Assimilation of height observations and Lagrangian data on unknown paths

Elaine Spiller

Marquette University

February 11, 2013

Outline

- En-route LaDA
    - Motivation and Lagrangian instruments
    - Linearized shallow water
    - Preliminary results and looking forward

Ocean "views"

Argo float

glider

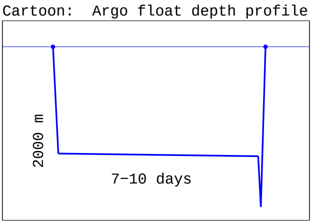
Lagrangian instruments

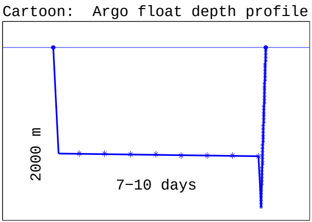
Argo float

glider

- Goal – collect below-surface measurements to better understand 3D dynamics and structures
- Lagrangian instruments collect data en route (temperature, pressure, salinity)
- Observations depend on unknown drifter paths
- What to do with that data?

Float depth profile

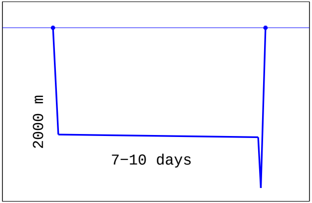




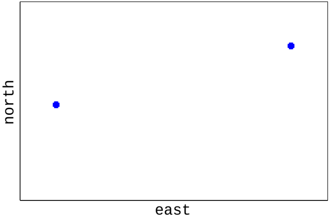
- 7-10 day float results in O(10)-O(100) km traveled
- high frequency data in dive/ascent just before surfacing in water column beneath “surfacing location”
- low frequency en-route measurements at depth, no latitude/longitude information
- en-route measurements averaged, not used in assimilation

Float depth and overview

Cartoon: Argo float depth profile

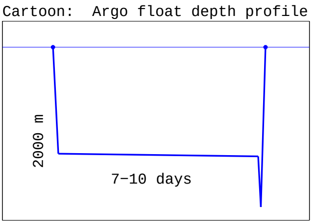


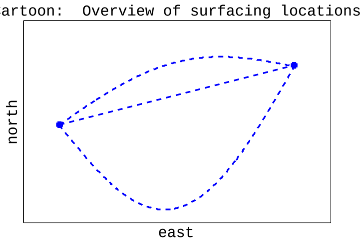
Cartoon: Overview of surfacing locations



- Lagrangian DA can help ascertain velocities w/o averaging

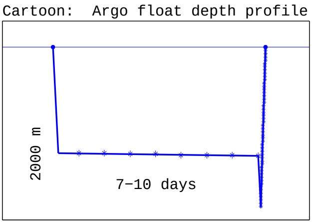
Float depth and overview

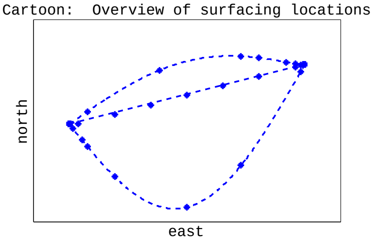




- Some possible Lagrangian paths

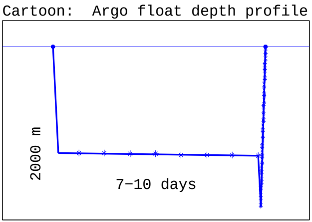
Float depth and overview

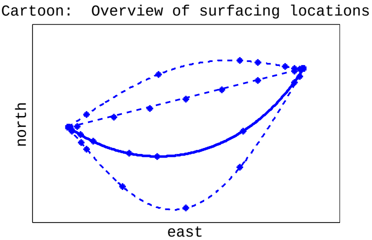




- need path & speed for subsurface observation locations

Float depth and overview





- Can en-route observations help Lagrangian DA?

