Available online at www.sciencedirect.com

ScienceDirect

ELSEVIER

Physica D 237 (2008) 1498–1506

PHYSICA D

www.elsevier.com/locate/physd

Modified particle filter methods for assimilating Lagrangian data into a point-vortex model

Elaine T. Spiller[a,*], Amarjit Budhiraja[b], Kayo Ide[c,d], Chris K.R.T. Jones[e]

[a] *Statistical and Applied Mathematical Sciences Institute, Research Triangle Park, NC, United States*
[b] *Department of Statistics & Operations Research, University of North Carolina at Chapel Hill, NC, United States*
[c] *Department of Atmospheric Sciences, University of California at Los Angeles, CA, United States*
[d] *Institute of Geophysics and Planetary Physics, University of California at Los Angeles, CA, United States*
[e] *Department of Mathematics, University of North Carolina at Chapel Hill, NC, United States*

Available online 22 March 2008

Abstract

The process of assimilating Lagrangian (particle trajectory) data into fluid models can fail with a standard linear-based method, such as the Kalman filter. We implement a particle filtering approach that affords a nonlinear estimation and does not impose Gaussianity on either the prior or the posterior distributions at the update step. Several schemes for reinitializing the particle filter, specifically tailored to the Lagrangian data assimilation problem, are applied to a point-vortex system. A comparison with the Extended Kalman Filter (EKF) for the same system demonstrates the effectiveness of particle filters for the assimilation of complex, nonlinear Lagrangian data.
© 2008 Elsevier B.V. All rights reserved.

Keywords: Particle filters; Extended Kalman filters; Lagrangian data assimilation; Resampling

1. Introduction

The issues surrounding the assimilation of data into models have recently gained prominence in almost all areas of geophysics. The quantity and quality of data has increased dramatically with ever-improving observational technologies. At the same time, our ability to run extensive simulations on ever-faster computers has enhanced the use of models. Data assimilation is the problem of estimating the optimal prediction that combines model output, with its attendant uncertainty, and observations, which will also contain errors.

Most of the traditional techniques of data assimilation are based on (linear) control theory and optimization. Their adaptation to highly nonlinear fluid flows, as in atmospheric and oceanic dynamics, presents many mathematical challenges. We focus here on one particular context in which such nonlinear behavior occurs, namely the trajectory (Lagrangian) motion in point-vortex flow. We take an approach, based on the statistical method of particle filtering, which is not based on any linearization and hence accounts well for nonlinear effects.

Lagrangian observations refer to sequences of position measurements along trajectories of (Lagrangian) tracers. Such data are abundant in geophysical systems. In contrast to the assimilation of Eulerian data, assimilation of Lagrangian data poses several complications. The reason is that most numerical models for geophysical systems are solved on a fixed grid in space or as spectral model and do not relate to the Lagrangian observations directly in terms of the model variables. Thus the conventional approach for assimilating Lagrangian data is to transform them into the Eulerian velocity data. A new approach for assimilating Lagrangian data into the fluid dynamical systems was introduced by Ide et al. [1]. The essential idea behind the approach was to augment the state space of the model by including tracer coordinates as additional model variables, thus the resulting model directly relates to the Lagrangian observations. In doing so, an augmented

* Corresponding address: 19 T.W. Alexander, P.O. Box 14006, RTP NC 27709, United States. Tel.: +1 919 685 9334.
E-mail address: espiller@samsi.info (E.T. Spiller).

error-covariance matrix contains the correlation between the Lagrangian (tracer) variables and the Eulerian (flow) variables that plays the key role in assimilating the Lagrangian data.

This approach, the so-called "Lagrangian data assimilation (LaDA)" method, can be implemented into either an Eulerian or a Lagrangian model. It was first formulated in terms of the extended Kalman filter (EKF), which computes the forecast error-covariance matrix using the tangent linear model of the system equations. Its effectiveness was shown by the application to the point-vortex model as a proof of concept [1,2]. Using the ensemble Kalman filter (EnKF) formulation in which the uncertainty of the model state is described by an ensemble [3], the LaDA method has been successfully applied to more realistic geophysical systems such as a single-layer shallow-water model [4,5] and multi-layer shallow-water model [6]. Kuznetsov et al. [2] and [4] reported that LaDA outperforms other methods that use more conventional approaches (see [7,8]). Although the LaDA method has been shown to be more efficient than conventional approaches, its formulation based on either the EKF or the EnKF approximates the model uncertainty up to second order using the error-covariance matrix. When model nonlinearity is strong and the observation period ΔT is long, this approximation may become invalid and filter divergence can occur [2,4].

The particle filter (PF), in contrast, is an approach in which a full distribution of the model uncertainty is attained by a large number of "particles" [9,13]. In other words, the resulting distributions are not approximated by Gaussians in the process of particle filtering. Highly nonlinear effects in the flow can therefore be accommodated without causing breakdown of the filter.

In this paper, we implement the PF with the LaDA into the point-vortex model and investigate the filter divergence further. Owing to small dimensionality and yet complex nonlinear (Lagrangian) dynamics, vortex models are a natural paradigm for the investigation of data assimilation system with Lagrangian or Eulerian observations [10,11]. Lagrangian and Eulerian observability can be investigated formally using the point-vortex model with one or two point vortices [12]. In this paper, we follow the work of Kuznetsov et al. [2] who carefully examined the performance of the EKF with LaDA using the point-vortex model. In this paper, we introduce a several improved PFs with LaDA tailored to the point-vortex model and compare the PF results with the EKF results.

We begin this study by reviewing the two-point vortex system and the general methodology of particle filters in Section 2. We follow this by describing how a PF (with minor modifications) can be applied to the point-vortex system in Section 3.1. In Section 3.2 we introduce the idea of occasional backtracking for the PF to overcome suspected divergence and describe three specific reinitialization schemes implemented with backtracking. In Section 4, we describe numerical experiments where we compare the performance of several PFs and the EKF applied to the point-vortex system. In that section we also present and interpret results of those experiments.

