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

Mid-year Progress 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.

December 20, 2011

1 Background

1.1 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* [15]. 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.

1.2 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 2.1.

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 $\mathbf{x}_D^t$ [6]. Thus, the overall state

of the system is given by

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

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.

1.3 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 [1]. 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 [9]. 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.

2 Approach

2.1 Phase I - Lagrangian Data Assimilation

2.1.1 Model System

For the system we wish to assimilate, we consider the deterministic point-vortex model with N_v vortices and N_d drifter [6]. 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).

2.1.2 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* [5]. 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 $t - s$.
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

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

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

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

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

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 [18]. 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 $\Delta\tau = 5 \times 10^{-3}$.

2.1.3 Extended Kalman Filter

The data assimilation problem outlined in Section 1.2 with linear dynamics, linear observations, and Gaussian noise has an exact solution [7]. 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.

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.

2.1.4 Ensemble Kalman Filter

Another approach to data assimilation that addresses some weaknesses inherent in the EKF is the ensemble Kalman filter (EnKF) [3]. 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

$$\bar{\mathbf{x}}^f = \frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_i^f. \tag{34}$$

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

$$\mathbf{P}_e^f = \frac{1}{N-1}\sum_{i=1}^{N}(\mathbf{x}_i^f - \bar{\mathbf{x}}^f)(\mathbf{x}_i^f - \bar{\mathbf{x}}^f)^T. \tag{35}$$

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. These two approaches are described below.

2.1.5 Ensemble Kalman Filter with Perturbed Observations

In the perturbed observation approach to the analysis step, we perturb the observation according to the probability model of the observation process. That is, form

$$\mathbf{y}_{k,i}^o = \mathbf{y}_k^o + \boldsymbol{\epsilon}_i, \qquad i = 1, \ldots, n \tag{36}$$

where $\boldsymbol{\epsilon}_i \sim N(\mathbf{0}, \mathbf{R}^o)$. It is ensured that the $\boldsymbol{\epsilon}_i$ have mean zero. Each member of the ensemble is then updated using these perturbed observations according to the analysis equations of the Kalman filter,

$$\mathbf{x}_{k,i}^a = \mathbf{x}_i^f(t_k) + \mathbf{K}_k(\mathbf{y}_{k,i}^o - h_k(\mathbf{x}_i^f(t_k))) \tag{37}$$

where

$$\mathbf{K}_k = \mathbf{P}_e^f(t_k)\mathbf{H}_k^T(\mathbf{H}_k\mathbf{P}_e^f(t_k)\mathbf{H}_k^T + \mathbf{R}_k^o)^{-1}. \tag{38}$$

That is, each ensemble member is updated separately using its own state and observation and the common sample covariance matrix.

2.1.6 Ensemble Transform Kalman Filter

The Ensemble Transform Kalman Filter (ETKF) is a type of square root filter that allows for the deterministic generation of the analysis ensemble given an observation and the forecast ensemble [17].

We define the ensemble spread matrix for the forecast as

$$\mathbf{X}^f = \begin{bmatrix} \mathbf{x}_1^f - \bar{\mathbf{x}} & \mathbf{x}_2^f - \bar{\mathbf{x}} & \ldots & \mathbf{x}_n^f - \bar{\mathbf{x}} \end{bmatrix}. \tag{39}$$

Thus we see that

$$\frac{1}{N-1}\mathbf{X}^f(\mathbf{X}^f)^T = \mathbf{P}_e^f \tag{40}$$

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

$$\mathbf{X}^a = \mathbf{X}^f\mathbf{W} \tag{41}$$

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

$$\mathbf{P}_e^a = \frac{1}{N-1}\mathbf{X}^a(\mathbf{X}^a)^T \tag{42}$$

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

$$= \frac{1}{N-1}\mathbf{X}^f\mathbf{W}\mathbf{W}^T(\mathbf{X}^f)^T. \tag{44}$$

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

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

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

$$\bar{\mathbf{x}}^a = \bar{\mathbf{x}}^f + \mathbf{X}^f\mathbf{D}(\mathbf{X}^f)^T\mathbf{H}^T(\mathbf{R}^\mathbf{o})^{-\mathbf{1}}(\mathbf{y}^o - \mathbf{H}\bar{\mathbf{x}}^f). \tag{46}$$

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

$$\mathbf{D} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T \tag{47}$$

and forming

$$\mathbf{W} = \dot{\mathbf{U}}\dot{\mathbf{\Lambda}}^{1/2}\dot{\mathbf{U}}^T \tag{48}$$

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 $\mathbf{D}$. Then we may form the analysis ensemble by applying (41). Note that this choice of $\mathbf{W}$ is *not* unique.

