Some extensions to the basic Particle filter

Adaptive resampling, resample move, auxiliary particle filter

Introduction

- Particle filtering works very well to approximate

$$\pi_n(x_n) = p_\theta(\, x_n |\, y_{0:n})$$

- Performance can vary due to model specifics:
  - mixing of dynamics, level of observation noise, dimensionality
- There can still be issues to resolve related to path or weight degeneracy
  - so far main tool has been improving the proposal $q_\theta(x_n|y_n, x_{n-1})$

Recipes to improve performance

- There are more elaborate particle filtering algorithms
  - adaptive resampling
  - the resample move PF
  - the auxiliary particle filter
  - **can combine all the above together**
- These work better than vanilla PF
  - in terms of variance of estimators, ESS, accuracy etc.
  - ... but they do not completely address path degeracy, often just postpone it for a while.

Adaptive resampling

- While resampling is a key component to have a good approximation it tends to leave early states being represented by few particles.
- adaptive resampling
  - Key idea: use resampling only when you need to
  - Resample only when $ESS_n \leq \alpha N$
    - e.g. $\alpha = 1/2$.
  - When you dont resample continue with SIS

SIR filter with adaptive resampling

At time $n \geq 1$

- Sample $X_n^i \sim q_\theta(x_n | y_n, X_{n-1}^i)$ and set $X_{0:n}^i \leftarrow (X_{0:n-1}^i, X_n^i)$.
- Compute the weights $w_n\left(X_{n-1:n}^i\right)$ and set $W_n^i \propto W_{n-1}^i w_n\left(X_{n-1:n}^i\right)$, $\sum_{i=1}^N W_n^i = 1$.
- IF $ESS_n \leq \alpha N$
  - resample $\left\{W_n^i, X_{0:n}^i\right\}$ to obtain N new equally-weighted particles $\left\{\frac{1}{N}, \overline{X}_{0:n}^i\right\}$.
  - set $X_{0:n}^i \leftarrow \overline{X}_{0:n}^i$, $W_n^i \leftarrow \frac{1}{N}$

The resample move particle filter

- (Berzuini & Gilkks 2001 JRSSB) Fight path degeneracy by re-inserting lost diversity in the particles using appropriate MCMC moves on the path space

At time $n \geq 1$

- Sample $X_n^i \sim q_\theta(x_n | y_n, \tilde{X}_{n-1}^i)$ and set $X_{0:n}^i \leftarrow \left(\tilde{X}_{0:n-1}^i, X_n^i\right)$.
- Compute the weights $w_n\left(X_{n-1:n}^i\right)$ and set $W_n^i \propto w_n\left(X_{n-1:n}^i\right)$, $\sum_{i=1}^N W_n^i = 1$.
- Resample $\left\{W_n^i, X_{0:n}^i\right\}$ to obtain N new equally-weighted particles $\left\{\frac{1}{N}, \overline{X}_{0:n}^i\right\}$.
- Move particles by independently (for each i) sampling

$$\tilde{X}_{0:n}^i \sim K_{MCMC}(\cdot | \overline{X}_{0:n}^i)$$

The resample move particle filter

- target density for MCMC move is

$$p_\theta\left(x_{0:n} \mid y_{0:n}\right) \propto \eta_\theta\left(x_0\right) \prod_{k=1}^{n} f_\theta\left(x_k \mid x_{k-1}\right) \prod_{k=0}^{n} g_\theta\left(y_k \mid x_k\right)$$

- MCMC proposal in this context
  - just provides a jitter or shake in the particle population
  - does not need to move the whole trajectory, moving only $X_{n-L+1:n}$ can still lead to correct algorithm
  - we are not relying on ergodic properties of MCMC, just want to preserve statistical properties of sample

The resample move particle filter

Random walk algorithm for $\tilde{X}^i_{0:n} \sim K_{MCMC}(\cdot|\overline{X}^i_{0:n})$

- Set $\Upsilon_{0:n} = \overline{X}^i_{0:n}$
- For $m = 1, \ldots, M$
  - Sample $\mathfrak{U} \sim N(0, S)$, with S of appropriate dimension
  - Propose $Z_{n-L+1:n} = \Upsilon_{n-L+1:n} + c\mathfrak{U}$
  - Compute acceptance ratio

$$\alpha = 1 \wedge \frac{\prod_{k=n-L+1}^{n} f_\theta\left(Z_k | Z_{k-1}\right) g_\theta\left(y_k | Z_k\right)}{\prod_{k=n-L+1}^{n} f_\theta\left(\Upsilon_k | \Upsilon_{k-1}\right) g_\theta\left(y_k | \Upsilon_k\right)}$$