2. Background

2.1. Two-point vortex system

We begin with the description of the unperturbed two-point vortex model with vortex locations (in the complex plane), z_i ($i = 1, 2$), and a single tracer with location, ξ, given by

$$\frac{dz_1^*}{dt} = \frac{i}{2\pi}\frac{\Gamma_2}{z_1 - z_2} \tag{1}$$

$$\frac{dz_2^*}{dt} = \frac{i}{2\pi}\frac{\Gamma_1}{z_2 - z_1} \tag{2}$$

$$\frac{d\xi^*}{dt} = \frac{i}{2\pi}\frac{\Gamma_1}{\xi - z_1} + \frac{i}{2\pi}\frac{\Gamma_2}{\xi - z_2}, \tag{3}$$

where * denotes the complex conjugate. We focus on a case with circulations $\Gamma_1 = \Gamma_2 = 2\pi$, and initial vortex locations $z_1(0) = 1$, and $z_2(0) = -1$. In this system, the two vortices rotate around the center of vorticity, $z_v = (z_1 + z_2)/2 = 0$, with a constant angular velocity of $\omega = (\Gamma_1 + \Gamma_2)/(2\pi|z_1 - z_2|^2) = 1/2$. We will consider a transformation to the Lagrangian frame that fixes the point vortices at their respective initial conditions. In this case Eq. (3) becomes

$$\begin{aligned}\frac{d\xi^*}{dt} &= \frac{i}{\xi - 1} + \frac{i}{\xi + 1} - \frac{i}{2}\xi \\ &= \frac{-y}{(x-1)^2 + y^2} + \frac{-y}{(x+1)^2 + y^2} + \frac{1}{2}y \\ &\quad + i\left[\frac{x-1}{(x-1)^2 + y^2} + \frac{x+1}{(x+1)^2 + y^2} - \frac{1}{2}x\right],\end{aligned} \tag{4}$$

where x and y are the real and imaginary components of the tracer location, respectively. The flow in this system is associated with a stream function which is given by

$$\begin{aligned}\psi(x, y) &= -1/2\log[(x-1)^2 + y^2] - 1/2\log[(x+1)^2 + y^2] \\ &\quad + \frac{1}{4}(x^2 + y^2),\end{aligned} \tag{5}$$

and is plotted in Fig. 1. Here, and throughout this paper, we focus on four tracer initial conditions, $0.3 - 0.6i$, $1 - 0.6i$, $1 - i$, $2.4 - 2.4i$, and $1.75i$, which we will refer to as cases 1–4, respectively. Unperturbed tracer paths corresponding to cases 1–4 are plotted on top of the stream function in Fig. 1. These cases are representative of three different types of tracer motion: motion near and around one vortex, motion around a ghost vortex, and motion around both vortices and both ghost vortices. It is worth noting that the initial conditions in cases 1 and 2 are quite close to the separatrices which divide the complex plane into these different regions. In such cases, if this system were perturbed by noise, it is likely that a tracer could experience multiple types of motion while traversing the flow.

2.2. General methodology of particle filters

Our ultimate goal is to estimate hidden states of a nonlinear, stochastic dynamical system by combining model predictions and noisy partial observations of the system. Sequential

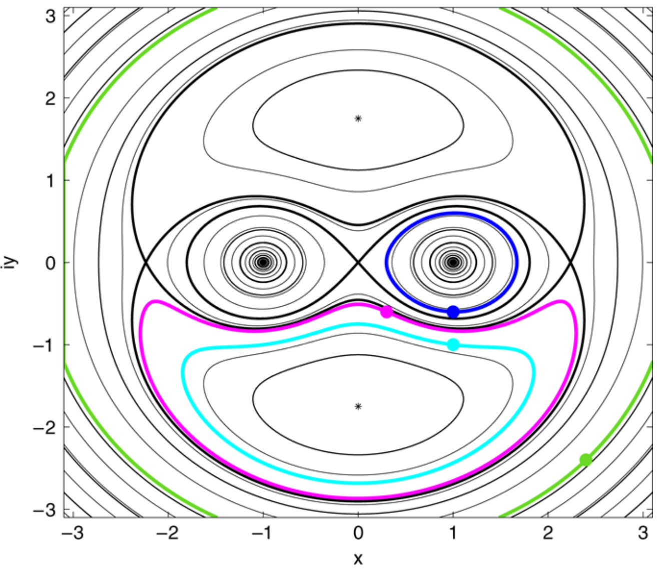

Fig. 1. Contours of the stream function corresponding to the point-vortex flow described by Eq. (4). Colored curves are tracer paths, $\xi(t)$, with dots representing different initial conditions corresponding to the four cases in consideration: $0.3-0.6\mathrm{i}$ (magenta), $1-0.6\mathrm{i}$ (blue), $1-\mathrm{i}$ (cyan), $2.4-2.4\mathrm{i}$ (green). These four cases will be further considered in Table 1. Also plotted are separatrices (thick, black curves) and ghost vortices at $0+1.75\mathrm{i}$ and $0-1.75\mathrm{i}$.

Monte Carlo (SMC) techniques allow one to approximate posterior distributions of hidden states at time t_j given all the observations up to that point without making assumptions of linearity on the dynamic model or of Gaussianity on the system/observation noise. Particle Filters are a way to implement SMC using a large number of random samples, or *particles*, to obtain discrete approximations of these distributions [13].

The problem we consider can be expressed as dynamics of state variable(s) described by stochastic differential equation(s) in the form

$$\mathrm{d}X_t = f(X_t,t)\mathrm{d}t + g(X_t,t)\mathrm{d}W_t, \tag{6}$$

where X is the state variable(s), W is a standard Wiener process, and f and g are the deterministic and random components of the evolution, respectively. We will consider the case of discrete observations collected at time instants $\{t_j; j \geq 1\}$ where the observation Y is a function of the state variables subject to observation noise

$$Y_j = \vartheta(X(t_j)) + \theta\eta_j, \tag{7}$$

where η_j's are i.i.d. $N(0,I)$ random variables and θ^2 is the variance of the observation noise.

At the jth observation, we are interested in calculating the conditional distribution of $X(t_j)$ given the observations $\{Y_k; k \leq j\}$. Here, and in the rest of the paper, we will denote the state variable vector as $x(t)$ (with the shorthand $x_j = x(t_j)$ at observation instances). We denote the density of this conditional distribution by $\pi(x_k|Y_{0,k})$ and refer to it as the filtering density. Calculation of π consists of two steps: *prediction* and *filtering*.

Prediction. In the prediction step, one computes the conditional distribution of x_j given all the prior observations $\{Y_k; k \leq j-1\}$. Using the Markov property, this calculation can be done by employing the filtering density at time t_{j-1} and the transition probability density $p_j(x_j|x_{j-1})$ of the Markov process in Eq. (6) with time step $\Delta t = t_j - t_{j-1}$. More precisely, denoting the conditional density of x_j given $\{Y_k; k \leq j-1\}$ by $\pi(x_j|Y_{0,j-1})$, one has

$$\pi(x_j|Y_{0,j-1}) = \int p_j(x_j|x_{j-1})\pi(x_{j-1}|Y_{0,j-1})\mathrm{d}x_{j-1}. \tag{8}$$

This conditional density henceforth will be referred to as the prediction density at the jth time step. In the above formula, the pdf $p_j(x_j|x_{j-1})$ describes the probability density at time t_j, and location x_j, given that the state at time t_{j-1} is x_{j-1}.