2.1.7 Weaknesses of the EnKF

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 must be relatively large to obtain good approximations to the mean and covariance matrix of the process. Finally, the EnKF still assumes a Gaussian posterior in the analysis step. While a Gaussian random vector is completely characterized by its mean and covariance matrix, this is not necessarily the case for an arbitrary random vector.

2.1.8 Particle Filter

The particle filter closely resembles the EnKF with the key advantage that it performs a full Bayesian update at the analysis step [13, 16]. 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

$$p(\mathbf{y}_k^o|\mathbf{x}(t_k)) = C_1\exp\left(-\frac{1}{2}(\mathbf{y}_k^o - h(\mathbf{x}(t_k)))^T(\mathbf{R}^o)^{-1}(\mathbf{y}_k^o - h(\mathbf{x}(t_k)))\right) \tag{49}$$

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 (65). In this case, Bayes's theorem takes the form of

$$w_{i,k} = C_2 w_{i,k-1} p(\mathbf{y}_k^o|\mathbf{x}_i^f(t_k))) \tag{50}$$

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

$$\bar{\mathbf{x}}^f(t_k) = \sum_{i=1}^{N} w_{i,k}\mathbf{x}_i^f(t_k). \tag{51}$$

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 [2]. 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 will use residual resampling [10].

2.2 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 [15]. 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.

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 defined region. In this case, we may define the following Lagrangian descriptor [11]

$$M(\mathbf{x}_D^t, t^*) = \int_{t^*-\tau}^{t^*+\tau} \left(\sum_{i=1}^{n} \left(\frac{dx_D^i(t)}{dt} \right)^2 \right)^{1/2} dt \tag{52}$$

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 (such as the one we consider), M provides a time-independent partition of the phase space of our system. Thus, we may consider $M(\mathbf{x}_D^t) \equiv M(\mathbf{x}_D^t, 0)$. This function may also be applied to time-dependent flows, in which case we must maintain the t^* dependency.

3 Implementation

All algorithms will initially be prototyped in MATLAB on a MacBook Pro with a 2.4 GHz Intel Core 2 Duo processor with 4 GB of RAM. The algorithms will initially be developed to run serially. As the year progresses (see Section 7 for more details), the code for evolving the particles in the ensemble Kalman filter and particle filter will be ported to run in parallel using MATLAB's Parallel Computing Toolbox. These modified versions will then be run on a computing cluster. Similarly, the manifold detection algorithm will initially be developed serially and then parallelized.

4 Databases

During Phase I, the databases will consist of archived numerical solutions to (1) with varied initial conditions for the positions of the drifters. These solutions will be generated using a stochastic Runge-Kutta second-order SDE solver with prescribed levels of dynamical noise [18]. Observational noise will be added to the locations of the drifters according to (2), again following prescribed levels of noise (see Section 5 for more details). These common databases will be 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.

5 Validation

5.1 Phase I - Lagrangian Data Assimilation

5.1.1 Validation of the Numerical SDE Solver

Consider the scalar SDE

$$dX(t) = \lambda X(t)dt + \mu X(t)dW(t) \tag{53}$$

$$X(0) = X_0. \tag{54}$$

This SDE has the solution

$$X(t) = X_0 \exp\left[\left(\lambda - \frac{1}{2}\mu^2\right)t + \mu W(t)\right]. \tag{55}$$

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 (55). 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 1, 2, and 3. 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.

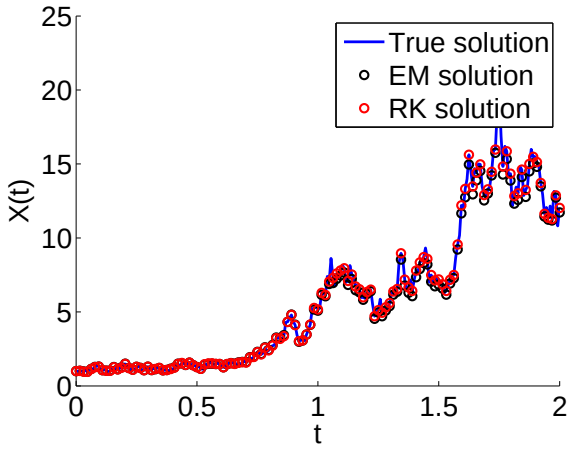


Figure 1: The true solution to (53) as well as the Euler-Marayuma and Runge-Kutta approximations.