Assimilated 3-D Lagrangian paths are (possibly) useful for

- aid in resolving Lagrangian structures

- assimilating data into high resolution models

- avoiding averaging via determining en-route data collection locales along paths which cross multiple grid cells

Inviscid linearized Shallow Water Equations, periodic BCs

Non-dimensional velocity fields

$$\frac{\partial u}{\partial t} = v - \frac{\partial h}{\partial x}$$

$$\frac{\partial v}{\partial t} = -u - \frac{\partial h}{\partial y}$$

$$\frac{\partial h}{\partial t} = -\frac{\partial u}{\partial x} - \frac{\partial v}{\partial y}$$

Lagrangian trajectories

$$\dot{x}(t) = u[x(t), y(t), t]$$

$$\dot{y}(t) = v[x(t), y(t), t]$$

Decomposition into Fourier Modes

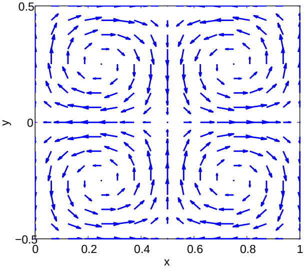
$$u(x, y, t) = -2\pi \sin(2\pi x)\cos(2\pi y)u_o + \cos(2\pi y)u_1(t)$$

$$v(x, y, t) = 2\pi \cos(2\pi x)\sin(2\pi y)u_o + \cos(2\pi y)v_1(t)$$

$$h(x, y, t) = \sin(2\pi x)\sin(2\pi y)u_o + \sin(2\pi y)h_1(t)$$

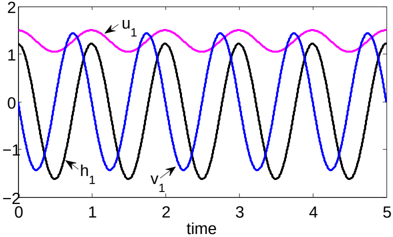
Cellular flow field

If $u_1 = v_1 = h_1 = 0$, flow field is constant & tracers stay w/in cells



$$\text{Otherwise, } \dot{u}_o = 0, \quad \dot{v}_1 = -u_1 - 2\pi h_1, \quad \dot{u}_1 = v_1, \ \ \& \quad \dot{h}_1 = 2\pi v_1$$

with initial conditions $[u_o(0), u_1(0), v_1(0), h_1(0)]$

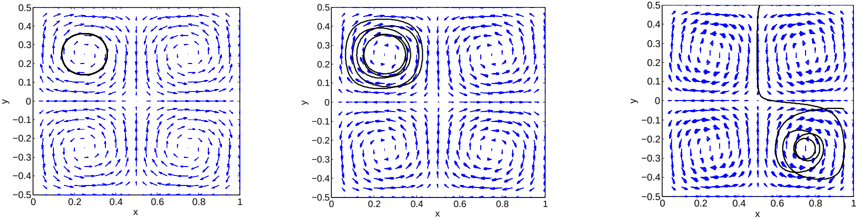


A few trajectories

Left: $u_1(0) = v_1(0) = h_1(0) = 0, x(0) = .2, \ y(0) = .3$

Middle: $u_1(0) = v_1(0) = h_1(0) = 0.5, x(0) = .2, \ y(0) = .3$

Right: $u_1(0) = 0.2, \ v_1(0) = 1.3, \ h_1(0) = 1.4, \ x(0) = .51, \ y(0) = .498$



Particle filter for standard LADA

Test problem:

- $u_1(0) = v_1(0) = h_1(0) = 0.5, \; x(0) = .2, \; y(0) = .3$
- broad priors on (u_1, v_1, h_1), tight on (x, y) at $t = 0$
- run to $t = T$ (1 period of coefficients)
- 5 noisy observations of drifter

Particle filter for standard LADA

Test problem:

- $u_1(0) = v_1(0) = h_1(0) = 0.5,\ x(0) = .2,\ y(0) = .3$
- broad priors on (u_1, v_1, h_1), tight on (x, y) at $t = 0$
- run to $t = T$ (1 period of coefficients)
- 5 noisy observations of drifter

Goal:

- learn about $u_1(0), v_1(0), h_1(0)$ from Lagrangian observations

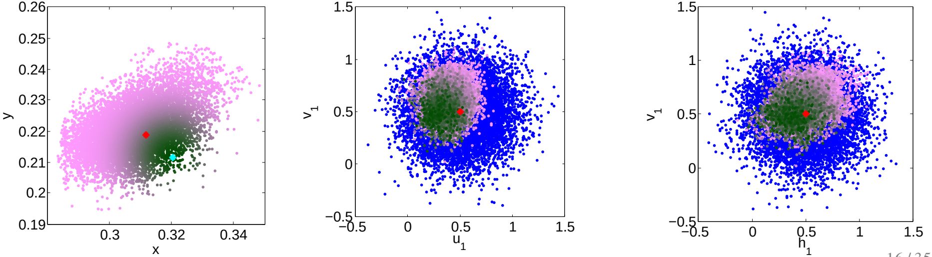
Particle filter for standard LADA

Test problem:

- $u_1(0) = v_1(0) = h_1(0) = 0.5,\ x(0) = .2,\ y(0) = .3$
- broad priors on (u_1, v_1, h_1), tight on (x, y) at $t = 0$
- run to $t = T$ (1 period of coefficients)
- 5 noisy observations of drifter

Goal:

- learn about $u_1(0), v_1(0), h_1(0)$ from Lagrangian observations



En route Lagrangian data – a test problem

- Idea treat height, $h(x, y, u_1, v_1, h_1)$, as proxy for temperature – typical quantity measured en route

En route Lagrangian data – a test problem

- Idea treat height, $h(x, y, u_1, v_1, h_1)$, as proxy for temperature – typical quantity measured en route
- Sample height, $\hat{h}(t) = h(x^d(t), y^d(t), t) + \textit{noise}$ between "surfacings", e.g. traditional observation instants t_j

En route Lagrangian data – a test problem

- Idea treat height, $h(x, y, u_1, v_1, h_1)$, as proxy for temperature – typical quantity measured en route
- Sample height, $\hat{h}(t) = h(x^d(t), y^d(t), t) + \textit{noise}$ between "surfacings", e.g. traditional observation instants t_j
- Changes the observation space, so now ($z = \{x^d, y^d, u_1, v_1, h_1\}$ whole state)

$$H(z) = \begin{cases} (x^d(t), y^d(t)) & \text{for } t = jT_{obs} \\ \hat{h}(t) & \text{for } t = t_k, \;\; (j-1)T_{obs} < t_k < jT_{obs} \end{cases}$$

En route Lagrangian data – a test problem

- Idea treat height, $h(x, y, u_1, v_1, h_1)$, as proxy for temperature – typical quantity measured en route
- Sample height, $\hat{h}(t) = h(x^d(t), y^d(t), t) + \textit{noise}$ between "surfacings", e.g. traditional observation instants t_j
- Changes the observation space, so now ($z = \{x^d, y^d, u_1, v_1, h_1\}$ whole state)

$$H(z) = \begin{cases} (x^d(t), y^d(t)) & \text{for } t = jT_{obs} \\ \hat{h}(t) & \text{for } t = t_k, \ \ (j-1)T_{obs} < t_k < jT_{obs} \end{cases}$$

- Update Likelihood at "surfacing" time t_j with data $\{x_j^o, y_j^o, \hat{h}^o_{k=1\ldots N_h}\}$

En route Lagrangian data – a test problem

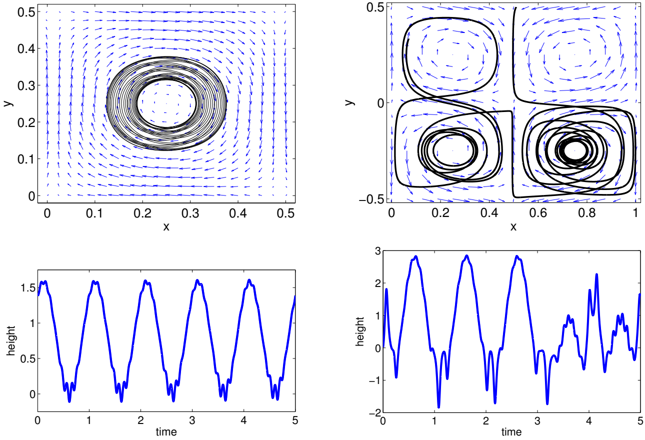
- Idea treat height, $h(x, y, u_1, v_1, h_1)$, as proxy for temperature – typical quantity measured en route
- Sample height, $\hat{h}(t) = h(x^d(t), y^d(t), t) + \textit{noise}$ between "surfacings", e.g. traditional observation instants t_j
- Changes the observation space, so now ($z = \{x^d, y^d, u_1, v_1, h_1\}$ whole state)

$$H(z) = \begin{cases} (x^d(t), y^d(t)) & \text{for } t = jT_{obs} \\ \hat{h}(t) & \text{for } t = t_k, \;\; (j-1)T_{obs} < t_k < jT_{obs} \end{cases}$$