In the numerical schemes considered in this paper, the prediction and filtering densities will be approximated by discrete probability measures. In particular given a discrete probability measure approximating $\pi(x_{j-1}|Y_{0,j-1})$, which we denote as π^a_{j-1}, one can use Eq. (8) to obtain a discrete probability measure approximating $\pi(x_j|Y_{0,j-1})$ (denoted as π^p_j) as follows. For each point x_{j-1} in the support of π^a_{j-1} (we call such a point a particle for π^a_{j-1} and its probability under π^a_{j-1}, $w^a_{j-1}(x_{j-1})$, as the weight of the particles; similar terminology will be used for π^p), we simulate the SDE in Eq. (6) with initial condition x_{j-1} for the duration between observations, Δt. The terminal state $\hat{x}$ in the simulation then yields a particle for π^p_j with weight $w^p_j(\hat{x}) = w^a_{j-1}(x_{j-1})$.

Filtering. In the filtering step one uses Bayes formula to combine the prediction density and the current observation to compute the posterior density at the current state, namely the filtering density $\pi(x_j|Y_{0,j})$ — more precisely

$$\pi(x_j|Y_{0,j}) \propto p(Y_j|x_j)\pi(x_j|Y_{0,j-1}), \tag{9}$$

where $p(Y_j|x_j)$ represents the conditional density of Y_j given $x(t_j) = x_j$. Using Eq. (7) and recalling that η_j's are standard normal random variables, one can see that

$$p(Y_j|x_j) = c\exp\left[-\frac{1}{2\theta^2}|\vartheta(x_j) - Y_j|^2\right], \tag{10}$$

where c is a normalizing constant. Thus Eq. (10) can be reexpressed as

$$\pi(x_i|Y_{0,j}) \propto R(x_j,Y_j)\pi(x_j|Y_{0,j-1}), \tag{11}$$

where

$$R(x,Y) = \exp\left[\frac{\vartheta(x)\cdot Y}{\theta^2} - \frac{|\vartheta(x)|^2}{2\theta^2}\right]. \tag{12}$$

Numerically, given a discrete approximation π^p_j of $\pi(x_j|Y_{0,j-1})$, one approximates $\pi(x_j|Y_{0,j})$ using Eq. (11) with a discrete probability measure π^a_j as follows. The particles for π^a_j are exactly the same as those for π^p_j. The weight $w^a_j(x_j)$ of a typical particle for π^a_j is given by the formula

$$w^a_j(x_j) = cw^p_j(x_j)R(x_j,Y_j), \tag{13}$$

where c is a normalization constant.

The prediction and filtering steps together then give a recursive way of computing the approximation π_j^a of the filtering density $\pi(x_j|Y_{0,j})$ from π_{j-1}^a. At $j=0$, since no prior information is available, one can initialize the particle cloud by selecting N states randomly. Thus, at $j=0$, π_j^a is a discrete probability measure with equal weights at selected states.

2.3. Resampling

One of the well-known problems with the recursive numerical scheme described above is that it suffers from severe degeneracies, especially in high dimensions [13]. Typically, after a few steps, most of the weight is concentrated on very few particles, thus drastically reducing the effective sample size. A common remedy for this paucity of significant particles is to occasionally resample in order to rejuvenate the particle cloud. Since resampling introduces extraneous randomness, it is the key that resampling is not done too frequently. Typically, one has to do preliminary experimentation to choose a resampling frequency. Furthermore, to minimize extraneous randomness, it is desirable to use some variance reduction scheme for resampling. The rest of this section is devoted to explaining one such reduction scheme.

For each particle x_j in the original discrete distribution π_j^a one takes round$(Nw_j^a(x_j))$ copies of x_j. This leads to $\sum$round$(Nw_j^a(x)) = N_o$ particles. One then selects an additional $N - N_o$ particles (with replacements) from the set of particles for π_j^a with selection biases given via weights $Nw_j^a(x_j) - \text{round}(w_j^a(x_j))$. We refer the reader to Section 3 for details. More precisely, the modified scheme works as follows. Denote the resampling period by $\alpha\Delta T$. Then for the first α steps, one computes π_j^a recursively as outlined in Section 2.2. At time instant α, one modifies π_α^a by using the resampling procedure described above. This leads to a set of N particles where particles with high weights under π_α^a are represented more often than those with low weights under π_α^a. The new set of particles are then given equal weights (i.e. $1/N$) and the resulting discrete probability measure is relabeled as π_α^a. In general for j, such that $(m-1)\alpha < j \le m\alpha$, π_j^a is computed recursively as in Section 3 and $\pi_{m\alpha}^a$ further modified using the resampling scheme as described above. Thus every α time steps the filtering measure is adjusted to a uniform probability measure on N points, avoiding the degeneracy problems associated with standard PFs.

3. Particle filter schemes for the point-vortex system

The PF scheme described in Section 2.2 (hereafter referred to as the "standard particle filter") is broadly applicable and works well for many nonlinear problems, however, one typically needs to further tune the scheme for any given application. This relies on preliminary experimentation to address specific features of the nonlinearity in the state dynamics and the signal-to-noise ratio. Recall, our goal is to estimate the vortices' locations based on observations of the passive tracer location. In this problem both systematic noise (vortex locations and tracer locations experience random fluctuations) and observation noise (the "true" tracer location at an observation time is known only up to some error), are nonnegligible and play a significant role.

In the point-vortex problem considered here, f, the deterministic component of Eq. (6), is given on (the negative of) the right-hand side of Eqs. (1)–(3) transformed into the Lagrangian frame. Thus, in the problem we are considering, (Eq. (6)) is a six-dimensional system of equations describing the evolution of the real and imaginary components of the tracer location and the two vortex locations. In our problem the system noise is additive and δ-correlated. Thus $G(X_t, t)\mathrm{d}W_t = \sigma\mathrm{d}\eta$ with variance σ^2 and with $\mathrm{d}\eta \sim N(0, \mathrm{d}tI)$. The observation Y_j is a vector of the real and imaginary components of the true tracer location at t_j plus observation noise.

