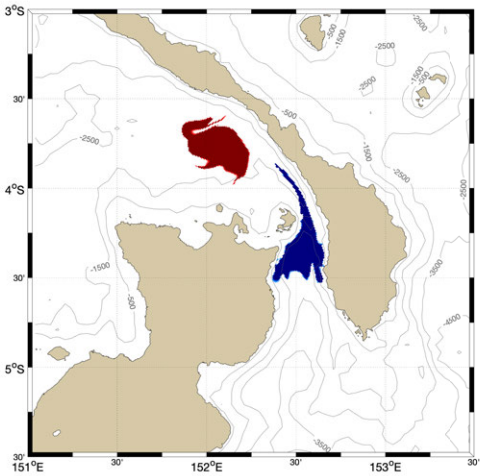
(f) 27 January 2016, 00Z

Figure 4-7: Sediment advection over 5 days from two possible deep sea mining sites, using regular advection with high-order WENO5 for the spatial gradients and TVD-RK3 time marching.

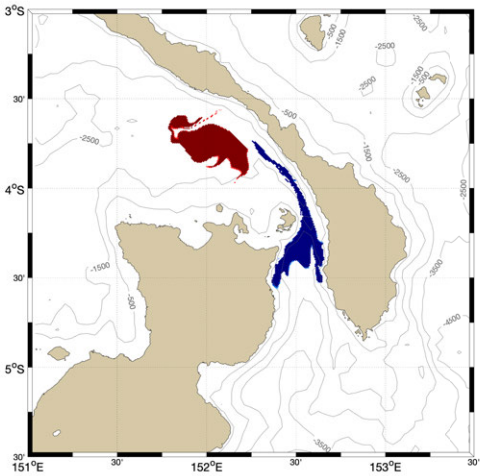


(a) 22 January 2016, 00Z

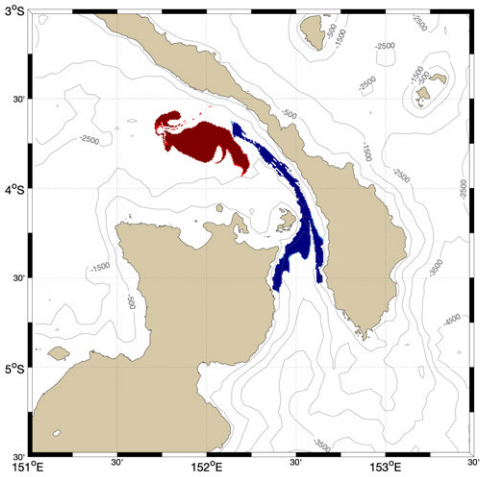
(b) 23 January 2016, 00Z



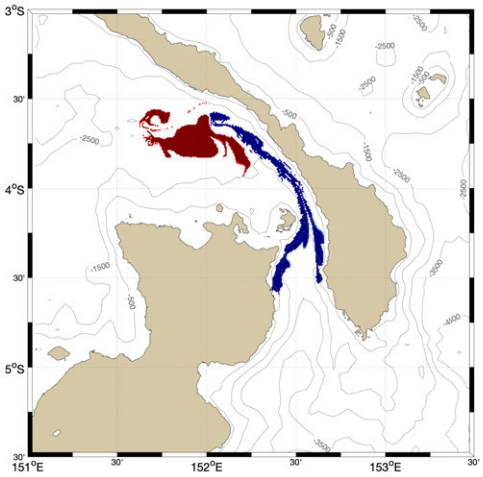
(c) 24 January 2016, 00Z



(d) 25 January 2016, 00Z

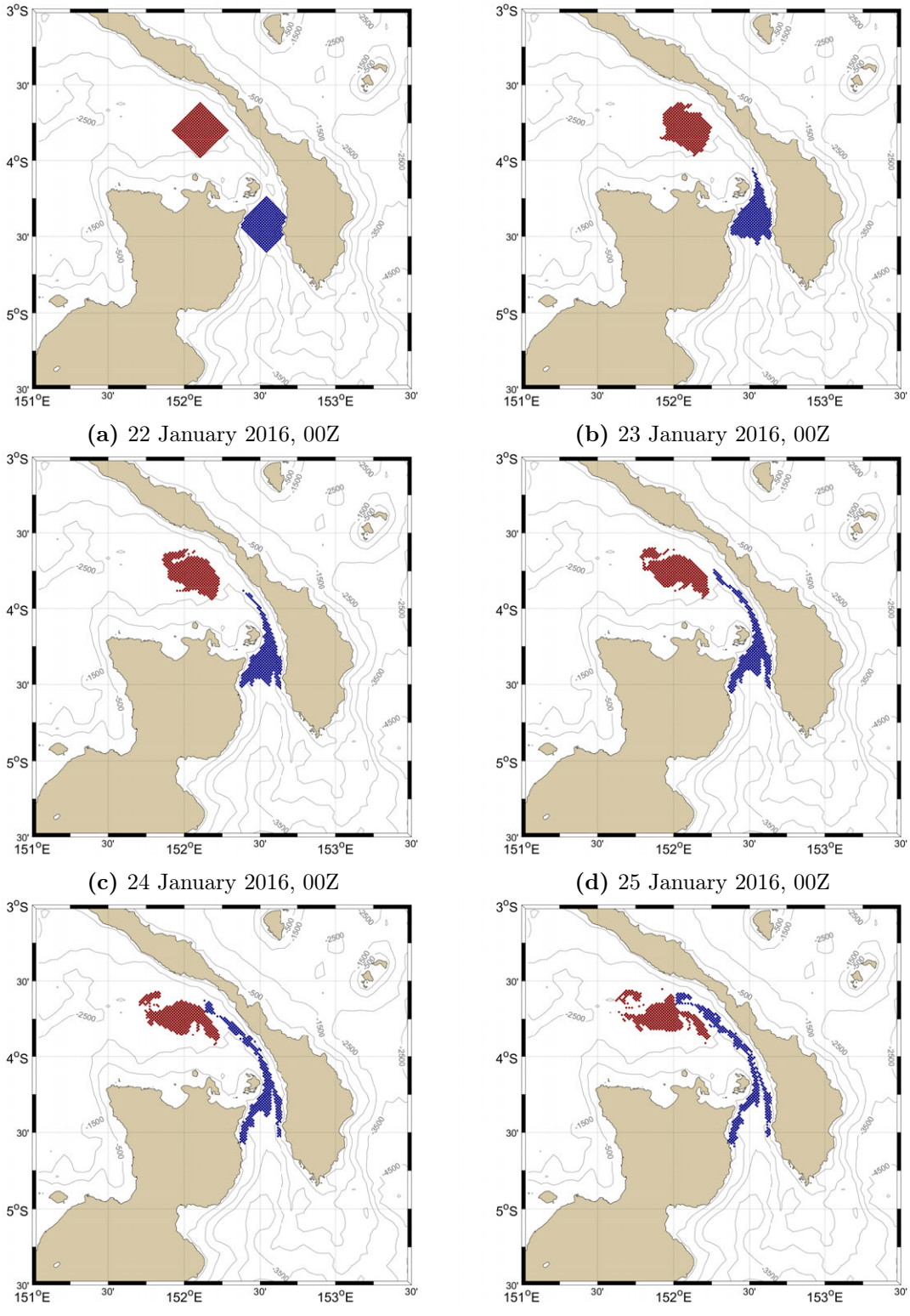


(e) 26 January 2016, 00Z



(f) 27 January 2016, 00Z

Figure 4-8: As in Fig. (4-7), but using the method of composition with the first-order donor-cell scheme for the spatial gradients and forward Euler time marching.



(a) 22 January 2016, 00Z

(b) 23 January 2016, 00Z

(c) 24 January 2016, 00Z

(d) 25 January 2016, 00Z

(e) 26 January 2016, 00Z

(f) 27 January 2016, 00Z

Figure 4-9: As in Fig. (4-7), but using Lagrangian trajectory advection using the Runge-Kutta 4 advection scheme.

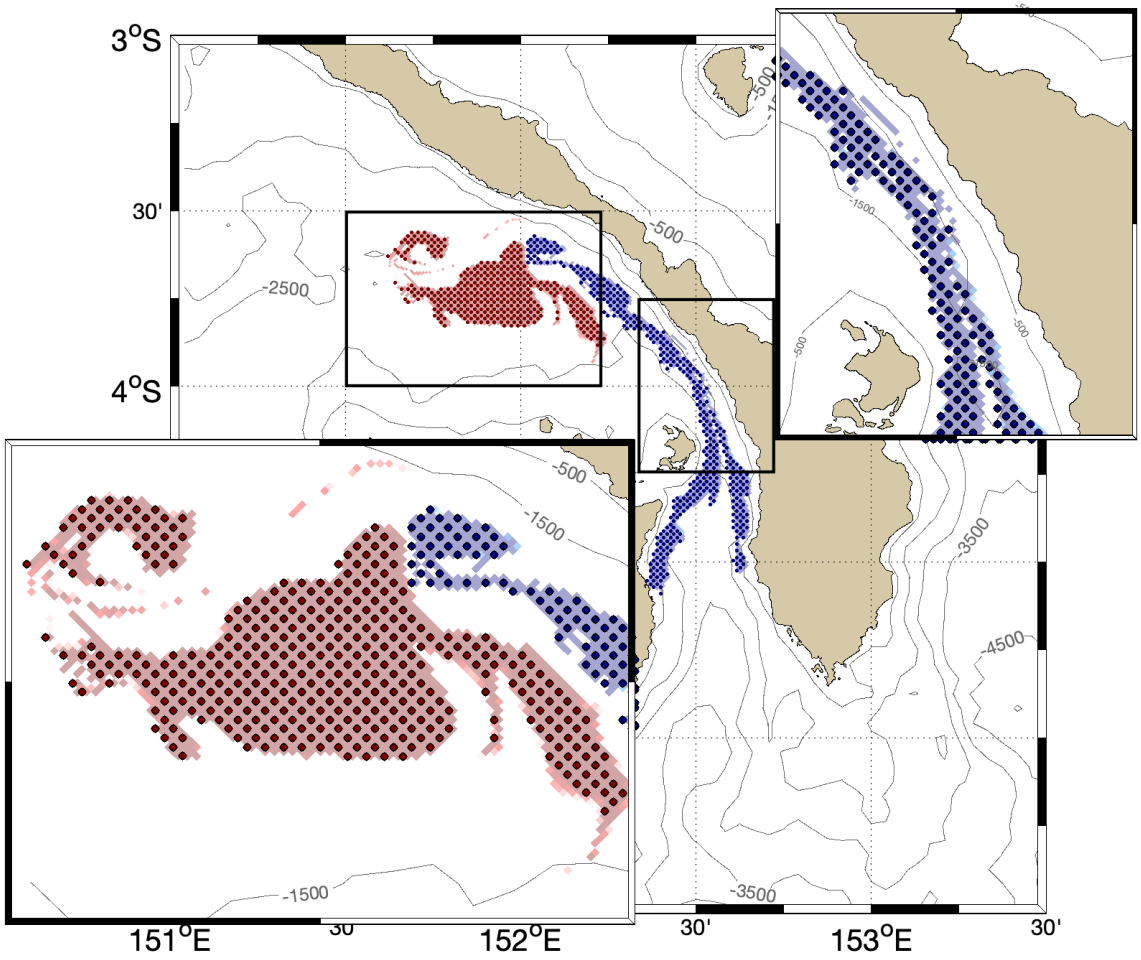


Figure 4-10: Comparison of the final sediment fields (on 27 January 2016, 00Z), when advected using the method of composition (Fig. (4-8f)) and trajectory advection (Fig. (4-9f)). The advected fields are extremely close to each other. However, the loss of spatial accuracy for trajectory advection can be seen in the zoomed sections.

Table 4.2: Relative errors in predicting sediment advection from two possible deep sea mining sites over 5 days.

Advection Method	Relative Error
Regular advection: WENO5, TVD-RK3	3.22%
Composition-based advection: Donor-cell, forward Euler	1.04%
Method of characteristics: RK4	1.09%

4.3.2 Real Time Prediction of Sediment Plumes in the Southern California Bight

Our modeling systems and software were employed to forecast the sediment plume transport in the southern California Bight during the 2018 PLUMEX sea exercise. This sea exercise occurred from 26 February to 05 March 2018 off the coast of San Diego. As a part of this experiment, sediment plumes mimicking those generated from deep sea mining operations were released and tracked. The results of the experiment provide insights into the behavior of such plumes and form the basis of our understanding of the potential environmental impacts of deep sea mining activities.

Real-Time Results

The considered modeling domain is off the coast of San Diego with an area of 687 $km \times$ 720 km as shown in Fig. (4-11). The red dashed lines show the special focus area around the gulf of San Catalina, where most of the at-sea experimental work occurred. The plume releases were planned at the starred locations in Fig. (4-11) at depths of 60 m, 80 m and 140 m below the ocean surface and the plumes were observed up to 8 hours. For the real-time ocean forecasting, the MSEAS modeling system was set up with 2-way nesting with resolutions of 1.5 km and 0.5 km for the full and nested domains (the special focus area within the dashed red line in Fig. (4-11)) respectively. Further, 100 optimized terrain following levels were used in the vertical direction. The model bathymetry used was obtained from the 15 arcseconds SRMT15 data set [231]. The tidal forcing fields were computed from the high resolution TPXO8-Atlas from OSU [55; 54] and reprocessed for the higher resolution bathymetry [160]

and nonlinear bottom drag. The forecasts were initialized from the 1/12° HYCOM (Hybrid Coordinate Ocean Model) analysis fields, but with updates based on available in-situ data.

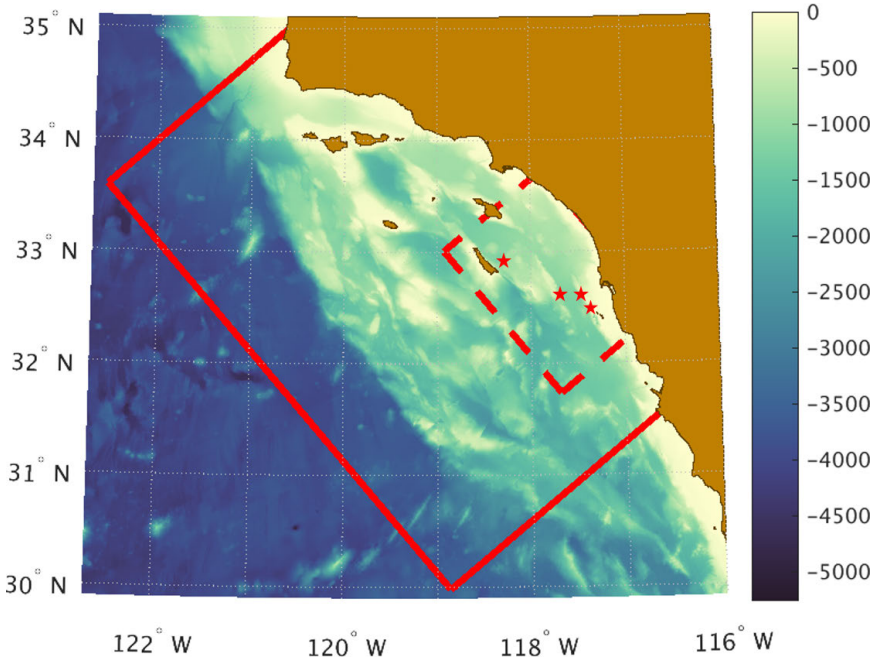


Figure 4-11: The MIT-MSEAS modeling domain for the 2018 PLUMEX sea exercise, along with the bathymetry of the region. The solid red line denotes the computational domain and the dashed red line demarcates the special focus area around the gulf of San Catalina. The red stars indicate potential plume release locations.

The forecasts of the ocean fields were used as the input to the plume advection computation, where the sediment plumes were assumed to be passively advected by the background ocean currents. Their evolution was numerically computed using the method of flow map composition in fully three-dimensional (3D) domains in real-time to support and guide the at-sea experiments. A collocated Cartesian grid was used with WENO5 scheme for spatial gradients and TVD-RK3 for time marching in a finite volume setup. To obtain the plume transport at a comparable scale to the ship movement, velocity output of the implicit 2-way nested ocean model was interpolated 4 times in each direction, resulting in a horizontal grid spacing of 125 m and vertical grid spacing of 2 m. A numerical timestep of 5 minutes was used.

The velocity fields as well as the plume transport forecasts were issued a day in advance, and an updated forecast for the day was issued early in the morning through

the MSEAS sea exercise web interface [87]. The sediment transport was predicted in real-time and reported in two ways: horizontal (2D) transport maps at different depths and fully (3D) maps. The plume transport was predicted for different start times ranging from 8AM PST to 8PM PST, starting every hour. This was done using the sliding time window approach and by storing the individual flow maps and composing them in the correct order to avoid recomputation (see Sec. 3.5). The intervals for plume tracking were up to 12 hours.

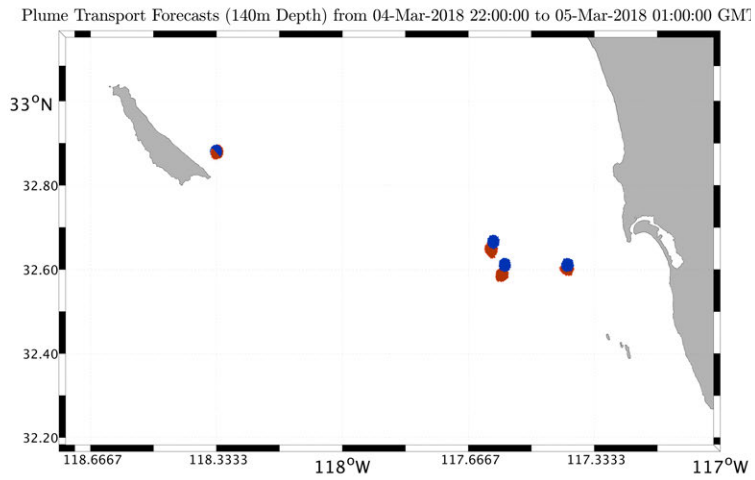
Let us now focus on a single plume release experiment that was conducted on 04 March 2018, 2PM PST to qualitatively better understand the dynamics of the 2D and the 3D dispersion and transport from the three potential locations over 12 hours. Fig. (4-12a) shows the plume transport in 2D at a depth of 140 m. Fig. (4-12b) shows the fully 3D evolution of the plume, assuming the plume was released at 140 m below the ocean surface.

It can be clearly seen that plumes from all the start locations are generally forecast to be transported southwards due to the ocean currents. Plumes from the two central locations are advected more as the strength of currents in this region is higher as compared to the currents near the California coast or around San Clemente island. As can be seen from Fig. (4-12b), the 3D forecast captures the vertical transport of the plumes which is quite substantial for the two eastern release locations. This is mainly because of the upwelling zones near the coast and other bathymetric effects. However the releases in the deeper ocean travel more coherently in the vertical.

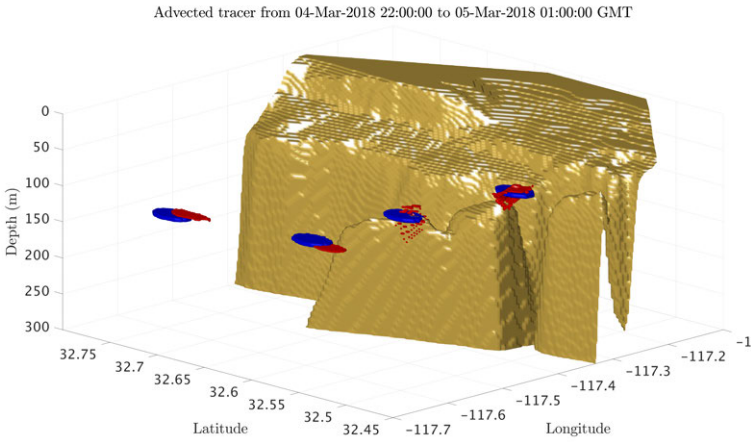
Comparison with Experimental Data

We now compare our plume transport predictions with the data from the experiment that was conducted at 04 March 2018, 2:11PM PST. A sediment plume of density 1030.4 kg/m^3 was released at a depth of 59 m aboard the research vessel Sally Ride [245]. The release location was $32.6920°N,\ 117.6126°W$. The plume was tracked for 371 minutes by using the shipboard transmissometer and tow-yo CTD [181].

We compare the accuracy of our plume advection predictions with the highlighted locations of the actual plume observation at 78, 120, 250, and 371 minutes after the



(a) Two-dimensional (2D) cross section at 140 m



(b) Three-dimensional (3D) view

Figure 4-12: Plume transport forecast over 3 hours when released at a depth of 140 m. The initial plume release markers are in blue and the final transported plume is in red.

start of the release. The locations of the direct observations of the plume by the tow-yo CTD are highlighted in Fig. (4-13a) with hollow red circles [181].

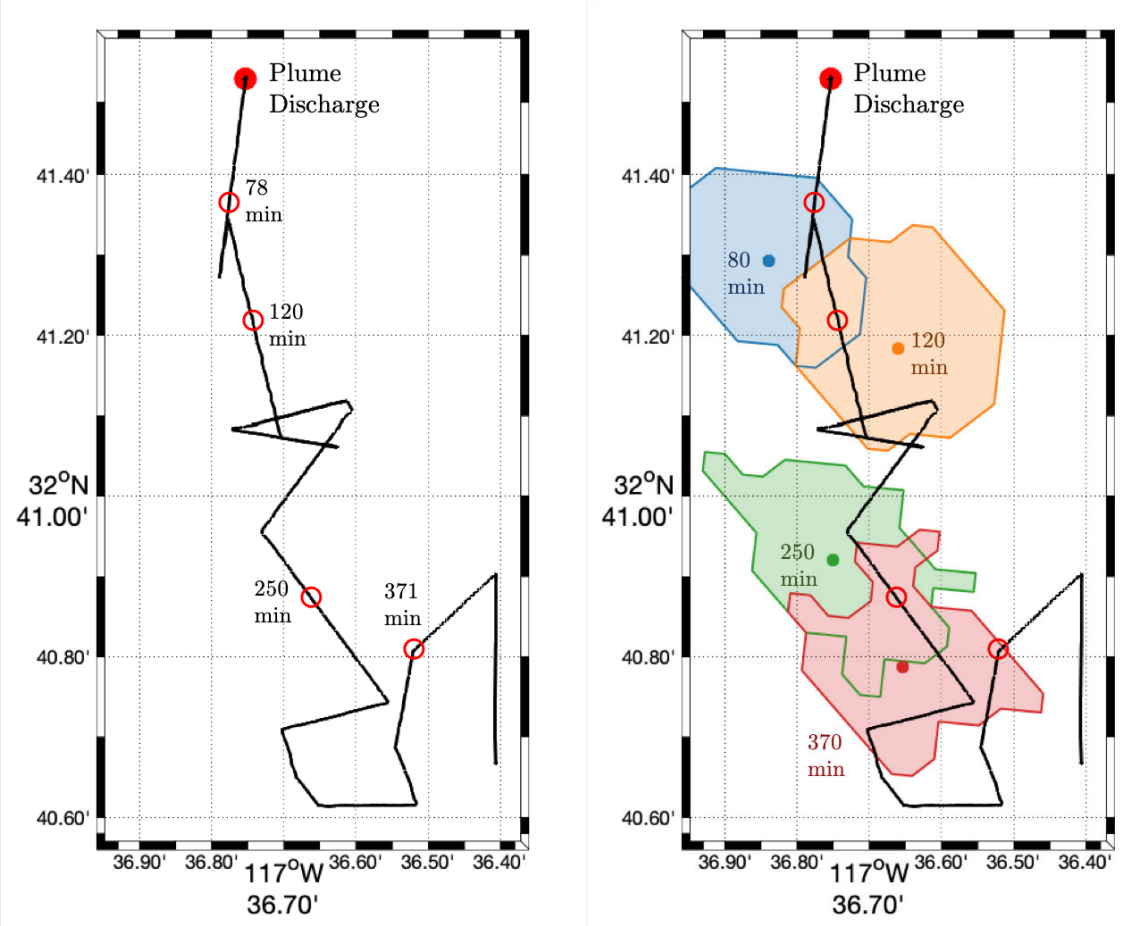
Fig. (4-13b) shows the plumes as predicted by our method of composition at the considered time instants along with the corresponding observations. Specifically, the blue patch shows the plume location at 80 minutes after the release, the orange patch at 120 minutes after the release, the green patch at 250 minutes after the release, and finally the red patch at 370 minutes after the release. We predict that the plume travels about 0.44 km to the south-southwest in the first 80 minutes, followed by a southeastward motion of about 0.34 km in the next 40 minutes. The plume then again

travels south-southwest for about 0.49 km in the next 130 minutes. The final plume location is predicted to be at about 1.39 km to the south of the initial release point after 370 minutes from the release time. We find that the forecast plume location match well with the actual measurements. Specifically, we observe that the plume location predicted by our forecasting methodology contains the actual observation locations for the 80, 120, and 370 minute observations. The distances between the centroids of the forecasted plumes and the corresponding observations are 0.16 km, 0.13 km, 0.25 km, and 0.17 km for 80 minute, 120 minute, 250 minute, and 370 minute timestamps (since the initial release time), respectively. Note that these deviations are of the order of the size of a single grid cell. Our forecasting system is thus able to predict the general trend of the plume transport very well. All the results in Fig. (4-13b) are a vertical projection of the three-dimensional plume onto the horizontal plane. Since the plume size is comparable to the model resolution, the numerical errors may have a non-negligible role in the net plume area in the horizontal.

Considering that the our numerical modeling system did not use any of the ADCP data or other shipboard instruments in real-time, these results are promising for future high-resolution modeling integrated with plume monitoring and serve as a great experimental validation of the composition-based advection method in real time.

4.4 Summary

The new method of flow map composition for computing advective tracer transport and its capabilities are thoroughly illustrated through a wide set of examples in this chapter, including new benchmark problems for advection-diffusion-reaction schemes. The advection and advection-diffusion in an analytical reversible swirl flow and a flow undergoing sudden expansion verify the behavior of errors, inclusion of diffusion and source terms, and the implementation of various tracer boundary conditions. The advection of sediment plumes by ocean currents resulting from potential deep sea mining operations in the Bismarck Sea are simulated for two possible mining sites. We find that, even for low-order schemes, advection using the method of composi-



(a) The ship track overlaid with plume observation locations and timestamps

(b) Predicted (vertically averaged) plume location at the four observation times

Figure 4-13: The observed (hollow red circles) and predicted (colored patches) plume locations and (vertically averaged) spread at 78, 120, 250, and 371 minutes after the start of plume discharge. It can be see that the observed plume locations lie within the plume spread predicted by our composition-based advection methodology.

tion yields minimal total numerical errors, much smaller than the errors of high-order Eulerian schemes, and with a spatial coverage much better than particle-based methods. Finally, we study the results from the 2018 PLUMEX real-time sea exercise. We compare the real-time 3D predictions of sediment plume transport made by the method of composition with the experimentally observed data to find a satisfactory match between the predictions and the observations, even with limited feedback and coarse simulation resolution.

Chapter 5

Lagrangian Analysis of Material Transport - Theory

In this chapter, we first utilize the method of flow map composition to compute the flow maps in dynamic open domains with multiple time-dependent inlets and outlets. We then develop theory and schemes to extract and compute material subregions in dynamic fluid flows that remain the most or the least coherent throughout the time interval of interest. Such subregions are extremely important to quantify fluid and tracer mixing and dispersion in ocean flows.

First, in Sec. 5.1, we briefly review the existing results in Lagrangian coherence and show how the most prominent coherence metrics are derived from the flow map, highlighting the importance of the flow map as the backbone of material transport. In Sec. 5.2, we discuss we state the problem statement and provide some preliminaries. Sec. 5.3 extends the method of composition to compute flow maps in dynamic open domains with time-dependent inlets and / or outlets, which is often required for realistic marine scenarios. We propose a new metric called *extended polar distance* in Sec. 5.4 to extract subregions of the fluid domain that exhibit persistent coherence or rigidity, *i.e.* sets that undergo minimal local distortion (stretching or compression) while undergoing arbitrary amounts of translation and rotation over the entire temporal domain of interest. Finally, we detail certain remarks about the objectivity of the proposed coherence measure and methodologies for its efficient computation using

the method of composition. We also relate this proposed metric with the expected strain on the material subregions and prove that persistently coherent sets maximally resist chaotic advection and diffusive mixing. This work is the theoretical basis of Kulkarni and Lermusiaux [136].

5.1 Introduction

Coherent structures have been extensively employed in fields such as fluid mechanics and dynamical systems to understand and study a wide range of phenomena ranging from the formulation of reduced order models of the flow [65] to understanding the material transport pathways [52; 222]. Specifically, studying the role of coherent structures as the 'skeleton' of Lagrangian material transport in fluid dynamic systems has been an active area of research. There exist several notable approaches to extract coherent structures in a flow, most often described through the coherent and incoherent sets and attracting and repelling barriers [93]. Many traditional methods for this purpose utilize an Eulerian approach by looking at individual snapshots of the flow field to analyze the coherent structures through streamlines, velocity stagnation points etc. [186; 256; 113; 120]. Although such approaches may be sufficient for steady flows (as the streamline structure does not change with time), they cannot capture the complex transport structures and features of material transport in unsteady flows.

The foundations of quantifying Lagrangian transport of passive materials in time-dependent flows over finite duration were laid by Haller and colleagues, who also coined the term *Lagrangian coherent structures* (LCSs) [103]. In this perspective, the attracting (stable) and repelling (unstable) manifolds are also advected with the underlying flow and hence are considered 'material surfaces', resulting in a much more rich and complex transport structures. The viewpoint in this setting is inherently Lagrangian, as it explicitly tracks the motion of individual parcels and their trajectories that are governed by an unsteady velocity field.

LCSs have been very helpful in improving our understanding of material transport

over finite durations in unsteady fluid flows [101]. It has long been known that such coherent structures are an intrinsic property of fluid flows [197] and they often refer to the distinguished (attracting or repelling) material surfaces or sub-domains in the fluid flow over the time of interest [59; 102]. There is a wide range of LCS applications, ranging from fluid dynamic and aerodynamic analyses [173; 56; 124], combustion studies [33; 27; 194] to biological and bio-inspired fluid flows [223; 86; 226]. However, the most prominent studies and applications of Lagrangian analyses have been in the field of geophysical and marine sciences. These include oil spill and spread predictions, pollution tracking, environmental hazards studies, and marine-biology studies [143; 217; 101; 117; 187] to name a few. Several definitions of LCS have been put forth by various researchers [59; 76; 101; 127; 224] along with multiple methodologies and schemes to extract the LCS from the given flow fields [4; 93].

So far, popular techniques for LCS studies have suggested that such relevant features can be co-dimension one (*i.e.* of dimension 1 less than that of Ω) surfaces across which the underlying flow map exhibits a discontinuity or a sharp gradient. Indeed, parcels that are located from either side of a discontinuity surface have diverging trajectories. Hence these co-dimension one LCSs, typically extracted from Finite Time Lyapunov Exponent (FTLE) fields [224] or from tensorlines of the flow map Jacobian [100; 104], exhibit extremal properties of repulsion or attraction, ideally globally, or at least in some neighborhood of these surfaces. These methods have the advantage of yielding LCS that can be computed at a relatively low cost even for highly resolved velocity fields, but may also have the flaw of offering hardly interpretable pictures when too many LCS are found in the domain [64; 66]. Recently, there has been significant interest in finding coherent / almost invariant material sets in autonomous and non-autonomous fluid flows. Some approaches determine these sets by partitioning the domain into sets that may have heavy mixing within, but that mix very slowly with their complement [71; 78; 74]. Other proposed methodologies include spectral clustering [92], fuzzy clustering [4], and braid theory [5]. The reader is pointed to recent articles [93; 250] for a critical comparison of these methods over a variety of simulated and realistic flow fields.

A common drawback of these methods is that they study coherence exclusively between the start and the end time. That is, a fluid set is considered coherent if its initial and final shapes are similar, even though it could undergo heavy distortion during the time interval of interest. There has been an interest in developing methodologies that consider the coherence of sets through the entire time interval. We refer to such sets that undergo minimal stress or distortion during the entire time of interest as 'persistently coherent sets'.

In this chapter, we propose an approach to compute persistently coherent sets in dynamic fluid flows. We discuss the theoretical underpinnings of the proposed objective metric and approaches for its efficient computation, with special attention towards realistic flow fields, including high-resolution ocean simulations with multiple time-dependent inlets and outlets.

5.1.1 Types of Material Coherence

As stated before, most techniques to compute coherent material subdomains consider the coherence or rigidity of fluid sets by comparing their configurations only at the initial and the final times. These methods determine material sets that have similar shape at the start and the end times, or that undergo minimal net strain. However, the possible deformations that these sets may undergo during the entire time interval of interest are not considered. There may be cases where the material set first undergoes severe stretching followed by compression to end up in a shape similar to its initial one. In real fluids, such transformations could not be defined as coherent since the set would be more prone to small-scale (turbulent) diffusion at its boundaries when in the intermediate stretched state (and thus may not actually remain coherent). These large-gradient effects are unaccounted for by theories focusing only on advective transport, but it would often be inaccurate to define such transformation as remaining coherent. However, such a set is more prone to small-scale diffusion at the boundaries (and thus may not actually remain coherent), which is unaccounted for by theories focusing only on advective transport [106].

Fig. (5-1) shows the possibilities for the evolution of a generic material set in an

unsteady fluid flow. Certain material sets will be continually distorted by the fluid, and will typically undergo a net non-zero strain. We term these as 'incoherent sets' or 'non-rigid sets', and these are shown in the yellow panel of Fig. (5-1). The second kind, called 'coherent sets' or 'rigid sets', undergo non-zero distortion at intermediate times, but these distortions cancel out over the total time of interest, and these sets return to their original shape at time $t = T$, as seen in the green panel. Finally the third kind, *i.e.* 'persistently coherent sets' or 'persistently rigid sets' are shown in the red panel. These sets maintain their shape throughout the entire time of interest. At any intermediate time, their shape is similar to (within certain tolerance) their initial shape. Persistently coherent sets fall under the umbrella of coherent sets for obvious reasons, however not all coherent sets are persistently coherent.

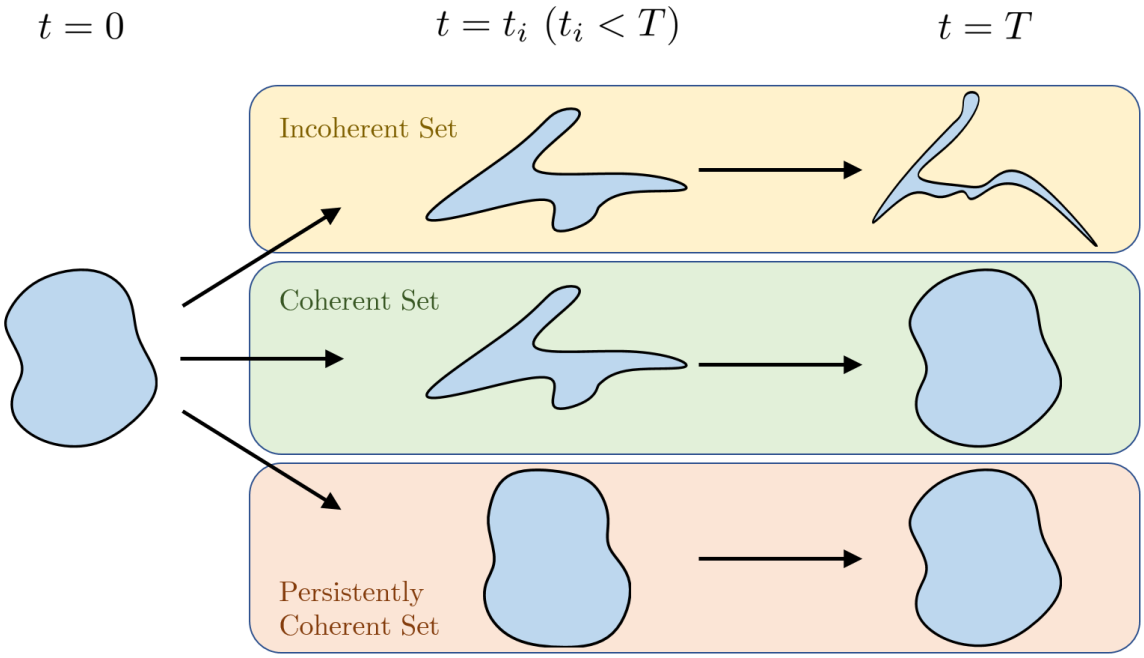


Figure 5-1: Schematic of the different types of coherent / incoherent sets in fluid flows. We assume that a material set of fluid starts with a particular shape at time $t = 0$. The yellow panel shows the evolution of this set as an incoherent set at times t_i ($< T$) and T, the green panel depicts its evolution as a coherent set, and the red panel showcases its evolution if it were to be a persistently coherent set. Persistently coherent sets are coherent sets, but not vice versa.

As flow maps quantify the generic advective transport features between the considered time instants, they are a key element in our methodology to determine persis-

tently coherent material sets. To achieve this, we use the method of composition to effectively compute and extract information from flow maps at all intermediate times. As the persistently rigid sets never undergo any appreciable strain / distortions, they are also the least prone to small-scale (turbulent) diffusion processes. Further, simply by inverting the developed criterion, we can also determine material sets that are the most incoherent, or are distorted the most. We can capture such sets by searching for the material sets that are either prone to large stretching / compression or to diffusive distortion or both. Such material sets with large gradients are often the primary regions where small-scale diffusion and chaotic and turbulent mixing occur. It is thus important to predict their existence and location.

5.1.2 Connections Between Prominent Lagrangian Coherence Metrics and the Flow Map

Let us now review some popular metrics to compute LCSs their connections to the flow map. This cements the role of flow maps as the central component of Lagrangian transport analysis. Following the discussion from Sec. 2.4, the major approaches of extracting LCSs from dynamic flows can be broadly classified into two categories: (i) those that attempt to extract the relevant features of the flow map fields, and (ii) those that focus on the action of the flow map operator on density distributions [64].

We first discuss the Lagrangian coherence metrics based in analyzing the exceptional features of the flow map fields. Specifically, we first analyze the criteria derived from the (right) Cauchy-Green (CG) strain tensor. We then focus our attention on the variational methods to determine LCSs. Finally, we look at the operator theoretic methods to determine LCSs.

The (right) CG strain tensor $C_0^T(\boldsymbol{x}) \in \mathbb{R}^{n\times n}$ is defined by Eq. (5.1) (where n is the spatial dimension of the system, *i.e.* $n = 2$ or 3),

$$C_0^T(\boldsymbol{x}) = \left[\nabla \boldsymbol{\phi}_0^T(\boldsymbol{x})\right]^* \left[\nabla \boldsymbol{\phi}_0^T(\boldsymbol{x})\right] \ . \tag{5.1}$$

It specifies the local strain that an infinitesimal material element at $\boldsymbol{x}$ experiences

between times 0 and T. It can be shown that $C_0^T(\boldsymbol{x})$ is a symmetric positive definite matrix for all $\boldsymbol{x}$ and T [101]. As $C_0^T(\boldsymbol{x})$ is the strain tensor, its eigenvectors and eigenvalues denote the principal directions and magnitudes of the strain respectively. One can easily compute these quantities using the gradient of the flow map.

Let us assume that the singular value decomposition (SVD) of the gradient of the flow map vector (Jacobian matrix of the flow map) is given by $\nabla \boldsymbol{\phi}_0^T(\boldsymbol{x}) = U^{(0,T)}\Sigma^{(0,T)}\left(V^{(0,T)}\right)^*$, where $U^{(0,T)}, V^{(0,T)} \in \mathbb{R}^{n\times n}$ are orthogonal matrices and $\Sigma^{(0,T)} \in \mathbb{R}^{n\times n}$ is a diagonal matrix whose entries $\sigma_1^{(0,T)}, \ldots, \sigma_n^{(0,T)} \geq 0$ are in decreasing order of magnitude. The columns of $U^{(0,T)}$, denoted by $U_1^{(0,T)}, \ldots, U_n^{(0,T)}$ are the left singular vectors of $\nabla \boldsymbol{\phi}_0^T$ and the columns of $V^{(0,T)}$, denoted by $V_1^{(0,T)}, \ldots, V_n^{(0,T)}$ are the right singular vectors of $\nabla \boldsymbol{\phi}_0^T$. Note that $U_j^{(0,T)}, V_j^{(0,T)}$, and $\sigma_j^{(0,T)}$ are a function of $\boldsymbol{x}$ for all $j = 1, \ldots, n$. The CG strain tensor can be written as $C_0^T(\boldsymbol{x}) = V^{(0,T)}\left(\Sigma^{(0,T)}\right)^2\left(V^{(0,T)}\right)^*$, implying that its eigenvectors are $V_1^{(0,T)}, \ldots, V_n^{(0,T)}$ and its eigenvalues, denoted by $\lambda_j^{(0,T)}\ \forall\ j = 1, \ldots, n$, are $\left(\sigma_1^{(0,T)}\right)^2, \ldots, \left(\sigma_n^{(0,T)}\right)^2$. Similarly, the eigenvalues of $C_T^0 = [\nabla \boldsymbol{\phi}_T^0]^*\,[\nabla \boldsymbol{\phi}_T^0]$, denoted by $\lambda_j^{(T,0)}\ \forall\ j = 1, \ldots, n$, are equal to $\left(\sigma_1^{(T,0)}\right)^2, \ldots, \left(\sigma_n^{(T,0)}\right)^2$. Invariants of the CG strain tensor (such as its eigenvalues) do not change under Euclidean change of coordinates [103]. Thus they tend to be ideal candidates to quantify material stretching and distortion.

Finite Time Lyapunov Exponent (FTLE) and Polar Rotation Angle (PRA)

The FTLE is a popular objective metric used to approximate the hyperbolic LCS in a flow (*i.e.* attracting and repelling manifolds) [101; 14; 225]. It is simply a logarithmic rescaling of the largest eigenvalue of the CG strain tensor. One can write the forward (repelling) FTLE in terms of σ_1 as given by Eq. (5.2); similar analogue holds for the backward (attracting) FTLE:

$$FTLE_0^T(\boldsymbol{x}) = \frac{1}{T}\log(\sigma_1^{(0,T)})\,. \tag{5.2}$$

It is clear that the FTLE field estimates the maximal local stretching experienced by a fluid parcel. Ridges of the forward and backward FTLEs, *i.e.* lines where the

forward and backward FTLE is locally maximal, have been used to define repelling and attracting LCS [224], respectively.

The polar rotation angle (PRA; denoted by $\theta_0^T(\boldsymbol{x})$) attempts to capture the elliptic LCS, *i.e.* material vortices in a flow, by identifying material elements that rotate by similar angles about some points over the considered time interval [62]. PRA can be computed from $\nabla\boldsymbol{\phi}_0^T(\boldsymbol{x})$ as specified by Eq. (5.3):

$$\cos\left(\theta_0^T(\boldsymbol{x})\right) = \frac{1}{2}\left(\sum_{j=1}^{n} U_j^{(0,T)} V_j^{(0,T)} - 1\right). \tag{5.3}$$

It is proven that the level curves of PRA are objective only for 1D and 2D flows but not objective in 3D flows, and better measures have been recently proposed to identify material vortices [105].

Variational Methods

Variational methods capture distinguished material surfaces across which a feature of the deformation field shows no leading-order change. Hyperbolic and parabolic LCSs are captured by identifying curves / surfaces for whom the variation of shear strain is an order of magnitude less than that of any neighboring curves / surfaces. Elliptic LCSs are identified as closed curves / surfaces across which the normal stretching varies by an order of magnitude less than any neighboring curves / surfaces.

In 2D, variational shearless LCSs are defined through the direction fields formed by the second eigenvectors of $C_0^T(\boldsymbol{x})$. Specifically, repelling LCSs are the trajectories of $\frac{d\boldsymbol{x}}{dt} = V_1^{(0,T)}(\boldsymbol{x})$ starting from the local maxima of $\lambda_2^{(0,T)}(\boldsymbol{x})$ and attracting LCSs are the trajectories of $\frac{d\boldsymbol{x}}{dt} = V_2^{(0,T)}(\boldsymbol{x})$ starting from the local minima of $\lambda_1^{(0,T)}(\boldsymbol{x})$.

Similarly, variational elliptic LCS are obtained by solving for the direction field Eq. (5.4), where $\lambda = 1$ yields perfectly elliptic LCS (while they are undefined at

locations where $\sigma_1^{(0,T)} = \sigma_2^{(0,T)}$):

$$\eta^{\pm}_{[0,T]}(\boldsymbol{x}) = \sqrt{\frac{\left(\sigma_2^{(0,T)}\right)^2 - \lambda^2}{\left(\sigma_2^{(0,T)}\right)^2 - \left(\sigma_1^{(0,T)}\right)^2}} V_1^{(0,T)} \pm \sqrt{\frac{\lambda^2 - \left(\sigma_1^{(0,T)}\right)^2}{\left(\sigma_2^{(0,T)}\right)^2 - \left(\sigma_1^{(0,T)}\right)^2}} V_2^{(0,T)}, \tag{5.4}$$

$$\text{for } \lambda \in \left(\left(\sigma_1^{(0,T)}\right)^2, \left(\sigma_2^{(0,T)}\right)^2\right).$$

Further details may be found in [105; 60].

Transfer Operator Based Methods

Finally, we summarize another major direction of research that uses the transfer operator, also called the Parron–Frobenius operator, to compute LCSs in dynamic flows [74; 78; 71; 64; 73]. These approaches is fundamentally different from the earlier approaches, as they analyze the features of the flow map from an operator theoretic aspect. These methods can also be thought of as probabilistic approaches to study material transport and quantify the LCSs through coherent and incoherent material sets. Coherent sets are defined as those material sets that may have mixing within the set but minimally mix with their complement. While we do not delve into the specifics of the different approaches in this area, we simply show how the transfer operator can be computed from the flow map and refer the reader to [70; 77; 75; 72; 65] for the details. The transfer operator between times 0 and T, denoted by P_0^T, is defined by Eq. (5.5).

$$P_0^T\left(\mu(\boldsymbol{x})\right) = \frac{\mu\left(\boldsymbol{\phi}_T^0(\boldsymbol{x})\right)}{\det\left(\nabla \boldsymbol{\phi}_0^T\right)}. \tag{5.5}$$

Here, μ is a density measure on Ω that denotes distribution or concentration of a quantity of interest. The numerator of Eq. (5.5) denotes the advection of the density μ from the initial to the final time due to the movement of fluid parcels, whereas the denominator denotes the change in the area measure of the domain Ω from the initial to the final time.

The transfer operator describes the density value at $\boldsymbol{\phi}_0^T(\boldsymbol{x})$ induced by the flow map. A finite rank numerical approximation for the transfer operator, when the

domain is split into small partitions (typically the numerical grid cells) $\{\mathcal{A}_1, \ldots, \mathcal{A}_{\mathcal{P}}\}$ is given by Eq. (5.6), where m denotes Lebesgue measure associated with μ:

$$\left[P_0^T\right]_{ij} = \frac{m\left(\mathcal{A}_i \cap \boldsymbol{\phi}_T^0\left(\mathcal{A}_j\right)\right)}{m\left(\mathcal{A}_i\right)}\,. \tag{5.6}$$

Essentially, $\left[P_0^T\right]_{ij}$ quantifies the fraction of the quantity of interest that starts in $\mathcal{A}_i$ at the start time and ends up in $\mathcal{A}_j$ at the end time. the Lebesgue measure value $m\left(\mathcal{A}_i\right)$ can be thought of as the amount of the quantity of interest originating in $\mathcal{A}_i$ that is advected by the velocity field. Similarly, $m\left(\mathcal{A}_i \cap \boldsymbol{\phi}_T^0\left(\mathcal{A}_j\right)\right)$ denotes the amount that starts in $\mathcal{A}_i$ and ends up in $\mathcal{A}_j$. If $\mu(\boldsymbol{x}) = \boldsymbol{x}\ \forall\ \boldsymbol{x} \in \Omega$, then $\left[P_0^T\right]_{ij}$ simply denotes the fraction of fluid parcel trajectories that start in $\mathcal{A}_i$ and end up in $\mathcal{A}_j$.

From the above definitions, it can be seen that most of the prominent techniques for detecting LCSs and coherent material sets are rooted in the flow map. Once the flow map of the flow field over the time interval of interest is known, these material transport metrics can be readily computed. This reinforces our notion that flow maps can be thought of as the backbone of Lagrangian material transport in dynamic flows.

5.2 Problem Statement

In this section, we state the problem statement using the notation from Table 2.1. We assume that the spatial domain Ω is open, with multiple time-dependent inlets and outlets. That is, in general, there is unrestricted fluid and material inflow and outflow through the domain boundary $\partial\Omega$. Thus, Eq. (2.11) and Eq. (2.12) imply that the flow maps are not defined over the entire domain at all times. Specifically, the forward flow map is not defined at the locations that leave Ω during $[0, T]$ and the backward flow map is not defined at the locations that enter Ω during the time interval $[0, T]$. We refer to the region of Ω where the flow map is defined as the 'active domain' and denote it by $\Omega_a(t)$. Ω_a is time-dependent, as the region over which the flow map is defined decreases monotonically with time, and $\Omega_a(0) = \Omega$.

Thus, Eq. (2.11) and Eq. (2.12) imply that the flow map is not defined over the

entire domain at all times. Specifically, the forward flow map is not defined at the locations that leave Ω during $[0, T]$ and the backward flow map is not defined at the locations that enter Ω during the time interval $[0, T]$. We refer to the region of Ω where the flow map is defined as the 'active domain' and denote it by $\Omega_a(t)$. Ω_a is time-dependent, as the region over which the forward / backward flow map is defined decreases monotonically with forward / backward time, with $\Omega_a(0) = \Omega$ / $\Omega_a(T) = \Omega$.

Our goal is two fold. First, we utilize and extend the method of flow map composition to accurately and efficiently compute flow maps in open domains, where the actual subdomain over which the flow maps are defined (*i.e.* $\Omega_a(t)$) is of an arbitrary shape that is time-dependent. Our second aim is to derive a criterion, which when thresholded to an appropriate tolerance would yield material subdomains of Ω whose connected components undergo minimal stretching and compression throughout the entire time duration of interest. We hope to relate this tolerance parameter to the expected amount of stretching experienced by the material set. We further enforce that high values of this criterion should indicate material subdomains that undergo the largest amount of distortion, either due to advective stretching or due to smaller-scale diffusion or both. The required criterion should be frame independent (objective), efficiently computable, and applicable to 2D and 3D unsteady flows.

5.3 Flow Map Computation for Open Domains

As seen in Chapter 3, one can compute the backward flow map $\boldsymbol{\phi}_T^0$ by solving Eq. (3.3) forward in time with the initial condition $\alpha_0(\boldsymbol{x}) = \boldsymbol{x}$. Similarly, the forward flow map $\boldsymbol{\phi}_0^T$ is obtained by solving Eq. (3.4) backward in time with $\alpha_T(\boldsymbol{x}) = \boldsymbol{x}$. As these differential equations are purely hyperbolic, boundary conditions are not required if the domains are closed, *i.e.* the velocity vectors everywhere on the boundary $\partial\Omega$ have no component along the local normal direction $\hat{\mathbf{n}}_{\partial\Omega}$.

However, for typical realistic cases, when there is either fluid inflow or outflow, open boundary conditions are necessary. Further, as mentioned before, the forward flow map is undefined at locations that exit the domain Ω during $[0, T]$, whereas the

backward flow map is undefined at locations that enter the domain Ω during $[0, T]$. This implies that the shape of the active domain Ω_a (*i.e.* domain over which the flow map is defined) is different for forward and backward flow maps and it varies with time. Thus, adapting the PDE-based method of composition to compute flow maps over such changing and shrinking domains requires careful design and imposition of very specific boundary conditions. As mentioned in Sec. 3.3.4, previous work in the area of PDE-based flow map computation [156; 157] has largely been applied to closed domains where no boundary conditions are required, and the examples that do consider open boundaries [156; 157] only apply a very specific boundary condition (Eq. (3.13)). These boundary conditions, although well-posed, are incorrect due to the fact that new positions are allowed to enter the domain. They do not distinguish between the positions that start in the domain at the initial time and those that enter the domain at intermediate times, and can thus lead to incorrect results.

5.3.1 Flow Map Open Boundary Conditions

The correct open boundary conditions can be obtained by considering the Lagrangian viewpoint of flow map computation. While computing the flow maps using the particle transport Eq. (2.1), all the trajectories that exit Ω at any time are discarded and never reconsidered. The flow maps corresponding to these trajectories are undefined. Further, no new trajectories are allowed to enter Ω at any point in time. The PDE-based flow map computation (Sec. 3.3.1) computes the flow maps by advecting positions over a fixed spatial domain in an Eulerian setting. Thus, we can classify these positions into four distinct categories based on the above-mentioned Lagrangian interpretation. The first type, called 'inside-inside' positions are the positions that start inside Ω and also end up inside Ω. The second type, 'inside-outside' positions are the ones that start inside Ω but leave the domain during the considered time interval. Conversely, 'outside-inside' positions start outside Ω but enter it during the considered time interval. Finally, 'outside-outside' positions start and also end up outside Ω. The PDE-based method requires backward (forward) advection to compute the forward (backward) flow map. Thus, the inside-outside positions (*i.e.* the ones that

leave Ω) correspond to Lagrangian trajectories that start outside Ω but end up inside. As the Lagrangian trajectories starting outside Ω have no bearing on the flow map definition inside Ω, the corresponding inside-outside positions should freely leave Ω in the PDE-solve. Conversely, the outside-inside positions (*i.e.* the ones that enter Ω) correspond to trajectories that start inside Ω but end up outside. As the flow map is not defined for such trajectories, the corresponding outside-inside positions should be prohibited from entering Ω in the PDE-solve. Thus the only locations where the flow maps are well-defined correspond to the positions that start inside Ω and also end up inside Ω. We refer to this as the *inside-inside rule*. This duality between the Lagrangian trajectories and the Eulerian positions is summarized in Table 5.1. Even though both the flow maps are only defined for the inside-inside positions, the

Table 5.1: Equivalence between the Lagrangian trajectories and the Eulerian positions.

Name	Lagrangian Trajectory	PDE Position
Inside-outside position	Start outside Ω, end up inside.	Start inside Ω and leave it during $[0, T]$.
Outside-inside position	Start inside Ω, end up outside.	Start outside Ω and enter it during $[0, T]$.
Inside-inside position	Start and end up inside Ω.	Start inside Ω and end up inside Ω.
Outside-outside position	Start and end up outside Ω.	Start outside Ω and end up outside Ω.

forward flow map is defined at the initial locations of these positions, whereas the backward flow map is defined at their final locations. Hence, the domain of definition of the forward and the backward flow maps are different from each other.

These open boundary conditions imply that there may be regions of domain without any tracer present (*i.e.* locations that are contained in Ω but not in Ω_a). Further, as the tracer is free to flow out, but prohibited to enter, the amount of tracer inside the domain is never-increasing, and hence the active domain Ω_a shrinks over time. Thus, the flow map PDEs Eq. (3.3) and Eq. (3.4) need to be solved on an increasingly irregular and shape-changing domain.

If the solver used for Eq. (3.3) or Eq. (3.4) is able to handle shape changing do-

mains then the implementation is straightforward, as the effective domain shape is readily available through the previous computations. However, this eliminates the parallelization capabilities of the method of composition, as now the resultant domain is an outcome of the previous flow map computations and the corresponding compositions. Hence, the individual flow map computations are not independent of each other and a sequentiality is established through the domain shape requirement.

We propose a novel approach to compute the active domain Ω_a by considering a new 'mask' field along with the flow map. This approach does not require the solver to adapt to shape changing domains, and can be used on prior-defined grids and domains. Further, it does not inhibit the parallelization capabilities of the proposed method. Intuitively, the mask field represents regions in the domain where the flow map is defined. The mask field is a function $\mathcal{M}(\boldsymbol{x}, t) : \Omega \to \{0, 1\}$, which is defined to be 1 when the corresponding location belongs to the active part of the domain, and is 0 when the location is outside the active part at time t. That is,

$$\begin{aligned} \mathcal{M}(\boldsymbol{x}, t) = 1 &\iff \boldsymbol{x} \in \Omega_a(t)\,, \\ \mathcal{M}(\boldsymbol{x}, t) = 0 &\iff \boldsymbol{x} \in \Omega \cap (\mathbb{R}^n \backslash \Omega_a(t))\ . \end{aligned}$$

Here, $\mathbb{R}^n \backslash \Omega_a(t)$ denotes the complement of $\Omega_a(t)$. The flow map computation is carried out on the original domain with open boundary conditions (OBCs) given by Eq. (5.7),

$$\text{OBCs: } \begin{cases} \alpha(\boldsymbol{x}, t) = \widehat{\boldsymbol{x}} \text{ if } \boldsymbol{v}(\boldsymbol{x}, t) \cdot \hat{\mathbf{n}}_{\partial\Omega}(\boldsymbol{x}) < 0\,, \\ \text{open (radiation) boundary if } \boldsymbol{v}(\boldsymbol{x}, t) \cdot \hat{\mathbf{n}}_{\partial\Omega}(\boldsymbol{x}) \geq 0\,. \end{cases} \tag{5.7}$$

where $\widehat{\boldsymbol{x}}$ is some fixed value not contained in Ω, and the final flow map is obtained by a bit-wise AND operation with the mask field $\mathcal{M}(\boldsymbol{x}, t)$.

5.3.2 Mask Field for Open Domains

The above operations requires the availability of the mask field $\mathcal{M}(\boldsymbol{x}, t)$. We now show that the evolution of the mask field is also governed by the advective transport Eq. (2.8), with specific well-posed initial and boundary conditions. Let us look at the computation of the mask for the backward flow map (*i.e.* forward advection). Analogous treatment follows for the forward flow map (with appropriate reversal of the temporal coordinate).

We show that the mask field for the backward flow map is obtained by solving the advective transport Eq. (5.8) forward in time, with a specified constant initial condition (1 in our case),

$$\begin{aligned} &\frac{\partial \mathcal{M}(\boldsymbol{x}, t)}{\partial t} + \boldsymbol{v}(\boldsymbol{x}, t) \cdot \nabla \mathcal{M}(\boldsymbol{x}, t) = 0 \,, \\ &\text{Initial condition: } \mathcal{M}(\boldsymbol{x}, 0) = 1 \;\forall\; \boldsymbol{x} \in \Omega \,. \end{aligned} \tag{5.8}$$

The OBCs for the computation are given by Eq. (5.9),

$$\text{OBCs: } \begin{cases} \mathcal{M}(\boldsymbol{x}, t) = 0 \text{ if } \boldsymbol{v}(\boldsymbol{x}, t) \cdot \hat{\mathbf{n}}_{\partial\Omega}(\boldsymbol{x}) < 0 \,, \\ \text{open (radiation) boundary if } \boldsymbol{v}(\boldsymbol{x}, t) \cdot \hat{\mathbf{n}}_{\partial\Omega}(\boldsymbol{x}) \geq 0 \,. \end{cases} \tag{5.9}$$

A simple OBC consist of numerically enforcing a null second-order-derivative, $\frac{\partial^2 \mathcal{M}}{\partial \hat{\mathbf{n}}_{\partial\Omega}^2} = 0$, when $\boldsymbol{v}(\boldsymbol{x}, t) \cdot \hat{\mathbf{n}}_{\partial\Omega}(\boldsymbol{x}) \geq 0$ (i.e. outward flow). However any other consistent OBC formulations including radiation or other semi-provided OBCs can be used [150]. In the Lagrangian sense, these OBCs for Eulerian fields model the trajectories incoming or outgoing the numerical domain.

The fact that $\mathcal{M}(\boldsymbol{x}, t)$ satisfies Eq. (5.8) can be proven as follows. We assume the existence of a globally defined Lipschtiz continuous velocity field $\mathscr{V} : \mathbb{R}^n \times [0, T] \to \mathbb{R}^n$, whose restriction to Ω at time t is the known velocity field $\boldsymbol{v}(\boldsymbol{x}, t)$ for all t (a practical approach for constructing such a velocity field is proposed in Tang et al. [242], for example).