  - with probab. α:
    - accept $\Upsilon_{0:n} \leftarrow (\Upsilon_{0:n-L}, Z_{n-L+1:n})$
    - otherwise reject proposal and $\Upsilon_{0:n}$ remains the same

The resample move particle filter

- M can be quite small 1-5
- Tuning
  - Can use particles to design S, e.g. look at the empirical covariance of the particles after resampling
  - c can be tuned for average acceptance ratio around $0.2 - 0.4$
- Other MCMC moves are possible,
  - Gibbs, Hybrid Monte Carlo, ...
- Method will increase diversity a bit, but notice that
  - it does not affect the weights
  - it might be more effective to use **likelihood informed proposals and weights**
- The last point is related to the auxiliary particle filter by (Pitt & Sheppard 99 JASA)

The auxiliary particle filter

- Resample Move and adaptive resampling are meant to improve path degeneracy
- What if weight degeneracy due to IS is still present?
- Consider the Bayesian recursion:

$$p_\theta\left(x_{0:n}\middle|\, y_{0:n}\right) = \frac{1}{Z_n} p_\theta\left(x_{0:n-1}\middle|\, y_{0:n-1}\right) f_\theta\left(x_n\middle|\, x_{n-1}\right) g_\theta\left(y_n\middle|\, x_n\right)$$

  with $Z_n = p_\theta\left(y_n\middle|\, y_{0:n-1}\right)$.

  - Bootstrap filter: move with $f_\theta\left(x_n\middle|\, x_{n-1}\right)$ and weight with $g_\theta\left(y_n\middle|\, x_n\right)$
  - Alternative route : weight with $p_\theta\left(y_n\middle|\, x_{n-1}\right)$ and then move with $p_\theta\left(x_n\middle|\, x_{n-1}, y_n\right)$

The auxiliary particle filter

- Alternative route : weight with $p_\theta\left(\left. y_n \right| x_{n-1}\right)$ and then move with $p_\theta\left(\left. x_n \right| x_{n-1}, y_n\right)$
  Recall

$$p_\theta\left(\left. x_n \right| x_{n-1}, y_n\right) = \frac{f_\theta\left(\left. x_n \right| x_{n-1}\right) g_\theta\left(\left. y_n \right| x_n\right)}{p_\theta\left(\left. y_n \right| x_{n-1}\right)}$$

  - (Pitt & Sheppard 99 JASA) Can reverse the steps:
    - move with $p_\theta\left(\left. x_n \right| x_{n-1}, y_n\right)$ weight with $p_\theta\left(\left. y_{n+1} \right| x_n\right)$ and then resample
- How could we use approximations:
  - move with $q_\theta\left(\left. x_n \right| x_{n-1}, y_n\right)$ and weight with $q_\theta\left(y_{n+1}, x_n\right)$

The auxiliary particle filter

- On approximations:
  - here $q_\theta\left(y_{n+1}, x_n\right)$ is not necessarily required to be a pdf
    - just an easy to evaluate non-negative function of (x_n, y_{n+1}).
    - often is called a score-function but this is misleading as the name appears elsewhere too
  - $q_\theta\left(\left. x_n \right| x_{n-1}, y_n\right)$ can be a good importance distribution
    - that takes into account the current observation

The auxiliary particle filter

- Instead of the original problem consider the target:

$$\begin{aligned}\tilde{\pi}_n\left(x_{0:n} \mid y_{0:n+1}\right) &\propto \eta_\theta\left(x_0\right) g_\theta\left(y_0 \mid x_0\right) q_\theta\left(y_1, x_0\right) \\ &\times \prod_{k=1}^{n} f_\theta\left(x_k \mid x_{k-1}\right) g_\theta\left(y_k \mid x_k\right) \frac{q_\theta\left(y_{k+1}, x_k\right)}{q_\theta\left(y_k, x_{k-1}\right)}\end{aligned}$$