The standard particle filter can easily track the hidden states of this system if the number of particles, N, is large. In practice, however, we have computational restrictions on the number of particles to use in the filter. Our goal is to use a relatively small number of particles and still avoid frequent filter divergence. To this end, we slightly modify the standard PF scheme.

3.1. Modified particle filter

First, we employ resampling, specifically residual resampling, at each observation time step [14]. For the inter-observation times (i.e. the specific Δts that are used in our study), we find that $\alpha = 1$, i.e. resampling at each observation time instant, works well. We further modify the resampling procedure as follows. At time instant t_j we start with π_j^a obtained from π_{j-1}^a and Y_j as in Section 2.2. We then consider only a subset of M particles in π_{j-1}^a with the largest weights (e.g. $M = N/10$) $w_j^{(k)} \propto R(\xi_j^{(k)}, Y_j)$, where $\xi_j^{(k)}$ is the tracer location of the kth particle. Henceforth, we will refer to elements of this subset as $\tilde{x}i^{(k)}$ where $k = 1, \dots, M$. For convenience we order $\{\tilde{x}i^{(k)}\}$ from largest to smallest corresponding weights. (Note, the whole state vector x_j at time t_j is modified in this manner, so $\tilde{x}_j$ also contains $\{\tilde{z}_i(t_j)\}$, $(i = 1, 2)$.) That is, $w_j^{(1)} > \cdots > w_j^{(M)}$ and we normalize the weights so that $\tilde{w}_j^{(k)} = w_j^{(k)} / \sum_1^M w_j^{(k)}$. We then sample N particle replicates from $\{\tilde{x}i_j^{(k)}\}$. Several particles of the resampled cloud will be duplicates. Of course, the noise in the dynamics from Eq. (6) ensures that such duplicate particles at one instant t_j will lead to distinct particles at t_{j+1}.

We use $\tilde{W}_j = \{\tilde{w}_j^{(k)}\ k = 1, \dots, M\}$ to decide how many copies to use for each $\tilde{x}i_j^{(k)}$. Each particle $\tilde{x}i^{(k)}$ will have m_k replicates in the resampled cloud where the value of m_k is determined by

$$m_k = \text{round}(\tilde{w}_j^{(k)} N). \tag{14}$$

We adjust $m_1, \dots, m_M$ appropriately to ensure $\sum m_k = N$. Fig. 2 shows a typical distribution of states in a resampled particle cloud. A common difficulty in PF schemes is that occasionally the particle cloud for π_j^p will significantly deviate from the measurement Y_j. In such a case the multiplier

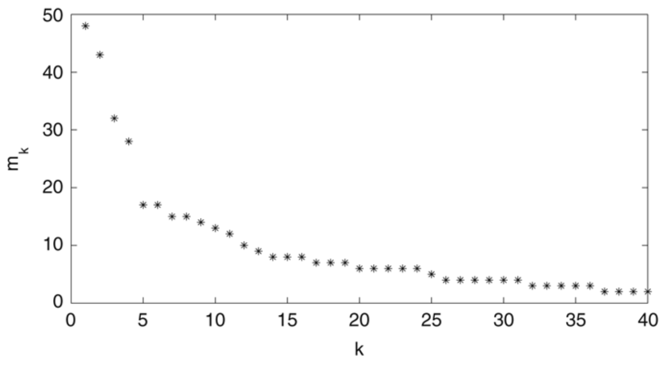

Fig. 2. Typical distribution of weights in a resampled cloud. Horizontal axis — index of particles chosen from π_j^a to reinitialize particle cloud. Vertical axis — number of particles reinitialized with the kth state.

$R(\xi_j^{(k)}, Y_j)$ in Eq. (11), particularly for low values of θ, can be extremely small for all the particles of π_j^p. The resulting weights might be so small that they numerically sum to zero. Additionally, if all the weights are very small, i.e. the predictor π_j^p does not conform with the observation Y_j, then one needs to be careful when resampling. Specifically one does not want resampling to lead to a cloud with particle replicates of only very few states. Even if particles are relatively closer to the observation than the rest of the cloud, they may still be poor estimates of the true tracer location and should not be very heavily weighted. A common procedure that is employed in such situations is that of artificial variance inflation. We multiply the observational variance parameter by $\theta_a^2 > 1$ and use

$$R_a(\xi_j^{(k)}, Y_j) = \exp\left[\frac{2\xi_j^{(k)} \cdot Y_j - |\xi_j^{(k)}|^2}{2\theta^2\theta_a^2}\right] \tag{15}$$

instead of $R(\xi_j^{(k)}, Y_j)$ in obtaining updated weights through Eq. (11). This adjustment of weights avoids numerical degeneracies.

The PF described above will henceforth be referred to as the *standard particle filter*.

3.2. Particle filters with backtracking

In any practical implementation of a filtering algorithm, one needs to periodically reinitialize the filter; however, where and how often such a reinitialization should take place is highly application dependent. For the model studied here, we now describe some reinitialization procedures that work particularly well.

In the Lagrangian point-vortex system, particle filters tend to fail, i.e. poorly track the vortices' locations if the coordinates of the particles in π_j^p that correspond to tracer locations deviate significantly from the observed tracer location. Typically there is a time lag between when such a deviation is observed and when particle filter begins to track the vortices' locations poorly. This behavior is natural since tracer dynamics are more rapid than the vortex dynamics. One can use this failure-mode to both identify the time at which the filter estimates become unreliable and as an indication that a more robust or

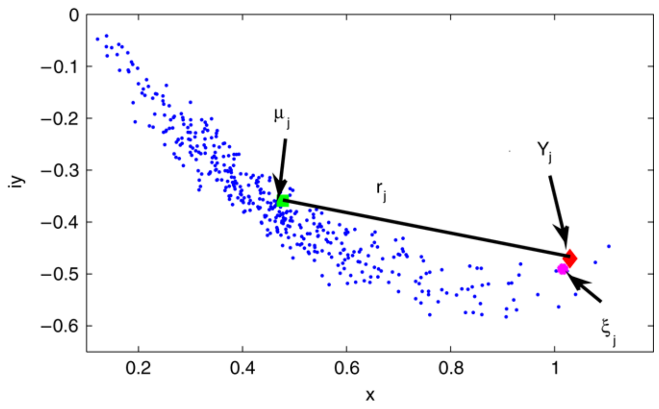

Fig. 3. Example of a predictor π_j^p deviating from the observation. In this example $\delta_j = 7 \times 10^{-6}$. The small dots are the tracer locations of the particles, μ_j is the predicted mean, Y_j is the observation, and ξ_j^t is the true tracer location.

sophisticated filter should be applied. We will explore multiple PF variations that employ correcting mechanisms when filter divergence is detected. All begin with the standard PF described in this section, but when the filter starts to track the observed states poorly, we backtrack a few steps and reinitialize the filter as we will describe. We refer to such a PF as a *backtracking particle filter* (BPF).

