A Tutorial on Particle Filtering

Group 15

Leo Carlsson Salma El Alaoui Talibi Ludvig Ericson
Paulina Hensman Hlynur Davíð Hlynsson

Kungliga Tekniska Högskolan

DD2434 Machine Learning, Advanced, 2016

Outline

HMMs and Stochastic Volatility

Sequential Monte Carlo applied to Filtering

Handling Degeneracy: Resampling

Method comparison

Real life stock data application

Good old HMMs

- Discrete time hidden Markov models
  - Observable emissions $\{Y_n\}_{n=1}^{T}$
  - Unobservable latent states $\{X_n\}_{n=1}^{T}$

Good old HMMs

- Discrete time hidden Markov models
  - Observable emissions $\{Y_n\}_{n=1}^T$
  - Unobservable latent states $\{X_n\}_{n=1}^T$
- A stochastic process
  - Initial and transition probabilities

$$X_1 \sim \mu(x_1) \quad \text{and} \quad X_n | X_{n-1} \sim f(x_n | x_{n-1})$$

  - Emission probability

$$Y_n | X_n \sim g(y_n | x_n)$$

Goal: infer state given the observations

- Filtering: sample from $p(x_n|y_{1:n})$ for $n \geq 1$
- Optimal filtering problem: estimate $p(x_{1:n}|y_{1:n})$ for $n \geq 1$
- Marginal likelihood: estimate $p(y_{1:n})$ for $n \geq 1$
- Smoothing: estimate $p(x_n|y_{1:T})$ for $n = 1 \dots T$

Example: Stochastic Volatility

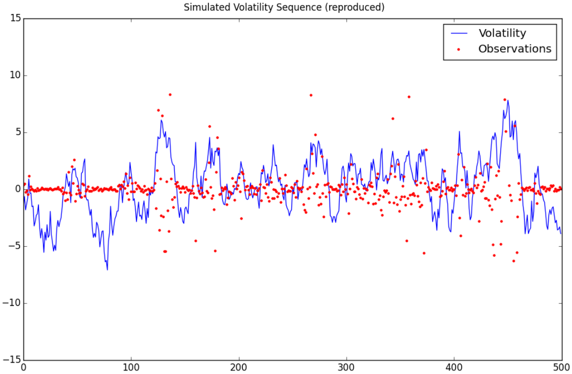

Figure: Simulated model output. The *variance* is an exponential function of the hidden state. Often used in econometrics.

Bayesian Inference

- Bayesian inference leads to two steps:
  - Updating step:

$$p(x_n|y_{1:n}) = \frac{g(y_n|x_n)p(x_n|y_{1:n-1})}{p(y_n|y_{1:n-1})}$$

  - Prediction step:

$$p(x_n|y_{1:n-1}) = \int f(x_n|x_{n-1})p(x_{n-1}|y_{1:n-1})dx_{n-1}$$

Bayesian Inference

- Bayesian inference leads to two steps:
  - Updating step:

$$p(x_n|y_{1:n}) = \frac{g(y_n|x_n)p(x_n|y_{1:n-1})}{p(y_n|y_{1:n-1})}$$

  - Prediction step:

$$p(x_n|y_{1:n-1}) = \int f(x_n|x_{n-1})p(x_{n-1}|y_{1:n-1})dx_{n-1}$$

- **Problem**
  - These distributions are often intractable in closed-form
  - Particularly in the case of non-linear, non-Gaussian models

The Monte Carlo Approximation

- **Solution**
  - Approximate the target distributions ($p(x_{1:n}|y_{1:n})$ for Filtering) by using a large number N of samples (or *particles*) from that distribution
  - The Monte Carlo Approximation converges to the actual distribution as N increases towards ∞

$$\hat{p}(x_{1:n}|y_{1:n}) = \frac{1}{N}\sum_{i=1}^{N} \delta_{X^i_{1:n}}(x_{1:n})$$

The Monte Carlo Approximation

- **Solution**
  - Approximate the target distributions ($p(x_{1:n}|y_{1:n})$ for Filtering) by using a large number N of samples (or *particles*) from that distribution
  - The Monte Carlo Approximation converges to the actual distribution as N increases towards ∞

$$\hat{p}(x_{1:n}|y_{1:n}) = \frac{1}{N}\sum_{i=1}^{N}\delta_{X^i_{1:n}}(x_{1:n})$$

