Particle filtering for high-dimensional nonlinear systems: An introduction and new ideas

Jon Poterjoy

Department of Meteorology
The Pennsylvania State University

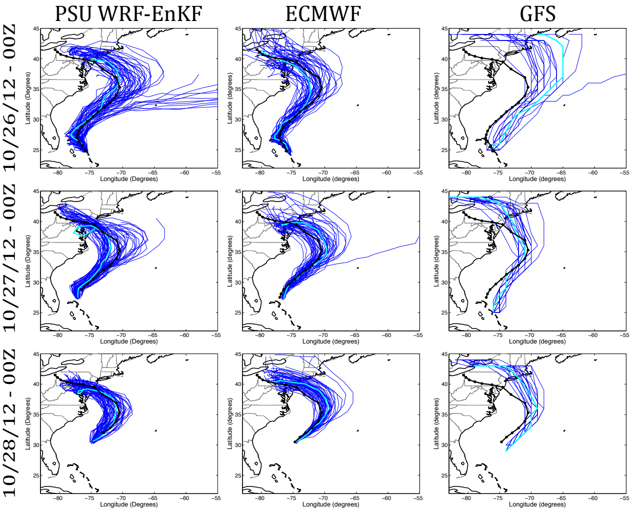
Thursday 12th December, 2013

Introduction

Our current state-of-the-art data assimilation systems (e.g., 4DVar, EnKF, and hybrid methods) operate under the assumptions of linear model dynamics and Gaussian errors.

Introduction

These assumptions have taken us a long way!



Hurricane Sandy
(2012)

Introduction

So, why bother trying to solve this problem?

- **Model resolution**: Gaussian assumption may be more severe at smaller scales.
- **Adoption of ensemble data assimilation**: useful information is ignored when ensembles are assumed to follow a Guassian error distribution.

Will our current data assimilation methods still be the best option as model resolution and ensemble size increase?

Bayesian filtering

Let $\mathbf{x}_t$ be a vector of state variables, and $\mathbf{y}_t$ be a vector of observations that are valid at time t.

$\mathbf{x}_t$ and $\mathbf{y}_t$ are given by the (possibly) nonlinear equations:

$$\begin{aligned} \mathbf{x}_t &= M(\mathbf{x}_{t-1}) + \eta_t, \\ \mathbf{y}_t &= H(\mathbf{x}_t) + \epsilon_t, \end{aligned}$$

where η_t and ϵ_t are stochastic terms that represent model and observation errors.

Let y_t denote all observations up to time t.

Bayesian filtering

The probability of $\mathbf{x}_t$, given all information up to this time is given by Bayes' theorem:

$$p(\mathbf{x}_t | y_t) \quad = \quad \frac{p(\mathbf{y}_t | \mathbf{x}_t) p(\mathbf{x}_t | y_{t-1})}{\int p(\mathbf{y}_t | \mathbf{x}_t) p(\mathbf{x}_t | y_{t-1}) d\mathbf{x}_t}.$$

Posterior: $p(\mathbf{x}_t | y_t)$

Likelihood: $p(\mathbf{y}_t | \mathbf{x}_t)$

Prior: $p(\mathbf{x}_t | y_{t-1})$

Evidence/support: $\int p(\mathbf{y}_t | \mathbf{x}_t) p(\mathbf{x}_t | y_{t-1})$

Particle filtering

Use an ensemble to construct an emperical estimate of the posterior probability distribution:

$$p(\mathbf{x}_t|y_t) \approx \sum_{n=1}^{N} w_t^n \delta(\mathbf{x}_t - \mathbf{x}_t^n).$$

Use the particle representation of $p(\mathbf{x}_t|y_t)$ to approximate expectations of functions f of the model state:

$$\begin{aligned} \overline{f(\mathbf{x}_t)} &= \int f(\mathbf{x}_t) p(\mathbf{x}_t|y_t) d\mathbf{x}_t, \\ &\approx \sum_{n=1}^{N} w_t^n f(\mathbf{x}_t^n). \end{aligned}$$

Examples of $\overline{f(\mathbf{x}_t)}$ are the mean and covariance of $\mathbf{x}_t$.

Particle filtering

For the simplest particle filter, the weights are given by

$$w_t^n \propto w_{t-1}^n p(\mathbf{y}_t | \mathbf{x}_t^n)$$

Particle filtering

When the particles begin to move away from observations, a resampling step is needed to produce a more informative ensemble.

The simplest example is the **bootstrap filter**: sample N new particles from the posterior error distribution with replacement, and assign each new particle a weight of $\frac{1}{N}$. The expression for w_t^n at the next cycle simplifies further:

$$w_t^n = \frac{p(\mathbf{y}_t|\mathbf{x}_t^n)}{\sum_{n=1}^{N}(\mathbf{y}_t|\mathbf{x}_t^n)}.$$

Particle filtering: example

For Gaussian observation errors, the n^{th} weight is given by

$$w_t^n \quad = \quad A \frac{exp\left\{-\frac{1}{2}[\mathbf{y}_t - H(\mathbf{x}_t^n)]^T \mathbf{R}^{-1}[\mathbf{y}_t - H(\mathbf{x}_t^n)]\right\}}{\sum_{k=1}^N exp\left\{-\frac{1}{2}[\mathbf{y}_t - H(\mathbf{x}_t^k)]^T \mathbf{R}^{-1}[\mathbf{y}_t - H(\mathbf{x}_t^k)]\right\}},$$

and the posterior mean can be approximated with

$$\overline{\mathbf{x}_t} \quad \approx \quad \sum_{n=1}^{N} w_t^n \mathbf{x}_t^n.$$

Problems with particle filtering

- **Filter degeneracy**: a finite ensemble will eventually lose track of the signal, in which case, the weights become concentrated on a small number of particles. This problem occurs faster when the dimensions of $\mathbf{x}$ and $\mathbf{y}$ are large (Snyder et al. 2008, MWR).
- **Resampling**: the process of resampling from the posterior can be problematic, since duplicate particles will be produced. This is a larger problem for deterministic models; i.e., no stochastic terms.