- Update Likelihood at "surfacing" time t_j with data $\{x_j^o, y_j^o, \hat{h}^o_{k=1\ldots N_h}\}$

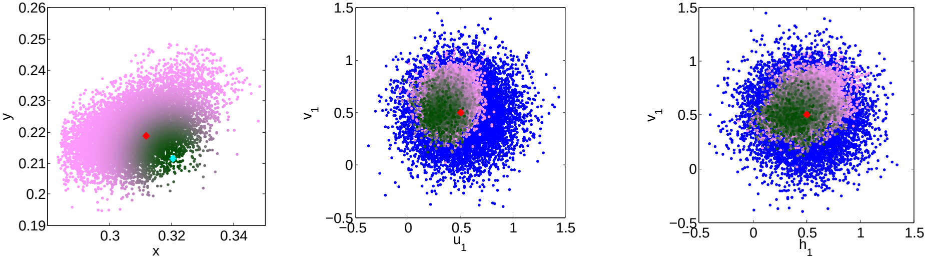
$$-\log(g) = \frac{(x^d - x^o)^2 + (y^d - y^o)^2}{2\sigma_d^2} + \frac{1}{N_h}\sum_{N_h}(h(z_k) - \hat{h}_k^o)^2/2\sigma_h^2$$

Center & Saddle — paths & height time series



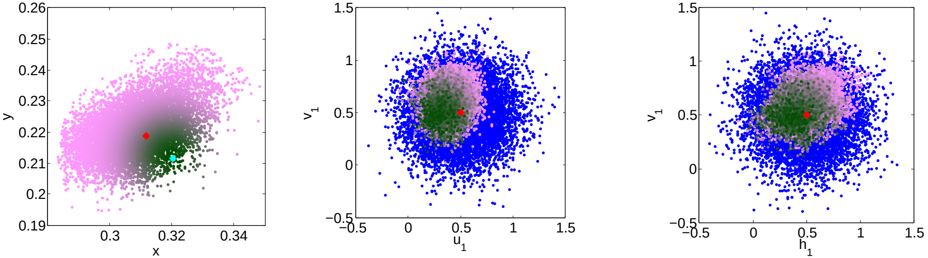
Particle filter w/en route observations

<u>“traditional” LADA</u>:

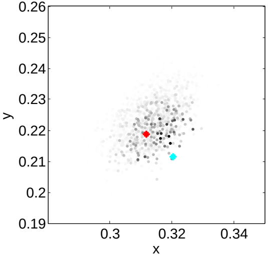


Particle filter w/en route observations

"traditional" LADA:

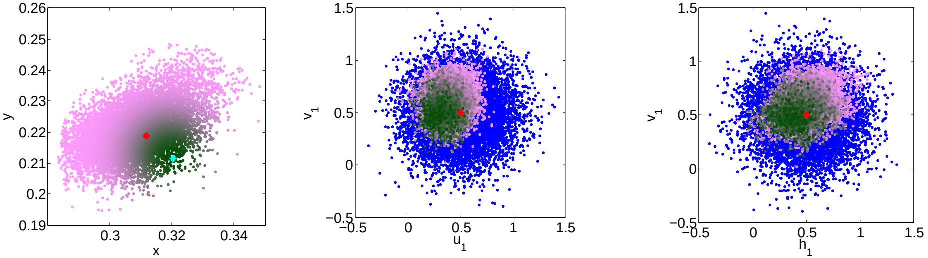


en route LADA:

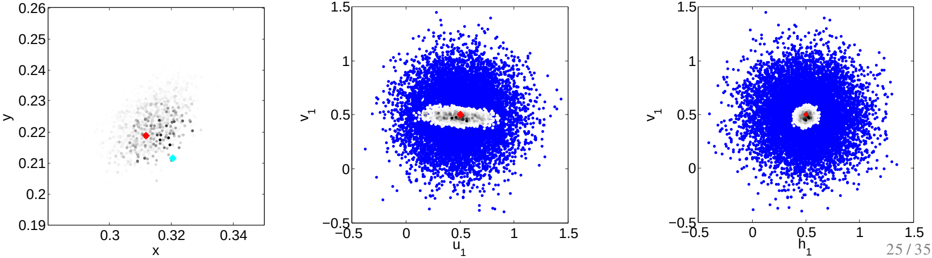


Particle filter w/en route observations

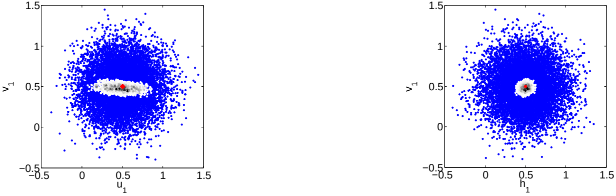
<u>"traditional" LADA</u>:



<u>en route LADA</u>:



Characterizing improvement



Characterizing improvement:
compare covariance matrices of prior and posterior distribution

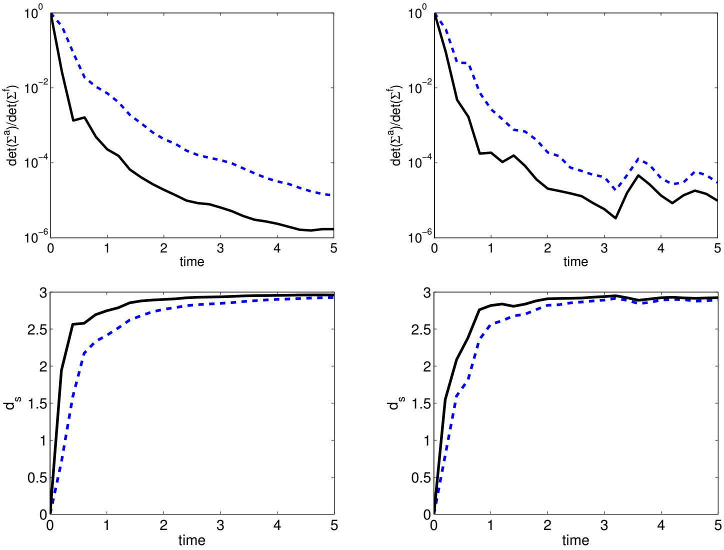
$$d_s(t) = tr[\mathbf{I} - \Sigma_F^a(t)(\Sigma_F^f)^{-1}] \qquad \text{(Zupanski, 2007)}$$

$$r(t) = det(\Sigma_F^a(t))/det(\Sigma_F^f)$$

Improvements w/assimilating en-route data

left column: center

right column: saddle

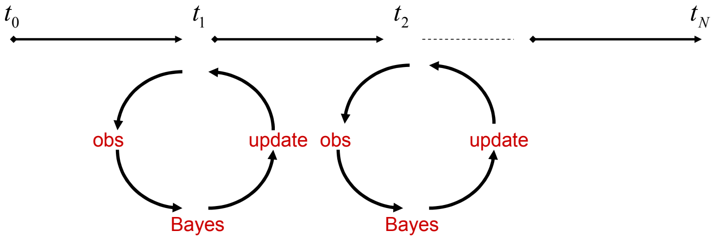


(S, Apte, Jones, submitted 2012)

Looking forward

- Consider two-layer problem w/observable-at-depth, spatially dependent variable
  - collect en-route observations on bottom layer
  - traditional Lagrangian observations, less frequent
- Improvements from en-route assimilation?
- Can we estimate Lagrangian paths at depth?
- How dependent is this on coupling strength?

Bayesian view of 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?

Markov transitions via model

We're interested in the probability of a state X_j as it evolves over time. Recall, for independent random variables, we have

$$p(x_{0:n}) = p(x_0) \prod_{j=1}^{n} p(x_j | x_{1:j-1})$$

For our case, we'll have some distribution of initial conditions $\mu(x_0)$ (background) and a model to move our state forward in time,

$$X_j | (X_{j-1} = x_{j-1}) \sim m(x_j | x_{j-1})$$

where $m(x_j | x_{j-1})$ is the *transition probability* or the probability that our model would take use from state x_{j-1} to state x_j.

Combining the ideas above gives us

$$p(x_{0:n}) = \mu(x_0) \prod_{j=1}^{n} m(x_j | x_{j-1})$$

Observations and likelihood

Recall, our observations will be related to the state variable by some observation function $y = H(x)$. We can think of observations as random variables distributed as

$$Y_j|(X_j = x_j) \sim g(y|x_j).$$

Or, $Y_j = H(X_j) +$ "*noise*".

$g(y|x)$ is the *likelihood* — how likely was an observation given the possible states?