We now consider the advective transport of a tracer field $\widehat{\alpha} : \mathbb{R}^n \to \mathbb{R}^n$, with an

initial value given by Eq. (5.10),

$$\begin{aligned} \widehat{\alpha}(\boldsymbol{x}, 0) &= \boldsymbol{x} \iff \boldsymbol{x} \in \Omega\,, \\ \widehat{\alpha}(\boldsymbol{x}, 0) &= \widehat{\boldsymbol{x}} \iff \boldsymbol{x} \in \mathbb{R}^n \backslash \Omega\,. \end{aligned} \tag{5.10}$$

The restriction of $\widehat{\alpha}(\boldsymbol{x}, 0)$ on Ω yields the requisite initial condition for the computation of the backward flow map on Ω_a, as $\Omega_a(0) = \Omega$. Further, as the advective transport equation Eq. (2.8) is hyperbolic (*i.e.* the equation is reversible in time, with no global irreversible effects such as diffusion), the restriction of $\widehat{\alpha}(\boldsymbol{x}, t)$ on $\Omega_a(t)$ always yields the flow map $\boldsymbol{\phi}_t^0(\boldsymbol{x})$ for all times t. As we solve Eq. (2.8) with $\widehat{\alpha}$ defined over the entire $\mathbb{R}^n$, the domain boundaries lie at infinity. This makes the imposed boundary conditions irrelevant. Further, the positions that initially start outside Ω but now lie inside are obtained by considering the locations where $\widehat{\alpha} = \widehat{\boldsymbol{x}}$, inside Ω. By definition, these are the locations where the mask should have value 0. Thus, we define the mask field as follows:

$$\mathcal{M}(\boldsymbol{x}, t) = \begin{cases} 0 \iff \widehat{\alpha}(\boldsymbol{x}, t) = \widehat{\boldsymbol{x}}\,, \\ 1 \quad \text{otherwise}\,. \end{cases} \tag{5.11}$$

$\mathcal{M}$ has a direct linear dependence on α, and hence it is also governed by the transport equation Eq. (2.8). Also, as $\Omega_a(0) = \Omega$, we have $\mathcal{M}(\boldsymbol{x}, 0) = 1$. Finally, while evolving $\widehat{\alpha}$ on $\mathbb{R}^n$, the tracer values are advected according to the velocity field $\mathscr{V}(\boldsymbol{x}, t)$. We choose $\mathscr{V}$ on $\mathbb{R}^n \backslash \Omega$ such that any parcel with a tracer value different than $\widehat{\boldsymbol{x}}$ that enters $\mathbb{R}^n \backslash \Omega$ (from Ω) is advected to infinity without it re-entering Ω (as is done in Tang et al. [242]). Note that the restriction of $\mathscr{V}(\boldsymbol{x}, t)$ to Ω is still $\boldsymbol{v}(\boldsymbol{x}, t)$. Thus, it is clear that the tracer values entering Ω after $t = 0$ are only $\widehat{\boldsymbol{x}}$. This implies that for $\boldsymbol{x} \in \partial\Omega$ with $\boldsymbol{v}(\boldsymbol{x}, t) \cdot \hat{\mathbf{n}}_{\partial\Omega}(\boldsymbol{x}) < 0$, we have $\widehat{\alpha}(\boldsymbol{x}, t) = \widehat{\boldsymbol{x}}$. Thus, at such $\boldsymbol{x}$, we have $\mathcal{M}(\boldsymbol{x}, t) = 0$. Finally, as tracer evolution is solved over $\mathbb{R}^n$, there is no restriction on the values that leave Ω, as they are advected away by the velocity field $\mathscr{V}$ and never return to Ω. This implies that there exists an open boundary for the tracer values to exit Ω. The above statements imply that the boundary conditions for the mask field

are the ones given by Eq. (5.9), thus completing the proof.

Numerically, the mask field brings forth other issues. Typically, any numerical advection scheme is diffusive and/or dispersive. This implies that the mask field values in Ω are not exactly equal to 0 or 1. In such cases, a heuristic can be used. The potential numerical issues in the application of the mask field are simplified to a great extent by composing the intermediate mask fields, similar to the method of composition. This is due to the following reason: We compute each individual mask field (of length one numerical timestep) independently of the others. As the interval of advection is just one timestep, there is virtually no numerical diffusion and/or dispersion in the field. Finally, while composing these individual mask fields, any point $\boldsymbol{x}$ is considered to be inside $\Omega_a(t)$ if and only if all the individual constituent mask fields have this point inside their corresponding Ω_a. This can be very easily achieved by setting the masked values of the flow map field in Ω (*i.e.* the values of the flow map field over $\Omega \cap (\mathbb{R}^n \backslash \Omega_a(t))$) to be *NaN*. This ensures that whenever the flow map composition operation encounters a *NaN* value in the mask field for any particular location $\boldsymbol{x}$, the corresponding final flow map value will always be *NaN*. Thus, using our method of composition greatly increases the accuracy of PDE-based flow map computation in open domains over longer time intervals, without the need for heuristics.

5.4 Persistent Lagrangian Coherence / Incoherence in Dynamic Flows

We now discuss the quantification of rigid, persistently rigid, and non-rigid material sets in dynamic fluid flows. We propose new mathematical criteria which define these sets when thresholded appropriately. The criteria are Lagrangian, objective, and can be efficiently computed through flow map composition. Unlike most existing Lagrangian measures, these criteria can be used to detect both rigid and non-rigid sets - low values of these coherence criteria indicate rigid/persistently rigid sets whereas

high values of these criteria indicate most non-rigid/incoherent sets. We also show how the obtained rigid material sets are the ones that maximally resist chaotic advection and material diffusion.

5.4.1 Polar Distance as a Coherence / Incoherence Metric

An infinitesimal vector $\xi_0(\boldsymbol{x})$ at position $\boldsymbol{x}$ at time $t = 0$ is transformed into $\xi_T(\boldsymbol{x})$ at time $t = T$ under the action of the flow map $\boldsymbol{\phi}_0^T$ such that:

$$\langle \xi_T(\boldsymbol{x}), \xi_T(\boldsymbol{x}) \rangle = \langle \nabla \boldsymbol{\phi}_0^T(\boldsymbol{x}) \xi_0(\boldsymbol{x}), \nabla \boldsymbol{\phi}_0^T(\boldsymbol{x}) \xi_0(\boldsymbol{x}) \rangle = \langle C_0^T(\boldsymbol{x}) \xi_0(\boldsymbol{x}), \xi_0(\boldsymbol{x}) \rangle \,. \tag{5.12}$$

This implies that the relative change in the magnitude, *i.e.* the stretching of this vector, is locally maximum if $\xi_0(\boldsymbol{x})$ corresponds to the eigenvector of $C_0^T(\boldsymbol{x})$ with the largest eigenvalue $\lambda_1^{(0,T)}(\boldsymbol{x})$, *i.e.* $\xi_0(\boldsymbol{x}) = V_1(\boldsymbol{x})$. In that case, $|\xi_T(\boldsymbol{x})|^2 = \lambda_1^{(0,T)}(\boldsymbol{x}) |\xi_0(\boldsymbol{x})|^2$. One can obtain the finite time Lyapunov exponent (FTLE) field by logarithmically rescaling the $\lambda_1^{(0,T)}(\boldsymbol{x})$ with the time duration of interest (see Sec. 5.1.2). Another approach defines LCS as integral curves of direction fields obtained from the singular vectors $V_j(\boldsymbol{x})$ (of either the forward or the backward flow map) [101]. Although these curves are locally most repelling when allowing deformations [102], this approach does not yield globally coherent structures since a LCS can be drawn from every point of the domain, and it is unclear how to retain the most influential ones. One can choose the curve passing through the global maxima of the FTLE field [101], however it is not guaranteed that the global-maximum property is maintained all along the curve. Other variants of this approach have been considered in [104] as well as some instantaneous reductions in [221]. Further, although FTLEs quantify ridges along which a fluid parcels experience high shear strain, it is not always true that fluid parcels in areas with low FTLE values experience low shear / distortion. Finally, all of these approaches help us compute material transport barriers that are co-dimension one surfaces (*i.e.* of dimension $n - 1$). However, our goal in this work is to extract coherent / rigid and incoherent / deformed material sets, which are subregions of Ω and hence have the same dimension as Ω (*i.e.* they are co-dimension zero).

A new criterion called the *polar distance*, first proposed in [64; 66] allows us to extract subdomains (instead of co-dimension one surfaces) that exhibit the highest or the lowest degree of material coherence. Feppon [64] makes use of the work by John [123] that states that any transformation $\mathcal{F}$ is rigid (*i.e.* expressible as a translation plus a rotation) *if and only if* its Jacobian $\nabla\mathcal{F}$ is orthonormal. Applying this general theorem when the transformation at hand is the flow map $\boldsymbol{\phi}_0^T$, we can say that a material set $\mathcal{A} \in \Omega_a(0)$ is perfectly rigid if and only if the flow map Jacobian is orthonormal when restricted over $\mathcal{A}$.

However, it is expected that no material set would exhibit perfect rigidity especially in the case of realistic flows. Thus, we require a quantitative metric to judge the deviation of material sets from perfect rigidity and the associated expected amount of stretching. To this end, [66] shows that a material set is expected to undergo stretching by a factor of $e\ell T$, where ℓ is the characteristic length of the set and $e > 0$ is such that $|\sigma_j - 1| \leq e \;\forall\; 1 \leq j \leq n$. A suggested choice of e is $|\nabla \boldsymbol{v}|_{\infty}^{\Omega\times[0,T]}$ [66]. One can further see that:

$$|\sigma_j - 1| \leq e \implies \left(\sigma_j^2 - 1\right) - 2\left(\sigma_j - 1\right) \leq e^2 \implies \left(\sigma_j^2 - 1\right) \leq e^2 + 2e\,.$$

However, as $\sigma_j^2 = \lambda_j^{(0,T)}$, we have $\left|\lambda_j^{(0,T)} - 1\right| \leq e^2 + 2e \implies \left|\lambda_j^{(0,T)} - 1\right| \lessapprox 2e$ for small e.

This result suggests that the expected amount of stretching of a material set can be quantified in terms of the deviation of the eigenvalues of the Cauchy-Green strain tensor from unity restricted over this set. This motivates us to define this deviation as a metric to determine materially rigid subdomains of a flow domain. Eq. (5.13) defines the resulting *polar distance* metric:

$$\mathcal{P}_0^T(\boldsymbol{x}) = \sum_{j=1}^{n} \frac{\left(1 - \lambda_j^{(0,T)}(\boldsymbol{x})\right)^2}{\lambda_j^{(0,T)}(\boldsymbol{x})}\,. \tag{5.13}$$

Note that our definition of the polar distances defers from that of [64; 66]. This is done to make the polar distance metric equaivalent when computed using the forward and the backward flow maps, and hence truly Lagrangian (see Sec. 5.4.3). However,

we can still relate the tolerance e with this metric as follows:
As we have that $\left|\lambda_j^{(0,T)} - 1\right| \lessapprox 2e$ for small e, we get that $\lambda_j^{(0,T)} \in [1-2e, 1+2e]$,

$$\implies \frac{1}{\sqrt{\lambda_j^{(0,T)}}} \in \left[(1+2e)^{-1/2}, (1-2e)^{-1/2}\right] .$$

Using second-order Taylor approximation,

$$\begin{aligned} \frac{1}{\sqrt{\lambda_j^{(0,T)}}} &\in \left[1 - e + \frac{3e^2}{2}, 1 + e + \frac{3e^2}{2}\right] , \\ \implies \frac{\lambda_j^{(0,T)} - 1}{\sqrt{\lambda_j^{(0,T)}}} &\in \left[-2e + 2e^2 - 3e^3, 2e + 2e^2 + 3e^3\right] . \end{aligned}$$

Thus, for small e, we have that $\left|\frac{\lambda_j^{(0,T)}-1}{\sqrt{\lambda_j^{(0,T)}}}\right| \leq 2e$. This implies:

$$\mathcal{P}_0^T = \sum_{j=1}^{n} \frac{\left(1 - \lambda_j^{(0,T)}\right)^2}{\lambda_j^{(0,T)}} \leq 4ne^2 . \tag{5.14}$$

Rigid sets $\mathcal{A}^{\epsilon,(0,T)}$ (with tolerance ϵ) are obtained as the connected components of:

$$\mathcal{A}^{\epsilon,(0,T)} = \{\boldsymbol{x} \in \Omega \mid \mathcal{P}_0^T(\boldsymbol{x}) \leq \epsilon\} . \tag{5.15}$$

As the polar distance is thresholded above by ϵ, we get that $e = 0.5(\epsilon/n)^{1/2}$ from Eq. (5.14). Thus a connected component of $\mathcal{A}^{\epsilon,(0,T)}$ with a characteristic length ℓ undergoes an expected stretching of $0.5\ell T\sqrt{\epsilon/n}$ over the time interval $[0, T]$.

Analogously, the most incoherent sets, $\mathcal{B}^{\epsilon',(0,T)}$, are obtained by considering locations with high polar distance values and by thersholding the value from below,

$$\mathcal{B}^{\epsilon',(0,T)} = \{\boldsymbol{x} \in \Omega \mid \mathcal{P}_0^T(\boldsymbol{x}) \geq \epsilon'\} . \tag{5.16}$$

5.4.2 Extended Polar Distance as a Persistent Coherence / Incoherence Metric

To determine persistently rigid sets, we need to consider the rigidity of material sets between the start time and all the intermediate times. That is, the values of the polar distance for all intermediate times (*i.e.* $\mathcal{P}_0^t \ \forall \ t \in [0,T]$) must be considered instead of only at the end time. The desired persistent coherence field (that is yet to be defined), denoted by $\overline{\mathcal{P}}_0^T : \Omega_a(T) \to \mathbb{R}$, should only indicate coherence in regions that are observed to be appreciably rigid for all intermediate times. Persistently coherent sets are then obtained by thresholding $\overline{\mathcal{P}}_0^T$ to an appropriate tolerance. We call this metric *extended polar distance*. $\overline{\mathcal{P}}_0^T$ is defined in Eq. (5.17):

$$\overline{\mathcal{P}}_0^T(\boldsymbol{x}) = \int_0^T w(\boldsymbol{x},t)\mathcal{P}_0^t(\boldsymbol{x})dt = \int_0^T w(\boldsymbol{x},t)\left(\sum_{i=1}^{n}\frac{\left(1-\lambda_j^{(0,t)}(\boldsymbol{x})\right)^2}{\lambda_j^{(0,t)}(\boldsymbol{x})}\right)dt\,, \tag{5.17}$$

where $w(\boldsymbol{x},t)$ is a normalized weighing function (*i.e.* $\int_0^T w(\boldsymbol{x},t)dt = 1 \ \forall \ \boldsymbol{x} \in \Omega_a$). This expression considers the coherence between the start time and an intermediate time t, for all $t \in [0,T]$. Typically $w(\boldsymbol{x},t) = \frac{1}{T}$; however unequal weight can be conveniently assigned depending on the specifics of the problem. While computing $\overline{\mathcal{P}}_0^T$, it is extremely important to consider $\mathcal{P}_{t_i}^{t_j}$ where at least one amongst t_i and t_j is either the start time or the end time. This way, it is ensured that the coherence at intermediate times is measured only with respect to the start or the end times. If coherence is considered over the individual subintervals of $[0,T]$, then the translation and rotation of the individual sets is not respected and only those sets that remain fixed in space and time are detected to be coherent. The numerical version of the extended polar distance (assuming uniform weighing function $w(t)$), denoted by $\tilde{\overline{\mathcal{P}}}_0^T(\boldsymbol{x})$, is given by Eq. (5.18),

$$\tilde{\overline{\mathcal{P}}}_0^T(\boldsymbol{x}) = \frac{1}{N_t}\sum_{i=1}^{N_t}\sum_{j=1}^{n}\frac{\left(1-\lambda_j^{(0,i\Delta t)}(\boldsymbol{x})\right)^2}{\lambda_j^{(0,i\Delta t)}(\boldsymbol{x})}\,. \tag{5.18}$$

Once $\overline{\mathcal{P}}_0^T$ is computed, persistently rigid / coherent sets, denoted $\overline{\mathcal{A}}^{\epsilon,(0,T)}$, can be obtained by thresholding $\overline{\mathcal{P}}_0^T$ from Eq. (5.17) up to a certain tolerance,

$$\overline{\mathcal{A}}^{\epsilon,(0,T)} = \{\boldsymbol{x} \in \Omega \mid \mathcal{P}_0^T(\boldsymbol{x}) \leq \epsilon\}\,. \tag{5.19}$$

Persistently incoherent sets, $\mathcal{B}^{\epsilon',(0,T)}$, are obtained by considering locations with high extended polar distance values and by thersholding the value from below,

$$\overline{\mathcal{B}}^{\epsilon',(0,T)} = \{\boldsymbol{x} \in \Omega \mid \mathcal{P}_0^T(\boldsymbol{x}) \geq \epsilon'\}\,. \tag{5.20}$$

In Sec. 5.4.3 we show that the extended polar distance captures the tendency of a material set to be distorted due to advection and its tendency to be diffused. Thus, the sets that are deemed to be persistently incoherent either undergo large distortions due to advection or are highly susceptible to small-scale diffusion or both.

5.4.3 Remarks

Objectivity

One of the foremost requirements for any Lagrangian coherence measure is objectivity, *i.e.* frame independence [101]. This means that the proposed metric should be appropriately transformed under any Gallilean transformation of the reference frame to detect the same coherent / incoherent material sets. This requirement arises from the fact that should the observer change reference frames, the material sets that are detected to be coherent / incoherent should not change, as they are an intrinsic property of the considered flow field.

Most Eulerian coherence criteria, such as the Okubo-Weiss criterion etc., are not objective. This is exemplified in Fig. (2-1), wherein a material blob considered coherent by the Okubo-Weiss criterion stretches out exponentially fast. In that case, even though the streamlines appear vortical in a stationary frame of reference, they transform into a saddle if one jumps onto a reference frame moving with a constant velocity, justifying the diverging material flow.

Our polar distance and extended polar distance criteria solely rely on the spectrum of the right CG strain tensor. As the spectrum of the CG strain tensor is known to be objective [99; 101; 225], the polar distance and the extended polar distance criteria are also objective.

Forward - Backward Duality

We have utilized the forward flow maps in defining the polar distance (Eq. (5.13)) and the extended polar distance (Eq. (5.17)). However, one can equivalently choose to use the backward flow maps to compute these quantities. A couple of natural questions that arise are: (i) which flow maps should be used to compute these criteria? and (ii) is there any relation between the material sets predicted by these criteria when using the forward flow maps and those predicted when using the backward flow maps?

As shall be seen, the sets obtained through the polar distance and the extended polar distance while using the forward and the backward flow map are equivalent. Specifically, the criteria when using either the forward or the backward flow map predict the same material sets but at different points in time, eliminating any ambiguity.

This can be proven as follows. We know that the forward and the backward flow maps are inverses of each other. That is:

$$\boldsymbol{x}_0 = \boldsymbol{\phi}_T^0\left(\boldsymbol{x}_T\right) = \boldsymbol{\phi}_T^0\left(\boldsymbol{\phi}_0^T\left(\boldsymbol{x}_0\right)\right) . \tag{5.21}$$

Differentiating Eq. (5.21) with respect to $\boldsymbol{x}_0$, we get that:

$$\mathbb{I}_n = \nabla\boldsymbol{\phi}_T^0(\boldsymbol{x}_T)\nabla\boldsymbol{\phi}_0^T(\boldsymbol{x}_0) . \tag{5.22}$$

Eq. (5.22) implies that $\boldsymbol{\phi}_T^0(\boldsymbol{x}_T)$ and $\boldsymbol{\phi}_0^T(\boldsymbol{x}_0)$ are inverses of each other, and thus their singular values are inverses of each other. Thus the eigenvalues of $C_0^T(\boldsymbol{x}_0)$ and $C_T^0(\boldsymbol{x}_T)$, *i.e.* $\lambda_j^{(0,T)}(\boldsymbol{x}_0)$ and $\lambda_j^{(T,0)}(\boldsymbol{x}_T)$ are inverses of each other, given by Eq. (5.23):

$$\lambda_j^{(0,T)}(\boldsymbol{x}_0) = \left(\lambda_j^{(T,0)}(\boldsymbol{x}_T)\right)^{-1} \;\; \forall\; 1 \leq j \leq n . \tag{5.23}$$

Substituting Eq. (5.23) in Eq. (5.13), we obtain:

$$\begin{aligned}
\mathcal{P}_0^T(\boldsymbol{x}_0) &= \sum_{j=1}^{n} \frac{\left(1-\lambda_j^{(0,T)}(\boldsymbol{x}_0)\right)^2}{\lambda_j^{(0,T)}(\boldsymbol{x}_0)} = \sum_{j=1}^{n} \frac{\left(1-\left(\lambda_j^{(T,0)}(\boldsymbol{x}_T)\right)^{-1}\right)^2}{\left(\lambda_j^{(T,0)}(\boldsymbol{x}_T)\right)^{-1}}\,, \\
\implies \mathcal{P}_0^T(\boldsymbol{x}_0) &= \sum_{j=1}^{n} \frac{\left(\lambda_j^{(T,0)}(\boldsymbol{x}_T)-1\right)^2}{\lambda_j^{(T,0)}(\boldsymbol{x}_T)} = \mathcal{P}_T^0(\boldsymbol{x}_T)\,.
\end{aligned} \tag{5.24}$$

Thus, we have that $\mathcal{P}_T^0(\boldsymbol{x}_T) = \mathcal{P}_0^T(\boldsymbol{x}_0) = \mathcal{P}_0^T(\boldsymbol{\phi}_T^0(x_T))$. Rigid sets obtained at time $t = 0$ by thresholding Eq. (5.13) computed using the forward flow map (denoted by $\mathcal{A}^{\epsilon,(0,T)}$) coincide with those obtained at time $t = T$ by thresholding Eq. (5.13) computed using the backward flow map (denoted by $\mathcal{A}^{\epsilon,(T,0)}$), advected backward in time. This is summarized by the polar distance duality of Eq. (5.25),

$$\begin{aligned}
\mathcal{A}^{\epsilon,(0,T)} &= \{\boldsymbol{x}_0 \in \Omega \mid \mathcal{P}_0^T(\boldsymbol{x}_0) \leq \epsilon\} = \boldsymbol{\phi}_T^0\left(\{\boldsymbol{x}_T \in \boldsymbol{\phi}_0^T(\Omega) \mid \mathcal{P}_T^0(\boldsymbol{x}_T) \leq \epsilon\}\right)\,, \\
\implies \mathcal{A}^{\epsilon,(0,T)} &= \boldsymbol{\phi}_T^0\left(\mathcal{A}^{\epsilon,(T,0)}\right)\,.
\end{aligned} \tag{5.25}$$

Similar to the polar distance (Eq. (5.25)), the forward - backward duality for the extended polar distance (when appropriately thresholded to obtain persistently rigid sets) also holds true, given by Eq. (5.26):

$$\begin{aligned}
\overline{\mathcal{A}}^{\epsilon,(0,T)} &= \{\boldsymbol{x}_0 \in \Omega \mid \overline{\mathcal{P}}_0^T(\boldsymbol{x}_0) \leq \epsilon\} = \boldsymbol{\phi}_T^0\left(\{\boldsymbol{x}_T \in \boldsymbol{\phi}_0^T(\Omega) \mid \overline{\mathcal{P}}_T^0(\boldsymbol{x}_T) \leq \epsilon\}\right)\,, \\
\implies \overline{\mathcal{A}}^{\epsilon,(0,T)} &= \boldsymbol{\phi}_T^0\left(\overline{\mathcal{A}}^{\epsilon,(T,0)}\right)\,.
\end{aligned} \tag{5.26}$$

Here, $\overline{\mathcal{A}}^{\epsilon,(0,T)}$ are the persistently rigid sets computed using the forward flow map $\boldsymbol{\phi}_0^T(\boldsymbol{x})$ whereas $\overline{\mathcal{A}}^{\epsilon,(T,0)}$ are the persistently rigid sets computed using the backward flow map $\boldsymbol{\phi}_T^0(\boldsymbol{x})$.

Simply put, these criteria predict the initial configuration of the rigid / persistently rigid sets while using the forward flow maps and predict the final configuration of the same rigid / persistently rigid sets when using the backward flow maps.

Efficient Computation of the Extended Polar Distance

The computation of $\overline{\mathcal{P}}_0^T$ requires the availability of flow maps $\boldsymbol{\phi}_0^t$ for all $t \in [0, T]$. Typically, this can be prohibitively expensive to compute, especially for large domains or longer time durations. However, the method of composition can be effectively used to minimize the extra computation required for Eq. (5.17) or Eq. (5.18).

The computation of $\mathcal{P}_0^t$ only requires the eigenvalues of the corresponding CG strain tensor (C_0^t), which are equal to the squares of the singular values of the gradient of the associated flow map ($\nabla \boldsymbol{\phi}_0^t$). We can use this observation to efficiently compute the extended polar distance. As the final flow map is a composition of the individual flow maps, its Jacobian is simply the multiplication of the corresponding individual flow map Jacobians through chain rule. That is:

$$\begin{aligned}
&\boldsymbol{\phi}_0^i(\boldsymbol{x}) = \boldsymbol{\phi}_{i-1}^i\left(\boldsymbol{\phi}_0^{i-1}(\boldsymbol{x})\right) = \boldsymbol{\phi}_{i-1}^i\left(\boldsymbol{\phi}_{i-2}^{i-1}\left(\ldots\left(\boldsymbol{\phi}_1^2\left(\boldsymbol{\phi}_0^1(\boldsymbol{x})\right)\right)\right)\right) , \\
\implies &\nabla_{\boldsymbol{x}}\boldsymbol{\phi}_0^i(\boldsymbol{x}) = \nabla_{\boldsymbol{\phi}_0^{i-1}(\boldsymbol{x})}\boldsymbol{\phi}_{i-1}^i\left(\boldsymbol{\phi}_0^{i-1}(\boldsymbol{x})\right) \cdot \nabla_{\boldsymbol{x}}\boldsymbol{\phi}_0^{i-1}(\boldsymbol{x}) = \nabla\boldsymbol{\phi}_{i-1}^i \cdot \nabla_{\boldsymbol{x}}\boldsymbol{\phi}_0^{i-1}(\boldsymbol{x}) .
\end{aligned}$$

This can recursively be computed, where the explicit form of $\nabla_{\boldsymbol{x}}\boldsymbol{\phi}_0^i(\boldsymbol{x})$ then becomes:

$$\nabla_{\boldsymbol{x}}\boldsymbol{\phi}_0^i(\boldsymbol{x}) = \prod_{k=1}^{i} \nabla_{\boldsymbol{x}}\boldsymbol{\phi}_{i-k}^{i-k+1}(\boldsymbol{x}) . \tag{5.27}$$

Eq. (5.27) allows us to compute $\nabla\boldsymbol{\phi}_0^i$ on the fly (along with $\boldsymbol{\phi}_0^i$). Similarly, for the backward flow map, we have:

$$\begin{aligned}
&\boldsymbol{\phi}_i^0(\boldsymbol{x}) = \boldsymbol{\phi}_1^0\left(\boldsymbol{\phi}_i^1(\boldsymbol{x})\right) = \boldsymbol{\phi}_1^0\left(\boldsymbol{\phi}_2^1\left(\ldots\left(\boldsymbol{\phi}_{i-1}^{i-2}\left(\boldsymbol{\phi}_i^{i-1}(\boldsymbol{x})\right)\right)\right)\right) , \\
&\nabla_{\boldsymbol{x}}\boldsymbol{\phi}_i^0(\boldsymbol{x}) = \prod_{k=0}^{i} \nabla_{\boldsymbol{x}}\boldsymbol{\phi}_{k+1}^k(\boldsymbol{x}) .
\end{aligned} \tag{5.28}$$

This simultaneous computation of the flow map and its Jacobian greatly reduces the cost of computing the extended polar distance. Further, efficient algorithms such as the one proposed by Golub et al. [84] can be utilized to compute the singular values of $\nabla\boldsymbol{\phi}_0^i$, given the singular values of $\nabla\boldsymbol{\phi}_{i-1}^i$ and $\nabla\boldsymbol{\phi}_0^{i-1}$. Cases where enough

computational power and / or memory both are unavailable can be handled by using approximate techniques. For example, the memory requirement can be decreased by considering the individual flow maps over longer duration (multiple timesteps) such that the number of flow map fields to be stored is reduced.

Equivalence with Existing Measures

We now show that the extended polar distance is closely related to the recently proposed 'diffusion barrier strength' (DBS) [106], which is an objective measure that indicates distinguished material surfaces that are local minimizers of diffusive transport. Analogous to the FTLE field in purely advective flows, the most prevailing diffusion barriers at the initial time are marked by the ridges of $DBS_0^T(\boldsymbol{x})$. These surfaces may undergo arbitrary amounts of advective distortion but the diffusive flux through them is an order of magnitude smaller than any material surface in their neighborhood. However unlike FTLE, DBS is a predictive diagnostic, *i.e.* it does not require any actual diffusive simulation and arises from the construction of diffusion extremizers. The diffusion barrier strength is defined by Eq. (5.29),

$$DBS_0^T(\boldsymbol{x}) = \mathrm{Tr}\left(\overline{\mathcal{T}}_0^T(\boldsymbol{x})\right) = \mathrm{Tr}\left(\frac{1}{T}\int_0^T \mathcal{T}_0^t(\boldsymbol{x})dt\right) = \frac{1}{T}\int_0^T \mathrm{Tr}\left(\mathcal{T}_0^t(\boldsymbol{x})\right) dt\,. \tag{5.29}$$

The tensor $\mathcal{T}_0^t(\boldsymbol{x}) \in \Re^{n\times n}$ is called the 'transport tensor' [106] and is defined in terms of the local diffusion tensor ($\mathcal{D}(\boldsymbol{x}) \in\in \Re^{n\times n}$) as:

$$\mathcal{T}_0^t(\boldsymbol{x}) = \left[\boldsymbol{\phi}_0^t(\boldsymbol{x})\right]^{-1} \mathcal{D}\left(\boldsymbol{\phi}_0^t(\boldsymbol{x})\right) \left[\left(\boldsymbol{\phi}_0^t(\boldsymbol{x})\right)^*\right]^{-1} \tag{5.30}$$

Under an isotropic diffusion assumption (*i.e.* $\mathcal{D}(\boldsymbol{x}) = \mathbb{I}_n \;\forall\; \boldsymbol{x} \in \Omega$), we have:

$$\mathcal{T}_0^t(\boldsymbol{x}) = \left[\boldsymbol{\phi}_0^t(\boldsymbol{x})\right]^{-1} \left[\left(\boldsymbol{\phi}_0^t(\boldsymbol{x})\right)^*\right]^{-1} = \left(C_0^t(\boldsymbol{x})\right)^{-1} \tag{5.31}$$

As the Cauchy-Green strain tensor is always symmetric positive-definite, the eigenvalues of its inverse are the reciprocals of its eigenvalues. Invoking the definition of

the trace of a matrix as the sum of its eigenvalues, we have:

$$\mathrm{Tr}\left(\mathcal{T}_0^t(\boldsymbol{x})\right) = \mathrm{Tr}\left(\left(C_0^t(\boldsymbol{x})\right)^{-1}\right) = \sum_{j=1}^{n} \frac{1}{\lambda_j^{(0,t)}(\boldsymbol{x})}\,. \tag{5.32}$$

Thus, the DBS field under isotropic diffusion is given by:

$$DBS_0^T(\boldsymbol{x}) = \frac{1}{T}\int_0^T \sum_{j=1}^{n} \frac{1}{\lambda_j^{(0,t)}(\boldsymbol{x})} dt\,. \tag{5.33}$$

Let's now show how this result is linked to our extended polar distance by expanding it, assuming uniform weighting,

$$\begin{aligned}
\overline{\mathcal{P}}_0^T &= \frac{1}{T}\int_0^T \mathcal{P}_0^t(\boldsymbol{x})dt\,,\\
&= \frac{1}{T}\int_0^T \sum_{j=1}^{n} \frac{\left(1-\lambda_j^{(0,t)}(\boldsymbol{x})\right)^2}{\lambda_j^{(0,t)}(\boldsymbol{x})} dt\,,\\
&= \frac{1}{T}\int_0^T \sum_{j=1}^{n} \left(\lambda_j^{(0,t)}(\boldsymbol{x}) + \frac{1}{\lambda_j^{(0,t)}(\boldsymbol{x})} - 2\right) dt\,,\\
&= \frac{1}{T}\int_0^T \sum_{j=1}^{n} \lambda_j^{(0,t)}(\boldsymbol{x})dt + \frac{1}{T}\int_0^T \sum_{j=1}^{d} \frac{1}{\lambda_j^{(0,t)}(\boldsymbol{x})} dt - 2n\,.
\end{aligned}$$

Thus, we have that:

$$\overline{\mathcal{P}}_0^T(\boldsymbol{x}) = \mathrm{Tr}\left(\overline{C_0^T}(\boldsymbol{x})\right) + DBS_0^T(\boldsymbol{x}) - 2n\,. \tag{5.34}$$

Here, $\overline{C_0^T}(\boldsymbol{x})$ denotes the averaged CG strain tensor, given by:

$$\overline{C_0^T}(\boldsymbol{x}) = \frac{1}{T}\int_0^T C_0^t(\boldsymbol{x})dt\,. \tag{5.35}$$

Recall that the CG strain tensor $C_0^T(\boldsymbol{x})$ is the instantaneous advective strain at time t experienced by an infinitesimal element initially located at $\boldsymbol{x}$. Thus, the averaged CG strain tensor, *i.e.* $\overline{C_0^T}(\boldsymbol{x})$ denotes the mean advective strain experienced by an infinitesimal element over the entire time interval $[0, T]$. $\mathrm{Tr}\left(\overline{C_0^T}(\boldsymbol{x})\right)$ quantifies the

magnitude of the mean strain through the trace norm of $\overline{C_0^T}(\boldsymbol{x})$.

Eq. (5.34) implies that the extended polar distance is the sum of the trace of the time averaged CG strain tensor and the diffusion barrier strength, offset by a constant $(2n)$. The first part, *i.e.* $\mathrm{Tr}\left(\overline{C_0^T}(\boldsymbol{x})\right)$, quantifies the time averaged strain on the material parcels due to advective transport, and the latter part, *i.e.* $DBS_0^T(\boldsymbol{x})$, captures the tendency of local material diffusion. A fluid set is classified as persistently rigid if and only if the time averaged stretching of the material and its susceptibility to diffusion over the time of interest are both low. Conversely, the set is deemed to be non-rigid if it either undergoes a lot of stretching or is prone to material diffusion during the time interval of interest.

In the case of a 2D incompressible flow, we have that:

$$\det\left(\boldsymbol{\phi}_0^t(\boldsymbol{x})\right) = \lambda_1^{(0,t)}(\boldsymbol{x})\lambda_2^{(0,t)}(\boldsymbol{x}) = 1 \;\forall\; t \in [0,T],\; \boldsymbol{x} \in \Omega_a(t)\,.$$
$$\implies DBS_0^T(\boldsymbol{x}) = \frac{1}{T}\int_0^T \sum_{j=1}^{2} \frac{1}{\lambda_j^{(0,t)}(\boldsymbol{x})} dt = \frac{1}{T}\int_0^T \sum_{j=1}^{2} \lambda_j^{(0,t)}(\boldsymbol{x}) dt = \mathrm{Tr}\left(\overline{C_0^T}\right)\,.$$

Thus the extended polar distance and the DBS in this case are related by:

$$\overline{\mathcal{P}}_0^T(\boldsymbol{x}) = 2DBS_0^T(\boldsymbol{x}) - 4\,. \tag{5.36}$$

That is, the extended polar distance is simply a scaled version of the DBS. This implies that the sets that are deemed to be persistently rigid are also the ones minimally susceptible to material diffusion. Similarly, the sets that are deemed to be non-rigid are also the ones for whom material diffusion is large, as evident from the high DBS value. This is also because the perimeter of non-rigid sets is expected to increase exponentially with time, which indicates enhanced material diffusion.

5.5 Summary

In this chapter, we utilize the method of flow map composition to compute forward and backward flow maps in realistic open domains, with multiple time-dependent in-

lets and/or outlets. This is achieved by defining a new 'mask' field. We prove that this mask field is also governed by the advective transport equation with well-posed initial and open boundary conditions. We further show that the mask field can be accurately computed through composition, and eliminates the need for heuristics to extract active domains from this mask field. Equipped with the ability to exactly compute flow maps in open domains, we develop a mathematical definition of persistent Lagrangian coherence and incoherence. Specifically, our new 'extended polar distance' metric quantitatively identifies material sets in open fluid flow domains that are persistently coherent or the most incoherent. We highlight the objectivity and the duality of the extended polar distance, eliminating any ambiguity and establishing this criterion as a fully Lagrangian coherence metric. We also show how the method of composition can be used to very efficiently compute this extended polar distance. Finally, we derive and describe the connections of the extended polar distance with some existing coherence metrics. We find that the extended polar distance can be expressed as a sum of the time averaged trace of the Cauchy-Green strain tensor and the diffusion barrier strength. The former component quantifies the average strain on the material set whereas the latter denotes the tendency of the material set to diffusive. We thus show that the sets deemed persistently coherent through the extended polar distance are the ones that resist both advective distortion and diffusive mixing. The applications of the developed theory are demonstrated on a wide array of analytical and realistic data-assimilative flow examples in Chapter 6.

Chapter 6

Lagrangian Analysis of Material Transport - Applications

In this chapter, we apply the theory and schemes developed in Chapter 5 in both ideal and realistic flow fields. In Sec. 6.1, we first analyze, compare, and benchmark the method of composition to compute flow maps and coherent structures in open domains. For this purpose, we consider a data-assimilative realistic ocean flow around the island of Palau in the Southern Pacific Ocean. We show that while regular advection schemes require heuristics to determine the active domains and lead to non-physical artifacts in the flow maps and coherent structures, the method of composition is able to accurately compute these quantities. Following this, we apply the computational toolbox based on the theory and schemes developed in Chapter 3 and Chapter 5 to compute determinants of Lagrangian material transport in 2D and 3D in diverse dynamical regimes and in various marine regions around the world. We then focus our attention on determining incoherent, coherent, and persistently coherent material sets in dynamic fluid flows in Sec. 6.2. To apply the related theory and schemes developed in Sec. 5.4, we first consider the analytical double gyre flow, which is a popular test case in the field of Lagrangian coherence due to the chaotic nature of material transport. Using our novel criteria, we find that there are four coherent and two persistently coherent sets in this flow field. We also study the evolution of three of the most incoherent sets in this flow to observe the chaotic transport

behavior. Finally, we determine such different material sets in the realistic multiscale flow around Palau and study their evolution. We find that even though the material sets might be initially close to each other, their evolution can be appreciably different. The results from this chapter appear in Kulkarni and Lermusiaux [136]; Doshi et al. [47]; Lermusiaux et al. [153].

6.1 Flow Maps and Lagrangian Material Transport Studies in the Real Ocean

We first validate and benchmark the extensions to the method of flow map composition to compute flow maps in open domains. To this end, we utilize a realistic data-assimilative flow around the island of Palau in the Southern Pacific Ocean over 6 days. We first describe the considered flow field and then benchmark and compare the computed and the actual active domains and the resulting flow maps and FTLE fields. We observe that the method of composition yields results as accurate as trajectory-based methods, while maintaining global error bounds and uniform spatial coverage.

6.1.1 Realistic Ocean Test Case - Flow Around the Island of Palau in the Southern Pacific Ocean

The island of Palau lies just to the east of the Malay archipelago and sits atop a plateau along Kyushu-Palau ridge with a deep trench to the southeast. The considered marine domain is centered at the Palau archipelago chain with narrow shelves and tides and includes the ridge with steep topography to the deep ocean with eddy fields and broad currents. Across the southern boundary of the domain, the North Equatorial Counter Current (NECC) flows to the east. North of the domain, the North Equatorial Current flows to the west. Between these two, a surface flow impinges on the archipelago, during the period considered from the southeast, and flows around and over the plateau to the northwest, generating vorticity and eddies.

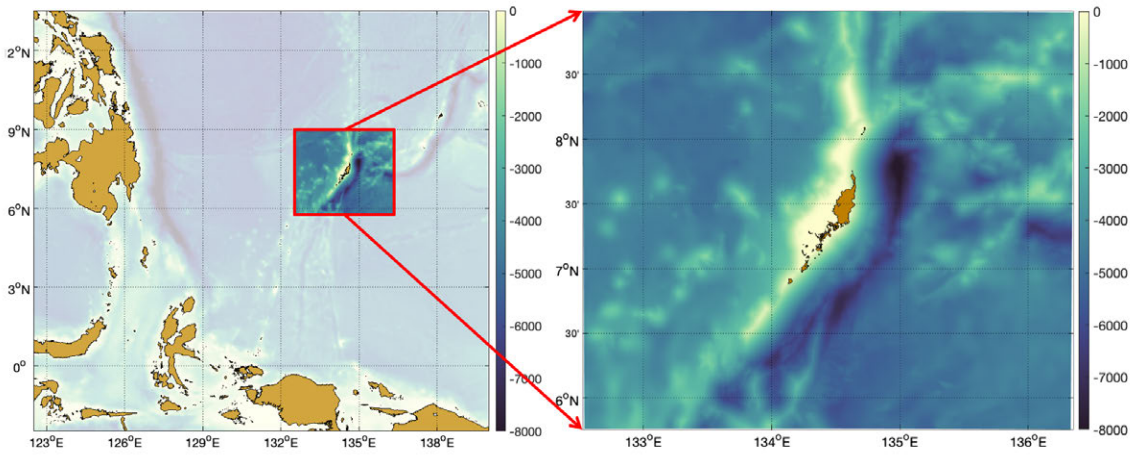


Figure 6-1: The MIT-MSEAS modeling domain around the island of Palau in the Southern Pacific Ocean along with the bathymetry of the region. The left panel shows the relative location of Palau with respect to the Malay archipelago and our modeling domain. The right panel is the island modeling domain, showing the complex shape of the island along with the steep bathymetry drop just southeast of the island.

As before, the dynamic ocean fields (currents, temperature, salinity, and ocean free-surface) are simulated by our MIT Multidisciplinary Simulation, Estimation and Assimilation System (MSEAS) [96; 155; 95] (see Sec. 4.3.1). The present MSEAS simulation domain covers a 420.24 $km \times 358.44$ km region, as seen in Fig. (6-1). The domain is discretized with $N_x = 853$ and $N_y = 728$ and a total integration time of 144 hours (6 days) from 08 May 2015, 00Z to 14 May 2015, 00Z. The initial ocean conditions were downscaled from the coarse analysis fields of 08 May 2015 of the 1/12° HYCOM (Hybrid Coordinate Ocean Model) [41]. The higher-resolution MSEAS simulations were then run up to 14 May 2015, with full tidal forcing [160].

All the boundaries of the domain are open and there is considerable inflow and outflow during the time of interest. Further, the domain involves an obstacle (*i.e.* the island) with complex geometry. Finally, due to the presence of vertical velocities and of a free-surface in the ocean, this flow field is not exactly divergence-free. These factors make it a challenging test case for the theory and schemes developed in this work.

6.1.2 Benchmarking and Comparisons

We now compare and contrast the different PDE-based approaches to compute flow maps in open domains, and benchmark them against the corresponding exact solution which is obtained through high-resolution Lagrangian trajectory advection.

Mask Field Computation in Open Domains

We compute the flow maps in the considered open domain by utilizing the novel mask field approach developed in Sec. 5.3. The mask field allows us to track the shape and the evolution of the active domain Ω_a with time. It satisfies the advective transport equation with well-defined initial and boundary conditions. As described before, the two possible approaches to compute the evolution of the mask field are (i) regular advection and (ii) composition-based advection. Any numerical scheme rooted in difference approximations when used to solve the advective transport equation suffers from numerical diffusion or dispersion. Thus, when numerically computing the evolution of the mask field $\mathcal{M}$ using regular advection, we observe that $\mathcal{M}$ takes all values in $[0, 1]$. However, we require the mask field to be binary at all times (equal to 1 inside Ω_a and 0 outside), and hence heuristics are needed to extract the binary mask field from this diffused mask field. This causes several issues - (i) the extracted mask field tends to be inaccurate and is heavily dependent on the employed heuristic, (ii) based on the heuristic used, one may observe sharp gradients in the flow map fields at the edges of the active domain, which manifest as high FTLE values, incorrectly classifying domain edges as hyperbolic attractors or repellers.

However, by using the method of flow map composition to evolve the mask field itself, the aforementioned pitfalls are greatly overcome. As the interval of numerical advection is just one timestep in the method of composition, there is virtually no diffusion and / or dispersion in the advection solve. For the composition (interpolation) step, one can resort to an interpolation method that maintains the binary nature of the mask field at all times. This ensures that the mask field is binary at all times and no heuristics are needed. Thus, we do not observe sharp gradients in the flow maps

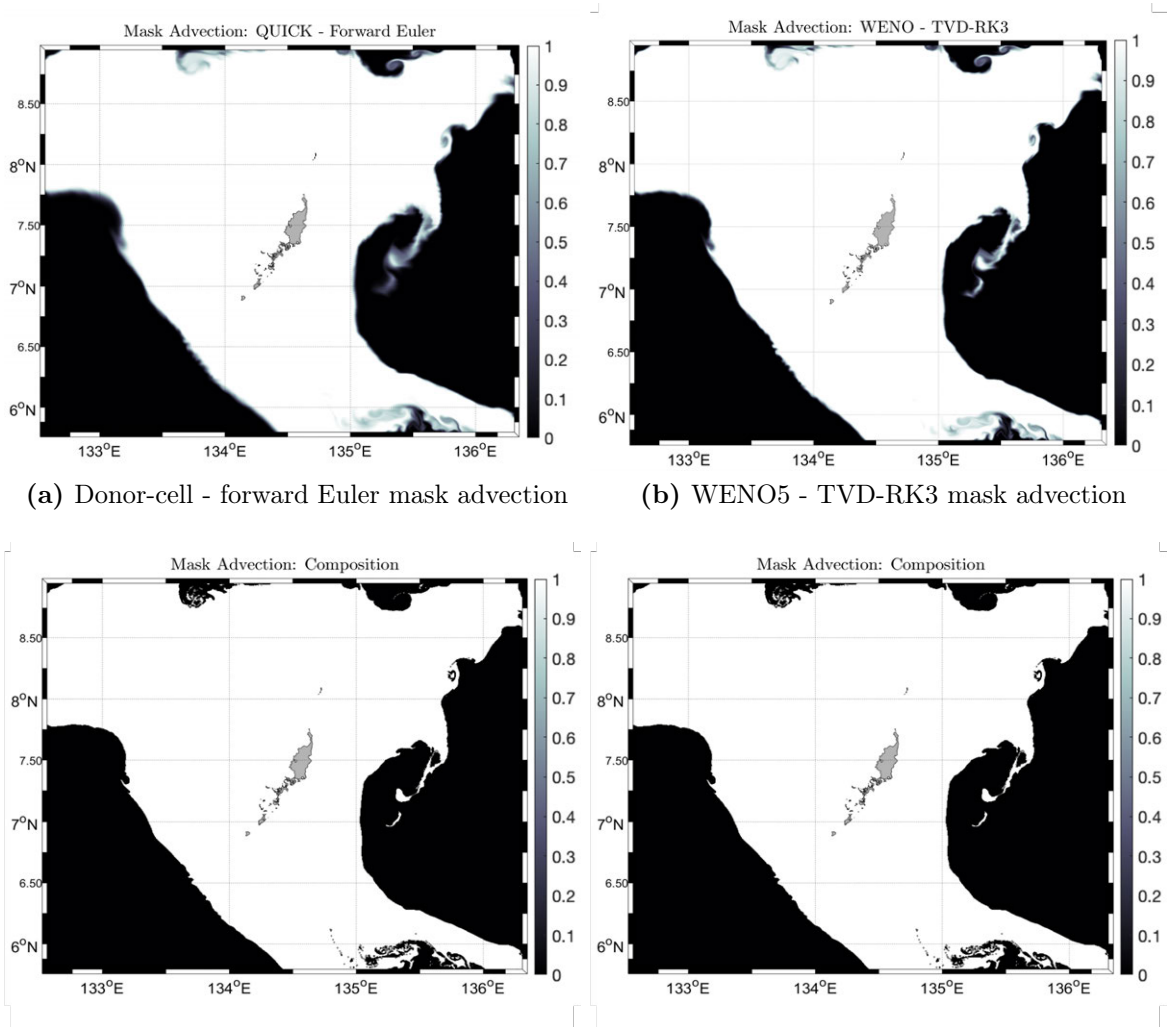
at the edges of the active domain, and the spurious FTLE ridges along the active domain boundaries are eliminated.

To compare the numerical diffusion in the mask field as well as its effects on the flow map and FTLE computation, we look at the backward flow map computation around Palau from 08 May 2015, 00Z to 14 May 2015, 00Z. Fig. (6-2) compares the mask fields obtained from the regular advection (with low and high-order schemes) and composition-based advection (with low-order schemes) with the true mask field (computed using Lagrangian trajectory advection).

As expected, one can clearly observe significant numerical diffusion of the mask field when advected using the first-order accurate donor-cell method for spatial gradients [151] and first-order accurate forward Euler time marching (Fig. (6-2a)). The numerical diffusion significantly reduces, but is still clearly present (especially arounnd the eddies formed towards the eastern side of the island) when using the 5^{th}-order accurate WENO5 method for spatial gradients [230] and 3^{rd}-order accurate TVD-RK3 time marching [189] (Fig. (6-2b)). For both these fields, we use a threshold of 0.5 for the mask field, *i.e.* the mask field value is set to 0 if less than 0.5, and it is otherwise set to 1. For the composition-based computation (Fig. (6-2c)), the obtained mask field is almost entirely diffusion free, even when using low-order accurate numerical methods (first-order donor-cell for spatial gradients and forward Euler time marching). The mask field also matches well with the true field from Fig. (6-2d) owing to the aforementioned reasons. Further, we observe that the mask field values in this case either belong to the interval $[0, 0.012]$ or to the interval $[0.983, 1]$, implying that no heuristics are required except for setting the values from the former interval to 0, and from the latter interval to 1.

Flow Map and Coherent Structures Computation in Open Domains

We now examine the 6 day long backward flow maps and the corresponding FTLE fields for the realistic data-assimilative ocean flow around Palau. Specifically, we compare 3 different approaches of computing the backward flow maps: (i) high-order accurate regular advection, (ii) method of flow map composition with low-order ac-



(a) Donor-cell - forward Euler mask advection

(b) WENO5 - TVD-RK3 mask advection

(c) Composition-based mask advection

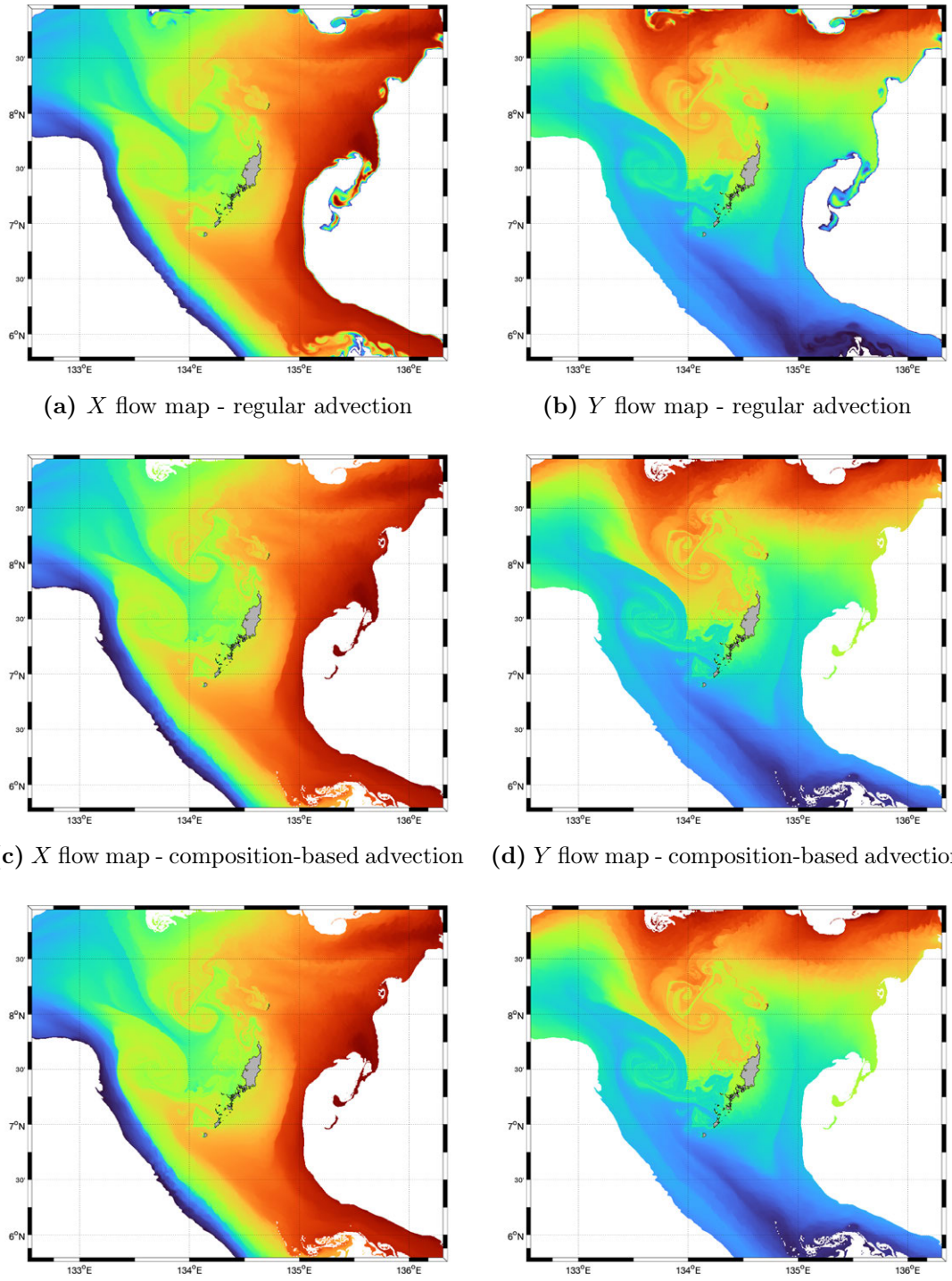
(d) True active domain

Figure 6-2: Mask fields denoting the active domains for the backward flow maps, for the realistic flow around the island of Palau. A value of 1 indicates that the corresponding location is in the active domain, whereas a value of 0 indicates that the location lies outside the active domain.

curate numerical schemes, and (iii) Lagrangian trajectory-based computation. In the first case, we use the WENO5 scheme for spatial gradients and TVD-RK3 scheme for time marching. In the second case, we use the first-order donor-cell (upwind) method for spatial gradients and forward Euler time marching with the method of flow map composition (with the optimal composition timestep). Finally, for the third case, we use Lagrangian trajectory-based computation with Runge Kutta 4 (RK4) temporal integration. Fig. (6-3) shows the 6 day backward X and Y flow maps over

the considered domain, using the above three numerical methods. As was the case for computing the mask fields, we observe that the flow maps computed using high-order regular advection are susceptible to numerical diffusion, seen around the eddies shed to the northwest of the island. Secondly, due to the heuristic employed in computing the mask field for the case of the regular advection, we see several non-physical artifacts near the boundaries of the active domain, especially at the northern, eastern and western boundaries. Such artifacts are heavily dependent on the threshold value chosen for the mask field and lead to spurious coherent structures, as will be seen. However, it can be clearly seen that the backward flow maps computed using the method of composition have minimal numerical diffusion, and they match well with the ones computed using Lagrangian trajectory integration. However, as the method of composition is rooted in a PDE solve, the error estimates and other theoretical guarantees are still maintained unlike the Lagrangian trajectory integration. To quantitatively compare the results from high-order regular advection and low-order composition-based advection, we compute the mean relative errors in the flow maps over the entire spatial domain. We find that the relative error of the regular advection computation is 12.78%, whereas that of the composition-based computation is 1.84%.

To better understand the resulting errors in the coherent structures, Fig. (6-4) shows the FTLE fields corresponding to the backward flow maps from Fig. (6-3). Hyperbolic LCSs are often extracted as ridges of the FTLE fields, and thus ideally we would want these ridges to be as sharp as possible. We can clearly see that the method of composition is able to produce such sharp ridges of the FTLE field very well when compared to the true field, indicating that it can maintain the underlying large gradients in the flow map without being numerically diffusive. However, the FTLE field computed using high-order regular advection schemes is unable to produce such sharp ridges, which indicates that numerical diffusion and other compounding errors are significant in areas where the gradients of the flow maps are large. Further, the FTLE field resulting from high-order regular advection also has spurious FTLE ridges at the eastern boundary of the active domain (Fig. (6-4a)). These ridges are created



(a) X flow map - regular advection

(b) Y flow map - regular advection

(c) X flow map - composition-based advection

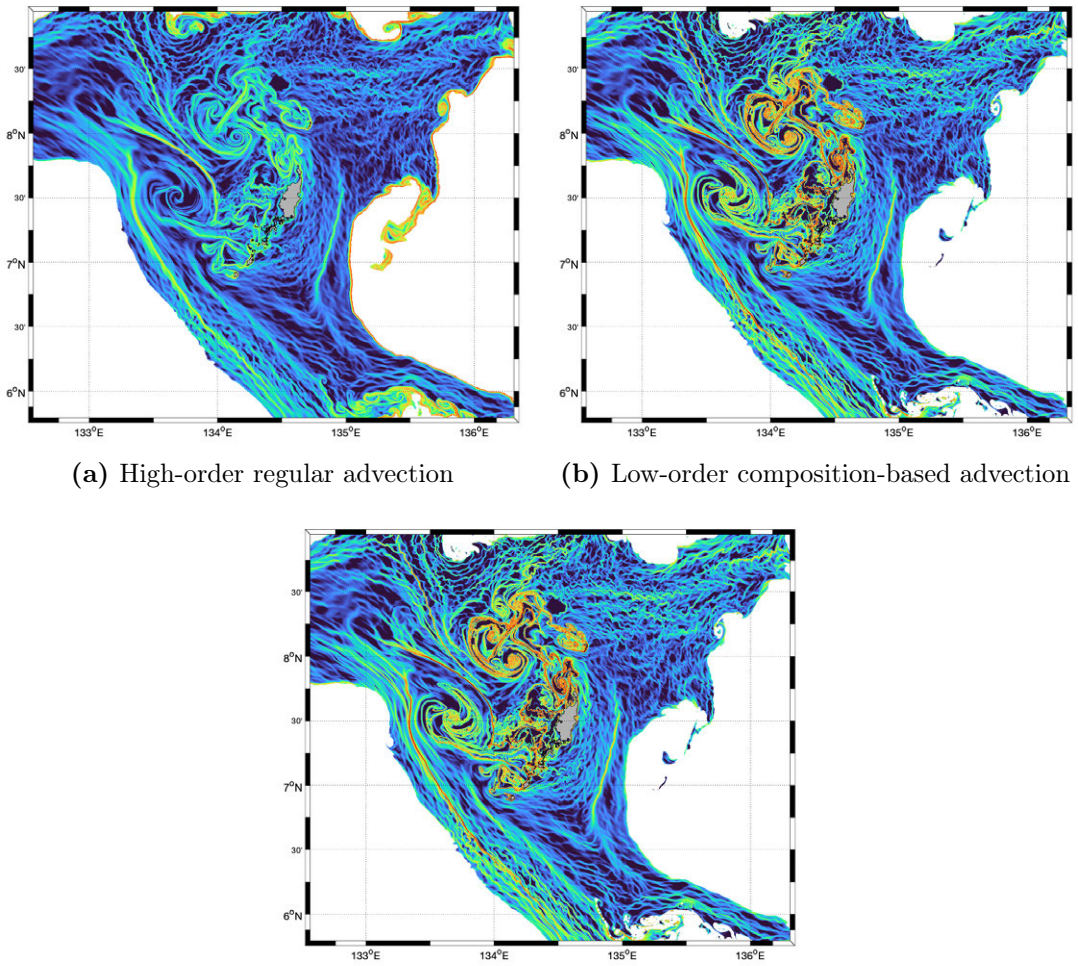
(d) Y flow map - composition-based advection

(e) True X flow map - trajectory integration

(f) True Y flow map - trajectory integration

Figure 6-3: Comparison of 6 day backward flow maps around Palau when computed using high-order regular advection (WENO5 - TVD-RK3), low-order composition-based advection (Donor-cell - forward Euler), and Lagrangian trajectory integration (RK4). The flow maps computed using high-order regular advection suffer from numerical diffusion / dispersion as well as non-physical artifacts around the active domain boundaries, whereas the low-order composition-based advection results are extremely close to the true results (computed using high-resolution Lagrangian trajectory integration).

because the heuristic chosen while determining the active domain is inaccurate. Such spurious ridges incorrectly classify the domain boundaries to be attracting / repelling manifolds and thus must be eliminated.



(a) High-order regular advection

(b) Low-order composition-based advection

(c) True field - trajectory integration

Figure 6-4: Comparison of the 6 day backward FTLE field computed using high-order regular advection, low-order composition-based advection, and Lagrangian trajectory integration. It can be seen that composition-based advection is able maintain high gradients without numerical diffusion as evident from the sharp FTLE ridges, whereas such ridges are smeared out for the regular advection computation due to the compounding numerical diffusion, even though it uses higher-order schemes.

6.1.3 Computation of Flow Maps in Varied Marine Domains

We now briefly showcase the applications of the computational toolbox based on the theory and schemes developed in Chapter 3 and Chapter 5 to study Lagrangian transport in realistic ocean flows. This computation and visualization toolbox has been extensively used in various marine regions to study material transport, coherent structures, and other associated metrics in three dimensions in hindcast mode as well as to support real-time sea exercises [38; 131; 181; 47; 153; 148; 88; 87]. Specifically, we illustrate four different applications of the computational and visualization framework and highlight its various capabilities in different contexts.