5.1.2 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 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* [8]. Substituting $\mathbf{x}_2$ into the deterministic differential equation we find

$$\frac{d}{dt}\mathbf{x}_2 = \frac{d}{dt}(\mathbf{x} + \mathbf{y}) = M(\mathbf{x} + \mathbf{y}) \tag{56}$$

$$= M(\mathbf{x}) + \left.\frac{\partial M(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}(t)} \mathbf{y} + \mathcal{O}(\mathbf{y}^2) \tag{57}$$

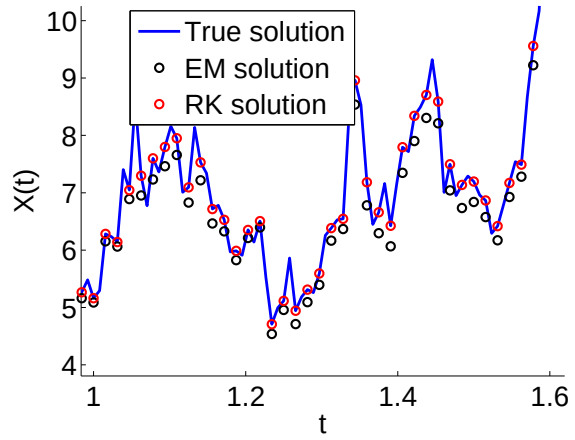


Figure 2: A magnification of a portion of Figure 1. This clearly demonstrates the greater accuracy of Runge-Kutta compared to Euler-Marayuma as expected by its better strong order of convergence.

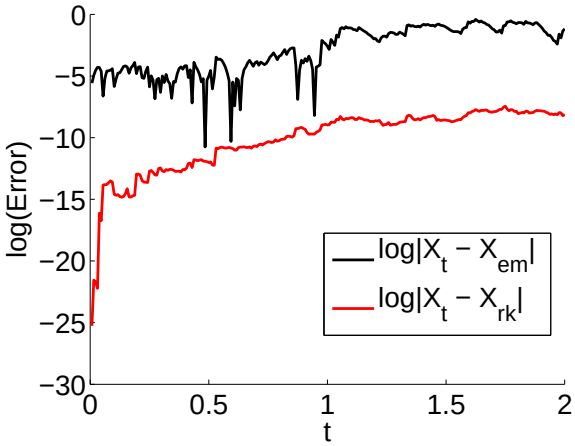


Figure 3: The log of the error between the true solution to (53) and the Euler-Marayuma (black) and Runge-Kutta (red) numerical approximations.

where in (57) 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

$$\frac{d\mathbf{y}}{dt} = \left.\frac{\partial M(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}(t)} \mathbf{y} \tag{58}$$

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 (58) reduces to

$$\frac{d\mathbf{y}}{dt} = \mathbf{M}\mathbf{y} \tag{59}$$

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 (59). 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 4. 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.

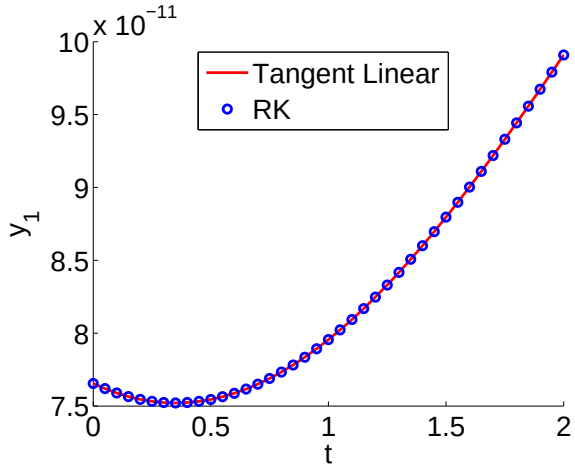


Figure 4: The evolution of the first component of the tangent linear vector $\mathbf{y}$ computed by the tangent linear model and by differencing the reference ($\mathbf{x}_1$) and perturbed ($\mathbf{x}_2$) trajectory. Note that $\mathbf{y}^0$ was chosen uniform randomly on the interval $[0, 10^{-10}]^D$ where D is the dimension of the state space.

5.1.3 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, $\sigma = 0.02$, $\rho = 0.02$.

