Lagrangian Data Assimilation and Manifold Detection for a Point-Vortex Model

Final Report

Author:
David Darmon
AMSC
ddarmon (at) math.umd.edu

Advisor:
Dr. Kayo Ide
AOSC, IPST, CSCAMM, ESSIC
ide (at) umd.edu

Abstract

The process of assimilating data into geophysical models is of great practical importance. Classical approaches to this problem have considered the data from an *Eulerian* perspective, where the measurements of interest are flow velocities through fixed instruments. An alternative approach considers the data from a *Lagrangian* perspective, where the position of particles are tracked instead of the underlying flow field. The Lagrangian perspective also permits the application of tools from dynamical systems theory to the study of flows. However, very simple flow fields may lead to highly nonlinear particle trajectories. Thus, special care must be paid to the data assimilation methods applied. This project will apply Lagrangian data assimilation to a model point-vortex system using three assimilation schemes: the extended Kalman filter, the ensemble Kalman filter, and the particle filter. The effectiveness of these schemes at tracking the hidden state of the flow will be quantified. The project will also consider opportunities for observing system design (the optimization of observing systems through knowledge of the underlying dynamics of the observed system) by applying a methodology for detecting manifolds within the structure of the flow.

May 11, 2012

1 Project Goals

The goals of this project can be divided into two phases. During the first phase of the project, we sought to implement and validate three data assimilation methods for the point vortex system with N_v vortices and N_d: the extended Kalman filter (EKF), the local ensemble transform Kalman filter (LETKF), and the particle filter. Once these filtering methods were implemented, we considered the performance of the filters in a two vortex, single drifter case that has been previously studied using the EKF and particle filter [19]. We successfully tested the filters' performance in various dynamical regimes by varying the initial placement of the drifter in the flow field induced by the point vortices.

The second phase of the project extended upon the work done during the first phase. During the second phase, we considered the dynamical structure inherent in the flow created by the point vortices and how we could take advantage of this structure to design a better observing system. Two Lagrangian descriptors, Mendoza and Mancho's M function and Finite-Time Lyapunov Exponents, were implemented and validated in the two point vortex case. Then we considered the case of a chaotic three vortex system, empirically determining manifolds in the flow field which partition qualitatively different flow behavior. Using these manifolds, we performed a second study of the filters with the three chaotic vortices and a single drifter.

2 Background

2.1 Geophysical Flows and Lagrangian Transport

The motion of fluids across the Earth's surface impacts weather, climate, trade, and many other systems of practical interest and concern. Typically, the study of such flows has considered the velocity fields of the flows themselves (e.g. consider the famous Navier-Stokes equation, the solution of which is a flow velocity). The study of the velocity field is called the *Eulerian approach*. Recently, a complementary approach which explicitly considers the motion of particles within the flow has been developed, the *Lagrangian approach* [18]. This approach considers the motion of particles within the framework of dynamical systems theory. The analogy becomes obvious when we consider that dynamical systems theory studies the trajectories of state vectors in a velocity field 'induced' by differential equations. Tools from dynamical systems theory may be directly applied to the study of Lagrangian transport. The overarching goal of this project is to use a dynamical systems-informed approach to data assimilation to develop methods for observing Lagrangian flows.

2.2 Data Assimilation

Data assimilation is an iterative process for optimally determining the true state of an indirectly observed dynamical system. The process occurs in two stages. First, given some initial state, or a probability distribution on the initial state, a model for the system is integrated forward in time. This step is called *forecasting*. Second, at discrete times, observations (frequently indirect) of the system's state are made available. The goal is to combine these observations with the current prediction of the system's state from the forecast in some optimal way. This step is called *analysis* or *filtering*.

More formally, denote the state of our system by the n-vector $\mathbf{x}^t$. Then the evolution of this state forward in time is governed by a stochastic differential equation

$$d\mathbf{x}^t = M(\mathbf{x}^t, t)dt + d\boldsymbol{\eta}^t. \tag{1}$$

Here, $M(\cdot)$ is a possibly nonlinear operator on the state, and $\boldsymbol{\eta}^t$ is a Brownian motion with covariance matrix $\mathbf{Q}$. As the system evolves forward in time, we obtain noisy observations $\mathbf{y}_k^o \triangleq \mathbf{y}^o(t_k)$ of the system via

$$\mathbf{y}_k^o = h(\mathbf{x}^t(t_k)) + \boldsymbol{\epsilon}_k^t \tag{2}$$

where $h_k(\cdot)$ is a time-dependent, possibly nonlinear observation function of the systems true state and we take

$$\boldsymbol{\epsilon}^{\boldsymbol{t}} \sim \mathrm{N}(\mathbf{0}, \mathbf{R}^t). \tag{3}$$

That is, the observational noise is assumed to be multivariate normal with mean vector $\mathbf{0}$ and covariance matrix $\mathbf{R}^t$.

Now that we have probabilistic models for the system dynamics and observation process, we may perform the forecasting and analysis stages of data assimilation. Let $\mathbf{x}^f(t_0)$ denote our prediction of the state at some initial time t_0. We wish to evolve this state forward in time. Typically this is done by either deterministic or stochastic numerical solutions of the (assumed) true dynamics (1) with an initial condition based on assumptions about the initial data (to start) or the analysis state (after observations). Once an observation becomes available, we use our observational model to update the state estimate. Through this iterative process of forecasting and analysis, we hope to devise a scheme to combine $\mathbf{x}^f(t_k)$ and $\mathbf{y}_k^o$ in an optimal way to obtain an analyzed state $\mathbf{x}^a(t_k)$ that, ideally, more closely matches the true state $\mathbf{x}^t(t_k)$ than either $\mathbf{x}^f(t_k)$ or $\mathbf{y}_k^o$ alone. This probabilistic approach also allows us to maintain information about the uncertainty in both $\mathbf{x}^f$ and $\mathbf{x}^a$. We will outline several such approaches in Section 3.2.

In the context of Lagrangian data assimilation, we consider $\mathbf{x}^t$ to be the concatenation of the state of the flow, which we will denote by $\mathbf{x}_F^t$, and the drifters, which we will denote by $\mathbf{x}_D^t$ [8]. Thus, the overall state of the system is given by

$$\mathbf{x}^t = \begin{pmatrix} \mathbf{x}_F^t \\ \mathbf{x}_D^f \end{pmatrix}. \tag{4}$$

We consider the drifters to be passive, and thus they do not affect the motion of the flow, and we may consider the flow dynamics to evolve according to (1), with $\mathbf{x}^t$ replaced by $\mathbf{x}_F^t$. Drifter are advected by the induced velocity field of the flow, and thus the evolution of their states is also governed by (1). A key feature of Lagrangian data assimilation is that the covariance matrix for $\mathbf{x}^t$ has the form

$$\mathbf{P}^t = E[(\mathbf{x} - \mathbf{x}^t)(\mathbf{x} - \mathbf{x}^t)^T] = E\begin{bmatrix} (\mathbf{x}_F - \mathbf{x}_F^t)(\mathbf{x}_F - \mathbf{x}_F^t)^T & (\mathbf{x}_F - \mathbf{x}_F^t)(\mathbf{x}_D - \mathbf{x}_D^t)^T \\ (\mathbf{x}_D - \mathbf{x}_D^t)(\mathbf{x}_F - \mathbf{x}_F^t)^T & (\mathbf{x}_D - \mathbf{x}_D^t)(\mathbf{x}_D - \mathbf{x}_D^t)^T \end{bmatrix}. \tag{5}$$

Thus, the covariance matrix captures correlations between the flow state and the drifters' states. Without these correlations, the instantaneous position of the drifters would offer no information concerning the state of the flow, even though their states are a direct consequence of the flow state.

2.3 Observing System Design

Up to this point, we have assumed that the initial positioning of the drifter governed by (9) was made arbitrarily. That is, we have not considered how the convergence of the state estimation might be optimized by the placement of the drifter. This is an important problem in oceanographic data assimilation, since measuring instruments are costly to deploy and obtaining measurements from such instruments requires great effort [2]. Under such circumstances, one might wish to choose an optimal placement of the drifter, given some constraints. We wish to design an optimal observing system, and thus this field of study is called observing system design. Early results have demonstrated that in the case of flow assimilation, Lagrangian methods are superior to Eulerian methods [11]. We wish to apply the tools of dynamical systems, especially the concept of a manifold, to optimally decide on the placement of our observing system, a passive drifter.

3 Approach

3.1 Model System

Over the course of this study, we will consider the deterministic point-vortex model with N_v vortices and N_d drifter [8]. The positions of the j^{th} vortex $\mathbf{z}_j = (x_j, y_j)$ and k^{th} drifter $\boldsymbol{\zeta}_k = (\xi_k, \eta_k)$ in the plane are

governed by the system of ordinary differential equations

$$\frac{dx_j}{dt} = -\frac{1}{2\pi}\sum_{j'=1,j'\neq j}^{N_v}\frac{\Gamma_{j'}(y_j - y_{j'})}{l_{jj'}^2}, \qquad j = 1,\ldots,N_v \tag{6}$$

$$\frac{dy_j}{dt} = \frac{1}{2\pi}\sum_{j'=1,j'\neq j}^{N_v}\frac{\Gamma_{j'}(x_j - x_{j'})}{l_{jj'}^2}, \qquad j = 1,\ldots,N_v \tag{7}$$

$$\frac{d\xi_k}{dt} = -\frac{1}{2\pi}\sum_{j=1}^{N_v}\frac{\Gamma_j(\eta_k - y_j)}{l_{kj}^2}, \qquad k = 1,\ldots,N_d \tag{8}$$

$$\frac{d\eta_k}{dt} = \frac{1}{2\pi}\sum_{j=1}^{N_v}\frac{\Gamma_j(\xi_k - x_j)}{l_{kj}^2}, \qquad k = 1,\ldots,N_d \tag{9}$$

where l_{ij}^2 is the square of the Euclidean distance from a point at i to a point at j. We will focus on the case where the vorticities, Γ_j, are taken to be fixed and equal to 2π. We will also fix the initial conditions of the vortices at $z_j(0) = z_j$.

In the context of the data assimilation problem formulated above, we have that the evolution of our system is governed by the SDE

$$d\mathbf{x}^t = M(\mathbf{x}^t, t)dt + d\boldsymbol{\eta}^t \tag{10}$$

where

$$\mathbf{x}^t = \begin{pmatrix}\mathbf{z}\\ \boldsymbol{\zeta}\end{pmatrix} \tag{11}$$

and the deterministic evolution of the system, M, is given by the right-hand sides of (6)-(9).

3.2 Phase I - Lagrangian Data Assimilation

3.2.1 Simulation of Realizations of the Stochastic Differential Equation

This presentation of material closely follows an eminently readable account of stochastic differential equations given by Higham in *The SIAM Review* [6]. Recall the system of stochastic ODEs governing the true dynamics of the point-vortex and drifter system given by (1). We wish to generate a *realization* of the solution to this stochastic differential equation, much as one might desire to generate a realization of a random variable distributed according to some probability distribution. That is, we do not wish to find the *solution* to the system of SDEs, which itself would be a probability distribution function, but instead desire a single path. We will consider the scalar case, but the results easily generalize to systems of SDEs such as (1).

In particular, consider a generic scalar SDE given by

$$dX(t) = f(X(t))dt + g(X(t))dW(t), \qquad X(0) = X_0, \qquad 0 \leq t \leq T. \tag{12}$$

In this case, $f(\cdot)$ corresponds to the deterministic contribution to the SDE and $g(\cdot)$ corresponds to the stochastic contribution. $dW(t)$ corresponds to an increment of the Brownian motion. A Brownian motion $W(t,\omega)$[1] is a stochastic process. That is, for any fixed t, $W(\cdot,\omega)$ is a random variable and for any fixed ω, $W(t,\cdot)$ is a function. A Brownian motion is characterized by the following three properties:

1. $W(0) = 0$ with probability 1.
2. For $0 \leq s < t \leq T$ the random variable given by the increment $W(t) - W(s)$ is normally distributed with mean zero and variance $t - s$.

[1]For brevity, we will omit the dependence on ω in what follows.

3. For $0 \leq s \leq t \leq u \leq v \leq T$, the increments $W(t) - W(s)$ and $W(v) - W(u)$ are independent.

Since it is not possible to deal with continuous quantities on a digital computer, we will be interested in *discretized* Brownian paths. We will thus approximate $W(t)$ at various times t_j. We may then denote, without any confusion, the value of the discretized Brownian motion at any discrete time t_j by W_j. We will discretize the interval $[0, T]$ into $N + 1$ subintervals giving a discretized step size of $\Delta t = T/N$. Thus, if we begin indexing t_j at $j = 0$, we find that $t_j = j\Delta t$. We may then use the definition of a Brownian motion to construct our discretized Browian path. In particular, condition 1 states that $W(t_0) = W(0) = 0$. Conditions 2 and 3 tell us that the Brownian increments $W_j - W_{j-1}$ are independent, identically distributed random variables from $N(0, \Delta t)$. Thus, we can construct the discretized Brownian path by computing

$$W_j = W_{j-1} + dW_j, \qquad j = 1, 2, \ldots, N, \tag{13}$$

where, as we have stated, dW_j is distributed according to $N(0, \Delta t)$ and $W_0 = 0$.

We have so far made one approximation in developing a method for computing a realization of the SDE (12), i.e. discretizing the Brownian motion. We will also discretize in time. First, note that the solution of (12) can be written

$$X(\tau_j) = X(\tau_{j-1}) + \int_{\tau_{j-1}}^{\tau_j} f(X(s))\, ds + \int_{\tau_{j-1}}^{\tau_j} g(X(s))\, dW(s), \tag{14}$$

where the second integral can be taken to be either an Itô or Stratonovich integral. We will take it to be the Itô integral. If we approximate the integrals in the most obvious way, we arrive at the difference equation

$$X_j = X_{j-1} + f(X_{j-1})\Delta\tau + g(X_{j-1})(W(\tau_j) - W(\tau_{j-1})), \qquad j = 1, 2, \ldots L, \tag{15}$$

where we have discretized the time interval into $L + 1$ subintervals of length $\Delta\tau = T/L$. This gives the Euler-Marayuma method for solving for a single path of the SDE. Note that in the case where $g(X) \equiv 0$, (15) reduces to the well-known Euler method for numerically solving deterministic IVPs.

