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

David Darmon, AMSC
Kayo Ide, AOSC, IPST, CSCAMM, ESSIC
Background

Data Assimilation

Iterative process

Diagram:

- Forecast
- Analysis
- Iterative process
Background

Data Assimilation

True System Dynamics

\[ dx^t = M(x^t, t)dt + d\eta^t \]

\begin{align*}
  x^t & \quad - \text{system state} \\
  \eta^t & \quad - \text{Brownian motion} \\
  M(\cdot, \cdot) & \quad - \text{deterministic evolution operator}
\end{align*}
Background

Data Assimilation

Observations

\[ y^o_k = h(x^t(t_k)) + \epsilon^t_k \]

\( y^o_k \) – observation of system at time \( t_k \)

\( \epsilon^t_k \) – Gaussian noise

\( h(\cdot) \)– observation operator
Background

Data Assimilation

Forecast and Analysis

Evolve a forecasted state forward

\[ \frac{d}{dt} x^f = f(x^f, t, \eta^f) \]

Perform analysis step upon receiving observation somehow

\[ x^a = g(x^f, y_k^o, \epsilon_k^t) \]
Background

Lagrangian Data Assimilation

Consider joint state of flow \((F)\) and drifters \((D)\)

\[
x^t = \begin{pmatrix} x^t_F \\ x^t_D \end{pmatrix}
\]
Background

Model

Point-Vortex (in complex plane)

\[
\frac{dz^*_j}{dt} = \sum_{j'=1, j' \neq j}^{N_v} \frac{i}{2\pi} \frac{\Gamma_{j'}}{z_j - z_{j'}}, \quad j = 1, \ldots, N_v
\]

\[
\frac{d\xi^*_k}{dt} = \sum_{j=1}^{N_v} \frac{i}{2\pi} \frac{\Gamma_j}{\xi_k - z_j}, \quad k = 1, \ldots, N_d
\]

\(z_j\) – \(j^{th}\) vortex location

\(\xi_k\) – \(k^{th}\) drifter location

\(\Gamma_j\) – vorticity of \(j^{th}\) vortex
Background
Model
Demo
Background

Observing System Design

Deploying drifters is expensive
Obtaining measurements is difficult

Q: Can we design a system such that we optimally place the drifters to determine the location of the vortices?
Background

Observing System Design

Deploying drifters is expensive
Obtaining measurements is difficult

Q: Can we design a system such that we optimally place the drifters to determine the location of the vortices?
Phase I

- Forecast
- Analysis
Approach, Phase I

Lagrangian Data Assimilation

Fokker-Planck Equation, Forecast

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

\( p \equiv p(x^t, t) \) – probability density of \( x^t \)

\( Q \) – covariance matrix from SDE
Approach, Phase I

Lagrangian Data Assimilation

Bayes’s Theorem, Analysis

\[ p(x^f | y^o_k) = \frac{p(y^o_k | x^f)p(x^f)}{\int p(y^o_k | x^f)p(x^f) \, dx^f} \]

- \( p(x^f | y^o_k) \) – posterior distribution
- \( p(y^o_k | x^f) \) – likelihood
- \( p(x^f) \) – prior distribution
Approach, Phase I

Fidelity to Solution of Fokker-Planck Equation and Bayes' Extended Kalman Filter

Ensemble Kalman Filter

Particle Filter

Computational Simplicity
Approach, Phase I

Lagrangian Data Assimilation

Extended Kalman Filter (EKF)

Generalization of Kalman filter to nonlinear equations

Gaussian assumptions on dynamical and observational noise

Represent pdf by mean and covariance matrix
Approach, Phase I

Lagrangian Data Assimilation
Extended Kalman Filter (EKF)

\[ x^f \] – mean state
\[ P^f \] – covariance matrix
Approach, Phase I

Lagrangian Data Assimilation
Extended Kalman Filter (EKF), Forecast

\[
\frac{d}{dt}x^f = M(x^f, t)
\]

\[
\frac{d}{dt}P^f = M(t)P^f + P^f M^T(t) + Q
\]

\(Q\) – covariance matrix from SDE

\(M(t) = J[M(x, t)]|_{x=x^f}\) - Jacobian of \(M\)
Approach, Phase I

Lagrangian Data Assimilation

Extended Kalman Filter (EKF), Forecast

\[
\begin{align*}
\frac{d}{dt} x^f &= M(x^f, t) \\
\frac{d}{dt} P^f &= M(t)P^f + P^f M^T(t) + Q
\end{align*}
\]

Q – covariance matrix from SDE

\[ M(t) = \left. J[M(x, t)] \right|_{x=x^f} \] - Jacobian of M
Approach, Phase I

Lagrangian Data Assimilation

Extended Kalman Filter (EKF), Forecast

\[
\frac{d}{dt} \mathbf{x}^f = \mathbf{M}(\mathbf{x}^f, t)
\]

\[
\frac{d}{dt} \mathbf{P}^f = \mathbf{M}(t)\mathbf{P}^f + \mathbf{P}^f \mathbf{M}^T(t) + \mathbf{Q}
\]