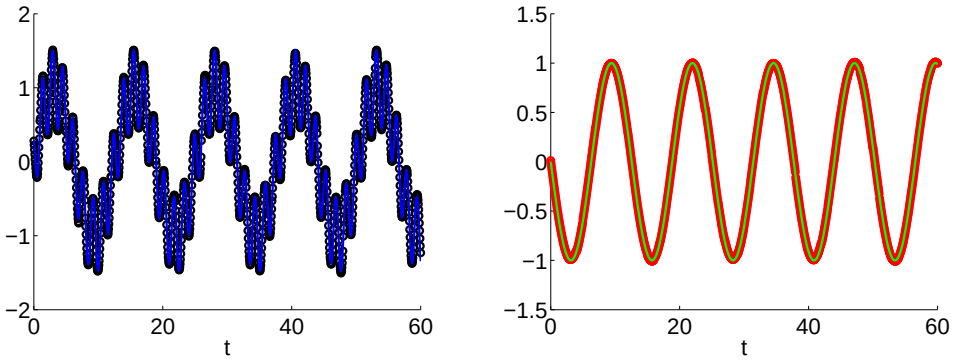
In all cases, we may compare the true state to the assimilated state using the distance between the two as a function of time, given by the Euclidean norm of their difference

$$\delta^{a,f}(t) = ||\mathbf{x}^t(t) - \mathbf{x}^{a,f}(t)||_2. \tag{60}$$

Finally, for the particle filter and the extended Kalman filter, we will compare our results to a published study on data assimilation for the point-vortex model [16]. We expect the ensemble Kalman filter to perform better than the EKF but worse than the particle filter, due to its ability to track nonlinearities in the flow but failure to incorporate non-Gaussianity.

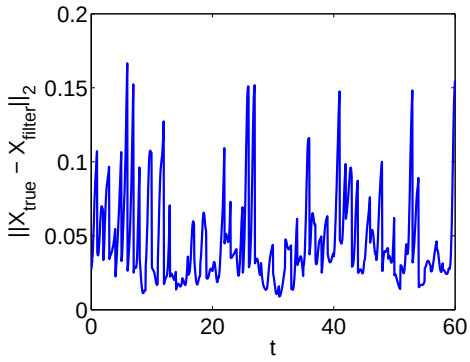
5.1.4 Validation of the EKF Filter

The results of applying the EKF to data under the conditions described for Cases 1, 2, and 3 are shown in Figures 5, 6, and 7, 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 7(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 7(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.



(a) ξ_1 vs. time.

(b) x_1 vs. time.

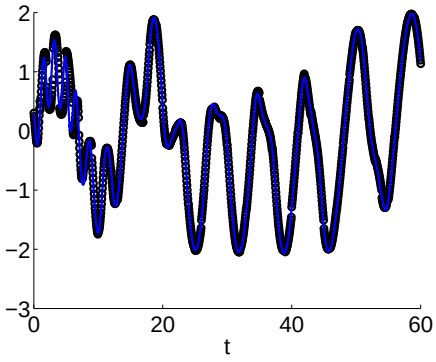


(c) $\delta^{a,f}(t)$ vs. time.

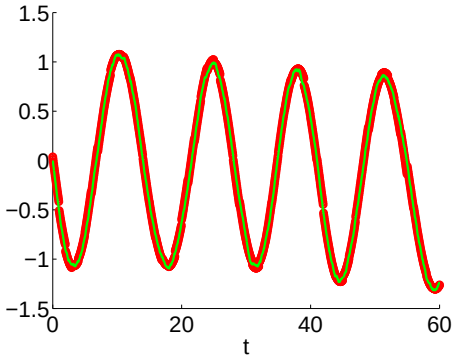
Figure 5: The true state (solid lines) and filtered state (circles) estimated using the EKF with the system fully observed. Dynamical and observational conditions correspond to **Case 1** above.

5.1.5 Validation of the EnKF Filters

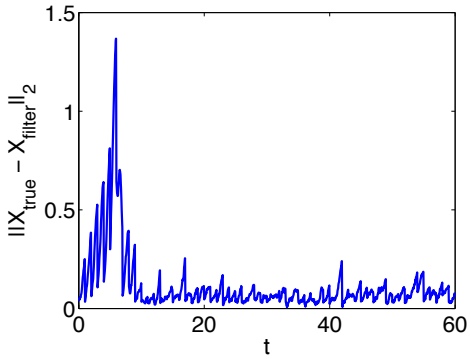
The results of applying the EnKF to data under the conditions described for Cases 1, 2, and 3 are shown in Figures 8, 9, and 10, respectively. These correspond to using the EnKF with perturbed observations for the analysis step with $N = 6$ ensemble members. The results using the EnKF were similar. We see that in



(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 2** above.