We begin by discussing criteria to define poor tracking. At time instant t_j, we calculate the mean vector μ_j and the covariance matrix Σ_j, for the discrete distribution π_j^p (specifically for the coordinates corresponding to the observable, tracer location). We define the distance between the mean and the observation, $r_j = Y_j - \mu_j$, and the discrepancy factor

$$\delta_j = \exp[-r_j \Sigma_j^{-1} r_j'/2]. \tag{16}$$

When δ_j is large (near one) the prior distribution is deemed to be estimating the observation well. When δ_j falls beneath a small threshold, we implement backtracking. Fig. 3 demonstrates a case where the separation factor δ_j is beneath the threshold. The reinitialization in its basic form amounts to backtracking to the t_{j-b} instant, where b is the number of steps we backtrack, doubling the number of particles in π_{j-b}^a in some manner, and recalculating the filtering distribution π_k^a for $j - b < k \leq j + 1$. At t_{j+1} we reduce the number of particles back to N and subsequently compute π_k^a as before. An important point is that the backtracking/reinitialization is implemented rarely and for most of the time steps N particles are used. We will investigate three variations of the basic reinitialization procedure described above.

3.2.1. Cloud expansion

In this method if δ_j is below the failure threshold, then we take the first N particles for the reinitialized filter at t_{j-b} as exactly those particles that constitute π_{j-b}^a. Note, however, that with different noise realizations these particles will in general lead to a different set of particles for the "new" next instant predictor π_{j-b+1}^p. Frequently, this alone corrects for a discrepancy of the form found in Fig. 3. Occasionally, however, different noise realizations alone are not enough. Thus we select additional N particles by adding independent mean zero Gaussian random variables (with small variance) to each of the observed states in the original N particles. The enlarged set of $2N$ particles is then used to define the reinitialized filter at t_{j-b}, relabeled as π^a_{j-b}. For the next $b+1$ steps, $2N$ particles are used and filtering is implemented as described at the beginning of this section. At the instant t_{j+1}, we return back to N particles and proceed in the standard manner until the next filtering breakdown is detected.

3.2.2. *Directed doubling*

In this method, we once more backtrack to t_{j-b} and reinitialize π^a_{j-b}. We first consider the "pre-sampling" version of π^a_{j-b} that is obtained from π^p_{j-b} by using Eq. (11). We then look at the M ($M = N/10$) particles with the largest weights. For each of these M particles we bias m_k particles toward the observation along the line from the tracer location of the kth particle to the observation Y_{j-b}. We space the m_k biased particles evenly along this line. To ensure that the mean of the reinitialized cloud remains the same, for each particle we bias toward the observation, we bias another an equal amount along the same line away from the observation. The numbers m_k are chosen in a manner such that the total number of particles is $2N$. Subsequently, the filter is computed as in Section 3.2.1, after reinitialization.

3.2.3. *Doubling with perturbed observations*

In this method we again begin by backtracking to t_{j-b}, and reinitialize π^a_{j-b}. In this case, however, we do not perturb the particle tracer locations, but instead perturb the Y_{j-b} observation as is often done in Kalman filter implementations [15–17]. Now we consider $\tilde{Y}^{(k)}_{j-b} = Y_{j-b} + \theta\eta$ where $k = 1, \ldots, 2N$, sampling $2N\eta$s where $\eta \sim N(0, I)$. We then use $R(x_{j-b}, \tilde{Y}_{j-b})$ to compute a new π^a_{j-b} with $2N$ particles employing Eq. (11). Subsequently, the filter is computed as in Section 3.2.1, after reinitialization.

4. Simulations and results

As a test of performance of the standard particle filter, and the backtracking particle filters we described in Sections 2.2 and 3, we perform two types of numerical experiments. In both experiments we also compare the performance of the different particle filters to that of the extended Kalman filter. For the first experiment, we test the different filtering methods dependence on initial condition by computing failure rates for each filter, for each initial condition. Note, we use the same initial conditions, cases 1–4, as shown in Fig. 1. For the second experiment we test the different filters' dependence on the time between observations for one initial condition (case 2), a case for which the EKF is known to perform poorly [2].

4.1. *Experimental setup*

As a test of these methods we perform twin identical numerical experiments of the two-point vortex model for one or more of the initial conditions cases 1–4 in Fig. 1. In every experiment we run 500 trials. For each trial we simulate one "true" run of the model. A truth run consists of the vortex and tracer locations as a function of time from a single numerical solution (realization) of the model, Eq. (6). For this (and all simulations of the model) we run the system to $T_{\text{final}} = 60$, taking N_{obs} observations. The observation period is then given by $\Delta t = T_{\text{final}}/N_{\text{obs}}$. From one observation instant to the next the states are evolved by a second-order Runge–Kutta (R–K) scheme with $\mathrm{d}t = 1/200$ [18]. For the truth run, however, we generate observations at observation instants, t_j, by taking the tracer locations there, $\xi(t_j)$, and adding i.i.d mean-zero normal random variables to both the real and imaginary parts of $\xi(t_j)$. That is, $Y_j = \xi_j + \theta\eta$ with $\eta \sim N(0, I)$. For the identical twin experiment we apply every filtering method to each truth runs using the same set of observations for each truth run. For each trial, we assimilate the vortices' locations from the truth run with the standard particle filter (PF), with the backtracking particle filters (BPFs), and with the extended Kalman filter (EKF).

Every filtering scheme consists of a filter prediction and filter analysis step, for the PFs these are described in detail in Sections 2.2 and 3.1. To clarify notation, let us define the filter prediction (denoted by superscript p) and filter analysis (denoted by superscript a) of the vortex and tracer locations. Specifically, we have

$$z_i^p(t) = \frac{1}{N}\sum_{k=1}^{N} z_i^{p,(k)}(t), \quad \text{and}$$

$$z_i^a(t_j) = \frac{1}{N}\sum_{k=1}^{M} m_k \tilde{z}_i^{p,(k)}(t_j) \quad (i = 1, 2),$$

$$\xi^p(t) = \frac{1}{N}\sum_{k=1}^{N} \xi^{p,(k)}(t), \quad \text{and} \quad \xi_j^a = \frac{1}{N}\sum_{k=1}^{M} m_k \tilde{\xi}_j^{p,(k)},$$

with m_k as defined in Eq. (14) (recall $\sum m_k = N$ and $\tilde{\xi}^{(k)}$ and $\tilde{z}^{(k)}$ as described in Section 3.1). We will use similar the same notation for the EKF case, although $z_i^p(t)$, $z_i^a(t_j)$, $\xi^p(t)$, and ξ_j^a are computed using the extended Kalman filter as described in Kuznetsov et al. [2].