- Note

$$q_\theta\left(y_1, x_0\right) \prod_{k=0}^{n} \frac{q_\theta\left(y_{k+1}, x_k\right)}{q_\theta\left(y_k, x_{k-1}\right)} = q_\theta\left(y_{n+1}, x_n\right)$$

- This means

$$\tilde{\pi}_n\left(x_{0:n} \mid y_{0:n+1}\right) \propto p_\theta\left(x_{0:n} \mid y_{0:n}\right) q_\theta\left(y_{n+1}, x_n\right)$$

The auxiliary particle filter

- What is the auxiliary PF?
  - it is a PF targetting $\tilde{\pi}_n$ using proposal $q_\theta\left(x_n \middle| y_n, x_{n-1}\right)$
- We will implement a PF targetting $\tilde{\pi}_n$ using as proposal $q_\theta\left(x_n \middle| y_n, x_{n-1}\right)$ and then reweight to get approximations for original π_n that is actually of interest. Why do we do this:
  - the PF for $\tilde{\pi}_n$ is more stable numerically
  - new likelihood $g_\theta\left(y_n \middle| x_n\right) \frac{q_\theta(y_{n+\mathbf{1}}, x_n)}{q_\theta(y_n, x_{n-\mathbf{1}})}$ might be less "peaky" or informative
  - $\tilde{\pi}_n$ closer to $\tilde{\pi}_{n-1}$

The auxiliary particle filter

- So in path space target can be written recursively

$$
\begin{aligned}
\tilde{\pi}_n\left(x_{0:n}\middle|\, y_{0:n+1}\right) &\propto \tilde{\pi}_{n-1}\left(x_{0:n-1}\middle|\, y_{0:n}\right) \\
&\times f_\theta\left(x_n\middle|\, x_{n-1}\right) g_\theta\left(y_n\middle|\, x_k\right) \frac{q_\theta\left(y_{n+1}, x_n\right)}{q_\theta\left(y_n, x_{n-1}\right)}
\end{aligned}
$$

  and proposal

$$
q(x_{0:n}) \propto \prod_{k=0}^{n} q_\theta\left(x_k | y_k, x_{k-1}\right)
$$

- This leads to the following incremental weights:

$$
\tilde{w}_n\left(x_n, x_{n-1}\right) = \frac{f_\theta\left(x_k\middle|\, x_{k-1}\right) g_\theta\left(y_k\middle|\, x_k\right) q_\theta\left(y_{n+1}, x_n\right)}{q_\theta\left(y_n, x_{n-1}\right) q_\theta\left(x_n | y_n, x_{n-1}\right)}
$$

The auxiliary particle filter

- We still need to compute approximations for $\Pi_n = p_\theta\left(x_{0:n} \middle| y_{0:n}\right)$
- Use the usual weights:

$$
\begin{aligned}
w_0\left(x_0\right) &= \frac{g_\theta\left(y_0|x_0\right)\eta_\theta\left(x_0\right)}{q_\theta\left(x_0|\, y_0\right)}, \\
w_n\left(x_{n-1:n}\right) &= \frac{g_\theta\left(y_n|x_n\right) f_\theta\left(x_n|x_{n-1}\right)}{q_\theta\left(x_n, y_n|x_{n-1}\right)} \text{ for } n \geq 1
\end{aligned}
$$

  where we denote
  - for $n \geq 1$, $q_\theta\left(x_n, y_n|x_{n-1}\right) = q_\theta\left(x_n|y_n, x_{n-1}\right) q_\theta\left(y_n, x_{n-1}\right)$
  - so note

$$
\tilde{w}_n\left(x_n, x_{n-1}\right) = w_n\left(x_n, x_{n-1}\right) q_\theta\left(y_{n+1}, x_n\right)
$$

The auxiliary particle filter

At time $n = 0$, for all $i \in \{1, ..., N\}$:

1. Sample $X_0^i \sim q_\theta(x_0 | y_0)$.
2. Compute $\overline{W}_1^i \propto w_0\left(X_0^i\right) q_\theta\left(y_1, X_0^i\right)$, $\sum_{i=1}^N \overline{W}_1^i = 1$.
3. Resample $\overline{X}_0^i \sim \sum_{i=1}^N \overline{W}_1^i \delta_{X_0^i}(dx_0)$.