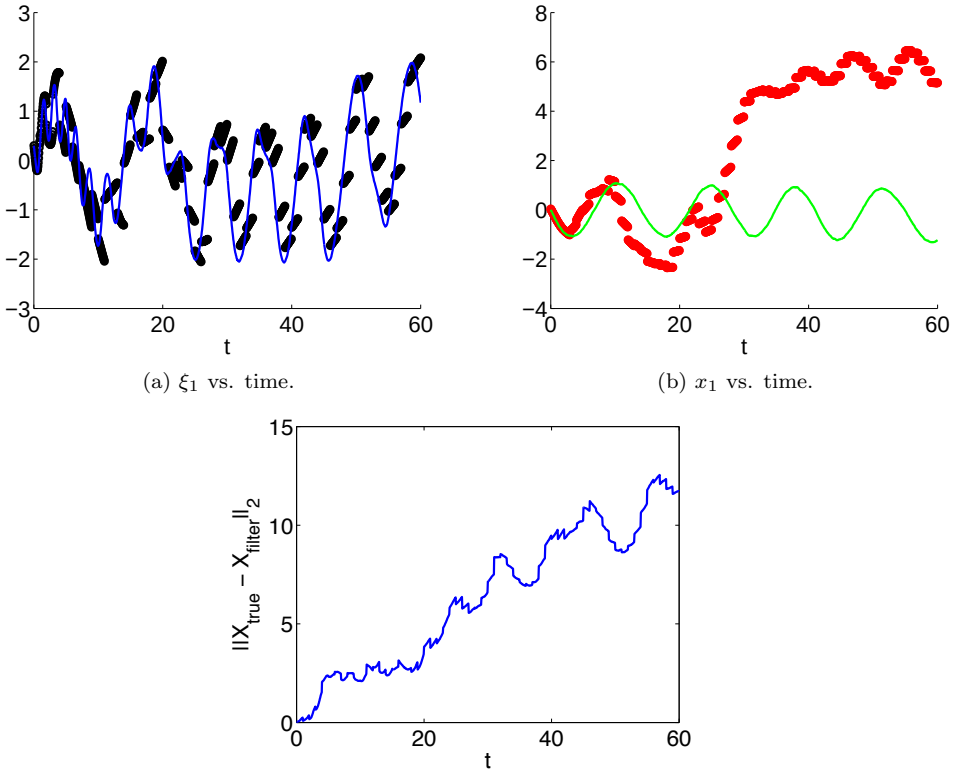
the case of no dynamical noise with full observation, the filter successfully tracks the state of the system. In the case with dynamical noise and full observation, the filter tracks the state of the system, though the positions of the vortices begin to drift over time. This results in inaccurate forecasts 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.

5.2 Phase II - Manifold Detection

Using the M function, we expect to obtain a good approximation to the analytically known stream function of the point-vortex model. For example, with two vortices, the stream function is given by

$$\psi(x, y) = \frac{1}{2}\log[(x-1)^2 + y^2] - \frac{1}{2}\log[(x+1)^2 + y^2] + \frac{1}{4}(x^2 + y^2) \tag{61}$$

where x and y are the real and imaginary parts of a ξ from Section 2.1.1 [16]. We may also compare the results from M to the results from computing finite-time Lyapunov exponents for the system [4]. As shown



(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 partially observed. Dynamical and observational conditions correspond to **Case 3** above.

in [11], these results should be similar.

Finally, we may consider the motion of drifters near the manifolds of the system. We know that these drifters should not cross the manifold, providing a sanity check for the computation of the M function.

6 Testing

6.1 Phase I - Lagrangian Data Assimilation

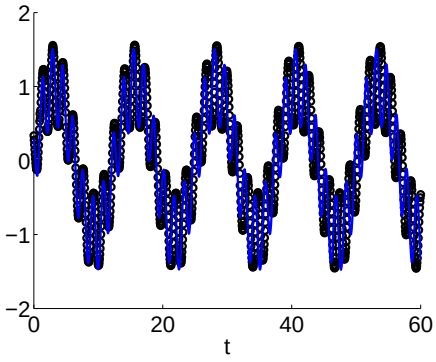
Consider the distance metric defined by (60) where we restrict ourselves to only the vortices' locations. Then as in [16], we may consider a *failure statistic*

$$\hat{f}_{\delta_{div},n}(t) = \text{ fraction of times } \delta(t) > \delta_{div} \text{ at time } t \text{ in } n \text{ trials} \tag{62}$$