Our computational setup was used to predict, support, and analyze the real-time sea exercise that focused on studying and comparing the various Lagrangian metrics through drifter deployment, mannequin release, and dye tracking as a part of the 2018 NSF ALPHA (Advanced Lagrangian Predictions for Hazards Assessments) real-time sea exercise, south of Cape Cod, Massachusetts [88]. As an example of our real-time Lagrangian predictions, Fig. (6-5) shows surface 6 hour backward FTLEs between 08 August 2018, 10Z and 08 August 2018, 16Z in the two 2-way nested modeling domains. We also predicted the statistics of the FTLE field using ensemble ocean forecasts obtained from the MSEAS modeling system, as seen in the two right panels in Fig. (6-5). Such stochastic Lagrangian predictions were imperative in designing and executing suitable real-time sea experiments while accounting for the underlying uncertainties.

Fig. (6-6) showcases an application of our computational setup in the Alboran Sea just to the east of the Strait of Gibraltar to support the 2019 DRI CALYPSO (Coherent Lagrangian Pathways from the Surface Ocean to Interior) sea exercise focused on determining the major modes of vertical transport and areas of subduction in the Alboran Sea. Our predictions focused on identifying subduction regions in the Alboran Sea through Lagrangian measures such as flow maps and coherent structures. Fig. (6-6) shows a sample prediction of the vertical (Z) flow map at a depth of 48 m, computed and visualized in real-time, where the red shaded areas in the Z flow map

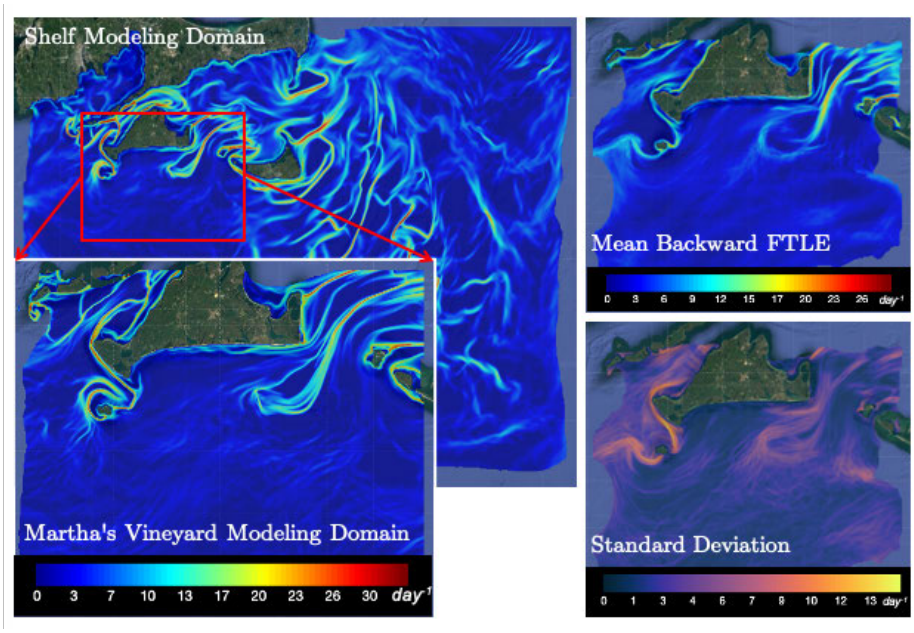


Figure 6-5: Summary of flow map and coherent structure results to support the 2018 NSF ALPHA real-time sea exercise south of Cape Cod and around the MarthaâĂŹs Vineyard and Nantucket islands. We computed the 3D flow maps, FTLE fields and their associated uncertainties along with various drifter deployment and dye release advisories.

contain water masses that rise up relative to their starting position, while the blue areas contain water masses subducting relative to their starting position, in each case over the considered 4 day period. Such maps in real-time help identify regions of interest and guide the drifter deployment as well as other observational strategies.

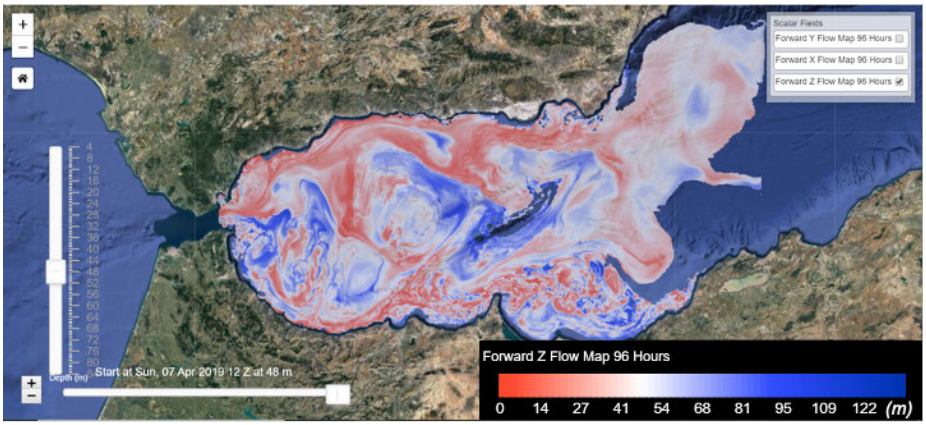


Figure 6-6: The 96 hour forward vertical (Z) flow map at 48 m depth in the Alboran Sea. The regions highlighted in red contain water parcels that rise, whereas the blue areas contain water parcels that sink, with respect to their initial depths over 96 hours

It is imperative to predict the residence times of water masses and connectivity

between ocean domains to understand the dynamics and health of marine ecosystems such as lagoons and coral reefs and to understand the impact of human activities on these systems. We focus on understanding such connectivity patterns and residence time / entrance time forecasts using 3D Lagrangian analyses in the pristine lagoons and coral reefs of the Red Sea. We use the method of flow map composition to predict the time needed for any particular water parcel to leave the domain of interest (*i.e.* a lagoon) as well as the time for any particular water parcel to enter this domain. We make these predictions around the Al Wajh lagoon in the Red Sea to understand the connectivity patterns between lagoons and coral reefs scattered all along its coasts, as seen in Fig. (6-7). More details about this work can be found in [47].

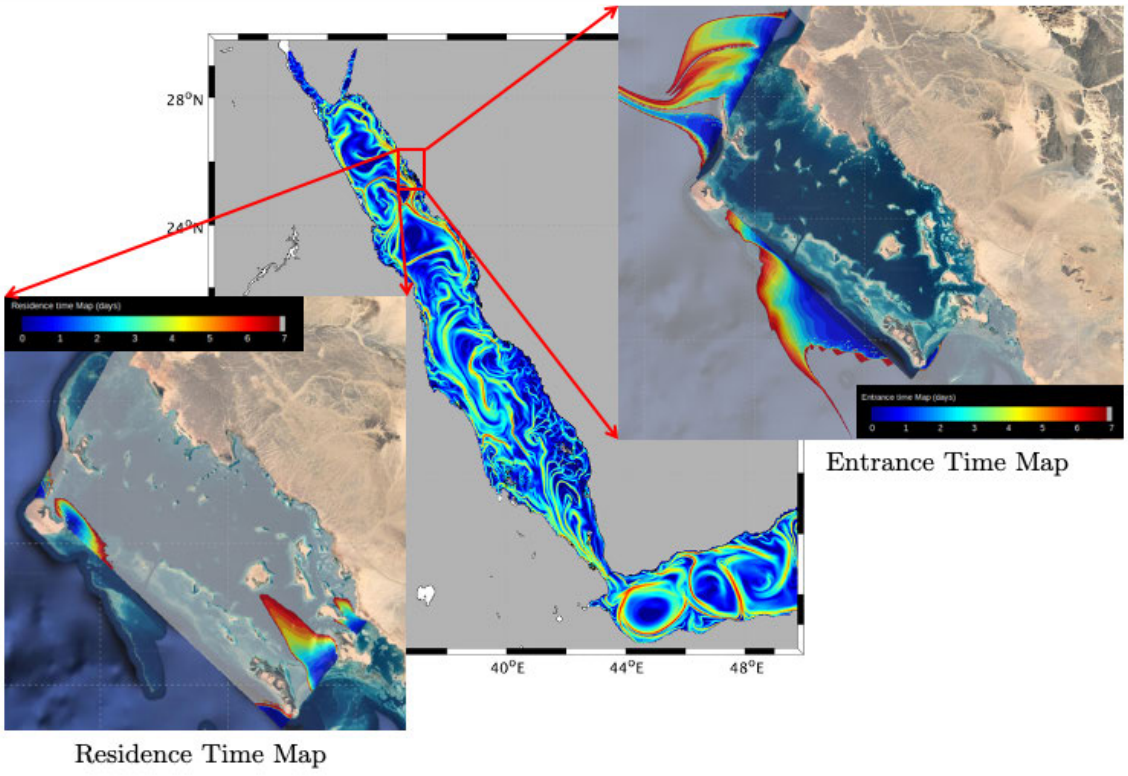


Figure 6-7: The residence time and the entrance time maps for the Al Wajh lagoon, along with the 13 day forward (repelling) FTLE field over the entire Red Sea. The residence time field indicates how long it takes for a water mass to leave the lagoon as a function of its initial position, whereas the entrance time field specifies the amount of time required for a water mass outside the lagoon to enter the lagoon.

Finally, Fig. (6-8) shows results from a recent study [153] that leverages the method of composition to understand the fate of marine plastics in Massachusetts

Bay. Specifically, red denotes surface plastics originating at the mouth of the Merrimack River, beige signifies plastics originating at the shoreline, blue are plastics originating in the rest of the Massachusetts Bay domain, and white plastics are considered to originate outside the domain. We see that after 20 days of passive advective transport due to the underlying ocean currents, most of the surface plastics that originated at the shoreline end up getting trapped in the Cape Cod Bay. Such studies can provide insight into the transport of marine pollutants and can guide in making effective policy decisions to mitigate the impact of the same.

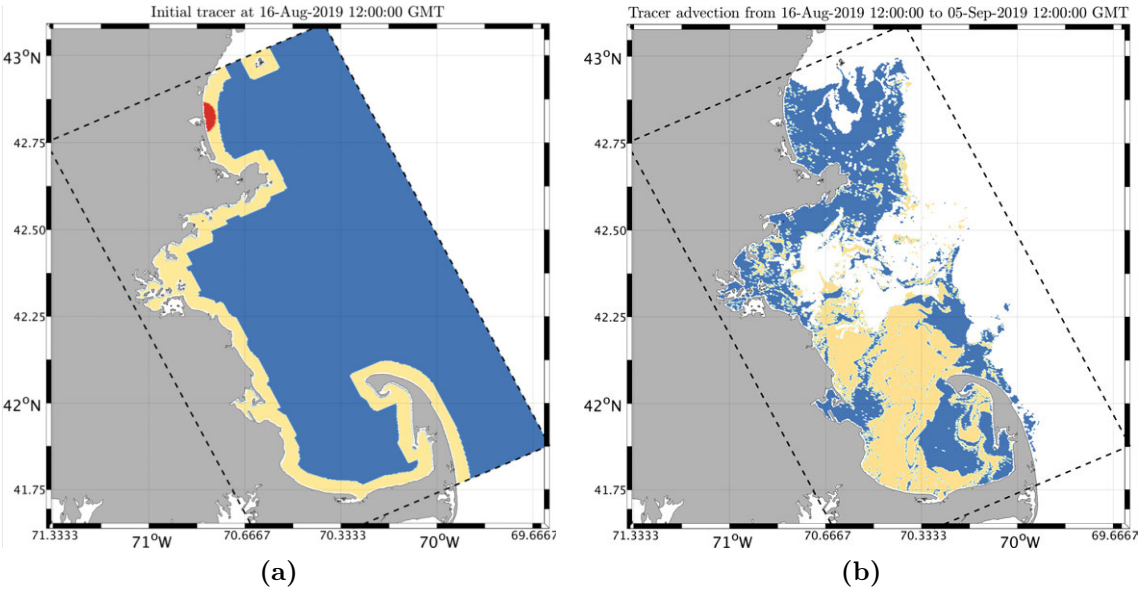


Figure 6-8: Surface plastic location initially and after 20 days of simulated passive advection. Red denotes plastics originating at the mouth of the Merrimack River, beige plastics originate at the shoreline, blue plastics originate in the rest of the Massachusetts Bay domain, and white plastics originate outside the domain.

6.2 Persistent Lagrangian Coherence / Incoherence

We now determine the coherent, persistently coherent, and incoherent material sets in two different flow fields. We first analyze our proposed coherence and persistent coherence criteria, the material sets resulting from these criteria, and their evolution in an unsteady double gyre flow. This is then followed by looking at the behavior of

an incoherent set, a coherent set, and a persistently coherent set around the island of Palau, using the realistic flow field described in Sec. 6.1.

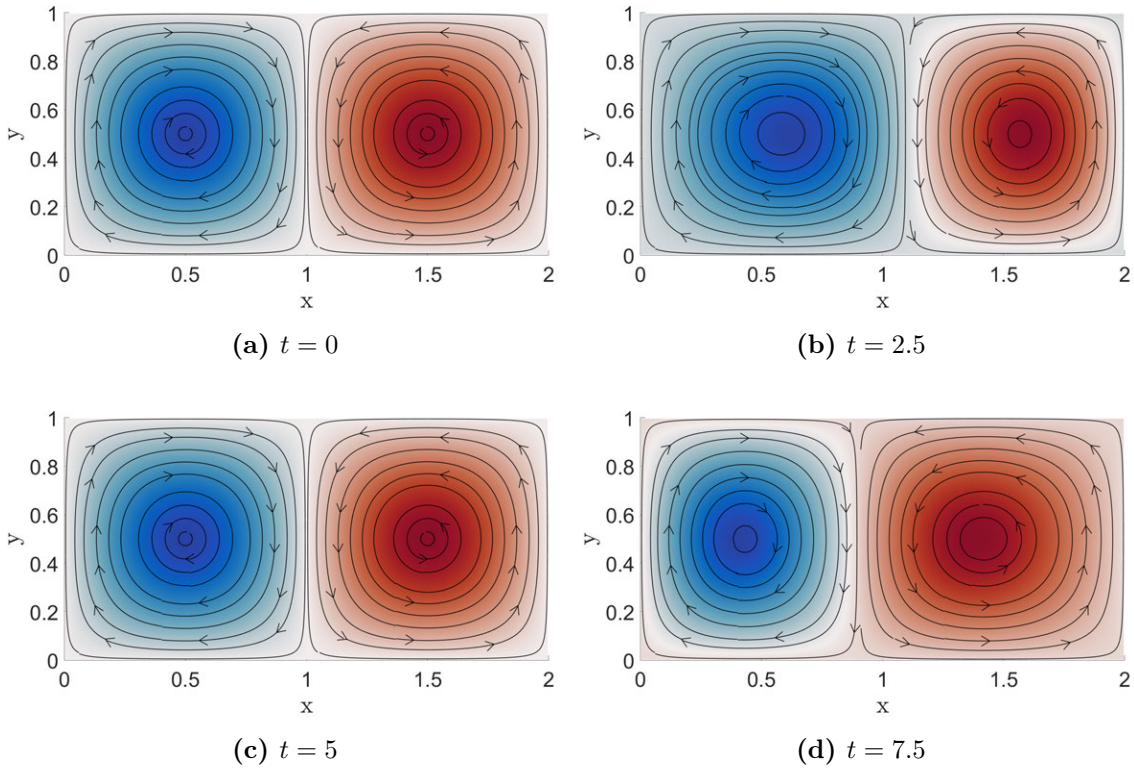
6.2.1 Analytical Double Gyre

Let us first look at the analytical unsteady double gyre . This analytical flow field is a popular 2D test case for studying Lagrangian material coherence [225; 74; 156]. It involves 2 vortices rotating in opposite directions while the vertical separatrix between the vortices oscillates horizontally in a sinusoidal fashion. It is described analytically by Eq. (6.1). This field satisfies the incompressibility criterion but it is not a solution of the Navier–Stokes' equations:

$$\begin{aligned} \boldsymbol{v}(\boldsymbol{x},t) &= \left(-\frac{\partial \boldsymbol{\psi}(\boldsymbol{x},t)}{\partial y}, \frac{\partial \boldsymbol{\psi}(\boldsymbol{x},t)}{\partial x}\right) , \\ \boldsymbol{\psi}(\boldsymbol{x},t) &= A \sin\left(\pi\left(\epsilon \sin(\omega t)\,\boldsymbol{x}^2 + (1 - 2\epsilon \sin(\omega t))\,\boldsymbol{x}\right)\right) . \end{aligned} \tag{6.1}$$

Here A is the magnitude of the velocity, ω is the oscillation frequency of the separatrix, and ϵ is (approximately) the amplitude of oscillation of the separatrix. We use values identical to Shadden et al. [225], with $A = 0.1, \omega = 0.2\pi$, and $\epsilon = 0.1$. The computational domain Ω is $[0, 2] \times [0, 1]$ and the flow field is considered on a 512×256 grid. The temporal domain of interest is $[0, 15]$. Note that, as Ω is closed at all times, the boundary conditions are irrelevant. Fig. (6-9) shows the streamlines of the velocity field overlaid on the vorticity field highlighting the two gyres. At $t = 0$, the two vortices are of equal size and the flow is symmetric about the separatrix. At $t = 2.5$, the separatrix is in its rightmost position, with the right counter-clockwise gyre assuming its narrowest shape, and the left clockwise gyre assuming its widest shape. The separatrix returns to the center with the two gyres being equal in size at $t = 5$. Finally, at $t = 7.5$, the separatrix assumes its leftmost position, with the clockwise gyre being the narrowest, and the counter-clockwise gyre being the widest.

This motion of the separatrix leads to very interesting material transport behavior in the flow field. Even though most of the material is retained in the vortex that



(a) $t = 0$ **(b)** $t = 2.5$

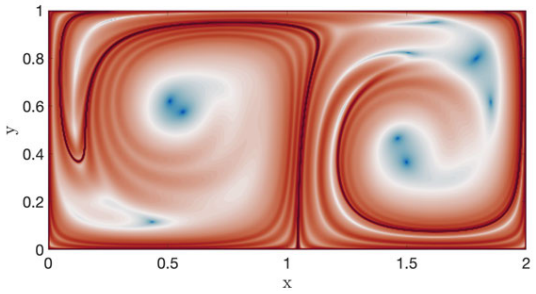
(c) $t = 5$ **(d)** $t = 7.5$

Figure 6-9: Velocity streamlines overlaid on the vorticity for the analytical double gyre test case over one time period of the flow. The time period of the flow is $2\pi/\omega = 10$.

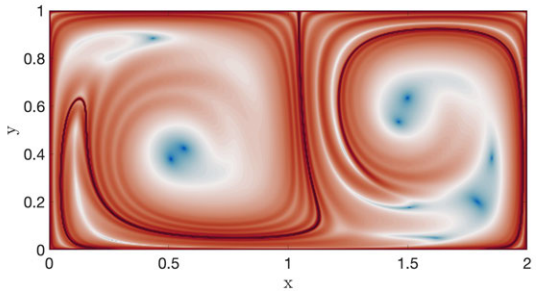
it starts in, there is some leakage, and thin filaments of this leaked material are formed that lead to intricate coherent structures [9]. This chaotic advective transport of passive tracers, manifested through thin tracer filaments and sharp boundaries between such filaments is also known as 'Lagrangian turbulence' [58; 18]. The presence of Lagrangian turbulence makes the analytical double gyre a challenging test case to determine incoherent, coherent, and persistently coherent material sets in.

Fig. (6-10) shows the polar distance field (over the time interval $[0, 15]$; on a logarithmic scale) when computed using the forward and the backward flow map to establish the equivalence between these fields. It can be clearly seen that the polar distance field from the forward flow map (Fig. (6-10a)) is equal to the polar distance field from the backward flow map, advected backward (Fig. (6-10c)). Analogously, the polar distance field from the backward flow map (Fig. (6-10b)) is equal to the polar distance field from the forward flow map, advected forward (Fig. (6-10d)). This

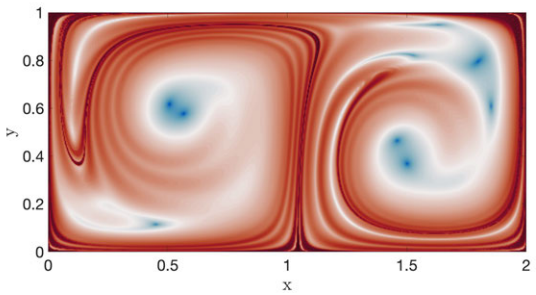
numerically validates the forward - backward duality of our proposed criteria.



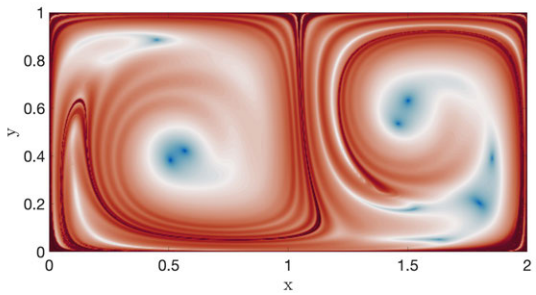
(a) Polar Distance using the forward flow map



(b) Polar Distance using the backward flow map



(c) Polar Distance using the backward flow map, advected backward



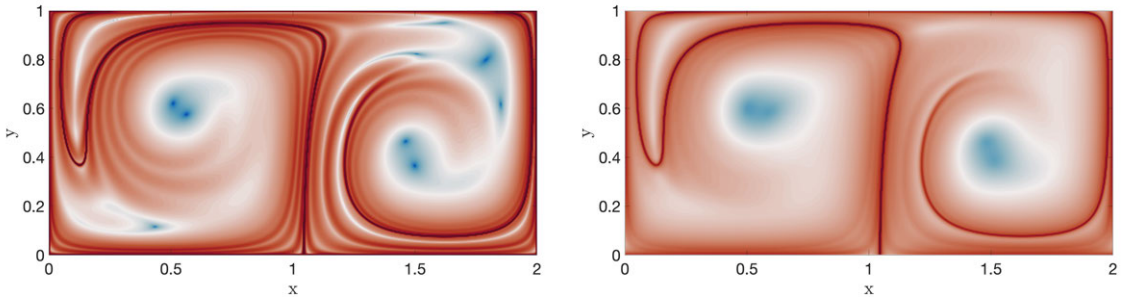
(d) Polar Distance using the forward flow map, advected forward

Figure 6-10: Equivalence of polar distance fields computed using the forward and the backward flow maps for the analytical double gyre. The polar distance field computed using the forward flow map matches the polar distance field computed using the backward flow map when advected backward, and vice-versa.

Fig. (6-11) shows the polar distance field and the extended polar distance field for direct comparison. It can be seen that even though the polar distance indicates the presence of coherent material sets in the bottom left and the top right regions of the domain, the extended polar distance has high values in these regions. This indicates that although the initial and the final shapes of the material sets starting in these regions are similar (indicating rigidity), they undergo severe distortion during the time interval of interest (*i.e.* indicating a lack of persistent rigidity). These results show that the persistently coherent sets represent 7% of the domain while incoherent regions where mixing is most likely to occur are concentrated on the edges of thin filaments of the extended polar distance.

The extended polar distance field indicates the presence of two persistently rigid

sets at the center of the two gyres. That is, these two material sets are predicted to approximately maintain their shape throughout the time interval of interest. Note that the polar distance field also has low values in these regions. This is expected, as every persistently rigid set is also a rigid set (but the converse is false).

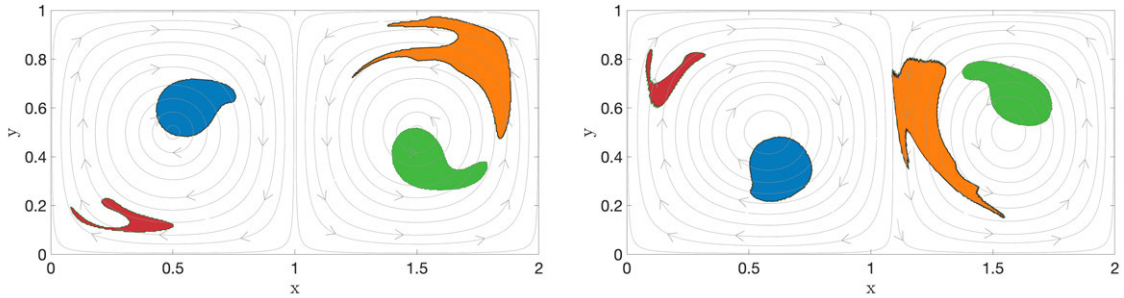


(a) Polar Distance **(b)** Extended Polar Distance

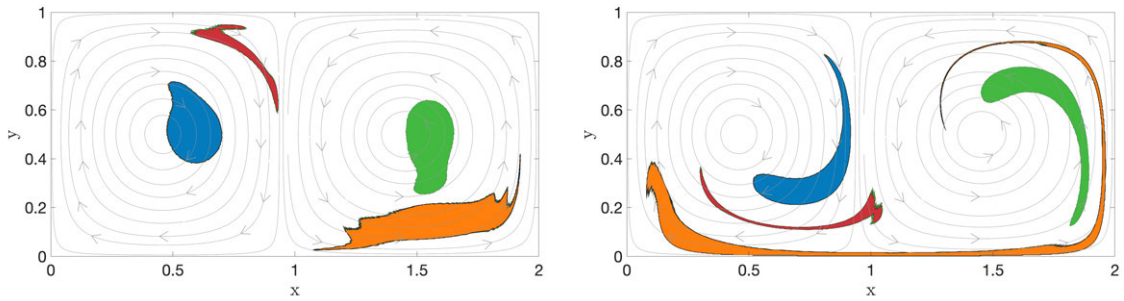
Figure 6-11: Polar distance and extended polar distance fields for the analytical double gyre over a time interval of $[0, 15]$.

Our predictions are corroborated by the observed evolution of these material sets, as seen in Fig. (6-12), wherein the rigid sets are obtained by thresholding the polar distance field. We can clearly see that these sets that are deemed rigid but not persistently rigid undergo significant stretching and compression throughout the time interval of interest, but return to a shape that is very close to their original shape at the end time. Further, the shape and the orientation of the rigid sets at the end time exactly matches the regions of low value in the polar distance field computed using the backward flow map in Fig. (6-10b), as expected. The large distortions in the shape of material sets lead to significant changes in the ratio of the set area to the set boundary. For example, it can be clearly seen that the boundary length of the orange set in Fig. (6-12) increases significantly around $t = 9$. Such an increased boundary length would lead to small-scale diffusive transport, and thus these sets are often prone to material diffusion (which is not modeled under purely advective transport).

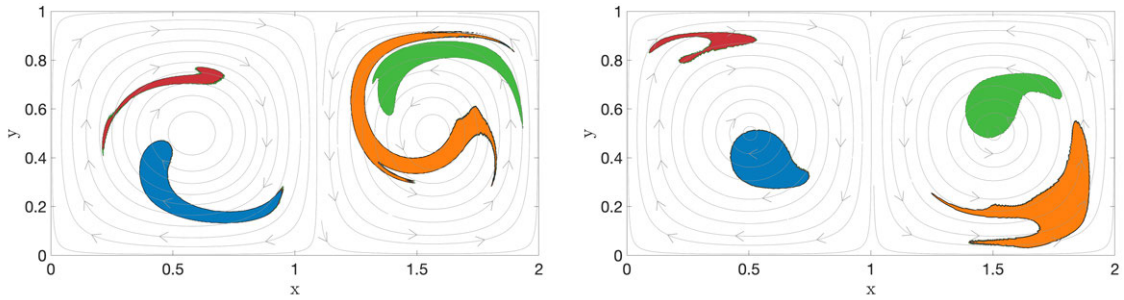
Fig. (6-13) depicts the evolution of persistently rigid sets determined by thresholding the extended polar distance from Fig. (6-11b). It can be clearly seen that these two persistently rigid sets are contained within the two central rigid sets from



(a) $t = 0$ **(b)** $t = 3$



(c) $t = 6$ **(d)** $t = 9$

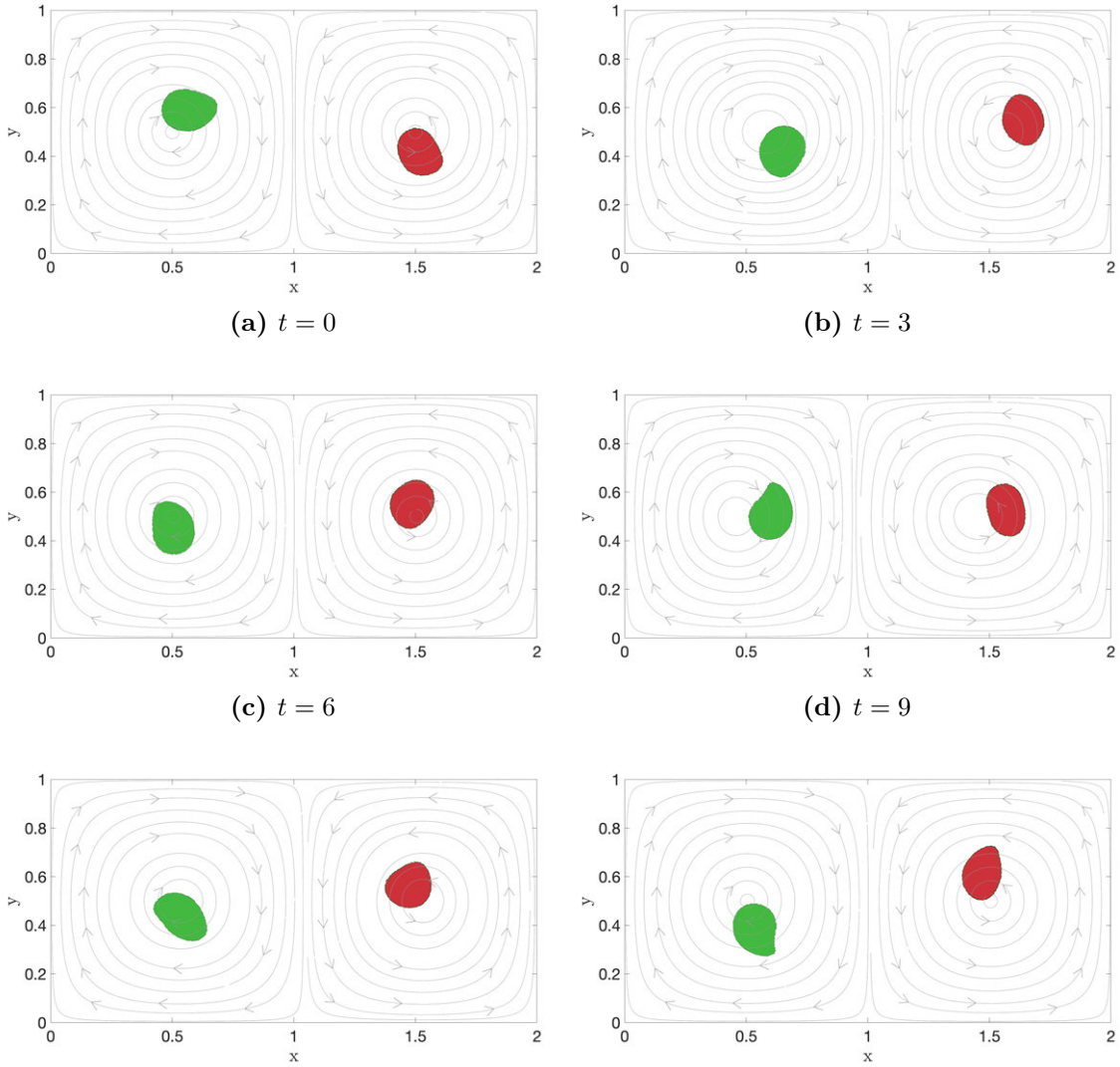


(e) $t = 12$ **(f)** $t = 15$

Figure 6-12: Evolution of the rigid sets in an analytical double gyre.These material sets are advected with the underlying flow field and are observed to undergo severe stretching at intermediate times, however their initial and final shapes are almost identical.

Fig. (6-12). However, unlike the material sets in Fig. (6-12), the two sets in Fig. (6-13) are observed to approximately maintain their shape throughout the time period of interest. Further, as these sets experience minimal change in their boundary length with time, they also tend to resist small-scale diffusion unlike non-persistently rigid sets.

Finally, we examine some of the most non-rigid sets by considering areas of high polar distance / extended polar distance values. Fig. (6-14) shows three of the most



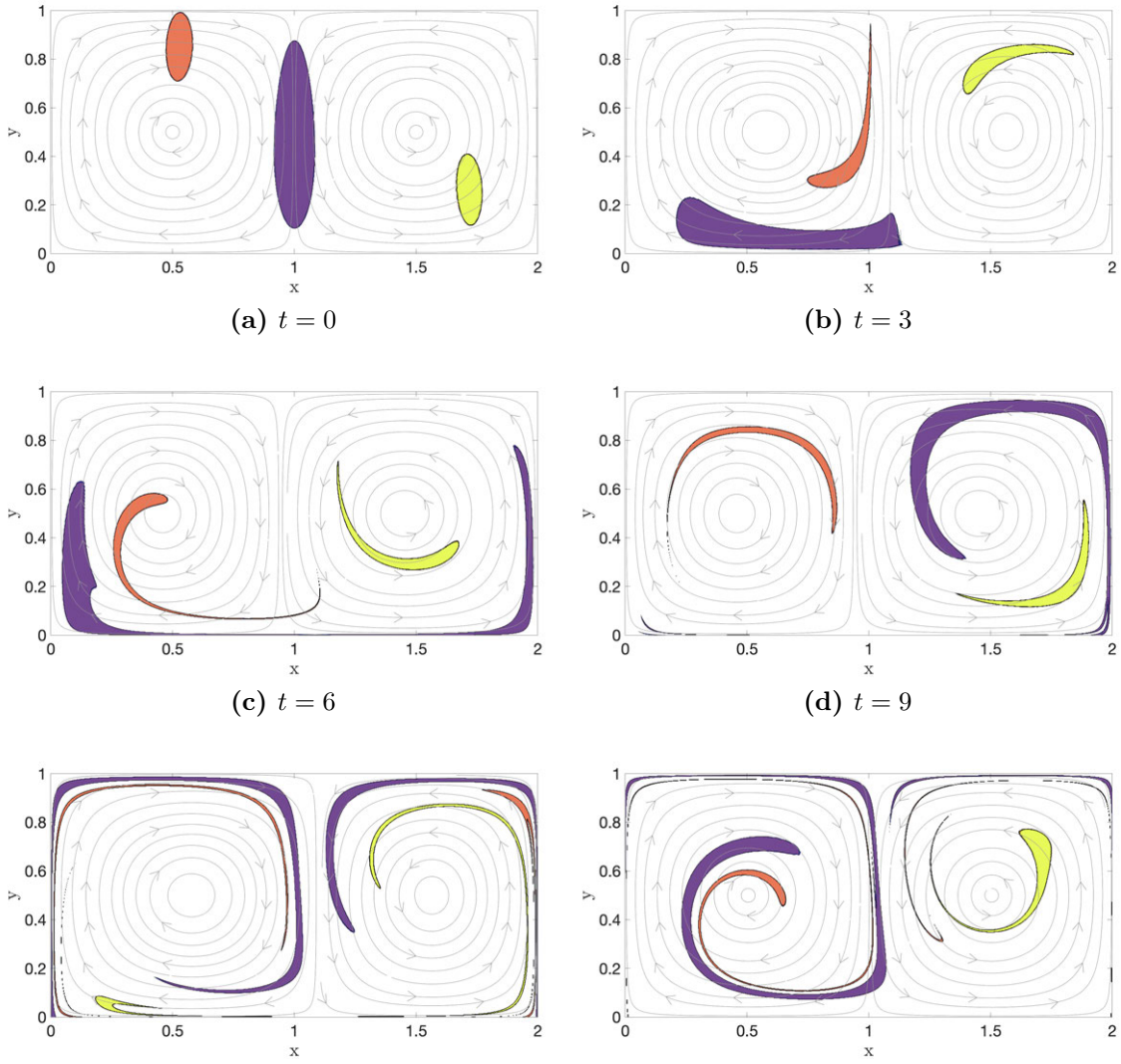
(a) $t = 0$ **(b)** $t = 3$

(c) $t = 6$ **(d)** $t = 9$

(e) $t = 12$ **(f)** $t = 15$

Figure 6-13: Evolution of the persistently rigid sets in an analytical double gyre. These material sets are advected with the fluid flow but undergo minimal distortion throughout the time interval and are able to approximately maintain their shape.

non-rigid sets in the double gyre flow. We can see that these sets undergo large amounts of chaotic advection and their intermediate and final shapes are completely different from their initial shapes. Local diffusion would also increase for such sets as their boundary length increases exponentially with time, leading to further mixing (not modeled in our case). Note that the method of flow map composition accurately captures the Lagrangian turbulence (manifested through thin and long material filaments) with minimal numerical diffusion during the evolution of these sets.



(a) $t = 0$ **(b)** $t = 3$

(c) $t = 6$ **(d)** $t = 9$

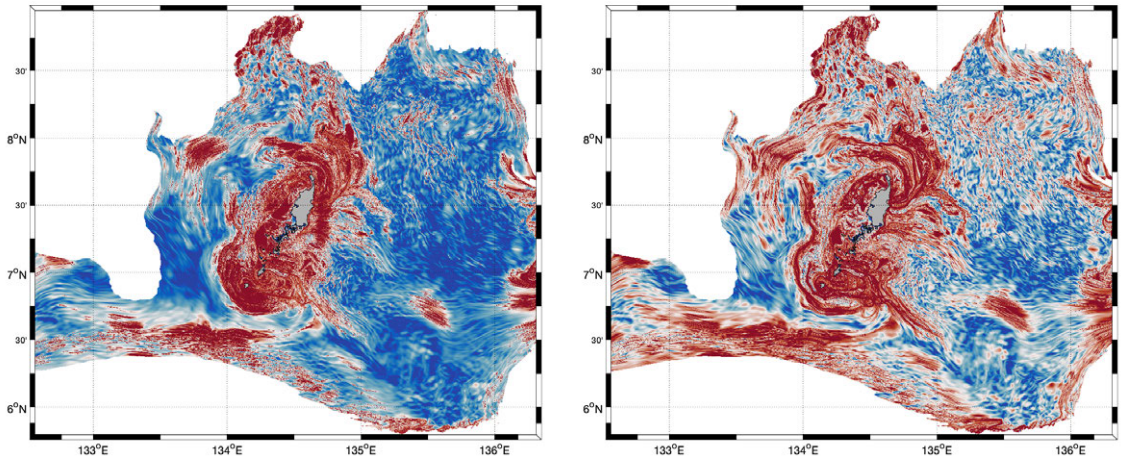
(e) $t = 12$ **(f)** $t = 15$

Figure 6-14: Evolution of most non-rigid sets in an analytical double gyre. We initialize three sets in regions with high polar distance values. We can clearly see that these sets undergo extreme amounts of stretching and their final shapes are completely dissimilar to their initial configurations.

6.2.2 Southern Pacific Ocean

In this section, we characterize different types of material sets in the realistic data-assimilative flow field around the island of Palau in the Southern Pacific Ocean (see Sec. 6.1). This flow field is highly unsteady, and is characterized by a northwestward flow impinging on the steep topography presented by the Palau archipelago, generating vorticity and eddies to the northwest of the island.

Fig. (6-15) shows the polar distance and the extended polar distance fields for the considered domain over the 6 day interval (08 May 2015 - 14 May 2015). It can be seen that there are regions of significant incoherence on all sides of the island, as evidenced from the high polar distance and extended polar distance values. This is expected as these are areas of high mixing due to the turbulence caused by the impinging northwestward flow. However, the polar distance field suggests that there exist rigid material sets to the east of the island. Further, the extended polar distance field indicates that the rigid sets originating to the east - southeast of the island are actually persistently rigid, *i.e.* they approximately maintain their shape throughout the time interval of interest.



(a) Polar Distance **(b)** Extended Polar Distance

Figure 6-15: The polar distance and the extended polar distance fields (plotted on a logarithmic scale) for the realistic data-assimilative flow around the island of Palau in the Southern Pacific Ocean. High values of the extended polar distance indicate low persistent rigidity, and low values indicate high persistent rigidity.

To illustrate the evolution of the different material sets in this complex multiscale dynamic flow field, we consider: (i) a coherent / rigid set, (ii) a persistently coherent / persistently rigid set, and (iii) an incoherent / non-rigid set. Fig. (6-16) shows the initial location of these considered material sets. We initialize these sets to be circular, and in areas of (i) low polar distance values but high extended polar distance values, (ii) low extended polar distance values, and (iii) high polar distance values,

respectively. It can be see that the initial positions of the coherent and the persistently coherent sets are close to each other. However, their behavior under advective transport is observed to be significantly different, as will be seen.

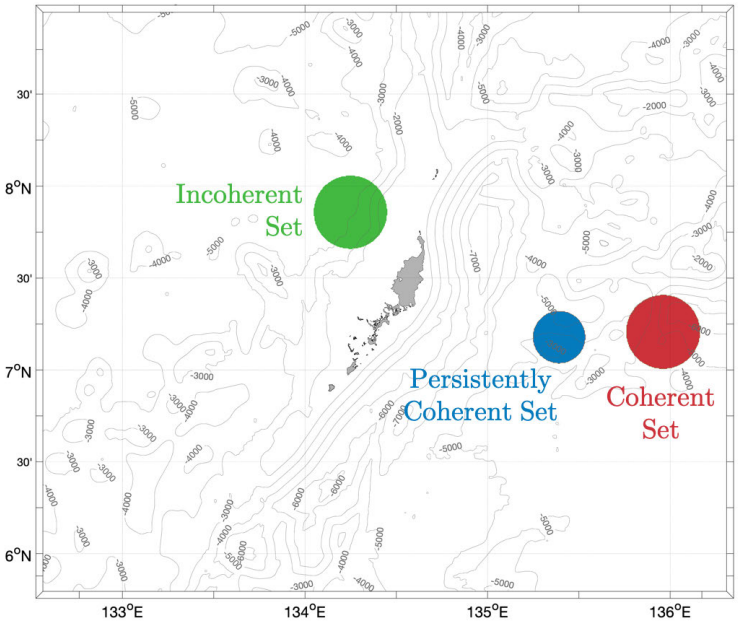
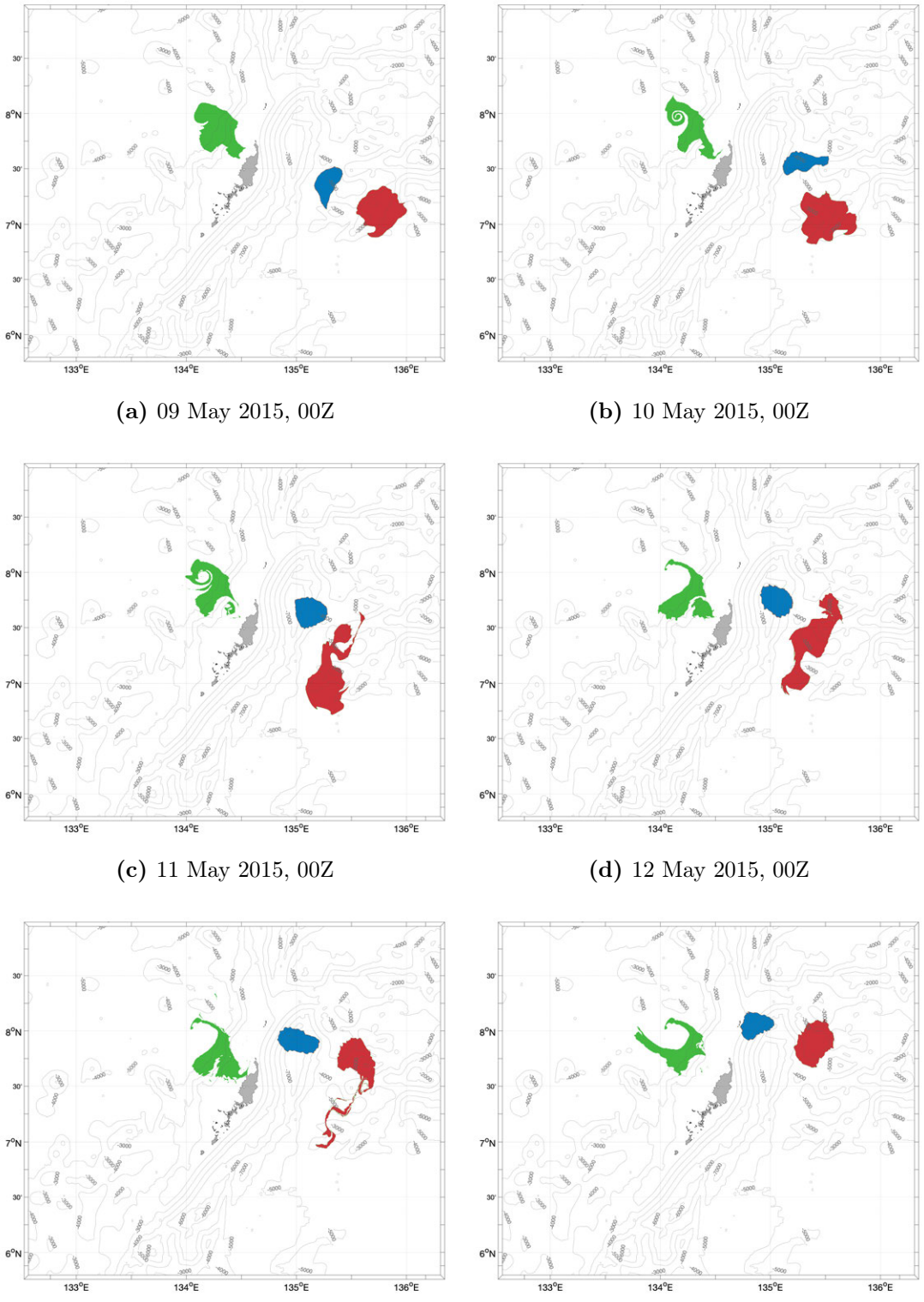


Figure 6-16: Initial location of a non-rigid set (green), a rigid set (red), and a persistently rigid set (blue) around the island of Palau on 08 May 2015 00Z, overlaid on the bathymetry contours in the region.

Fig. (6-17) shows the evolution of three considered material sets. We first look at the motion of the incoherent/non-rigid set (colored green). This set is located just behind the island, and it undergoes heavy strain throughout the considered time interval due to the eddies formed behind the island (especially seen in panels (b) and (c) of Fig. (6-17)). Evidently, its final shape is extremely different from its initial configuration. The initial and final shapes of the rigid set (colored red) are similar to each other (Fig. (6-16) and Fig. (6-17f)), however, it undergoes significant stretching and compression at intermediate times, which can make it prone to material diffusion. The persistently rigid set, colored in blue, does not experience any appreciable change in its shape over the entire time duration. It undergoes translation and rotation in an approximately rigid manner unlike the other two material sets. We also observe that the area of these material sets slightly decreases over time. This is mainly due to the nonzero divergence of the flow field, as this flow field is simply a 2D slice of an otherwise 3D flow with both vertical velocities and a dynamic free-surface.

6.3 Summary

The extensions of the method of composition to compute flow maps in open domains as well as the theory and schemes to determine coherent, incoherent, and persistently coherent material sets are illustrated through various examples and applications. We first benchmark the method of composition for computing flow maps and coherent structures in open domains with multiple inlets and / or outlets. By analyzing a complex multiscale dynamic flow field around the island of Palau in the Southern Pacific Ocean, we show that the method of composition yields flow maps similar to those from Lagrangian trajectory advection, while maintaining the theoretical guarantees and uniform spatial coverage. We then highlight the various applications of the software toolbox based on the method of composition to perform Lagrangian analyses in a variety of realistic marine flows. This is followed by the applications of the developed theory and schemes to determine rigid, persistently rigid, and non-rigid material sets in dynamic fluid flows. First, we consider the analytical double gyre flow, a popular test case to study Lagrangian coherence. We numerically verify the duality of the proposed criteria, and determine that there are four rigid and two persistently rigid material sets in this flow. Some of the most incoherent sets are also analyzed to observe the underlying chaotic advection patterns and Lagrangian turbulence. Finally, we look at rigid and persistently rigid material sets around the island of Palau. We find that the regions around the island are highly incoherent due to the turbulent dynamics. However, there exist areas of material coherence and persistent coherence to the east and southeast of the island. We study the evolution of an incoherent set, a coherent set, and a persistently coherent set to observe that even though the coherent and the persistently coherent sets are initially close to each other, the persistently coherent set maintains its shape throughout the considered 6 days. However, the coherent set undergoes significant distortion, but its final shape is similar to its initial shape, as predicted.



(a) 09 May 2015, 00Z

(b) 10 May 2015, 00Z

(c) 11 May 2015, 00Z

(d) 12 May 2015, 00Z

(e) 13 May 2015, 00Z

(f) 14 May 2015, 00Z

Figure 6-17: Evolution of a non-rigid set (green), a rigid set (red), and a persistently rigid set (blue) around the island of Palau, from 08 May 2015, 00Z until 13 May 2015, 00Z.

Chapter 7

Machine Inference and Prediction of Material Transport Features - Theory

In this chapter, we investigate the machine inference and prediction of transport features and coherent structures only from snapshot data about the advective and advective–diffusive transport of some tracer quantity (e.g. snapshots of the sea surface temperature or salinity fields), without knowing the underlying flow field. Such a machine inference framework enables one to use large-scale Eulerian observations of geophysical systems that have become ubiquitous, in part due to advances in satellite imagery and other similar techniques. To achieve this, we utilize and advance different approaches from machine learning (ML). Specifically, we use and extend recurrent neural networks, including a combination of long short-term memory networks with hypernetworks. The ML methods we develop enforce our knowledge of the physical system and the constraints it must satisfy (e.g. mass conservation) in the algorithm design and the training loss function to limit the ML search space. This allows us to train the networks only with partial supervision, without samples of the expected output fields, and still infer and predict physically consistent transport properties and coherent structures only from sparse, coarse resolution data.

Sec. 7.1 provides a deeper introduction and a brief literature review of the rapidly expanding area of scientific machine learning and other developments related to the problem at hand. We then describe the problem paradigm and setup in Sec. 7.2.

Sec. 7.3 presents the proposed novel ML algorithm in detail, including the network architecture, the loss function, and performance analysis. Sec. 7.4 builds upon the developed ML algorithm to learn the underlying advective transport properties from data about advective–diffusive tracer, while simultaneously learning its diffusivity. Finally, Sec. 7.5 extends the machine inference algorithm to i) learning the transport properties from multiple tracer observations, ii) optimally combining Eulerian and Lagrangian information, and iii) learning flow maps in open domains. These extensions are critical for the application of the developed theory and schemes to realistic ocean data. The theoretical developments from this chapter are also developed in Kulkarni and Lermusiaux [133, 134].

7.1 Introduction and Motivation

As seen in the prior chapters, Lagrangian coherent structures (LCSs) such as material barriers to flow, rigid sets etc. serve as important features to understand passive material transport, and are referred to as the 'skeleton of the flow field' [198]. Most such Lagrangian determinants of material transport are directly computed from the flow maps (see Sec. 5.1.2). Thus, it is sufficient to know the flow maps of the underlying flow field to learn the generic advective transport features. Computing flow maps, given the dynamic flow field is a well-posed problem. It is often solved by first computing flow maps either by numerically integrating several particle trajectories (Eq. (2.1)) [101; 20] or in our case by solving Eq. (3.3) and Eq. (3.4). However, for both of these approaches, the exact time-dependent velocity field is required to be known over the entire domain. This is often not the case in real ocean flows. Furthermore, the various in-situ observation devices (such as buoys, floats, gliders, autonomous vehicles, ships, satellite, etc.) yield measurements that are sparse in space-time and often noisy. Thus observations alone fall short of describing the time-dependent ocean fields over the entire domain of interest.

Instead of using the velocity field to compute the flow maps, we propose to machine infer and predict the flow maps only from Eulerian data about advective and

advective–diffusive material transport in the domains of interest. Specifically in marine settings, tracers such as salt, temperature, energy, nutrients, particulate matter, pollutants, etc., are advected with the fluid flow. Thus, the data about these tracers, often observed in an Eulerian setting at periodic intervals through satellite imagery or other macro-observation techniques, can be utilized to machine infer and predict the flow maps. These learned flow maps can then be used to characterize the landscape of material transport, either through the associated LCSs or through any other means. The machine inference and prediction of flow maps directly from Eulerian snapshot data of material transport is the inverse problem corresponding to the forward problem described in Chapter 3. It is an ill-conditioned problem [20], and thus extremely challenging in nature. Some of the reasons that make this a problem difficult to tackle include the (i) multiscale and chaotic nature of Lagrangian transport in unsteady flows, (ii) large computational domains with many degrees of freedom, and (iii) no direct access to true flow map observations.

7.1.1 Literature Review

The nascent explosion of artificial intelligence (AI) methods, from dynamic Bayesian inference to deep learning (DL), provides an unprecedented opportunity for researchers analyze data and accelerate scientific progress by extracting new knowledge [23; 178; 107]. Scientific machine learning broadly involves the idea of using known physical invariants and constraints along with certain observations to learn the quantities of interest. For example, Brunton et al. [25]; Schaeffer [218]; Rudy et al. [213]; Kulkarni et al. [137] utilize sparse regression techniques to learn symbolic forms of governing ODEs or PDEs of dynamical systems from data. Long et al. [164] learn the functional forms of PDEs by using convolutional neural networks with specifically constrained kernels. Bar-Sinai et al. [13] use ML to obtain data driven numerical discretizations of differential equations. Raissi et al. [205, 206] develop approaches to solve PDEs using Gaussian processes and deep neural networks, and further this approach to infer the velocity and pressure fields from flow visualizations [207]. There has also been a great push in using ML methods as proxies to model the subgrid-scale phenomena in geo-

physical flows and to learn closure models in turbulence simulations [209; 81; 267; 259]. Generative adversarial networks (GANs) have also been explored as a viable tool for problems such as stochastic simulations and super-resolution [264; 260]. Lu [166]; Lu and Lermusiaux [165]; Gupta and Lermusiaux [89] developed a Bayesian approach to simultaneously discover unknown model formulations such as the functional forms of the governing equations, domain geometry, state variables, and other parameters, by using and extending a nonlinear non-Gaussian filter [232; 233] to dynamical model learning. The use of machine learning in physical sciences and especially in fluid mechanics remains an active area of research, and we refer the readers to Brunton et al. [26] for an overview.

Some recent works have explored the idea of using the available observations to directly predict physical variables of interest in oceanic settings. Aksamit et al. [2] utilize ocean drifter trajectory data to train a recurrent neural network that minimizes the error in a reduced-order Maxey–Riley equation while improving the spatial and temporal resolution of the underlying coarse velocity field. An analytical approach to reconstruct the velocity field from high resolution measurements of a diffusive tracer is presented in Sharma et al. [227]. Finally, Balasuriya et al. [11] define 'generalized Lagrangian coherent structures' and look at approaches to extract these structures using the u^* approach of Pratt et al. [204].

7.2 Problem Statement

We now describe the problem statement using the notation from Table 2.1. In this work, the unsteady velocity field $\boldsymbol{v}$ is not available to us, nor do we intend to directly compute or infer it. It is simply the velocity field that governs the flow maps, which we are interested to infer and predict. We are given a snapshots of a tracer field, α, at $N_t + 1$ discrete times $t_0 = 0, t_1, t_2, \ldots, t_{N_t} = T$. Without significant loss of generality, we assume these observation times to be equally distributed in $[0, T]$, with a separation Δt. We denote the snapshot of the tracer α at time t_i as α_i. We assumed that α is a scalar, hence, $\alpha_i : \Omega \to \mathbb{R}$. Further, in this first part of our study, it is assumed that

the tracer α is passively advected with the ocean flows, and thus satisfies Eq. (2.8). In Sec. 7.4, we extend our ML framework to account for diffusive tracers, governed through the advection–diffusion equation (Eq. (3.9)) with an unknown diffusivity.

We denote the flow map admitted by the (unknown) velocity field $\boldsymbol{v}(\boldsymbol{x}, t)$ from time s until time t by $\boldsymbol{\phi}_s^t(\boldsymbol{x})$. Further, when dealing with the discrete time instants of the form $t_i = i\Delta t$, we use the abbreviated notation $\boldsymbol{\phi}_i^j(\boldsymbol{x})$ to denote $\boldsymbol{\phi}_{i\Delta t}^{j\Delta t}(\boldsymbol{x})$.

Our goal is to infer the flow maps $\boldsymbol{\phi}_0^i$ (or $\boldsymbol{\phi}_i^0$) $\forall\ i = 0, 1, \ldots, N_t$ and also predict the flow maps $\boldsymbol{\phi}_0^i$ (or $\boldsymbol{\phi}_i^0$) for $i = N_t + 1, N_t + 2, \ldots$; given only the snapshot data $\alpha_i(\boldsymbol{x})\ \forall\ i = 0, 1, \ldots, N_t$. In this process, we make the assumption that the flow field is incompressible, *i.e.* $\nabla \cdot \boldsymbol{v} = 0$. Further, for the initial part of the work, we also assume that the domain Ω is closed. Sec. 7.5.3 extends our framework to open domains where the tracer (and the fluid) can freely enter and exit the domain.

7.3 Machine Learning for Flow Map Inference and Prediction

To machine infer flow maps from snapshot tracer transport data, we utilize and build upon the plethora of ML techniques. Specifically, we encode our physical knowledge of the problem and the expected solution in the design of the ML algorithm and the corresponding loss function to be minimized.

We turn to using ML methods instead of the more conventional tools such as data assimilation for a variety of reasons: (i) the lack of governing equations for the flow map evolution that do not involve the velocity field $\boldsymbol{v}$, (ii) no direct observations of the flow map, and a complex nonlinear relationship between the observations (the tracer fields) and the variable of interest (the flow map), and (iii) little prior knowledge to generate an appropriate initial ensemble distribution.

Eq. (2.27) and Eq. (2.28) suggest that the flow maps we desire to infer and predict, say $\boldsymbol{\phi}_0^i$, can be recursively written as a composition of elementary functions, one after the other. Overall, $i-1$ of these elementary functions are (implicitly) governed by the

velocity field $\boldsymbol{v}$ from time 0 to t_{i-1}, whereas the i^{th} elementary function is governed solely by the velocity field between t_{i-1} and t_i. One can see striking similarities when comparing this setup to the functional form of neural networks (Eq. (7.1)), which approximates a complex function through a recursive composition of trainable elementary functions [23; 107]:

$$\mathcal{N}(x) = \boldsymbol{\psi}\left(W_L\left(\sigma\left(W_{L-1}\left(\ldots\sigma\left(W_1\left(x\right)\right)\right)\right)\right)\right) . \tag{7.1}$$

Here, $\mathcal{N}$ is the neural network function approximator, W_l are the (trainable) weights of different layers ($l = 1, \ldots, L$), and σ and $\boldsymbol{\psi}$ are the nonlinearities. This in turn implies that neural networks are designed to and hence are capable of approximating functions that live in the same functional space as the flow map. Further, typical ML methods are empirically known to avoid the curse of dimensionality, can be easily parallelized for efficient computation, and do not require a prior ensemble construction [23; 107]. These observations and benefits prompt us to look at ML as a viable tool, and guide us in designing an effective inference/prediction engine.

Most (supervised) machine learning methods require 'training data' through which the tunable parameters in the algorithm are learned by minimizing a pre-defined loss function. This training data is most commonly in the form of pairs of input and the corresponding output, where the training input is assumed to be from the same distribution as the expected (unseen) inputs. The amount of required input and expected output data may vary based on the application, and several techniques to reduce the requirement for training data have been proposed. However, this need for training data is a central element of most supervised ML algorithms [23; 130]. In our case, the output of the algorithm is the flow map, of which we have no training samples, *i.e.* we do not have any pairs of a suitable tracer field input and the corresponding flow map output. Thus, we cannot directly use supervised ML algorithms. However, the expected output (*i.e.* the flow map) is defined implicitly through Eq. (2.16), Eq. (2.17), and Eq. (2.20), Eq. (2.30), and Eq. (2.31). Thus we need to ensure that the outputs we obtain always satisfy these physics constraints. We achieve this by considering the

flow maps as an unknown latent state governing the transport dynamics, and learn this latent state through data about material transport and physics constraints that are embedded in the loss function.

Fig. (7-1) shows a high-level schematic of highlighting the basic function and the input/output of our ML algorithm. As stated in Sec. 7.2, we require our algorithm to output the flow maps $\boldsymbol{\phi}_0^i$ for $i = 0, \ldots, N_t$ (*i.e.* the inference phase) and for $i = N_t + 1, \ldots$ (*i.e.* the prediction phase). As the spatial grid $\boldsymbol{x}$ and the flow maps are both defined on Ω, we provide $\boldsymbol{x}$ as an input to the ML framework. Further, as we require the flow maps over various time instants ($t_i \ \forall \ i = 1, \ldots$), we require this framework to have recurrence in time, as will be seen.