- **New Problems**

The Monte Carlo Approximation

- **Solution**
  - Approximate the target distributions ($p(x_{1:n}|y_{1:n})$ for Filtering) by using a large number N of samples (or *particles*) from that distribution
  - The Monte Carlo Approximation converges to the actual distribution as N increases towards ∞

$$\hat{p}(x_{1:n}|y_{1:n}) = \frac{1}{N}\sum_{i=1}^{N}\delta_{X_{1:n}^{i}}(x_{1:n})$$

- **New Problems**
  1. It is difficult to sample from **complex/high-dimensional** distributions

The Monte Carlo Approximation

- **Solution**
  - Approximate the target distributions ($p(x_{1:n}|y_{1:n})$ for Filtering) by using a large number N of samples (or *particles*) from that distribution
  - The Monte Carlo Approximation converges to the actual distribution as N increases towards ∞

$$\hat{p}(x_{1:n}|y_{1:n}) = \frac{1}{N}\sum_{i=1}^{N}\delta_{X^i_{1:n}}(x_{1:n})$$

- **New Problems**
  1. It is difficult to sample from **complex/high-dimensional** distributions
  2. Sampling is computationally expensive (increasing at least **linearly** with n)

Solution 1: Importance Sampling

- We introduce an **importance density** $q_n(x_{1:n})$ such as :

$$p(x_{1:n}|y_{1:n}) = \frac{w_n(x_{1:n})q_n(x_{1:n})}{Z_n}$$

Solution 1: Importance Sampling

- We introduce an **importance density** $q_n(x_{1:n})$ such as :

$$p(x_{1:n}|y_{1:n}) = \frac{w_n(x_{1:n})q_n(x_{1:n})}{Z_n}$$

- $q_n(x_{1:n})$ is selected such that:
  - It is easy to sample from
  - It minimises the variance of the weights, ie: it is close to $p(x_{1:n}|y_{1:n})$

Solution 1: Importance Sampling

- We introduce an **importance density** $q_n(x_{1:n})$ such as :

$$p(x_{1:n}|y_{1:n}) = \frac{w_n(x_{1:n})q_n(x_{1:n})}{Z_n}$$

- $q_n(x_{1:n})$ is selected such that:
  - It is easy to sample from
  - It minimises the variance of the weights, ie: it is close to $p(x_{1:n}|y_{1:n})$
- The density can then be estimated as:

$$\hat{p}_n(x_{1:n}|y_{1:n}) = \sum_{i=1}^{N} W_n^i \delta_{X_{1:n}^i}(x_{1:n})$$

Solution 2: Sequential Importance Sampling

- We select an importance density so that:

$$q_n(x_{1:n}) = q_{n-1}(x_{1:n-1})q_n(x_n|x_{1:n-1}) = q_{n-1}(x_{1:n-1})q(x_n|y_n, x_{n-1})$$

Solution 2: Sequential Importance Sampling

- We select an importance density so that:

$$q_n(x_{1:n}) = q_{n-1}(x_{1:n-1})q_n(x_n|x_{1:n-1}) = q_{n-1}(x_{1:n-1})q(x_n|y_n, x_{n-1})$$

- At the first time step, we draw samples $\{X_1^i\}_{1\leq i\leq N}$ from $q_1(x_1|y_1)$ and compute their weights W_1^i

Solution 2: Sequential Importance Sampling

- We select an importance density so that:

$$q_n(x_{1:n}) = q_{n-1}(x_{1:n-1})q_n(x_n|x_{1:n-1}) = q_{n-1}(x_{1:n-1})q(x_n|y_n, x_{n-1})$$

- At the first time step, we draw samples $\{X_1^i\}_{1\leq i\leq N}$ from $q_1(x_1|y_1)$ and compute their weights W_1^i
- For the next steps, we sample $\{X_n^i\}_{1\leq i\leq N}$ from $q_n(x_n|y_n, X_{1:n-1}^i)$ and compute their weights W_n^i