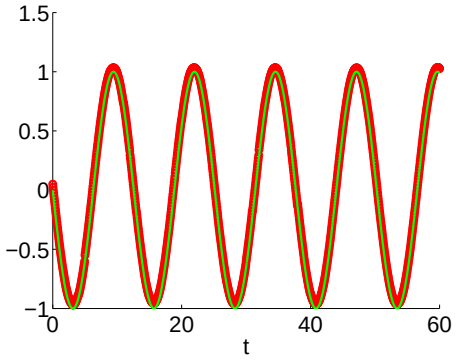
where we define a failure distance δ_{div} beyond which we consider the filter to have diverged. Thus, by using a common sample of n realizations of the system described by (1), we may obtain an estimate for the relative performance of the different data assimilation methods over the time course of data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

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

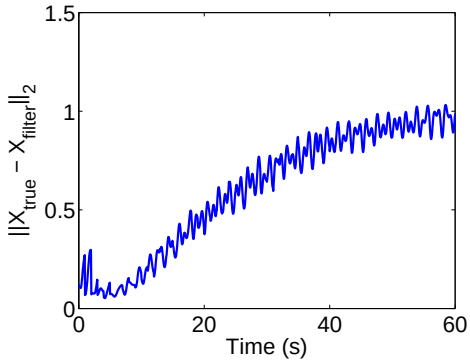
- **Position 1:** (0.3, -0.6)
- **Position 2:** (1, -0.6)



(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 EnKF with the system fully observed. Dynamical and observational conditions correspond to **Case 1** above.

- **Position 3:** (1, -1)
- **Position 4:** (2.4, -2.4)

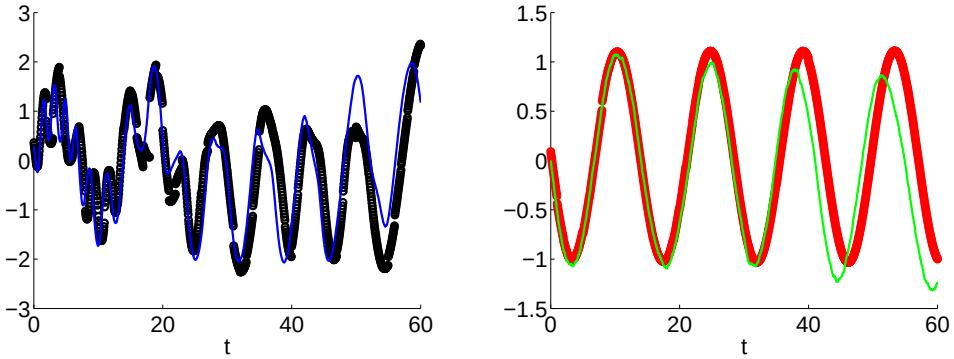
See Figure 11 for the placement of these positions on the deterministic stream function in a corotating frame. For each initial position of the drifter, we generate 500 instances of the SDE where the dynamical noise level is $\sigma = 0.02$ and the observational noise level is $\rho = 0.02$.

6.1.1 Statistics for the EKF and EnKF

Several statistics for the trial using **Position 1** (the hardest case of the four) are shown in Table 1. We see that both EnKF-based filters outperformed the EKF, giving a mean time-to-failure three times larger than the EKF. However, we also note that the standard deviation in the time to failure for the EnKF-based filters is larger. The reason for this can most easily be seen in Figure 12. We see that the EKF failed early for the majority of trials, whereas the trials are spread more uniformly over failure times for the EnKF-based filters. The EnKF-based filters still exhibit a mode near smaller failure times. This should be improved after implementing multiplicative covariance inflation and localization.

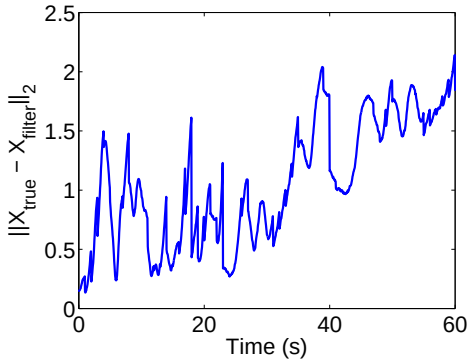
6.2 Phase II - Manifold Detection

Once we have successfully detected manifolds using the M function, we will be able to investigate how the placement of drifters with respect to the manifolds affects the accuracy of data assimilation [14]. We will



(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 EnKF with the system fully observed. Dynamical and observational conditions correspond to **Case 2** above.

Table 1: The mean time-to-failure and its standard deviation across the 500 trials using **Position 1** for the drifter. The fraction of the 500 trials that reached $t = 60$ without diverging is also listed.