In our test system the observation noise has a standard deviation of $\theta = 0.02$ (see Eq. (7)) and the system noise has a standard deviation of $\sigma = 0.02$ (see Eq. (6) and Section 3). We use a particle cloud of size $N = 400$, and in the case of the backtracking particle filters we use $2N = 800$ for the steps where backtracking/doubling is implemented. We augment the observational variance by a factor of $\theta_a^2 = 50$ (see Eq. (15)) when calculating particle weights (this value was obtained by experimentation and works well for this system) as explained in Section 3.1. At each observation step, we keep (up to) $M = N/10 = 40$ particles from which to resample the next cloud.

For the results presented here, we evolve the system in the Lagrangian frame directly, but one could easily evolve the system in the Eulerian frame. For the BPF schemes, backtracking is implemented when the cloud-observation discrepancy factor, $\delta_j < .05$ as defined in Eq. (16) (and only at observation times when backtracking has not been applied

Table 1
Experiment results comparing five different particle filters and the extend Kalman filter

	0.3 − 0.6i	1 − 0.6i	1 − i	2.4 − 2.4i
Standard PF	7.8%, na	3.0%, na	4.6%, na	12.0%, na
Standard BPF w/doubling	6.4%, (6.4)	2.6%, (3.7)	2.6%, (6.0)	9.6%, (7.3)
Cloud expanding BPF	0.4%, (5.3)	0.2%, (3.1)	0.2%, (5.4)	4.0%, (6.4)
Directed doubling BPF	0.4%, (5.4)	0.0%, (3.2)	0.0%, (5.5)	4.8%, (6.4)
Perturbed observation BPF	1.6%, (4.1)	1.6%, (2.3)	2.0%, (4.2)	7.0%, (7.1)
Extended Kalman filter	4.6%, na	78.2%, na	0.6%, na	2.0%, na

(Note that the standard BPF with doubling reinitializes π^a_{j-b} when a cloud-tracer discrepancy is detected with the same weights and particles, except with an extra copy of each particle.) Failure rates are the first number listed and correspond to the initial tracer location and the assimilation method. Also listed, in parentheses, is the average number of times backtracking is implemented, C^{avg}, for each BPF in each experiment. Note, $C^{\text{avg}} \ll N_{\text{obs}}$ in all cases.

previously). When backtracking is employed, the system is reinitialized at the $j-2$nd time instant, i.e. $b = 2$.

Recall in the cloud expansion scheme, we bias the cloud particles using i.i.d. normal random variables with mean zero and standard deviation based the distance of each particles' tracer locations from the observation. This criteria for choosing the standard deviation of the expanding perturbation is scaled by the distance between observation and the mean of particle cloud, $r_j = \mu_j - Y_j$, (as previously defined in Section 3.2 where μ_j is the mean of the predicted tracer distribution), up to some maximum that is proportional to the observation error. Specifically, the standard deviation is given by $\min(6\theta, |r_j|)$, if the tracer is not near a vortex ($|z_i^p(t_j) - \mu_j| > 0.3$), and $\min(3\theta, |r_j|)$ if the tracer is near a vortex estimate ($|z_i^p(t_j) - \mu_j| < 0.3$). We make this distinction because we do not want to perturb particles near a vortex so much that we introduce an artificial singularity (i.e. for some k, $\xi_j^{(k)} = z_i^{p,(k)}(t_j)$ where $i = 1, 2$). Such a criteria is reasonable, but one could imagine other criteria that would also work well. We should emphasize that the noise added in this scheme is only added to the tracer locations (observed state variables) and not to the vortices' locations (hidden state variables).

Of course, since we know the true vortices' states (denoted by superscript t and exactly the hidden states that we would like to infer if we applied a PF to a physical problem) we can directly measure how well the different filters are tracking the hidden states. As a test of performance we define a *failure* metric. The distance between the true vortex locations and a filter's predictions and analyses of vortex locations is given by $d_i^m(t) = |z_i^t(t) - z_i^m(t)|$, where $i = 1, 2$ and $m = p$ for prediction or $m = a$ for analysis (again we use the shorthand $d_{i,j}^a = d_i^a(t_j)$). We will refer to $d(t)$ as the (prediction or analysis) error. We deem that the assimilation has failed if, for either vortex, $d_{i,j}^a > 1$ at any observation time t_j during assimilation. In these experiments for each trial (i.e. "truth run"), we keep track of how many times backtracking is used by individual BPF schemes during assimilation, denoting this count as C_i ($i = 1, \ldots, N_{\text{trials}}$). We then calculate the average of this count, C^{avg}, over the 500 trial.

4.2. Numerical results

4.2.1. Filter dependence on initial condition

In this experiment we have $N_{\text{trial}} = 500$ "truth" runs for each of the initial conditions in cases 1–4, and we apply all the PF methods described in Section 3 and the EKF. For this experiment we use an observation period of $\Delta t = 1$. The results of this experiment are listed in Table 1. The PF and BPF methods have relatively low failure rates ($<15\%$) for all of the initial conditions. With the exception of case 4, the directed doubling and cloud expansion BPFs significantly outperform the EKF. This is likely because the transition probability density, $p_j(x_j|x_{j-1})$, used by the particle filters captures the nonlinearity of Eq. (6). Also, with the addition of occasional backtracking, doubling, and particle perturbation, these BPF methods avoid the filter divergence typically experienced by standard PF (i.e. instances when the cloud pulls away from the observation). Recall, with an unlimited number of particles, PFs applied to the point-vortex system would not diverge. These reinitialization schemes enable PFs to overcome divergence due to the discrete approximations of the system's states *without* significantly increasing in the number of particles needed throughout assimilation.