As with deterministic ODE solvers, we are interested in the accuracy of our numerical method. Keeping in mind that the true solution $X(\tau_n)$ and the numerical solution X_n at any time τ_n are random variables, we must define accuracy in probabilistic terms. We say that a method has *strong order of convergence* equal to γ if there exists a constant C such that

$$\mathbb{E}|X_n - X(\tau)| \leq C\Delta t^{\gamma} \tag{16}$$

for any fixed $\tau = n\Delta t \in [0, T]$ and Δt sufficiently small. It can be shown that Euler-Marayuma has a strong order of convergence of $1/2$. Again, as in the deterministic case, we may seek higher order methods. One such method is the stochastic analog to the deterministic fourth-order Runge Kutta [21]. As with the deterministic method, the stochastic version is derived via the Taylor expansion of the solution to the SDE. For a given time discretization $\Delta\tau$ and discretized Brownian increment ΔW, define

$$h(X_j, \tau_j) = \left[f(X_j, \tau_j) - \frac{1}{2} g(X_j, \tau_j) \frac{\partial g(X_j, \tau_j)}{\partial X_j} \right] \Delta\tau + g(X_j, \tau_j)(W(\tau_j) - W(\tau_{j-1})). \tag{17}$$

Then we may construct the following four stage method:

$$K_1 = h(X_j, \tau_j) \tag{18}$$

$$K_2 = h(X_j + \frac{1}{2}K_1, \tau_j + \frac{1}{2}\Delta\tau) \tag{19}$$

$$K_3 = h(X_j + \frac{1}{2}K_2, \tau_j + \frac{1}{2}\Delta\tau) \tag{20}$$

$$K_4 = h(X_j + K_3, \tau_j + \Delta\tau) \tag{21}$$

$$X_{j+1} = X_j + \frac{1}{6}(K_1 + 2K_2 + 2K_3 + K_4) \tag{22}$$

This method has strong order of convergence equal to 2.

As stated, this method easily generalizes to the system of SDEs given by (1). In the generic notation, our system takes the form

$$d\mathbf{X} = \mathbf{f}(\mathbf{X}, t)dt + \mathbf{g}(\mathbf{X}, t)d\mathbf{W}(t) \tag{23}$$

where

$$\mathbf{f}(\mathbf{X}, t) = M(\mathbf{X}, t) \tag{24}$$

$$\mathbf{g}(\mathbf{X}, t) = \sqrt{2}\mathbf{Q}^{1/2} \tag{25}$$

given a covariance matrix $\mathbf{Q}$ for the dynamical noise. In this study we will assume that the dynamical noise is uncorrelated and the same magnitude across all states. Thus, we take

$$\mathbf{Q} = 2\sigma^2\mathbf{I}. \tag{26}$$

In all simulations of (1), the Brownian motion discretization was $\Delta t = 2.5 \times 10^{-3}$ and the time discretization was $\Delta\tau = 5 \times 10^{-3}$.

3.2.2 Extended Kalman Filter

The data assimilation problem outlined in Section 2.2 with linear dynamics, linear observations, and Gaussian noise has an exact solution [9]. The stochastic process which solves (1) in this case is Gaussian, and can be fully characterized by its mean $\mathbf{x}^t$ and covariance matrix $\mathbf{P}^t$. By directly evolving both the mean and covariance matrix forward in time and performing a minimum mean-square estimate of $\mathbf{x}^a$, the optimal filter can be derived. This filter was first proposed by Rudolf Kalman, and as such is named the Kalman filter. An extension of the Kalman filter to deal with nonlinearities in both the dynamical equations and the observation equation is appropriately called the extended Kalman filter (EKF). This modification to the Kalman filter uses a tangent-linear model for the system dynamics and measurement operator, centered at the forecasted value $\mathbf{x}^f$. Thus, we denote the Jacobians of the evolution operator in (1) and the observation operator in (2) by

$$\mathbf{M}(t) = J[M(\mathbf{x}, t)]\big|_{\mathbf{x}=\mathbf{x}^f} \tag{27}$$

$$\mathbf{H}_k = J[h(\mathbf{x}, t_k)]\big|_{\mathbf{x}=\mathbf{x}^f}. \tag{28}$$

During the forecasting stage, we evolve forecasts for $\mathbf{x}^t$ and $\mathbf{P}^t$ forward in time by

$$\frac{d}{dt}\mathbf{x}^f = M(\mathbf{x}^f, t) \tag{29}$$

$$\frac{d}{dt}\mathbf{P}^f = \mathbf{M}(t)\mathbf{P}^f + \mathbf{P}^f\mathbf{M}^T(t) + \mathbf{Q}, \tag{30}$$

where the superscript T denotes a transpose. As an observation becomes available at time t_k, we assimilate this observation to obtain the analyzed estimates $\mathbf{x}^a$ and $\mathbf{P}^a$ using

$$\mathbf{x}_k^a = \mathbf{x}^f(t_k) + \mathbf{K}_k(\mathbf{y}_k^o - h_k(\mathbf{x}^f(t_k))) \tag{31}$$

$$\mathbf{P}_k^a = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}^f(t_k) \tag{32}$$

where $\mathbf{K}_k$ is the Kalman gain and is given by

$$\mathbf{K}_k = \mathbf{P}^f(t_k)\mathbf{H}_k^T(\mathbf{H}_k\mathbf{P}^f(t_k)\mathbf{H}_k^T + \mathbf{R}_k^o)^{-1}. \tag{33}$$

This analysis estimate is then used as the initial condition for the forecast equations, and the next iterate of the data assimilation process begins.

It should be noted that we can take advantage of properties of the flow induced by (6)-(9) to simplify our computation of the Jacobian matrix $\mathbf{M}(t)$. In particular, the flow induced by the point vortex system is both incompressible and irrotational (except at the point vortices) [1]. Recall that a velocity field $\mathbf{u}$ is incompressible if div $\mathbf{u} = 0$ and irrotational if curl $\mathbf{u} = 0$. Taking $\mathbf{u} = (\frac{dx}{dt}, \frac{dy}{dt})$, we see that

$$\text{div } \mathbf{u} = 0 \implies \frac{\partial}{\partial x}\frac{dx}{dt} = -\frac{\partial}{\partial y}\frac{dy}{dt} \tag{34}$$

$$\text{curl } \mathbf{u} = 0 \implies \frac{\partial}{\partial x}\frac{dy}{dt} = \frac{\partial}{\partial y}\frac{dx}{dt}. \tag{35}$$

Therefore, for a given block of the Jacobian matrix, we need only compute half of the entries. Similarly, when computing both the Jacobian matrix and the right-hand side of (6)-(9), we may take advantage of the fact that the vortex-vortex interactions are equal and opposite. Thus, for any given pair of vortices or a vortex-drifter pair, we need only compute half of the interactions. This significantly reduces the amount of computation necessary to numerically evolve the point-vortex model.

The EKF has many deficiencies. As just described, it only weakly assumes nonlinearity in the governing dynamical equation (1) by using the tangent-linear model. It also assumes that the posterior estimate of the state is Gaussian: this is no longer guaranteed in the nonlinear case. Both of these violations of the assumptions of the Kalman filter result in a suboptimal filter that may diverge depending on the dynamics of the system under consideration.

3.2.3 Local Ensemble Transform Kalman Filter

Another approach to data assimilation that addresses some weaknesses inherent in the EKF is the ensemble Kalman filter (EnKF) [4]. The EnKF represents the probability distribution function describing the state $\mathbf{x}^t$ by an *ensemble* of states $\{\mathbf{x}_i^f\}_{i=1}^N$. That is, we may use this ensemble to construct an empirical distribution function which may be used to compute (approximate) moments of interest.

Let $\bar{\mathbf{x}}^f$ be the ensemble average given by

$$\bar{\mathbf{x}}^f = \frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_i^f. \tag{36}$$

Then the ensemble covariance matrix $\mathbf{P}_e^f$ may be computed by

$$\mathbf{P}_e^f = \frac{1}{N-1}\sum_{i=1}^{N}(\mathbf{x}_i^f - \bar{\mathbf{x}}^f)(\mathbf{x}_i^f - \bar{\mathbf{x}}^f)^T. \tag{37}$$

Thus, we may evolve the ensemble states forward in time using (1) and then compute the the ensemble average and covariance matrices. These now approximate the *true* mean value and covariance matrix, without any linearity assumptions, and the only approximation is in the ensemble size. Once an observation is obtained, there are two approaches to performing the analysis step to update the ensemble. We focus below on the deterministic approach, in particular on the Local Ensemble Transform Kalman Filter.

The Local Ensemble Transform Kalman Filter (LETKF) is a type of square root filter that allows for the deterministic generation of the analysis ensemble given an observation and the forecast ensemble [20].

We define the ensemble spread matrix for the forecast as

$$\mathbf{X}^f = \begin{bmatrix} \mathbf{x}_1^f - \bar{\mathbf{x}} & \mathbf{x}_2^f - \bar{\mathbf{x}} & \dots & \mathbf{x}_n^f - \bar{\mathbf{x}} \end{bmatrix}. \tag{38}$$

Thus we see that

$$\frac{1}{N-1}\mathbf{X}^f(\mathbf{X}^f)^T = \mathbf{P}_e^f \tag{39}$$

The matrix $\mathbf{X}^a$ is defined in a similar fashion. We form the analysis ensemble from the forecast ensemble by computing

$$\mathbf{X}^a = \mathbf{X}^f \mathbf{W} \tag{40}$$

where $\mathbf{W}$ is an unknown matrix containing weighting coefficients for the columns of $\mathbf{X}^f$. We seek $\mathbf{W}$ such that

$$\mathbf{P}_e^a = \frac{1}{N-1}\mathbf{X}^a(\mathbf{X}^a)^T \tag{41}$$

$$= \frac{1}{N-1}\mathbf{X}^f\mathbf{W}(\mathbf{X}^f\mathbf{W})^T \tag{42}$$

$$= \frac{1}{N-1}\mathbf{X}^f\mathbf{W}\mathbf{W}^T(\mathbf{X}^f)^T. \tag{43}$$

Let $\mathbf{D} = \mathbf{W}\mathbf{W}^T$. Then we call $\mathbf{W}$ the *matrix square root* of $\mathbf{D}$. The LETKF uses

$$\mathbf{D} = (\mathbf{I} + (\mathbf{X}^f)^T\mathbf{H}^T(\mathbf{R}^o)^{-1}\mathbf{H}\mathbf{X}^f)^{-1}. \tag{44}$$

This choice of $\mathbf{D}$ is convenient because it allows for the computation of the new ensemble mean by

$$\bar{\mathbf{x}}^a = \bar{\mathbf{x}}^f + \mathbf{X}^f\mathbf{D}(\mathbf{X}^f)^T\mathbf{H}^T(\mathbf{R}^\mathbf{o})^{-\mathbf{1}}(\mathbf{y}^o - \mathbf{H}\bar{\mathbf{x}}^f). \tag{45}$$

Finally, we may determine $\mathbf{W}$ by computing the eigenvector decomposition of $\mathbf{D}$

$$\mathbf{D} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T \tag{46}$$

and forming

$$\mathbf{W} = \dot{\mathbf{U}}\dot{\mathbf{\Lambda}}^{1/2}\dot{\mathbf{U}}^T \tag{47}$$

where $\dot{\mathbf{U}}$ contains the eigenvectors of $\mathbf{D}$ corresponding to nonzero eigenvalues and $\dot{\mathbf{\Lambda}}^{1/2}$ contains the square root of the nonzero eigenvalues of $\mathbf{D}$. Note that in the ensemble sizes we consider, all eigenvalues will generally be non-zero. Then we may form the analysis ensemble by applying (40). Note that this choice of $\mathbf{W}$ is *not* unique.

The description of the LETKF so far did not include a localization step. For spatiotemporally chaotic systems, spatial localization in the analysis step is key to any ensemble-based filter. Recall that we are approximating the true distribution of the stochastic process described by (1) using a finite number N of ensemble members. In this case, the forecast sample covariance matrix $\mathbf{P}_e^f$ contains information only about the subspace spanned by the N ensemble members. Especially in the case where the dimension of the system is much larger than the number of ensemble members, this results in an inability of the sample covariance matrix to accurately approximate the true covariance matrix of the stochastic process. However, it has been found that locally such systems may behave like low-dimensional systems with nonautonomous forcing from non-local states [15].

The localization procedure is a straightforward adaptation of the ETKF procedure described above. At the analysis step, instead of updating all states simultaneously, the states are updated one-by-one, only including in each states analysis those states within a certain spatial window. See [7] for a detailed explanation of the localization procedure.

For localization with the point-vortex model, we will always update the complete position (ξ_k, η_k) of a drifter with its own observation. We will update the complete position (x_j, y_j) of a given vortex with the observation from drifter k, weighting the observation by a Gaussian windowing function. This weighting is achieved by scaling the entries of the observational covariance matrix $\mathbf{R}^o$ corresponding to drifter k. That is, we compute the scaling factor by

$$W((x_j, y_j), (\xi_k^o, \eta_k^o); \beta) = \exp\left[-\frac{1}{2\beta^2}\left\{(x_j - \xi_k^o)^2 + (y_j - \nu_k^o)^2\right\}\right], \tag{48}$$

and then multiply the elements of $\mathbf{R}^o$ corresponding to drifter k by this factor. In (48), β is a bandwidth parameter that may be determined experimentally. Thus, we see that a drifter observation is given full weight if it coincides with the vortex position, and the weight contributed by this observation decays to zero as the distance between the vortex and observation increases. Thus, our procedure differs from [7] in the choice of the windowing function. Their localization scheme uses a boxcar window to determine which states to include in the local analysis step for an analyzed state. That is, they take as their windowing function

$$W'((x_j, y_j), (\xi_k^o, \eta_k^o); \gamma) = \begin{cases} 1 & : \sqrt{(x_j - \xi_k^o)^2 + (y_j - \nu_k^o)^2} < \gamma \\ 0 & : \text{otherwise} \end{cases} . \tag{49}$$

A smooth windowing function is preferred due to the non-differentiability of the boxcar window at the cutoff circle determined by γ.

Another known issue with the LETKF and filters in general is ensemble deflation [4]. The analysis step tends to underestimate the covariance matrix. Thus, an additional modification is made to the LETKF procedure outlined above. The analysis covariance matrix is inflated by multiplicative factor at the end of each analysis step. That is, at each analysis step we perform

$$\mathbf{P}_e^a := \lambda \mathbf{P}_e^a \tag{50}$$

where λ is a small multiplicative slightly greater than 1.