\( \mathbf{Q} \) – covariance matrix from SDE

\( \mathbf{M}(t) = J[M(\mathbf{x}, t)]|_{\mathbf{x} = \mathbf{x}^f} \) - Jacobian of \( \mathbf{M} \)
Approach, Phase I

Lagrangian Data Assimilation

Extended Kalman Filter (EKF), Analysis

\[
x_k^a = x^f(t_k) + K_k(y_k^o - h_k(x^f(t_k)))
\]

\[
P_k^a = (I - K_k H_k)P^f(t_k)
\]

\[
K_k = P^f(t_k)H_k^T(H_k P^f(t_k)H_k^T + R_k^o)^{-1}
\]

\[R^o\] – covariance matrix from observation

\[H_k = J[h(x, t_k)]|_{x=x^f}\] - Jacobian of \(h_k\)
Approach, Phase I

Lagrangian Data Assimilation

Extended Kalman Filter (EKF), Analysis

\[
x^a_k = x^f(t_k) + K_k (y^o_k - h_k(x^f(t_k)))
\]

\[
P^a_k = (I - K_k H_k) P^f(t_k)
\]

\[
K_k = P^f(t_k) H_k^T (H_k P^f(t_k) H_k^T + R^o_k)^{-1}
\]

\(R^o\) – covariance matrix from observation

\(H_k = J[h(x, t_k)]_{x=x^f}^{-1}\) - Jacobian of \(h_k\)
Approach, Phase I

Lagrangian Data Assimilation
Extended Kalman Filter (EKF), Analysis

\[
x^a_k = x^f(t_k) + K_k(y^o_k - h_k(x^f(t_k)))
\]

\[
P^a_k = (I - K_k H_k) P^f(t_k)
\]

\[
K_k = P^f(t_k) H_k^T (H_k P^f(t_k) H_k^T + R^o_k)^{-1}
\]

\(R^o\) – covariance matrix from observation

\(H_k = J[h(x, t_k)]\)\(\big|_{x=x^f}\) - Jacobian of \(h_k\)
Approach, Phase I

Lagrangian Data Assimilation

Extended Kalman Filter (EKF), Analysis

\[ \mathbf{x}_k^a = \mathbf{x}_k^f(t_k) + \mathbf{K}_k(\mathbf{y}_k^o - h_k(\mathbf{x}_k^f(t_k))) \]
\[ \mathbf{P}_k^a = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_k^f(t_k) \]
\[ \mathbf{K}_k = \mathbf{P}_k^f(t_k)\mathbf{H}_k^T(\mathbf{H}_k\mathbf{P}_k^f(t_k)\mathbf{H}_k^T + \mathbf{R}_k^o)^{-1} \]

\( \mathbf{R}_k^o \) – covariance matrix from observation

\[ \mathbf{H}_k = \mathbf{J}[h(\mathbf{x}, t_k)]_{\mathbf{x} = \mathbf{x}_k^f} \] - Jacobian of \( h_k \)
Approach, Phase I

Lagrangian Data Assimilation

Extended Kalman Filter (EKF)

Weakly assumes nonlinearity

Assumes Gaussianity

**NOT** valid for nonlinear systems
Approach, Phase I

Lagrangian Data Assimilation

Ensemble Kalman Filter (EnKF)

Approximate pdf by an ensemble of particles:

\[ \{x_i^f\}_{i=1}^N \]

Forecast: Evolve particles forward using SDE

Analysis: Perform Kalman-like analysis
Approach, Phase I

Lagrangian Data Assimilation

Ensemble Kalman Filter (EnKF)

Computing (ensemble) moments

\[ \bar{x}_f^e = \frac{1}{N} \sum_{i=1}^{N} x_i^f. \]

\[ P_{e e}^f = \frac{1}{N-1} \sum_{i=1}^{N} (x_i^f - \bar{x}_f^e)(x_i^f - \bar{x}_f^e)^T. \]
Approach, Phase I

Lagrangian Data Assimilation

Ensemble Kalman Filter (EnKF), Analysis

*Stochastic*: Generate ensemble of observations and perform Kalman-like analysis on combined ensemble

*Deterministic*: Square root filters
Approach, Phase I

Lagrangian Data Assimilation

Ensemble Kalman Filter (EnKF)

Better captures nonlinearity (in forecast)
(up to ensemble approximation)

Still assumes Gaussianity (in analysis)

**NOT** valid for nonlinear systems
Approach, Phase I

Lagrangian Data Assimilation

Particle Filter

Approximate pdf by an ensemble of weighted particles:

\[ \{x_i^f(t)\}_{i=1}^N \quad \{w_{i,k}\}_{i=1}^N \]

Forecast: Evolve particles forward using SDE

Analysis: Perform full Bayesian update at analysis
Approach, Phase I

Lagrangian Data Assimilation

Particle Filter, Analysis

\[ y^o_k = h(x^t(t_k)) + \epsilon^t_k \]