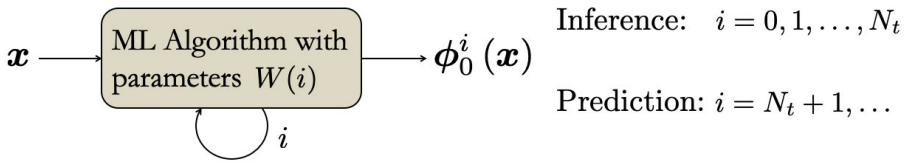


Figure 7-1: Schematic of the proposed ML framework. We input the spatial grid $\boldsymbol{x}$, and expect the framework to output $\boldsymbol{\phi}_0^i(\boldsymbol{x})$ for $i = 0, 1, \ldots, N_t, N_t + 1, \ldots$. To train the network, we utilize the available tracer transport data as well as the known physical constraints on $\boldsymbol{\phi}_0^i(\boldsymbol{x})$ in the loss function.

We utilize three key insights obtained from the physical knowledge of the flow maps. First, flow maps have local spatial structure but weak global spatial dependence, Second, flow maps encode the sequential effect of Lagrangian motion and thus can be thought to have memory of past times. For example, the flow map between times 0 and t_i can be divided into two components - a component capturing the events between 0 and t_{i-1} and the other component capturing events between t_{i-1} and t_i. This suggests that the flow map $\boldsymbol{\phi}_0^i$ retains information from $[0, t_{i-1}]$. Third, the elementary flow maps ($\boldsymbol{\phi}_i^{i+1}$) change with time due to the underlying dynamic flow field. However, they are not significantly different from each other as the underlying flow field is assumed to change smoothly and slowly (as is the case in most ocean flows).

The above insights broadly guide our design of a suitable network architecture. First, we incorporate fully convolutional layers in our network to respect the local

spatial structure in flow maps. Such convolutional layers also reduce the number of trainable parameters as compared to a corresponding fully connected network (with a stacked input) [23]. Second, to encode the sequential effect of Lagrangian motion, we utilize neural networks with memory, *i.e.* recurrent neural networks (RNNs). These allow us to efficiently infer and predict flow maps over different time intervals while using only a single trained network. However, a drawback of RNNs is strictly weight sharing in time. That is, the neural network uses the same weights and biases through time. In our application, however, each iteration (i) of the RNN implicitly corresponds to the composition of an elementary flow map $\boldsymbol{\phi}_i^{i+1}$ (that changes in time). Thus, we require the parameters of the RNN to also vary in time.

To effectively design a network while incorporating the above architecture elements, we merge two existing ML algorithms to construct the core of our flow map inference and prediction engine. Specifically, we utilize convolutional LSTMs (long short term memory networks) [229] which are RNNs with convolutional kernels, and hypernetworks [91] which allow for variable RNN parameters in time. We first briefly look at these two architectures. This is followed by the specifics of our new network architecture and loss function for flow map inference and prediction.

7.3.1 Convolutional LSTMs

The convolutional LSTMs (convLSTM) were introduced by Shi et al. [229] for precipitation nowcasting, but have found several uses in areas where known existence of local spatial structure in the a sequential dataset needs to be leveraged along with the larger spatio-temporal correlations. Such fields include video sequence prediction [254], learning of driving models from video datasets [261], anomaly detection [175], and semantic video segmentation [199].

The (fully connected) LSTM is an approach for general purpose sequence modeling, and has been an effective solution to the typical exploding and vanishing gradients of the vanilla RNNs [111; 193]. The LSTM architecture involves a cell state (c_i) at iteration i ($1 \leq i \leq N_t$) which contains the accumulated state information. This cell state is written to, cleared, and accessed through different 'gates'. These gates are

certain tensors (defined through Eq. (7.2)) that regulate the removal or addition of information to the cell state. The input gate (i_i) controls the accumulation of new information in c_i when a new input comes. The forget gate (f_i) enables the provision to forget the past cell state (c_{i-1}) as required. The output gate (o_i) controls how much of the cell state c_i will be propagated to the final state h_i, which is output at every iteration i. The convolutional LSTM extends the fully connected LSTM by using convolutional structures in the input-to-state and the state-to-state transitions. In this setup, the fully connected LSTM can be viewed as a spatial case of the convLSTM with a 1×1 spatial domain.

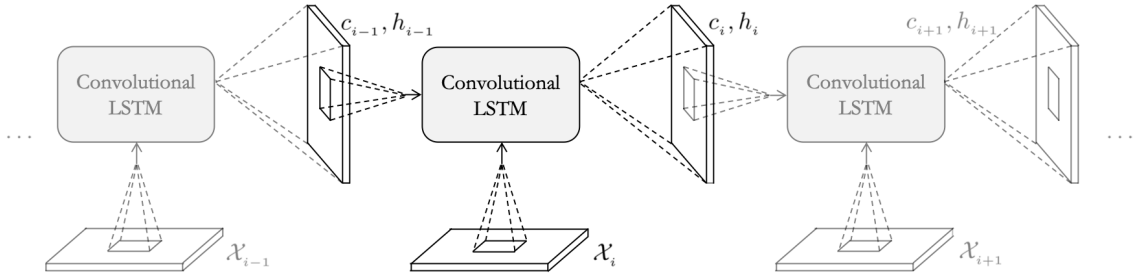


Figure 7-2: Schematic of a convolutional LSTM. Each convolutional unit takes the previous cell state (c_{i-1}) and cell output (h_{i-1}) along with the input $\mathcal{X}_i$ to produce c_i and h_i according to Eq. (7.2). This operation is performed for all $1 \leq i \leq N_t$.

Fig. (7-2) shows the schematic of the convLSTM structure, and Eq. (7.2) details its functional representation, where σ denotes the sigmoid function, *i.e.* $\sigma(t) = (1+e^{-t})^{-1}$ and $\boldsymbol{\psi}$ is the tanh function, *i.e.* $\boldsymbol{\psi}(t) = (e^{2t}-1)\,/\,(e^{2t}+1)$,

$$
\begin{aligned}
y_i &= \sigma\left(W_h^y * h_{i-1} + W_x^y * \mathcal{X}_i + W_c^y \otimes c_{i-1} + b^y\right) \quad \text{where} \quad y = \{i, f\}\,, \\
c_i &= f_i \otimes c_{i-1} + i_i \otimes \boldsymbol{\psi}\left(W_h^c * h_{i-1} + W_x^c * \mathcal{X}_i + b^c\right)\,, \\
o_i &= \sigma\left(W_h^o * h_{i-1} + W_x^o * \mathcal{X}_i + W_c^o \otimes c_i + b^o\right)\,, \\
h_i &= o_i \otimes \boldsymbol{\psi}\left(c_i\right)\,.
\end{aligned}
\tag{7.2}
$$

The input at iteration i is denoted by $\mathcal{X}_i$. We denote the convolution operation (with pre-decided kernel sizes and appropriate zero-padding where necessary) by '$*$' and the Hadamard product by '$\otimes$'. The initial state of the convLSTM is assumed to be zero,

indicating no prior knowledge. All the weights $W_{\bullet}^{\bullet}$ and the biases $b^{\bullet}$ are iteratively learned by minimizing some loss function.

7.3.2 Hypernetwork LSTMs

Typical RNNs employ strict weight sharing - the weights W and the biases b are the same for all iterations (i). Thus when unfurled, every RNN cell is identical to all other cells. This inherently restricts our ability to model a dynamic flow field (and the corresponding flow maps). We thus use the idea of hypernetworks [91]. The core idea behind hypernetworks is simple: we use a separate neural network, called the hypernetwork, to generate weights and biases for the main neural network. In our case, both the hypernetwork and the main network are convLSTMs. As the weights of the main LSTM are generated by a hypernetwork LSTM (hyperLSTM), they can vary in time. However the hypernetwork provides some implicit regularization over the variation of these weights in time and hence controls how much the individual LSTM units differ from each other, limiting abrupt changes. As the weights of the main LSTM change in time but not completely randomly, it is referred to as 'relaxed weight sharing in time' [185]. This notion of relaxed weight sharing can be seen as a trade-off between RNNs (strict weight sharing) and fully connected neural networks (no weight sharing) [91]. HyperLSTMs are end-to-end-trainable with backpropagation through time (BPTT), and achieve respectable results with fewer learnable parameters [91].

The equations defining the hyperLSTM can be divided in multiple stages. In the first step, the hyperLSTM uses the previous main LSTM state h_{i-1} and the current input $\mathcal{X}_i$ to generate a set of embeddings unique to each of the gates in the main LSTM. These embeddings are linear projections of the corresponding gates of the hyperLSTM. Finally, the main LSTM weights and biases are generated by scaling the embeddings through a weight scaling vector that is learned. The hyperLSTM equations are given by Eq. (7.3), where the hyperLSTM gate corresponding to $\bullet$ is

denoted by $\hat{\bullet}$,

$$
\begin{aligned}
\hat{\mathcal{X}}_i &= [h_{i-1}, \mathcal{X}_i]^T \ , \\
\hat{y}_i &= \mathrm{LN}\left(W_{\hat{h}}^{\hat{y}} \hat{h}_{i-1} + W_{\hat{x}}^{\hat{y}} \hat{\mathcal{X}}_i + \hat{b^y}\right) \quad \text{where} \quad y = \{i, g, o, f\} \, , \\
\hat{c}_i &= \sigma\left(\hat{f}_i\right) \otimes \hat{c}_{i-1} + \sigma\left(\hat{i}_i\right) \otimes \boldsymbol{\psi}\left(\hat{g}_i\right) \ , \\
\hat{h}_i &= \sigma\left(\hat{o}_i\right) \otimes \boldsymbol{\psi}\left(\mathrm{LN}\left(\hat{c}_i\right)\right) \ .
\end{aligned}
\tag{7.3}
$$

The operator LN($\bullet$) denotes the Layer Normalization layer [10], similar to [91]. The LN is a normalization method that normalizes the inputs across the features instead of across a mini-batch as in batch normalization [119] or across the weights of a layer as in weight normalization [215]. Once the hyperLSTM gate tensors are computed, the embeddings for the weight matrices of each of the gates in the main LSTM are generated as linear projections of the corresponding hyperLSTM gates,

$$
\begin{aligned}
z_h^y &= W_{\hat{h}h}^{\hat{y}} \hat{h}_i \quad \text{where} \quad y = \{i, g, o, f\} \, , \\
z_x^y &= W_{\hat{h}x}^{\hat{y}} \hat{h}_i \, , \\
z_b^y &= W_{\hat{h}b}^{\hat{y}} \hat{h}_i \, .
\end{aligned}
\tag{7.4}
$$

Finally, the main LSTM weights are generated through the embeddings as given by Eq. (7.5) (where $y = \{i, g, o, f\}$):

$$
y_i = \mathrm{LN}\left((W_{hz}^y z_h^y) \otimes W_h^y h_{i-1} + (W_{xz}^y z_x^y) \otimes W_x^y \mathcal{X}_i + W_{bz}^y z_b^y + b_0^y\right) \ , \tag{7.5}
$$

$$
c_i = \sigma\left(f_i\right) \otimes c_{i-1} + \sigma\left(i_i\right) \otimes \boldsymbol{\psi}\left(g_i\right) \ , \tag{7.6}
$$

$$
h_i = \sigma\left(o_i\right) \otimes \boldsymbol{\psi}\left(\mathrm{LN}\left(c_i\right)\right) \ . \tag{7.7}
$$

Similar to convLSTMs, all the weights $W_{\bullet}^{\bullet}$ and the biases $b_{\bullet}^{\bullet}$ are iteratively learned by minimizing some loss function typically through gradient based algorithms.

7.3.3 Proposed Network Architecture

As discussed before, to infer and predict the flow maps generated by an unknown dynamic flow field, we require our ML algorithm to have local spatial structure, memory, and relaxed weight sharing in time. While convLSTMs provide us with the first two components, hyperLSTMs account for the latter two. Thus an obvious approach is to combine these two ideas. We use a convLSTM to output flow maps, where the weights of this main convLSTM are generated by a hyperLSTM.

Specifically, our RNN setup takes the spatial grid $\mathcal{X}_i = \boldsymbol{x} = [x, y] \in \mathbb{R}^{2\times N_x \times N_y}$ as an input at every iteration i and outputs the flow map $\boldsymbol{\phi}_0^i(\boldsymbol{x}) = \boldsymbol{\phi}_0^{i\Delta t}(\boldsymbol{x})$, where $i = 0, \ldots, N_t$. We use the information about the tracer fields $\alpha_i(\boldsymbol{x})$ in the loss function, along with the physics constraints that the flow maps must satisfy. The grid $\boldsymbol{x}$ is passed as an input for multiple epochs and the loss is computed over all the iterations i for every epoch. The weights and biases are then trained through BPTT to minimize this loss function. Once the solution converges, the network output for iteration i is the flow map $\boldsymbol{\phi}_0^i$, for $0 \leq i \leq N_t$. Thus, these outputs are the inferred flow maps at the times that the tracer transport data is available (recall Fig. (7-1)). However, as our network is a RNN, we can simply continue to run this network for iterations $i > N_t$, which corresponds to prediction of flow maps for future times. This together addresses both the objectives posed in Sec. 7.2.

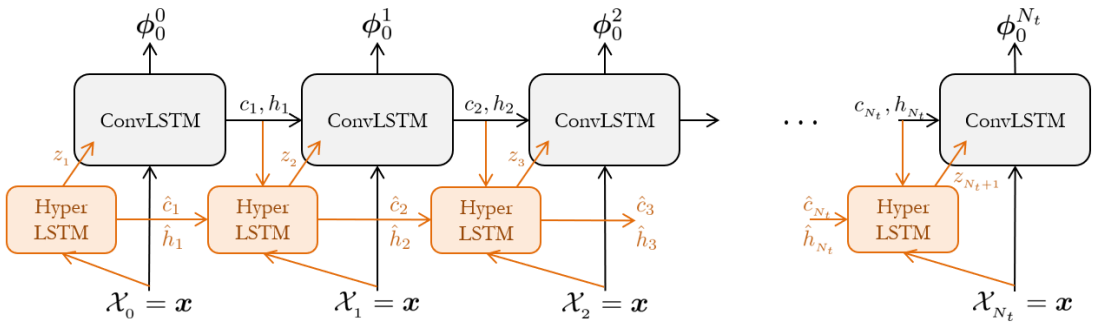


Figure 7-3: Schematic of our proposed network architecture to infer and predict flow maps from tracer transport data. There are two main components: a main convLSTM and a hyperLSTM. The hyperLSTM generates the weights and biases for the main convLSTM, which then predicts the flow map $\boldsymbol{\phi}_0^i$ in its i^{th} iteration. The information about the tracer transport fields is used in the loss function, along with the other physical constraints imposed on the flow map.

Fig. (7-3) illustrates the schematic of our algorithm. The hyperLSTM (shown in orange) receives both the input $\boldsymbol{x}$ and the main LSTM state h and generates the embeddings z for the main LSTM. These embeddings are used to generate the weights in the main LSTM, which then outputs the flow map $\boldsymbol{\phi}$. To obtain the functional formulation of the proposed recurrent cell, we introduce a tensor product, denoted by $\odot$ and defined by Eq. (7.8), along with convolution ($*$) and Hadamard product ($\otimes$):

$$\begin{aligned}
&\text{If } A \in \mathbb{R}^{a_1 \times a_2 \times \ldots \times a_p \times c_1 \times \ldots \times c_q} \text{ and } B \in \mathbb{R}^{c_1 \times \ldots \times c_q \times b_1 \times b_2 \times \ldots \times b_r}\,, \\
&\text{Then } A \odot B \in \mathbb{R}^{a_1 \times a_2 \times \ldots \times a_p \times b_1 \times b_2 \times \ldots \times b_r}\text{, such that:} \\
&(A \odot B)_{i_1,\ldots,i_p,j_q,\ldots j_q} \coloneqq A_{i_1,\ldots,i_p,k_1,\ldots,k_r} B_{k_1,\ldots,k_r,j_1,\ldots,j_q} \text{ (in Einstein notation)}\,.
\end{aligned} \tag{7.8}$$

We now look at the equations of our proposed network design. First, we generate the hyperLSTM cell gates (denoted by $\hat{\bullet}$), using the current input $\mathcal{X}_i$ and the previous main LSTM state h_{i-1} given by Eq. (7.9).

$$\begin{aligned}
\hat{\mathcal{X}}_i &= [h_{i-1}, \mathcal{X}_i]^T\,, \\
\hat{y}_i &= \mathrm{LN}\left(W_{\hat{h}}^{\hat{y}} * \hat{h}_{i-1} + W_{\hat{x}}^{\hat{y}} * \hat{\mathcal{X}}_i + \hat{b^y}\right) \quad \text{where} \quad y = \{i, g, o, f\}\,, \\
\hat{c}_i &= \sigma\left(\hat{f}_i\right) \otimes \hat{c}_{i-1} + \sigma\left(\hat{i}_i\right) \otimes \boldsymbol{\psi}\left(\hat{g}_i\right)\,, \\
\hat{h}_i &= \sigma\left(\hat{o}_i\right) \otimes \boldsymbol{\psi}\left(\mathrm{LN}\left(\hat{c}_i\right)\right)\,.
\end{aligned} \tag{7.9}$$

Once the hyperLSTM cell gates are computed, we compute the embeddings z according to Eq. (7.10) that are to be fed to the main LSTM:

$$\begin{aligned}
z_h^y &= W_{\hat{h}h}^{\hat{y}} \odot \hat{h}_i \quad \text{where} \quad y = \{i, g, o, f\}\,, \\
z_x^y &= W_{\hat{h}x}^{\hat{y}} \odot \hat{h}_i\,, \\
z_b^y &= W_{\hat{h}b}^{\hat{y}} \odot \hat{h}_i + b_0^y\,.
\end{aligned} \tag{7.10}$$

The main convLSTM (introduced in Eq. (7.2)) uses these embeddings to generate the

final output using Eq. (7.11):

$$
\begin{aligned}
y_i &= (z_h^y \otimes W_h^y) * h_{i-1} + (z_x^y \otimes W_x^y) * \mathcal{X}_i + b^y \quad \text{where} \quad y = i, g, o, f \,, \\
c_i &= \sigma(f_i) \otimes c_{i-1} + \sigma(i_i) \otimes \boldsymbol{\psi}(g_i) \,, \\
h_i &= \sigma(o_i) \otimes \boldsymbol{\psi}(c_i) \,.
\end{aligned}
\tag{7.11}
$$

Again, all the weight tensors $W_\bullet^\bullet$ and the bias tensors $b_\bullet^\bullet$ are trainable, and are learned by minimizing the considered loss function.

In the above, $\mathcal{X}_i = \boldsymbol{x} \in \mathbb{R}^{2 \times N_x \times N_y}$ and $h_i = \boldsymbol{\phi}_0^i(\boldsymbol{x}) \in \mathbb{R}^{2 \times N_x \times N_y}$. Let us denote the number of hidden units in the hyperLSTM cell by p. Thus, we have that $\hat{h}_i \in \mathbb{R}^{p \times N_x \times N_y}$. Once the hyperLSTM state is generated, we take the tensor product of this state $\hat{h}_i$ with $W_{\hat{h}h}^{\hat{y}} \in \mathbb{R}^{k_1 \times k_2 \times 2 \times p}$ according to Eq. (7.8) to obtain $z_h^y \in \mathbb{R}^{k_1 \times k_2 \times 2 \times N_x \times N_y}$, where $k_1 \times k_2$ is the kernel size used for h_{i-1} in the main convLSTM. Similarly, we take the tensor product of $\hat{h}_i$ with $W_{\hat{h}x}^{\hat{y}} \in \mathbb{R}^{k_3 \times k_4 \times 2 \times p}$ to obtain $z_x^y \in \mathbb{R}^{k_3 \times k_4 \times 2 \times N_x \times N_y}$, where $k_3 \times k_4$ is the kernel size used for $\mathcal{X}_i$ in the main convLSTM. Even though z_x^y and z_h^y are both 5D tensors, they do not need to be pre-computed, and instead their entries be computed independently for each grid point as required. z_x^y and z_h^y are then rescaled (with W_x^y and W_h^y) and used as kernels for the input-to-state and state-to-state convolutions in the main convLSTM. Fig. (7-4) illustrates the inner details for a single unit in the recurrent setup (from Fig. (7-3)) to show the data flow and approximate sizes of the various variables and parameters.

An intuitive interpretation for our network is as follows. We look at each of $W_{\hat{h}h}^{\hat{y}}$ and $W_{\hat{h}x}^{\hat{y}}$ as a collection of p different learned convolutional kernels that are to be used in the main convLSTM. The output of the hyperLSTM, $\hat{h}_i$ of size $p \times N_x \times N_y$ is a weight matrix detailing the weights of each of the kernels in $W_{\hat{h}h}^{\hat{y}}$ and $W_{\hat{h}x}^{\hat{y}}$ for each spatial grid point. That is, z_h^y and z_x^y are formed by taking the linear combination of the kernels in $W_{\hat{h}h}^{\hat{y}}$ and $W_{\hat{h}x}^{\hat{y}}$ according to the weights in $\hat{h}_i$, for every grid point (m_x, m_y) where $1 \leq m_x \leq N_x$ and $1 \leq m_y \leq N_y$. Thus, the embeddings are essentially a collection of convolutional kernels to be used for all the grid points in the main convLSTM computation: $z_h^y(\bullet, \bullet, \bullet, m_x, m_y)$ and $z_x^y(\bullet, \bullet, \bullet, m_x, m_y)$ are used in the

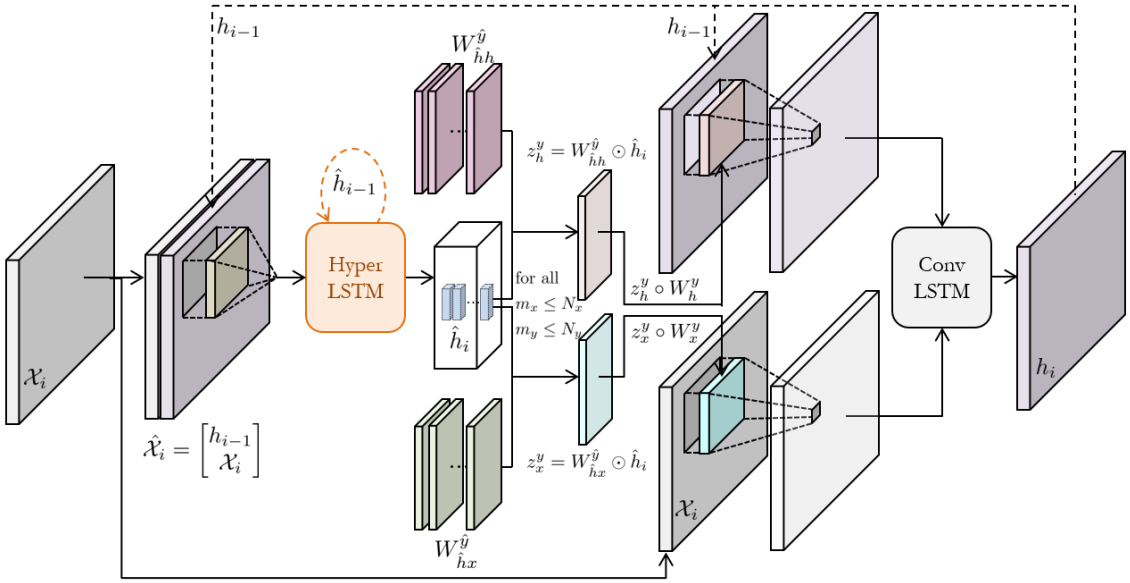


Figure 7-4: Schematic of the individual cells in the proposed architecture. One can observe how the hyperLSTM and the main convLSTM interact with each other, and how the hyperLSTM effectively enables us to choose a linear combination from a collection of learned kernels for the main convLSTM, based on the spatial location of the point of interest. Solid arrows indicate the flow of data, and dotted arrows indicate the flow of data from the past iteration of the recurrent network.

convLSTM computation for the grid point (m_x, m_y). Our network has the ability to choose appropriate kernel weights (through $\hat{h}_i$) based on the spatial location and the iteration number (time). Although the collection of kernels is itself learned, all the kernels used in the main convLSTM are always some linear combination of these collections and cannot be arbitrary. This provides an implicit regularization. The layer normalization used in Eq. (7.9) ensures the weighted combination is normalized, and the filter weights are of comparable orders of magnitudes.

7.3.4 Loss Function

The final component of our proposed setup is to define an appropriate loss function to be minimized to the train the neural network. The loss function encodes the information about the tracer transport through which the flow maps are inferred, along with the physical constraints that they must satisfy. Our proposed loss function comprises of three components: (i) tracer advection loss, (ii) incompressibility loss, and

(iii) velocity smoothness loss. In what follows, our convLSTM–hyperLSTM network provides the estimate of the flow map, i.e. $\boldsymbol{\phi}_0^i(\boldsymbol{x}) = h_i$, during its training, see above.

Tracer Advection Loss

Without diffusion and reactions, we know that the flow map and the observed tracer fields have to satisfy Eq. (2.30). This equation can be enforced while learning the flow map $\boldsymbol{\phi}_0^i$ by adding a term given by Eq. (7.12) to the loss function. This is the only component in our entire setup that uses the tracer transport data:

$$\mathcal{L}_{\text{trc}}^i = \sum_{\boldsymbol{x}\in\Omega} \left(\alpha_0\left(\boldsymbol{x}\right) - \alpha_i\left(\boldsymbol{\phi}_0^i\left(\boldsymbol{x}\right)\right)\right)^2 . \tag{7.12}$$

Incompressibility Loss

We assume that the underlying dynamic field that defines the flow maps is incompressible. This implies that flow maps have to be area preserving, see Eq. (2.20). Thus the incompressibility of the flow map $\boldsymbol{\phi}_0^i$ is enforced by including the term given by Eq. (7.13) to the loss function:

$$\mathcal{L}_{\text{ic}}^i = \sum_{\boldsymbol{x}\in\Omega} \left(\det\left(\nabla\boldsymbol{\phi}_0^i(\boldsymbol{x})\right) - 1\right)^2 . \tag{7.13}$$

Eq. (7.13) involves computing the gradients of $\boldsymbol{\phi}_0^i(\boldsymbol{x})$ with respect to $\boldsymbol{x}$. However, as $\boldsymbol{x}$ is the input to our network while $\boldsymbol{\phi}_0^i(\boldsymbol{x})$ is its output, these gradients are available using automatic differentiation [16], and no numerical approximations are required. Further, even though the loss term involves computing $\det\left(\nabla\boldsymbol{\phi}_0^i(\boldsymbol{x})\right)$, the determinant computation is inexpensive as $\nabla\boldsymbol{\phi}_0^i(\boldsymbol{x}) \in \mathbb{R}^{2\times 2}$ $, \forall\ \boldsymbol{x} \in \Omega$.

Velocity Smoothness Loss

The final term in the loss function captures our physical intuition that the underlying velocity fields change smoothly in space, which is the case for most marine and geophysical systems. Note that this does not imply smoothness at all scales, but simply that the grid is sufficiently refined such that the finest and sharpest gradients of in-

terest are resolved (if the grid is too coarse, physical fields could appear discontinuous and not smooth). This translates to the fact that $|\nabla \boldsymbol{v}|$ is finite. Unfortunately, we do not have access to (nor do we infer) the underlying velocity field, and hence we cannot directly impose any restrictions on it. However, Eq. (2.16) relates $\nabla \boldsymbol{v}$ to the gradient of the flow map. Using this, we can impose the smoothness of the underlying velocity field by adding the term given by Eq. (7.14) to the loss function:

$$\frac{d}{dt}\nabla \boldsymbol{\phi}_0^i(\boldsymbol{x}) = \nabla \boldsymbol{v}(\boldsymbol{\phi}_0^i(\boldsymbol{x}), t)\nabla \boldsymbol{\phi}_0^i(\boldsymbol{x}) \implies \left|\frac{d}{dt}\nabla \boldsymbol{\phi}_0^i(\boldsymbol{x})\right|_2 \lessapprox \left|\nabla \boldsymbol{v}(\boldsymbol{\phi}_0^i(\boldsymbol{x}), t)\right|_2 \left|\nabla \boldsymbol{\phi}_0^i(\boldsymbol{x})\right|_2 ,$$

$$\mathcal{L}_{\boldsymbol{v}}^i = \sum_{\boldsymbol{x}\in\Omega} \frac{\left|\frac{d}{dt}\nabla \boldsymbol{\phi}_0^i(\boldsymbol{x})\right|_2^2}{\left|\nabla \boldsymbol{\phi}_0^i(\boldsymbol{x})\right|_2^2} . \tag{7.14}$$

We require numerical approximations to compute $\frac{d}{dt}\nabla \boldsymbol{\phi}_0^i(\boldsymbol{x})$, and we use a backward Euler discretization, given by Eq. (7.15):

$$\frac{d}{dt}\nabla \boldsymbol{\phi}_0^i(\boldsymbol{x}) = \frac{\boldsymbol{\phi}_0^i(\boldsymbol{x}) - \boldsymbol{\phi}_0^{i-1}(\boldsymbol{x})}{\Delta t} . \tag{7.15}$$

Net Loss

We obtain the total loss as the weighted sum of the above three components, averaged over all iterations, given by Eq. (7.16) and Eq. (7.17):

$$\mathcal{L} = \frac{1}{(N_t+1)} \sum_{i=0}^{N_t} \left(w_{\text{trc}} \mathcal{L}_{\text{trc}}^i + w_{\text{ic}} \mathcal{L}_{\text{ic}}^i + w_{\boldsymbol{v}} \mathcal{L}_{\boldsymbol{v}}^i \right) , \tag{7.16}$$

$$\begin{aligned} \mathcal{L} = \frac{1}{N(N_t+1)} \sum_{i=0}^{N_t} \sum_{\boldsymbol{x}\in\Omega} \Bigg(& w_{\text{trc}} \left(\alpha_0 \left(\boldsymbol{x} \right) - \alpha_i \left(\boldsymbol{\phi}_0^i \left(\boldsymbol{x} \right) \right) \right)^2 + \\ & w_{\text{ic}} \left(\det \left(\nabla \boldsymbol{\phi}_0^i(\boldsymbol{x}) \right) - 1 \right)^2 + w_{\boldsymbol{v}} \frac{\left|\frac{d}{dt}\nabla \boldsymbol{\phi}_0^i(\boldsymbol{x})\right|_2^2}{\left|\nabla \boldsymbol{\phi}_0^i(\boldsymbol{x}) v\right|_2^2} \Bigg) . \end{aligned} \tag{7.17}$$

The weighing factors w_{trc}, w_{ic}, and $w_{\boldsymbol{v}}$ are chosen such that the loss terms are of the same order of magnitude and the loss function is robust to any perturbations. However, these factors can be edited according to any prior knowledge. For example,

one may want to decrease the weight of the incompressibility loss if the underlying flow field is known to only be weakly incompressible (such as a 2D slice of a 3D flow with large vertical velocities).

7.3.5 Information Content and Predictive Capability

Information Content of the Tracer Fields

The quality of the flow maps inferred by our algorithm depends on two principal components: (i) the amount of information contained in the available tracer fields, and (ii) the accuracy of the inference algorithm. Even though the inference procedure may be devoid of any errors, the quality of the learned fields is still constrained by the inherent amount of information in the tracer field. For example, if the initial tracer field is simply a single value (*i.e.* $\alpha_0(\boldsymbol{x}) = c \ \forall \ \boldsymbol{x} \in \Omega$) then even a perfect inference algorithm cannot learn the underlying flow maps, since there is no distinction between any domain positions based on their corresponding tracer values. On the other extreme, if the initial tracer field has N distinct values (N being the number of grid points in the discretized domain), then a perfect inference algorithm can exactly infer the underlying flow maps only through tracer advection, without using incompressibility and velocity smoothness. This can be achieved by tracking a particular tracer value through time and setting the flow map value to be its final position. However, most physical tracer fields lie somewhere between these two extremes in that they contain more than a single distinct tracer value but less than N unique tracer values. This is illustrated schematically in Fig. (7-5).

We now attempt to quantify the extent of information contained in the tracer field, given a perfect inference algorithm. Clearly, the amount of information is related to the number of unique values in the tracer, and not on the actual values themselves. To quantify the information content, we look at the expected fraction of locations without an error in the inferred flow map. Let us assume that there are r unique values in the tracer initial condition, with n_q locations bearing the initial tracer value α^q, for $q = 1, \ldots, r$. Clearly, $n_1 + n_2 + \ldots + n_r = N$. Further, we expect the

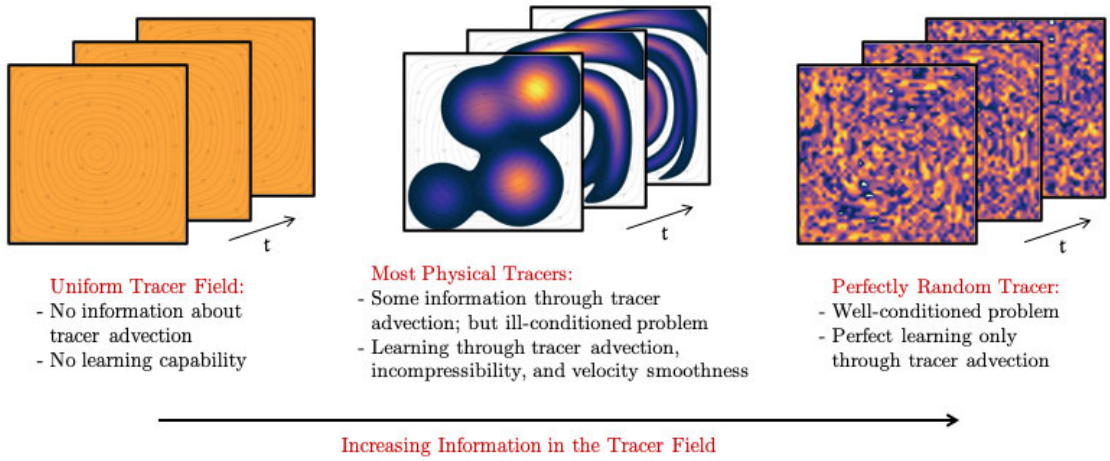


Figure 7-5: Schematic qualitatively depicting the different possible types of tracer snapshot data. There is no information and hence no learning when the tracer values are uniform and constant (left). However, in the case of a perfectly random tracer (right), one can exactly learn the flow maps only using tracer advection. Most physical tracers lie somewhere in between, for whom the flow maps are inferred using tracer advection along with incompressibility and velocity smoothness.

error in inference at a particular location to be dependent on the the number of other locations in its neighborhood with the same tracer value. Let us denote this (unknown) neighborhood by $\Omega_n \subseteq \Omega$. The area measure of Ω_n, $A(\Omega_n)$, is given by:

$$A(\Omega_n) = sA(\Omega) \quad \text{where} \quad s \in [0,1]\,,$$

where $A(\Omega)$ is the area measure of Ω. Let $\mathbb{E}(\bullet)$ and $\mathbb{P}(\bullet)$ denote the expected value and the probability of $\bullet$, respectively. We have that:

$$\mathbb{E}\begin{bmatrix}\text{fraction of locations}\\ \text{without an error}\end{bmatrix} = \sum_{q=1}^{r} \mathbb{E}\begin{bmatrix}\text{fraction of locations without error}\\ \text{that have initial tracer value } \alpha^q\end{bmatrix}\,, \qquad (7.18)$$

$$\mathbb{E}\begin{bmatrix}\text{fraction of locations without error}\\ \text{that have initial tracer value } \alpha^q\end{bmatrix} = \frac{n_q}{N}\left(1 - \mathbb{P}\begin{bmatrix}\text{error at a location with}\\ \text{initial tracer value } \alpha^q\end{bmatrix}\right)\,.$$

As there are a total of n_q locations with the initial tracer value α^q in Ω, the expected number of locations with the initial tracer value α^q in Ω_n is $s{\cdot}n_q$. Thus, the probability

correctly inferring the flow map at the considered location is $1/(s \cdot n_q)$. Thus we have:

$$\mathbb{P}\left[\begin{matrix}\text{error at a location with} \\ \text{initial tracer value } \alpha^q\end{matrix}\right] = 1 - \frac{1}{s \cdot n_q}\,. \tag{7.19}$$

Substituting Eq. (7.19) in Eq. (7.18), we get:

$$\mathbb{E}\left[\begin{matrix}\text{fraction of locations} \\ \text{without an error}\end{matrix}\right] = \sum_{q=1}^{r} \frac{n_q}{N}\left(1 - \left(1 - \frac{1}{s \cdot n_q}\right)\right) = \frac{r}{sN}\,. \tag{7.20}$$

Note that the expression given by Eq. (7.20) is valid for $r/N \in [0, s]$, and the expected fraction is equal to 1 for $r/N \in (s, 1]$. This simply signifies that the inference algorithm is not expected to make any errors if all the tracer values in every neighborhood Ω_n are distinct. Thus, we have:

$$\mathbb{E}\left[\begin{matrix}\text{fraction of locations} \\ \text{without an error}\end{matrix}\right] = \begin{cases} \frac{r}{sN} \quad \text{for } r/N \in [0, s] \quad \textit{i.e.} \quad s \in [r/N, 1]\,, \\ 1 \quad \text{for } r/N \in (s, 1] \quad \textit{i.e.} \quad s \in [0, r/N)\,. \end{cases} \tag{7.21}$$

To quantify the information content as the expected fraction of locations without an error, the specifics about Ω_n, especially the value of s must be known. However, this value (signifying the size of the neighborhood Ω_n) depends on several parameters such as the (unknown) characteristic velocity scale of the flow, the timestep etc. To eliminate this dependence on s, we define the *information content* (denoted by $\mathcal{I}$) as the mean expected fraction of the locations without an error, given by Eq. (7.22):

$$\mathcal{I}(r, N) = \mathbb{E}_s\left[\mathbb{E}\left[\begin{matrix}\text{fraction of locations} \\ \text{without an error}\end{matrix}\right]\right] = \int_{s=0}^{s=1} \mathbb{E}\left[\begin{matrix}\text{fraction of locations} \\ \text{without an error}\end{matrix}\right] ds\,. \tag{7.22}$$

Substituting Eq. (7.21) in Eq. (7.22), we get the following:

$$\mathcal{I}(r,N) = \int_{s=0}^{s=1} \mathbb{E}\left[\begin{matrix}\text{fraction of locations}\\ \text{without an error}\end{matrix}\right] ds = \left(\int_{s=0}^{s=r/N} 1 \cdot ds + \int_{s=r/N}^{s=1} \frac{r}{sN} \cdot ds\right) .$$

Thus, we define the information content in the initial tracer field as:

$$\mathcal{I}(r,N) = \frac{r}{N}\left(1 - \log\left(\frac{r}{N}\right)\right) . \tag{7.23}$$

$\mathcal{I} = 0$ signifies no learning whereas $\mathcal{I} = 1$ implies perfect inference. Note that this analysis to determine the information content does not require the exact flow maps to be known. However, it only looks at inferring flow maps through tracer advection, and does account for the effects of incompressibility and velocity smoothness that are additionally imposed. Finally, this formulation to quantify the information content in the tracer field is not unique, and other consistent approaches are possible. One may look at approaches to quantify the information content that is dependent on the spatial distribution of the tracer field and also relate such the information content of the tracer field to other information theoretic metrics [169; 40].

Predictive Capability of the Inference Algorithm

The other important aspect is the capability of our inference algorithm to predict the flow maps (and thus the tracer fields) at times beyond the availability of the tracer transport data. To asses the accuracy of these predicted flow maps, we use standard validation techniques from ML [23]. However, one needs to be cautious about the two underlying factors that eventually determine the predictive capability [211; 200] of our inference framework. These factors are (i) the predictive power of our trained algorithm and (ii) the inherent predictability limit of the underlying dynamical system. The first factor is trivial – in general, the more data our algorithm sees in the training phase, the longer it can predict in future. Of course, this is not

always true, but this is often the case [23; 182]. However, the underlying dynamical system governing the flow maps is often chaotic. This means that even when the exact state of the system is known, there is an inherent limit to how long one can faithfully predict its future states, called the predictability limit [129].

We expect that the more tracer snapshots our neural network sees during training, the more snapshots it can predict in the future, until this prediction time gets closer to the predictability limit of the underlying system. As the data fed to our neural network is normalized, neither the time difference between two consecutive snapshots nor the length of the time interval over which the snapshots are presented are expected to individually affect the prediction duration. However, we expect that the predictive capability of our algorithm reduces significantly as it attempts to predict flow maps closer to the predictability limit.

We study this phenomenon by training our neural network using tracer snapshots at times $t_0 = 0, t_1, \ldots, t_{N_t} = T$, and running it at future times to predict the flow maps between t_0 and $t_{N_t+1}, t_{N_t+2}, \ldots$ We then predict the advected tracer fields at these future times by composing the predicted flow maps with the tracer initial condition. We quantify the predictive capability of our ML algorithm by considering the relative error between the predicted and the actual (observed) tracer fields.

7.4 Machine Inference and Prediction from Advective–Diffusive Tracers

We now extend our machine learning schemes to the inference and prediction of flow maps using Eulerian snapshot data about advective–diffusive tracers. Most marine tracers such as temperature and salinity are advective–diffusive in nature. Further, the transport of certain particulates that may not be diffusive (e.g. plastics) is better modeled through an advection–diffusion (turbulent) process as compared to a purely advective process as the former accounts for the random motion on smaller scales than those resolved by the flow field.

Learning the underlying flow maps from the transport of an advective–diffusive tracer presents several challenges: (i) often, the diffusivity or turbulence closure of the tracer is not known a priori and depends on the flow field, and thus estimating the advective and diffusive contributions separately is not possible, (ii) adding diffusion eliminates the purely hyperbolic nature of the advective transport equation. This implies that the tracer field cannot be expressed as a composition of the initial condition with the corresponding flow maps.

It must be made clear that even though advective–diffusive tracer transport is not directly governed by flow maps, the underlying flow field that the tracer is transported in still admits well-defined flow maps, as they are an intrinsic property of the flow field. The advective component of the tracer transport, *i.e.* how the tracer is transported by the velocity field, is governed by these well-defined flow maps. However, the additional diffusive component of tracer transport at a particular time now depends on the relative concentration gradients of the tracer at that time, and possibly the flow field itself for several turbulence models. The velocity field only implicitly impacts the advective component of tracer transport only through flow maps when tracer diffusion and turbulent mixing do not depend directly on the velocity field. That is, in such cases, access to the actual flow field is never required to learn the flow maps governed by it. However, it is impossible to learn the flow maps without the knowing / learning the velocity field when the turbulent mixing of the tracer explicitly depends on the velocity field (or its derivatives). This is because in such situations the velocity field implicitly impacts the advective transport (through flow maps) and explicitly impacts the diffusive mixing. Hence the velocity field is explicitly required to quantify the diffusive mixing.

We extend the previously developed ML framework to learn the underlying flow maps as well as the unknown diffusivity of the tracer only from Eulerian tracer observations. Following upon the earlier discussion, we assume that the tracer diffusion does not depend on the flow field in this work. This implies that the velocity field only affects the flow maps, and explicit access to the velocity field is not required to learn the flow maps. It is assumed that the unknown diffusivity of the tracer, denoted by

$\kappa(\boldsymbol{x}, t)$, is a constant. That is, $\kappa(\boldsymbol{x}, t) = \kappa$. However, extensions of the presented work for spatio-temporally varying diffusivity as well as flow-dependent turbulent models are certainly possible, as will be seen.

In such a setting, the governing equation for α is given by Eq. (7.24), which is a special case of Eq. (3.9) with constant diffusivity and an absent source term:

$$\frac{\partial \alpha(\boldsymbol{x}, t)}{\partial t} + \boldsymbol{v} \cdot \nabla \alpha(\boldsymbol{x}, t) = \kappa \nabla^2 \alpha(\boldsymbol{x}, t)\,. \tag{7.24}$$

Our goal now is to infer and predict the flow maps $\boldsymbol{\phi}_0^i$ $(i = 0, \ldots, N_t, N_t + 1, \ldots)$ and the unknown diffusivity κ only from the tracer field observations α_i $(i = 0, \ldots, N_t)$.

As mentioned before, under advective and diffusive transport, one cannot express a direct relationship between $\alpha_0(\boldsymbol{x})$, $\alpha_i(\boldsymbol{x})$, and $\boldsymbol{\phi}_0^i(\boldsymbol{x})$. This makes the task of inferring $\boldsymbol{\phi}_0^i(\boldsymbol{x})$ only from $\alpha_i(\boldsymbol{x})$ extremely challenging. To simplify the task at hand and make the problem more tractable, we resort to operator splitting, as done in Sec. 3.3.3. That is we assume that advection and diffusion act independently during the time interval Δt. Of course, this is an assumption and not exactly true in real life. However, it is well justified when Δt is small compared to the time interval T [235]. By assuming the independence of advection and diffusion over all time intervals $[t_i, t_{i+1}]$, we draw a relationship between $\alpha_i(\boldsymbol{x})$, $\alpha_{i+1}(\boldsymbol{x})$, and $\boldsymbol{\phi}_i^{i+1}(\boldsymbol{x})$ (or $\boldsymbol{\phi}_{i+1}^i(\boldsymbol{x})$). Hence, instead of learning flow maps of the form $\boldsymbol{\phi}_0^i$ (or $\boldsymbol{\phi}_i^0$), we learn the incremental flow maps of the form $\boldsymbol{\phi}_i^{i+1}$ (or $\boldsymbol{\phi}_{i+1}^i$), for $i = 0, \ldots, N_t - 1, N_t, \ldots$ Once these incremental flow maps are learned, the corresponding flow maps over the entire duration can be obtained by flow map composition Eq. (2.27) and Eq. (2.28).

Motivated by this, we alter our inference algorithm in two main aspects. First, we modify the algorithm to now predict the incremental flow maps $\boldsymbol{\phi}_i^{i+1}$ instead of the full flow maps $\boldsymbol{\phi}_0^i$. The new schematic of our ML algorithm is given by Fig. (7-6).

We also modify our loss function to be minimized to account for the advective–diffusive transport instead of purely advective transport. However, the incremental flow maps still must satisfy the incompressibility and the velocity smoothness constraints, and hence those components of the loss function are unaffected.

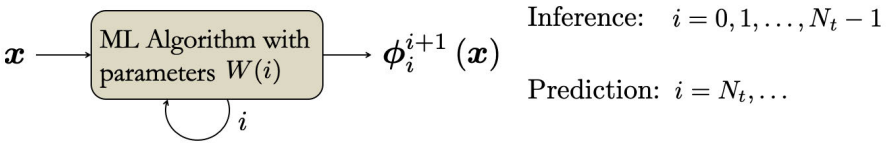


Figure 7-6: Schematic of the proposed ML framework for flow map inference and prediction from advective–diffusive tracer transport. Unlike Fig. (7-1), we now expect the framework to output the incremental flow maps $\boldsymbol{\phi}_i^{i+1}(\boldsymbol{x})$ (or $\boldsymbol{\phi}_{i+1}^{i}(\boldsymbol{x})$) for $i = 0, 1, \ldots, N_t - 1, N_t, \ldots$ To train the network, we utilize the available tracer transport data as well as the known physical constraints that $\boldsymbol{\phi}_i^{i+1}(\boldsymbol{x})$ (or $\boldsymbol{\phi}_{i+1}^{i}(\boldsymbol{x})$) must satisfy.

As stated before, we assume that advection and diffusion act independently over a time interval $[t_i, t_{i+1}]\,,\ \forall\ i$. We use a simple first-order Lie splitting (see Sec. 3.3.3), where the advection and diffusion operators are assumed to act sequentially.

We split Eq. (7.24) into its components given by Eq. (7.25). Each of the constituent PDEs is solved over $[t_i, t_{i+1}]$ and the solution of the diffusion PDE, α_*, is used as the initial condition for the advection PDE, whose final solution is α_{i+1}:

$$\begin{aligned} &\frac{\partial \alpha(\boldsymbol{x}, t)}{\partial t} = \kappa \nabla^2 \alpha(\boldsymbol{x}, t)\,, \\ &\frac{\partial \alpha(\boldsymbol{x}, t)}{\partial t} + \boldsymbol{v} \cdot \nabla \alpha(\boldsymbol{x}, t) = 0\,. \end{aligned} \tag{7.25}$$

We compute α_* by using a simple forward Euler time marching for the diffusion PDE:

$$\frac{\alpha_* - \alpha_i}{\Delta t} \approx \kappa \nabla^2 \alpha_i \quad \Longrightarrow \quad \alpha_* \approx \alpha_i + \kappa \Delta t \nabla^2 \alpha_i\,. \tag{7.26}$$

The advection component of Eq. (7.25) can be represented through flow maps as:

$$\alpha_{i+1}(\boldsymbol{x}) = \alpha_*(\boldsymbol{\phi}_{i+1}^{i}(\boldsymbol{x})) \quad \Longrightarrow \quad \alpha_* = \alpha_{i+1}\left(\boldsymbol{\phi}_i^{i+1}\right)\,. \tag{7.27}$$

Comparing Eq. (7.26) and Eq. (7.27), we get that:

$$\alpha_{i+1}\left(\boldsymbol{\phi}_i^{i+1}(\boldsymbol{x})\right) \approx \alpha_i(\boldsymbol{x}) + \kappa \Delta t \nabla^2 \alpha_i(\boldsymbol{x})\,. \tag{7.28}$$

This suggests the tracer advection–diffusion loss component at time instant t_i (with

$i = 0, \ldots, N_t - 1$) should be:

$$\mathcal{L}^i_{\text{diff}} = \sum_{\boldsymbol{x} \in \Omega} \left(\alpha_{i+1} \left(\boldsymbol{\phi}_i^{i+1}(\boldsymbol{x}) \right) - \alpha_i(\boldsymbol{x}) + \kappa \Delta t \nabla^2 \alpha_i(\boldsymbol{x}) \right)^2 . \tag{7.29}$$

As the tracer fields $\alpha_i(\boldsymbol{x})$ are given, their spatial gradients $(\nabla \alpha_i, \nabla^2 \alpha_i)$ can be easily computed with any difference approximation. Clearly, this form of $\mathcal{L}^i_{\text{diff}}$ is well-suited to learn the incremental forward flow maps $(\boldsymbol{\phi}_i^{i+1})$.

Now, instead of writing $\alpha_* = \alpha_{i+1} \left(\boldsymbol{\phi}_i^{i+1} \right)$ as in Eq. (7.27), we invert this form to obtain $\alpha_{i+1} = \alpha_* \left(\boldsymbol{\phi}_{i+1}^i \right)$. Using this relation with Eq. (7.26), we obtain Eq. (7.30):

$$\begin{aligned} \alpha_{i+1} = \alpha_* \left(\boldsymbol{\phi}_{i+1}^i \right) &\approx \alpha_i \left(\boldsymbol{\phi}_{i+1}^i \right) + \kappa \Delta t \left(\nabla^2 \alpha_i \left(\boldsymbol{\phi}_{i+1}^i \right) \right) , \\ \implies \alpha_{i+1}(\boldsymbol{x}) &\approx \alpha_i \left(\boldsymbol{\phi}_{i+1}^i(\boldsymbol{x}) \right) + \kappa \Delta t \nabla^2 \left(\alpha_i \left(\boldsymbol{\phi}_{i+1}^i(\boldsymbol{x}) \right) \right) . \end{aligned} \tag{7.30}$$

However, $\nabla^2 \left(\alpha_i \left(\boldsymbol{\phi}_{i+1}^i(\boldsymbol{x}) \right) \right)$ can be efficiently computed through the chain rule as:

$$\begin{aligned} \nabla^2 \left(\alpha_i \left(\boldsymbol{\phi}_{i+1}^i(\boldsymbol{x}) \right) \right) = &\left(\nabla^2 \alpha_i(\boldsymbol{x}) \big|_{\boldsymbol{x} = \boldsymbol{\phi}_{i+1}^i(\boldsymbol{x})} \right) \left(\nabla \boldsymbol{\phi}_{i+1}^i(\boldsymbol{x}) \right)^2 \\ &+ \left(\nabla \alpha_i(\boldsymbol{x}) \big|_{\boldsymbol{x} = \boldsymbol{\phi}_{i+1}^i(\boldsymbol{x})} \right) \left(\nabla^2 \boldsymbol{\phi}_{i+1}^i(\boldsymbol{x}) \right) . \end{aligned} \tag{7.31}$$

$\nabla \boldsymbol{\phi}_{i+1}^i$ and $\nabla^2 \boldsymbol{\phi}_{i+1}^i$ are computed using automatic differentiation [16].

Note that Eq. (7.30) can also be obtained by first applying the advection operator and then applying the diffusion operation on this advected field.Thus, the tracer advection–diffusion loss for the learning the backward flow maps $(\boldsymbol{\phi}_{i+1}^i)$ is:

$$\mathcal{L}^i_{\text{diff}} = \sum_{\boldsymbol{x} \in \Omega} \left(\alpha_{i+1}(\boldsymbol{x}) - \kappa \Delta t \nabla^2 \left(\alpha_i \left(\boldsymbol{\phi}_{i+1}^i(\boldsymbol{x}) \right) \right) \right)^2 . \tag{7.32}$$

Finally, once the incremental flow maps are obtained, the full flow maps $\boldsymbol{\phi}_0^i$ and $\boldsymbol{\phi}_i^0$ are obtained through flow map composition Eq. (2.27) and Eq. (2.28).

By keeping the diffusivity κ as a learnable parameter in our neural network, we can infer its unknown value. Further, if the (turbulent) diffusive processes are parameterized through multiple unknown parameters, we can also learn values of all these parameters in the same fashion. If the forward and the backward flow maps

are predicted simultaneously (see Sec. 7.5.3), the diffusivity can be eliminated using Eq. (7.28) and Eq. (7.30):

$$\begin{aligned}
&\alpha_{i+1}(\boldsymbol{x}) \approx \alpha_i\left(\boldsymbol{\phi}_{i+1}^i(\boldsymbol{x})\right) + \frac{\left(\alpha_{i+1}\left(\boldsymbol{\phi}_0^{i+1}(\boldsymbol{x})\right) - \alpha_i\right)}{\nabla^2\alpha_i(\boldsymbol{x})}\nabla^2\left(\alpha_i\left(\boldsymbol{\phi}_{i+1}^i(\boldsymbol{x})\right)\right) , \\
\implies &\mathcal{L}_{\text{diff}}^i = \sum_{\boldsymbol{x}\in\Omega}\left(\alpha_{i+1}(\boldsymbol{x}) - \alpha_i\left(\boldsymbol{\phi}_{i+1}^i(\boldsymbol{x})\right) + \frac{\left(\alpha_{i+1}\left(\boldsymbol{\phi}_0^{i+1}(\boldsymbol{x})\right) - \alpha_i\right)}{\nabla^2\alpha_i(\boldsymbol{x})}\nabla^2\left(\alpha_i\left(\boldsymbol{\phi}_{i+1}^i(\boldsymbol{x})\right)\right)\right)^2 .
\end{aligned} \tag{7.33}$$

This loss formulation emphasizes a stronger coupling between operator splitting by indirectly imposing that the order of operations in the Lie splitting should be irrelevant. However, this is done at a cost of requiring to infer both the forward and the backward flow maps. Once they are learned, the unknown diffusivity can be inferred by using Eq. (7.28) or Eq. (7.30).

7.5 Extensions Towards Realistic Flows

We now develop several extensions to our flow map inference and prediction engine to allow applications realistic flow fields. We first show how the inference and prediction of flow maps can be improved by using data from multiple tracers. We then show how sparse, high resolution Lagrangian information (such as float / drifter trajectories) can be combined with low resolution Eulerian tracer transport data (e.g. data obtained through satellite altimetry) to infer and predict high resolution flow maps and coherent structures. Finally, we extend our approach to handle open domains. This is of utmost importance in realistic cases, as regional ocean flow fields are rarely closed, and there is significant inflow and outflow of fluid (and the tracer that is transported with it) which needs to be addressed.

7.5.1 Flow Map Inference Using Multiple Advective Tracers

Often times in marine applications, a measuring device (e.g. satellite) may observe several different Eulerian fields (such as the sea surface temperature, salinity etc.)

simultaneously. The ability to incorporate data about the transport of additional tracers in our inference procedure is beneficial, as the transport of all such tracers is governed by the same underlying flow maps. Thus, with such additional data, we expect our algorithms to better learn the underlying flow maps.

Let us assume that along with the data about tracer α at times $t_0 = 0, t_1, \ldots, t_{N_t} = T$, we also have data about another tracer β, also at times $t_0, \ldots, t_{N_t}$ (for example, α is the temperature and β is the salinity). Similar to α_i, let us denote $\beta(\boldsymbol{x}, t_i)$ as $\beta_i(\boldsymbol{x})$. Obviously, this methodology can be extended to more than two tracers. However, for clarity, we demonstrate it only for the case of two tracers. To account for the additional tracer, we amend the tracer advection loss component in the net loss function, as given by Eq. (7.34):

$$\mathcal{L}^i_{\text{trc}} = \sum_{\boldsymbol{x}\in\Omega} \left[w_\alpha \left(\alpha_0 \left(\boldsymbol{x} \right) - \alpha_i \left(\boldsymbol{\phi}^i_0 \left(\boldsymbol{x} \right) \right) \right)^2 + w_\beta \left(\beta_0 \left(\boldsymbol{x} \right) - \beta_i \left(\boldsymbol{\phi}^i_0 \left(\boldsymbol{x} \right) \right) \right)^2 \right] . \tag{7.34}$$

Here, the weights w_α and w_β are required to normalize the scale and units of the tracers so that both these components are of the same order of magnitude. As this extension of our algorithm to account for multiple tracers only requires us to amend the loss function, it can easily be used in conjunction with any of the other extensions.

7.5.2 Merging Eulerian and Lagrangian Information

Often times, only low resolution data about the transport of the tracer is available, possibly because it is measured over a larger domain or is a product of certain indirect measurements. One can obviously use these Eulerian observations to infer and predict the flow maps and coherent structures on the same coarse grid. However, that is typically not sufficient as most coherent structure predictions require high resolution flow maps. For example, the resolution of satellite remote sensing in coastal and estuaries regions is commonly not sufficient to resolve the coastal scales of motion. Thus, to infer and predict flow maps at a fine scale, an interesting idea is to augment the coarse Eulerian data with the Lagrangian data that is high resolution but sparse. Such data is typically obtained in the form of the trajectories of marine drifters or

floats in the domain of interest over the considered time interval (e.g. the Argo float program [212], targeted drifter deployments such as the CARTHE program [192], or combinations of Lagrangian assets [29; 148] etc.).

Although such float/drifter data are available at a high resolution, *i.e.* each drifter reports its own position with high accuracy, they are also sparse in an Eulerian sense. That is, these data is available for several drifters but only along their individual trajectories. It does not cover the entire spatial domain of interest and is only available as a time series of the drifter/float positions (one time series for each drifter). Let us assume that the tracer data is available at times $t_i,\ i = 0, \dots N_t$ on a coarse grid, denoted by $\boldsymbol{x}_c$. Further, we assume that we have position data from D different drifters, each as a time series, denoted as $\boldsymbol{r}_d(t_i)$ for $d = 1, \dots, D$. We wish to predict the flow maps and the coherent structures on a grid with a fine spatial resolution, denoted by $\boldsymbol{x}_f$. The spatial accuracy of the drifter data $\boldsymbol{r}_d(t_i)$ should be comparable to the grid spacing of $\boldsymbol{x}_f$. Let the flow map between times t_i and t_j on $\boldsymbol{x}_f$ be denoted by $\boldsymbol{\phi}^j_{i,f}$ and its restriction on $\boldsymbol{x}_c$ be denoted by $\boldsymbol{\phi}^j_{i,c}$. Then our goal is to infer and predict $\boldsymbol{\phi}^i_{0,f}\,,\ \forall i = 0, \dots, N_t, N_t + 1, \dots$, given $\alpha_i(\boldsymbol{x}_c)$ and $\boldsymbol{r}_d(t_i)\,,\ \forall\ i = 0, \dots, N_t$ and $d = 1, \dots, D$.