With a whole set of observations $\{Y_j\}$ we can write down the likelihood for the time-series of observations

$$p(y_{1:j}|x_{1:j}) = \prod_{j=1}^{n} g(y_k|x_k)$$

Inference: goal for data assimilation

Given a background distribution of initial conditions, $\mu(x_0)$, and observations, $Y_{1:n}$, we want to infer the distribution of physical states $X_{0:n}$.

- Prior

$$p(x_{0:n}) = \mu(x_o) \prod_{j=1}^{n} m(x_j | x_{j-1})$$

- Likelihood

$$p(y_{1:n} | x_{1:n}) = \prod_{j=1}^{n} g(y = H(x_j) | x_j)$$

- Posterior, obtained by Bayes' rule

$$p(x_{1:n} | y_{1:n}) = \frac{p(y_{1:n} | x_{1:n}) p(x_{0:n})}{p(y_{1:n})}$$

recall, $p(y_{1:n}) = \int p(y_{1:n} | x_{1:n}) p(x_{0:n}) dx_{1:n}$

A Monte Carlo simulation or really sampling $p(x_{1:n}|y_{1:n})$

- takes a discrete set of samples from $X_0 \sim p(x_0)$
- moves them forward accord to the model, e.g. samples $X_{0:j} \sim p(x_j|x_{0:j-1})$
- evaluates likelihood between samples and observations

Note, after a few (say $k = 2$ or 3 observations) you will have samples from $X_{0:k} \sim p(x_{0:k}|y_{1:k})$ but they will not be useful.

Sequential Monte Carlo with Importance Sampling (SIS)

Idea — normalize at every step, treat that posterior distribution as an *importance* prior distribution for the next step.

1. Start with $X_o \sim p(x_o)$, each particle $X_o^{(k)}$ has weight $w_1^{(k)} = 1/N$

Sequential Monte Carlo with Importance Sampling (SIS)

Idea — normalize at every step, treat that posterior distribution as an *importance* prior distribution for the next step.

1. Start with $X_o \sim p(x_o)$, each particle $X_o^{(k)}$ has weight $w_1^{(k)} = 1/N$
2. Transition each $X_0^{(k)}$ forward, this gives sample $X_1^{(k)} \sim p(x_1|x_0) = m(x_1|x_0)$

Sequential Monte Carlo with Importance Sampling (SIS)

Idea — normalize at every step, treat that posterior distribution as an *importance* prior distribution for the next step.

1. Start with $X_o \sim p(x_o)$, each particle $X_o^{(k)}$ has weight $w_1^{(k)} = 1/N$
2. Transition each $X_0^{(k)}$ forward, this gives sample $X_1^{(k)} \sim p(x_1|x_0) = m(x_1|x_0)$
3. Evaluate the likelihood function of each sample ("particle") $X_1^{(k)}$ against Y_1, $g(Y_1|X_1^{(k)})$

Sequential Monte Carlo with Importance Sampling (SIS)

Idea — normalize at every step, treat that posterior distribution as an *importance* prior distribution for the next step.

1. Start with $X_o \sim p(x_o)$, each particle $X_o^{(k)}$ has weight $w_1^{(k)} = 1/N$
2. Transition each $X_0^{(k)}$ forward, this gives sample $X_1^{(k)} \sim p(x_1|x_0) = m(x_1|x_0)$
3. Evaluate the likelihood function of each sample ("particle") $X_1^{(k)}$ against Y_1, $g(Y_1|X_1^{(k)})$
4. *Weight* each particle by

$$w_1^{(k)} = \frac{g(Y_1|X_1^{(k)})w_0^{(k)}}{\sum_{k=1}^{N} g(Y_1|X_1^{(k)})w_0^{(k)}}$$

Sequential Monte Carlo with Importance Sampling (SIS)

Idea — normalize at every step, treat that posterior distribution as an *importance* prior distribution for the next step.

