Particle Filters for Lagrangian Data Assimilation

Elaine Spiller

Marquette University

IMAGe Workshop on Data Assimilation & Climate Research
June 24, 2010

Introduction

Began thinking about PF as a postdoc at SAMSI

Been lucky to work with

- A. Budhiraja (U NC) , K. Ide (U Maryland), CKRT. Jones (U NC)

and more recently

- Amit Apte (Tata Institute), and Sherry Scott (Marquette U)

Brief outline

- General background on Lagrangian DA & particle filters
- Applied to point-vortex model
- Some problems and potential solutions for PFs

Model:

$\mathbf{x} \in \mathbb{R}^N$ — state vector containing all relevant dynamic info (e.g. flow velocity, temperate, salinity, etc)

$$d\mathbf{x} = M(\mathbf{x}, t)dt + G(\mathbf{x}, \mathbf{t})\mathbf{dW_t}$$

M — deterministic model of state evolution

$G(X_t, t)dW_t$ — stochastic component

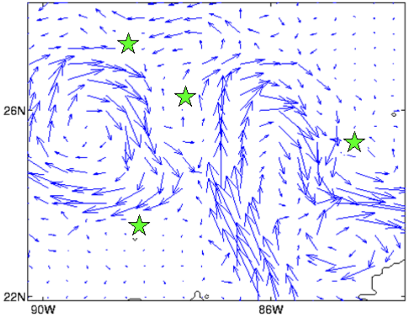
Note: M is often nonlinear

Observations:

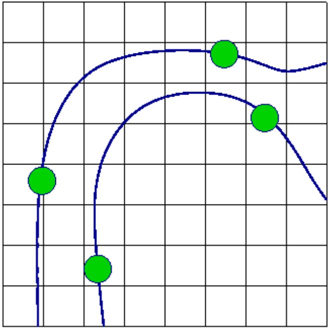
$Y_j^o = H[\mathbf{x_j^t}] + \epsilon_\mathbf{j}$

H — observation operator

ϵ_j — observation error



- Much data in the ocean is Lagrangian in nature

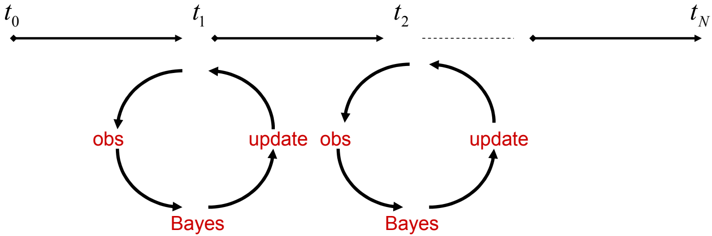


Problem: Lagrangian observations from drifters and floats do not give data in terms of model variables

Solution: Include drifter coordinates into model

- Direct method of assimilating Lagrangian data (Kuznetsov, Ide, Jones, 2003)

Bayesian view of sequential DA – estimate flow field



$x =$ state

$Y =$ obs

$$p(x \mid Y) \propto p(Y \mid x)\, p(x)$$

Key question: how do we obtain the distributions on RHS?

Point-vortex flows (2 vortices, 1 tracer)

vortices:

$$\frac{dz_1^*}{dt} = \frac{i}{2\pi}\frac{\Gamma_1}{z_1 - z_2}$$

$$\frac{dz_2^*}{dt} = \frac{i}{2\pi}\frac{\Gamma_2}{z_2 - z_1}$$

$\mathbf{x} = \{z_1, z_2, \xi\}$ — state variable

Γ — circulation strength

tracer:

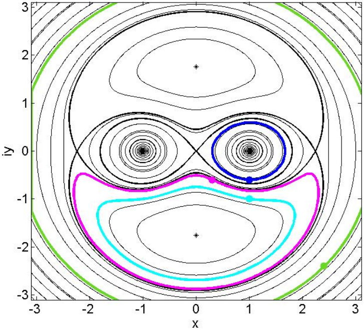
$$\frac{d\xi^*}{dt} = \frac{i}{2\pi}\frac{\Gamma_1}{\xi - z_1} + \frac{i}{2\pi}\frac{\Gamma_2}{\xi - z_2}$$

test bed:

- complex, nonlinear dynamics
- six-dimensional state space

Tracer paths

Stream function



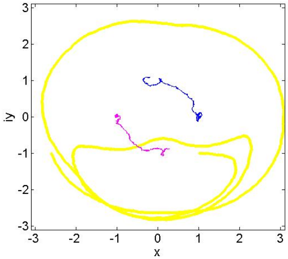
- transformed to lagrangian coordinates
- tracer paths for deterministic flow
- focus on four tracer IC