For case 4, ($\xi(0) = 2.4 - 2.4\text{i}$), the EKF outperforms all the PF methods. Note, as can be seen in Fig. 1, this is the case when the tracer is far from the vortices. In this case, the influence of noise-induced vortex drift on the tracer motion is insignificant. Thus observations of the tracer do not lend much insight into the vortex motion; hence, PFs struggle with assimilation of point-vortex systems with tracers that have this initial condition. The EKF also performs well in the other cases with the exception of case 2, ($\xi(0) = 1 - 0.6\text{i}$), where its failure rate is extremely high. In this case the EKF diverges due to the fast tracer rotation [2] while the PFs are able to handle this case naturally. It is worth noting that, for initial condition case 2, backtracking is needed infrequently for the BPF schemes as shown in Table 1. We can monitor the performance of an assimilation scheme between observation times ($t_j < t < t_{j+1}$) by calculating error, i.e. the distance between the true hidden states (vortex locations) $z_i^t(t)$ and the filter's prediction of vortex locations, $z_i^p(t)$, $i = 1, 2$. In Fig. 4 this comparison is demonstrated for initial condition case 1, $\xi(0) = 0.3-0.6\text{i}$, with observation/resampling period of $\Delta t = 2.0$. (Note, the results for only one of the two vortices are plotted.) For this experiment we use $N_{\text{trials}} = 100$ and average $d^p(t)$ and d_j^a over these trials. For the case the EKF has a 21% failure rate while the standard PF has a 6% failure rate. When the EKF diverges, there tends to be large jumps between the predicted error and the analysis error. Note also, when an EKF diverges these errors can grow quite large ($d(t) > 5$) and tend to grow larger as time increase.

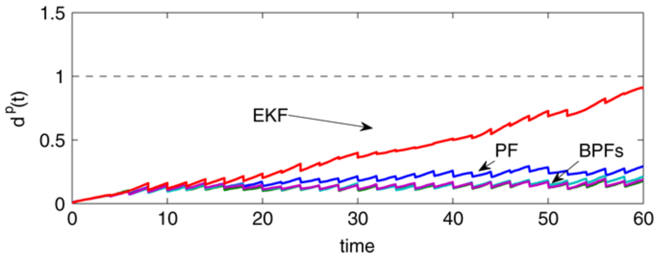

Fig. 4. Plotted above is the average (over 100 trials) $d_j^p(t)$ for each assimilation method versus time. The EKF and standard PF are labeled. The other lines, which overlap significantly, correspond to the cloud expansion, directed doubling, and perturbed observation BPFs.

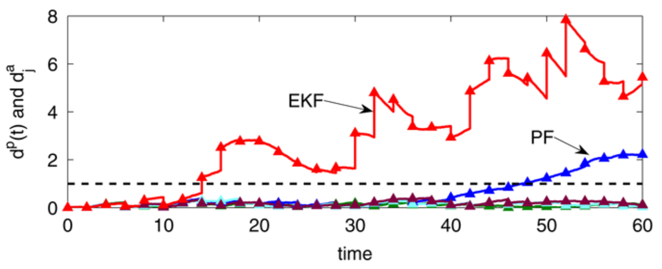

Fig. 5. Each line above represents the predicted error, $d^p(t)$, for a different assimilation scheme (the EKF and each PF scheme). The triangles plotted on top of each PF curve represent the corresponding error, d_j^a, between the filter analyses and the true vortex states (note, these are plotted at observation times). The standard PF and the EKF have failed in this case, i.e. their errors exceeds 1, while the BPFs (whose lines overlap) all performed well.

A specific instance of this behavior is plotted in Fig. 5. This kind of behavior accounts for the linear trend in the average EKF error curve plotted in Fig. 4.

In contrast, for PFs the predicted error and analysis error tend to grow together when the filter fails. Once the filter diverges, the particle cloud (typically) no longer conforms well to the observed tracer locations. That is, none of the particles in the cloud are good estimates of the observable state variables. Thus the filtering step losses its mechanism to distinguish the particles that are good estimates of the hidden state variables from those that are poor estimates. The filtering/resampling step will still select a subset of the particles to run through the dynamics and use as predictions at the next observation time, but effectively observations no longer inform this decision. It is as if we are making predictions based on the particle cloud without assimilation. Note that the error in cases where PFs diverge does not experience the same sort of growth that it does when the EKF diverges. Fig. 5 shows a specific case where the standard PF failed in this manner, but the BPF methods were successful at tracking the vortex locations.

4.2.2. *Filter dependence on observation period*

As a second test of these methods we simulate the point-vortex model with the initial tracer location $\xi(0) = 1 - 0.6\mathrm{i}$. This is a case where the EKF performs quite poorly (see Section 4.2.1 and Table 1) and the goal of the experiment is to test the performance of each assimilation method based on the frequency of observations.

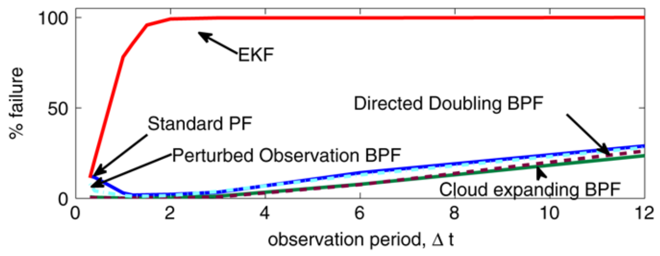

Fig. 6. Failure rate versus observation period for an experiment comparing the different assimilation methods for a point-vortex system with an initial condition of $\xi(0) = 1 - 0.6\mathrm{i}$. The curves are labeled according to the assimilation scheme implemented.

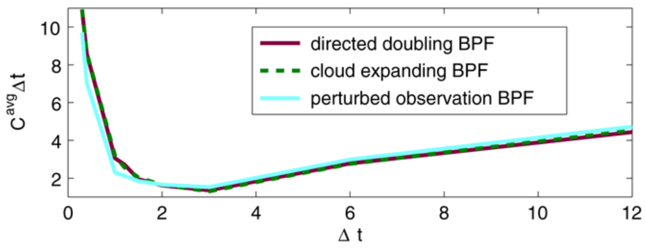

Fig. 7. Plotted above is the product of the average backtracking count and the observation frequency versus the observation frequency for the BPFs. The quantity $C^{\mathrm{avg}}\Delta t \propto C^{\mathrm{avg}}/N_{\mathrm{obs}}$.

In this experiment we vary the observations period Δt from $\Delta t = 0.4$ to $\Delta t = 12$, again with a time step of $\mathrm{d}t = 1/200$ between observations. For each sampling period Δt we perform 500 trials and calculate the filter analysis error, d_j^a, i.e. the distance between filtered vortex estimates and the true vortex values. We use same failure criterion as the previous experiment. The results of this experiment are plotted in Fig. 6.