While the EnKF has several advantages over the EKF, it still has several weaknesses. The forecasting stage preserves the full non-linearity and non-Gaussianity (up to the ensemble approximation) of (1), but the analysis step still assumes that the posterior distribution of the state is Gaussian. As stated above, this is not necessarily the case for a nonlinear system. The EnKF also requires the evolution of ensembles forward in time: thus, whereas the EKF only requires the evolution of a single state mean and covariance matrix, the EnKF requires that we integrate N ensembles forward in time, where N might need to be relatively large to obtain good approximations to the mean and covariance matrix of the process.

3.2.4 Particle Filter

The particle filter closely resembles the EnKF with the key advantage that it performs a full Bayesian update at the analysis step [17, 19]. Like the EnKF, the particle filter evolves an ensemble of states forward in time by using (1). Unlike the EnKF, at the analysis step, instead of generating a new analysis ensemble, the particle filter updates weights associated with each of the ensemble members by applying Bayes's rule. Recall that from (2), the observations are modeled as Gaussian random vectors. Thus, the likelihood of an observation is given by

$$p(\mathbf{y}_k^o | \mathbf{x}(t_k)) = C_1 \exp\left(-\frac{1}{2} (\mathbf{y}_k^o - h(\mathbf{x}(t_k)))^T (\mathbf{R}^o)^{-1} (\mathbf{y}_k^o - h(\mathbf{x}(t_k))) \right) \tag{51}$$

where C_1 is a normalization constant. Let $w_{i,k}$ be the weight of particle $\mathbf{x}_i^f$ at time t_k. At the start of data assimilation, we have no information about the system and thus initialize all the $w_{0,k}$ to equal values, ensuring that they sum to unity. That is, we assume a discrete uniform distribution. As an observation is made at time t_k, we update the weights by applying Bayes's theorem, given by (72). In this case, Bayes's theorem takes the form of

$$w_{i,k} = C_2 w_{i,k-1} p(\mathbf{y}_k^o | \mathbf{x}_i^f(t_k))) \tag{52}$$

where C_2 is again a normalization constant. At any analysis stage, we may recover the moments of the state of the system by applying the proper weighting of the states. For example, we recover the mean state by

$$\bar{\mathbf{x}}^f(t_k) = \sum_{i=1}^{N} w_{i,k} \mathbf{x}_i^f(t_k). \tag{53}$$

While the formulation presented above seems straightforward to implement, special attention must be paid to the weight of the particles. It is well known that in high dimensions, most of the weights become concentrated on a small number of particles, thus reducing the effective number of particles [3]. When the effective number of particles drops below a certain threshold, we wish to resample the particles to obtain an ensemble with a larger effective number of particles. This resampling step is key to the proper functioning of the particle filter. There are various methods for doing this. We implemented residual resampling [12].

Residual resampling occurs after the analysis step. The updated weights $w_{i,k}$ are taken as the parameters of a multinomial distribution with N possible outcomes. Thus, the weight $w_{i,k}$ corresponds to the multinomial parameter for state $\mathbf{x}_i^a(t_k)$. A new collection of N particles is chosen by sampling from this multinomial distribution, for instance by using `mnrnd` in MATLAB. This amounts to choosing the N resampled particles with replacement from the original particles where the probability of choosing any given particle $\mathbf{x}_i^a(t_k)$ is equal to its weight $w_{i,k}$[2]. The resampled particles $\hat{\mathbf{x}}_i^a(t_k)$ are then used in the next forecast step, with all of the resampled particle weights $\hat{w}_{i,k}$ set to $\frac{1}{N}$ to maintain approximately the same empirical distribution as the original collection of particles.

3.3 Phase II - Manifold Detection

Invariant manifolds play a key role in the description of Lagrangian flows, just as they do in the description of dynamical systems [18]. For flows, the streamlines of a system are invariant manifolds since a drifter trajectory starting on that streamline will remain on that streamline. This is similar to eigenpair solutions to a linear differential equation. If a trajectory begins on an eigenpair solution of the system, it must remain on that solution for all time. Similarly, due to the uniqueness of solutions, no trajectory may cross an invariant manifold. Thus, the invariant manifolds of a Lagrangian flow partition the flow into different types of behavior. By identifying these manifolds (and thus these partitions), we may be able to take advantage of the flow behavior within a given region to better perform data assimilation for the system.

3.3.1 Mendoza and Mancho's M Function

We focus on the deterministic point-vortex system (6)-(9). We may imagine we have a drifter at each point on a grid in a region of the plane. In this case, we may define the following Lagrangian descriptor [13]

$$M(\mathbf{x}_D^t, t^*) = \int_{t^*-\tau}^{t^*+\tau} \left(\sum_{i=1}^{n} \left(\frac{dx_D^i(t)}{dt} \right)^2 \right)^{1/2} dt \tag{54}$$

where x_D^i represents the i^{th} position component of the drifter. For our system, the drifter moves on the plane, and thus we have $n = 2$. In this case, M measures the Euclidean arc length of the curve passing through $\mathbf{x}_D^t$ at time t. Clearly, M depends on our choice of τ and t^*. For a time independent flow, M provides a time-independent partition of the phase space of a dynamical system. However, M may also be used for time-dependent flows, as will be of interest to us. In this case, it is important to recognize the dependence of M on t^*.

The philosophy behind M is straightforward: regions of the flow field that correspond to coherent structures will travel together. Thus, they will have similar arc lengths. By inspecting regions where sharp differences in M occur, we may identify boundaries that separate qualitatively distinct behaviors. Such boundaries are known as Lagrangian coherent structures, or LCSs.

3.3.2 Finite-Time Lyapunov Exponents

An alternative method for determining manifolds in a Lagrangian flow is finite-time Lyapunov exponents (FTLEs). The Lyapunov exponents of a dynamical system characterize how quickly infinitesimally nearby trajectories diverge in phase space [14]. More formally, consider an N-dimensional flow. Let $\mathbf{x}_1(t_0)$ be an

[2]In the case of resampling after every analysis step, this procedure reduces to biasing the collection of particles towards those that were closest to the observation, since these particles will have the largest posterior weight after the analysis step.

initial condition, and consider an infinitesimal perturbation of this trajectory $\mathbf{x}_2(t_0) = \mathbf{x}_1(t_0) + \mathbf{y}(t_0)$. We may track how the perturbation $\mathbf{y}(t)$ evolves over time. In particular, we may consider a time average of the logarithm of the absolute growth of $\mathbf{y}(t)$. In the limit as $t \to \infty$, we recover a Lyapunov exponent for the system,

$$\lambda = \lim_{t\to\infty} \frac{1}{t} \log \frac{||\mathbf{y}(t)||}{||\mathbf{y}(0)||}. \tag{55}$$

An N-dimensional dynamical system has N Lyapunov exponents (known as its Lyapunov spectrum), and the Lyapunov exponent we recover in the limit depends on the orientation of the infinitesimal perturbation $\mathbf{y}(t_0)$. Generally, for a random perturbation direction, the Lyapunov exponent obtained will be the largest positive Lyapunov exponent.

If we consider (55) without the infinite limit, we recover a finite-time Lyapunov exponent for the system, given by

$$\lambda(T) = \frac{1}{T} \log \frac{||\mathbf{y}(T)||}{||\mathbf{y}(0)||}. \tag{56}$$

Since analytically evaluating (55) is intractable in all but the simplest of cases, computing finite-time approximations of this infinite limit is the best that can be done.

FTLEs are useful in their own right, however. We now sketch the rational behind their use in detecting structure in flow fields. Consider the case of a system of two linear, autonomous ordinary differential equation that give rise to hyperbolic trajectories. In this case, each eigenpair solution lies along a ray determined by the eigenvector corresponding to the matrix from the system of differential equations. In the case of a hyperbolic system, the eigenvalues will be opposite in sign, and thus one of the rays will correspond to a **stable** solution that approaches the origin in the limit of infinite time, and the other will correspond to an **unstable** solution that diverges from the origin. Recall that in the hyperbolic case, any initial conditions not on the stable eigensolution will diverge from the origin. Consider placing two initial conditions extremely close to each other, but on either side of the stable eigensolution. In this case, the solutions will diverge in time. Thus, if we compute the finite-time Lyapunov exponent of such a perturbed trajectory, we will find that it has a large positive value. In fact, in phase space, trajectories near the stable manifold will have maximal FTLEs compared to other regions. Thus, stable manifolds of the system correspond to ridges of the FTLE field [5]. To identify unstable manifolds, we may consider the same argument, but in reverse time. That is, we place two trajectories near each other on either side of an unstable manifold. Going backwards in time, they will diverge from each other, and thus have a large backward-time finite-time Lyapunov exponent. By determining where these forward/backward-time ridges occur, we may identify the stable manifolds of the system, and thus identify qualitatively distinct regions of the phase space.

4 Implementation

All algorithms were developed in MATLAB on a MacBook Pro with a 2.4 GHz Intel Core 2 Duo processor with 4 GB of RAM. The algorithms were initially developed to run serially. The code for evolving the particles in the LETKF and particle filter were ported to run in parallel using MATLAB's Parallel Computing Toolbox. Similarly, the manifold detection algorithm was initially developed serially and then parallelized.

5 Databases

During Phase I, the databases consist of archived numerical solutions to (1) with varied initial conditions for the positions of the drifters. These solutions were generated using a stochastic Runge-Kutta second-order SDE solver with prescribed levels of dynamical noise [21]. Observational noise was added to the locations of the drifters according to (2), again following prescribed levels of noise (see Section 6 for more details).

These common databases were used for all assimilation methods, allowing comparisons between and within methods.

Phase II does not require databases. The computation of M values is completely model dependent and does not require model-independent inputs.

6 Validation

6.1 Phase I - Lagrangian Data Assimilation

6.1.1 Determination of the Time Step

The point-vortex system has several conserved quantities that may be used to determine a suitable step size for the Runge-Kutta integration scheme. For instance, the Hamiltonian for the two point vortex is given by

$$H = -\frac{1}{4\pi} \sum_{\substack{i,j \\ i \neq j}} \Gamma_i \Gamma_j \log l_{ij} \tag{57}$$

where Γ_i is the vorticity of vortex i and l_{ij} is the Euclidean distance between vortex i and j [1]. Other conserved quantities include the linear impulse and the angular momentum. By computing these conserved quantities over time, we determined that a step size of $\Delta t = 0.005$ is sufficiently small to ensure their conservation within numerical error. Figure 1 shows the Hamiltonian of a two vortex system with one vortex at $(1, 0)$ and other other vortex at $(-1, 0)$ using a time step of $\Delta t = 0.005$. We see that the Hamiltonian remains constant within numerical error over the 60 second integration interval.

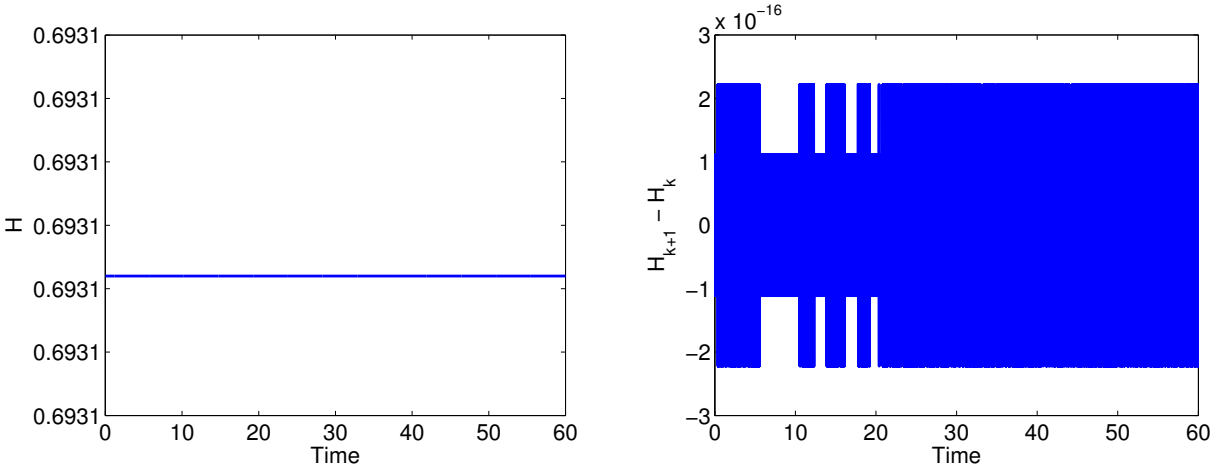


Figure 1: The Hamiltonian (left) and difference in Hamiltonian over time (right) of a two vortex system with one vortex at $(1, 0)$ and other other vortex at $(-1, 0)$ using $\Delta t = 0.005$.

6.1.2 Validation of the Numerical SDE Solver

Consider the scalar SDE

$$dX(t) = \lambda X(t)dt + \mu X(t)dW(t) \tag{58}$$

$$X(0) = X_0. \tag{59}$$

This SDE has the solution

$$X(t) = X_0 \exp\left[\left(\lambda - \frac{1}{2}\mu^2\right)t + \mu W(t)\right]. \tag{60}$$

Thus, for any realization of the Brownian path $W(t)$, we can compare the numerical solution obtained using Euler-Marayuma and Runge-Kutta to the analytical solution given by (60). The results for Euler-Marayuma and Runge-Kutta with time discretization $\Delta\tau = 2^{-7}$ and Brownian motion discretization $\Delta t = 2^{-8}$ are shown in Figures 2, 3, and 4. As expected, the Runge-Kutta method provides much greater accuracy than Euler-Marayuma due to its higher level of strong convergence. Recall that we will use the strong second-order Runge-Kutta method for this project.

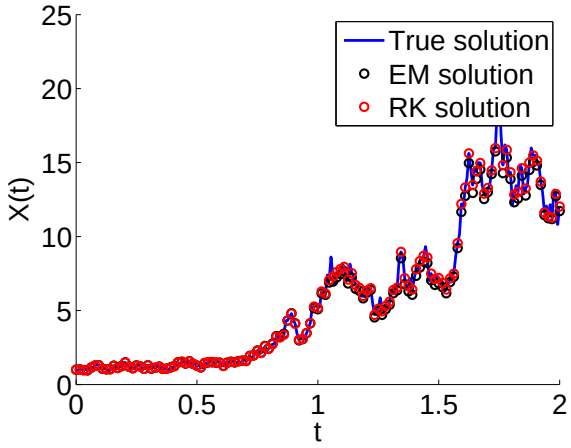


Figure 2: The true solution to (58) as well as the Euler-Marayuma and Runge-Kutta approximations.