We achieve this by modifying the network slightly from what is shown in Fig. (7-3) and by amending our loss function. Specifically, we input $\mathcal{X}_i = \boldsymbol{x}_f$ instead of $\boldsymbol{x}_c$ and expect the network to output $\boldsymbol{\phi}^i_0 = \boldsymbol{\phi}^i_{0,f}$ instead of $\boldsymbol{\phi}^i_{0,c}$. We also amend the tracer advection loss, as the tracer transport data (*i.e.* $\alpha_i(\boldsymbol{x})$) is only available on the coarse grid $\boldsymbol{x}_c$. We do this by imposing the tracer advection loss on $\boldsymbol{\phi}^i_{0,c}$, which is obtained by restricting $\boldsymbol{\phi}^i_{0,f}$ to $\boldsymbol{x}_c$. This modified tracer advection loss component is given by Eq. (7.35):

$$\mathcal{L}^i_{\text{trc}} = \sum_{\boldsymbol{x}\in\Omega} \left(\alpha_0\left(\boldsymbol{x}_c\right) - \alpha_i\left(\boldsymbol{\phi}^i_{0,c}\left(\boldsymbol{x}_c\right)\right)\right)^2 . \tag{7.35}$$

Further, we add another loss component to the net loss Eq. (7.17) to enforce that the drifter positions predicted by the flow map and the observed drifter positions should be close. Note that, if the initial position of a drifter is $\boldsymbol{r}_0$, then its position at time

t is given by $\boldsymbol{\phi}_0^t(\boldsymbol{r}_0)$. Thus, we can write this drifter position loss component as:

$$\mathcal{L}^i_{\text{drft}} = \sum_{d=1}^{D} \left(\boldsymbol{r}_d(t_i) - \boldsymbol{\phi}^i_{0,f} \left(\boldsymbol{r}_d(t_0) \right) \right)^2 . \tag{7.36}$$

The net loss in this case is given by:

$$\mathcal{L} = \frac{1}{(N_t+1)} \sum_{i=0}^{N_t} \left(w_{\text{drft}} \mathcal{L}^i_{\text{drft}} + w_{\text{trc}} \mathcal{L}^i_{\text{trc}} + w_{\text{ic}} \mathcal{L}^i_{\text{ic}} + w_{\boldsymbol{v}} \mathcal{L}^i_{\boldsymbol{v}} \right) , \tag{7.37}$$

As before, w_{drft}, w_{trc}, w_{ic}, and $w_{\boldsymbol{v}}$ are the appropriate weighing factors. The incompressibility loss and the velocity smoothness loss are applied to $\boldsymbol{\phi}^i_{0,f}$. By training our inference and prediction engine using the loss function in Eq. (7.37), we obtain high resolution flow map predictions.

7.5.3 Handling Open Domains

Most often, real ocean domains are open, *i.e.* there is a significant amount of inflow and outflow of fluid (and the tracer that it carries). As we are only given the Eulerian snapshots of the tracer field at discrete time instants, there is no direct way to determine which tracer values were present inside the domain and which tracer values came from the outside. As was discussed in Sec. 5.3.1, flow maps are only defined at the inside-inside positions, *i.e.* at the positions that were inside the domain at both the start and the end times. The positions corresponding to the trajectories that enter the domain, *i.e.* inside-outside positions, and the positions corresponding to the trajectories that leave the domain, *i.e.* outside-inside positions, need to be eliminated. Further, the forward flow map is defined at the start position of the inside-inside trajectory, whereas the backward flow map is defined at the end position of the inside-inside trajectory. This implies that the domains of definitions of the forward and the backward flow map are different. Thus, along with inferring the flow map fields, the active domains corresponding to the flow maps also need to be learned in this case. This makes the problem of inferring and predicting the flow maps in open domains extremely challenging.

Even though there is little information about what fluid parcels enter or exit the domain, we utilize an important observation to extend our ML framework to infer and predict flow maps in such domains. As the forward and the backward flow maps are inverses of each other, the domain of definition of the backward flow map is the range of the forward flow map and vice versa. Specifically, as mentioned before, the forward flow map is defined at the start location of an inside-inside trajectory, whose value is the final location of this trajectory. Conversely, the backward flow map is defined at the end location of an inside-inside trajectory, whose value is the initial location of this trajectory.

Thus, the forward flow map values (*i.e.* the final locations of the inside-inside trajectories) are the locations where the backward flow map is defined. Similarly, the backward flow map values (*i.e.* the initial locations of the inside-inside trajectories) are the locations where the forward flow map is defined.

Thus, we can extract the domain of definition of the forward flow map by looking at the unique values in the range of the backward flow map and vice-versa.

Proposed Network Architecture

The above insight implies that if we learn both the forward and the backward flow maps simultaneously, then we can use the range of the former to extract the domain of definition of the latter, and vice-versa. Motivated by this, we modify the basic requirements of our ML algorithm to simultaneously output both the forward and the backward flow maps, as in Fig. (7-7).

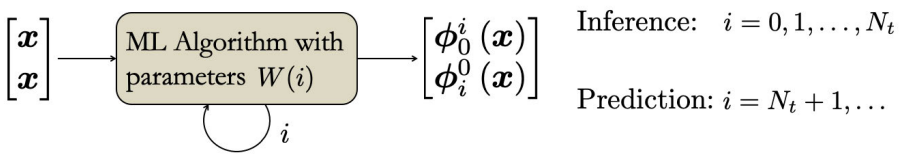


Figure 7-7: Schematic of the proposed ML framework to infer and predict flow maps in open domains. Unlike Fig. (7-1), we now expect the framework to output both $\boldsymbol{\phi}_0^i(\boldsymbol{x})$ and $\boldsymbol{\phi}_i^0(\boldsymbol{x})$ for $i = 0, 1, \ldots, N_t, N_t + 1, \ldots$. In order to train the network, we utilize the available tracer transport data as well as the known physical constraints and invertibility conditions that $\boldsymbol{\phi}_0^i(\boldsymbol{x})$ and $\boldsymbol{\phi}_i^0(\boldsymbol{x})$ must satisfy.

By combining the forward and the backward flow map predictions in the same network, we can impose additional invertibility constraints (*i.e.* the fact that the forward and the backward flow maps must be inverses of each other) in the loss function and reduce the number of trainable parameters. This greatly enhances the training and prediction of our algorithm. Secondly, we could have also simply provided $\boldsymbol{x}$ as an input to the algorithm instead of $[\boldsymbol{x}, \boldsymbol{x}]$. However, the current input specification ensures that this new network design is consistent in size with the original building block (Fig. (7-4)) and also allows for an easy inversion of the outputs as follows. Let us denote the ML function approximator from Fig. (7-7) as $\mathcal{F}_i$ for the i^{th} iteration. We then require:

$$\mathcal{F}_i\left(\begin{bmatrix}\boldsymbol{x}\\ \boldsymbol{x}\end{bmatrix}\right) = \begin{bmatrix}\boldsymbol{\phi}_0^i(\boldsymbol{x})\\ \boldsymbol{\phi}_i^0(\boldsymbol{x})\end{bmatrix}. \tag{7.38}$$

However, as the forward and the backward flow maps are mutual inverses, we get:

$$\mathcal{F}_i\left(\begin{bmatrix}\boldsymbol{\phi}_i^0(\boldsymbol{x})\\ \boldsymbol{\phi}_0^i(\boldsymbol{x})\end{bmatrix}\right) = \begin{bmatrix}\boldsymbol{\phi}_0^i(\boldsymbol{\phi}_i^0(\boldsymbol{x}))\\ \boldsymbol{\phi}_i^0(\boldsymbol{\phi}_0^i(\boldsymbol{x}))\end{bmatrix} = \begin{bmatrix}\boldsymbol{x}\\ \boldsymbol{x}\end{bmatrix} \implies \mathcal{F}_i\left(\text{flip}\left(\mathcal{F}_i\left(\begin{bmatrix}\boldsymbol{x}\\ \boldsymbol{x}\end{bmatrix}\right)\right)\right) = \begin{bmatrix}\boldsymbol{x}\\ \boldsymbol{x}\end{bmatrix}. \tag{7.39}$$

Here the 'flip' operator flips the order of the inputs, *i.e.* $\text{flip}([\boldsymbol{x}_1, \boldsymbol{x}_2]) = [\boldsymbol{x}_2, \boldsymbol{x}_1]$. The identity given by Eq. (7.39) yields a convenient approach to impose the constraint that $\boldsymbol{\phi}_0^i$ and $\boldsymbol{\phi}_i^0$ are inverses of each other without actually inverting the ML algorithm $\mathcal{F}_i$. Similar ideas have been used in supervised and generative learning algorithms [45; 210; 167].

Loss Function

As stated before, as we learn the forward and the backward flow maps simultaneously, we can impose several constraints in the loss function. First, we impose the tracer advection loss, the incompressibility loss, and the velocity smoothness loss for both the forward flow map and the backward flow map simultaneously. These loss components

are given by Eq. (7.40):

$$\begin{aligned}
\mathcal{L}^i_{\text{trc}} &= \left(\alpha_0\left(\boldsymbol{x}\right) - \alpha_i\left(\boldsymbol{\phi}^i_0\left(\boldsymbol{x}\right)\right)\right)^2 + \left(\alpha_i\left(\boldsymbol{x}\right) - \alpha_0\left(\boldsymbol{\phi}^0_i\left(\boldsymbol{x}\right)\right)\right)^2 , \\
\mathcal{L}^i_{\text{ic}} &= \left(\det\left(\nabla\boldsymbol{\phi}^i_0(\boldsymbol{x})\right) - 1\right)^2 + \left(\det\left(\nabla\boldsymbol{\phi}^0_i(\boldsymbol{x})\right) - 1\right)^2 , \\
\mathcal{L}^i_{\boldsymbol{v}} &= \frac{\left|\frac{d}{dt}\nabla\boldsymbol{\phi}^i_0(\boldsymbol{x})\right|^2_2}{|\nabla\boldsymbol{\phi}^i_0(\boldsymbol{x})v|^2_2} + \frac{\left|\frac{d}{dt}\nabla\boldsymbol{\phi}^0_i(\boldsymbol{x})\right|^2_2}{|\nabla\boldsymbol{\phi}^0_i(\boldsymbol{x})v|^2_2} .
\end{aligned} \tag{7.40}$$

We further impose an additional constraint which enforces the fact that $\boldsymbol{\phi}^i_0$ and $\boldsymbol{\phi}^0_i$ are inverses of each other for all i. This is enforced by adding a loss component given by Eq. (7.41) to the total loss function:

$$\mathcal{L}^i_{\text{inv}} = \left(\boldsymbol{\phi}^0_i\left(\boldsymbol{\phi}^i_0\left(\boldsymbol{x}\right)\right) - \boldsymbol{x}\right)^2 + \left(\boldsymbol{\phi}^i_0\left(\boldsymbol{\phi}^0_i\left(\boldsymbol{x}\right)\right) - \boldsymbol{x}\right)^2 . \tag{7.41}$$

By using Eq. (7.39), $\mathcal{L}^i_{\text{inv}}$ can be efficiently computed as:

$$\mathcal{L}^i_{\text{inv}} = \left| \mathcal{F}_i\left(\text{flip}\left(\mathcal{F}_i\left(\begin{bmatrix}\boldsymbol{x}\\ \boldsymbol{x}\end{bmatrix}\right)\right)\right) - \begin{bmatrix}\boldsymbol{x}\\ \boldsymbol{x}\end{bmatrix}\right|^2_2 . \tag{7.42}$$

Thus after training our modified ML algorithm by minimizing the above-described loss function, we obtain the forward and the backward flow maps simultaneously. Note that these obtained fields are still defined over the entire domain. That is, these fields have (incorrect) values at the initial / final locations of the trajectories that leave / enter the domain. Such locations need to be filtered out from these fields to obtain the correct forward / backward flow maps defined over their corresponding active domains.

Extraction of the Active Domain

Let us denote the output of the inference and prediction framework at time instant t_i as $[\Psi^i_0(\boldsymbol{x}), \Psi^i_0(\boldsymbol{x})]$ and the active domains corresponding to the forward and the backward flow maps as $\Omega^f_a(t_i)$ and $\Omega^b_a(t_i)$ respectively. The restriction of $\Psi^i_0(\boldsymbol{x})$ to $\Omega^f_a(t_i)$ is the desired forward flow map $\boldsymbol{\phi}^i_0(\boldsymbol{x})$ and the restriction of $\Psi^0_i(\boldsymbol{x})$ to $\Omega^b_a(t_i)$ is

the desired backward flow map $\boldsymbol{\phi}_i^0(\boldsymbol{x})$.

The active domains Ω_a^f and Ω_a^b are simply defined by the initial and the final locations of the inside-inside trajectories respectively. Thus, in order to extract these domains, we iteratively eliminate locations that either left or entered the domain Ω, as given in Algorithm 1.

We first obtain the range of the backward flow map, *i.e.* the set of unique values of the backward flow map field. We then only retain the forward flow map values at the locations contained in the range of the backward flow map and eliminate all the locations that are not in the range. As the range of the backward flow map is essentially all the domain positions where trajectories start at the initial time, this operation corresponds to eliminating the outside-inside positions. We then consider the updated forward flow map (defined only at the aforementioned locations), and compute its range. The range of the forward flow map denotes all the end locations of the trajectories that start inside Ω. We then only retain the backward flow map at the locations in the range of the forward flow map. This corresponds to eliminating all the inside-outside positions. Thus by alternatively completing these two steps, we eliminate the outside-inside and the inside outside positions. The outside-outside positions are never considered and hence do not need to be eliminated. Thus, through these two steps, we eliminate the outside-inside and the inside-outside position, thereby only retaining the inside-inside positions as desired. Ideally, just one such iteration of filtering the forward and the backward flow map should yield the true active domains. However, we observe that in practice we require about 2-5 iterations to obtain convergence of results due to numerical effects.

Algorithm 1 Active domain extraction for the forward and the backward flow maps

Require: $\Psi_0^i(\boldsymbol{x})$ and $\Psi_i^0(\boldsymbol{x})$
  Set $\boldsymbol{\phi}_0^i(\boldsymbol{x}) = \Psi_0^i(\boldsymbol{x})$ and $\boldsymbol{\phi}_i^0(\boldsymbol{x}) = \Psi_i^0(\boldsymbol{x})$
  while not converged **do**
    Compute the unique values in $\boldsymbol{\phi}_0^i(\boldsymbol{x})$ and $\boldsymbol{\phi}_i^0(\boldsymbol{x})$:
    Set $\Omega_a^f(t_i) = \text{Unique}\left(\boldsymbol{\phi}_i^0(\boldsymbol{x})\right)$ and $\Omega_a^b(t_i) = \text{Unique}\left(\boldsymbol{\phi}_0^i(\boldsymbol{x})\right)$
    Update $\boldsymbol{\phi}_0^i(\boldsymbol{x}) = \boldsymbol{\phi}_0^i(\boldsymbol{x}) \;\; \forall \boldsymbol{x} \in \Omega_a^f(t_i)$ and NaN otherwise
    Update $\boldsymbol{\phi}_i^0(\boldsymbol{x}) = \boldsymbol{\phi}_i^0(\boldsymbol{x}) \;\; \forall \boldsymbol{x} \in \Omega_a^b(t_i)$ and NaN otherwise

7.6 Summary

In this chapter, we develop the theoretical framework to machine infer and predict flow maps and coherent structures only from Eulerian data about the transport of advective and advective–diffusive tracers. We achieve this by designing a novel recurrent neural network using our insights about the flow map. Specifically, we use and extend recurrent neural networks, including a combination of long short-term memory networks (LSTMs) with hypernetworks. The LSTM captures the sequentiality of Lagrangian trajectory motion. The hypernetwork enables relaxed weight sharing in time, *i.e.* it allows for the main LSTM weights to change every iteration, thereby capturing the dynamic nature of the incremental flow maps. The loss function of the said neural network accounts for the accuracy of the observed tracer evolution (that is implicitly governed by the underlying flow maps), along with incompressibility and velocity smoothness to obtain physically consistent fields, at the scales resolved by the data. We further build upon these developments to learn from transport data about multiple tracers, account for both Eulerian and Lagrangian information, and build a more general setup to infer the forward and the backward flow maps simultaneously, which then allows us to learn coherent structures in dynamic open domains. Applications of the theory developed in this chapter are showcased on various analytical and realistic flow fields in Chapter 8.

Chapter 8

Machine Inference and Prediction of Material Transport Features - Applications

In this chapter, we first look at the applications of the ML framework developed in Chapter 7 to two canonical flow fields often used as idealized manifestations of typical marine flows - a steady swirl flow (in Sec. 8.1) and an unsteady double gyre flow (in Sec. 8.2). We look at the application of the proposed framework to learn flow maps and coherent structures, our approach of combining Eulerian and Lagrangian information, and also some interesting improvements in the results when certain physical insights about the flow field are accounted for. We also consider the advective–diffusive transport of a tracer in the analytical double gyre, where we learn the underlying flow maps and the unknown diffusivity of this tracer. This is followed by the application of the proposed framework to a realistic marine flow field in the Western Mediterranean (Alboran) Sea in Sec. 8.3. We show how the flow maps and coherent structures can be learned from (i) simulated advective tracer transport data, (ii) sea surface temperature, and (iii) combined data about the sea surface temperature and salinity in this open domain. We find that our algorithms are able to infer the flow maps and coherent structures accurately, even in the presence of significant fluid inflow and outflow. Further, we find that in the Alboran Sea, our algorithms can

accurately predict future flow maps for more than 1 day beyond the availability of the observed data. However, the errors increase beyond that and the predictions become unreliable. The results from this chapter are published in Kulkarni and Lermusiaux [133, 136].

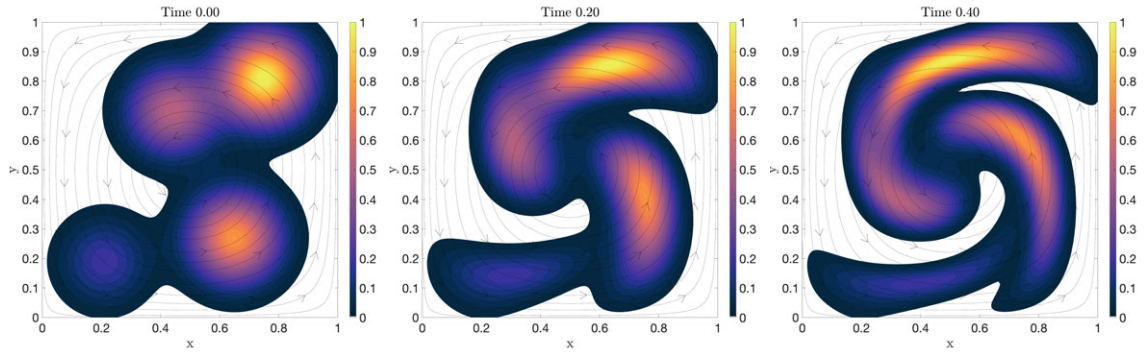
8.1 Analytical Swirl Flow

We first look at an analytical swirl flow in a closed square domain $\Omega = [0,1] \times [0,1]$ [49; 161], where the non-dimensional velocity is given by Eq. (8.1) (same as Eq. (4.1); restated for convenience):

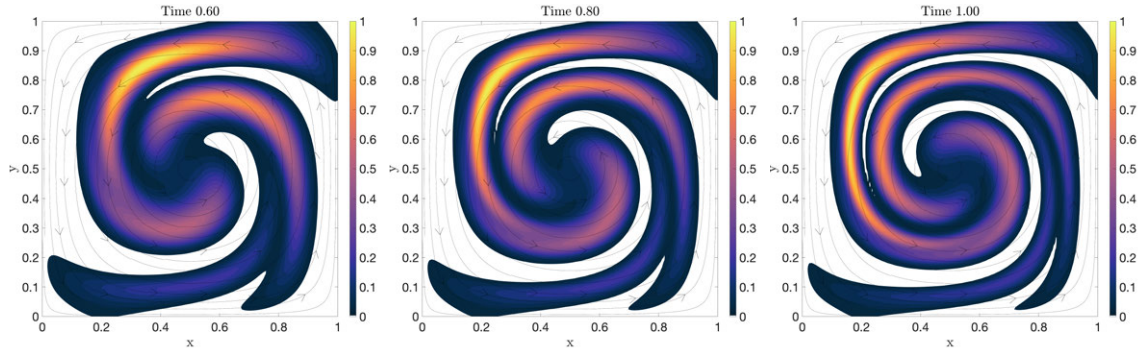
$$\boldsymbol{v}(x,y,t) = \left(\sin^2(\pi x)\sin(2\pi y), -\sin^2(\pi y)\cdot\sin(2\pi x)\right) . \tag{8.1}$$

The velocity field $\boldsymbol{v}(\boldsymbol{x},t)$ satisfies incompressibility, but it is not a solution of the Navier–Stokes' equations. The advection of the tracer field $\alpha(\boldsymbol{x},t)$ is simulated using the method of composition (see Chapter 3) for $t \in [0,1]$ with a timestep of $\Delta t = 0.01$. Thus we have $N_t = 100$. The simulation domain is discretized such that $N_x = N_y = 100$. The exact flow maps, which are never seen by the ML algorithm and only used for estimating errors in the machine inference, are also computed using the method of composition. The initial condition for the tracer field is chosen to be 4 randomly located Gaussian bumps of random intensities. Fig. (8-1) shows the advected tracer field at various times in $[0,1]$. These advected tracer fields (on the (x,y) grid) at the 100 time instants are the only data seen by our algorithm.

We use backpropagation through time (BPTT) [257] with RMSProp optimizer [19] and a learning rate of 10^{-3} and a decay rate of 0.9 to train our ML algorithm. The loss is minimized until the relative difference is less than 0.01 for 10 consecutive iterations, indicating that it has stabilized. Table 8.1 lists the neural network parameters that are used for the machine inference and prediction of the flow maps in this case. The hyperparameters of the ML algorithm are tuned by an exhaustive search over a pre-defined range.



(a) $t = 0.0$ **(b)** $t = 0.2$ **(c)** $t = 0.4$



(d) $t = 0.6$ **(e)** $t = 0.8$ **(f)** $t = 1.0$

Figure 8-1: Tracer advection in a steady swirl flow. Panel (a) shows the initial condition of the tracer which is 4 randomly initialized Gaussian bumps. The proceeding five panels show the eventual advection of the tracer at the various times mentioned. Our flow map inference algorithm sees such 101 fields (one field for each time instance $t = 0, 0.01, \ldots, 0.99, 1.00$).

Table 8.1: Parameters for flow map inference and prediction in the swirl flow.

Parameter	Value
Number of hidden units in the hyperLSTM (p)	10
Convolutional kernel size in the hyperLSTM	3×3
Convolutional kernel size in the main convLSTM	3×3
Number of convLSTM-hyperLSTM layers	5

Using the advected tracer data $\alpha_i(\boldsymbol{x})$ with $i = 0, 1, \ldots, 100$, we infer the flow maps $\boldsymbol{\phi}_0^{i\Delta t}(\boldsymbol{x})$ for $i = 0, 1, \ldots, 100$ (*i.e.* over the time interval $[0, 1]$) and predict the flow maps for $i = 101, 102, \ldots, 200$, *i.e.* for the time interval $(1, 2]$. Note that in this case the flow field is steady and the domain is closed. This consequently implies that the flow maps for the swirl flow are independent of the actual start (or end) time but

only depend on the duration of the flow map. This is represented by Eq. (8.2),

$$\boldsymbol{\phi}_t^{t+\Delta t}(\boldsymbol{x}) = \boldsymbol{\phi}_0^{\Delta t} \quad \text{and} \quad \boldsymbol{\phi}_{t+\Delta t}^{t}(\boldsymbol{x}) = \boldsymbol{\phi}_{\Delta t}^{0} \quad \forall\, t\,. \tag{8.2}$$

This implies that all the incremental flow maps $\boldsymbol{\phi}_{i\Delta t}^{(i+1)\Delta t}$ are equal, and one can write:

$$\boldsymbol{\phi}_0^{i\Delta t}(\boldsymbol{x}) = \boldsymbol{\phi}_0^{\Delta t}\left(\boldsymbol{\phi}_0^{\Delta t}\left(\ldots \boldsymbol{\phi}_0^{\Delta t}\left(\boldsymbol{x}\right)\ldots\right)\right) = \left[\boldsymbol{\phi}_0^{\Delta t}\right]^i\left(\boldsymbol{x}\right). \tag{8.3}$$

The purpose of the hyperLSTM in our algorithm is to allow the main LSTM weights to vary in time to account for the changing incremental flow maps $\boldsymbol{\phi}_i^{i+1}$. However, given that our flow field is steady, we can enforce that the incremental flow maps are the same by removing hyperLSTM altogether and only using the main convLSTM. We expect this version of our network without the hyperLSTM to perform better for steady flows as it enforces the equality of the intermediate flow maps, and also reduces the effective number of trainable parameters. We thus compare results obtained with and without the use of a hyperLSTM in this steady flow field.

For the hyperparameters mentioned in Table 8.1, the number of trainable parameters for the version with the hyperLSTM is 462,238 and for the version without the hyperLSTM is 40,086. As the number of trainable parameters is lower in the latter case (even though the network maintains its ability to perfectly learn the underlying dynamical system), we expect the quality of inference and predictions to be better.

Fig. (8-2) shows the inferred flow maps, the corresponding exact flow maps and the error for $t = 1$, for both the variants of the algorithm. One can see that the version without the hyperLSTM is better at inferring the flow maps. However, for either cases, the error norms are small at $t = 1$ with a relative error of 0.076 for the case with the hyperLSTM and 0.059 for the case without the hyperLSTM.

Fig. (8-3) plots the loss function values with training iterations (epochs) for the two versions of the ML algorithm. Although the loss generally decreases for both the versions, we can clearly see that the loss value stabilizes at about 0.088 for the version with the hyperLSTM whereas it stabilizes at around 0.052 for the version without the hyperLSTM, indicating that the inferred flow maps are better fit to the provided

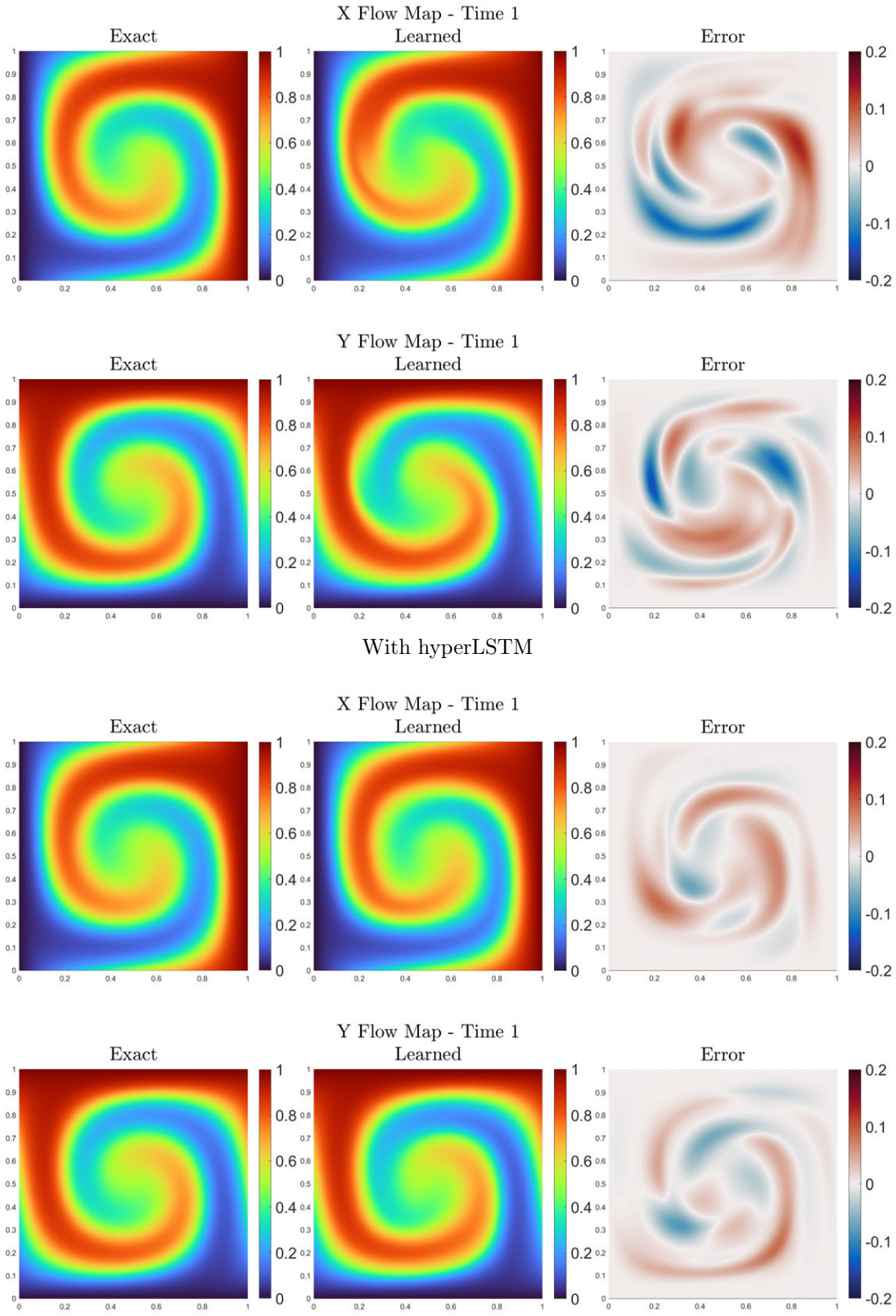


Figure 8-2: Exact and inferred X and Y flow maps along with the corresponding errors for both variants of the algorithm (*i.e.* with and without hyperLSTM) at $t = 1$

tracer transport data by the latter version.

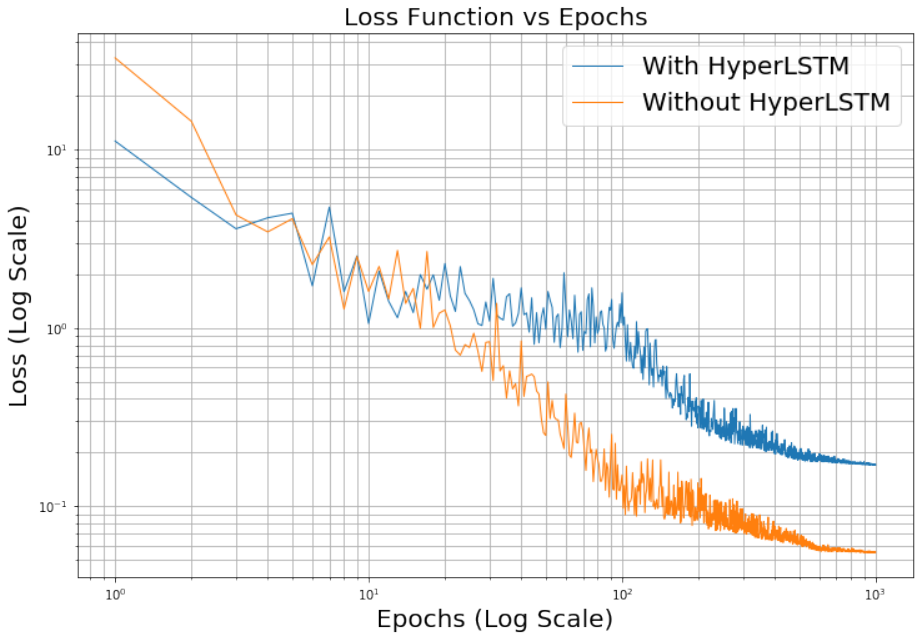


Figure 8-3: Training loss values for both variants of the algorithm. It can be seen that the variant without the hyperLSTM has a lower loss value than the variant with the hyperLSTM due to less trainable parameters and thus a smaller optimization search space.

Fig. (8-4) plots the predicted flow maps by running the LSTM beyond $t = 1$. We look at the flow maps at $t = 2$ to observe that our ML algorithm predicts the flow maps very well even for times when no tracer data is available, with a relative error of about 0.124 for the version with the hyperLSTM and 0.098 for the version without the hyperLSTM. As expected, we see that the relative errors are larger than those during the inference time due to the unavailability of tracer data.

Finally, Fig. (8-5) shows the actual and the learned trajectories of passive particles (derived from the flow maps above) at $t = 1$ and $t = 2$. One can clearly see that the learned trajectories are a good approximation to the actual trajectories throughout the entire time period considered. The learned trajectories from the variant without the hyperLSTM are consistently better than the variant with the hyperLSTM.

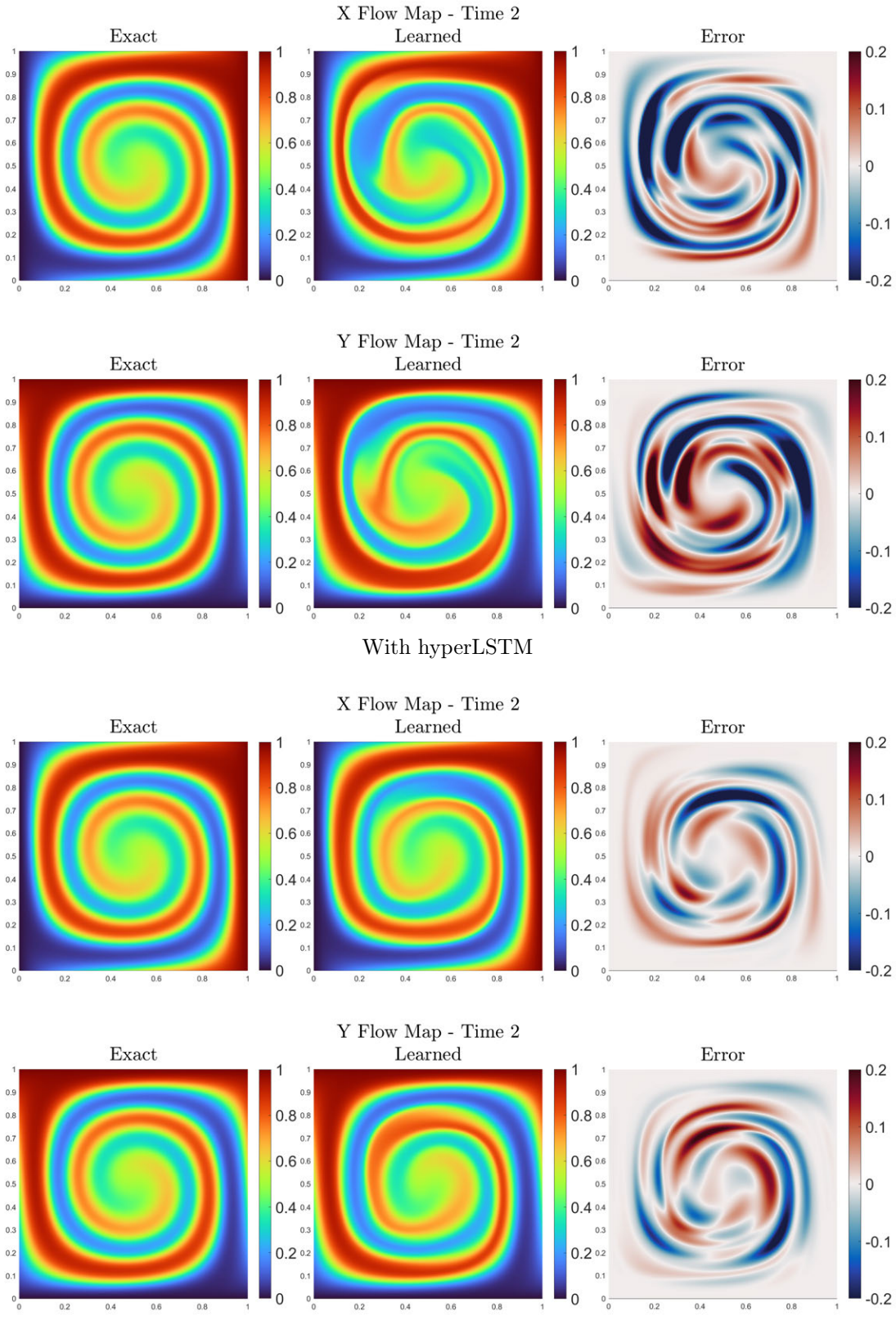
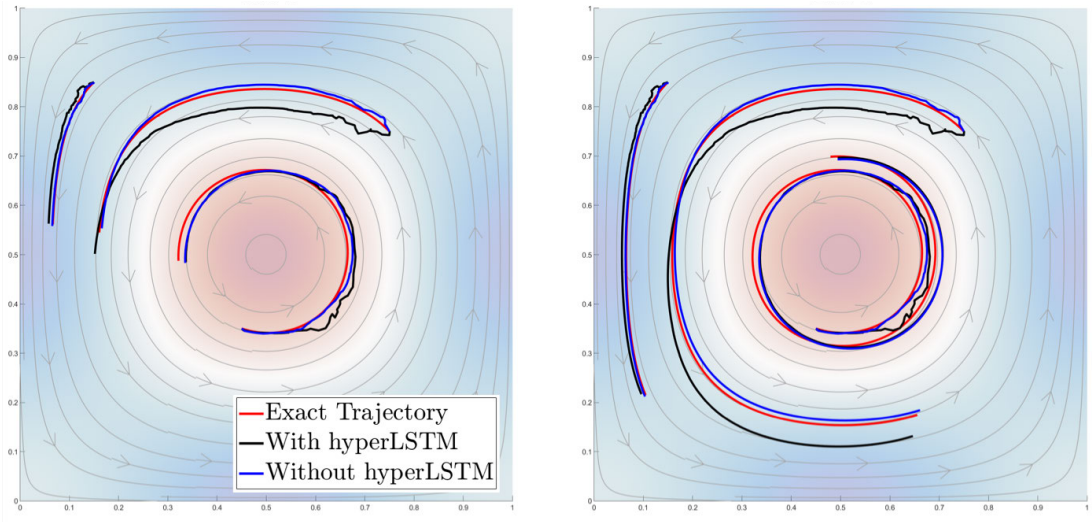


Figure 8-4: Exact and predicted X and Y flow maps along with the corresponding errors for both variants of the algorithm (*i.e.* with and without hyperLSTM) at $t = 2$



(a) $t = 1$ **(b)** $t = 2$

Figure 8-5: Inferred and predicted sample trajectories (constructed by using the inferred / predicted flow maps) for 3 sample passive particles starting at different locations. One can see that although both the variants are able predict the trajectories well, the variant without the hyperLSTM consistently does better.

8.2 Analytical Double Gyre

We now look at the performance of our algorithm in an unsteady flow field. As seen before in Sec. 6.2.1, the analytical double gyre is an unsteady flow that involves two vortices abreast of each other rotating in opposite directions, while the separatrix between the vortices oscillates horizontally. It is described analytically by Eq. (8.4):

$$\begin{aligned} \boldsymbol{v}(\boldsymbol{x}, t) &= \left(-\frac{\partial \boldsymbol{\psi}(\boldsymbol{x}, t)}{\partial y}, \frac{\partial \boldsymbol{\psi}(\boldsymbol{x}, t)}{\partial x} \right) , \\ \boldsymbol{\psi}(\boldsymbol{x}, t) &= A \sin \left(\pi \left(\epsilon \sin (\omega t)\, \boldsymbol{x}^2 + (1 - 2\epsilon \sin (\omega t))\, \boldsymbol{x} \right) \right) . \end{aligned} \tag{8.4}$$

We use $A = 0.1, \omega = 0.2\pi$, and $\epsilon = 0.1$ (same as Sec. 6.2.1). The computational domain Ω is $[0, 2] \times [0, 1]$. Like the swirl flow, this field is also incompressible but not a solution of the Navier–Stokes' equations. However, it mimics certain oceanic flow fields such as the quasi-geostrophic double gyre flow [263]. As mentioned before, even though the unsteady double gyre is a smooth periodic flow, one observes chaotic tracer transport, or 'Lagrangian turbulence', manifested through thin tracer filaments and

sharp boundaries between these filaments [9]. Such chaotic tracer transport presents a challenging test for our machine inference and prediction algorithm.

8.2.1 Inference and Prediction of Flow Map and Coherent Structures

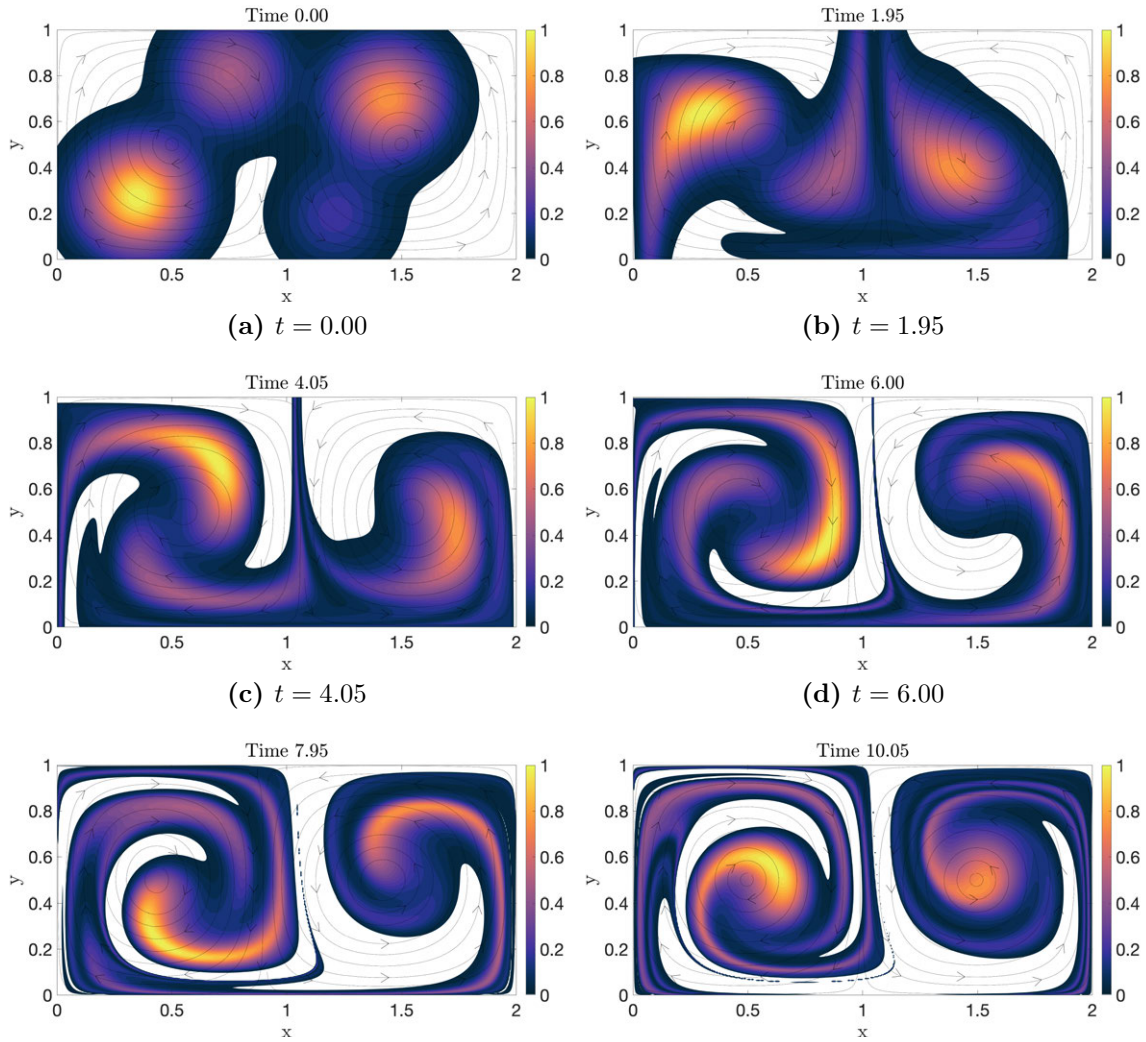
Similar to the swirl flow, The advection of the tracer field $\alpha(\boldsymbol{x}, t)$ and the computation of the exact flow maps is done using the composition-based advection scheme for $t \in [0, 10.05]$ with a timestep of $\Delta t = 0.15$. Thus we have $N_t = 67$. The simulation domain is discretized such that $N_x = 200$ and $N_y = 100$. The initial tracer field is chosen to be 4 randomly located Gaussian bumps of random intensities.

Fig. (8-6) shows the advected tracer field at various times in $[0, 10.05]$. As before, these advected tracer fields (on the (x, y) grid) are the only data fed to our algorithm to learn the flow maps. Using this data, we plan to infer the flow maps $\boldsymbol{\phi}_0^t$ for $t \in [0, 10.5]$, *i.e.* $\boldsymbol{\phi}_0^{i\Delta t}(\boldsymbol{x})$ for $i = 1, \ldots, 67$ and also predict the flow maps for $t \in (10.05, 15]$, *i.e.* $\boldsymbol{\phi}_0^{i\Delta t}(\boldsymbol{x})$ for $i = 68, \ldots, 100$ when no tracer data is available. Table 8.2 lists the network parameters used for the machine inference and prediction of the flow maps, which are tuned by an exhaustive search over a pre-defined range.

Table 8.2: Parameters for flow map inference and prediction in the analytical double gyre.

Parameter	Value
Number of hidden units in the hyperLSTM (p)	10
Convolutional kernel size in the hyperLSTM	3×3
Convolutional kernel size in the main convLSTM	5×5
Number of convLSTM-hyperLSTM layers	10

Similar to the previous case, we test two versions of our algorithm: one with the hypernetwork and other one without. As the flow field at hand is now dynamic, we expect the former version to perform significantly better even though it has larger number of trainable parameters (2,343,528), as the latter version (with 76,204 trainable parameters) is unable to account for the time-evolving nature of the flow field. We expect the latter version to learn an 'averaged' version of the incremental flow



(a) $t = 0.00$ **(b)** $t = 1.95$

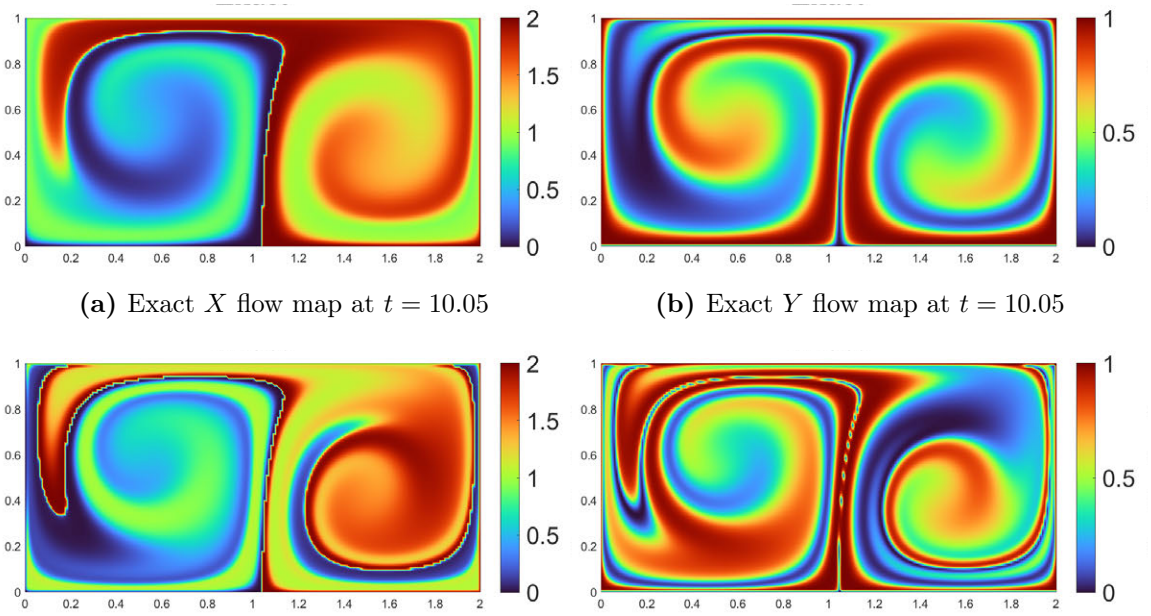
(c) $t = 4.05$ **(d)** $t = 6.00$

(e) $t = 7.95$ **(f)** $t = 10.05$

Figure 8-6: Tracer advection in an unsteady double gyre flow. Panel (a) shows the initial condition of the tracer which is 4 randomly initialized Gaussian bumps. The proceeding five panels show the eventual advection of the tracer at the various times mentioned. Our flow map inference algorithm sees such 68 fields (one field for each time instance $t = 0, 0.15, \ldots, 9.90, 10.05$.

maps ($\boldsymbol{\phi}_i^{i+1}$) in this case. Fig. (8-7) shows the exact X and Y flow maps for the analytical double gyre at $t = 10.05$ and $t = 15$. These are only used to compare the inferred and predicted fields by the different variants of the algorithm.

Fig. (8-8) shows the inferred flow maps for the unsteady double gyre along with the corresponding error for $t = 10.05$ for both the versions of the algorithm. In this case, as the flow is unsteady, we observe that the version with the hyperLSTM does



(a) Exact X flow map at $t = 10.05$

(b) Exact Y flow map at $t = 10.05$

(c) Exact X flow map at $t = 15$

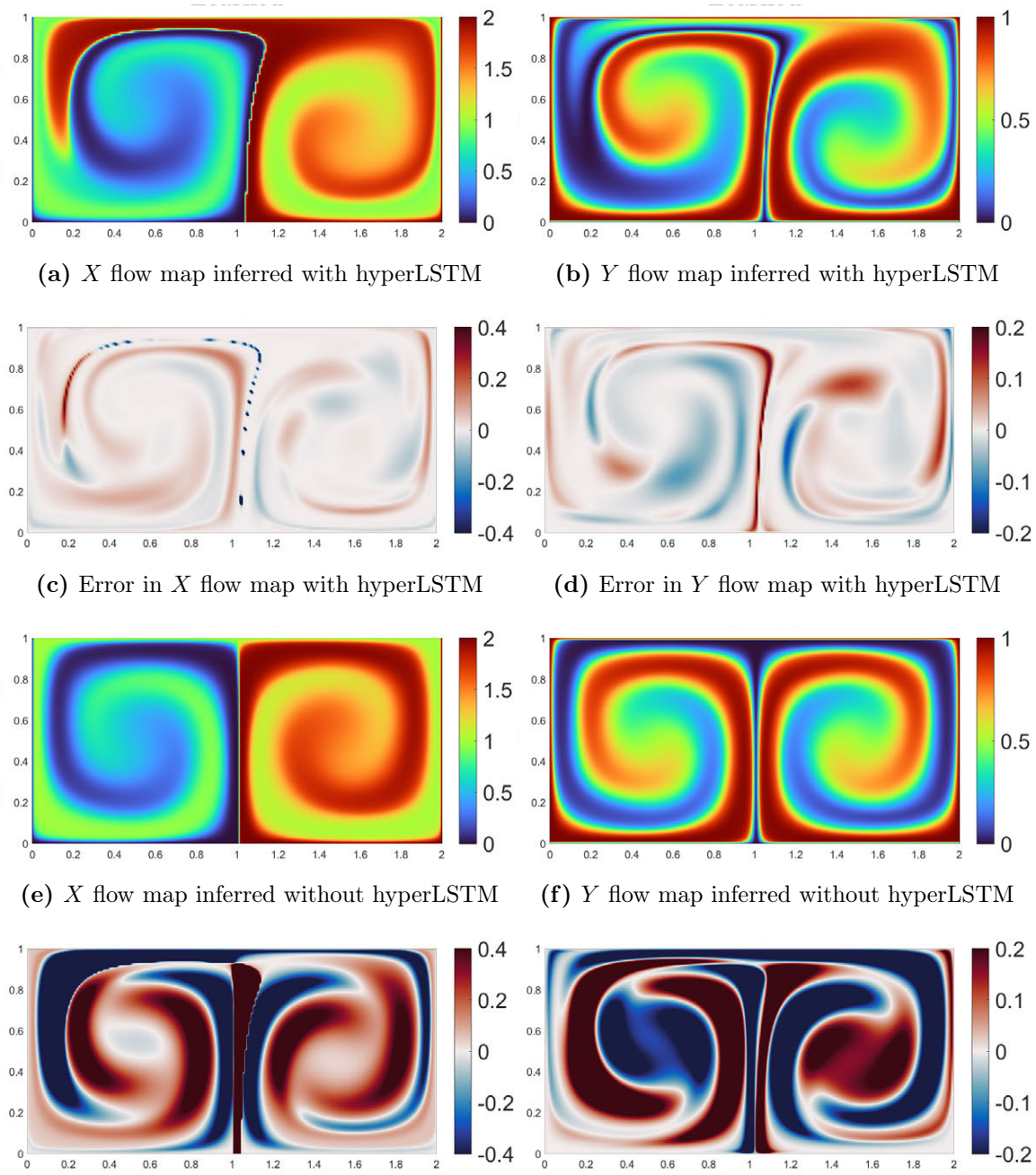
(d) Exact Y flow map at $t = 15$

Figure 8-7: Exact forward X and Y flow maps at $t = 10.05$ and $t = 15$ for the analytical double gyre flow.

significantly better at inferring the flow maps. However, the version without the hyperLSTM learns a steady approximation of the underlying unsteady flow maps. The relative error made by the version with the hyperLSTM at $t = 10.05$ is 0.097 whereas the one for the version without the hyperLSTM is 0.320.

Fig. (8-9) shows the predicted forward flow maps between $t = 0$ and $t = 15$, when the algorithms are trained on 68 tracer snapshots between times $t = 0$ and $t = 10.05$. We can clearly see that the version with the hyperLSTM predicts the flow map well, and the errors are high near the edges of the protruding lobes. This simply indicates that our algorithm makes an error in learning the exact location of the lobe, which is expected. The version without the hyperLSTM performs poorly in prediction, as it can only learn a constant incremental flow map to be composed recursively in time. The relative errors made by the two variants in predicting the flow map at $t = 15$ is about 0.130 and 0.446 respectively.

The training losses while inferring the flow maps for the unsteady double gyre are plotted in Fig. (8-10). We can clearly see that the version without the hyperLSTM



(a) X flow map inferred with hyperLSTM

(b) Y flow map inferred with hyperLSTM

(c) Error in X flow map with hyperLSTM

(d) Error in Y flow map with hyperLSTM

(e) X flow map inferred without hyperLSTM

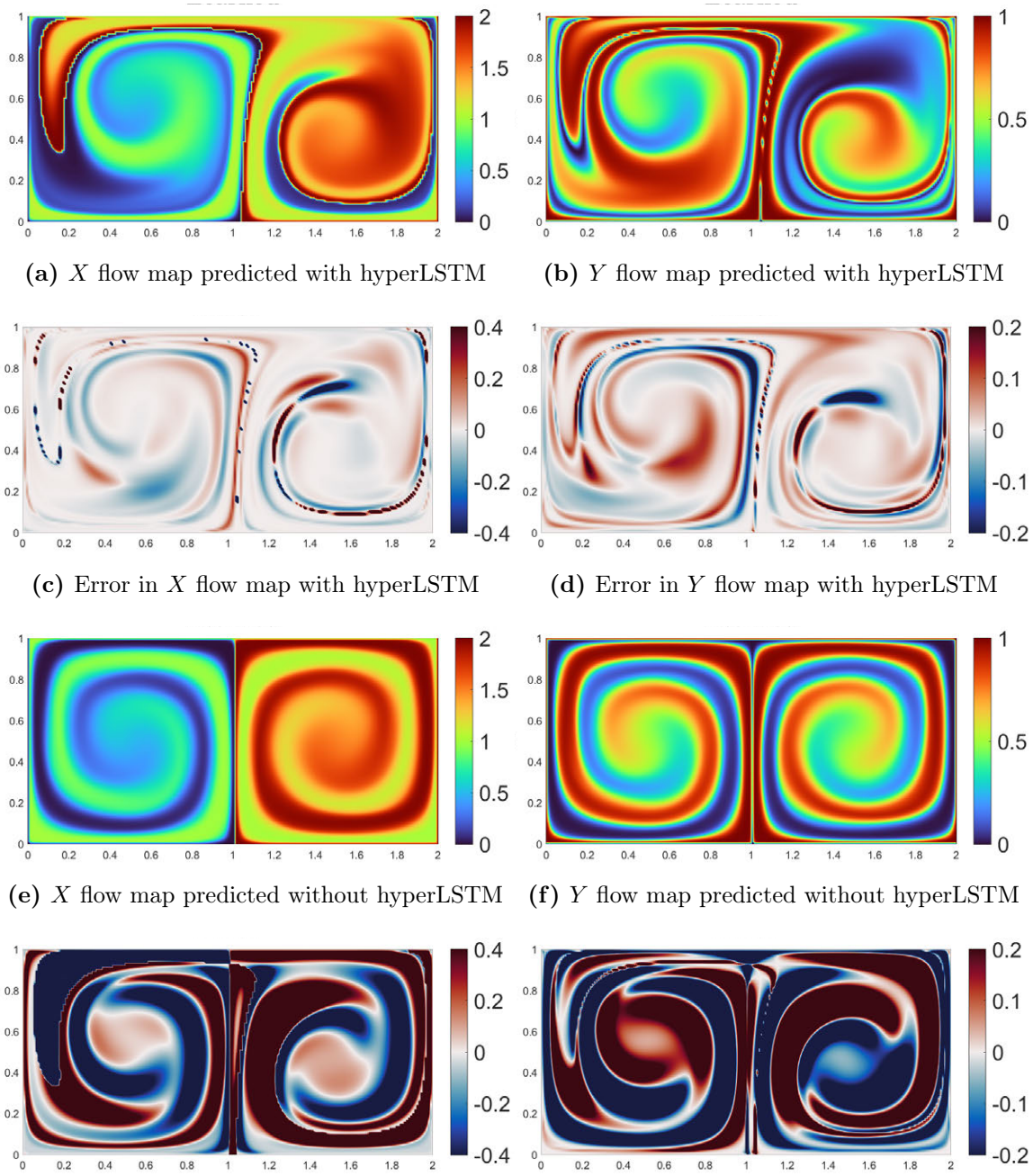
(f) Y flow map inferred without hyperLSTM

(g) Error in X flow map without hyperLSTM

(h) Error in Y flow map without hyperLSTM

Figure 8-8: Exact and inferred forward X and Y flow maps along with the corresponding errors at $t = 10.05$ for the variants of the algorithm with and without a hyperLSTM

only learns steady incremental flow map, and hence it is never able to fit the flow maps well to the available data, resulting in a large loss value. Specifically, the tracer advection loss component is large and non-decaying for this version, as the tracer



(a) X flow map predicted with hyperLSTM

(b) Y flow map predicted with hyperLSTM

(c) Error in X flow map with hyperLSTM

(d) Error in Y flow map with hyperLSTM

(e) X flow map predicted without hyperLSTM

(f) Y flow map predicted without hyperLSTM

(g) Error in X flow map without hyperLSTM

(h) Error in Y flow map without hyperLSTM

Figure 8-9: Exact and predicted forward X and Y flow maps along with the corresponding errors at $t = 15$ for the variant of the algorithm with a hyperLSTM

transport cannot be described through constant incremental flow maps. However, the version with the hyperLSTM is able to fit the data well and thus has a decreasing loss value, and a low final loss.