At time $n \geq 1$, for all $i \in \{1, ..., N\}$:

1. Sample $X_n^i \sim q_\theta(x_n | y_n, \overline{X}_{n-1}^i)$ and set $X_{0:n}^i \leftarrow \left(\overline{X}_{0:n-1}^i, X_n^i\right)$.
2. Compute $\overline{W}_{n+1}^i \propto w_n\left(X_{n-1:n}^i\right) q_\theta\left(y_{n+1}, X_n^i\right)$, $\sum_{i=1}^N \overline{W}_{n+1}^i = 1$.
3. Resample $\overline{X}_{0:n}^i \sim \sum_{i=1}^N \overline{W}_{n+1}^i \delta_{X_{0:n}^i}(dx_{0:n})$.

The auxiliary particle filter

- BUT note we want the approximations of
  - $p_\theta\left(x_{0:n}\middle|\, y_{0:n}\right)$ and $p_\theta\left(y_n\middle|\, y_{0:n-1}\right)$
- These are given by:

$$\widehat{p}_\theta\left(dx_{0:n}\middle|\, y_{0:n}\right) = \sum_{i=1}^{N} W_n^i \delta_{X_{0:n}^i}\left(dx_{0:n}\right), \tag{1}$$

$$\widehat{p}_\theta\left(y_n\middle|\, y_{0:n-1}\right) = \left(\frac{1}{N}\sum_{i=1}^{N} w_n\left(X_{n-1:n}^i\right)\right)\left(\sum_{i=1}^{N} W_{n-1}^i q_\theta\left(y_n, X_{n-1}^i\right)\right) \tag{2}$$

where

$$W_n^i \propto w_n\left(X_{n-1:n}^i\right), \sum_{i=1}^{N} W_n^i = 1$$

and

$$\widehat{p}_\theta\left(y_0\right) = \frac{1}{N}\sum_{i=1}^{N} w_0\left(X_0^i\right).$$

Discussion on APF

- (Pitt & Sheppard 99 JASA) recommends using if available $q_\theta\left(x_n|y_n, x_{n-1}\right) = p_\theta\left(x_n|y_n, x_{n-1}\right)$ and $q_\theta\left(y_n, x_{n-1}\right) = p_\theta\left(y_n|x_{n-1}\right)$ or approximations of them
- What are we doing
  - we are changing carefully the weight so that algorithm is well behaved
    - by multiplying with something and dividing at the next step
- This is effective when
  - g_θ too informative or
  - in some other way Π_n is a bit different from Π_{n+1} so we need to bridge them in some way.

Discussion on APF: extensions

- Neat extension for discretised continuous time models (e.g. SDEs)
- Set
$$\frac{q\left(y_{k+1}, x_k\right)}{q\left(y_k, x_{k-1}\right)} = \prod_{m=1}^{M} \frac{r_{k,m}\left(y_{k+1}, y_k, x_{k,m}\right)}{r_{k,m-1}\left(y_{k+1}, y_k, x_{k,m-1}\right)}$$
with $X_{0,m} = X_{k-1}$and $X_{k,M} = X_k$.
- Doing the same thing as above means that you do intermediate M weight resample steps to process observation Y_{k+1}.
- Detailed exposition in (Del Moral, Murray 2015).

Discussion on extensions: tempering

- Another example that fits this framework is tempering with PF (Godsill & Clapp 2001)
- Consider

$$r_{k,m} = g(Y_{k+1}|x_{k,m})^{\phi_m}$$

$$r_{k,0} = g(Y_k|x_{k,0})$$

  with $\phi_M = 1$ and $0 < \phi_1 < \phi_2 < \ldots < \phi_m$
- Can tune ϕ_m according to ESS.
- Can use MCMC steps if dynamics cannot be split in m steps like continuous time models.
  - ie we combined together with resample move MCMC jittering
- Quite effective in high dimensions

Discussion: summary

- Path degeneracy can be addressed partially by:
  - adaptive resampling: applying resampling only when necessary
  - using MCMC moves to jitter the particles and reintroduce lost diversity in particle approximations
  - note that path degeneracy will be still present!
- Weight degeneracy can be addressed
  - by good selection of importance proposals
  - changing the target sequence to an easier problem as in APF
  - introducing intermediate artificial weighting-resampling sequence, e.g. tempering.
- Can use all ideas above together to get a very powerful algorithm
  - but also a bit complicated algorithm