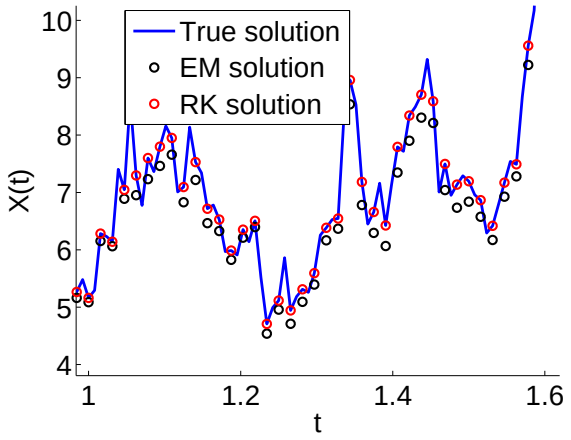


Figure 3: A magnification of a portion of Figure 2. This clearly demonstrates the greater accuracy of Runge-Kutta compared to Euler-Marayuma as expected by its better strong order of convergence.

6.1.3 Validation of the Jacobian Matrix for the Point Vortex Model

The EKF assumes that we evolve the forecast covariance matrix forward according to the a tangent linear model. This requires linearizing (6)-(9) about the current forecast state, and thus requires the computation

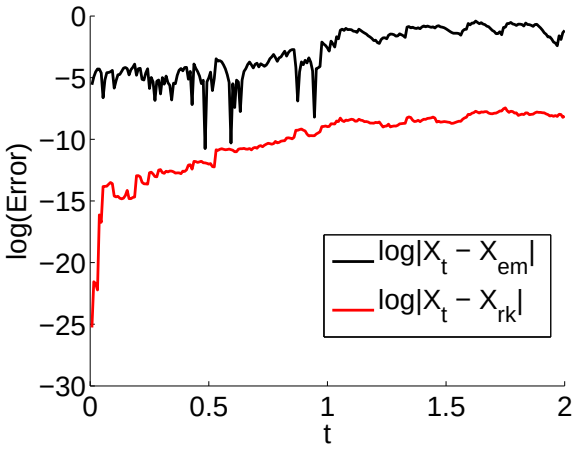


Figure 4: The log of the error between the true solution to (58) and the Euler-Marayuma (black) and Runge-Kutta (red) numerical approximations.

of the Jacobian matrix $\mathbf{M}(\mathbf{x}^f(t))$. In order to validate that the Jacobian matrix was computed correctly, we may verify that the tangent linear model holds. Consider a reference trajectory starting from $\mathbf{x}_1(0) = \mathbf{x}^0$ and a perturbed trajectory starting from $\mathbf{x}_2(0) = \mathbf{x}^0 + \mathbf{y}^0$. Thus, the evolution of the perturbed trajectory $\mathbf{x}_2$ will depend on the evolution of $\mathbf{y}$. We call the small perturbation $\mathbf{y}$ a *tangent vector* [10]. Substituting $\mathbf{x}_2$ into the deterministic differential equation we find

$$\frac{d}{dt}\mathbf{x}_2 = \frac{d}{dt}(\mathbf{x}+\mathbf{y}) = M(\mathbf{x}+\mathbf{y}) \tag{61}$$

$$= M(\mathbf{x}) + \left.\frac{\partial M(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}(t)} \mathbf{y} + \mathcal{O}(\mathbf{y}^2) \tag{62}$$

where in (62) we have linearized about the point $\mathbf{x}(t)$. Noting the linearity of differentiation and recalling that $d\mathbf{x}/dt = M(\mathbf{x})$, we thus obtain

$$\frac{d\mathbf{y}}{dt} = \left.\frac{\partial M(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}(t)} \mathbf{y} \tag{63}$$

for the evolution equation of the tangent vector. Recall that $\partial M(\mathbf{x})/\partial \mathbf{x}$ is the Jacobian of the deterministic equations evaluated at the reference trajectory $\mathbf{x}_1$ which we had previously called $\mathbf{M}$. Thus (63) reduces to

$$\frac{d\mathbf{y}}{dt} = \mathbf{M}\mathbf{y} \tag{64}$$

and we see that the evolution of the tangent vector only depends on the Jacobian of the system. There, we use the evolution equation for the tangent vector to validate the Jacobian as follows. We generate two trajectories with initial condition $\mathbf{x^0}$ and $\mathbf{x^0} + \mathbf{y^0}$ where $\mathbf{y}^0$ is a small perturbation (on the order of 10^{-10}). We then integrate these trajectories forward in time numerically according to (6)-(9). At the same time, we integrate $\mathbf{y}$ forward in time according to (64). Then we compare $\mathbf{y}$ obtained via taking the difference of the true and perturbed trajectories to $\mathbf{y}$ obtained by the tangent linear model. The results of this experiment are shown in Figure 5. As expected, for small enough initial perturbation over a small enough time window, the tangent linear model correctly predicts the evolution of the tangent vector.

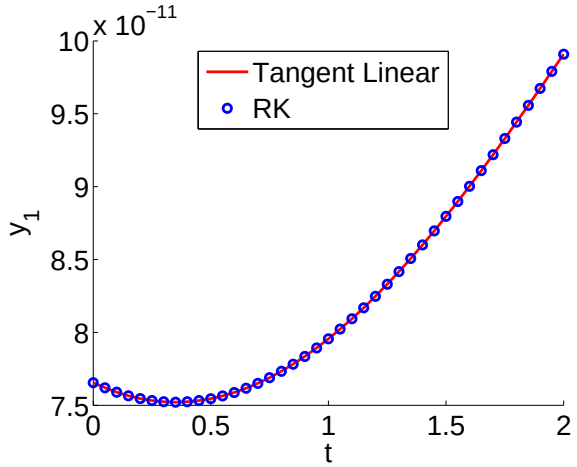


Figure 5: The evolution of the first component of the tangent linear vector $\mathbf{y}$ computed by the tangent linear model and by differencing the reference ($\mathbf{x}_1$) and perturbed ($\mathbf{x}_2$) trajectory. Note that $\mathbf{y}^0$ was chosen uniform randomly on the interval $[0, 10^{-10}]^D$ where D is the dimension of the state space.

6.1.4 Validation of the Analysis Step for the LETKF and Particle Filter

We consider the analysis portion of the LETKF and particle filter in the case of a Gaussian prior. That is, we assume the prior is a multivariate Gaussian with mean $\mathbf{x}^f(t_k)$ and covariance $\mathbf{P}^f(t_k)$. In this case, the Kalman analysis equations (31)-(33) are exact. Thus, we may assume an observational covariance $\mathbf{R}^o$ and an observation $\mathbf{y}_k^o$ which we take to be a deterministic perturbation of the prior $\mathbf{x}^f(t_k)$, and using these we may compute the exact analysis mean $\mathbf{x}_k^a$ and covariance matrix $\mathbf{P}_k^a$. Then we generate an ensemble (in the case of the LETKF) or a particle cloud (in the case of the particle filter) by sampling from a multivariate Gaussian with mean $\mathbf{x}^f(t_k)$ and covariance $\mathbf{P}^f(t_k)$. Using the same observational covariance $\mathbf{R}^o$ and observation $\mathbf{y}_k^o$, we may perform the analysis portion of these filters and compare their analysis means and covariances to the exact analysis mean and covariance. They should agree within sampling error.

To determine the prior mean and covariance matrix, we chose the mean and covariance matrix resulting from the EKF using **Position 1** with two vortices and a single drifter under conditions corresponding to **Case 3**. We used the mean and covariance matrix at the middle of the filter run (30 seconds into the data assimilation process). This ensured a covariance matrix with realistic cross-correlations.

We compare the sample means to the exact mean using the Euclidean distance between them

$$D_{\text{mean}}^{f/a} = ||\mathbf{x}^{f/a} - \mathbf{x}_e^{f/a}||_2 \tag{65}$$

and the sample covariances to the exact covariance using the Frobenius norm of their difference

$$D_{\text{cov}}^{f/a} = ||\mathbf{P}^{f/a} - \mathbf{P}_e^{f/a}||_F \tag{66}$$

where $\mathbf{x}_e$ and $\mathbf{P}_e$ are the sample means and covariances and we use the superscripted f and a to correspond to their prior / analysis value, respectively. We computed these difference values for 1000 initial ensembles, each ensemble containing 100 ensemble members / particles. The results of this study is summarized in Table 1. We include the difference between the true and sample values for the prior mean and covariance as a reminder that there is sampling error in the generation of the initial ensemble itself. Thus, we cannot expect the sample analysis values to much improve over this error. We see that the analysis mean and covariance for the LETKF is closer to the true mean and covariance matrix than the particle filter's analysis mean and covariance. This is to be expected, since the particle filter amounts to a Monte Carlo solution to the analysis problem, while the LETKF would be exact for the Gaussian prior, up to sampling error. However, for both filters, the distance between the true and sample moments remains relatively constant.

Table 1: Distances between the exact means and covariances and the sample means and covariances resulting from the LETKF and particle filters. The distance between the means was measured using the Euclidean norm and the distance between the covariances was measured using the Frobenius norm. The superscript f corresponds to the prior means / covariances and the superscript a corresponds to the analysis means and covariances.

	D_{mean}^f	D_{mean}^a	D_{cov}^f	D_{cov}^a
LETKF	0.012	0.014	0.002	0.002
PF	-	0.015	-	0.0025

6.1.5 Validation of Filters

For validation of the Lagrangian data assimilation methods, we will consider a hierarchical approach. We will begin with the (physically unrealizable) condition that the true dynamics are deterministic and the position of the vortices and drifter are known up to observational noise and estimate the positions of the drifters using the three data assimilation methods. This represents a 'best case' scenario and we expect the schemes to easily track the positions of the drifters under these conditions. Next we consider the case where the true dynamics are stochastic and the system is fully observed under measurement uncertainty. Finally, we will consider the case where the the point-vortices locations are unknown and where the dynamical noise in (1) and the observational noise in (2) take on realistic values. These cases are summarized below:

- **Case 1:** Full observation, $\sigma = 0$, $\rho = 0.02$.
- **Case 2:** Full observation, $\sigma = 0.02$, $\rho = 0.02$.
- **Case 3:** Partial observation, $\sigma = 0.02$, $\rho = 0.02$.

In all cases, we may compare the true state to the assimilated state using the distance between the two as a function of time, given by the Euclidean norm of their difference

$$\delta^{a,f}(t) = ||\mathbf{x}^t(t) - \mathbf{x}^{a,f}(t)||_2. \tag{67}$$

6.1.6 Validation of the Extended Kalman Filter

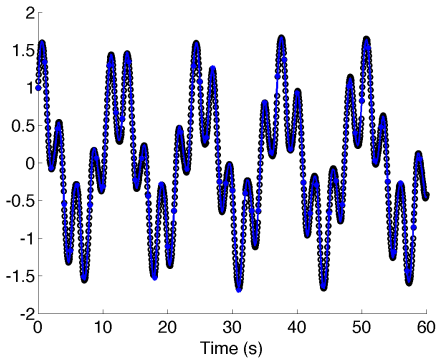
The results of applying the EKF to data under the conditions described for Cases 1, 2, and 3 are shown in Figures 6, 7, and 8, respectively. In all cases (and in what follows), we assume that observations arrive every second, that is $t_{k+1} - t_k = 1$. We also assume that the true initial position of the vortices are (0, 1) and (0, -1) and the true initial position of the drifter is (0.3, -0.6). This corresponds to a 'hard' case as the drifter is strongly advected by the close neighboring vortex at (0, -1). As expected, the EKF successfully tracks the state of the system under full observation both with and without dynamical noise as evident by the magnitude of $\delta^{a,f}(t)$ over time. The EKF fails to track the system under dynamical noise with only partial observations. We see in Figure 8(a) that after approximately $t = 5$ seconds the filter begins to diverge. The analysis step still successfully brings the drifter state 'back to earth' in that it returns the analysis position of the drifter closer to the observed state. However, the forecast immediately begins to diverge. We can see why when we consider 8(b). After time $t = 25$, the forecasted position of the vortex has drifted to another equilibrium position around $x_1 = 6$. Thus, the state of the system between observations evolves as if the vortex is at this position and fails to track the true state of the system well.

6.1.7 Validation of the LETKF

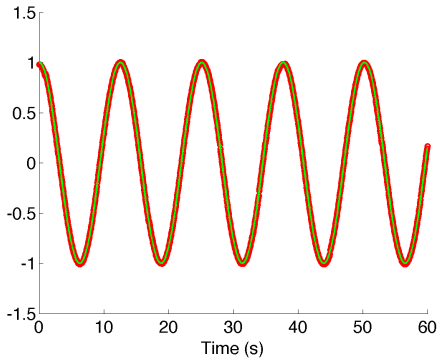
The results of applying the LETKF to data under the conditions described for Cases 1, 2, and 3 are shown in Figures 9, 10, and 11, respectively. These correspond to using the LETKF with $N = 6$ ensemble members, the bandwidth parameter set to $\beta = 2$, and the inflation parameter set to $\lambda = 1.1$. We see that in the case of no dynamical noise with full observation, the filter successfully tracks the state of the system. In the case with dynamical noise and full observation, the filter tracks the state of the system, though the positions of the vortices begin to drift over time. This results in inaccurate forecasts for the drifter. This situation highlights the necessity of covariance inflation during the analysis step, since we see that around $t = 50$ the filter begins to ignore the observations in favor of the forecast state. By observing the trace of the covariance matrix, we can see that this occurs because the matrix becomes near singular with a trace on the order of 10^{-5}. In the case with dynamical and observational noise and partial observation, we see that the filtered state of the vortex gets out of sync with the true state. However, unlike with the EKF, we see that the state for both the drifter position and vortex position remain in the correct neighborhood of the true state. We again observe the necessity of covariance inflation as after time $t = 20$ the filter ignores the observations in favor of the forecast.

6.1.8 Validation of the Particle Filter

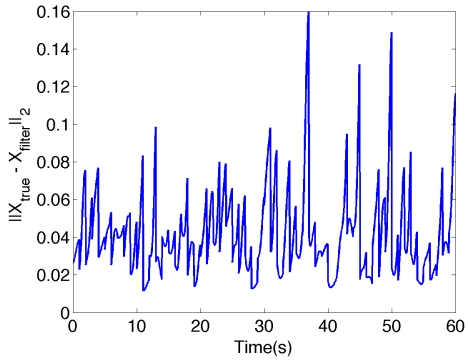
The results of applying the particle filter to data under the conditions described for Cases 1, 2, and 3 are shown in Figures 12, 13, and 14, respectively. The particle filter was run using $N = 100$ particles. This corresponds to a very conservative number of particles compared to a similar study [19]. We see that the particle filter tracks the state of the system extremely well in the case of full observation, both with and without dynamical noise. In the case of partial observation, we see that the filter experiences filter collapse. This occurs due to the finite number of particles used with the filter. During the analysis step, if none of the particles is sufficiently close to the observation, the updated weights of all of the particles may be smaller than roundoff error from floating point arithmetic. In this case, the filter collapses since none of the particles



(a) ξ_1 vs. time.