It is remarkable that all of the PF methods achieve relative low failure rates even as the observation period gets very large. This plot in particular demonstrates the effectiveness of PFs over EKFs in assimilation of this strongly nonlinear point-vortex system. As anticipated, the EKFs performance improves as the observation period gets small. It is also worth noting that all the PFs implemented in this experiment (standard, directed doubling, cloud expansion, and perturbed observations) see minimum failure rates with sampling periods between $2 \geq \Delta t \geq 1$. At first glance it might seem counterintuitive that the PFs' performance decreases as the sampling period decreases from two towards zero (i.e. as N_{obs} gets large). This phenomenon is likely due to resampling at every observation, keeping $\alpha = 1$ (recall, the resampling period is $\alpha\Delta t$) regardless of Δt. That is, when there are many observations, a small resampling period is unnecessary, and in this case, detrimental to the performance of the particle filters since the resampling procedure adds spurious noise to the system approximations.

We can further study the relationship between observation/resampling period and the efficiency of BPFs. In Fig. 7 we consider the ratio of the average number of times backtracking is implement $C^{\text{avg}}(\Delta t)$ to the number of observations. Here we see that the BPFs for this system are most efficient, require the fewest instances of backtracking per observation, for $2 \leq \Delta t \leq 3$. It is particularly interesting to focus on the perturbed observation PF. Recall, for this BPF, the cloud itself is never perturbed (as it is for the other BPFs). For the observation period $\Delta t = 2$, this BPF applied to the two-point vortex system is both efficient and achieves a low failure rate. This suggests that this observation period is enough time to allow the random dynamics to spread cloud sufficiently so frequent reinitialization is not needed. With this period the cloud does not disperse too much, however, so it still adequately approximates the true system and thus achieves a low failure rate.

In contrast for small Δts, the cloud does not have much opportunity to spread before it is resampled and too frequent resampling deteriorates filter performance as mentioned previously. As filter performance decreases, backtracking will be implemented more frequently in an attempt to overcome poor filter performance. As one can see in Fig. 6, frequent backtracking with cloud perturbation can still achieve very low failure rates although it becomes computationally expensive. At the other extreme of very large observation periods, performance is limited by the discrete nature of the approximate posterior distributions. For long observation times the particle cloud can spread significantly between observations. This effectively reduces the number of particles that has an accurate approximation of the true state at any observation time. Naturally, since the filter performance deteriorates in this case, backtracking will be implemented more frequently for large Δt as seen in Fig. 7.

In summary, the backtracking particle filters introduced in this paper and applied to the two-point vortex system are an effective assimilation tool for this complex nonlinear system. The cloud expanding BPF and the directed doubling BPF both achieved lowest failure rates (for nearly every case) in the experiment testing dependence of initial condition. In the experiment testing dependence on observation period, all particle filter methods outperformed the EKF.

Typically, for a given application, any data assimilation method requires some application-specific tuning to perform well. The ideas we introduced here for particle filters in the context of the point-vortex model — specifically backtracking to a previous observation time t_{j-b}, reinitializing the filter, doubling the number of particles, and running assimilation forward until t_j — are effective at overcoming filter divergence, and general enough to be applied to other nonlinear physical systems.

Acknowledgements

This research was supported in part by the Army Research Office Grant W911NF-0-1-0080 (Budhiraja), the Office of Naval Research N00014-04-1-0191 (Ide), the Office of Naval Research N00014-03-1-0174 (Jones), and the National Science Foundation No. DMS-0112069 (Spiller).

References

[1] K. Ide, L. Kuznetsov, C.K.R.T. Jones, Lagrangian data assimilation for point vortex systems, J. Turbul. 3 (2002) http://www.tandf.co.uk/journals/title/14685248.asp.
[2] L. Kuznetsov, K. Ide, C.K.R.T. Jones, A method for assimilation of Lagrangian data, Mon. Weather Rev. 131 (2003) 2247–2260.
[3] E. Evensen, Sequential data assimilation with a nonlinear quasi-geostrophic model using Monte Carlo methods to forecast error statistics, J. Geophys. Res. 99 (1994) 10143–10162.
[4] H. Salman, L. Kuznetsov, C.K.R.T. Jones, K. Ide, A method for assimilating Lagrangian data into a shallow-water equation ocean model, Mon. Weather Rev. 134 (2006) 1081–1101.
[5] H. Salman, K. Ide, C.K.R.T. Jones, Using flow geometry for drifter deployment in Lagrangian data assimilation, Tellus 60 (2) (2008) 321–335.
[6] G. Vernieres, K. Ide, C.K.R.T. Jones, Lagrangian data assimilation in Gulf of Mexico, Ocean Modelling (in press), preprint.
[7] A. Molcard, L. Piterbarg, A. Griffa, T. Özgökmen, A. Mariano, Assimilation of drifter positions for the reconstruction of the Eulerian circulation field, J. Geophys. Res. 108 (2003).
[8] T. Özgökmen, A. Molcard, T. Chin, L. Piterbarg, A. Griffa, Assimilation of drifter positions in primitive equation models of midlatitude ocean circulation, J. Geophys. Res. 108 (2003).
[9] A. Budhiraja, L. Chen, C. Lee, A survey of numerical methods for nonlinear filtering problems, Physica D 230 (2007) 27–36.
[10] K. Ide, M. Ghil, Extended Kalman filtering for vortex systems. Part I. Methodology and point vortices, Dynam. Atmos. Oceans 27 (1997) 301–332.
[11] K. Ide, M. Ghil, Extended Kalman filtering for vortex systems. Part II. Rankine vortices, Dynam. Atmos. Oceans 27 (1997) 335–355.
[12] A.J. Krener, Eulerian and Lagrangian observability of point vortex flows, 2007 (submitted for publication), preprint.
[13] A. Doucet, N. de Freitas, N. Gordon, Sequential Monte Carlo Methods in Practice, Statistics for Engineering and Information Science, Springer-Verlag, New York, 2001.
[14] J. Liu, R. Chen, Sequential Monte Carlo methods for dynamic systems, J. Amer. Statist. Assoc. 93 (1998) 1032–1044.
[15] P. Houtekamer, J. Derome, Methods for ensemble prediction, Mon. Weather Rev. 123 (1995) 2181–2196.
[16] G. Burgers, P. van Leeuwen, G. Evensen, Analysis scheme in the ensemble Kalman filter, Mon. Weather Rev. 126 (1998) 1719–1724.
[17] P.L. Houtekamer, H.L. Mitchell, Data assimilation using an ensemble Kalman filter technique, Mon. Weather. Rev. 126 (1998) 796–811.
[18] J. Hansen, C. Penland, Efficient approximate techniques for integrating stochastic differential equations, Mon. Weather. Rev. 134 (2006) 3006–3014.