SIS estimates

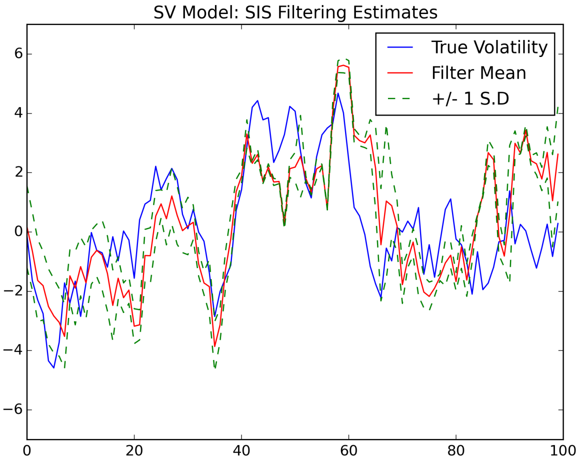

Figure: Filtering estimates obtained for the stochastic volatility model using SIR

Weight degeneracy

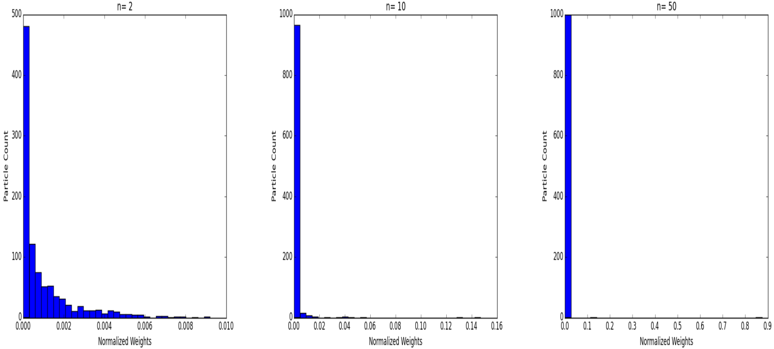

Figure: Distributions of the particle weights obtained with SIS for the stochastic volatility model at iterations 2, 5, 50

Problems

- The variance of these estimates increases (often exponentially) with n
- We don't want to focus computations on regions of high probability, i.e.: we do not want to carry forward particles with low weight.

Solution: Resampling

- We sample again from the approximation distributions, ie from: $\hat{p}_n(x_{1:n}|y_{1:n})$ to obtain N new equally weighted particles

Solution: Resampling

- We sample again from the approximation distributions, ie from: $\hat{p}_n(x_{1:n}|y_{1:n})$ to obtain N new equally weighted particles
- Each particle is associated with a number of "offspring" samples, so that:
  - particles with large importance weights are replicated
  - particles with small importance weights are eliminated

Weight degeneracy

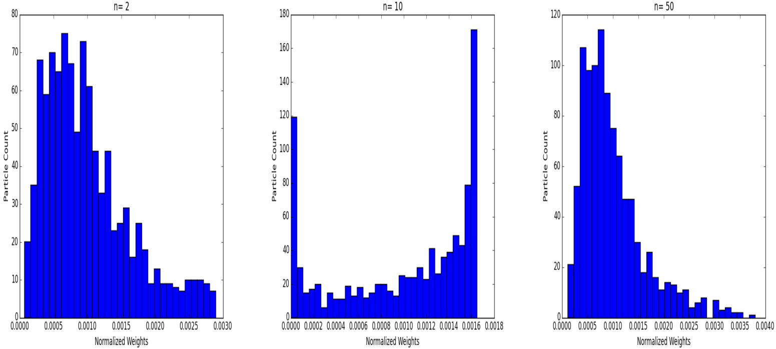

Figure: Distributions of the particle weights obtained with SIR for the stochastic volatility model at iterations 2, 5, 50

SIR estimates

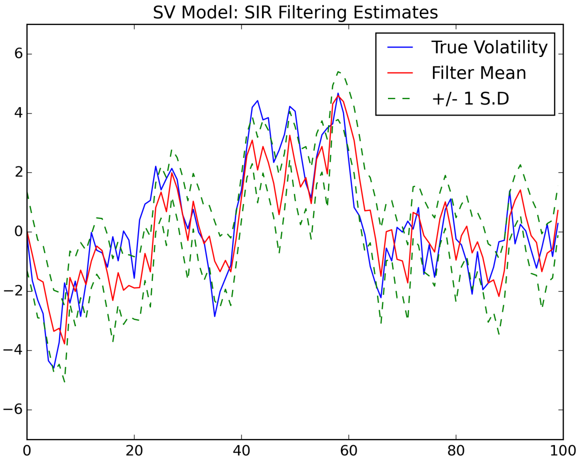

Figure: Filtering estimates obtained for the stochastic volatility model using SIR