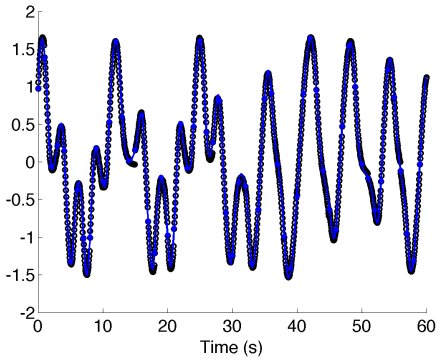
(b) x_1 vs. time.



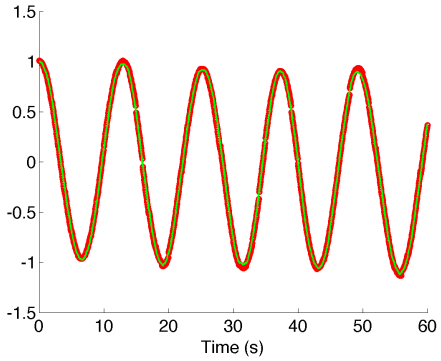
(c) $\delta^{a,f}(t)$ vs. time.

Figure 6: The true state (solid lines) and filtered state (circles) estimated using the EKF with the system fully observed. Dynamical and observational conditions correspond to **Case 1** above.

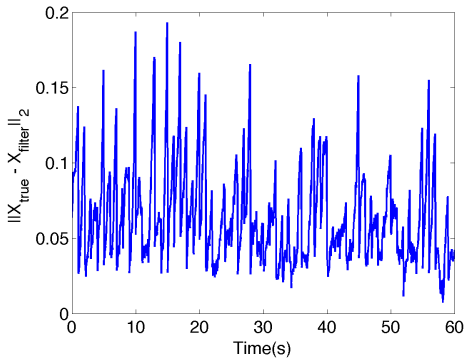
have non-zero weight. We can clearly see this in Figure 14, where around 15 seconds the filtered state has drifted from the true state, indicating that the associated particles themselves will not be near the true state, and thus not near the next observation. After the next analysis step, the particle filter collapses. With more particles, this collapse could be avoided.



(a) ξ_1 vs. time.

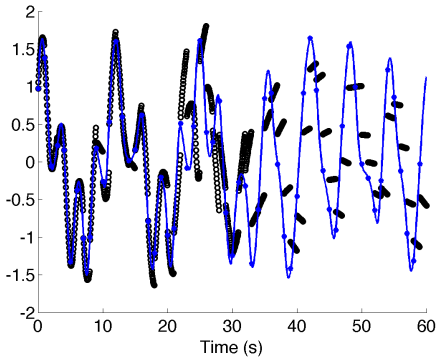


(b) x_1 vs. time.

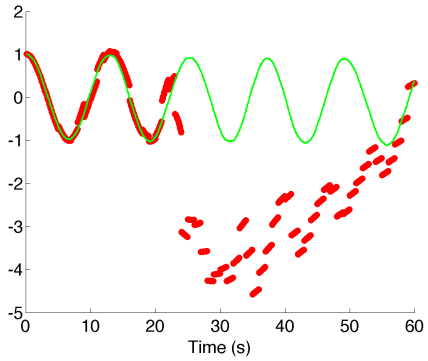


(c) $\delta^{a,f}(t)$ vs. time.

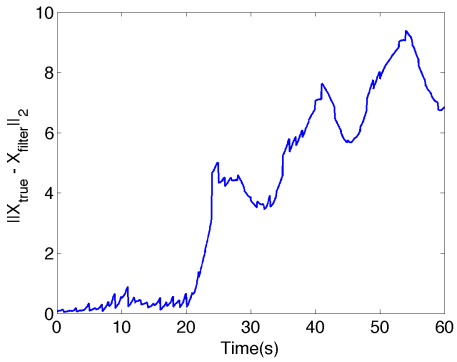
Figure 7: The true state (solid lines) and filtered state (circles) estimated using the EKF with the system fully observed. Dynamical and observational conditions correspond to **Case 2** above.



(a) ξ_1 vs. time.

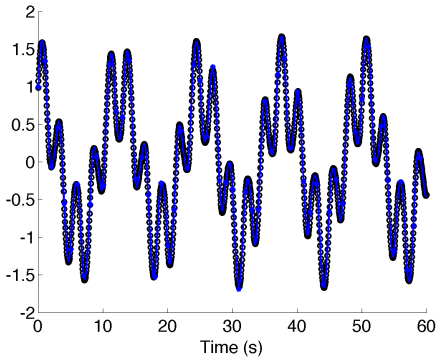


(b) x_1 vs. time.

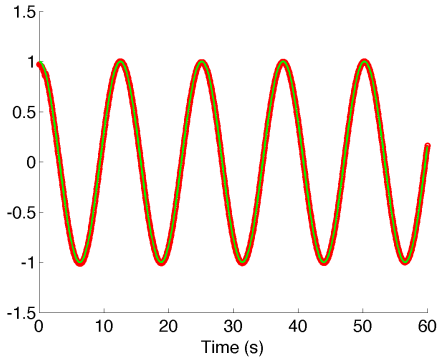


(c) $\delta^{a,f}(t)$ vs. time.

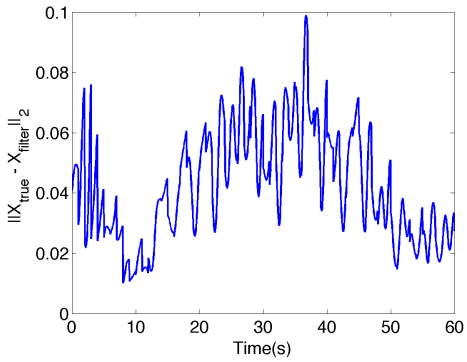
Figure 8: The true state (solid lines) and filtered state (circles) estimated using the EKF with the system partially observed. Dynamical and observational conditions correspond to **Case 3** above.



(a) ξ_1 vs. time.

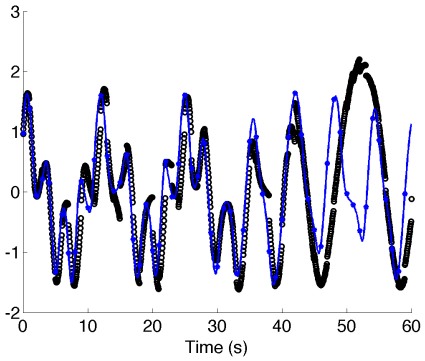


(b) x_1 vs. time.

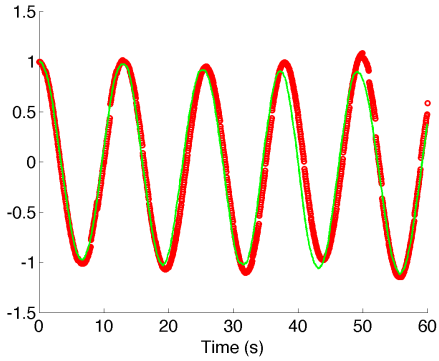


(c) $\delta^{a,f}(t)$ vs. time.

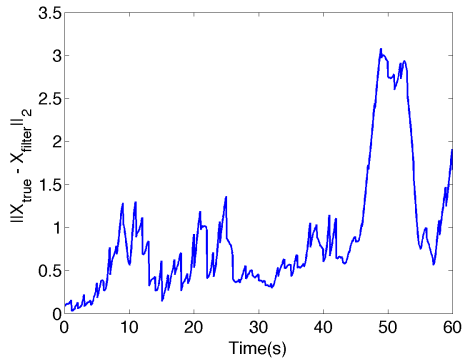
Figure 9: The true state (solid lines) and filtered state (circles) estimated using the LETKF with the system fully observed. Dynamical and observational conditions correspond to **Case 1** above.



(a) ξ_1 vs. time.

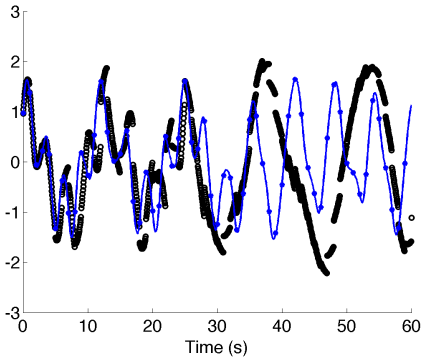


(b) x_1 vs. time.

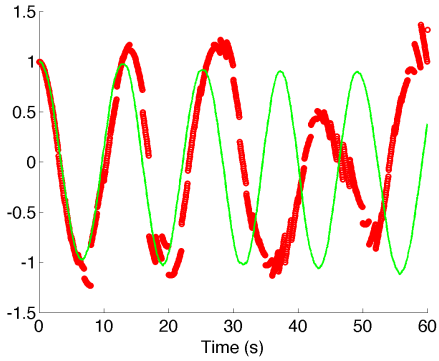


(c) $\delta^{a,f}(t)$ vs. time.

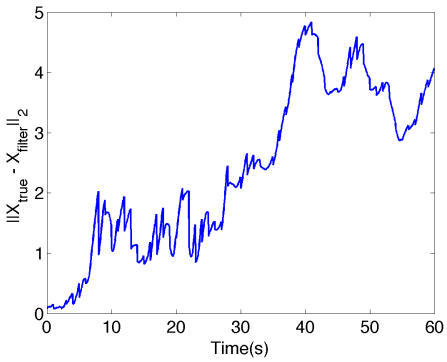
Figure 10: The true state (solid lines) and filtered state (circles) estimated using the LETKF with the system fully observed. Dynamical and observational conditions correspond to **Case 2** above.



(a) ξ_1 vs. time.

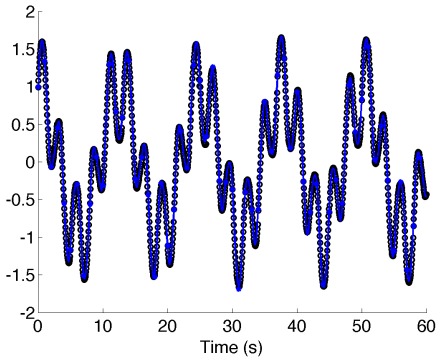


(b) x_1 vs. time.

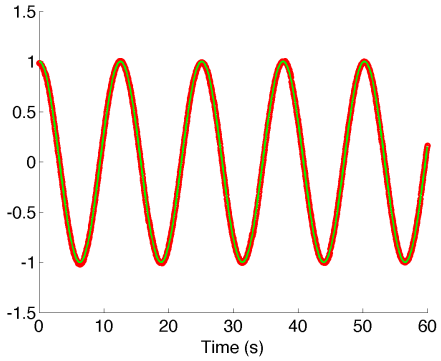


(c) $\delta^{a,f}(t)$ vs. time.

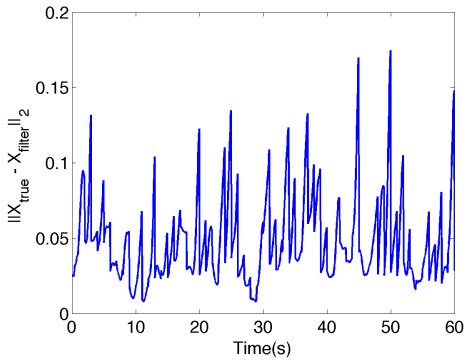
Figure 11: The true state (solid lines) and filtered state (circles) estimated using the LETKF with the system partially observed. Dynamical and observational conditions correspond to **Case 3** above.



(a) ξ_1 vs. time.

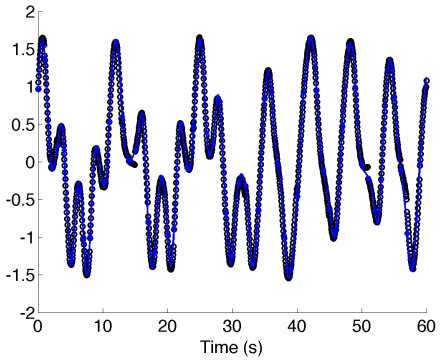


(b) x_1 vs. time.

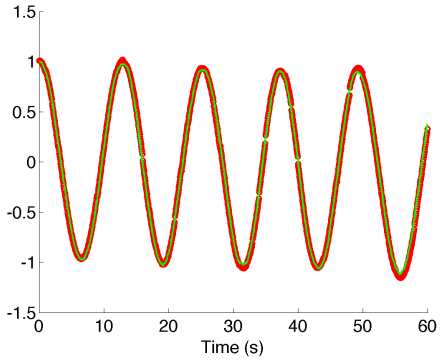


(c) $\delta^{a,f}(t)$ vs. time.

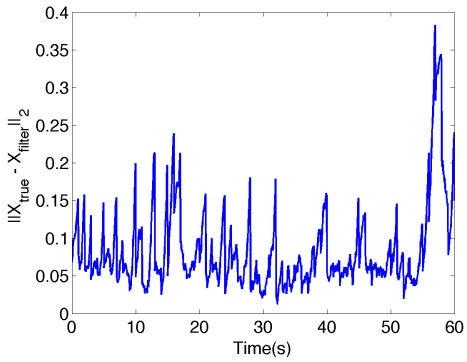
Figure 12: The true state (solid lines) and filtered state (circles) estimated using the particle with the system fully observed. Dynamical and observational conditions correspond to **Case 1** above.



(a) ξ_1 vs. time.

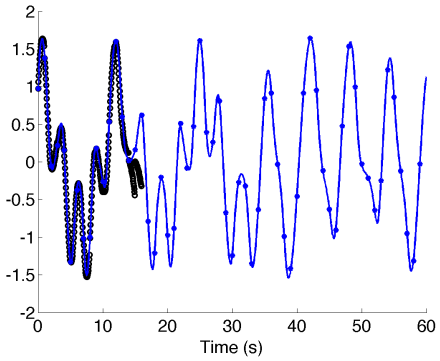


(b) x_1 vs. time.

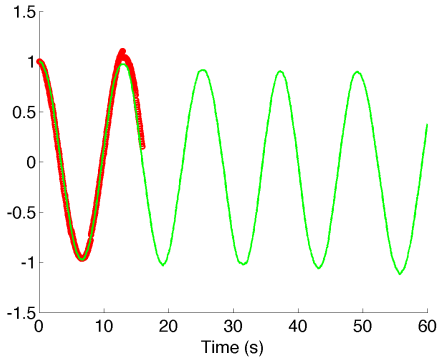


(c) $\delta^{a,f}(t)$ vs. time.