1. Start with $X_o \sim p(x_o)$, each particle $X_o^{(k)}$ has weight $w_1^{(k)} = 1/N$
2. Transition each $X_0^{(k)}$ forward, this gives sample $X_1^{(k)} \sim p(x_1|x_0) = m(x_1|x_0)$
3. Evaluate the likelihood function of each sample ("particle") $X_1^{(k)}$ against Y_1, $g(Y_1|X_1^{(k)})$
4. *Weight* each particle by

$$w_1^{(k)} = \frac{g(Y_1|X_1^{(k)})w_0^{(k)}}{\sum_{k=1}^{N} g(Y_1|X_1^{(k)})w_0^{(k)}}$$

Repeat process transition from X_{j-1} to X_j instead of 0 to 1.
$\pi(x_j|Y_{1:j}) = \{x_j = X_j^{(k)}, w^{(k)}\}$

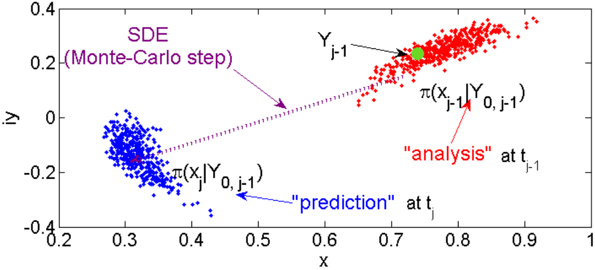
prediction step:

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

discrete approx:

Particles are the support of the discrete approximations to these distributions

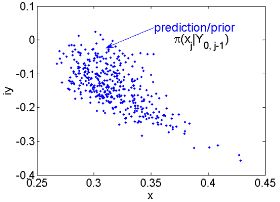
Each particle is associated with a weight, $w_j(x_j)$



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

Know (discrete approximation):

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



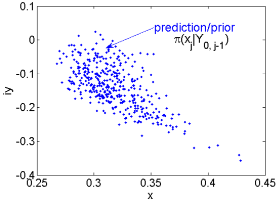
Particle filters: update/analysis at $t = t_j$

Know (discrete approximation):

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

Bayes:

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



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

Know (discrete approximation):

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

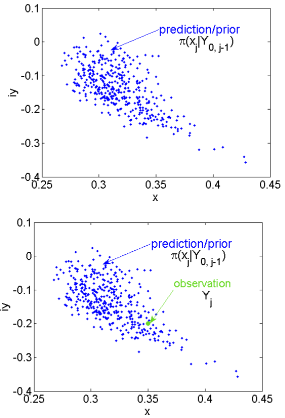
Bayes:

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

Likelihood:

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

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



Know (discrete approximation):

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

Bayes:

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

Likelihood:

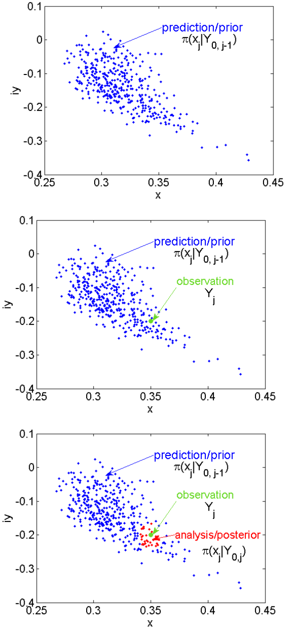
$$g(Y|x) = \exp\left[\frac{H(x) \cdot Y}{\theta^2} - \frac{|H(x)|^2}{2\theta^2}\right]$$

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

Update (discrete Bayes):

$$w_j(x_j) \propto g(Y_j | x_j) w_j^p(x_j)$$

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



Problem with SIS and Solution: Resampling

With a large number of samples, SIS works pretty well on moderate (small) dimensional deterministic (perfect model) problems.

Problem:

- A significant problem, though, is that most (or all) of the weight can be taken over by *one particle*

Solution:

- Resampling, e.g., bootstrapping

Sequential

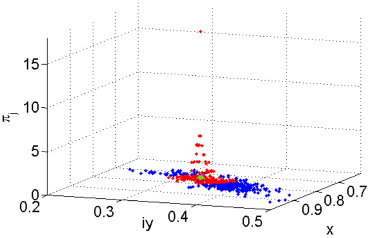
Strategy:

- Monitor weights, if problematic
- *Resample* or "bootstrap" by treating $\tilde{\pi}_j(x_{0:k}|Y_{1:k})$ as an *importance* empirical distribution
- Set all weights to $w_j^{(k)} = 1/N$
- Transition $j+1$ step, repeating resampling as necessary

The strategy is referred to an *SIR (sequential importance resampling) filter* and also goes by the names *particle filter*, *bootstrap filter*, and *sequential Monte Carlo*.

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:

- stochastic evolution to t_{j+1} "spreads out" cloud
- add "jitter" to each particle for deterministic evolution