$(0.3 - 0.6i,\ 1 - 0.6i,\ 1 - i,\ 2.4 - 2.4i)$

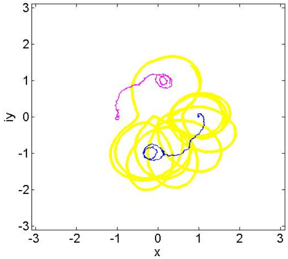
$dX_t = M(X_t, t)dt + G(X_t, t)dW_t$, W_t – standard Wiener process

- model noise $G(X_t, t)dW_t = \sigma d\eta$ with $\eta \sim N(0, dt\mathbf{I})$
  – unresolved small scale effects & uncertainty
- tracers can experience multiple “types" of flow

Noisy flow examples



($\xi(0) = 1 - i$)

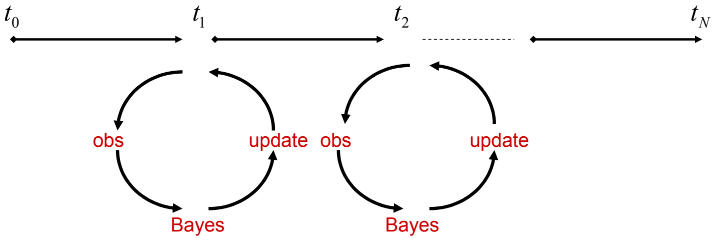


($\xi(0) = 1 - 0.6i$)

experiment:

- generate one “truth run”
- observe tracer locations periodically ($t_j = j\Delta t$),
- $Y_j = \xi^o(t_j) = \xi_j^t + \theta\eta_j$ with $\eta_j \sim N(0, \mathbf{I})$
- use DA to infer vortex locations

Bayesian view of sequential DA



$x =$ state

$Y =$ obs

$$p(x \mid Y) \propto p(Y \mid x)\, p(x)$$

Key question: how do we obtain the distributions on RHS?

We can rewrite Bayes formula conditioning on all previous observations

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

where $R(Y_j|x_j)$ is the likelihood of the j observation and where

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

- transition probability, $p_j(x_j|x_{j-1})$, is tricky
- PF approximates integral with Monte Carlo
- resulting prior is discrete approx of $\pi(x_j|Y_{0,j-1})$

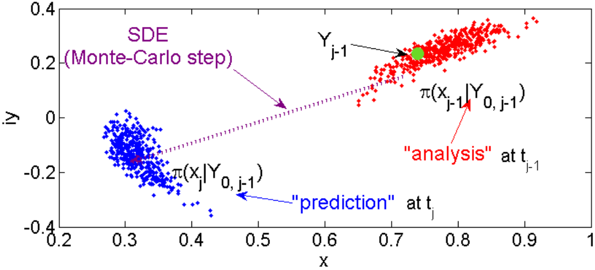
prediction step:

$$\pi(x_j | Y_{0,j-1}) = \{x_j, w_j^p(x_j) : w_j^p(x_j) = w_{j-1}(x_{j-1}) \text{ where } x_{j-1} \xrightarrow{\text{SDE}} x_j\}$$

discrete approx:

Particles are the support of the discrete approximations to these distributions

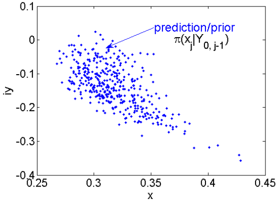
Each particle is associated with a weight, $w_j(x_j)$



Particle filters: update/analysis at $t = t_j$

Know (discrete approximation):

$\pi(x_j | Y_{0,j-1})$ (from last page)



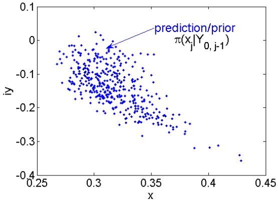
Particle filters: update/analysis at $t = t_j$

Know (discrete approximation):

$$\pi(x_j | Y_{0,j-1}) \text{ (from last page)}$$

Bayes:

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



Particle filters: update/analysis at $t = t_j$

Know (discrete approximation):

$$\pi(x_j | Y_{0,j-1}) \text{ (from last page)}$$

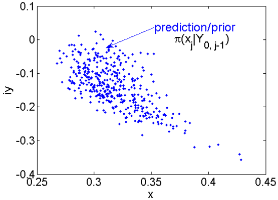
Bayes:

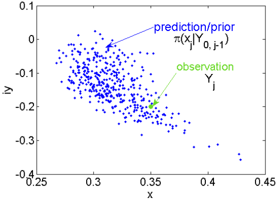
$$\pi(x_j | Y_{0,j}) \propto R(x_j, Y_j)\pi(x_j | Y_{0,j-1})$$

Likelihood:

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

(recall $x = \{\xi, z_1, z_2\}$, but $H(x) = \xi$)





Particle filters: update/analysis at $t = t_j$

Know (discrete approximation):

$$\pi(x_j | Y_{0,j-1}) \text{ (from last page)}$$

Bayes:

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

Likelihood:

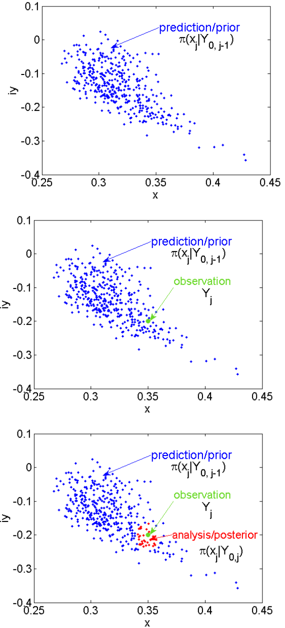
$$R(x, Y) = \exp\left[\frac{H(x) \cdot Y}{\theta^2} - \frac{|H(x)|^2}{2\theta^2}\right]$$

(recall $x = \{\xi, z_1, z_2\}$, but $H(x) = \xi$)

Update (discrete Bayes):

$$w_j(x_j) \propto R(x_j, Y_j) w_j^p(x_j)$$

$$\pi(x_j | Y_{0,j}) = \{x_j, w_j(x_j)\}$$



Particle filter algorithm

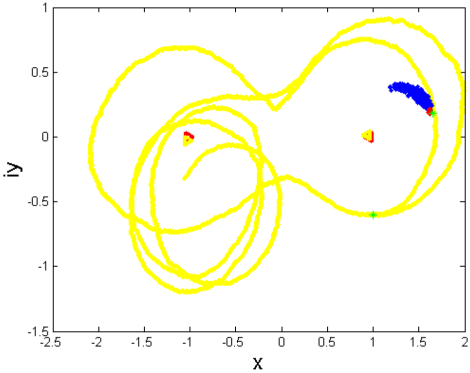
Sequential Monte-Carlo algorithm

1. Generate the "truth" – one numerical simulation of SDE
2. Generate N "particles", i.e., N copies of the initial state
3. Evolve the N-particle "cloud" to next observation instant
4. Observe the tracer location (obs = "truth" + "uncertainty")
5. Calculate $R(x_j, Y_j)$ and posterior distribution π_j (posterior cloud is a reweighted estimate of prior cloud)
6. Filter approximates hidden states (vortex locations)

$$z^a_{(1,2)}(t_j) = E_{\pi_j}[z_{(1,2)}(t_j)]$$

7. Posterior cloud is now best estimate of current state, repeat steps 3-7 until $t_j = t_{final}$

Movie



Movie

The good and bad of particle filters

Benefits

- Naturally handles nonlinearity
- Don't need to make Gaussian assumptions on prior or posterior distributions
  – no problem with bi-modal or skew distributions

Drawbacks

- degeneracy
  – a few particles hold all the weight $\rightarrow$ poor MC approx
- loss of support
  – particle cloud pulls away from observations
- poor performance in high dimensional problems

Strategy

- some form of *importance sampling* on the prior

–note, can perturb observations, see
(Houtekamer & Derome, 1995), (Burgers *et al* 1998)

Back to the prior

In the most basic sense, importance sampling is sampling from a distribution that has been biased/nudged/perturbed/altered somehow and accounting for it.

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

To do so here, we can alter

- $\pi(x_{j-1}|Y_{0,j-1})$, posterior from $j-1^{st}$ observation

or

- $p_j(x_j|x_{j-1})$, transition probability

Typically, the former is altered when applying PFs to DA for example...