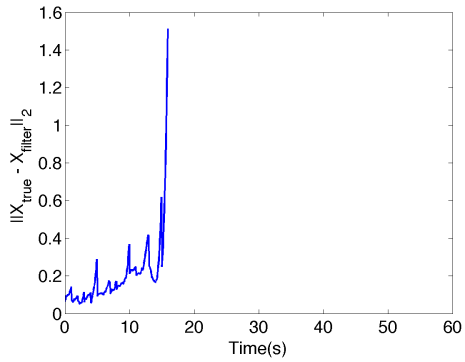
Figure 13: The true state (solid lines) and filtered state (circles) estimated using the particle filter with the system fully observed. Dynamical and observational conditions correspond to **Case 2** above.



(a) ξ_1 vs. time.



(b) x_1 vs. time.



(c) $\delta^{a,f}(t)$ vs. time.

Figure 14: The true state (solid lines) and filtered state (circles) estimated using the particle with the system partially observed. Dynamical and observational conditions correspond to **Case 3** above.

6.2 Phase II - Manifold Detection

For all of the results reported here, M was computed over a 250×250 grid covering the square $[-2.5, 2.5]^2$. M was computed using the same code developed for the LETKF and particle filters, with the addition of a term in the script for the right-hand side to compute M over time. Thus, now the 'particles' in the system correspond to positions on the grid covering $[-2.5, 2.5]^2$, and we may track the arc length traveled by a particle starting at one of these grid points accordingly. It is important to note that we consider the case of (1) **without** dynamical noise (that is, take $\sigma = 0$.) Any possible structure in the deterministic flow field would likely be washed out if not destroyed by a significant amount of dynamical noise. Thus, we consider the case where the system is observed perfectly, and hence we may infer the velocity field (which is necessary to compute both M and finite-time Lyapunov exponents).

The domain $[-2.5, 2.5]^2$ was chosen using the stream function: outside of this region, the flow behaves very regularly and would not contribute much information about the flow structure. The 250×250 grid was determined by experimenting with the number of grid points necessary to obtain a fine enough resolution to resolve the M function in comparison to the analytical stream function, described below. As in the generation of the databases, the time discretization was taken to be $\Delta t = 0.005$.

6.2.1 Computation of M in the Corotational Frame

Using the M function, we expect to obtain a good approximation to the analytically known stream function of the point-vortex model. For the two point-vortex system with the geometry given as in the validation cases above[3], the stream function for the system in a corotating frame[4] is

$$\psi(x, y) = \frac{1}{2}\log[(x-1)^2 + y^2] - \frac{1}{2}\log[(x+1)^2 + y^2] + \frac{1}{4}(x^2 + y^2) \tag{68}$$

where x and y correspond to positions in the plane [19]. By computing M in the corotating frame, we expect to recover this stream function. We see the result of computing M in Figure 15 using $\tau = 20$. Clearly, M captures the same structure as the stream function. Recall that in this frame of reference, a particle starting on a particular streamline remains on that streamline for all time. Thus, the streamlines and their arc lengths completely determine the value of M at any given point in the plane.

6.2.2 Computation of M in the Fixed Frame

Next, we consider computing M in the reference frame of interest, namely the case where the reference frame is fixed and the vortices themselves corotate. This corresponds to equations (6)-(9) for the vortices with the particles lying on the grid. M is known to be a frame-dependent Lagrangian descriptor, but should still capture important behavior of the flow field. As we can see in Figure 16, this is the case. Again, M was computed over an interval of $\tau = 20$. In the fixed frame, M captures the same qualitatively distinct regions of the flow field as in Figure 15: the saddle at $(0, 0)$, the ghost vortices, etc.

6.2.3 Computation of FTLEs for the Symmetric Two Vortex Case

As a final validation of M, we computed finite-time Lyapunov exponents for the symmetric, two vortex system. As mentioned in Section 3.3.2, the rationale and methodology for computing FTLEs differs from M. However, they both have been used to determine coherent structures in simulated and physical flows [13]. The FTLEs were computed over the same domain as M, using the same discretization in time and space. The result for forward-time FTLEs, using $T = 20$, is shown in Figure 17. Again, we see that the ridges in the FTLE field, which correspond to stable manifolds, distinguish qualitatively distinct regions of the flow: near the vortices, the saddle point near the origin, the ghost vortices, etc. Thus, despite the differing philosophies behind these two approaches, we see that they give us essentially the same information.

[3] That is, we take the two vortices to have equal vorticities $\Gamma_1 = \Gamma_2 = 2\pi$ and place the vortices at $(1, 0)$ and $(-1, 0)$.

[4] By corotating frame, fix the two vortices at $(1, 0)$ and $(-1, 0)$ and rotate the frame of reference along with them. This is possible since with this geometry, both vortices rotate around each other in a circle at a constant angular velocity.

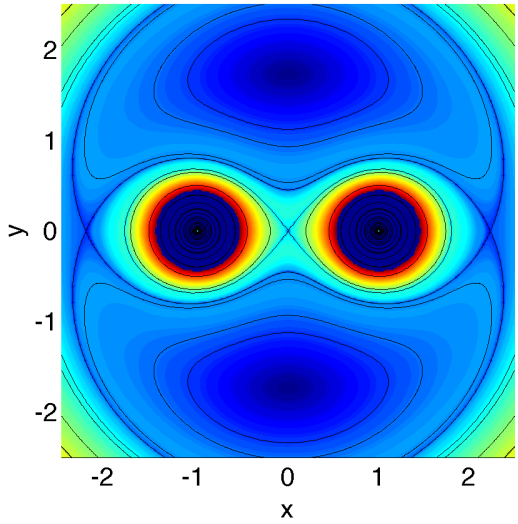


Figure 15: M computed in the corotational frame over a time interval of 40 seconds (that is, $\tau = 20$). The black contour lines correspond to the stream function given by (68) above.

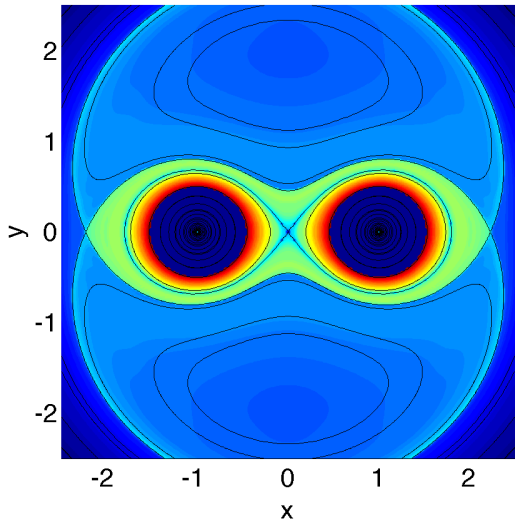


Figure 16: M computed in the fixed frame over a time interval of 40 seconds. The black contour lines correspond to the stream function given by (68) above.

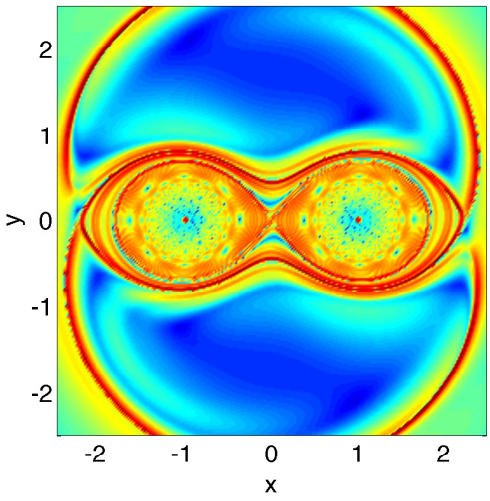


Figure 17: Forward FTLEs computed over a time interval of 20 seconds for the symmetric two vortex case.

7 Results

7.1 Phase I - Lagrangian Data Assimilation

Consider the distance metric defined by (67) where we restrict ourselves to only the vortices' locations. Then as in [19], we may consider a *failure statistic*

$$\hat{f}_{\delta_{div},n}(t) = \text{ fraction of times } \delta(t) > \delta_{div} \text{ at time } t \text{ in } n \text{ trials} \tag{69}$$

where we define a failure distance δ_{div} beyond which we consider the filter to have diverged. Thus, by using a common sample of n realizations of the system described by (1), we may obtain an estimate for the relative performance of the different data assimilation methods over the time course of data assimilation.

We generate the sample database as follows. In all cases, we set the true initial position of vortex 1 to (1, 0) and vortex 2 to (-1, 0). We set the true initial position of the drifter to one of

- **Position 1:** (0.3, -0.6)
- **Position 2:** (1, -0.6)
- **Position 3:** (1, -1)
- **Position 4:** (2.4, -2.4)

See Figure 18 for the placement of these positions on the deterministic stream function in a corotating frame. For each initial position of the drifter, we generate 500 sample paths from the SDE with the dynamical noise level set to $\sigma = 0.02$. To these true values, we add observational noise according to (2) with $\rho = 0.02$. Note that we assume the observational noise is uncorrelated, so we take $\mathbf{R}^t = \rho^2\mathbf{I}$.

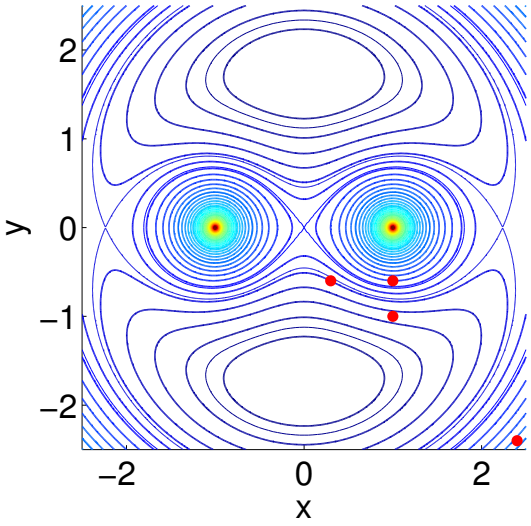


Figure 18: The four initial positions for the drifter used in testing the filters. The figure shows the deterministic stream function in a corotational frame (i.e. the frame of reference follows the corotating vortices). The four drifter positions are signified by the red dots.

7.1.1 Statistics for the EKF, LETKF, and Particle Filter

Several statistics for the trial using **Position 1** (the hardest case of the four) are shown in Table 2. Before proceeding, we emphasize that the data is right-censored. That is, all filter runs were stopped at 60 seconds, whether the filter had failed or not. Thus, the means and standard deviations shown in Table 2 correspond to the means and standard deviations out of the trials that failed.

Positiion 1 corresponds to starting the drifter near the right-most vortex, where it will experience a highly nonlinear trajectory due to the singularity at the vortex center. Given this fact, the filters behave as expected. The EKF performs the worst, with a mean failure time less than half of the mean failure for both the LETKF and the particle filter. The particle filter performs the best, with the largest mean failure time and largest fraction completed. The LETKF performed in the middle, with a larger mean failure time than the EKF, but a smaller fraction of trials completed. The reason for this can be seen in Figure 19. The distribution of failure times for the EKF has a large mode near 0, while still resulting in overall more trials that did not fail within the prescribed 60 time unit trial.

Table 2: The mean and standard deviation of the time-to-failure across the 500 trials that failed using **Position 1** for the drifter. The fraction of the 500 trials that reached $t = 60$ without diverging is also listed. These correspond to the symmetric two vortex case.

	EKF	LETKF	Particle
Mean	14.82	29.14	38.25
Standard Deviation	10.58	12.30	13.10
Fraction Completed	0.24	0.15	0.71

7.2 Phase II - Manifold Detection

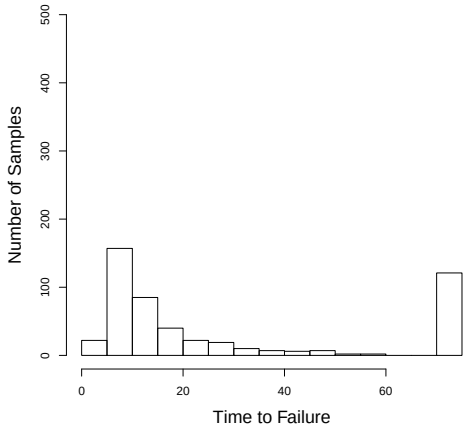
7.2.1 Manifolds for a Chaotic Three Vortex Case

We now consider the more interesting case of three vortices, all with vorticities $\Gamma = 2\pi$, with initial positions (0.2361, -1.8069), (-1.7535, -1.2125), (1.7036, -1.2286), respectively[5]. In this scenario, no corotational frame of reference exists. In fact, in the three vortex case, the advection of fluid particles is chaotic [1]. This scenario allows us to consider more interesting Lagrangian coherent structures than those observed in the two vortex case, and also provides a more strenuous case for the three filters.

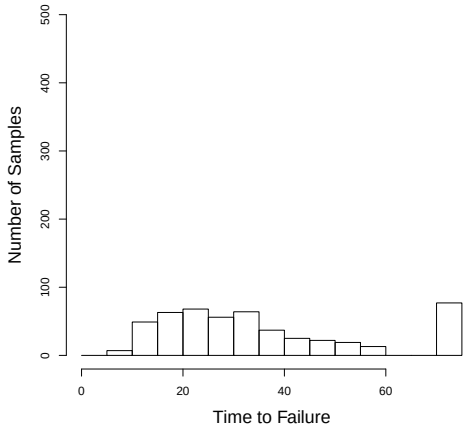
Again, all computations were performed over the square $[-2.5, 2.5]^2$ using a 250×250 grid with time discretization of $\Delta t = 0.005$. As with before, this corresponds to (1) **without** dynamical noise. A forward snapshot of the Lagrangian coherent structures with $\tau = 30$ using both M and FTLEs is shown in Figure 20. We see that in the three vortex case, both mixing and manifolds are present in the flow field. As before, M and the FTLEs clearly distinguish between qualitatively distinct regions of the flow field. For instance, near the vortices as compared to away from the vortices near $(0, 0)$.

So far, we have considered computing M and the FTLEs forward in time. This gives us information about the stable manifolds of the system. We may also compute M and the FTLEs **backwards** in time, thus giving us information about the unstable manifolds of the system. By extracting both of these, we may begin to get a picture of the structures in the flow field that should allow us to intelligently place drifters as to best perform filtering. The extraction is a qualitative process, since there is not yet a rigorous definition of what stable / unstable manifolds correspond to in general, time-dependent dynamical system. We consider extracting manifolds from the FTLEs. Stable manifolds were determined by fixing a cutoff above which we consider a region to correspond to a manifold, since manifolds are known to correspond to ridges of the FTLE field [5]. Similarly, unstable manifolds were determined by a similar process using backward-time FTLEs. An example of the resulting structure is shown in Figure 21.