One solution

For high-dimensional systems, these weights are determined by applying a method called **sequential importance sampling**.

Particles are sampled from a subspace of the model space using a **proposal distribution** that is typically conditioned on observations between cycles (e.g., Leeuwen 2010, QJRMS).

My solution

Expand w_n^t from a scalar to a vector with the same dimension as $\mathbf{x}$, so that

$$\overline{\mathbf{x}_t} \approx \sum_{n=1}^{N} w_t^n \circ \mathbf{x}_t^n.$$

This is the same as estimating a weight for every state variable in each single particle.

Local likelihood particle filter

One way of obtaining this result is to evaluate the likelihood locally for each variable, so that observations at large distances from a grid point will not effect the local weight.

This requires a likelihood function that depends on the physical distance between the model grid point and observation locations.

Local likelihood particle filter (LLPF)

A simple example is to use a function that decays exponentially away from the location of an observation, so that the likelihood of the the i^{th} variable given the j^{th} observation is written

$$p(y_{t,j}|x^n_{t,i}) \quad = \quad [p(y_{t,j}|\mathbf{x}^n_t) - \frac{1}{N_e N_y}] exp(-\frac{d_{i,j}}{R}) + \frac{1}{N_e N_y},$$

where $d_{i,j}$ is the physical distance between the observation and model grid point, and R is a tunable localization radius.

Local likelihood particle filter (LLPF)

The Lorenze-96 model is used to test this idea and compare with EnKF:

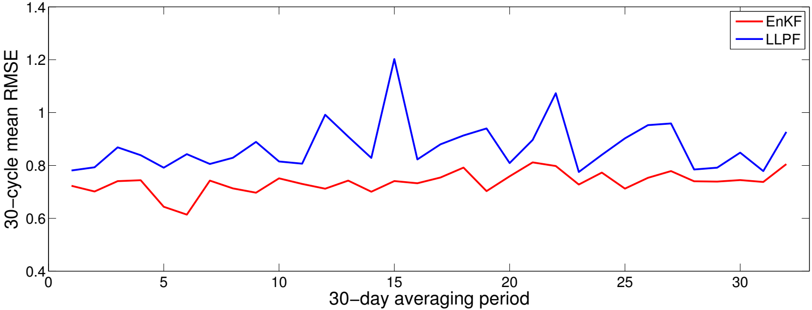
$$\frac{dx_{t+1,i}}{dt} = (x_{t,i+1} - x_{t,i-1})x_{t,i-1} - x_{t,i} + 10.$$

Experiment details:

- 100 variables
- Observations are taken from a "truth" run at every other grid point, with added Gaussian noise ($\sigma = 1$).
- These observations are assimilated every 12 h ($dt = 0.1$) for 1000 cycles.
- Relaxation coefficient and localization radius for EnKF are tuned for each ensemble size.

Local likelihood particle filter (LLPF)

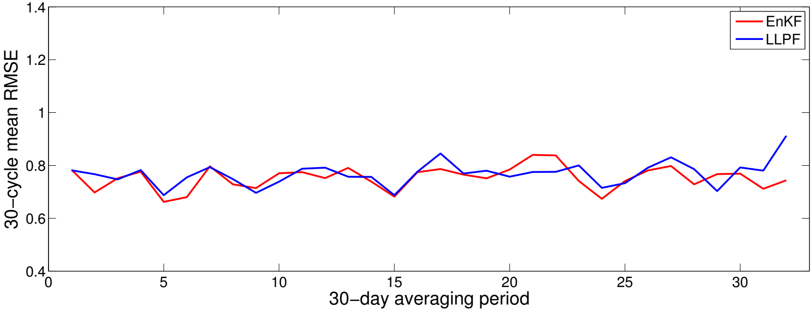
$N_e = 50$



RMSEs are averaged every 30 cycles.

Local likelihood particle filter (LLPF)

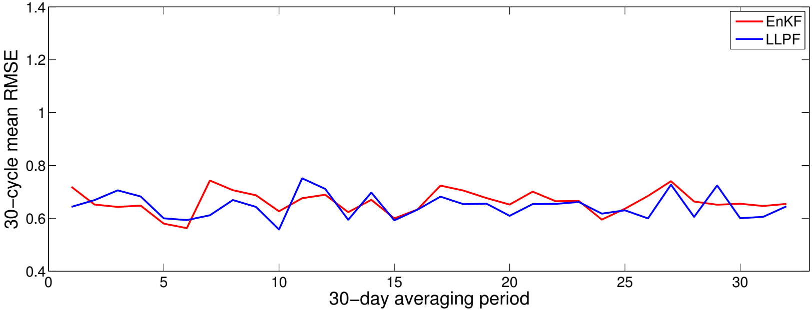
$$N_e = 100$$



RMSEs are averaged every 30 cycles.

Local likelihood particle filter (LLPF)

$$N_e = 200$$



RMSEs are averaged every 30 cycles.

Conclusions

- A new approach to particle filtering, called LLPF, has been introduced for systems that contain a large spatial dimension.
- This method makes use of a distance-dependent likelihood function to prevent filter degeneracy.
- The LLPF is shown to be stable without the use of an optimal proposal density.
- Much more work needs to be done; e.g., resampling, inflation, better choices for likelihood function, etc.