- \( y^o_k \) – observation of system at time \( t_k \)
- \( \epsilon^t_k \) – Gaussian noise
- \( h(\cdot) \) – observation operator
Approach, Phase I

Lagrangian Data Assimilation

Particle Filter, Analysis

Likelihood of Observation:

\[ p(y^o_k|x^f(t_k)) = \]
\[ C_1 \exp \left( -\frac{1}{2}(y^o_k - h(x^f(t_k)))^T (R^o)^{-1}(y^o_k - h(x^f(t_k))) \right) \]
Approach, Phase I

Lagrangian Data Assimilation
Particle Filter, Analysis

Likelihood of Observation:

\[ p(y|x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp \left( -\frac{1}{2\sigma^2} (y - h(x))^2 \right) \]
Approach, Phase I

Lagrangian Data Assimilation

Particle Filter, Analysis

Weight of particle $i$ at analysis step $k$:

$$w_{i,k}$$

Bayesian Update:

$$w_{i,k} = C_2 w_{i,k-1} p(y_k^o | x_i^f(t_k))$$
Lagrangian Data Assimilation

Particle Filter

Better captures nonlinearity (up to ensemble approximation)

Does not assume Gaussianity on posterior or prior
Approach, Phase I

Lagrangian Data Assimilation

Particle Filter

**BUT**

Requires a large number of particles

For large, nonlinear systems, requires frequent resampling
Phase II
Approach, Phase II

Manifold Detection for Observing System Design

Stream functions

$$\psi(x, y) = C$$

Consider the trajectories in phase space

Trajectories will lie along the level curves of the stream function for steady flows

‘Streamlines’

Trajectories may not cross streamlines

Uniqueness
Stream function in corotating frame
Approach, Phase II

Manifold Detection for Observing System Design

Mendoza and Mancho’s Lagrangian Descriptor $M$

\[
M(x_D^t, t^*) = \int_{t^*-\tau}^{t^*+\tau} \left( \sum_{i=1}^{n} \left( \frac{dx_D(t)}{dt} \right)^2 \right)^{1/2} dt
\]
Approach, Phase II

Manifold Detection for Observing System Design

Mendoza and Mancho’s Lagrangian Descriptor $M$

Approach, Phase II

Manifold Detection for Observing System Design

Mendoza and Mancho’s Lagrangian Descriptor $M$

Improve Phase I by using knowledge of dynamics
Implementation

Develop serial code for EKF, EnKF, and Particle Filter in MATLAB on MacBook Pro

Parallelize the EnKF and particle filters

Parallelize computation of $M$

Use MATLAB Parallel Computing Toolbox
Databases

Phase I

Archived numerical solutions to point-vortex model

Generate using stochastic 4\textsuperscript{th}-order Runge-Kutta method

Phase II

No databases: completely model dependent
Validation, Phase I

Validate filter by varying noise

Three stages

Stage 1:
Vortices’ positions known, low noise

Stage 2:
Vortices’ positions unknown, low noise

Stage 3:
Vortices’ positions unknown, realistic noise
Validation, Phase I

Validate filter by varying noise

Validation Metric

$$\delta^{a,f}(t) = \|x^t(t) - x^{a,f}(t)\|_2$$
Validation, Phase I

Validate filter by comparison

Compare to published studies with

\[ N_v = 2 \]

\[ N_d = 1 \]
Validation, Phase II

Validate $M$ by comparison

Compare $M$ to analytically known stream function

Compare $M$ to finite-time Lyapunov exponents
Testing, Phase I

Failure statistic

Compare EKF, EnKF, and particle filter across databases using

\[ \hat{f}_{\delta_{div}, n}(t) = \]

fraction of times \( \delta(t) > \delta_{div} \) at time \( t \) in \( n \) trials
Testing, Phase II

Failure statistic

Compare EKF, EnKF, and particle filter, including decision of initial drifter position

\[ \hat{f}_{\delta_{div},n}(t, x_D(0)) = \]

fraction of times \( \delta(t) > \delta_{div} \) at time \( t \) in \( n \) trials
Project Schedule and Milestones

- **Phase I**
  - Produce database: now through mid-October
  - Develop extended Kalman Filter: now through mid-October
  - Develop ensemble Kalman Filter: mid-October through mid-November
  - Develop particle filter: mid-November through end of January
  - Validation and testing of three filters (serial): Beginning in mid-October, complete by February
  - Parallelize ensemble methods: mid-January through March

- **Phase II**
  - Develop serial code for manifold detection: mid-January through mid-February
  - Validate and test manifold detection: mid-February through mid-March
  - Parallelize manifold detection algorithm: mid-March through mid-April
Project Schedule and Milestones

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

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

Database of sample trajectories used for validation and testing

Suite of parallelized software for performing EKF, EnKF, and particle filtering for point-vortex model

Software to compute $M$ for point-vortex model


Questions???