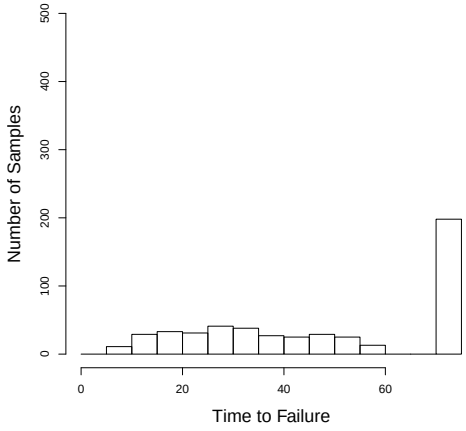
[5]These positions were chosen uniform randomly in the $[-2.5, 2.5]^2$ square.



(a) EKF



(b) LETKF



(c) Particle Filter

Figure 19: Fraction of trials failed during 60 seconds for 500 trials using **Position 1** for the drifter in the two vortex case. Note that the data is right-censored due to the hard stop at 60 seconds. The bin to the right of 60 corresponds to those trials that did not fail within 60 seconds.

7.2.2 *A priori* Drifter Placement using Observed Manifolds

We consider the same testing procedure as outlined in Section 7.1 with the following changes. We consider the three vortex case described in Section 7.2.1. The choice of drifter positions was made using the extracted stable/unstable manifolds shown in Figure 21. The positions used are given below:

- **Position 1:** (0.7794, -0.2328)
- **Position 2:** (0, -1.5)
- **Position 3:** (-1.913, 0.3543)
- **Position 4:** (-0.2935, 2.318)

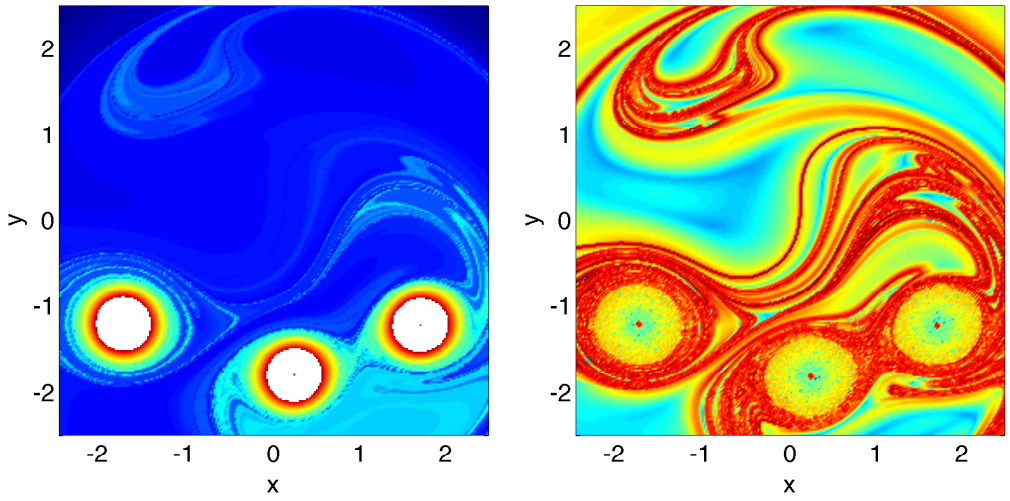


Figure 20: M (on the left) and FTLEs (on the right) for the three vortex case with initial conditions given in the body of the text. These were computed in forward-time over an interval of $\tau = 30$.

Two of the initial conditions for the drifter are near the stable manifolds (**Positions 1** and **4**), one of the initial conditions is near an unstable manifold (**Position 2**), and one is near one of the point vortices (**Position 2**). This last position was chosen to further investigate the performance of the EKF and LETKF in this highly nonlinear regime.

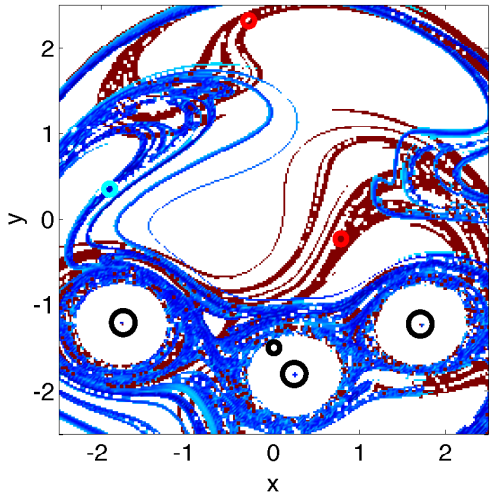


Figure 21: Placement of the drifters in the observing system design study. The red structures correspond to stable manifolds, extracted using forward-time FTLEs. The blue structures correspond to unstable manifolds, determined using backward-time FTLEs. Red circles correspond to drifters placed near stable manifolds, the blue circle corresponds to a drifter placed near an unstable manifold, and the black circle corresponds to a drifter placed near one of the point vortices.

Another difference in the testing procedure is that we generate the true trajectories of the vortices and drifter **without** dynamical noise. We do this to ensure that the manifolds observed in the previous section remain intact: the addition of dynamical noise might perturb or destroy these manifolds. However, for each position, we generate 500 samples by adding observational noise with $\rho = 0.02$, as before. We use $N = 8$ ensemble members for the LETKF and $N = 250$ particles for the particle filter. Observations of the system were assimilated every second.

The results of these trials are shown in Figures 22, 23, and 24. As with the testing in the two vortex case, the data is right-censored since the trial ended after 60 seconds, regardless of whether the filter failed or not. Summary statistics for the three filters across the four positions are available in Table 3.

Table 3: The mean and standard deviation of the time-to-failure across the 500 trials that failed using the four positions in the three vortex case. The fraction of the 500 trials that reached $t = 60$ without diverging is also listed.

Position 1	EKF	LETKF	Particle
Mean	29.9	32	29.5
Standard Deviation	10.43	11.9	11.4
Fraction Completed	0.8	0.86	0.18

Position 2	EKF	LETKF	Particle
Mean	5.1	21.4	33.6
Standard Deviation	3.3	12.5	11.4
Fraction Completed	0	0.05	0.21

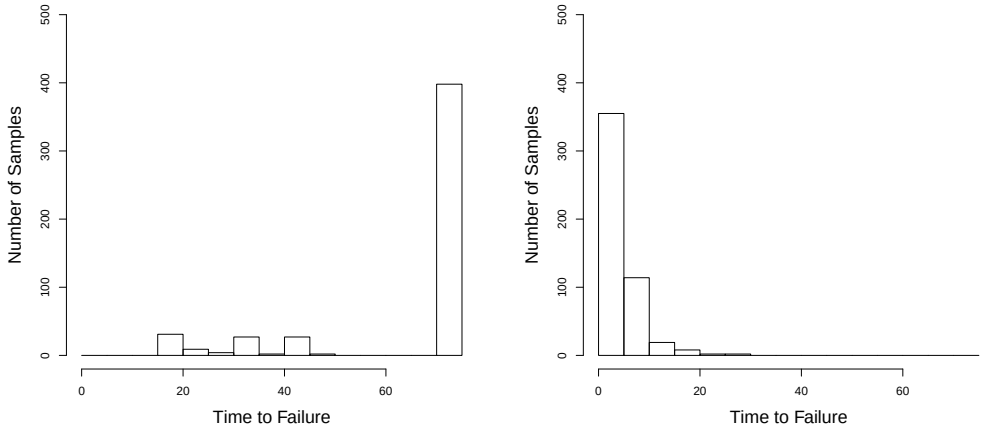
Position 3	EKF	LETKF	Particle
Mean	29.6	41.8	37.7
Standard Deviation	11.4	8.6	11.8
Fraction Completed	0.93	0.81	0.67

Position 4	EKF	LETKF	Particle
Mean	48.4	51	36.5
Standard Deviation	8.2	8.2	13.5
Fraction Completed	0.9	0.89	0.23

We see that in the case without dynamical noise, the LETKF outperforms the particle filter and EKF in all but position 2. Similarly, the EKF outperforms the particle filter in all cases except position 2. Recall that position 2 corresponds to an initial drifter position near one of the vortices. Due to the nature of the point vortex model, such a drifter exhibits highly nonlinear motion. Because the EKF depends upon linearization through the tangent linear model, its performance in this case is as expected.

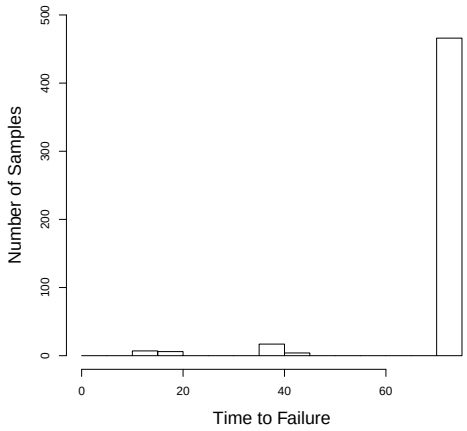
We can also consider the approximate prior distribution of the system in various cases. This was accomplished by using the particle filter with $N = 1000$ particles starting from a Gaussian prior distribution. This prior was evolved for 1 second, equivalent to a single forecast interval. The evolved prior distribution for positions 1 and 2 are shown in Figure 25. For the drifter starting in position 1, we see that the evolved prior can be well-approximated by a Gaussian distribution. However, for the drifter starting in position 2, we see that the prior is non-elliptical, and thus could not be approximated by a Gaussian distribution. This helps explain the reason for the superior performance of the particle filter for this drifter position: both the EKF and LETKF assume a Gaussian prior in their analysis steps, and this assumption clearly does not hold for the drifter starting in position 2.

We also note that the placement of a drifter near a stable vs. unstable manifold had little effect on the filter performance for the EKF and LETKF in the three cases we studied. However, it did have a significant impact on the particle filter. We see that the particle filter performed better with a drifter placed near an

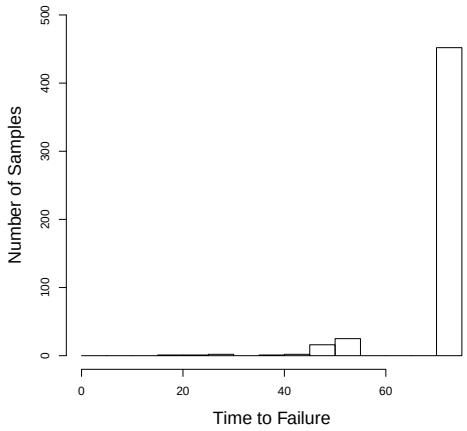


(a) **Position 1**

(b) **Position 2**



(c) **Position 3**

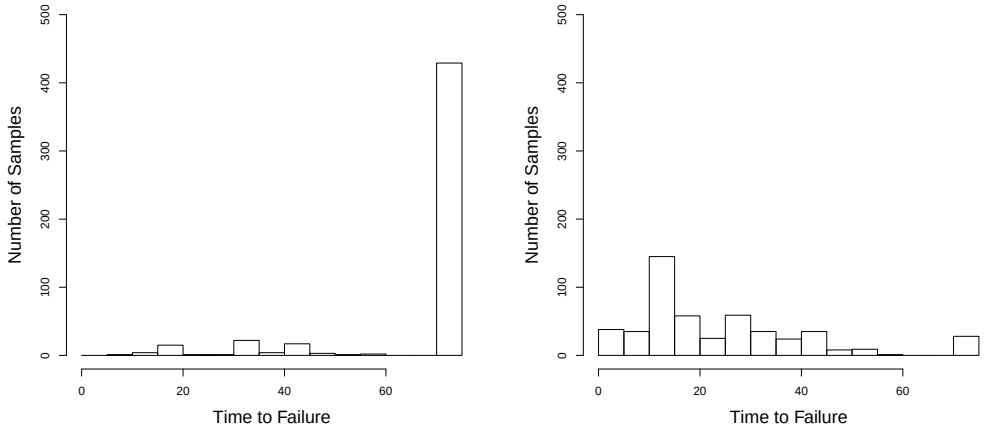


(d) **Position 4**

Figure 22: Distribution of the failure times for the EKF in the three vortex case across the four drifter positions. Note that the data is right-censored due to the hard stop at 60 seconds. The bin to the right of 60 corresponds to those trials that did not fail within 60 seconds.

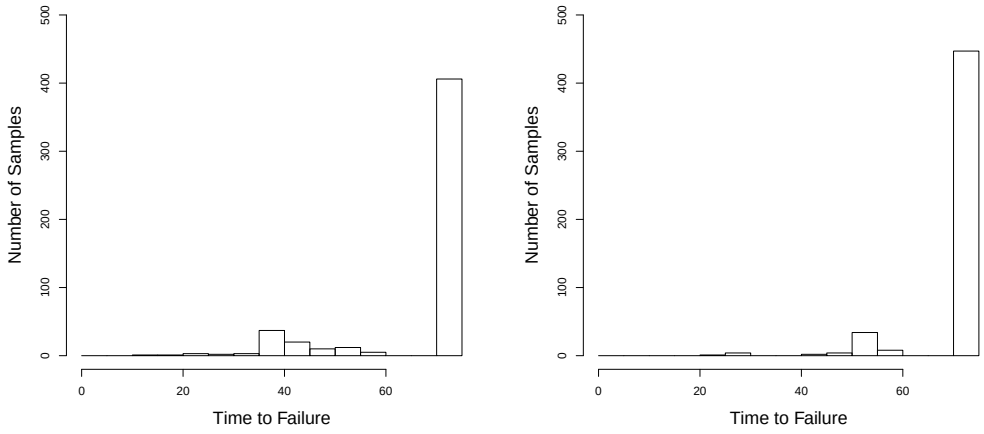
unstable manifold. To make sense of this, we recall the discussion of how adjacent trajectories near stable / unstable manifolds behave. Trajectories starting on either side of a stable manifold with diverge in time, while trajectories starting on either side of an unstable manifold will converge in time. For drifter placement near a stable manifold, the particle cloud most likely spans the manifold, and thus we expect the particles to diverge in time. With a finite number of particles, this can result in a degradation of the particle cloud and possible filter collapse over time.