PF divergence and resampling

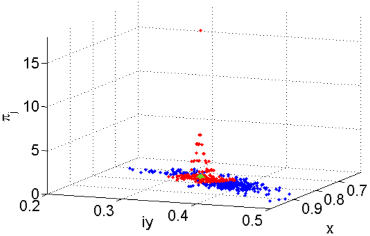
problem: degeneracy

- all the weight gets centered on a few particles
- well known and studied (Doucet *et al*)

solution: resampling

idea:

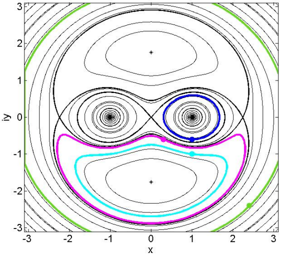
- pick subset of "best" particles $k = 1, \ldots, M$
- make m_k copies of each particle where $m_k \propto w_j(x_j^{(k)})$ where $\sum m_k = N$



reasonable:

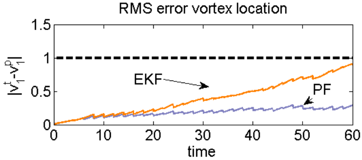
- doesn't add sampling error
- stochastic evolution to t_{j+1} "spreads out" cloud

Application of PF with resampling



Experiment:

- run 2,000 truth runs (500 $\times$ 4 IC)
- use particle filter (PF) & EKF to approx vortex trajectories
- Calculate RMS error between true vortex locations & assimilated approximations
- Failure if error exceeds threshold

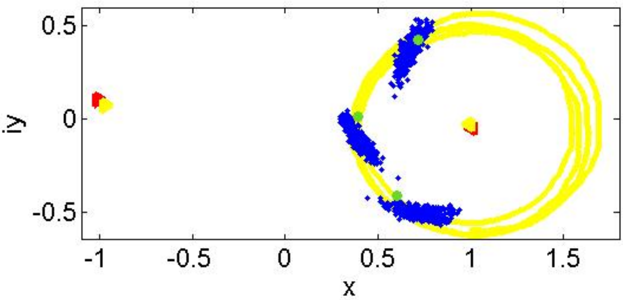


Failure rate:

	$0.3 - 0.6i$	$1 - 0.6i$	$1 - i$	$2.4 - 2.4i$
PF	7.8%	3.0 %	4.6%	12.0%
EKF	4.6%	78.2 %	0.6%	2.0%

PF does well in strongly nonlinear case, but we can do better problem: losing support

- prior cloud pulls away from observation



notice:

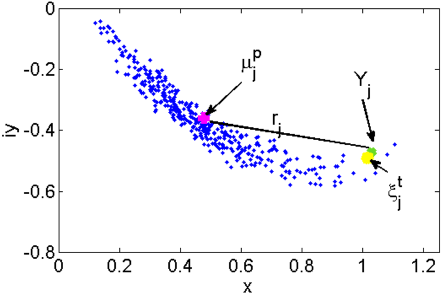
- cloud pulls away, but vortex approx OK

Modified particle filter: monitor cloud

At each observation, calculate *discrepancy factor*

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

where $r_j = |Y_j - \mu_j^p|$ and Σ_j covariance matrix



If δ_j beneath a small threshold ($\sim$ 0.01) employ backtracking

Modified particle filter: backtracking

idea:

- when $\delta_j <$ threshold, back up to observation instant $j - 2$

reinitialize:

- double number of particles that approx $\pi(x_{j-2}|Y_{0,j-2})$
- evolve forward to t_{j+1}, assimilating along the way
- sometimes different noise realization enough

idea: perturb states in $\pi(x_{j-2}|Y_{0,j-2})$

- perturb tracer coordinates (not hidden state)

two schemes:

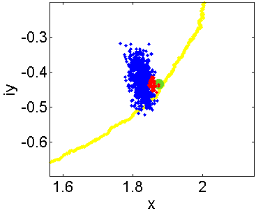
cloud expansion directed doubling

Modified particle filters: back up

At observation instant t_{j-2} ...

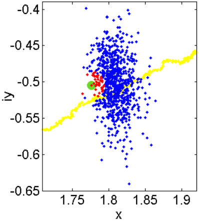
Cloud expansion

- Add i.i.d.normal RV to each tracer component of each particle's state ($2N$ particles)



Directed doubling

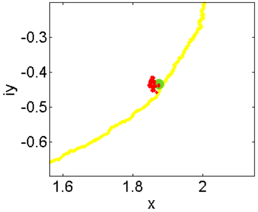
- pick M particles with highest weight
- bias particles along a line toward (& away from) Y_{j-2}



At observation instant t_{j-2} ...