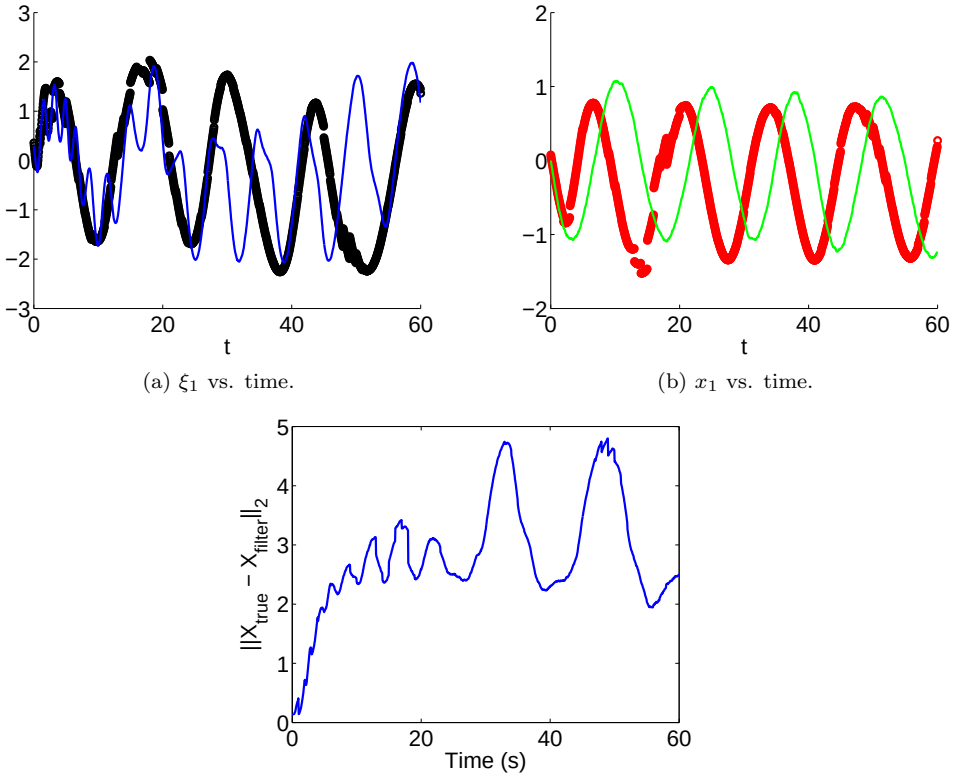
	EKF	EnKF, Pert	ETKF
Mean	8.23	25.64	25.27
Standard Deviation	9.53	17.17	17.24
Fraction Completed	0.024	0.094	0.084

characterize the performance using our knowledge of the true trajectory by computing the distance between the true and assimilated states and $\hat{f}_{\delta_{div},n}$. Thus, we can compare the performance for 'arbitrary' placement of drifters to those placed using approximate knowledge of the flow's stream function.

7 Project Schedule and Milestones

The tentative project schedule is as follows:

- Phase I
  - Produce database: now through mid-October
  - Develop extended Kalman Filter: now through mid-October



(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 EnKF with the system partially observed. Dynamical and observational conditions correspond to **Case 3** above.

  - 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

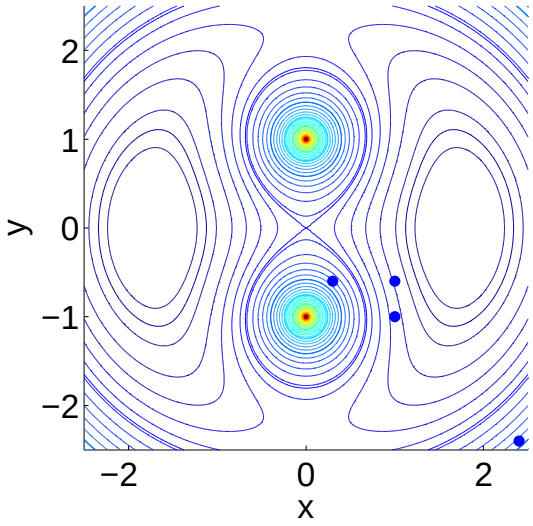


Figure 11: The four initial positions for the drifter used in testing the filters. The figure shows the deterministic stream function in a corotational frame (i.e. the frame of reference follows the corotating vortices). The four drifter positions are signified by the blue dots.

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

8 Deliverables

The deliverables for this project will include: the collection of databases used for the filter validation and testing, a suite of software for performing EKF, EnKF, and particle filtering on the stochastic point-vortex model, and a suite of software for performing manifold detection on the deterministic point-vortex model. The ensemble Kalman filter and particle filter, as well as the computation of M, will all be parallelized.