Overall, this study highlights the weaknesses of the EKF in the case of nonlinearity in the flow field and of both the LETKF and EKF in the case of non-Gaussianity. However, when the flow field is relatively linear, as it is away from the vortices, the distribution of the system remains approximately Gaussian and the EKF and LETKF perform very well. This study focused on the case with no dynamical noise in the system. Preliminary investigation of the three vortex system with dynamical noise indicate that the particle



(a) **Position 1**

(b) **Position 2**

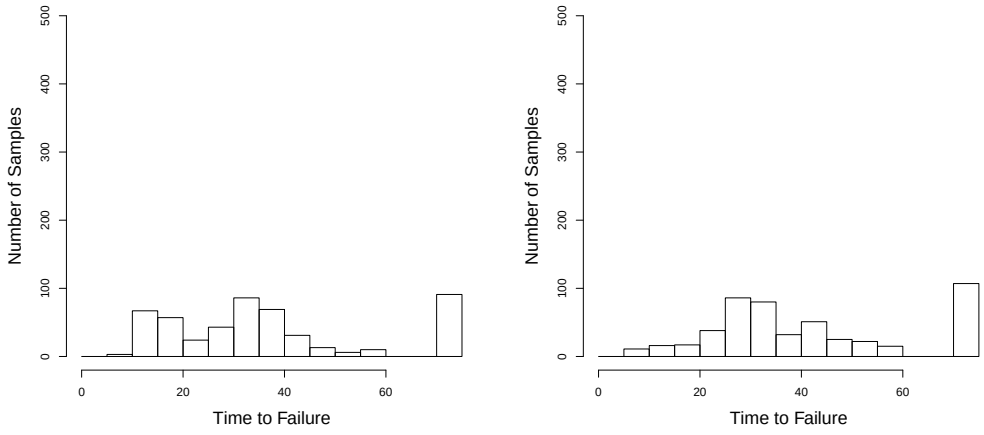


(c) **Position 3**

(d) **Position 4**

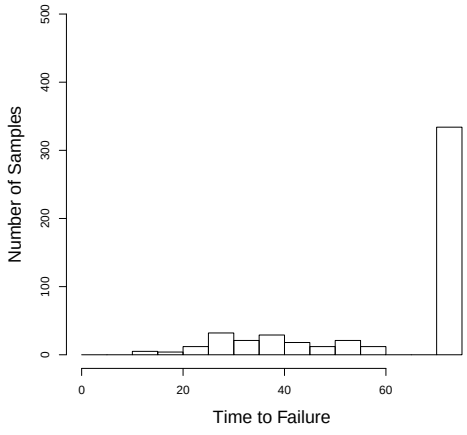
Figure 23: Distribution of the failure times for the LETKF in the three vortex case across the four drifter positions. Note that the data is right-censored due to the hard stop at 60 seconds. The bin to the right of 60 corresponds to those trials that did not fail within 60 seconds.

filter outperforms the other filters in this case.

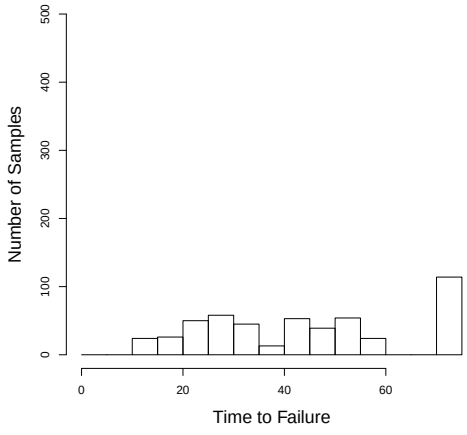


(a) **Position 1**

(b) **Position 2**



(c) **Position 3**



(d) **Position 4**

Figure 24: Distribution of the failure times for the particle filter in the three vortex case across the four drifter positions. Note that the data is right-censored due to the hard stop at 60 seconds. The bin to the right of 60 corresponds to those trials that did not fail within 60 seconds.

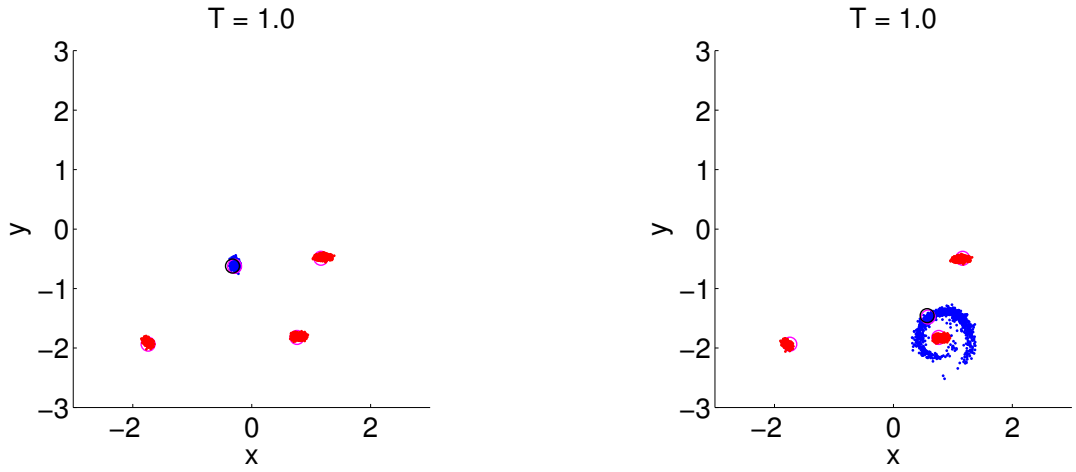


Figure 25: A comparison between the distribution of particles in **Position 1** and **Position 2** after a single forecast interval. The figures were generated using 1000 particles.

8 Conclusions

The point-vortex system offers an excellent model for Lagrangian data assimilation. It highlights one of the defining characteristics of the Lagrangian approach: when observing a particle in the flow and not the flow itself, we must track the covariance function of the system over time. In the Lagrangian case, the instantaneous position of the drifter provides no information about the positions of the vortices. Only through correlations in higher-order moments can this information be recovered. We have seen how important accurate tracking of these higher-order moments can be. All of the data assimilation methods considered (EKF, LETKF, particle filter) estimate these moments to differing degrees of accuracy depending on the dynamical behavior of the drifter. In the case where the dynamics induced by the point vortices are near linear, the EKF performs well. However, as the dynamics become more nonlinear and the process becomes more non-Gaussian, we have seen that the EKF cannot accurately track the covariance function over time. In this case, the LETKF and particle filter outperform the EKF, with the particle filter performing the best.

Both M and FTLEs capture the dynamical structure inherent in the flow field. We were able to determine this structure even without a corotating frame, and in the chaotic three vortex study, in a case where no such frame of reference exists. We saw that M and FTLEs give similar information about the manifolds of the flow field despite their very different approaches to LCSs. We successfully extracted this information in the chaotic three vortex case and used the detected manifolds to decide drifter placement *a priori*. We discovered that the particle filter performs best when the drifter is placed near an unstable manifold. Thus, the structure of the flow field provides us with valuable information which may be used to design better observing systems. This part of the study relied on the true system evolving in time according to deterministic ODEs. However, Lagrangian coherent structures has been identified in real world systems using these methods, and thus the assumption of stochasticity in the dynamics may not be necessary to capture these overall flow structures.

9 Project Schedule and Milestones

The project schedule is as follows:

- Phase I
  - Produce database: now through mid-October
  - Develop extended Kalman Filter: now through mid-October
  - Develop ensemble Kalman Filter: mid-October through mid-November
  - Develop particle filter: mid-November through end of January
  - Validation and testing of three filters (serial): Beginning in mid-October, complete by February
  - Parallelize ensemble methods: mid-January through March
- Phase II
  - Develop serial code for manifold detection: mid-January through mid-February
  - Validate and test manifold detection: mid-February through mid-March
  - Parallelize manifold detection algorithm: mid-March through mid-April

The corresponding milestones are the following:

- Phase I
  - Complete validation and testing of extended Kalman filter: beginning of November
  - Complete validation and testing of (serial) ensemble Kalman filter: beginning of December
  - Complete validation and testing of (serial) particle filter: end of January

- Phase II
  - Complete validation and testing of (serial) manifold detection: mid-March
  - Parallelize ensemble methods: beginning of April
  - Parallelize manifold detection algorithm: end of April

10 Deliverables

The deliverables for this project include: the collection of databases used for the filter validation and testing, a suite of software for performing EKF, LETKF, and particle filtering on the stochastic point-vortex model, and a suite of software for performing manifold detection on the deterministic point-vortex model using Mendoza and Mancho's M function as well as FTLEs. The LETKF and particle filter, as well as the computation of M, will all be parallelized.

References

[1] H. Aref. Point vortex dynamics: A classical mathematics playground. *Journal of mathematical Physics*, 48:065401, 2007.

[2] J. Baehr, D. McInerney, K. Keller, and J. Marotzke. Optimization of an observing system design for the north atlantic meridional overturning circulation. *Journal of Atmospheric and Oceanic Technology*, 25(4):625, 2008.

[3] A. Doucet, N. De Freitas, and N. Gordon. *Sequential Monte Carlo methods in practice*. Springer Verlag, 2001.

[4] G. Evensen. *Data assimilation: the ensemble Kalman filter*. Springer Verlag, 2009.

[5] G. Haller. Finding finite-time invariant manifolds in two-dimensional velocity fields. *Chaos*, 10(1):99–108, 2000.

[6] D.J. Higham. An algorithmic introduction to numerical simulation of stochastic differential equations. *SIAM review*, pages 525–546, 2001.

[7] B.R. Hunt, E.J. Kostelich, and I. Szunyogh. Efficient data assimilation for spatiotemporal chaos: a local ensemble transform kalman filter. *Physica D: Nonlinear Phenomena*, 230(1-2):112–126, 2007.

[8] K. Ide, L. Kuznetsov, and C.K.R.T. Jone. Lagrangian data assimilation for point vortex systems. *Journal of Turbulence*, 3, 2002.

[9] A.H. Jazwinski. *Stochastic processes and filtering theory*. Dover Publications, 2007.

[10] E. Kalnay. *Atmospheric modeling, data assimilation, and predictability*. Cambridge Univ Press, 2003.

[11] A.J. Krener and K. Ide. Measures of unobservability. In *Proceedings of the 48th IEEE Conference on Decision and Control*, pages 6401–6406. IEEE, 2009.

[12] J.S. Liu and R. Chen. Sequential monte carlo methods for dynamic systems. *Journal of the American statistical association*, pages 1032–1044, 1998.

[13] C. Mendoza and A.M. Mancho. Hidden geometry of ocean flows. *Physical review letters*, 105(3):38501, 2010.

[14] E. Ott. *Chaos in dynamical systems*. Cambridge Univ Pr, 2002.

[15] DJ Patil, B.R. Hunt, E. Kalnay, J.A. Yorke, and E. Ott. Local low dimensionality of atmospheric dynamics. *Physical Review Letters*, 86(26):5878–5881, 2001.

[16] H. Salman. A hybrid grid/particle filter for lagrangian data assimilation. i: Formulating the passive scalar approximation. *Quarterly Journal of the Royal Meteorological Society*, 134(635):1539–1550, 2008.

[17] H. Salman. A hybrid grid/particle filter for lagrangian data assimilation. ii: Application to a model vortex flow. *Quarterly Journal of the Royal Meteorological Society*, 134(635):1551–1565, 2008.

[18] R.M. Samelson and S. Wiggins. *Lagrangian transport in geophysical jets and waves: the dynamical systems approach*. Springer Verlag, 2006.

[19] E.T. Spiller, A. Budhiraja, K. Ide, and C.K.R.T. Jones. Modified particle filter methods for assimilating lagrangian data into a point-vortex model. *Physica D: Nonlinear Phenomena*, 237(10-12):1498–1506, 2008.

[20] M.K. Tippett, J.L. Anderson, C.H. Bishop, T.M. Hamill, and J.S. Whitaker. Ensemble square root filters. *Monthly Weather Review*, 131:1485–1490, 2010.

[21] J. Wilkie. Numerical methods for stochastic differential equations. *Physical Review E*, 70(1):017701, 2004.

A Overview of the Idealized Data Assimilation Problem

Data assimilation is an iterative process that combines a forecast determined using prior observations with a current observation to optimally predict the current state of a system. These two steps of data assimilation, forecasting and analysis, are explained below in their idealized form.

Forecast: Let $p \equiv p(\mathbf{x}^t, t)$ be the probability density of the state $\mathbf{x}^t$ described by (1). Then the Fokker-Planck equation for p is

$$\frac{\partial p}{\partial t} + \sum_i \frac{\partial m_i p}{\partial x_i} = \frac{1}{2} \sum_{i,j} \frac{\partial^2 p(\mathbf{Q})_{ij}}{\partial x_i \partial x_j} \tag{70}$$

where m_i is the i^{th} component of the evolution operator $M(\cdot)$ from (1) and $\mathbf{Q}$ is the covariance matrix characterizing $\boldsymbol{\eta}^t$ in (1) [16]. Given p, we 'know everything' about $\mathbf{x}^t$ in the sense that we can compute all possible moments of the distribution, which we may then use in the analysis step of data assimilation. However, tractable solutions to (70) rarely exist. Thus, the methods outlined in this report (extended Kalman filter, ensemble Kalman filter, and particle filter) represent attempts to approximate p by applying various assumptions on the form of p that make the solution to (70) tractable.

Analysis: When an observation arrives at time t_k, we wish to perform the analysis step. This step allows us to determine the posterior estimate of the state given the current observation, $p(\mathbf{x}^f(t_k)|\mathbf{y}_k^o)$, by applying Bayes's rule to combine the likelihood of the data $p(\mathbf{y}_k^o|\mathbf{x}^f(t_k))$ and the prior $p(\mathbf{x}^f(t_k))$. We drop the t_k dependence for brevity of presentation. Thus, the analysis estimate is given by

$$p(\mathbf{x}^f|\mathbf{y}_k^o) = \frac{p(\mathbf{x}^f, \mathbf{y}_k^o)}{p(\mathbf{y}_k^o)} = \frac{p(\mathbf{y}_k^o|\mathbf{x}^f)p(\mathbf{x}^f)}{p(\mathbf{y}_k^o)} \tag{71}$$

$$= \frac{p(\mathbf{y}_k^o|\mathbf{x}^f)p(\mathbf{x}^f)}{\int p(\mathbf{y}_k^o|\mathbf{x}^f)p(\mathbf{x}^f)\, d\mathbf{x}^f}. \tag{72}$$

As with the forecasting step, the application of Bayes's rule is an idealization of the analysis stage. As we will see, several assumptions may be made about the distribution of the prior and likelihood that simplify (72).