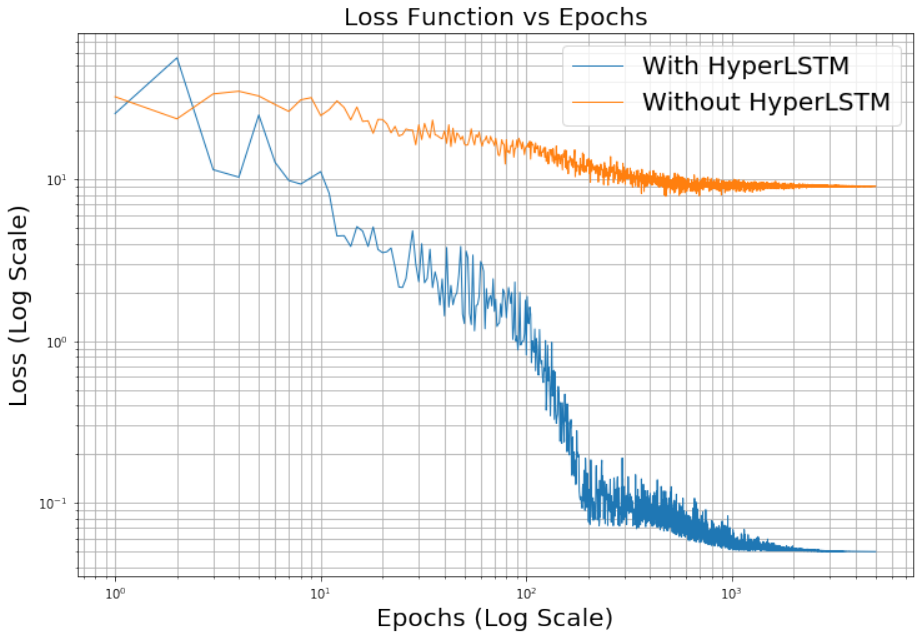
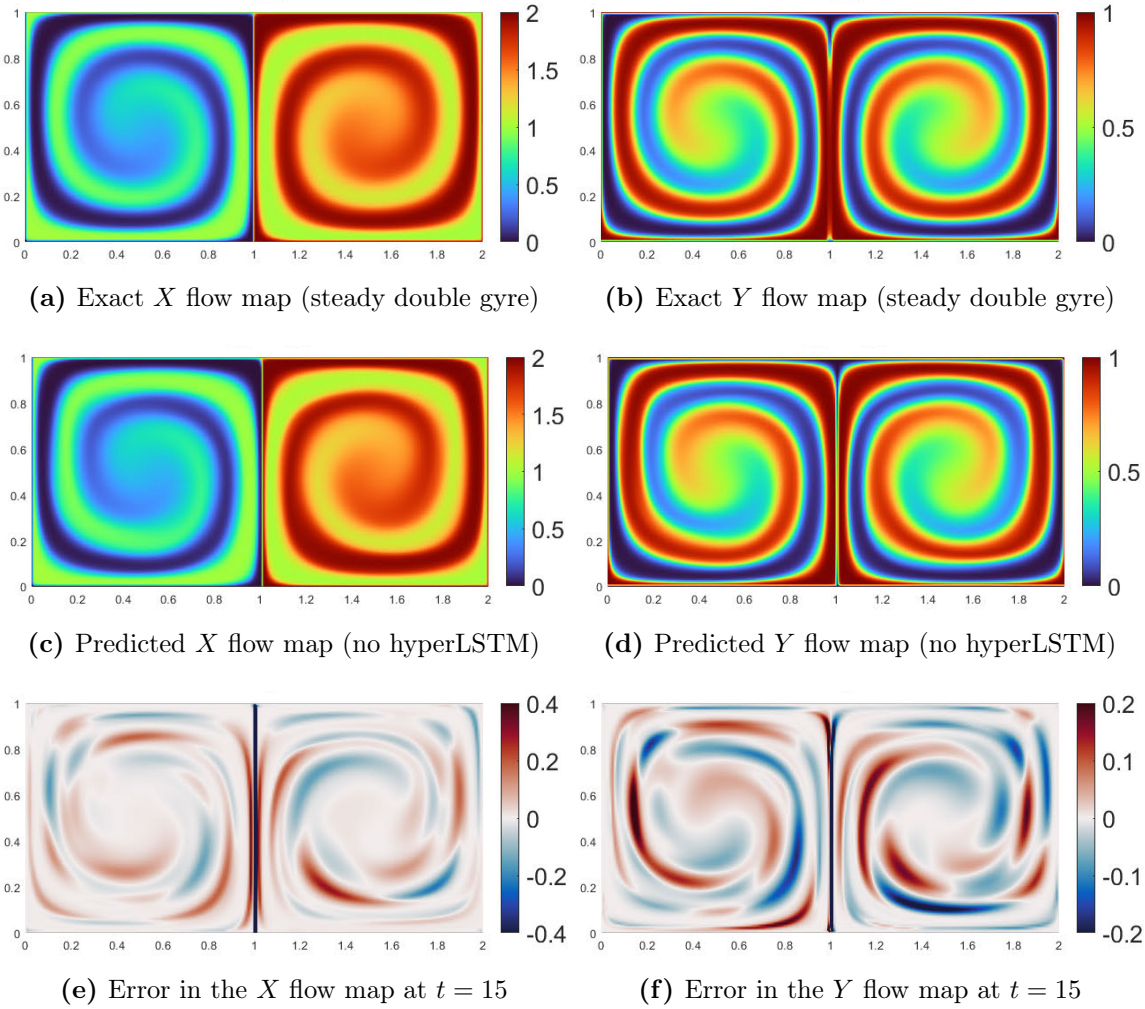


Figure 8-10: Training loss values for both variants of the algorithm in the analytical double gyre. It can be seen that the variant without the hyperLSTM cannot fit the (steady) flow maps to the available data and hence its loss value remains high throughout. However, the variant with the hyperLSTM fits the dynamic flow maps well to the available data, yielding a low loss value.

As mentioned before, the version of the algorithm without the hyperLSTM learns an averaged version of the unsteady incremental flow maps. Hence, we compare the flow maps generated by the said version with the exact flow maps arising from a steady double gyre flow (*i.e.* $w = 0$ in Eq. (8.4)) at $t = 15$ in Fig. (8-11). We can clearly see that the flow maps that our algorithm learns are very close to the steady double gyre flow maps. This highlights that in an unsteady flow field, the version without the hyperLSTM learns the flow maps arising from a steady approximation of the unsteady field.

Fig. (8-12) plots the exact forward FTLE field for the unsteady and steady double gyre and the predicted forward FTLEs by the two versions between $t = 0$ and $t = 15$. One can see that the version with the hyperLSTM accurately captures the prominent ridges of the FTLE field, while the version without the hyperLSTM predicts an FTLE field close to that of a steady double gyre flow - especially noticeable is the absence of the ridges delineating the lobes in either gyres and the presence of a strong vertical



(a) Exact X flow map (steady double gyre)

(b) Exact Y flow map (steady double gyre)

(c) Predicted X flow map (no hyperLSTM)

(d) Predicted Y flow map (no hyperLSTM)

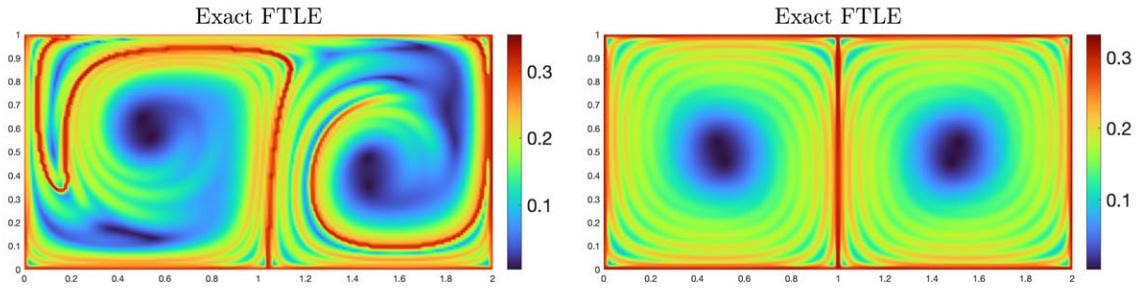
(e) Error in the X flow map at $t = 15$

(f) Error in the Y flow map at $t = 15$

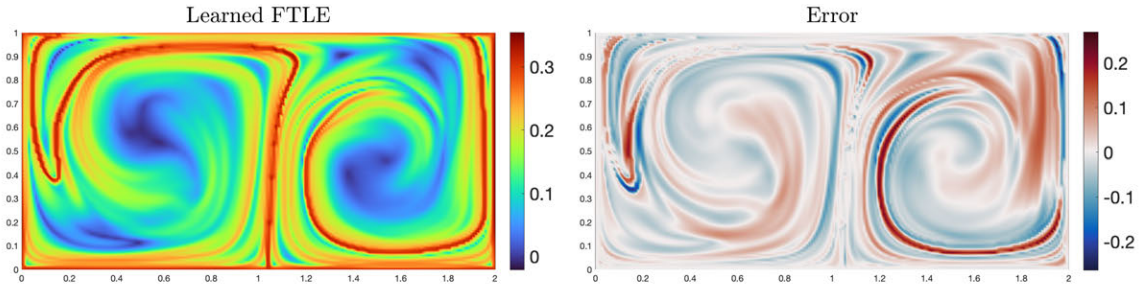
Figure 8-11: Forward flow maps predicted by the variant of the machine learning algorithm without hyperLSTM at $t = 15$ show great agreement when compared to the exact flow maps of a steady double gyre flow. This suggests that the variant of our algorithm without the hyperLSTM learns the closest steady approximation to the underlying unsteady flow.

ridge at the separatrix.

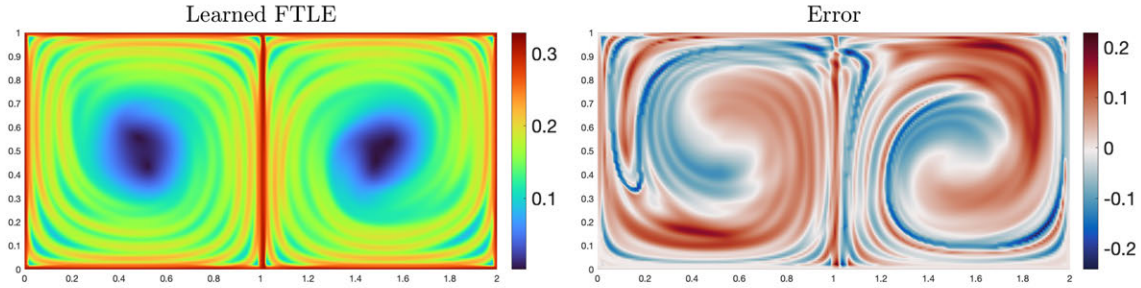
Finally, we look at the inferred rigid sets and their predicted evolution from our algorithm (with the hyperLSTM). In Fig. (8-13), we show the rigid sets and their evolution obtained by thresholding the polar distance metric (Eq. (5.13)) computed using the learned forward flow map over the time interval $[0, 15]$. The evolution of these sets is computed by composing the initial position of the sets with the corresponding learned flow maps. Comparing the evolution of these sets to the 'exact' evolution from Fig. (6-12), we can clearly see that our framework learns these rigid



(a) Exact forward FTLE: unsteady double gyre **(b)** Exact forward FTLE: steady double gyre



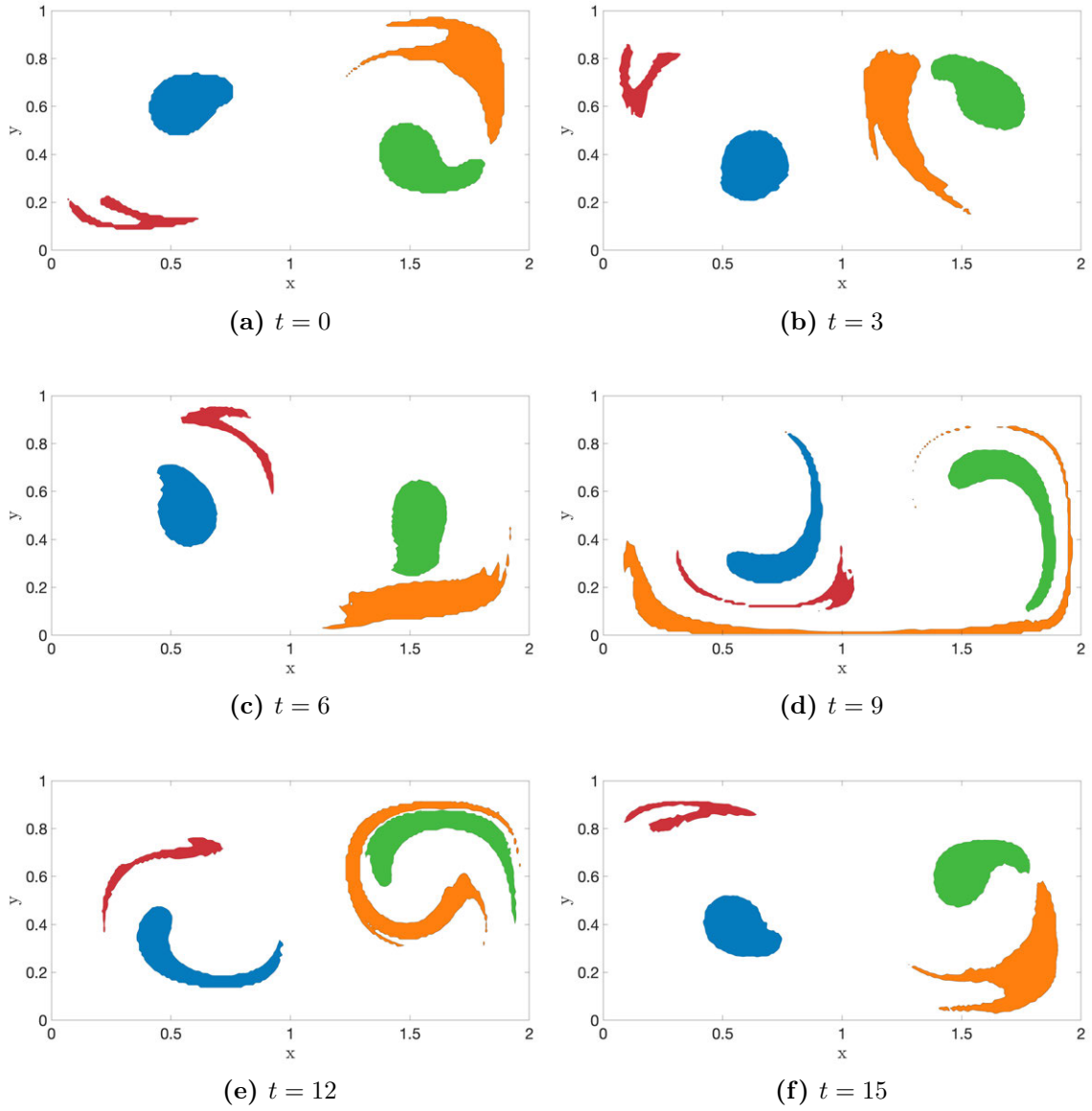
(c) Learned forward FTLE with the hyperLSTM and the corresponding error



(d) Learned forward FTLE without the hyperLSTM and the corresponding error

Figure 8-12: The exact and predicted forward FTLEs between $t = 0$ and $t = 15$ along with the corresponding errors for the two variants of the algorithm. In line with the prior observations, the predictions from the algorithm with the hyperLSTM are very close to the truth, and the errors are larger only at the ridges of the FTLE field. However, the version without the hyperLSTM is only able to learn the steady version of the flow maps, and hence its predicted FTLE field is similar to that of a steady double gyre (panel (b)).

sets extremely well, without any information about the underlying flow field.



(a) $t = 0$ **(b)** $t = 3$ **(c)** $t = 6$ **(d)** $t = 9$ **(e)** $t = 12$ **(f)** $t = 15$

Figure 8-13: Evolution of the inferred rigid sets using the developed ML framework. The evolution of these inferred rigid sets almost exactly matches the corresponding 'exact' evolution from Fig. (6-12), suggesting that our algorithm learns such coherent structures well without any information about the underlying flow field.

8.2.2 Information Content and Predictive Capability

Information Content of the Tracer Field

Based on the theory developed in Sec. 7.3.5, we quantify information content of the tracer field in terms of the fraction of total grid points where no error is made in

flow map prediction, which we show to be a function of the number of unique values in the tracer field α (denoted by r). Specifically, we consider different simulation configurations, namely $N_x = 20, N_y = 10$ ($N = 200$), $N_x = 40, N_y = 20$ ($N = 800$), and $N_x = 50, N_y = 25$ ($N = 1250$). Fig. (8-14) plots the fraction of grid points without an error in the flow map inference at $t = 10.05$ as a function of the number of unique values in the tracer field α. Each result is a 10-run average, and the shaded areas denote the associated standard errors. This plot can be qualitatively split into three distinct regions, indicated by the vertical gray lines. The first region corresponds to tracer fields with too few distinct values - we observe poor learning in this case, as expected. The second region, with r/N values in $[0.25, 0.75]$ reflects most physical tracer fields. The observed fraction of locations without an error is comparable to the analytical value for all N in this region, as the ill-conditioned problem is solved by imposing incompressibility and velocity smoothness constraints in addition to the tracer transport data. Finally, in the third region with $r/N > 0.75$, the observed values saturate, their standard errors increase, and they fall below the analytical value. This is mainly due to the errors involved in the machine inference algorithm. At such high r/N values, the tracer fields are increasingly random. Even though this is ideal to learn from for a perfect algorithm, the numerical machine inference algorithm suffers from errors and thus the accuracy saturates.

Predictive Capability of the Inference Algorithm

Fig. (8-15) quantifies the predictive capability of our neural network for the case of the analytical double gyre. As stated before, the predictive capability of our machine inference algorithm constitutes of two key components: (i) the predictive power of the trained algorithm, and (ii) the predictability limit of the underlying dynamical system. We first consider the relative error in the predicted tracer field when our neural network is trained using tracer data from the time interval $[0, 10.05]$ for different simulation configurations, as seen in Fig. (8-15a). We can see that for all configurations, the results (which are a 10-run average) have low errors while inferring the tracer fields (*i.e.* in the time interval $[0, 10.05]$). These errors start to

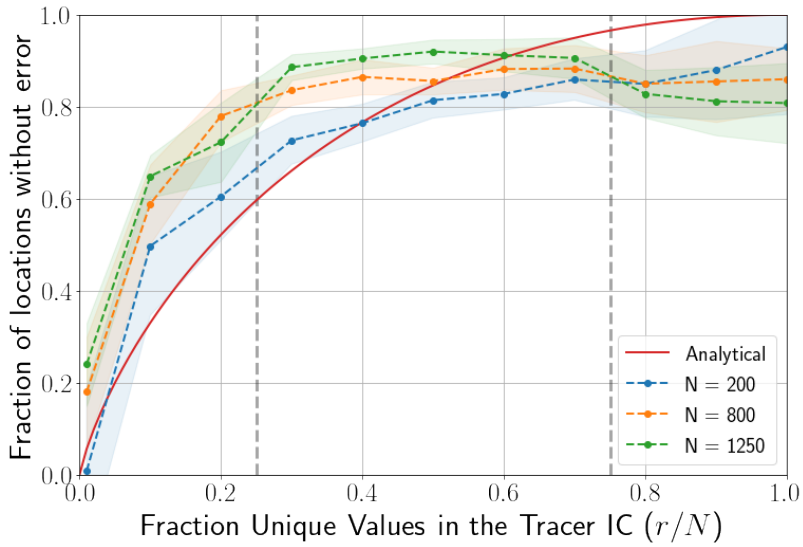
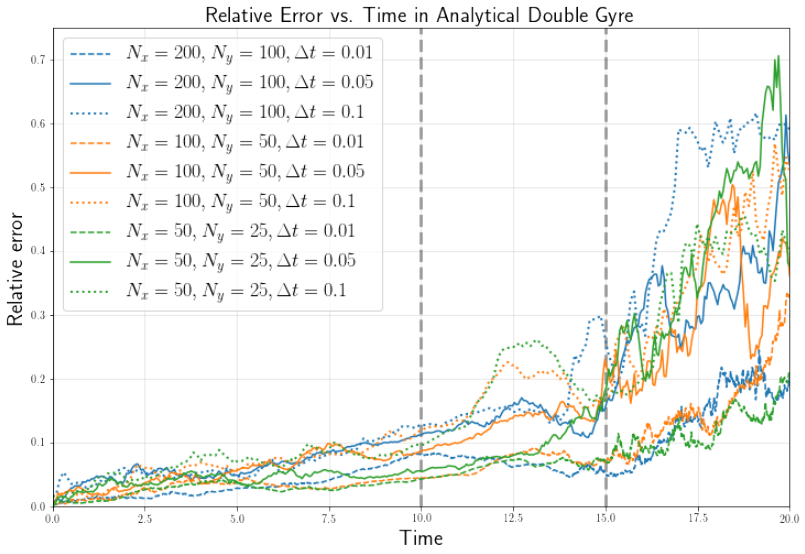


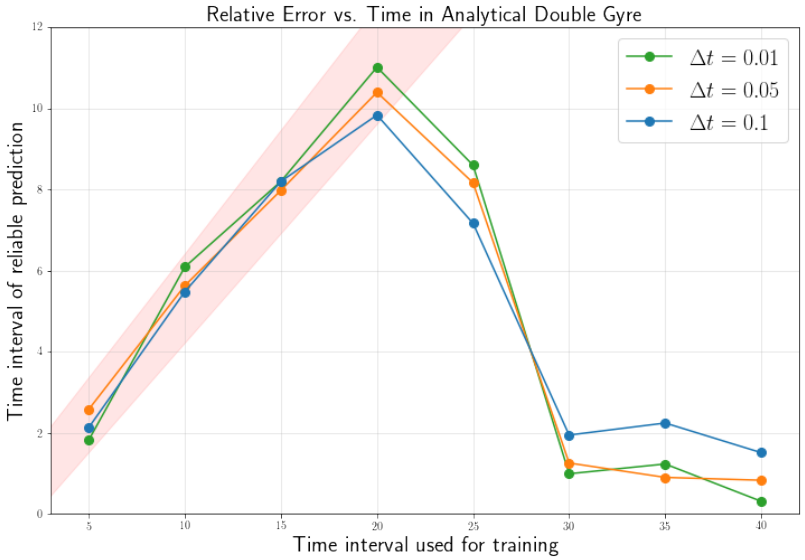
Figure 8-14: Information content, quantified by the fraction of locations in Ω where the flow map value is correctly predicted, versus the fraction of unique values in the tracer field α_0 (normalized by the number of grid points N).

increase after during the prediction phase (*i.e.* $t = 10.05$), however, they do not grow fast until about $t = 15$. After $t \approx 15$, the errors are erratic and significantly larger in magnitude. One can thus say that the when trained on the data from the time interval $[0, 10.05]$, our neural network can predict the flow maps reliably for up to $t = 15$ for the analytical double gyre. It is interesting to observe that this predictive power of the algorithm is largely independent of the numerical discretization as well as the timestep.

To illustrate the interplay between the predictive power of the trained algorithm, and the predictability limit of the underlying dynamical system, we plot the size of the time interval of reliable prediction against the size of the time interval used for training in Fig. (8-15b). The time interval for reliable prediction is determined by considering the rolling average of the relative errors, and thresholding it to be twice that of the average relative error over the inference interval. We find that there is largely a linear relationship between these quantities (indicated by the shaded red cone) until the size of the training interval is about 20. This indicates that the predictive capability of our algorithm in this region is limited by the amount of data it sees. The more data it sees, the more it can predict. Beyond this point, the size of



(a) Relative error in tracer fields vs. time



(b) Reliable prediction vs. training time interval

Figure 8-15: Quantifying the predictive capability of our ML algorithm for the analytical double gyre flow. Panel (a) looks at the relative error in the inferred and predicted tracer fields as a function of time for different simulation configurations. Each time-series is a 10-run average for the considered configuration. Panel (b) shows the size of the time interval with reliable prediction versus the size of the time interval used for training the network in the case of $N_x = 200$ and $N_y = 100$ for different Δt values.

the interval with reliable prediction rapidly drops and we actually observe that the more we train our algorithm, the less it is able to predict. This occurs because we reach the predictability limit of the analytical double gyre for such a large training

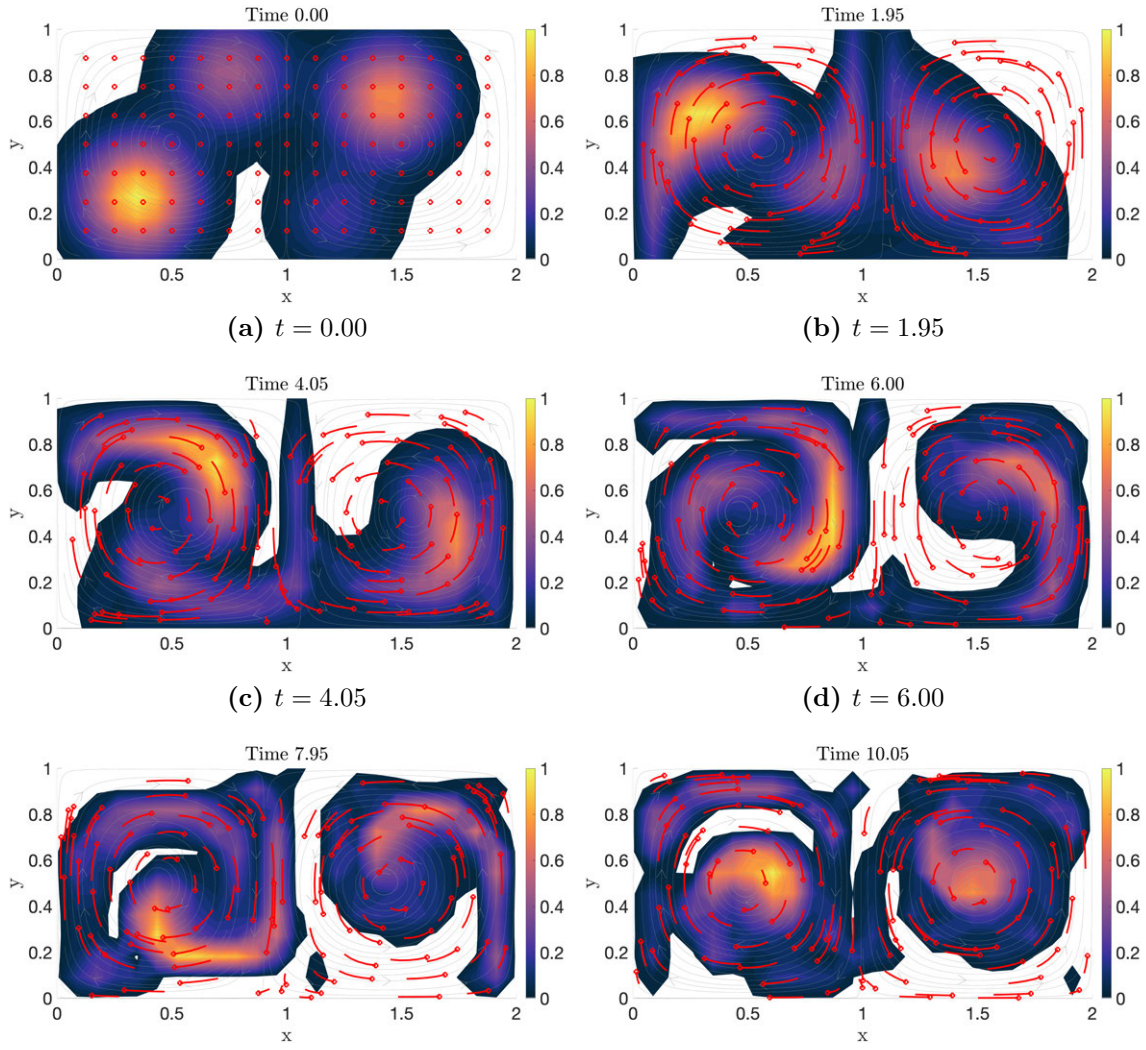
time interval size. Beyond this point, the more data that our algorithm sees, the more it gets confused and is unable to predict reliably.

8.2.3 Merging Eulerian and Lagrangian Information

To infer high resolution flow maps from low resolution Eulerian data combined with Lagrangian trajectories in the analytical double gyre, we consider the following setup. We simulate the low resolution Eulerian tracer snapshots by considering the same four Gaussian bumps at $t = 0$ as Fig. (8-6), but now on a 20×10 grid. As before, our machine learning algorithm is presented with 68 snapshots of this evolving tracer field (on the 20×10 grid) between times $t = 0$ and $t = 10.05$, with a uniform timestep $\Delta t = 0.15$. In addition to this Eulerian data, we also present our algorithm data about 105 Lagrangian trajectories, originating at 15 uniformly spaced X coordinates and 7 uniformly spaced Y coordinates, at the same time instants as the tracer fields. The data seen by our algorithm is plotted in Fig. (8-16), where the Lagrangian trajectories are overlaid in red on top of the coarse Eulerian tracer field.

As before, our goal is to infer the flow maps $\boldsymbol{\phi}_0^t$ for $t \in [0, 10.5]$, *i.e.* $\boldsymbol{\phi}_0^{i\Delta t}(\boldsymbol{x})$ for $i = 1, \ldots, 67$ and also predict the flow maps for $t \in (10.05, 15]$, *i.e.* $\boldsymbol{\phi}_0^{i\Delta t}(\boldsymbol{x})$ for $i = 68, \ldots, 100$ using the above-mentioned Eulerian and Lagrangian information at a high resolution, *i.e.* on a 200×100 grid.

Fig. (8-17) shows the inferred forward flow maps at $t = 10.05$. Specifically, we compare results obtained from three approaches: the first row shows the X and the Y flow maps obtained only by using the coarse resolution Eulerian tracer transport data. As the Eulerian data is only available on a 20×10 grid, the corresponding results are also only available on this coarse grid. The inferred flow maps lack any fine structure but coarsely resemble the actual underlying flow maps. The second row shows results obtained only by using Lagrangian trajectory data. This is achieved by dropping the tracer advection loss term from Eq. (7.37). These results are very interesting. The inferred flow maps are almost unchanged from the original x and y grids, but the initial locations of the considered Lagrangian trajectories bear very different flow map values as compared to their neighborhood. Such a structure arises



(a) $t = 0.00$ **(b)** $t = 1.95$

(c) $t = 4.05$ **(d)** $t = 6.00$

(e) $t = 7.95$ **(f)** $t = 10.05$

Figure 8-16: Coarse resolution tracer advection along with high resolution trajectories in an unsteady double gyre flow. Panel (a) shows the initial condition of the tracer which is 4 randomly initialized Gaussian bumps along with the starting locations of the 105 Lagrangian trajectories. The proceeding five panels show the eventual advection of the tracer and the motion of the Lagrangian particles at the various times mentioned. Our flow map inference algorithm sees such 68 fields (one field for each time instance $t = 0, 0.15, \ldots, 9.90, 10.05$.

because the drifter position loss component ($\mathcal{L}^i_{\text{drft}}$) correctly enforces the flow map value to be the current trajectory position at the trajectory initial locations, but provides no information about the flow map anywhere else in the domain. At the same time, the incompressibility and velocity smoothness loss components force the flow map field to be as smooth and regular as possible everywhere else in the domain. Thus the X and the Y flow maps appear to be very close to the x and y grids

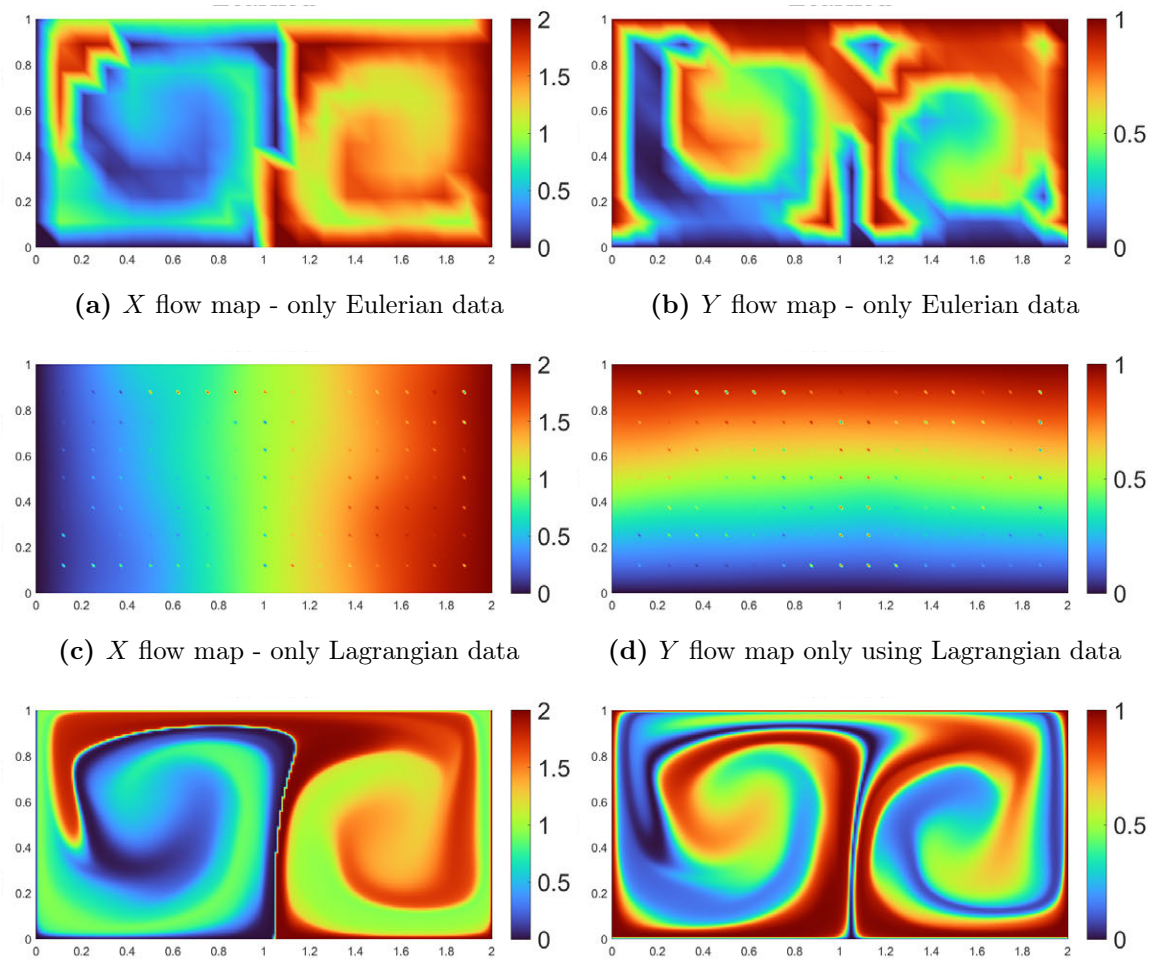
respectively at locations other than the trajectory initial locations.

Ideally, we want our algorithm to utilize the coarse Eulerian data to infer the basic underlying structure, and utilize the Lagrangian information to locally correct the flow maps to yield the high resolution results. This is exactly what we observe in the last row of Fig. (8-17). Our algorithm is able to use the coarse Eulerian field along with high resolution Lagrangian trajectories to infer flow maps at a high resolution that are very close to the true fields (see panels (a) and (b) of Fig. (8-7)). Obviously, the inferred fields are less accurate than those predicted using high resolution tracer transport data (panels (a) and (b) of Fig. (8-8)). This is expected, as high resolution tracer data can be thought of as Lagrangian data at all locations on a fine grid, which provides much more information than only using Lagrangian information from the 105 locations. However, these inferred flow maps are able to satisfyingly capture the general structure of the gyres as well as elements of chaotic advective transport through the presence of long and thin filaments.

Fig. (8-18) shows the predicted forward X and Y flow maps at $t = 15$. Similar to Fig. (8-17), we see that when only using the Eulerian snapshots, we are able to predict the general structure of the flow maps but only at a coarse resolution. When only using Lagrangian data, our algorithm again predicts a background fields close to the x or y grids with very different values at the trajectory start locations. However, the variant of the algorithm utilizing both Eulerian and Lagrangian information predicts the X and Y flow maps that are close to the exact fields (Fig. (8-7)(c) and (d)).

8.2.4 Learning from Advective–Diffusive Tracers

Finally, we look at learning the flow maps from data about an advective–diffusive tracer field (see Sec. 7.4). In this work, we assume that the (unknown) diffusivity of the tracer, denoted by κ, is a constant. However, straightforward extensions are possible wherein the diffusivity may be allowed to vary or be parametrized, as mentioned earlier. The simulation setup is the same as that in Sec. 8.2.1. However, to incorporate the effect of tracer diffusion, we utilize Lie splitting (see Sec. 3.3.3). As the diffusion operator is numerically stiff, it is solved implicitly using a central differ-



(a) X flow map - only Eulerian data

(b) Y flow map - only Eulerian data

(c) X flow map - only Lagrangian data

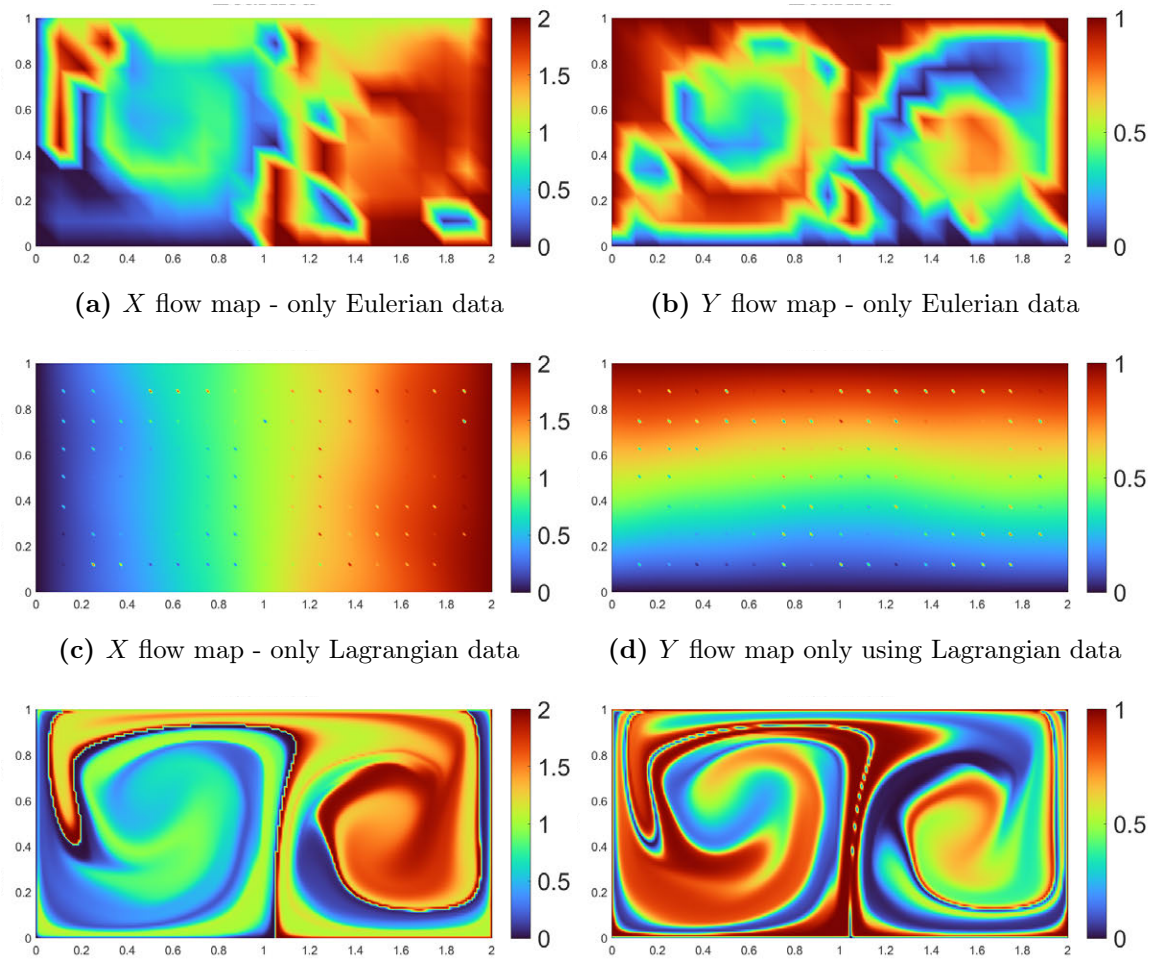
(d) Y flow map only using Lagrangian data

(e) X flow map - Eulerian and Lagrangian data

(f) Y flow map - Eulerian and Lagrangian data

Figure 8-17: Forward X and Y flow maps at $t = 10.05$ inferred only using Eulerian information, *i.e.* coarse tracer field (row 1), only using Lagrangian information, *i.e.* trajectories (row 2)), and using both the Eulerian and Lagrangian information simultaneously (row 3). These can be compared to the corresponding exact fields from Fig. (8-7).

ence scheme for the Laplacian and backward Euler time marching. Fig. (8-19) shows the snapshots of the diffusive tracer field with diffusivity $\kappa = 5 \times 10^{-3}$ at 6 different time instants. When compared to Fig. (8-6), it can be clearly seen that tracer diffusion leads to a much smoother field, and the signatures of chaotic advection (such as long and thin filaments) are absent. For larger values of κ or longer time durations, the tracer occupies the entire domain and has a constant, average value everywhere. Alluding to the earlier discussion, this implies that the information in the tracer field decreases with time in advective–diffusive transport.



(a) X flow map - only Eulerian data

(b) Y flow map - only Eulerian data

(c) X flow map - only Lagrangian data

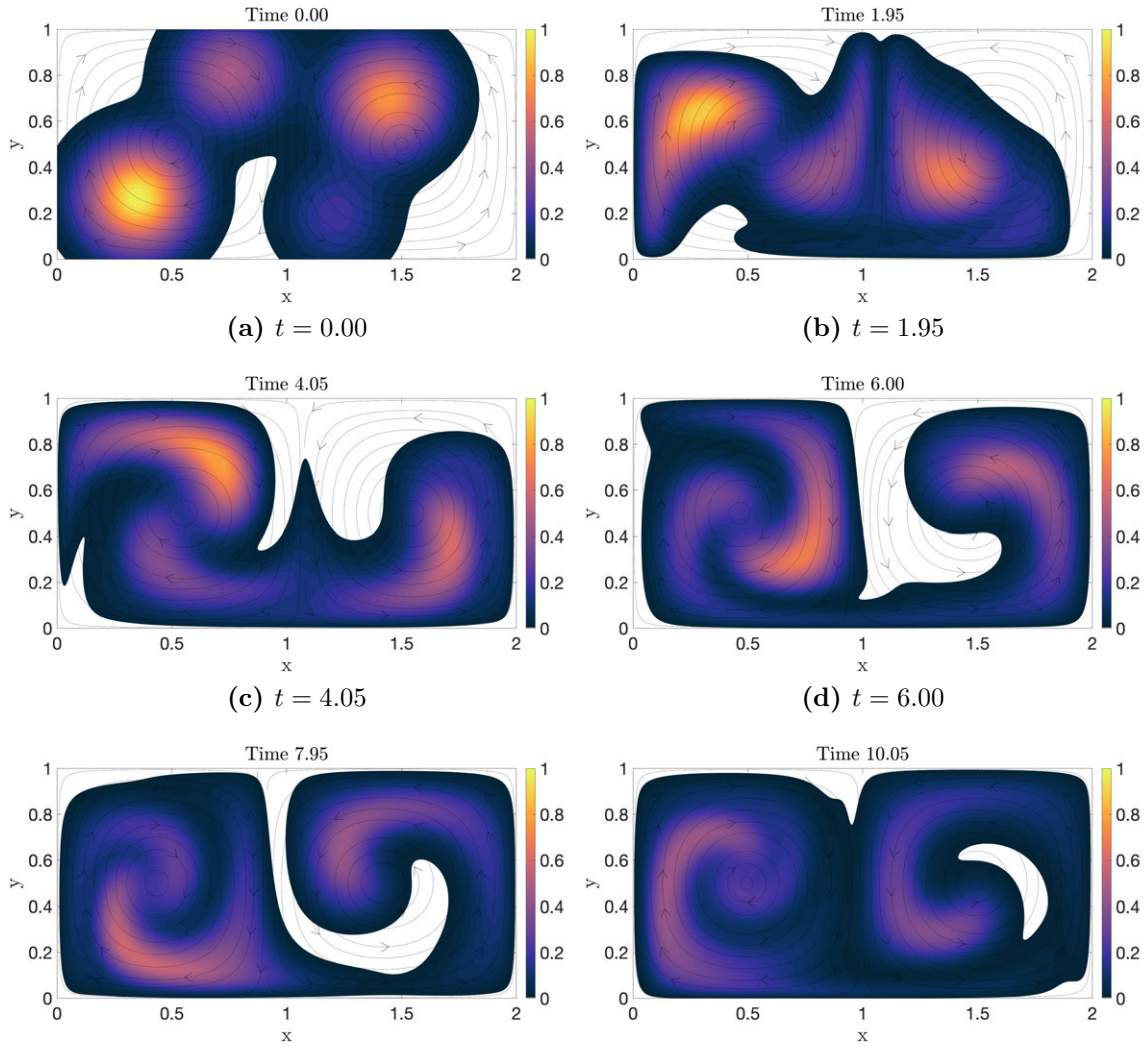
(d) Y flow map only using Lagrangian data

(e) X flow map - Eulerian and Lagrangian data

(f) Y flow map - Eulerian and Lagrangian data

Figure 8-18: Forward X and Y flow maps at $t = 15$ predicted only using Eulerian information, *i.e.* coarse tracer field (row 1), only using Lagrangian information, *i.e.* trajectories (row 2)), and using both the Eulerian and Lagrangian information simultaneously (row 3). These can be compared to the corresponding exact fields from Fig. (8-7).

Fig. (8-20) shows the inferred flow maps and their corresponding errors at $t = 10.05$, whereas Fig. (8-21) shows the predicted flow maps at $t = 15$. The appreciable agreement of these results when compared to the exact flow maps (Fig. (8-7)) suggests that our algorithm effectively distills the underlying advective features even from advective–diffusive tracer transport. Even though the errors are larger than the earlier results, we can see that the critical features of the flow maps (such as the vertical separatrix and the material leakage) are correctly learned by our algorithm. Further, by keeping the diffusivity as a learnable parameter, our neural network is able to infer



(a) $t = 0.00$ **(b)** $t = 1.95$

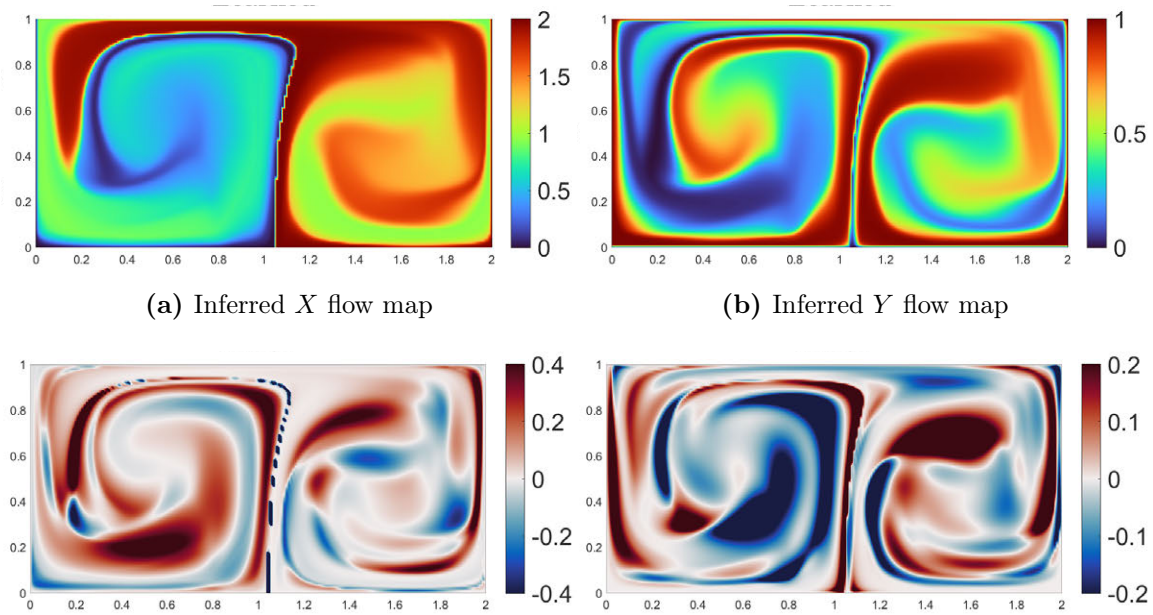
(c) $t = 4.05$ **(d)** $t = 6.00$

(e) $t = 7.95$ **(f)** $t = 10.05$

Figure 8-19: Tracer advection and diffusion in an unsteady double gyre flow with tracer diffusivity $\kappa = 5 \times 10^{-3}$. The initial condition of the tracer field is the same as that from Fig. (8-6). When compared to Fig. (8-6), tracer diffusion can be clearly observed throughout the time interval. Further, the lack of long thin filaments indicates that chaotic advection is suppressed due to diffusion.

its value. The inferred value of κ by the network is 4.65×10^{-3} which is 7% off from the true value of 5×10^{-3}.

We now analyze the learning capability of our algorithm as a function of the tracer diffusivity. Fig. (8-22a) plots the relative error in the inferred flow maps at $t = 10.05$ as a function of the diffusivity κ. We consider two cases: (i) the tracer transport is (incorrectly) modeled as pure advection (as in Sec. 8.2) and (ii) the tracer transport



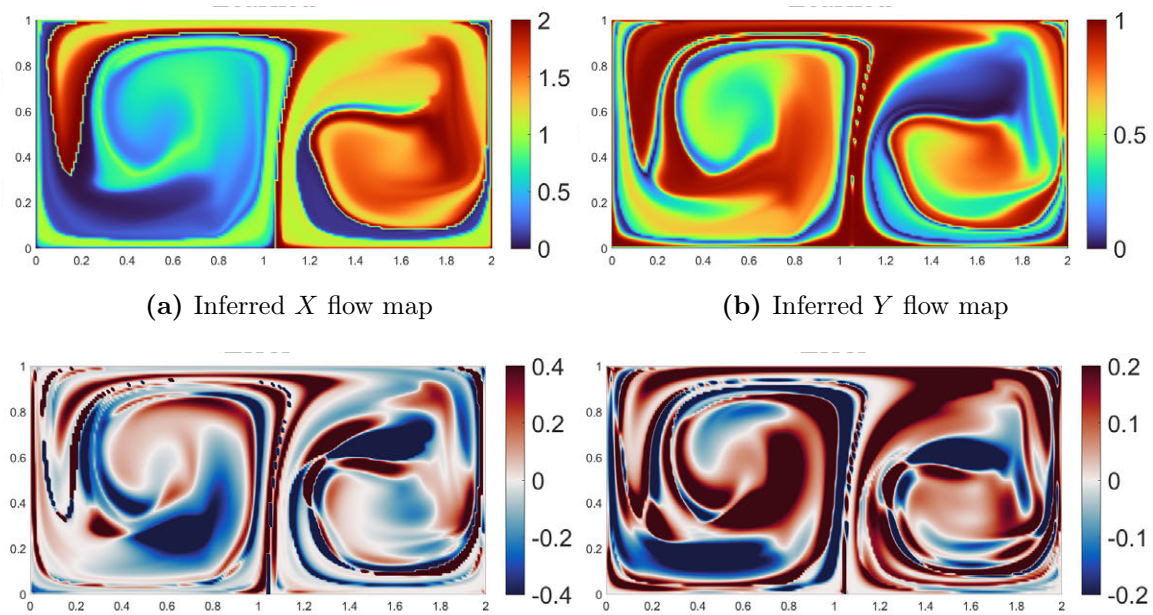
(a) Inferred X flow map

(b) Inferred Y flow map

(c) Error in the X flow map

(d) Error in the Y flow map

Figure 8-20: Forward flow maps at $t = 10.05$ inferred by the variant of the ML algorithm learning from advective–diffusive tracers.



(a) Inferred X flow map

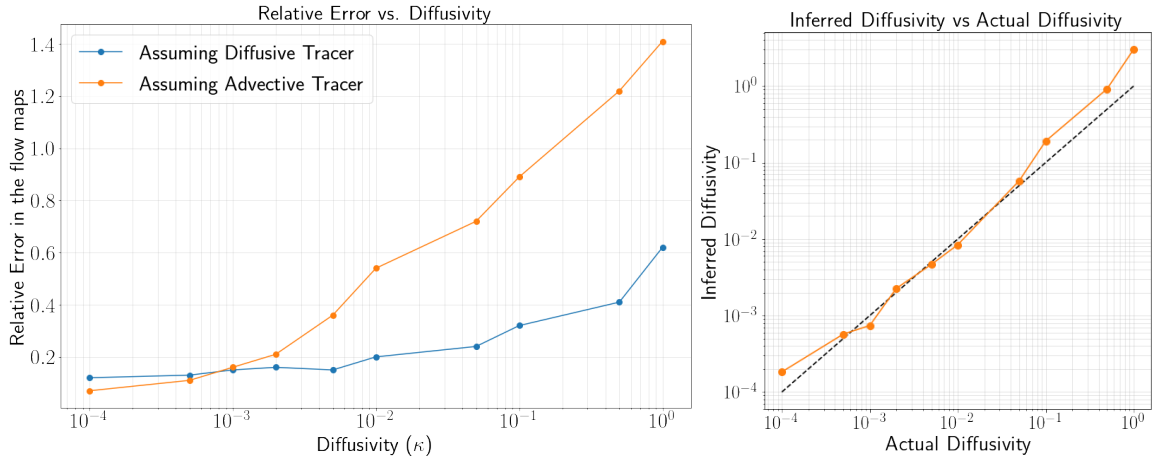
(b) Inferred Y flow map

(c) Error in the X flow map

(d) Error in the Y flow map

Figure 8-21: Forward flow maps at $t = 15$ inferred by the variant of the ML algorithm learning from advective–diffusive tracers.

is (correctly) modeled as an advective–diffusive process (as done above). Clearly the latter case has lower error overall, especially at higher values of κ. However, the former performs better for extremely low κ, as at such low values, the diffusion effect is negligible. The error for both cases is large for large κ values. This is because at high diffusivity, the tracer almost immediately fills the entire domain with a uniform value and the tracer field then contains no information going forward (see Sec. 8.2.2). Fig. (8-22b) plots the inferred diffusivity (learned by the algorithm) against the actual diffusivity. We see that for κ between 10^{-3} and 10^{-1}, our algorithm makes little error in inferring the value. For κ lower than 10^{-3}, the values are off mainly due to the numerical errors as the diffusion process has minimal impact. At high values of κ, inference is poor as the tracer field is too diffusive and quickly fills the entire domain with a constant tracer value.



(a) Relative error in the flow maps with diffusivity **(b)** Inferred against actual diffusivity

Figure 8-22: Performance of the flow map inference algorithm while learning from a diffusive tracer field. Panel (a) shows the relative error in the inferred flow maps at $t = 10.05$ against tracer diffusivity κ. Panel (b) shows the value of κ inferred against the actual κ. We can clearly see that unless the value is extremely low ($< 10^{-3}$) or extremely high ($> 10^{-1}$), our algorithm is able to accurately infer it.

8.3 Western Mediterranean (Alboran) Sea

The final part of this chapter deals with the machine inference and prediction of flow maps and coherent structures in realistic marine domains. Such marine domains most often have open boundaries, *i.e.* there is material inflow and outflow at the domain boundaries. We thus utilize the algorithm developed in Sec. 7.5.3 to predict both the forward and the backward flow maps simultaneously and extract the active domains using them.

To apply our theory and schemes, we consider a realistic ocean flow in the Alboran Sea, which is the westernmost part of the Mediterranean Sea. In this case, we showcase our ML framework by simultaneously inferring forward and backward flow maps over three days and predicting them for the fourth day using data about (simulated) advective tracer transport, sea surface temperature, and salinity.

8.3.1 Modeling Region and Computational Details

The Alboran Sea in the Western Mediterranean has a strong semi-permanent front between the fresher Atlantic water that enters the Western Mediterranean at Gibraltar, and the more saline Mediterranean waters. It is populated by organized, time-evolving features (jets, fronts, and gyres) that provide an ideal test bed for our dynamical systems-based Lagrangian analysis. Specifically, the Western Alboran Gyre near Gibraltar, whose Lagrangian signature is clearly visible through flow maps and coherent structures, provides a good case to compare and contrast the predictions obtained while using different types of data.

As before, the dynamic ocean fields (currents, temperature, salinity, and ocean free-surface) are simulated by our MIT Multidisciplinary Simulation, Estimation and Assimilation System (MSEAS) [96; 155; 95] (see Sec. 4.3.1). The present MSEAS simulation domain covers a 430.13 $km \times 266.53$ km region, as seen in Fig. (8-23). The domain is discretized to have $N_x = 773$ and $N_y = 480$ with a total integration time of 96 hours from 20 March 2019, 00Z to 24 May 2019, 00Z. However as this domain is too large for our computational setup, we upscale the spatial grid by a factor of 3 to

arrive at a resolution of $N_x = 258$ and $N_y = 160$. The initial ocean conditions were downscaled from the coarse analysis fields of 20 March 2019 of the 1/12° HYCOM (Hybrid Coordinate Ocean Model) [41]. The higher-resolution MSEAS simulations were then ran up to 24 May 2019, with full tidal forcing [160]. All the domain boundaries are open and there is significant fluid inflow and outflow.

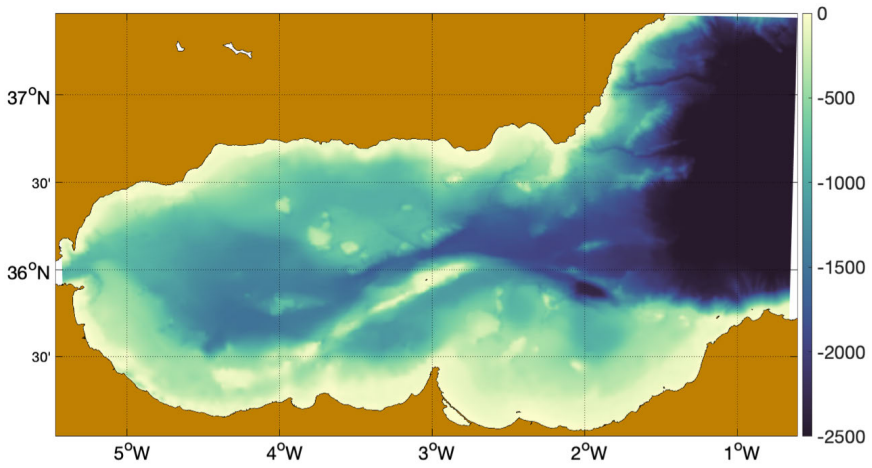
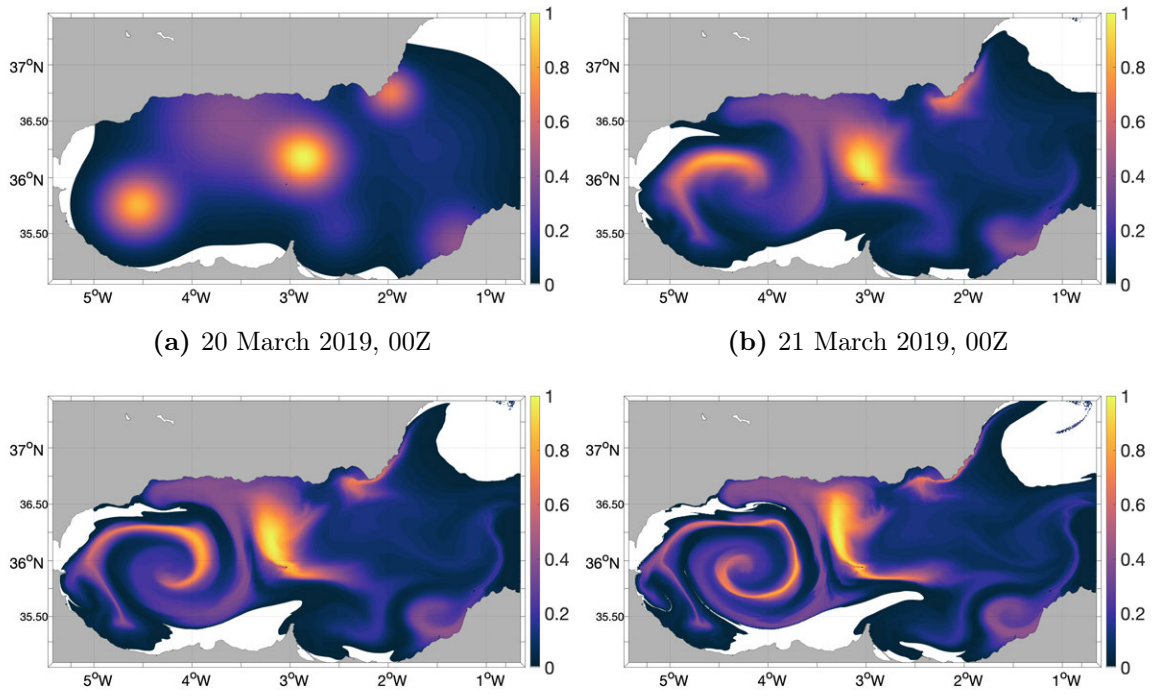


Figure 8-23: The MIT-MSEAS modeling domain in the Western Mediterranean (Alboran) Sea along with the bathymetry of the region.

Specifically, we consider the variants where our ML algorithm is provided information about: (i) simulated transport of an advective tracer, (ii) sea surface temperature (SST), and (iii) sea surface temperature and sea surface salinity (SSS) together. The initial simulated tracer field consists of 8 Gaussian bumps of random intensity, and its evolution is obtained through the method of composition, using the true velocity field. The sea surface temperature and salinity are obtained directly through the MSEAS regional ocean modeling system.