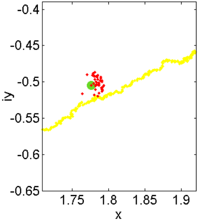
Cloud expansion

- Add i.i.d.normal RV to each tracer component of each particle's state ($2N$ particles)



Directed doubling

- pick M particles with highest weight
- bias particles along a line toward (& away from) Y_{j-2}

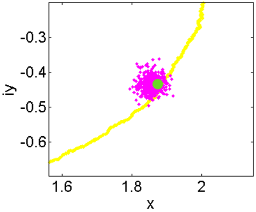


Modified particle filters: reinitialize

At observation instant t_{j-2} ...

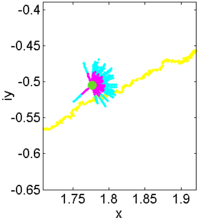
Cloud expansion

- Add i.i.d.normal RV to each tracer component of each particle's state ($2N$ particles)



Directed doubling

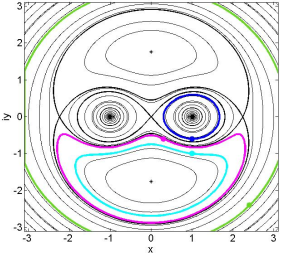
- pick M particles with highest weight
- bias particles along a line toward (& away from) Y_{j-2}



Backtracking particle filters: some results

Experiment:

- 2,000 truth runs (500×4 IC)
- $T_{final} = 60$, $\Delta t = 1$, $\delta = 0.02$, $\sigma = 0.02$, $N = 400$ particles

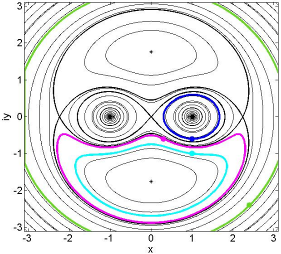


	$0.3 - 0.6i$	$1 - 0.6i$	$1 - i$	$2.4 - 2.4i$
standard PF	7.8%	3.0%	4.6%	12.0%
standard BPF w/doubling	6.4%	2.6%	2.6%	9.6%
cloud expanding BPF	0.4%	0.2%	0.2%	4.0%
directed doubling BPF	0.4%	0.0%	0.0%	4.8%
perturbed observation BPF	1.6%	1.6%	2.0%	7.0%
extended Kalman filter	4.6%	78.2%	0.6%	2.0%

Backtracking particle filters: some results

Experiment:

- 2,000 truth runs (500×4 IC)
- $T_{final} = 60$, $\Delta t = 1$, $\delta = 0.02$, $\sigma = 0.02$, $N = 400$ particles

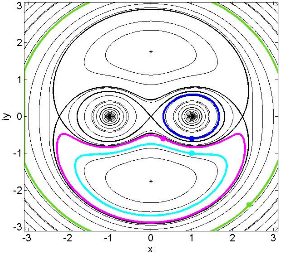


	$0.3 - 0.6i$	$1 - 0.6i$	$1 - i$	$2.4 - 2.4i$
standard PF	7.8%	3.0%	4.6%	12.0%
standard BPF w/doubling	6.4%	2.6%	2.6%	9.6%
cloud expanding BPF	0.4%	0.2%	0.2%	4.0%
directed doubling BPF	0.4%	0.0%	0.0%	4.8%
perturbed observation BPF	1.6%	1.6%	2.0%	7.0%
extended Kalman filter	4.6%	78.2%	0.6%	2.0%

Backtracking particle filters: some results

Experiment:

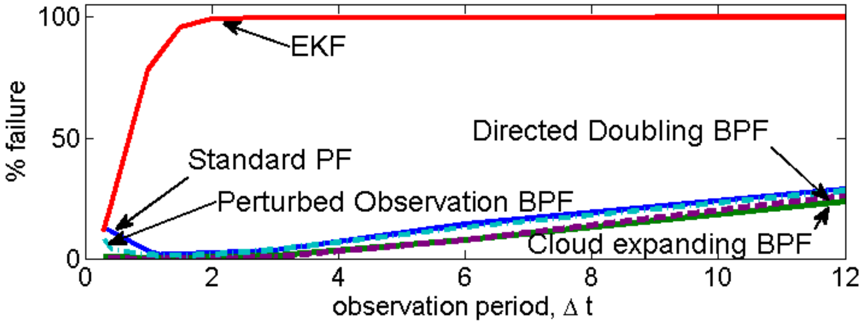
- 2,000 truth runs (500$\times$4 IC)
- $T_{final} = 60$, $\Delta t = 1$, $\delta = 0.02$, $\sigma = 0.02$, $N = 400$ particles