References

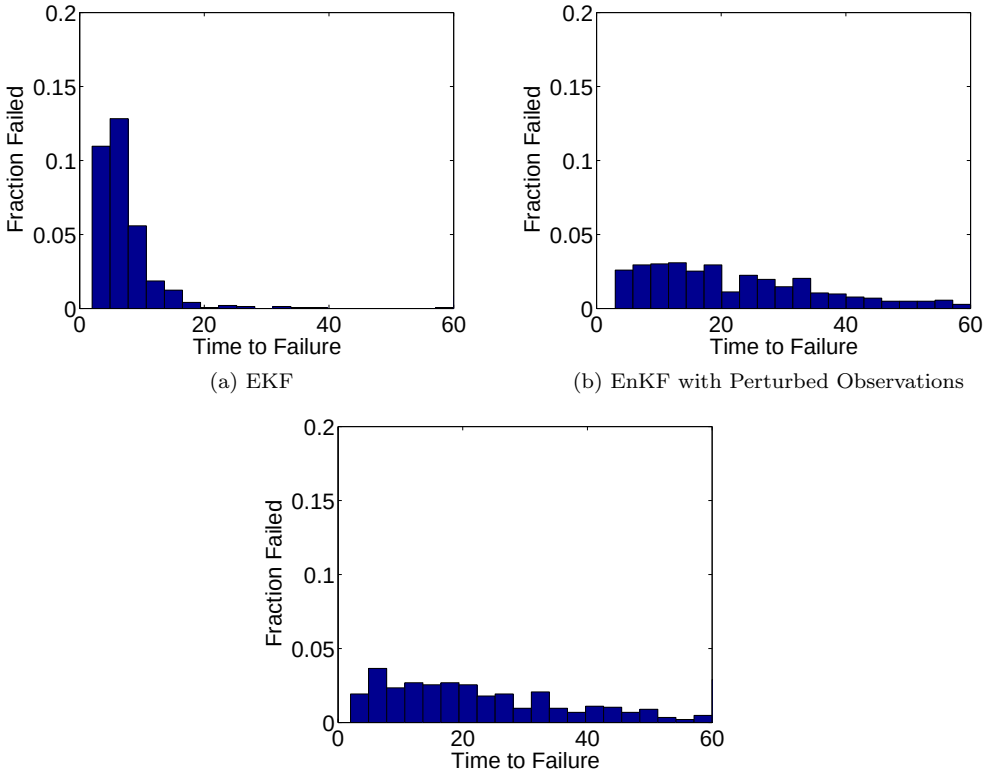
[1] J. Baehr, D. McInerney, K. Keller, and J. Marotzke. Optimization of an observing system design for the north atlantic meridional overturning circulation. *Journal of Atmospheric and Oceanic Technology*, 25(4):625, 2008.

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

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

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

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



(a) EKF

(b) EnKF with Perturbed Observations

(c) ETKF

Figure 12: Fraction of trials failed during a time interval over the course the 500 trials using **Position 1** for the drifter.

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

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

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

[9] 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.

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

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

[12] 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.

[13] 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.

[14] H. Salman, K. Ide, and C. Jones. Using flow geometry for drifter deployment in lagrangian data assimilation. *Tellus A*, 60(2):321–335, 2008.

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

[16] 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.

[17] 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.

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

A Overview of the Idealized Data Assimilation Problem

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

$$\frac{\partial p}{\partial t} + \sum_i \frac{\partial m_i p}{\partial x_i} = \frac{1}{2} \sum_{i,j} \frac{\partial^2 p(\mathbf{Q})_{ij}}{\partial x_i \partial x_j} \tag{63}$$

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) [12]. 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 (63) rarely exist. Thus, the following methods (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 (63) tractable.

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

$$p(\mathbf{x}^f|\mathbf{y}_k^o) = \frac{p(\mathbf{x}^f, \mathbf{y}_k^o)}{p(\mathbf{y}_k^o)} = \frac{p(\mathbf{y}_k^o|\mathbf{x}^f)p(\mathbf{x}^f)}{p(\mathbf{y}_k^o)} \tag{64}$$

$$= \frac{p(\mathbf{y}_k^o|\mathbf{x}^f)p(\mathbf{x}^f)}{\int p(\mathbf{y}_k^o|\mathbf{x}^f)p(\mathbf{x}^f)\, d\mathbf{x}^f}. \tag{65}$$

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 (65).