These fields are provided with $\Delta t = 1$ hour, from 20 March 2019, 00Z to 23 March 2019, 00Z, *i.e.* at $N_t = 72$ uniformly spaced time instants. We infer the flow maps at all these 72 time instants, as well as predict them for 24 more uniformly space time instants (from 23 March 2019, 01Z to 24 March 2019, 00Z) by running the LSTM for future times. That is, we infer $\boldsymbol{\phi}_0^{i\Delta t}$ and $\boldsymbol{\phi}_{i\Delta t}^0$ for $i = 0, \ldots 72$ and predict $\boldsymbol{\phi}_0^{i\Delta t}$ and $\boldsymbol{\phi}_{i\Delta t}^0$ for $i = 73, \ldots 96$. We compare these results with the corresponding exact flow maps computed using the actual velocity fields. Note that our algorithms only



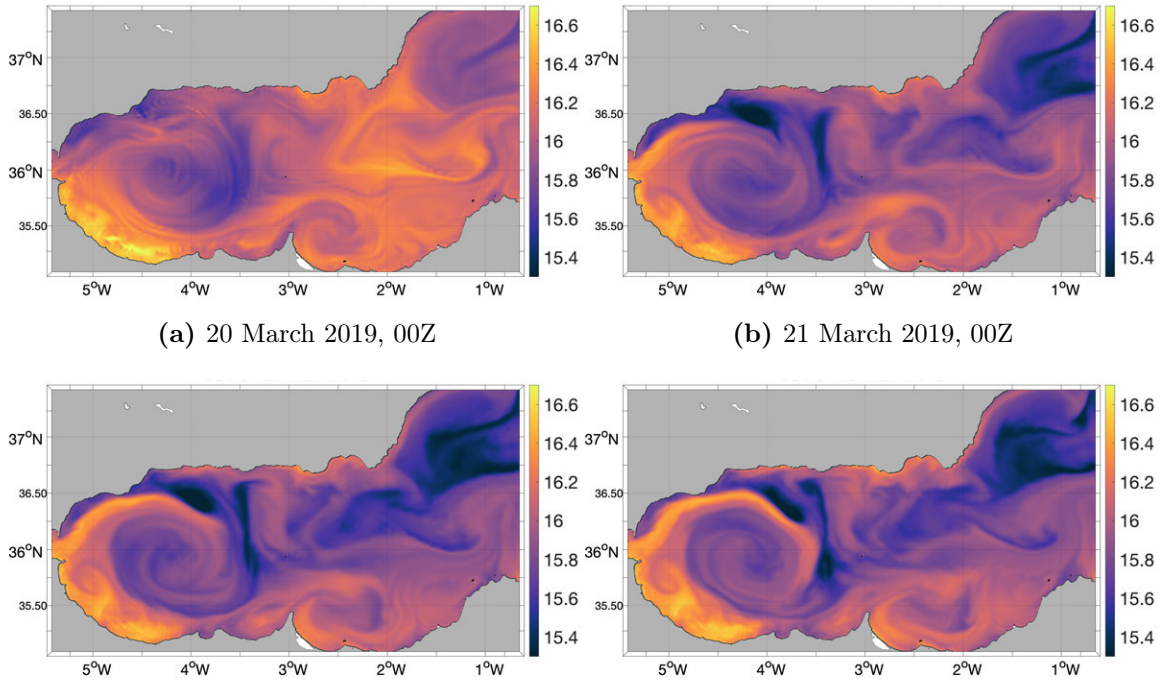
(a) 20 March 2019, 00Z **(b)** 21 March 2019, 00Z

(c) 22 March 2019, 00Z **(d)** 23 March 2019, 00Z

Figure 8-24: The initial and advected tracer field. The initial tracer field consists of 8 Gaussian bumps of random intensities, as seen in panel (a). Panels (b), (c), and (d) show the simulated advection of the tracer after 24 hours, 48 hours, and 72 hours respectively.

receive data until 23 March 2019, 00Z, and no data is seen by the algorithms between 23 March 2019 and 24 March 2019. Fig. (8-24) shows the initial and the advected tracer fields at 00Z on the considered 3 days. Fig. (8-25) and Fig. (8-26) show the SST and SSS fields respectively at the same times as in Fig. (8-24). Through these snapshots, one can clearly see the structure of the Western Alboran Gyre, as well as the other dynamic currents and features affecting tracer transport.

Table 8.3 lists the neural network parameters used for the machine inference and prediction of the flow maps in this case, tuned by a local search starting from those used for the analytical double gyre. The total number of learnable parameters in our neural network setup is 9,365,400. As stated before, we infer the forward and backward flow maps over the considered time window simultaneously and use Algorithm 1 to extract the active domains. We empirically observe that running the iterative domain extraction procedure 3 to 5 times yields good convergence.



(a) 20 March 2019, 00Z **(b)** 21 March 2019, 00Z

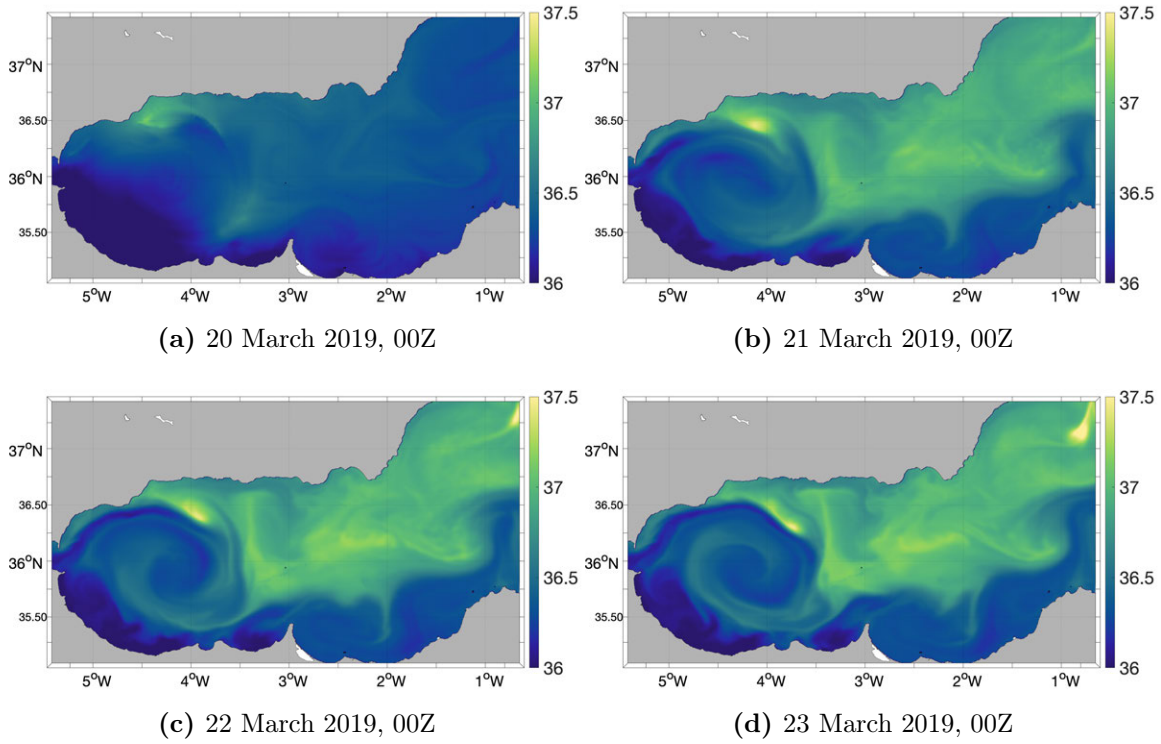
(c) 22 March 2019, 00Z **(d)** 23 March 2019, 00Z

Figure 8-25: The sea surface temperature (SST; in $°C$) field on 20, 21, 22, and 23 March 2019 at 00Z. The SST is governed by an advection–diffusion equation with external forcing.

Table 8.3: Parameters for flow map inference and prediction in the Alboran Sea.

Parameter	Value
Number of hidden units in the hyperLSTM (p)	13
Convolutional kernel size in the hyperLSTM	4×4
Convolutional kernel size in the main convLSTM	4×4
Number of convLSTM-hyperLSTM layers	12

Note that the simulated tracer is governed by an advective transport equation, and as we use the method of composition to compute its evolution, minimum numerical errors are committed. However, SST and SSS are not perfectly advective and include atmospheric forcing (net heat flux, evaporation-precipitation), implemented as flux boundary conditions in the vertical diffusion terms. In the horizontal, a Shapiro filter with a variable equivalent diffusivity is employed. The resulting effective diffusivity (for both the SST and the SSS) is about 800 m^2/s for a 2 grid point resolution (~ 1 km-scales) and 45 m^2/s for a 10 grid point resolution (~ 5 km-scales), resulting in a mean diffusivity of $\kappa = 103.58$ m^2/s. This implies that both the temperature



(a) 20 March 2019, 00Z

(b) 21 March 2019, 00Z

(c) 22 March 2019, 00Z

(d) 23 March 2019, 00Z

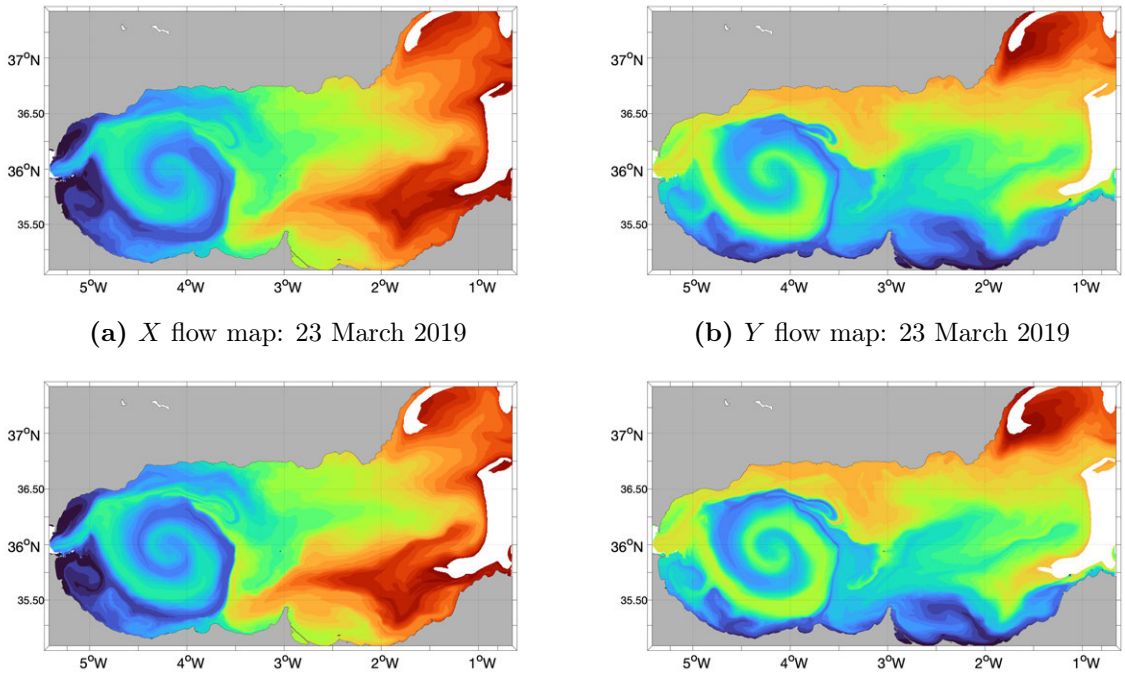
Figure 8-26: The sea surface salinity (SSS; in PSU) field on 20, 21, 22, and 23 March 2019 at 00Z. The SSS is governed through an advection–diffusion equation with external forcing similar to the SST.

and the salinity are governed by an advection–diffusion equation rather than a pure advective transport equation. Hence, we utilize the extension of our methodology that accounts for diffusive tracers to learn the flow maps and the unknown tracer diffusivity from the SST and SSS data. In this work, we assume that SST and SSS have constant and equal diffusivity. Adding this constraint implies that our algorithm is able to better infer the unknown value. However, one can also parameterize the diffusivity in an appropriate way to learn its parametric form, as discussed before.

8.3.2 Exact Flow Maps

Fig. (8-27) and Fig. (8-28) show the exact forward and backward flow maps respectively between 20 March 2019 and 23 March 2019 (72 hour time window), and 20 March 2019 and 24 March 2019 (96 hour time window), all at 00Z. These flow maps are computed with the exact velocity fields and using the method of flow map compo-

sition for open domains (Sec. 6.1), with a timestep of 24 minutes and WENO5 scheme for spatial gradients and TVD-RK3 time marching. We use these results to compute the relative errors in the learned flow maps from the three cases considered. These fields are never seen by the ML algorithm and are only used for comparison.



(a) X flow map: 23 March 2019

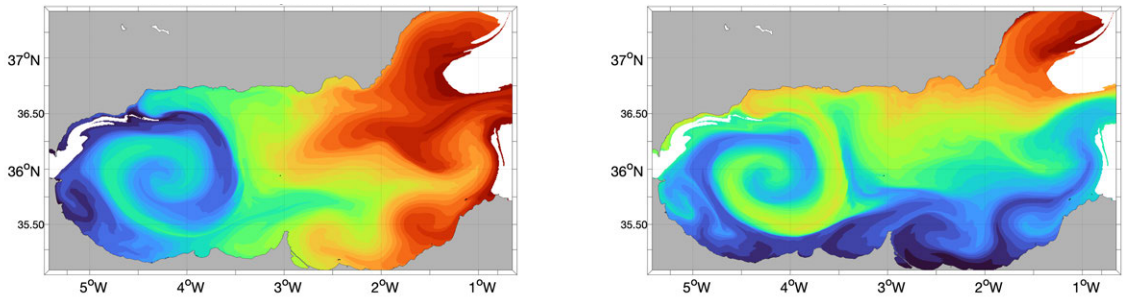
(b) Y flow map: 23 March 2019

(c) X flow map: 24 March 2019

(d) Y flow map: 24 March 2019

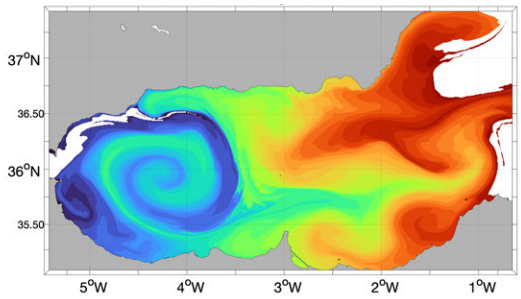
Figure 8-27: Exact forward X and Y flow maps between 20 March 2019 and 23 March 2019 (panels (a), (b)) and between 20 March 2019 and 24 March 2019 (panels (c), (d)).

One can clearly see the Lagrangian signature of the Western Alboran Gyre in the western region of the domain. Further, the backward flow maps are undefined near the Strait of Gibraltar as waters from the Atlantic Ocean (outside the considered domain) enter the Alboran Sea in this area. The forward and backward flow maps are undefined in the eastern areas of the domain due to the inflow and the outflow from the central Mediterranean Sea. We further notice that the active domains reduce in size as the time window increases for both the flow maps.

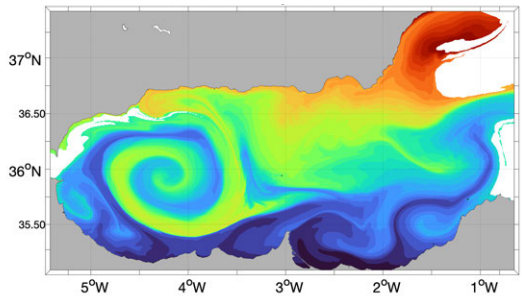


(a) X flow map: 23 March 2019

(b) Y flow map: 23 March 2019



(c) X flow map: 24 March 2019

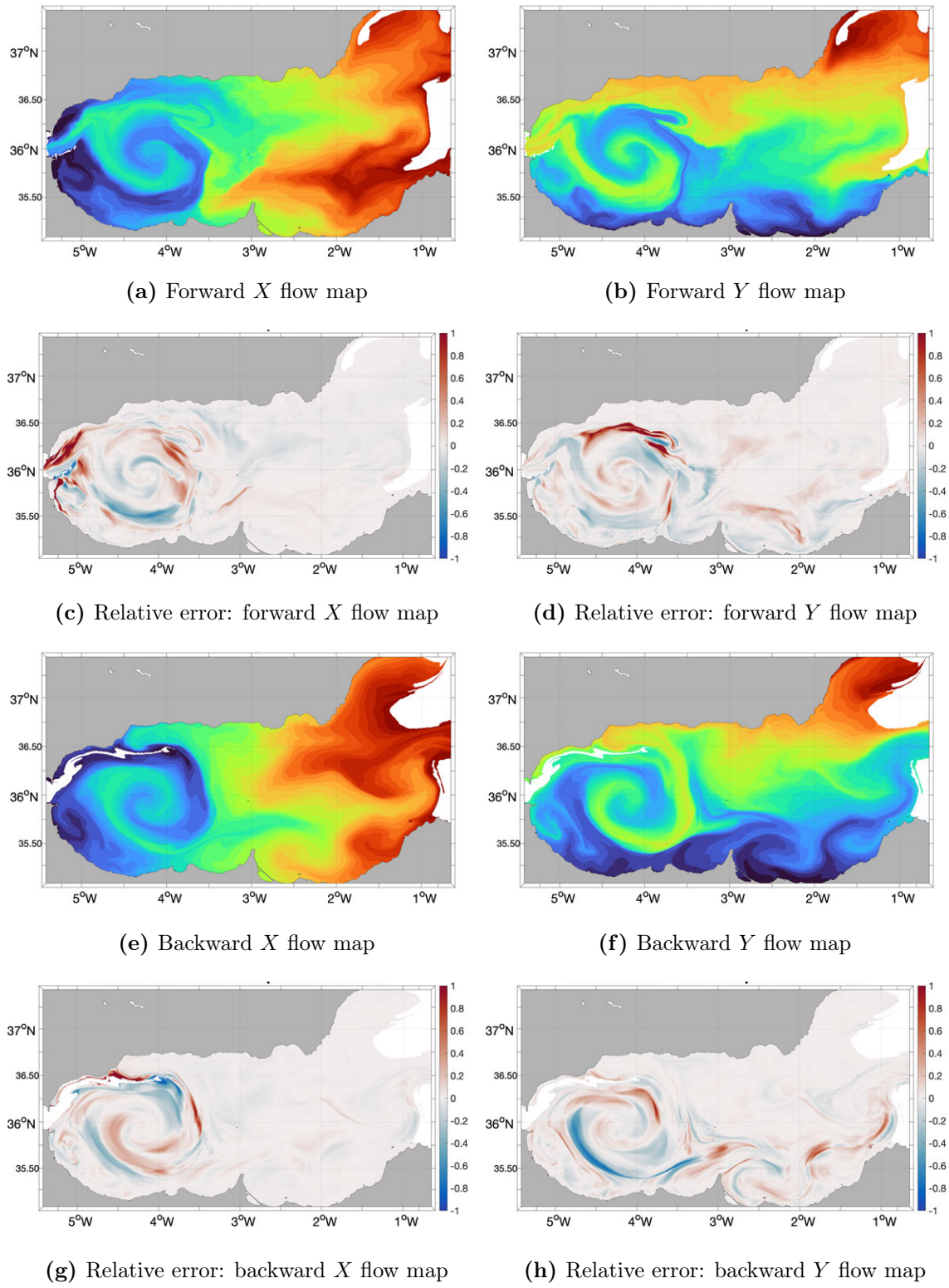


(d) Y flow map: 24 March 2019

Figure 8-28: Exact backward X and Y flow maps between 20 March 2019 and 23 March 2019 (panels (a), (b)) and between 20 March 2019 and 24 March 2019 (panels (c), (d)).

8.3.3 Learning from Simulated Advective Tracer

Fig. (8-29) shows the 72 hour inferred flow maps learned only using the hourly simulated advective tracer transport fields. It can be clearly seen that the flow maps learned by our ML algorithm are close to the exact fields (Fig. (8-27) and Fig. (8-28)) and the relative errors are low. Our algorithm infers the Western Alboran Gyre boundaries to be slightly off from the exact situation and thus there exist small regions with high errors around the periphery of the gyre. However, often times in real scenarios, errors from the data resolution and its inherent stochasticity may dominate these localized errors. Further, one can clearly see that our active domain extraction algorithm is able to correctly decipher the active domains for the forward and the backward flow maps, and is able to capture the inflow of the Atlantic waters through the Strait of Gibraltar (as seen in the backward flow maps) and the inflow and outflow of water masses to the central Mediterranean Sea (as seen in both forward and backward flow maps).



(a) Forward X flow map

(b) Forward Y flow map

(c) Relative error: forward X flow map

(d) Relative error: forward Y flow map

(e) Backward X flow map

(f) Backward Y flow map

(g) Relative error: backward X flow map

(h) Relative error: backward Y flow map

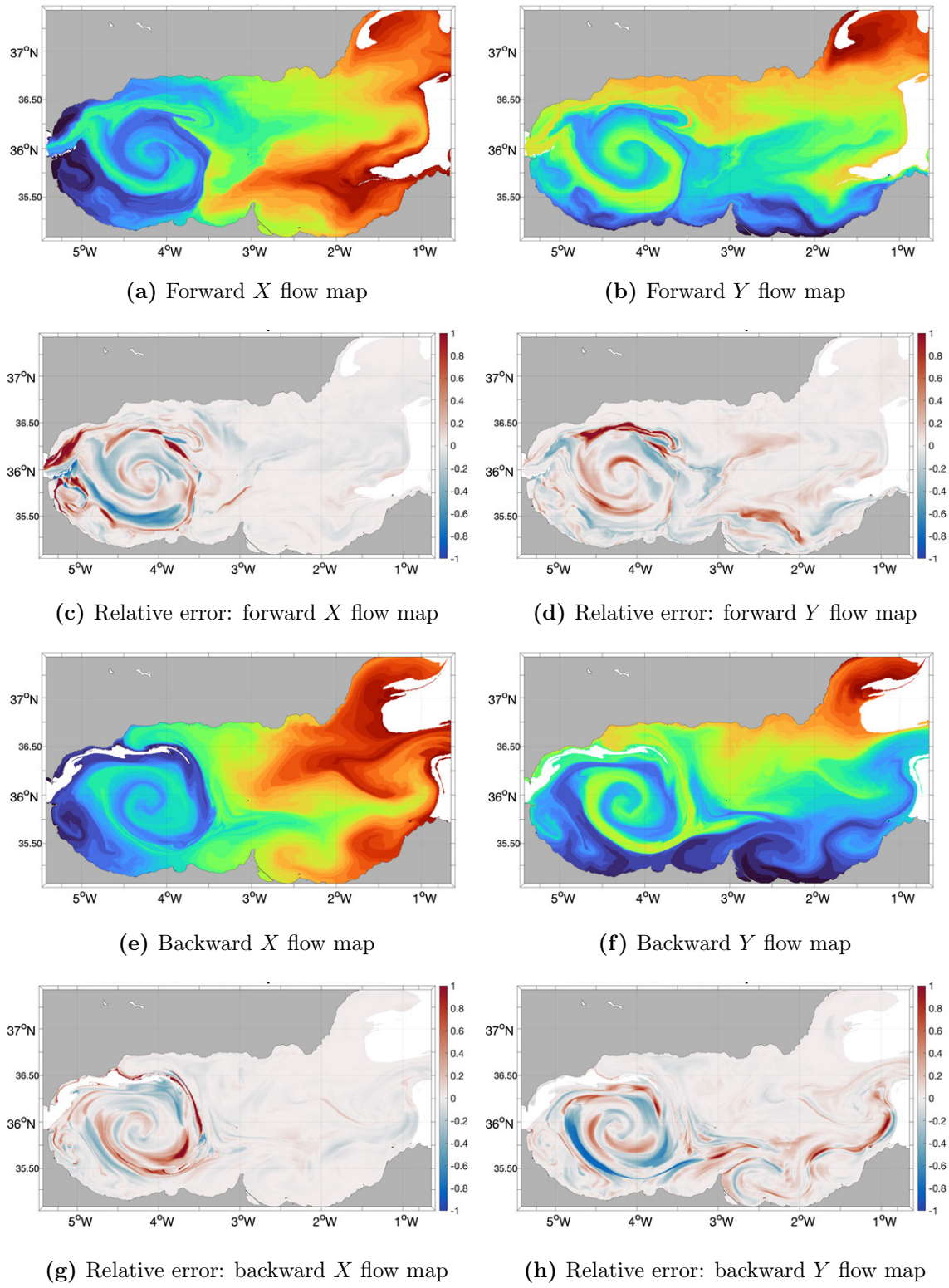
Figure 8-29: Forward and backward flow maps between 20 March 2019, 00Z and 23 March 2019, 00Z inferred using simulated advective tracer transport data. These can be compared to their exact counterparts in panels (a) and (b) of Fig. (8-27) and Fig. (8-28).

Fig. (8-30) shows the flow maps predicted by our ML algorithm for the 96 hour time window between 20 March 2019, 00Z and 24 March 2019, 00Z, only using the advective tracer transport data for the first 72 hours. We can clearly see that the predicted flow maps capture the expected clockwise rotation of the gyre and also the intrusion of the Atlantic waters through the Strait of Gibraltar. We also see that the relative errors are higher, especially around the Western Alboran Gyre. This is expected, as this is the most dynamically active area in the considered domain with larger sensitivity to small prediction errors. Finally, the predicted active domains for the forward and the backward flow maps are similar to their exact counterparts.

8.3.4 Learning from Sea Surface Temperature

Fig. (8-31) shows the forward and backward flow maps and their corresponding relative errors between 20 March 2019 and 20 March 2019 (all 00Z). These are inferred using the data about SST through the extension of our algorithm that accounts for advective–diffusive tracers. As both the forward and the backward flow maps are simultaneously inferred (required to determine the active domains), we enforce Eq. (7.33) to be the advection–diffusion loss component to emphasize stronger coupling between advection and diffusion in the operator splitting scheme. The inferred value of the diffusivity κ is 119.78 m^2/s which is close to the the mean diffusivity of 103.58 m^2/s. The errors in the learned diffusivity and flow maps arise due to a variety of factors: (i) we assume that the diffusivity is a constant, which is not the case in the MSEAS simulation as mentioned before, (ii) we do not account for the vertical transport, mixing, diffusion, and atmospheric forcing, and (iii) numerical errors and approximate machine inference, as the problem is inherently ill-conditioned. The relative errors are higher around the gyre as well as around Gibraltar from where the Atlantic waters enter the Alboran Sea. However, our active domain extraction algorithm captures the actual domain well.

Fig. (8-32) shows the predicted flow maps over a 96 hour time window, from 20 March 2019, 00Z to 24 March 2019, 00Z, using SST data only for the first 72 hours. It can be clearly seen that the predictions get worse with increasing time, as there



(a) Forward X flow map

(b) Forward Y flow map

(c) Relative error: forward X flow map

(d) Relative error: forward Y flow map

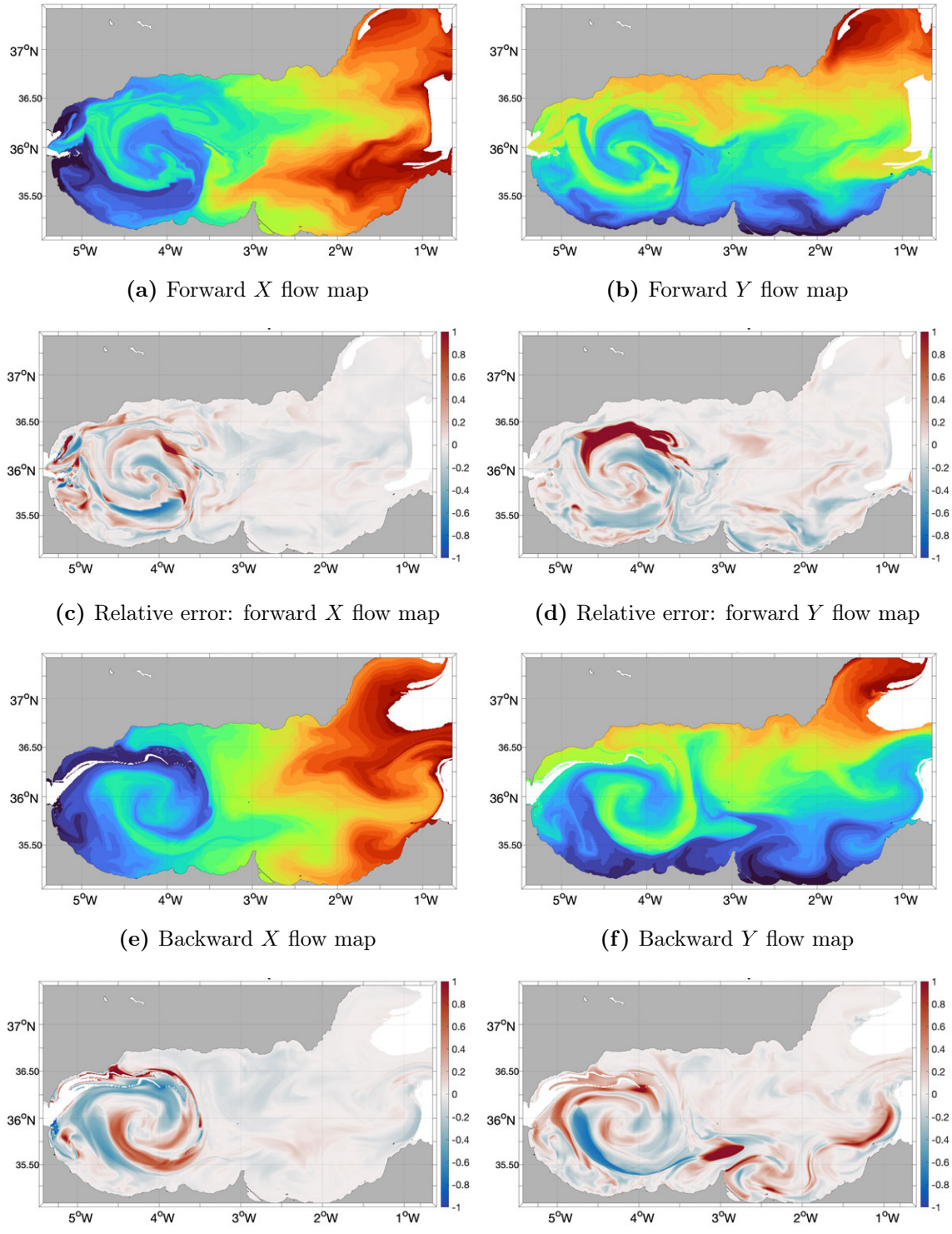
(e) Backward X flow map

(f) Backward Y flow map

(g) Relative error: backward X flow map

(h) Relative error: backward Y flow map

Figure 8-30: Forward and backward flow maps between 20 March 2019, 00Z and 24 March 2019, 00Z predicted using simulated advective tracer transport data. These can be compared to their exact counterparts in panels (c) and (d) of Fig. (8-27) and Fig. (8-28).



(a) Forward X flow map

(b) Forward Y flow map

(c) Relative error: forward X flow map

(d) Relative error: forward Y flow map

(e) Backward X flow map

(f) Backward Y flow map

(g) Relative error: backward X flow map

(h) Relative error: backward Y flow map

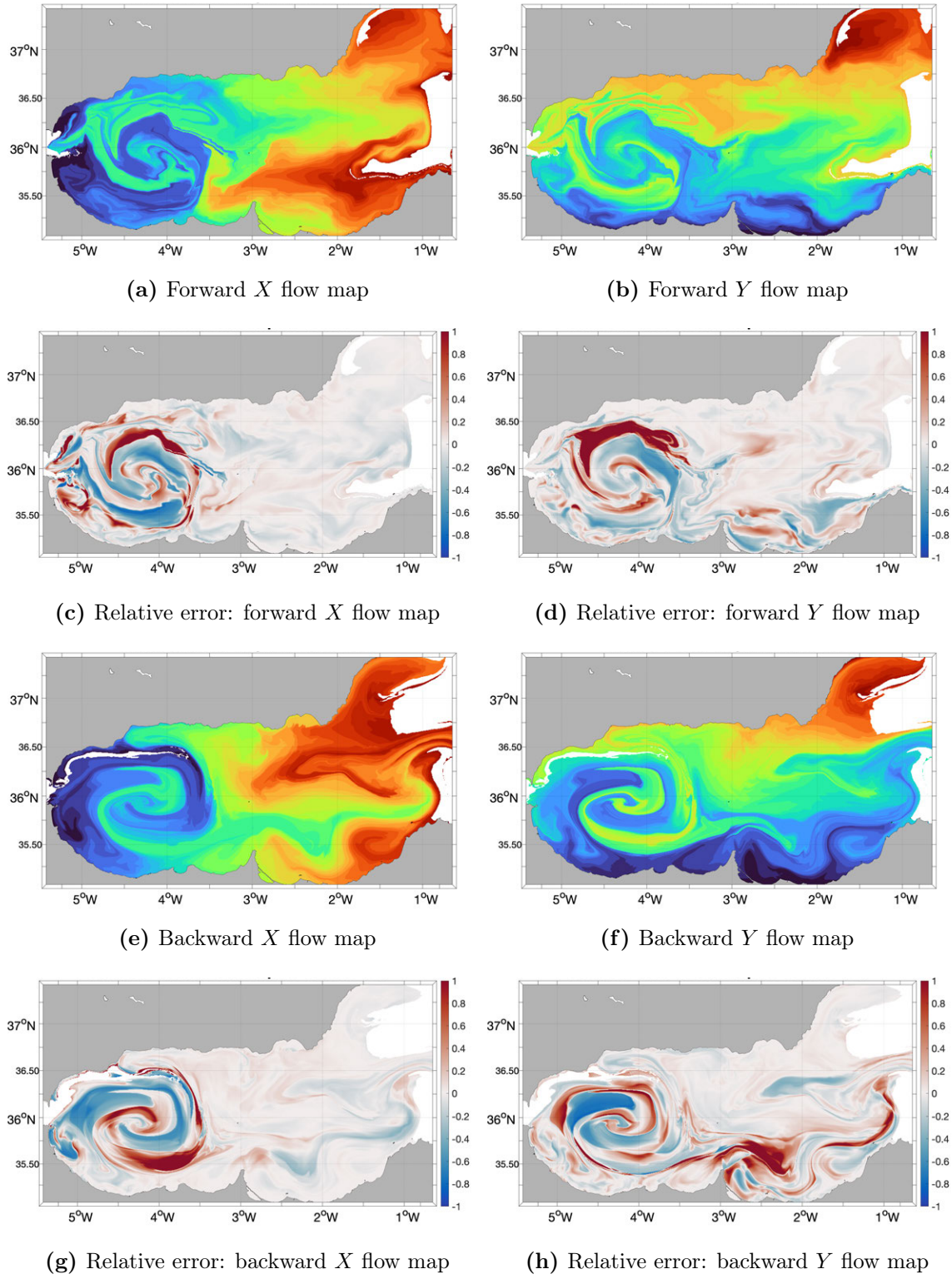
Figure 8-31: Forward and backward flow maps between 20 March 2019, 00Z and 23 March 2019, 00Z inferred only using the SST data. These can be compared to their exact counterparts in panels (a) and (b) of Fig. (8-27) and Fig. (8-28).

is no transport data available during this period. However, even with this limited amount of data, our algorithm forecasts the flow maps and the active domains well. Specifically, it correctly predicts the further entrainement of the Atlantic waters at Gibraltar and the clockwise rotation of the Western Alboran Gyre. The entering and escaping waters in the northeastern region of the domain are also correctly captured.

8.3.5 Learning from Sea Surface Temperature and Salinity

Finally, we look at the machine inference and prediction results obtained by using data about SST and SSS together. The ML algorithm is trained by incorporating the tracer advection–diffusion loss for both the temperature and salinity fields as detailed in Sec. 7.5.1, assuming constant and equal diffusivity for both. One can observe considerable improvement over the results obtained only by using the SST data, as now our algorithm has access to two independent tracer transport datasets, allowing it to infer the underlying common advective transport features better. The learned diffusivity value by the algorithm is 115.21 m^2/s, which is an improvement over the prior prediction of 119.78 m^2/s. However, a larger improvement is seen in the learned flow map fields. Fig. (8-33) shows the inferred forward and backward flow maps between 20 March 2019 and 23 March 2019. As with the prior two cases, it can be seen that the ML algorithm infers both the forward and the backward flow maps well, along with their corresponding active domains. Errors are committed around the Western Alboran Gyre as well as around the influx of the Atlantic waters. However, the inferred fields are accurate and are able to capture the small-scale features of the flow maps in the central Alboran Sea. When compared to Fig. (8-31), we see that the shape of the Western Alboran gyre is inferred more accurately when using both SST and SSS data.

Fig. (8-34) shows the forward and backward flow maps between 20 March 2019 and 24 March 2019 predicted using SST and SSS data only until 23 March 2019 (all 00Z). Similar to the previous two cases, we obtain appreciably close fields for both flow maps. However, the errors around the Western Alboran Gyre are larger



(a) Forward X flow map

(b) Forward Y flow map

(c) Relative error: forward X flow map

(d) Relative error: forward Y flow map

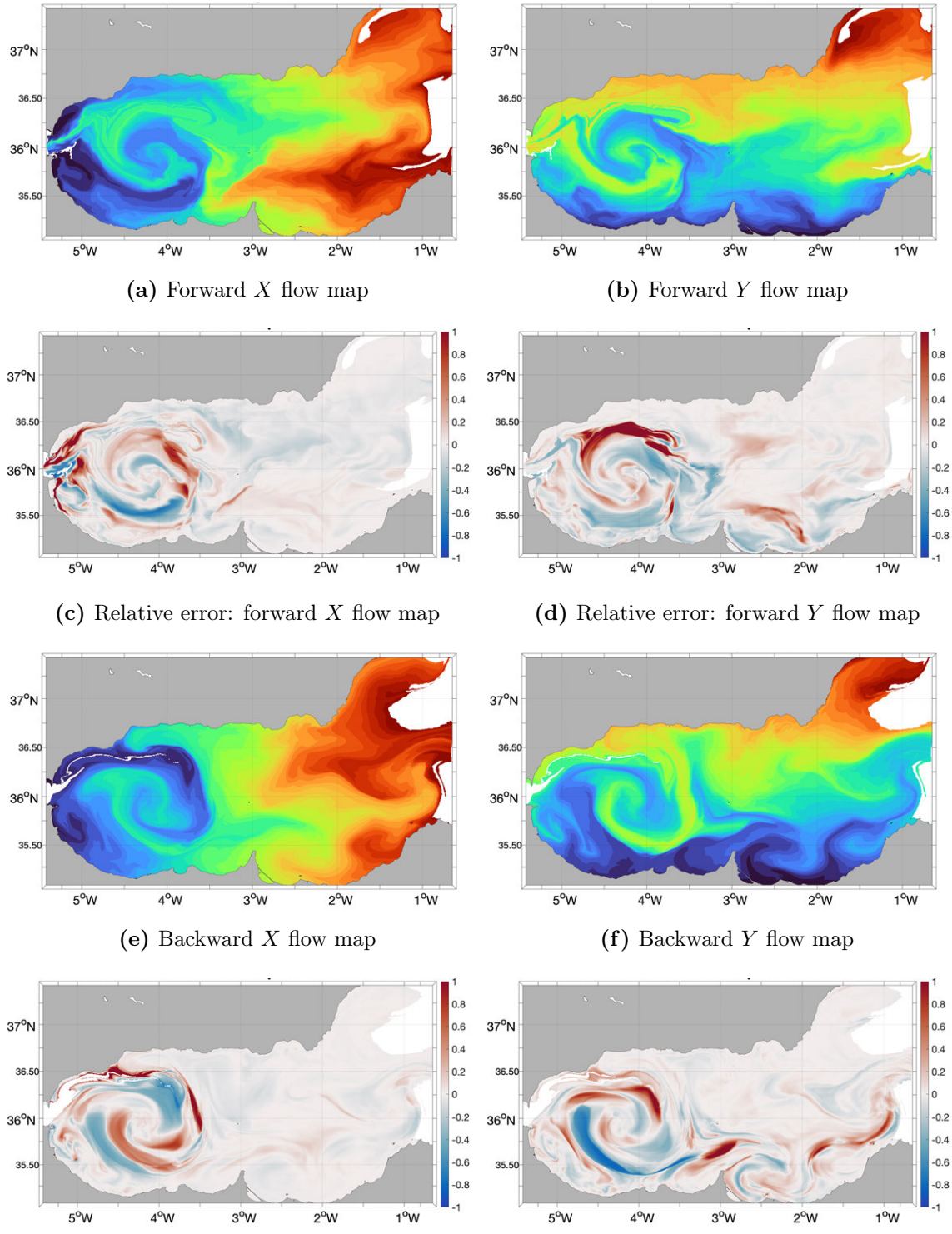
(e) Backward X flow map

(f) Backward Y flow map

(g) Relative error: backward X flow map

(h) Relative error: backward Y flow map

Figure 8-32: Forward and backward flow maps between 20 March 2019, 00Z and 24 March 2019, 00Z predicted only using the SST data. These can be compared to their exact counterparts in panels (c) and (d) of Fig. (8-27) and Fig. (8-28).



(a) Forward X flow map

(b) Forward Y flow map

(c) Relative error: forward X flow map

(d) Relative error: forward Y flow map

(e) Backward X flow map

(f) Backward Y flow map

(g) Relative error: backward X flow map

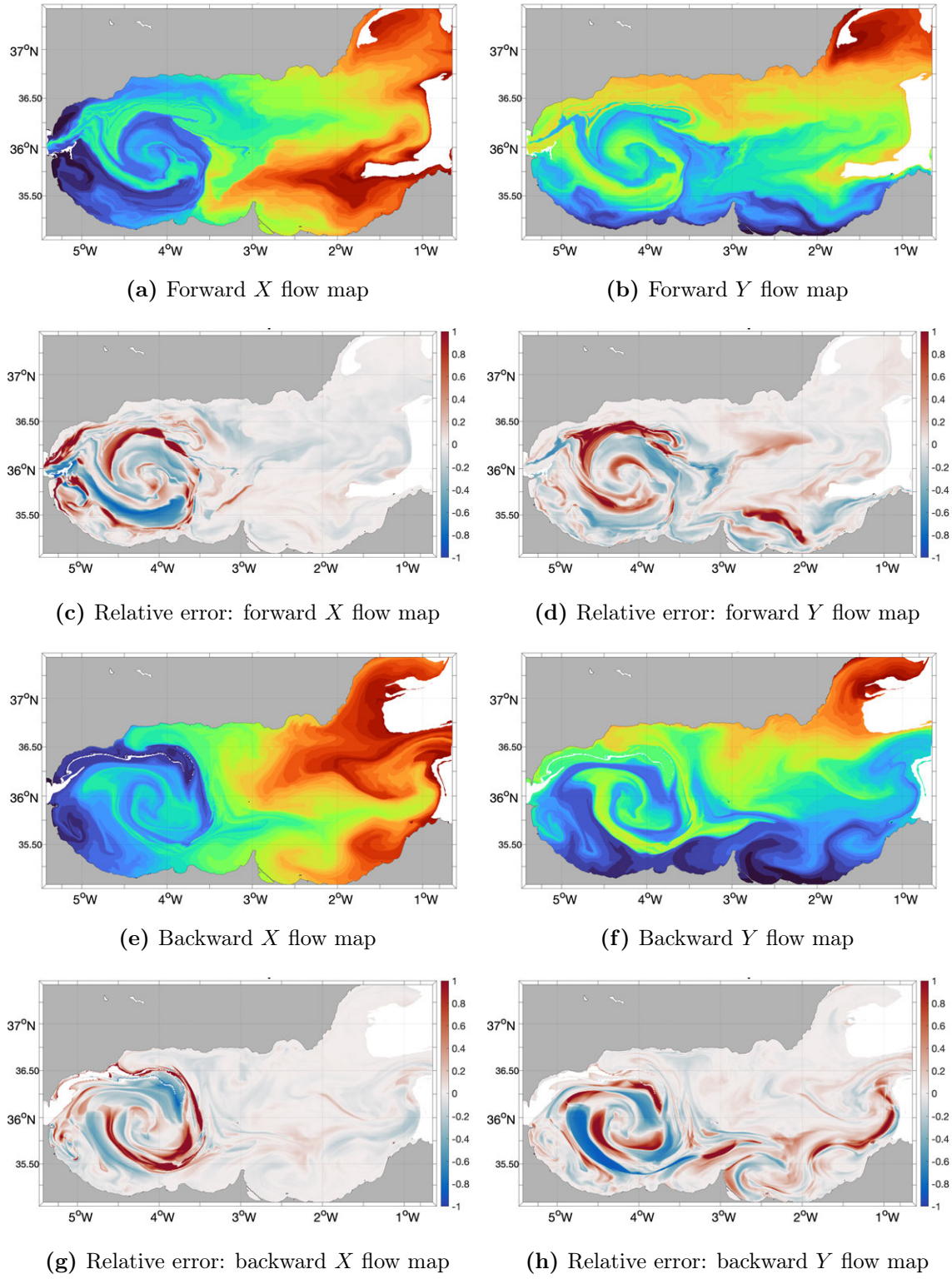
(h) Relative error: backward Y flow map

Figure 8-33: Forward and backward flow maps between 20 March 2019, 00Z and 23 March 2019, 00Z inferred using SST and SSS data. These can be compared to their exact counterparts in panels (a), (b) of Fig. (8-27) and Fig. (8-28).

when compared to the 3 day inferred flow maps due to the lack of data beyond 23 March 2019. Like before, our algorithm correctly captures the clockwise rotation of the gyre and the shrinkage of the active domains due to the influx and egress of the water masses. When compared to Fig. (8-32), the quality of the flow map predictions improves when additional data about the SSS is used along with the SST data.

8.3.6 Comparison of Coherent Structures

The first row of Fig. (8-35) shows the exact forward and backward FTLE field highlighting the repelling and the attracting coherent structures over 4 days in the considered domain. One can clearly see the Lagrangian boundary of the Western Alboran gyre as well as several other repelling and attracting ridges scattered throughout the domain. We compare the FTLE fields predicted by the considered three simulations with these exact fields to comment on the capability of our algorithm to capture the prominent LCSs in the domain. The second, third, and fourth rows of Fig. (8-35) show the forward and backward FTLEs learned using data about (i) simulated advective tracer (second row), (ii) SST only (third row), and (iv) SST and SSS together (fourth row). It can be clearly seen that the FTLEs resulting from the first case are most similar to the exact FTLEs, and they capture most of the prominent attracting and repelling LCSs. The FTLEs learned only from SST show less similarity with the corresponding exact fields, but they approximately capture the signature of the gyre. As stated before, this is mainly due to the fact that our algorithm does not account for some components governing the transport of the SST (such as vertical transport, atmospheric forcing, variable diffusivity etc.). Finally, the FTLEs learned when using data about both SST and SSS more accurate than when only using SST as expected, but fall short of the results from when using the simulated advective tracer. This can be alluded to the fact that now both SST and SSS independently satisfy the advection–diffusion equation (with the same diffusivity), and hence our algorithm is better able to learn the common advective transport features by using this data simultaneously. However, as it does not have access to any purely advective transport data, the results fall short of those obtained when using simulated advective



(a) Forward X flow map

(b) Forward Y flow map

(c) Relative error: forward X flow map

(d) Relative error: forward Y flow map

(e) Backward X flow map

(f) Backward Y flow map

(g) Relative error: backward X flow map

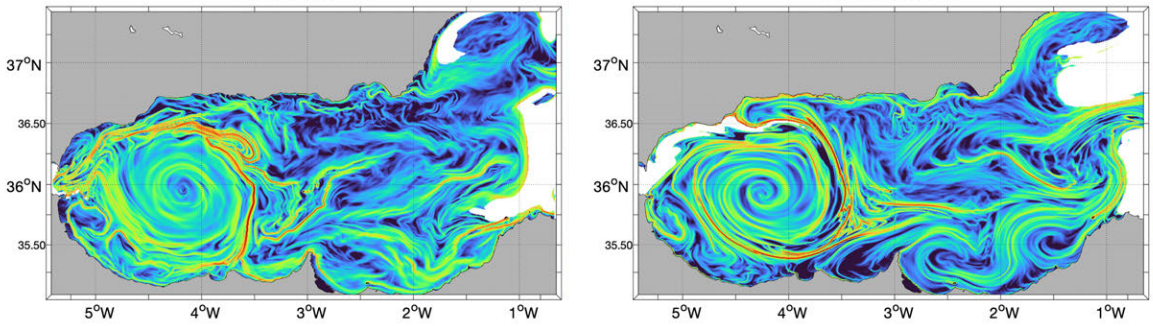
(h) Relative error: backward Y flow map

Figure 8-34: Forward and backward flow maps between 20 March 2019, 00Z and 24 March 2019, 00Z predicted using SST and SSS data. These can be compared to their exact counterparts in panels (c), (d) of Fig. (8-27) and Fig. (8-28).

tracer data.

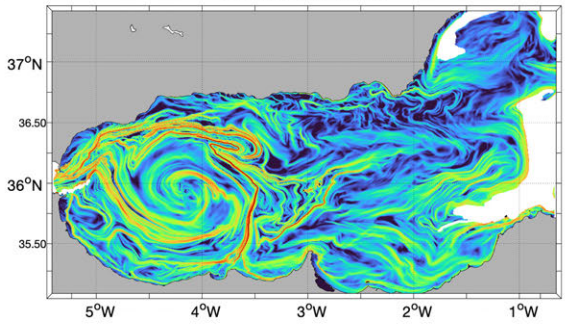
8.4 Summary

In this chapter, we showcase the applications of the theory and schemes developed in Chapter 7 to infer and predict the flow maps only from tracer transport data. We highlight the various aspects of the developed theory and schemes in two analytical flow fields and one realistic flow field. We first look at the analytical swirl flow, wherein we show that the knowledge about the underlying flow field (e.g. knowing that it is steady) greatly enhances our learning capabilities. We then focus on the analytical double gyre, wherein we showcase our ability to learn the flow maps, coherent structures, and coherent sets. We also learn these quantities by optimally combining low resolution Eulerian information and high resolution Lagrangian information, and from transport data about an advective–diffusive tracer field. Finally, we look at a realistic ocean flow in the Western Mediterranean (Alboran) Sea, and highlight the applications of our ML framework to learn and predict flow maps and LCSs from data from (i) simulated advective tracer transport, (ii) sea surface temperature, and (iii) sea surface temperature and salinity together. We observe that our algorithm best learns the flow maps and coherent structures from simulated advective tracer transport, followed by from sea surface temperature and salinity. The machine inference and prediction only using sea surface temperature is the least accurate, but our algorithm is still able to capture the dominant Lagrangian features of the flow.

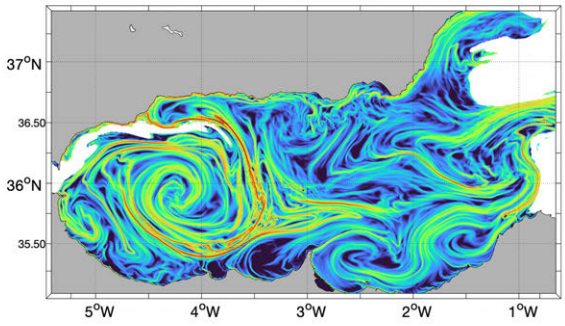


(a) Exact forward FTLE

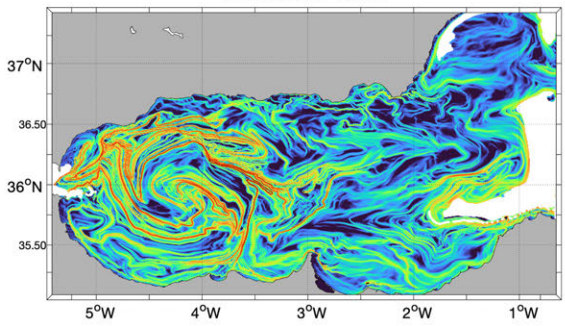
(b) Exact backward FTLE



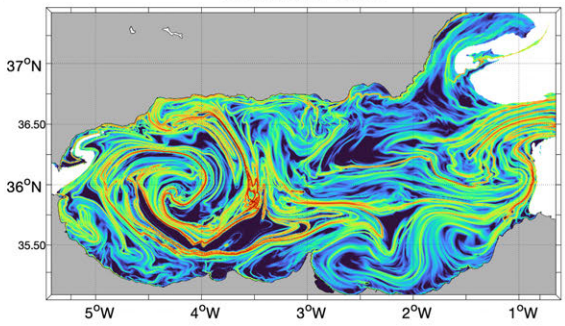
(c) Forward FTLE using advective tracer



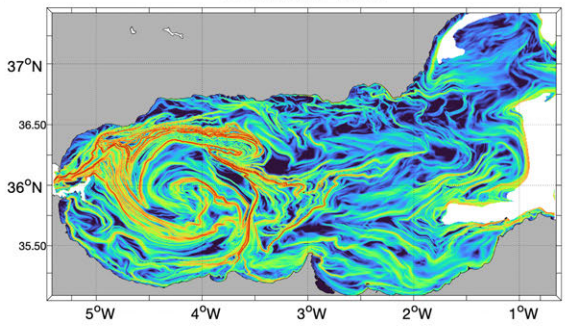
(d) Backward FTLE using advective tracer



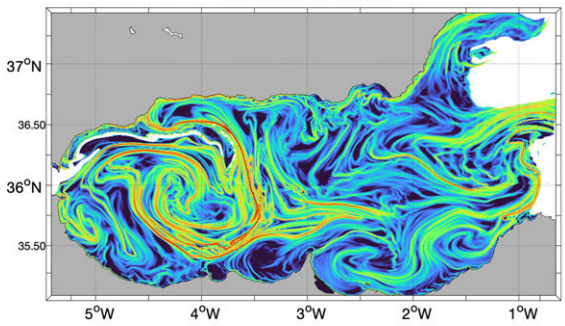
(e) Forward FTLE only using SST



(f) Backward FTLE only using SST



(g) Forward FTLE using SST and SSS



(h) Backward FTLE using SST and SSS

Figure 8-35: Exact forward and backward FTLEs between 20 March 2019, 00Z and 24 March 2019, 00Z and when computed from flow maps learned from simulated advective tracer (second row), SST only (third row), and SST and salinity together (fourth row).

Chapter 9

Conclusions and Future Work

The transport of anthropogenic and natural material in environmental flows is ubiquitous and profoundly impacts society. Preparedness and effective response can save many lives, untold environmental damage, and enormous financial cost. Predicting, quantifying, and inferring the passive transport and the subsequent dispersal of such substances in geophysical environments is imperative to designing effective mitigation, control, and conservation strategies. Using this as the primary motivation, the present thesis develops fundamental theories and rigorous methodologies for predicting, uncovering, and learning advective and advective–diffusive material transport in dynamic fluid flows. Research contributions of this work are: (i) the development of a highly accurate numerical method to solve the advective transport equation, (ii) the formulation of theory and schemes to efficiently determine material sets that remain the most or the least coherent over the considered time to quantify fluid mixing, and (iii) the development of a machine learning framework to infer and predict the underlying generic advective transport features from snapshot data about the transport of any material quantity.

9.1 Advection Through Flow Map Composition

A novel methodology was derived for the numerical computation of advective transport and diffusion-reaction of tracer quantities through flow map composition. The

method of composition for advection is 'super-accurate', yielding numerical solutions almost devoid of compounding numerical errors with an accuracy as that of particle-based methods but with the advantage of Eulerian resolution in space. It is also readily parallelizable in the temporal direction and can utilize existing computational frameworks. Instead of advecting a tracer field, we advect the spatial positions in the domain independently over smaller intervals and then compose (map) them with the corresponding initial (or diffused and forced) tracer value. There are several benefits to the new methodology: (i) as advection computations over each time interval are independent, the numerical errors are not compounded, which results in much higher accuracy and much lower computational expense for comparable accuracy, (ii) these independent advection computations are parallelizable, (iii) only a single advection computation is required for multiple tracers, and (iv) any PDE toolbox can be used to compute the individual position advections, hence minimal programming is required. Theoretically, we derive rigorous expressions for numerical errors and for the 'optimal composition timestep', *i.e.* the timestep value that results in the minimum total numerical error. This is achieved by balancing the advection error and the composition error with each other. We also develop schemes for the addition of tracer diffusion, reaction, and source terms, and for the implementation of boundary conditions.

The new methodology and its capabilities are thoroughly illustrated through a wide set of examples, including new benchmark problems for advection–diffusion-reaction schemes. The forward-backward advection and advection–diffusion in an analytical reversible swirl flow verify the behavior of errors, illustrate the effectiveness of the optimal composition timestep, and show that the new methodology eliminates the compounding of numerical errors while being faster than regular methods. The flow undergoing sudden expansion is then used to highlight the applicability of the method of composition with tracer sources and multiple types of tracer boundary conditions. By comparison with highly-resolved tracer fields, we confirm that the novel methodology is more accurate and efficient than the regular advection schemes. Finally, the applications of our method of composition in realistic geophysical flows is demonstrated through two studies. The advection of sediment plumes by ocean

currents resulting from deep sea mining operations in the Bismarck Sea is simulated for two possible mining sites. We find that, even for low-order schemes, advection using the method of composition yields minimal total numerical errors, much smaller than the errors of high-order Eulerian schemes and with a spatial coverage much better than particle-based methods. The second application highlights the use of the method of composition to support and analyze the predictions and observations during the 2018 PLUMEX real-time sea exercise in the Southern California Bight. We compare the real-time 3D predictions of sediment plume transport made by the method of composition with the experimentally observed data to find a satisfactory match between the predictions and the observations.

Portability and the ease of implementation of our method of flow map composition make it attractive for applications in various domains. The methodology can be extended with minimal effort to the advection of stochastic tracers: even if the uncertainties in the tracer and/or velocity fields are large, an accurate numerical advection is still needed for accurate probabilistic prediction, either in a Monte-Carlo sense or using dynamic reduced-order models [145; 65; 109]. For the latter, it would also decrease the computational cost dependence in the number of stochastic modes from quadratic to linear, which is a substantial gain. Our method of composition is also related to the reinitialization ideas from level set methods [190; 1], and it can be used effectively in optimal path planning computations [162; 238; 239; 148; 149; 138; 135]. Composition-based advection would also be effective in iterative pressure correction methods for Navier–Stokes' equations [8]. Finally, other broad disciplines that rely on accurate simulations of material transport such as atmospheric emissions and plume dispersal [82; 83], marine pollution [118; 146; 142; 228; 22], and biogeochemical prediction [174; 195; 90; 97] could substantially benefit from our composition-based advection.

9.2 Lagrangian Analysis of Material Transport

The second part of this thesis extends the method of composition to compute flow maps in realistic ocean flow fields and develops efficient theoretical and computational techniques to extract incoherent, coherent, and persistently coherent material subregions in dynamic flow fields. We first extend the method of flow map composition to compute flow maps in open domains where the fluid (and the material it carries) is free to enter of leave the domain, which is often the case in realistic ocean flows. This is achieved by defining and computing a new 'mask' field, which we prove is governed by the advective transport equation with well-defined initial and boundary conditions. Thus we use composition-based advection to compute the mask field as a function of time, and then extract the active domains of the flow maps using it. We show that this PDE-based method yields solutions with comparable accuracy to trajectory-based methods, and eliminates spurious numerical artifacts.

Second, a novel approach to determine the material sets that remain coherent throughout the entire time duration, *i.e.* 'persistently coherent sets' is proposed. The metric, called the 'extended polar distance' quantifies the deviation of a material set transformation from being a rigid body motion at all times. The level sets of the extended polar distance field yield the persistently rigid sets up to a certain tolerance value, where this tolerance is connected to the expected amount of stretching that these material sets undergo. Further, as this metric highlights material subdomains and not co-dimension one surfaces, it can be used to also detect lack of coherence (*i.e.* incoherence or non-rigidity) simply by considering locations with high extended polar distance values. We prove that the proposed metrics are frame-independent (objective) and yield equivalent results irrespective of whether the forward or the backward flow map is used for their computation. Such persistently coherent sets are of paramount importance in quantifying fluid mixing, as we prove that these sets are the ones to maximally resist both advective stretching and local diffusion.

In order to benchmark the developed theory and schemes and to demonstrate their applicability in analytical and realistic flow fields, we consider two canonical

flows, namely the analytical double gyre flow and a realistic ocean flow around the island of Palau in the Southern Pacific Ocean. We first benchmark our flow map computation scheme for open domains, and show that through the use of the mask field and the method of composition, we can accurately compute the flow maps in open domains with minimal numerical errors and while maintaining theoretical guarantees and uniform spatial coverage. We also show the various examples of the real-time and hindcast studies conducted in various marine regions in the world in which our composition-based advection and Lagrangian analysis systems were utilized. We then move to analyzing the proposed coherence and persistent coherence metrics. We first demonstrate that the coherence measures computed using the forward and the backward flow map are equivalent to each other. We then study the evolution of the rigid and persistently rigid material sets as well as some of the most non-rigid sets in the analytical double gyre flow. The second example looks at the behavior of an incoherent set, a coherent set, and a persistently coherent set in the realistic ocean flow around the island of Palau. The incoherent set continually gets distorted due to the turbulent fluid motion westward of the island. The coherent set initially gets distorted but later undergoes compressive strain to end up with a final shape similar to its initial shape. The persistently coherent set does not undergo any appreciable strain at any time during the considered interval and approximately maintains its shape throughout. We show that even though the coherent and the persistently coherent sets are initially located close to each other, their dynamics during the considered time interval are very different. This emphasizes the need to use accurate Lagrangian analyses in predicting and quantifying material transport in dynamic flows.

The present analysis can be readily applied to study the transport features, attraction zones, and mixing zones for passive tracer materials such as plastics, debris, oil, and plankton [187; 143; 198; 131]. Further, given the specific transport dynamics of a particular material, say through an ODE for its motion, the presented analysis can be extended to determine the corresponding persistently rigid sets and non-rigid sets. Beyond the domain of geophysical flows, Lagrangian analyses have been used in a variety of different disciplines, from studying carotid artery bifurcations [253]

and cardiovascular systems [223] to understanding the foraging patterns of marine predators such as tuna, seabirds etc. [125]. We believe that the present work would be useful to several studies in these disciplines.

9.3 Inference and Prediction of Material Transport Features

The final part of this thesis deals with inferring and predicting the generic advective transport features and coherent structures only by using Eulerian snapshot data about the transport of advective and advective–diffusive tracers. Conventional data assimilation techniques tend to be ineffective for this task due to the lack of evolution equations and a complex nonlinear relationship between the observations and the variables of interest. This, along with the fact that the compositionality of flow maps closely resembles the functional form of neural networks lead us towards scientific machine learning (ML) as a viable solution approach.