	$0.3 - 0.6i$	$1 - 0.6i$	$1 - i$	$2.4 - 2.4i$
standard PF	7.8%	3.0%	4.6%	12.0%
standard BPF w/doubling	6.4%	2.6%	2.6%	9.6%
cloud expanding BPF	0.4%	0.2%	0.2%	4.0%
directed doubling BPF	0.4%	0.0%	0.0%	4.8%
perturbed observation BPF	1.6%	1.6%	2.0%	7.0%
extended Kalman filter	4.6%	78.2%	0.6%	2.0%

Experiment:

- IC $\xi(0) = 1 - 0.6i$
- 500 truth runs for each Δt
- $0.5 \leq \Delta t \leq 12$
- $T_{final} = 60$, $\Delta t = 1$, $\delta = 0.02$, $\sigma = 0.02$, $N = 400$ particles



Another approach to combat PF divergence

Recently proposed by van Leeuwen (preprint, QJRMetSoc, 2010)

"Nudge" model evolution of particles toward next observation

- new to PF for DA in geosciences, but old idea
- effectively just importance sampling on $p_j(x_j|x_{j-1})$
- rewrite prior distribution

$$\pi(x_j|Y_{0,j-1}) = \int \frac{p_j(x_j|x_{j-1})}{p_j^*(x_j|x_{j-1})} p_j^*(x_j|x_{j-1}) \pi(x_{j-1}|Y_{0,j-1}) dx_{j-1}.$$

- again, approximate with Monte Carlo

$$X_j \sim p_j^*(x_j|x_{j-1}) \pi(x_{j-1}|Y_{0,j-1})$$

- adjust weights to correct for biasing with likelihood ratio

$$\frac{p_j(x_j|x_{j-1})}{p_j^*(x_j|x_{j-1})}$$

Choosing how to nudge/bias

Rewrite state evolution:
f deterministic evolution and β_j stochastic model error

$$X_j = f(X_{j-1}) + \beta_j \qquad \text{or} \qquad \beta_j = X_j - f(X_{j-1})$$

Now, $\beta_j \sim p(x_j|x_{j-1})$, and we can 'nudge' β_j (drawing from $p^*(x_j|x_{j-1})$) to move $H(X_j)$ closer to Y_j.

Van Leeuwen proposes

$$X_j = f(X_{j-1}) + \beta_j + K(Y_j - H(X_{j-1}))$$

but "we have enormous freedom here, we can choose 'any' term that forces the model towards the future observations."

Note – I take this as a word of caution.

"Nudging" for point vortex problem

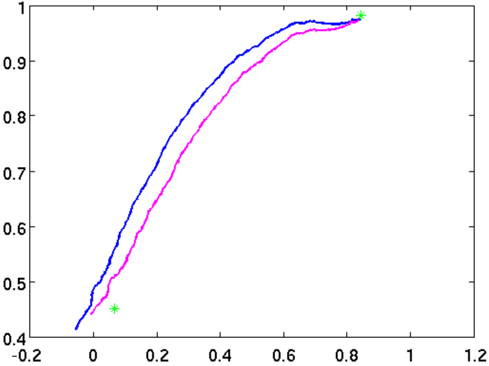
The scheme we used (for small dimensional problems, most "reasonable" schemes should work)

$$X_j = f(X_{j-1}) + \beta_j + K(Y_j - H(f(X_{j-1})))$$

- Find the line between next observation and a particle location at t_j if it moved from t_{j-1} to t_j deterministically, Δd
- Evolve particle forward in time biasing Weiner process by $\Delta d/(2N)$.
- Update likelihood ratio during evolution

$$\color{red}{w_j(x_j)} \propto \color{green}{R(x_j, Y_j)} \frac{p(x_j|x_{j-1})}{p^*(x_j|x_{j-1})} \color{blue}{w_j^p(x_j)}$$

Example sample paths



blue – unbiased
pink – biased

- This choice of biasing is most likely not optimal
- “optimal” choice would be the smallest available (in some norm)

Movie – PF diverging

Movie – PF with “nudging” model toward observation

Discussion

- PF with importance sampling addresses degeneracy and loss of support for Lagrangian DA
- Seems this strategy could be useful for high-dimensional problems
- If done well, IS can vastly reduce number of particles needed
- requires intelligent biasing

“If an unlikely event occurs, it is very likely to occur in the most likely way.” – anonymous

- optimization problem on $J[b(t)]$ conditioned on particles landing near observations, linearization OK
- possibly ideas from Jonathan’s talk