Central to our inference and prediction algorithm is a long-short term memory network (LSTM) designed based on our knowledge and insights of the physical system at hand. Specifically, we utilize a convolutional LSTM (convLSTM) to account for strong local spatial dependence of the flow maps, and employ a hypernetwork to account for relaxed weight sharing in time. Further, we incorporate the known constraints on the flow map in the loss function to ensure physically consistent results. Once trained using the given Eulerian tracer transport data, the neural network can infer the flow maps over the time interval of the given tracer transport data and can also predict the flow maps at future times. To quantify the quality of the learned flow maps, we analyze the predictive capability of the neural network and the information contained in the Eulerian tracer field. We further develop extensions of our algorithm to handle data from multiple advective tracers, optimally combine Eulerian and Lagrangian information, and infer and predict flow maps in open domains.

Three flow fields are considered to highlight different facets of our ML algorithm.

First, we demonstrate the effect of strict and relaxed weight sharing to infer the flow maps in an analytical swirl flow and conclude that strict weight sharing performs better in steady flow fields whereas relaxed weight sharing performs better in unsteady flows. We then look at tracer transport in the analytical double gyre wherein we infer the unsteady flow maps, coherent structures, and coherent sets. We also highlight the ability to optimally combine low resolution Eulerian and high resolution Lagrangian information, and demonstrate the capacity to learn the flow maps from the observations of an advective–diffusive tracer. Finally, we demonstrate and comment on the information content of the tracer field and the predictive capability of our algorithm, *i.e.* the accuracy of future flow map predictions in this flow field. Finally, we consider a realistic flow in the Alboran Sea, wherein we infer and predict the flow maps and coherent structures using the observations of (i) a simulated advective tracer, (ii) sea surface temperature (SST), and (iii) SST along with sea surface salinity (SSS). We discover that our algorithm performs the best when using a simulated advective tracer, followed by when using SST and SSS together, followed by when only using the SST data. This is because while the simulated tracer is exactly advective, both the SST and SSS are also diffusive, thus hampering the learning capability. However, when using information about both these fields, our algorithm is better able to learn the underlying common advective transport features and the tracer diffusivities than when only using data about SST. Further, we show that, our algorithm can accurately predict future flow maps for more than 1 day beyond the availability of the observed data. However, the errors increase beyond that and the predictions become unreliable.

Our work towards inferring the transport features from observational data is only a first step at solving this ill-conditioned inverse problem. There are several extensions to this work that would enable it be be employed in a wide gamut of real-time settings. Often times, the transported tracer data is noisy, sparse, and not available at uniform time intervals or over fixed spatial regions. Our current approach can be extended to account for such inconsistencies in the tracer transport data by appropriately modifying the loss function. Our algorithm may further be utilized and extended to

not only infer the flow maps and coherent structures but also infer regions of specific interest, such as regions with the highest tracer mixing, areas of subduction etc. One can utilize our approach to develop methods that can quantify the uncertainty associated with the flow map predictions. The recent developments in new neural network architectures such as neural ODEs [31; 244] may be well suited to extend our proposed ML algorithm to infer and predict material transport features as a continuous function of time. Finally, in this work, we assume that the considered tracers are passive and non-inertial. However, the present work can be extended to account for dynamically active and inertial tracers through state augmentation.

Bibliography

[1] David Adalsteinsson and James A. Sethian. A fast level set method for propagating interfaces. *Journal of Computational Physics*, 118(2):269–277, 1995.

[2] Nikolas O Aksamit, Themistoklis P Sapsis, and George Haller. Machine-learning ocean dynamics from lagrangian drifter trajectories. *arXiv preprint arXiv:1909.12895*, 2019.

[3] Wael Hajj Ali, Mohamad H. Mirhi, Abhinav Gupta, Chinmay S. Kulkarni, Corbin Foucart, Manan M. Doshi, Deepak N. Subramani, Chris Mirabito, Patrick J. Haley, Jr., and Pierre F. J. Lermusiaux. Seavizkit: Interactive maps for ocean visualization. In *OCEANS 2019 MTS/IEEE SEATTLE*, pages 1–10, Seattle, October 2019. IEEE. doi: 10.23919/OCEANS40490.2019.8962794.

[4] Michael R Allshouse and Thomas Peacock. Lagrangian based methods for coherent structure detection. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 25(9):097617, 2015.

[5] Michael R Allshouse and Jean-Luc Thiffeault. Detecting coherent structures using braids. *Physica D: Nonlinear Phenomena*, 241(2):95–105, 2012.

[6] Luigi Ambrosio. Transport equation and cauchy problem for non-smooth vector fields. *LECTURE NOTES IN MATHEMATICS-SPRINGER-VERLAG-*, 1927: 1, 2008.

[7] ANSYS Inc. ANSYS Fluent theory guide, 2011.

[8] Jean Aoussou, Jing Lin, and Pierre F. J. Lermusiaux. Iterated pressure-correction projection methods for the unsteady incompressible Navier–Stokes equations. *Journal of Computational Physics*, 373:940–974, November 2018. doi: 10.1016/j.jcp.2018.06.062.

[9] Hassan Aref, John R Blake, Marko Budisić, Silvana SS Cardoso, Julyan HE Cartwright, Herman JH Clercx, Kamal El Omari, Ulrike Feudel, Ramin Golestanian, Emmanuelle Gouillart, et al. Frontiers of chaotic advection. *Reviews of Modern Physics*, 89(2):025007, 2017.

[10] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[11] Sanjeeva Balasuriya, Nicholas T Ouellette, and Irina I Rypina. Generalized lagrangian coherent structures. *Physica D: Nonlinear Phenomena*, 372:31–51, 2018.

[12] Antonio E.M. Baptista. *Solution of advection-dominated transport by Eulerian-Lagrangian methods using the backwards method of characteristics*. PhD thesis, Massachusetts Institute of Technology, 1987.

[13] Yohai Bar-Sinai, Stephan Hoyer, Jason Hickey, and Michael P Brenner. Learning data-driven discretizations for partial differential equations. *Proceedings of the National Academy of Sciences*, 116(31):15344–15349, 2019.

[14] Luis Barreira and Ya B Pesin. *Lyapunov exponents and smooth ergodic theory*, volume 23. American Mathematical Soc., 2002.

[15] G K Batchelor. *An introduction to fluid dynamics*. Cambridge university press, 2000.

[16] Atılım Günes Baydin, Barak A Pearlmutter, Alexey Andreyevich Radul, and Jeffrey Mark Siskind. Automatic differentiation in machine learning: a survey. *The Journal of Machine Learning Research*, 18(1):5595–5637, 2017.

[17] Ş. T. Beşiktepe, P. F. J. Lermusiaux, and A. R. Robinson. Coupled physical and biogeochemical data-driven simulations of Massachusetts Bay in late summer: Real-time and post-cruise data assimilation. *Journal of Marine Systems*, 40–41: 171–212, 2003. doi: 10.1016/S0924-7963(03)00018-6.

[18] Christian Beck. Statistics of three-dimensional lagrangian turbulence. *Physical review letters*, 98(6):064502, 2007.

[19] Yoshua Bengio. Rmsprop and equilibrated adaptive learning rates for nonconvex optimization. *corr abs/1502.04390*, 2015.

[20] Andrew Bennett. *Lagrangian fluid dynamics*. Cambridge University Press, 2006.

[21] Francisco J Beron-Vera, Yan Wang, María J Olascoaga, Gustavo J Goni, and George Haller. Objective detection of oceanic eddies and the agulhas leakage. *Journal of Physical Oceanography*, 43(7):1426–1438, 2013.

[22] Manmeet S. Bhabra, Manan Doshi, Benjamin C. Koenig, Patrick J. Haley, Jr., C. Mirabito, Pierre F. J. Lermusiaux, C. A. Goudey, J. Curcio, D. Manganelli, and H. Goudey. Optimal harvesting with autonomous tow vessels for offshore macroalgae farming. In *OCEANS 2020 IEEE/MTS*. IEEE, October 2020. In press.

[23] Christopher M Bishop. *Pattern recognition and machine learning*. springer, 2006.

[24] Steven L Brunton and Clarence W Rowley. Fast computation of finite-time Lyapunov exponent fields for unsteady flows. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 20(1):017503, 2010.

[25] Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the national academy of sciences*, 113(15):3932–3937, 2016.

[26] Steven L Brunton, Bernd R Noack, and Petros Koumoutsakos. Machine learning for fluid mechanics. *Annual Review of Fluid Mechanics*, 52:477–508, 2020.

[27] CE Cala, ECv Fernandes, MV Heitor, and SI Shtork. Coherent structures in unsteady swirling jet flow. *Experiments in Fluids*, 40(2):267–276, 2006.

[28] Michael A. Celia, Thomas F. Russell, Ismael Herrera, and Richard E. Ewing. An Eulerian-Lagrangian localized adjoint method for the advection-diffusion equation. *Advances in Water Resources*, 13(4):187–206, 1990.

[29] L. R. Centurioni et al. Northern Arabian Sea circulation-autonomous research (NASCar): A research initiative based on autonomous sensors. *Oceanography*, 30(2):74–87, June 2017. doi: 10.5670/oceanog.2017.224. Special issue on Autonomous and Lagrangian Platforms and Sensors (ALPS).

[30] Steven C. Chapra and Raymond P. Canale. *Numerical Methods for Engineers*. McGraw Hill Eduction, 2015.

[31] Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. In *Advances in neural information processing systems*, pages 6571–6583, 2018.

[32] Wolfgang Cherdron, Franz Durst, and James H. Whitelaw. Asymmetric flows and instabilities in symmetric ducts with sudden expansions. *Journal of Fluid Mechanics*, 84(1):13–31, 1978.

[33] CM Coats. Coherent structures in combustion. *Progress in Energy and Combustion Science*, 22(5):427–509, 1996.

[34] Bernardo Cockburn. Discontinuous Galerkin methods for computational fluid dynamics. *Encyclopedia of Computational Mechanics Second Edition*, pages 1–63, 2018.

[35] Coffey Natural Systems Pvt. Ltd. Environmental impact statement, Solwara 1 project, vol. A., 2008. URL `http://www.cares.nautilusminerals.com/`.

[36] M. E. G. D. Colin, T. F. Duda, L. A. te Raa, T. van Zon, P. J. Haley, Jr., P. F. J. Lermusiaux, W. G. Leslie, C. Mirabito, F. P. A. Lam, A. E. Newhall, Y.-T. Lin, and J. F. Lynch. Time-evolving acoustic propagation modeling in a complex ocean environment. In *OCEANS - Bergen, 2013 MTS/IEEE*, pages 1–9, 2013. doi: 10.1109/OCEANS-Bergen.2013.6608051.

[37] G. Cossarini, P. F. J. Lermusiaux, and C. Solidoro. Lagoon of Venice ecosystem: Seasonal dynamics and environmental guidance with uncertainty analyses and error subspace data assimilation. *Journal of Geophysical Research: Oceans*, 114 (C6), June 2009. doi: 10.1029/2008JC005080.

[38] J. Coulin, P. J. Haley, Jr., S. Jana, C. S. Kulkarni, P. F. J. Lermusiaux, and T. Peacock. Environmental ocean and plume modeling for deep sea mining in the Bismarck Sea. In *Oceans 2017 - Anchorage*, Anchorage, AK, September 2017.

[39] Richard Courant, Eugene Isaacson, and Mina Rees. On the solution of nonlinear hyperbolic differential equations by finite differences. *Communications on Pure and Applied Mathematics*, 5(3):243–255, 1952.

[40] T M Cover and J A Thomas. *Elements of information theory*. John Wiley & Sons, 2006.

[41] James A. Cummings and Ole Martin Smedstad. Variational data assimilation for the global ocean. In *Data Assimilation for Atmospheric, Oceanic and Hydrologic Applications (Vol. II)*, pages 303–343. Springer, 2013.

[42] Benoit Cushman-Roisin and Jean-Marie Beckers. *Introduction to geophysical fluid dynamics: physical and numerical aspects*. Academic press, 2011.

[43] Ronald J. DiPerna and Pierre-Louis Lions. Ordinary differential equations, transport theory and Sobolev spaces. *Inventiones Mathematicae*, 98(3):511–547, 1989.

[44] Ronald J DiPerna and Pierre-Louis Lions. Ordinary differential equations, transport theory and sobolev spaces. *Inventiones mathematicae*, 98(3):511–547, 1989.

[45] Carl Doersch. Tutorial on variational autoencoders. *arXiv preprint arXiv:1606.05908*, 2016.

[46] John R. Dormand. *Numerical methods for differential equations: A computational approach*, volume 3. CRC Press, 1996.

[47] Manan M. Doshi, Chinmay S. Kulkarni, Wael Hajj Ali, Abhinav Gupta, Pierre F. J. Lermusiaux, Peng Zhan, Ibrahim Hoteit, and Omar Knio. Flow maps and coherent sets for characterizing residence times and connectivity in lagoons and coral reefs: The case of the Red Sea. In *OCEANS 2019 MTS/IEEE SEATTLE*, pages 1–8, Seattle, October 2019. IEEE. doi: 10.23919/OCEANS40490.2019.8962643.

[48] Jim Jr. Douglas and Thomas F. Russell. Numerical methods for convection-dominated diffusion problems based on combining the method of characteristics with finite element or finite difference procedures. *SIAM Journal on Numerical Analysis*, 19(5):871–885, 1982.

[49] Dale R. Durran. *Numerical methods for wave equations in geophysical fluid dynamics*, volume 32. Springer Science & Business Media, 2013.

[50] Franz Durst, A. Melling, and James H. Whitelaw. Low Reynolds number flow over a plane symmetric sudden expansion. *Journal of Fluid Mechanics*, 64(1): 111–128, 1974.

[51] Arkopal Dutt. High order stochastic transport and Lagrangian data assimilation. Master's thesis, Massachusetts Institute of Technology, Department of Mechanical Engineering, Cambridge, Massachusetts, February 2018.

[52] Eric A DâĂŹAsaro, Andrey Y Shcherbina, Jody M Klymak, Jeroen Molemaker, Guillaume Novelli, Cédric M Guigand, Angelique C Haza, Brian K Haus, Ed-

ward H Ryan, Gregg A Jacobs, et al. Ocean convergence and the dispersion of flotsam. *Proceedings of the National Academy of Sciences*, 115(6):1162–1167, 2018.

[53] Eftekhari, Ali A. FVTool, 2017. URL `http://fvt.simulkade.com/`.

[54] Gary D Egbert and Svetlana Y Erofeeva. Efficient inverse modeling of barotropic ocean tides. *Journal of Atmospheric and Oceanic Technology*, 19 (2):183–204, 2002.

[55] Gary D Egbert and Svetlana Y Erofeeva. OSU tidal inversion. `http\://volkov.oce.orst.edu/tides/tpxo8_atlas.html`, 2013.

[56] Jeff D Eldredge and Kwitae Chong. Fluid transport and coherent structures of translating and flapping wings. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 20(1):017509, 2010.

[57] Richard E. Ewing, Thomas F. Russell, and Mary Fanett Wheeler. Convergence analysis of an approximation of miscible displacement in porous media by mixed finite elements and a modified method of characteristics. *Computer Methods in Applied Mechanics and Engineering*, 47(1-2):73–92, 1984.

[58] G Falkovich, K Gawedzki, and M Vergassola. Lagrangian description of turbulence. In *New trends in turbulence Turbulence: nouveaux aspects*, pages 505–554. Springer, 2001.

[59] Mohammad Farazmand and George Haller. Computing Lagrangian coherent structures from their variational theory. *Chaos*, 22(1):1–12, 2012. ISSN 10541500.

[60] Mohammad Farazmand and George Haller. Computing lagrangian coherent structures from their variational theory. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 22(1):013128, 2012. doi: 10.1063/1.3690153.

[61] Mohammad Farazmand and George Haller. How coherent are the vortices of two-dimensional turbulence? *arXiv preprint arXiv:1402.4835*, 2014.

[62] Mohammad Farazmand and George Haller. Polar rotation angle identifies elliptic islands in unsteady dynamical systems. *Physica D: Nonlinear Phenomena*, 315:1âĂŞ12, 2016. doi: 10.1016/j.physd.2015.09.007.

[63] R. M. Fearn, T. Mullin, and K. A. Cliffe. Nonlinear flow phenomena in a symmetric sudden expansion. *Journal of Fluid Mechanics*, 211:595–608, 1990.

[64] Florian Feppon. Riemannian geometry of matrix manifolds for Lagrangian uncertainty quantification of stochastic fluid flows. Master's thesis, Massachusetts Institute of Technology, Department of Mechanical Engineering, Cambridge, Massachusetts, February 2017.

[65] Florian Feppon and Pierre F. J. Lermusiaux. Dynamically orthogonal numerical schemes for efficient stochastic advection and Lagrangian transport. *SIAM Review*, 60(3):595–625, 2018. doi: 10.1137/16M1109394.

[66] Florian Feppon and Pierre F. J. Lermusiaux. Extracting Lagrangian coherent subregions in the realistic ocean. 2020. In preparation.

[67] Joel H. Ferziger and Milovan Peric. *Computational methods for fluid dynamics*. Springer Science & Business Media, 2012.

[68] C. Foucart, C. Mirabito, P. J. Haley, Jr., and P. F. J. Lermusiaux. Distributed implementation and verification of hybridizable discontinuous Galerkin methods for nonhydrostatic ocean processes. In *OCEANS Conference 2018*, Charleston, SC, October 2018. IEEE. doi: 10.1109/OCEANS.2018.8604679.

[69] Leopoldo P. Franca, Sergio L. Frey, and Thomas Jr. Hughes. Stabilized finite element methods: I. Application to the advective-diffusive model. *Computer Methods in Applied Mechanics and Engineering*, 95(2):253–276, 1992.

[70] Gary Froyland. An analytic framework for identifying finite-time coherent sets in time-dependent dynamical systems. *Physica D: Nonlinear Phenomena*, 250: 1–19, 2013. ISSN 01672789.

[71] Gary Froyland. An analytic framework for identifying finite-time coherent sets in time-dependent dynamical systems. *Physica D: Nonlinear Phenomena*, 250: 1–19, 2013.

[72] Gary Froyland. Dynamic isoperimetry and the geometry of Lagrangian coherent structures. *Nonlinearity*, 28(10):3587, 2015.

[73] Gary Froyland and Eric Kwok. A dynamic Laplacian for identifying Lagrangian coherent structures on weighted Riemannian manifolds. *arXiv preprint arXiv:1610.01128*, 2016.

[74] Gary Froyland and Kathrin Padberg. Almost-invariant sets and invariant manifoldsâĂŤconnecting probabilistic and geometric descriptions of coherent structures in flows. *Physica D: Nonlinear Phenomena*, 238(16):1507–1523, 2009.

[75] Gary Froyland and Kathrin Padberg. Almost-invariant and finite-time coherent sets: directionality, duration, and diffusion. In *Ergodic Theory, Open Dynamics, and Coherent Structures*, pages 171–216. Springer, 2014.

[76] Gary Froyland, Kathrin Padberg, Matthew H England, and Anne Marie Treguier. Detection of coherent oceanic structures via transfer operators. *Physical review letters*, 98(22):224503, 2007. ISSN 00319007.

[77] Gary Froyland, Simon Lloyd, and Naratip Santitissadeekorn. Coherent sets for nonautonomous dynamical systems. *Physica D: Nonlinear Phenomena*, 239 (16):1527–1541, 2010. ISSN 01672789.

[78] Gary Froyland, Simon Lloyd, and Naratip Santitissadeekorn. Coherent sets for nonautonomous dynamical systems. *Physica D: Nonlinear Phenomena*, 239 (16):1527–1541, 2010.

[79] Martin J. Gander. 50 years of time parallel time integration. In *Multiple Shooting and Time Domain Decomposition Methods*, pages 69–113. Springer, 2015.

[80] Avijit Gangopadhyay, Pierre F.J. Lermusiaux, Leslie Rosenfeld, Allan R. Robinson, Leandro Calado, Hyun Sook Kim, Wayne G. Leslie, and Patrick J. Haley, Jr. The California Current system: A multiscale overview and the development of a feature-oriented regional modeling system (FORMS). *Dynamics of Atmospheres and Oceans*, 52(1–2):131–169, September 2011. doi: 10.1016/j.dynatmoce.2011.04.003. Special issue of Dynamics of Atmospheres and Oceans in honor of Prof. A. R. Robinson.

[81] Pierre Gentine, Mike Pritchard, Stephan Rasp, Gael Reinaudi, and Galen Yacalis. Could machine learning break the convection parameterization deadlock? *Geophysical Research Letters*, 45(11):5742–5751, 2018.

[82] Elena Gerwing, Matthias Hort, Jörn Behrens, and Bärbel Langmann. An adaptive semi-Lagrangian advection model for transport of volcanic emissions in the atmosphere. *Natural Hazards and Earth System Sciences*, 18(5):1517–1534, 2018.

[83] Thomas Gerz, Tilman Dürbeck, and Paul Konopka. Transport and effective diffusion of aircraft emissions. *Journal of Geophysical Research: Atmospheres*, 103(D20):25905–25913, 1998.

[84] Gene Golub, Knut Solna, and Paul Van Dooren. Computing the svd of a general matrix product/quotient. *SIAM Journal on Matrix Analysis and Applications*, 22(1):1–19, 2000.

[85] Sigal Gottlieb and Chi-Wang Shu. Total variation diminishing runge-kutta schemes. *Mathematics of Computation of the American Mathematical Society*, 67(221):73–85, 1998.

[86] Melissa A Green, Clarence W Rowley, and Alexander J Smits. Using hyperbolic

lagrangian coherent structures to investigate vortices in bioinspired fluid flows. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 20(1):017510, 2010.

[87] MSEAS Group. Deep sea mining sea experiment 2018; `http://mseas.mit.edu/Sea_exercises/DeepSeaMining/`, 2018. URL `http://mseas.mit.edu/Sea_exercises/DeepSeaMining/`.

[88] MSEAS Group. Nsf alpha sea experiment 2018; `http://mseas.mit.edu/Sea_exercises/NSF_ALPHA/2018/`, 2018. URL `http://mseas.mit.edu/Sea_exercises/NSF_ALPHA/2018/`.

[89] Abhinav Gupta and P. F. J. Lermusiaux. Bayesian learning of coupled physical-biogeochemical models. 2020. In preparation.

[90] Abhinav Gupta, Patrick J. Haley, Deepak N. Subramani, and Pierre F. J. Lermusiaux. Fish modeling and Bayesian learning for the Lakshadweep Islands. In *OCEANS 2019 MTS/IEEE SEATTLE*, pages 1–10, Seattle, October 2019. IEEE. doi: 10.23919/OCEANS40490.2019.8962892.

[91] David Ha, Andrew Dai, and Quoc V Le. Hypernetworks. *arXiv preprint arXiv:1609.09106*, 2016.

[92] Alireza Hadjighasem, Daniel Karrasch, Hiroshi Teramoto, and George Haller. Spectral-clustering approach to lagrangian vortex detection. *Physical Review E*, 93(6):063107, 2016.

[93] Alireza Hadjighasem, Mohammad Farazmand, Daniel Blazevski, Gary Froyland, and George Haller. A critical comparison of lagrangian methods for coherent structure detection. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 27(5):053104, 2017.

[94] P. J. Haley, Jr., P. F. J. Lermusiaux, A. R. Robinson, W. G. Leslie, O. Logoutov, G. Cossarini, X. S. Liang, P. Moreno, S. R. Ramp, J. D. Doyle, J. Bellingham, F. Chavez, and S. Johnston. Forecasting and reanalysis in the Monterey

Bay/California Current region for the Autonomous Ocean Sampling Network-II experiment. *Deep Sea Research Part II: Topical Studies in Oceanography*, 56 (3–5):127–148, February 2009. doi: 10.1016/j.dsr2.2008.08.010.

[95] P. J. Haley, Jr., A. Agarwal, and P. F. J. Lermusiaux. Optimizing velocities and transports for complex coastal regions and archipelagos. *Ocean Modeling*, 89:1–28, 2015. doi: 10.1016/j.ocemod.2015.02.005.

[96] Patrick J. Haley, Jr. and Pierre F. J. Lermusiaux. Multiscale two-way embedding schemes for free-surface primitive equations in the "Multidisciplinary Simulation, Estimation and Assimilation System". *Ocean Dynamics*, 60(6): 1497–1537, December 2010. doi: 10.1007/s10236-010-0349-4.

[97] Patrick J. Haley, Jr., Abhinav Gupta, Chris Mirabito, and Pierre F. J. Lermusiaux. Towards Bayesian ocean physical-biogeochemical-acidification prediction and learning systems for Massachusetts Bay. In *OCEANS 2020 IEEE/MTS*. IEEE, October 2020. In press.

[98] George Haller. Lagrangian coherent structures from approximate velocity data. *Physics of Fluids (1994-present)*, 14(6):1851–1861, 2002.

[99] George Haller. An objective definition of a vortex. *Journal of fluid mechanics*, 525:1–26, 2005.

[100] George Haller. A variational theory of hyperbolic Lagrangian coherent structures. *Physica D: Nonlinear Phenomena*, 240(7):574–598, 2011.

[101] George Haller. Lagrangian coherent structures. *Annual Review of Fluid Mechanics*, 47:137–162, 2015.

[102] George Haller and Francisco J Beron-Vera. Geodesic theory of transport barriers in two-dimensional flows. *Physica D: Nonlinear Phenomena*, 241(20):1680–1702, 2012.

[103] George Haller and Guocheng Yuan. Lagrangian coherent structures and mixing in two-dimensional turbulence. *Physica D: Nonlinear Phenomena*, 147(3):352–370, 2000.

[104] George Haller, Alireza Hadjighasem, Mohammad Farazmand, and Florian Huhn. Defining coherent vortices objectively from the vorticity. *Journal of Fluid Mechanics*, 795:136–173, 2016.

[105] George Haller, Alireza Hadjighasem, Mohammad Farazmand, and Florian Huhn. Defining coherent vortices objectively from the vorticity. *Journal of Fluid Mechanics*, 795:136–173, 2016.

[106] George Haller, Daniel Karrasch, and Florian Kogelbauer. Material barriers to diffusive and stochastic transport. *Proceedings of the National Academy of Sciences*, 115(37):9074–9079, 2018.

[107] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The elements of statistical learning: data mining, inference, and prediction*. Springer Science & Business Media, 2009.

[108] J.M. Hervouet. Application of the method of characteristics in their weak formulation to solving two-dimensional advection equations on mesh grids. *Recent Advances in Numerical Methods in Fluids*, 5:149–185, 1986.

[109] Jacob P. Heuss, Patrick J. Haley, Jr., Chris Mirabito, Emanuel Coelho, Martha C. Schönau, Kevin Heaney, and Pierre F. J. Lermusiaux. Reduced order modeling for stochastic prediction onboard autonomous platforms at sea. In *OCEANS 2020 IEEE/MTS*. IEEE, October 2020. In press.

[110] Porter Hoagland, Stace Beaulieu, Maurice A. Tivey, Roderick G. Eggert, Christopher German, Lyle Glowka, and Jian Lin. Deep-sea mining of seafloor massive sulfides. *Marine Policy*, 34(3):728–732, 2010.

[111] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[112] Paul Houston, Christoph Schwab, and Endre Süli. Discontinuous *hp*-finite element methods for advection-diffusion-reaction problems. *SIAM Journal on Numerical Analysis*, 39(6):2133–2163, 2002.

[113] BL Hua and P Klein. An exact criterion for the stirring properties of nearly two-dimensional turbulence. *Physica D: Nonlinear Phenomena*, 113(1):98–110, 1998.

[114] Thomas Jr. Hughes. A simple scheme for developing 'upwind' finite elements. *International Journal for Numerical Methods in Engineering*, 12(9):1359–1365, 1978.

[115] Thomas Jr. Hughes, Leopoldo P. Franca, and Gregory M. Hulbert. A new finite element formulation for computational fluid dynamics: VIII. the Galerkin/least-squares method for advective-diffusive equations. *Computer Methods in Applied Mechanics and Engineering*, 73(2):173–189, 1989.

[116] Willem Hundsdorfer and Jan G. Verwer. *Numerical solution of time-dependent advection-diffusion-reaction equations*, volume 33. Springer Science & Business Media, 2013.

[117] Helga S. Huntley, Bruce L. Jr. Lipphardt, and Albert D. Kirwan. Surface drift predictions of the deepwater horizon spill: The Lagrangian perspective. *Monitoring and Modeling the Deepwater Horizon Oil Spill: A Record-Breaking Enterprise*, pages 179–195, 2013.

[118] T Ilyina, T Pohlmann, G Lammel, and J Sündermann. A fate and transport ocean model for persistent organic pollutants and its application to the North Sea. *Journal of Marine Systems*, 63(1-2):1–19, 2006.

[119] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[120] Jinhee Jeong and Fazle Hussain. On the identification of a vortex. *Journal of fluid mechanics*, 285:69–94, 1995.

[121] Hongen Jia and Kaitai Li. A third accurate operator splitting method. *Mathematical and Computer Modelling*, 53(1-2):387–396, 2011.

[122] Guang-Shan Jiang and Chi-Wang Shu. Efficient implementation of weighted ENO schemes. *Journal of Computational Physics*, 126(1):202–228, 1996.

[123] Fritz John. Rotation and strain. *Communications on Pure and Applied Mathematics*, 14(3):391–413, 1961.

[124] Hossein Amini Kafiabad, Pak Wai Chan, and George Haller. Lagrangian detection of wind shear for landing aircraft. *Journal of Atmospheric and Oceanic Technology*, 30(12):2808–2819, 2013.

[125] Emilie Tew Kai, Vincent Rossi, Joel Sudre, Henri Weimerskirch, Cristobal Lopez, Emilio Hernandez-Garcia, Francis Marsac, and Veronique Garçon. Top marine predators track lagrangian coherent structures. *Proceedings of the National Academy of Sciences*, 106(20):8245–8250, 2009.

[126] George Karniadakis and Spencer Sherwin. *Spectral/ hp element methods for computational fluid dynamics*. Oxford University Press, 2013.

[127] Daniel Karrasch. Attracting Lagrangian coherent structures on Riemannian manifolds. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 25(8): 087411, 2015.

[128] S. M. Kelly and P. F. J. Lermusiaux. Internal-tide interactions with Gulf Stream and Middle Atlantic Bight shelfbreak front. *Journal of Geophysical Research: Oceans*, 121:6271–6294, 2016. doi: 10.1002/2016JC011639.

[129] Richard Kleeman. Information theory and dynamical system predictability. *Entropy*, 13(3):612–649, 2011.

[130] Sotiris B Kotsiantis, I Zaharakis, and P Pintelas. Supervised machine learning: A review of classification techniques. *Emerging artificial intelligence applications in computer engineering*, 160(1):3–24, 2007.

[131] C. S. Kulkarni, P. J. Haley, Jr., P. F. J. Lermusiaux, A. Dutt, A. Gupta, C. Mirabito, D. N. Subramani, S. Jana, W. H. Ali, T. Peacock, C. M. Royo, A. Rzeznik, and R. Supekar. Real-time sediment plume modeling in the Southern California Bight. In *OCEANS Conference 2018*, Charleston, SC, October 2018. IEEE. doi: 10.1109/OCEANS.2018.8653642.

[132] Chinmay S. Kulkarni and Pierre F. J. Lermusiaux. Advection without compounding errors through flow map composition. *Journal of Computational Physics*, 398:108859, December 2019. doi: 10.1016/j.jcp.2019.108859.

[133] Chinmay S. Kulkarni and Pierre F. J. Lermusiaux. Learning Lagrangian transport features from Eulerian observations. 2020. In preparation.

[134] Chinmay S. Kulkarni and Pierre F. J. Lermusiaux. Learning material transport features from advective and diffusive tracer observations in ocean flows. 2020. In preparation.

[135] Chinmay S. Kulkarni and Pierre F. J. Lermusiaux. Three-dimensional time-optimal path planning in the ocean. *Ocean Modelling*, 152, August 2020. doi: 10.1016/j.ocemod.2020.101644.

[136] Chinmay S. Kulkarni and Pierre F. J. Lermusiaux. Persistent rigid sets in realistic fluid flows using flow map composition. *Ocean Modelling*, 2020. In preparation.

[137] Chinmay S. Kulkarni, Abhinav Gupta, and Pierre F. J. Lermusiaux. Sparse regression and adaptive feature generation for the discovery of dynamical systems. In *InfoSymbiotics/DDDAS2020*, Boston, October 2020. In press.

[138] Chinmay Sameer Kulkarni. Three-dimensional time-optimal path planning in dynamic and realistic environments. Master's thesis, Massachusetts Insti-

tute of Technology, Department of Mechanical Engineering, Cambridge, Massachusetts, June 2017.

[139] PK Kundu, IM Cohen, and DR Dowling. Fluid mechanics, 5th version. *Academic, Berlin*, 2012.

[140] Debby Lanser and Jan G. Verwer. Analysis of operator splitting for advection–diffusion–reaction problems from air pollution modelling. *Journal of Computational and Applied Mathematics*, 111(1-2):201–216, 1999.

[141] Leon Lapidus and George F. Pinder. *Numerical solution of partial differential equations in science and engineering*. John Wiley & Sons, 2011.

[142] L Lebreton, B Slat, F Ferrari, B Sainte-Rose, J Aitken, R Marthouse, S Hajbane, Serena Cunsolo, A Schwarz, A Levivier, et al. Evidence that the great Pacific garbage patch is rapidly accumulating plastic. *Scientific Reports*, 8(1): 4666, 2018.

[143] Francois Lekien, Chad Coulliette, Arthur J Mariano, Edward H Ryan, Lynn K Shay, George Haller, and Jerry Marsden. Pollution release tied to invariant manifolds: A case study for the coast of florida. *Physica D: Nonlinear Phenomena*, 210(1):1–20, 2005.

[144] P. F. J. Lermusiaux. Evolving the subspace of the three-dimensional multiscale ocean variability: Massachusetts Bay. *Journal of Marine Systems*, 29(1):385–422, 2001. doi: 10.1016/S0924-7963(01)00025-2.

[145] P. F. J. Lermusiaux and F. Lekien. Dynamics and Lagrangian coherent structures in the ocean and their uncertainty. In Jerrold E. Marsden and Jurgen Scheurle, editors, *Extended Abstract in report of the Dynamical System Methods in Fluid Dynamics Oberwolfach Workshop*, page 2, Germany, July 31st - August 6th 2005. Mathematisches Forschungsinstitut Oberwolfach. doi: 10.1121/1.2988093.

[146] P. F. J Lermusiaux, P. J. Haley, Jr, and N. K. Yilmaz. Environmental prediction, path planning and adaptive sampling: sensing and modeling for efficient ocean monitoring, management and pollution control. *Sea Technology*, 48(9): 35–38, 2007.

[147] P. F. J. Lermusiaux, P. J. Haley, W. G. Leslie, A. Agarwal, O. Logutov, and L. J. Burton. Multiscale physical and biological dynamics in the Philippine Archipelago: Predictions and processes. *Oceanography*, 24(1):70–89, 2011. doi: 10.5670/oceanog.2011.05. Special Issue on the Philippine Straits Dynamics Experiment.

[148] P. F. J. Lermusiaux, P. J. Haley, Jr., S. Jana, A. Gupta, C. S. Kulkarni, C. Mirabito, W. H. Ali, D. N. Subramani, A. Dutt, J. Lin, A. Shcherbina, C. Lee, and A. Gangopadhyay. Optimal planning and sampling predictions for autonomous and Lagrangian platforms and sensors in the northern Arabian Sea. *Oceanography*, 30(2):172–185, June 2017. doi: 10.5670/oceanog.2017.242. Special issue on Autonomous and Lagrangian Platforms and Sensors (ALPS).

[149] P. F. J. Lermusiaux, D. N. Subramani, J. Lin, C. S. Kulkarni, A. Gupta, A. Dutt, T. Lolla, P. J. Haley, Jr., W. H. Ali, C. Mirabito, and S. Jana. A future for intelligent autonomous ocean observing systems. *Journal of Marine Research*, 75(6):765–813, November 2017. doi: 10.1357/002224017823524035. The Sea. Volume 17, The Science of Ocean Prediction, Part 2.

[150] Pierre F J Lermusiaux. *Error Subspace Data Assimilation Methods for Ocean Field Estimation: Theory, Validation and Applications*. Harvard University, 1997.

[151] Pierre F. J. Lermusiaux. Numerical fluid mechanics. MIT OpenCourseWare, May 2015. URL `https://ocw.mit.edu/courses/mechanical-engineering/2-29-numerical-fluid-mechanics-spring-2015/lecture-notes-and-references/`.

[152] Pierre F. J. Lermusiaux, Jinshan Xu, Chi-Fang Chen, Sen Jan, L.Y. Chiu, and Yiing-Jang Yang. Coupled ocean–acoustic prediction of transmission loss in a continental shelfbreak region: Predictive skill, uncertainty quantification, and dynamical sensitivities. *IEEE Journal of Oceanic Engineering*, 35(4):895–916, October 2010. doi: 10.1109/JOE.2010.2068611.

[153] Pierre F. J. Lermusiaux, Manan Doshi, Chinmay S. Kulkarni, Abhinav Gupta, Patrick J. Haley, Jr., Chris Mirabito, Francesco Trotta, S. J. Levang, G. R. Flierl, J. Marshall, Thomas Peacock, and C. Noble. Plastic pollution in the coastal oceans: Characterization and modeling. In *OCEANS 2019 MTS/IEEE SEATTLE*, pages 1–10, Seattle, October 2019. IEEE. doi: 10.23919/OCEANS40490.2019.8962786.

[154] W. G. Leslie, A. R. Robinson, P. J. Haley, Jr, O. Logutov, P. A. Moreno, P. F. J. Lermusiaux, and E. Coelho. Verification and training of real-time forecasting of multi-scale ocean dynamics for maritime rapid environmental assessment. *Journal of Marine Systems*, 69(1):3–16, 2008. doi: 10.1016/j.jmarsys.2007.02.001.

[155] W. G. Leslie, P. J. Haley, Jr., P. F. J. Lermusiaux, M. P. Ueckermann, O. Logutov, and J. Xu. MSEAS Manual. MSEAS Report 06, Department of Mechanical Engineering, Massachusetts Institute of Technology, Cambridge, MA, 2010. URL `http://mseas.mit.edu/?p=2237`.

[156] Shingyu Leung. An Eulerian approach for computing the finite time Lyapunov exponent. *Journal of Computational Physics*, 230(9):3500–3524, 2011.

[157] Shingyu Leung. The backward phase flow method for the Eulerian finite time Lyapunov exponent computations. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 23(4):043132, 2013.

[158] Randall J LeVeque. Nonlinear conservation laws and finite volume methods. In *Computational Methods for Astrophysical Fluid Flow*, pages 1–159. Springer, 1998.

[159] Randall J. LeVeque. *Finite volume methods for hyperbolic problems*, volume 31. Cambridge University Press, 2002.

[160] O. G. Logutov and P. F. J. Lermusiaux. Inverse barotropic tidal estimation for regional ocean applications. *Ocean Modelling*, 25(1–2):17–34, 2008. ISSN 1463-5003. doi: 10.1016/j.ocemod.2008.06.004. URL `http://www.sciencedirect.com/science/article/pii/S1463500308000851`.

[161] T. Lolla and P. F. J. Lermusiaux. A Gaussian mixture model smoother for continuous nonlinear stochastic dynamical systems: Applications. *Monthly Weather Review*, 145:2763–2790, July 2017. doi: 10.1175/MWR-D-16-0065.1.

[162] T. Lolla, P. F. J. Lermusiaux, M. P. Ueckermann, and P. J. Haley, Jr. Time-optimal path planning in dynamic flows using level set equations: Theory and schemes. *Ocean Dynamics*, 64(10):1373–1397, 2014. doi: 10.1007/s10236-014-0757-y.

[163] T. Lolla, P. J. Haley, Jr., and P. F. J. Lermusiaux. Path planning in multiscale ocean flows: Coordination and dynamic obstacles. *Ocean Modelling*, 94:46–66, 2015. doi: 10.1016/j.ocemod.2015.07.013.

[164] Zichao Long, Yiping Lu, Xianzhong Ma, and Bin Dong. Pde-net: Learning pdes from data. *arXiv preprint arXiv:1710.09668*, 2017.

[165] P. G. Y. Lu and Pierre F. J. Lermusiaux. Pde-based bayesian inference of high-dimensional dynamical models. MSEAS Report 19, Department of Mechanical Engineering, Massachusetts Institute of Technology, Cambridge, MA, USA, 2014.

[166] Peter Guang Yi Lu. Bayesian inference of stochastic dynamical models. Master's thesis, Massachusetts Institute of Technology, Department of Mechanical Engineering, Cambridge, Massachusetts, February 2013.

[167] Bethany Lusch, J Nathan Kutz, and Steven L Brunton. Deep learning for

universal linear embeddings of nonlinear dynamics. *Nature communications*, 9 (1):1–10, 2018.

[168] Zu-Hui Ma, Weng Cho Chew, and Lijun Jiang. A novel fast solver for Poisson equation with the Neumann boundary condition. *arXiv preprint arXiv:1207.4260*, 2012.

[169] David JC MacKay and David JC Mac Kay. *Information theory, inference and learning algorithms*. Cambridge university press, 2003.

[170] Shev MacNamara and Gilbert Strang. Operator splitting. In *Splitting Methods in Communication, Imaging, Science, and Engineering*, pages 95–114. Springer, 2016.

[171] Andrei V. Malevsky and Stephen J. Thomas. Parallel algorithms for semi-Lagrangian advection. *International Journal for Numerical Methods in Fluids*, 25(4):455–473, 1997.

[172] Gianmarco Manzini and Alessandro Russo. A finite volume method for advection–diffusion problems in convection-dominated regimes. *Computer Methods in Applied Mechanics and Engineering*, 197(13-16):1242–1261, 2008.

[173] Manikandan Mathur, George Haller, Thomas Peacock, Jori E Ruppert-Felsot, and Harry L Swinney. Uncovering the lagrangian skeleton of turbulence. *Physical Review Letters*, 98(14):144502, 2007.

[174] Dennis J McGillicuddy Jr. Mechanisms of physical-biological-biogeochemical interaction at the oceanic mesoscale. *Annual Review of Marine Science*, 8: 125–159, 2016.

[175] Jefferson Ryan Medel. Anomaly detection using predictive convolutional long short-term memory units. *PhD Thesis*, 2016.

[176] JD Meiss. Symplectic maps, variational principles, and transport. *Reviews of Modern Physics*, 64(3):795, 1992.

[177] Olivier Mercier and Jean-Christophe Nave. The characteristic mapping method for the linear advection of arbitrary sets. *arXiv preprint arXiv:1309.2731*, 2013.

[178] Eric Mjolsness and Dennis DeCoste. Machine learning for science: state of the art and future prospects. *science*, 293(5537):2051–2055, 2001.

[179] Akbar Mohebbi and Mehdi Dehghan. High-order compact solution of the one-dimensional heat and advection–diffusion equations. *Applied Mathematical Modelling*, 34(10):3071–3084, 2010.

[180] MSEAS Group. MSEAS Software, 2013. URL `http://mseas.mit.edu/software/`.

[181] Carlos Munoz-Royo, Thomas Peacock, Matthew H. Alford, Jerome Smith, Arnaud Le Boyer, Chinmay S. Kulkarni, Pierre F. J. Lermusiaux, Patrick J. Haley, Jr., Chris Mirabito, Dayang Wang, E. Eric Adams, Alexander Breugem, Boudewijn Decrop, Thijs Lanckriet, Rohit Supekar, Andrew Rzeznik, and Amy Gartman. Assessing the scale of deep-sea nodule mining midwater discharge sediment plumes. *Nature Communications Earth & Environment*, 2020. Subjudice.

[182] Preetum Nakkiran, Gal Kaplun, Yamini Bansal, Tristan Yang, Boaz Barak, and Ilya Sutskever. Deep double descent: Where bigger models and more data hurt. *arXiv preprint arXiv:1912.02292*, 2019.

[183] Jean-Christophe Nave, Rodolfo Ruben Rosales, and Benjamin Seibold. A gradient-augmented level set method with an optimally local, coherent advection scheme. *Journal of Computational Physics*, 229(10):3802–3827, 2010.

[184] Shlomo P. Neuman and Shaul Sorek. Eulerian-Lagrangian methods for advection-dispersion. In *Finite Elements in Water Resources*, pages 849–876. Springer, 1982.

[185] Jeeheh Oh, Jiaxuan Wang, Shengpu Tang, Michael W Sjoding, and Jenna Wiens. Relaxed parameter sharing: Effectively modeling time-varying relation-

ships in clinical time-series. In *Machine Learning for Healthcare Conference*, pages 27–52, 2019.

[186] Akira Okubo. Horizontal dispersion of floatable particles in the vicinity of velocity singularities such as convergences. In *Deep sea research and oceanographic abstracts*, volume 17-3, pages 445–454. Elsevier, 1970.

[187] María J Olascoaga and George Haller. Forecasting sudden changes in environmental pollution patterns. *Proceedings of the National Academy of Sciences*, 109(13):4738–4743, 2012.

[188] Anabela Pacheco de Oliveira. A comparison of Eulerian-Lagrangian methods for the solution of the transport equation, 1994.

[189] Stanley Osher and Ronald Fedkiw. *Level set methods and dynamic implicit surfaces*, volume 153. Springer Science & Business Media, 2006.

[190] Stanley Osher and James A. Sethian. Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations. *Journal of Computational Physics*, 79(1):12–49, 1988.

[191] Stanley Osher and Chi-Wang Shu. High-order essentially nonoscillatory schemes for Hamilton–Jacobi equations. *SIAM Journal on Numerical Analysis*, 28(4): 907–922, 1991.

[192] Tamay M Özgökmen, Francisco J Beron-Vera, Darek Bogucki, Shuyi S Chen, Clint Dawson, William Dewar, Annalisa Griffa, Brian K Haus, Angelique C Haza, Helga Huntley, et al. Research overview of the consortium for advanced research on transport of hydrocarbon in the environment (carthe). In *International Oil Spill Conference Proceedings*, volume 2014-1, pages 544–560. American Petroleum Institute, 2014.

[193] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *International conference on machine learning*, pages 1310–1318, 2013.

[194] Christian Oliver Paschereit, Ephraim Gutmark, and Wolfgang Weisenstein. Coherent structures in swirling flows and their role in acoustic combustion control. *Physics of Fluids*, 11(9):2667–2678, 1999.

[195] Claudia Pasquero, Annalisa Bracco, and Antonello Provenzale. Coherent vortices, Lagrangian particles and the marine ecosystem. *Shallow Flows*, pages 399–412, 2004.

[196] Suhas V. Patankar. Numerical heat transfer and fluid flow. *Hemisphere, Washington, DC*, pages 115–120, 1980.

[197] Thomas Peacock and John Dabiri. Introduction to focus issue: Lagrangian coherent structures, 2010.

[198] Thomas Peacock and George Haller. Lagrangian coherent structures: The hidden skeleton of fluid flows. *Physics today*, 66(2):41–47, 2013.

[199] Andreas Pfeuffer and Klaus Dietmayer. Separable convolutional lstms for faster video segmentation. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 1072–1078. IEEE, 2019.

[200] Nadia Pinardi, P. F. J. Lermusiaux, K. H. Brink, and R. Preller. The sea: The science of ocean prediction. *Journal of Marine Research*, 75(3):101–102, May 2017. Special issue: The Science of Ocean Prediction, vol. 17 of The Sea.

[201] George F. Pinder and Hilton H. Cooper. A numerical technique for calculating the transient position of the saltwater front. *Water Resources Research*, 6(3): 875–882, 1970.

[202] S. V. Prants. Chaotic Lagrangian transport and mixing in the ocean. *The European Physical Journal Special Topics*, 223(13):2723–2743, 2014.

[203] Michael J. Prather. Numerical advection by conservation of second-order moments. *Journal of Geophysical Research: Atmospheres*, 91(D6):6671–6681, 1986.

[204] Larry Pratt, Roy Barkan, and Irina Rypina. Scalar flux kinematics. *Fluids*, 1 (3):27, 2016.

[205] Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. Machine learning of linear differential equations using gaussian processes. *Journal of Computational Physics*, 348:683–693, 2017.

[206] Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019.

[207] Maziar Raissi, Alireza Yazdani, and George Em Karniadakis. Hidden fluid mechanics: Learning velocity and pressure fields from flow visualizations. *Science*, 367(6481):1026–1030, 2020.

[208] Steven R. Ramp, Pierre F. J. Lermusiaux, Igor Shulman, Yi Chao, Rebecca E. Wolf, and Frederick L. Bahr. Oceanographic and atmospheric conditions on the continental shelf north of the Monterey Bay during August 2006. *Dynamics of Atmospheres and Oceans*, 52(1–2):192–223, September 2011. doi: 10.1016/j.dynatmoce.2011.04.005. Special issue of Dynamics of Atmospheres and Oceans in honor of Prof. A. R. Robinson.

[209] Stephan Rasp, Michael S Pritchard, and Pierre Gentine. Deep learning to represent subgrid processes in climate models. *Proceedings of the National Academy of Sciences*, 115(39):9684–9689, 2018.

[210] Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows. *arXiv preprint arXiv:1505.05770*, 2015.

[211] A. R. Robinson, P. J. Haley, P. F. J. Lermusiaux, and W. G. Leslie. Predictive skill, predictive capability and predictability in ocean forecasting. In *Proceedings of "The OCEANS 2002 MTS/IEEE" conference*, pages 787–794. Holland Publications, September 2002. doi: 10.1109/oceans.2002.1192070.

[212] Dean Roemmich, Gregory C Johnson, Stephen Riser, Russ Davis, John Gilson, W Brechner Owens, Silvia L Garzoli, Claudia Schmid, and Mark Ignaszewski. The argo program: Observing the global ocean with profiling floats. *Oceanography*, 22(2):34–43, 2009.

[213] Samuel H Rudy, Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. Data-driven discovery of partial differential equations. *Science Advances*, 3(4): e1602614, 2017.

[214] Thomas F. Russell. Time stepping along characteristics with incomplete iteration for a Galerkin approximation of miscible displacement in porous media. *SIAM Journal on Numerical Analysis*, 22(5):970–1013, 1985.

[215] Tim Salimans and Durk P Kingma. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. In *Advances in neural information processing systems*, pages 901–909, 2016.

[216] Carmelo Santaera, Rosa M. Pidatella, and Filippo Stanco. ENO/WENO interpolation methods for the zooming of digital images. *Communications to SIMAI Congress*, 1, 2007.

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

[218] Hayden Schaeffer. Learning partial differential equations via data discovery and sparse optimization. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 473(2197):20160446, 2017.

[219] A. Schelin, G. Károlyi, A.P.S. De Moura, N.A. Booth, and C. Grebogi. Chaotic advection in blood flow. *Physical Review E*, 80(1):016213, 2009.

[220] Benjamin Seibold, Jean-Christophe Nave, and Rodolfo Ruben Rosales. Jet schemes for advection problems. *arXiv preprint arXiv:1101.5374*, 2011.

[221] Mattia Serra and George Haller. Objective Eulerian Coherent Structures. *arXiv preprint arXiv:1512.02112*, 2015.

[222] Mattia Serra, Pratik Sathe, Irina Rypina, Anthony Kirincich, Shae D. Ross, Pierre Lermusiaux, Arthur Allen, Thomas Peacock, and George Haller. Search and rescue at sea aided by hidden flow structures. *Nature Communications*, 11: 1–7, 2020. doi: 10.1038/s41467-020-16281-x.

[223] Shawn C Shadden and Charles A Taylor. Characterization of coherent structures in the cardiovascular system. *Annals of biomedical engineering*, 36(7): 1152–1162, 2008.

[224] Shawn C Shadden, Francois Lekien, and Jerrold E Marsden. Definition and properties of Lagrangian coherent structures from finite-time Lyapunov exponents in two-dimensional aperiodic flows. *Physica D: Nonlinear Phenomena*, 212(3):271–304, 2005.

[225] Shawn C Shadden, Francois Lekien, and Jerrold E Marsden. Definition and properties of lagrangian coherent structures from finite-time lyapunov exponents in two-dimensional aperiodic flows. *Physica D: Nonlinear Phenomena*, 212(3):271–304, 2005.

[226] Shawn C Shadden, Matteo Astorino, and Jean-Frédéric Gerbeau. Computational analysis of an aortic valve jet with lagrangian coherent structures. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 20(1):017512, 2010.

[227] Arjun Sharma, Irina I Rypina, Ruth Musgrave, and George Haller. Analytic reconstruction of a two-dimensional velocity field from an observed diffusive scalar. *Journal of Fluid Mechanics*, 871:755–774, 2019.

[228] Peter Sherman and Erik Van Sebille. Modeling marine surface microplastic transport to assess optimal removal locations. *Environmental Research Letters*, 11(1):014006, 2016.

[229] Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *Advances in neural information processing systems*, pages 802–810, 2015.

[230] Chi-Wang Shu. Essentially non-oscillatory and weighted essentially non-oscillatory schemes for hyperbolic conservation laws. In *Advanced numerical approximation of nonlinear hyperbolic equations*, pages 325–432. Springer, 1998.

[231] Walter HF Smith and David T Sandwell. Global sea floor topography from satellite altimetry and ship depth soundings. *Science*, 277(5334):1956–1962, 1997.

[232] T. Sondergaard and P. F. J. Lermusiaux. Data assimilation with Gaussian Mixture Models using the Dynamically Orthogonal field equations. Part I: Theory and scheme. *Monthly Weather Review*, 141(6):1737–1760, 2013. doi: 10.1175/MWR-D-11-00295.1.

[233] T. Sondergaard and P. F. J. Lermusiaux. Data assimilation with Gaussian Mixture Models using the Dynamically Orthogonal field equations. Part II: Applications. *Monthly Weather Review*, 141(6):1761–1785, 2013. doi: 10.1175/MWR-D-11-00296.1.

[234] Sudhir Srivastava. Operator splitting methods for the advection-diffusion-reaction equation. Centre for Analysis, Scientifc Computing and Applications, TU Eindhoven, 2008.

[235] Gilbert Strang. On the construction and comparison of difference schemes. *SIAM Journal on Numerical Analysis*, 5(3):506–517, 1968.

[236] Mark A. Stremler, Frederick R. Haselton, and Hassan Aref. Designing for chaos: Applications of chaotic advection at the microscale. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 362(1818):1019–1036, 2004.

[237] Steven H Strogatz. *Nonlinear dynamics and chaos: with applications to physics, biology, chemistry, and engineering*. CRC Press, 2018.

[238] D. N. Subramani and P. F. J. Lermusiaux. Energy-optimal path planning by stochastic dynamically orthogonal level-set optimization. *Ocean Modeling*, 100: 57–77, 2016. doi: 10.1016/j.ocemod.2016.01.006.

[239] D. N. Subramani, P. J. Haley, Jr., and P. F. J. Lermusiaux. Energy-optimal path planning in the coastal ocean. *Journal of Geophysical Research: Oceans*, 122:3981–4003, 2017. doi: 10.1002/2016JC012231.

[240] D. N. Subramani, P. F. J. Lermusiaux, P. J. Haley, Jr., C. Mirabito, S. Jana, C. S. Kulkarni, A. Girard, D. Wickman, J. Edwards, and J. Smith. Time-optimal path planning: Real-time sea exercises. In *Oceans '17 MTS/IEEE Conference*, Aberdeen, June 2017. doi: 10.1109/OCEANSE.2017.8084776.

[241] Endre Süli and David F. Mayers. *An introduction to numerical analysis*. Cambridge University Press, 2003.

[242] Wenbo Tang, Pak Wai Chan, and George Haller. Accurate extraction of lagrangian coherent structures over finite domains with application to flight data analysis over hong kong international airport. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 20(1):017502, 2010.

[243] Vladimir A. Titarev and Eleuterio F. Toro. ADER: Arbitrary high order Godunov approach. *Journal of Scientific Computing*, 17(1-4):609–618, 2002.

[244] Belinda Tzen and Maxim Raginsky. Neural stochastic differential equations: Deep latent gaussian models in the diffusion limit. *arXiv preprint arXiv:1905.09883*, 2019.

[245] UC San Diego. R/V Sally Ride, 2020. URL `https://scripps.ucsd.edu/ships/sally-ride`.

[246] M. P. Ueckermann and P. F. J. Lermusiaux. High order schemes for 2D unsteady biogeochemical ocean models. *Ocean Dynamics*, 60(6):1415–1445, December 2010. doi: 10.1007/s10236-010-0351-x.

[247] M. P. Ueckermann and P. F. J. Lermusiaux. 2.29 Finite Volume MATLAB Framework Documentation. MSEAS Report 14, Department of Mechanical Engineering, Massachusetts Institute of Technology, Cambridge, MA, 2012. URL `http://mseas.mit.edu/?p=2567`.

[248] M. P. Ueckermann and P. F. J. Lermusiaux. Hybridizable discontinuous Galerkin projection methods for Navier–Stokes and Boussinesq equations. *Journal of Computational Physics*, 306:390–421, 2016. doi: 10.1016/j.jcp.2015.11.028.

[249] Albert J. Valocchi and Michael Malmstead. Accuracy of operator splitting for advection-dispersion-reaction problems. *Water Resources Research*, 28(5): 1471–1476, 1992.

[250] Erik van Sebille, Stephen M Griffies, Ryan Abernathey, Thomas P Adams, Pavel Berloff, Arne Biastoch, Bruno Blanke, Eric P Chassignet, Yu Cheng, Colin J Cotter, et al. Lagrangian ocean analysis: fundamentals and practices. *Ocean Modelling*, 2017.

[251] A Vanreusel, A Hiliario, PA Ribeiro, L Menot, and P Arbizu Martinez. Managing impacts of deep sea resource exploitation, 2016.

[252] Maria Vernet, Ingrid H Ellingsen, Lena Seuthe, Dag Slagstad, Mattias R Cape, and Patricia A Matrai. Influence of phytoplankton advection on the productivity along the atlantic water inflow to the arctic ocean. 2019.

[253] Jérôme Vétel, André Garon, and Dominique Pelletier. Lagrangian coherent structures in the human carotid artery bifurcation. *Experiments in fluids*, 46 (6):1067–1079, 2009.

[254] Ruben Villegas, Jimei Yang, Seunghoon Hong, Xunyu Lin, and Honglak Lee. Decomposing motion and content for natural video sequence prediction. *arXiv preprint arXiv:1706.08033*, 2017.

[255] Hong Wang, Helge K. Dahle, Richard E. Ewing, Magne S. Espedal, Robert C. Sharpley, and Shushuang Man. An ellam scheme for advection-diffusion equations in two dimensions. *SIAM Journal on Scientific Computing*, 20(6):2160–2194, 1999.

[256] John Weiss. The dynamics of enstrophy transfer in two-dimensional hydrodynamics. *Physica D: Nonlinear Phenomena*, 48(2-3):273–294, 1991.

[257] Paul J Werbos. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990.

[258] Mary F. Wheeler and Clint N. Dawson. An operator-splitting method for advection-diffusion-reaction problems. *The Mathematics of Finite Elements and Applications VI*, pages 463–382, 1987.

[259] Jin-Long Wu, Heng Xiao, and Eric Paterson. Physics-informed machine learning approach for augmenting turbulence models: A comprehensive framework. *Physical Review Fluids*, 3(7):074602, 2018.

[260] You Xie, Erik Franz, Mengyu Chu, and Nils Thuerey. tempogan: A temporally coherent, volumetric gan for super-resolution fluid flow. *ACM Transactions on Graphics (TOG)*, 37(4):1–15, 2018.

[261] Huazhe Xu, Yang Gao, Fisher Yu, and Trevor Darrell. End-to-end learning of driving models from large-scale video datasets. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2174–2182, 2017.

[262] J. Xu, P. F. J. Lermusiaux, P. J. Haley Jr., W. G. Leslie, and O. G. Logutov. Spatial and Temporal Variations in Acoustic propagation during the PLUSNet-07 Exercise in Dabob Bay. In *Proceedings of Meetings on Acoustics (POMA)*,

volume 4, page 11. Acoustical Society of America 155th Meeting, 2008. doi: 10.1121/1.2988093.

[263] Huijun Yang and Zhengyu Liu. Chaotic transport in a double gyre ocean. *Geophysical research letters*, 21(7):545–548, 1994.

[264] Liu Yang, Dongkun Zhang, and George Em Karniadakis. Physics-informed generative adversarial networks for stochastic differential equations. *SIAM Journal on Scientific Computing*, 42(1):A292–A317, 2020.

[265] Guoqiao You and Shingyu Leung. Eulerian based interpolation schemes for flow map construction and line integral computation with applications to Lagrangian coherent structures extraction. *Journal of Scientific Computing*, pages 1–27, 2017.

[266] Guoqiao You, Tony Wong, and Shingyu Leung. Eulerian methods for visualizing continuous dynamical systems using Lyapunov exponents. *SIAM Journal on Scientific Computing*, 39(2):A415–A437, 2017.

[267] Ze Jia Zhang and Karthikeyan Duraisamy. Machine learning methods for data-driven turbulence modeling. In *22nd AIAA Computational Fluid Dynamics Conference*, page 2460, 2015.