Path degeneracy

- If the variance if the importance weights is high, it will be necessary to resample frequently
- ie for the early time steps, $p_n(x_{1:n}|y_{1:n})$ will be approximated by a single particle
- Only the most recent particles $\{X_n^i\}$ are sampled at time n, but the path values remain $\{X_{1:n-1}^i\}$ are fixed

Reducing degeneracy

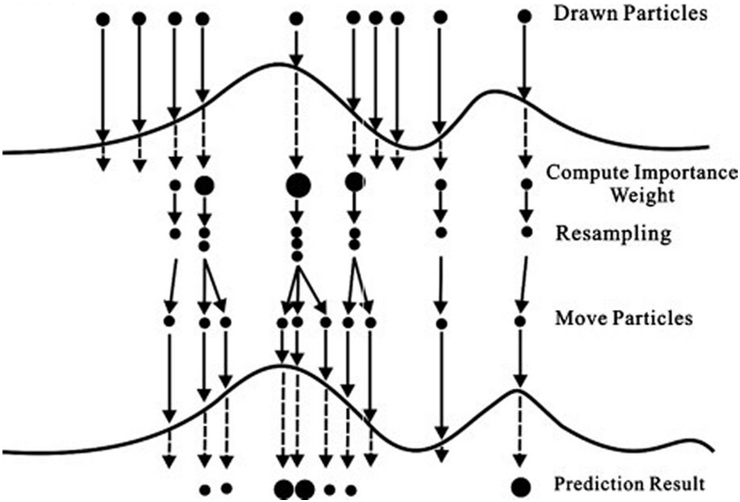

Figure: Path degeneracy for smoothing distributions

Solution: Resample-move

- A method to reduce degeneracy
- At each time step t we "jitter" the last L particles in each chain
- The particles are resampled from a distribution that is conditional on estimated past and future states
- Direct sampling from such a distribution is hard so we use a Metropolis-Hastings step

Solution: Resample-move

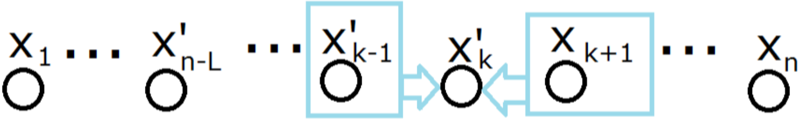

Figure: In a chain a L steps ago

Metropolis-Hastings

- Construct a proposal distribution

$$\mathcal{N}\left(x'_k; \frac{\alpha x'_{k-1} + \frac{1}{\alpha} x_{k+1}}{2}, \frac{\sigma^2}{2}\right)$$

- Accept this with probability

$$\min\left(1, \frac{g(y_k|x'_k) f(x_{k+1}|x'_k) f(x'_k|x'_{k-1}) q(x_k|x'_{k-1}, x_{k+1})}{g(y_k|x_k) f(x_{k+1}|x_k) f(x_k|x'_{k-1}) q(x'_k|x'_{k-1}, x_{k+1})}\right)$$

Solution: Block sampling

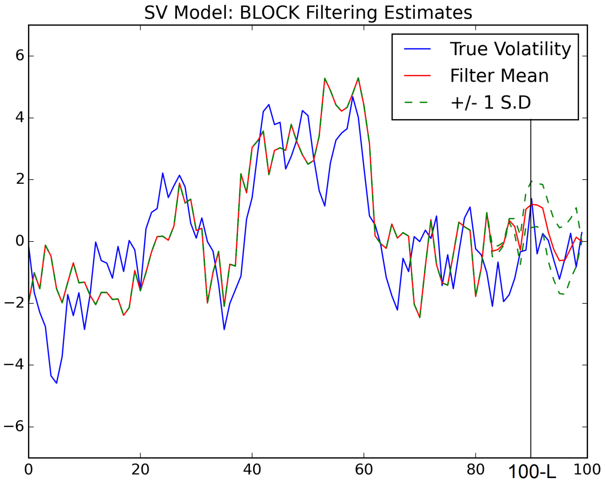

Figure: Block Sampling samples within a fixed lag

Comparison of methods

Model	Average absolute error, full series	Average absolute error, final 5	Average standard deviation
SIS	1.43	1.90	0.64
SIR	0.90	0.87	1.10
MCMC	0.90	0.88	1.10
Block	1.12	0.83	0.16

Table: Performance of all models on data generated by the Stochastic Volatility model.

In real life: Google closing prices

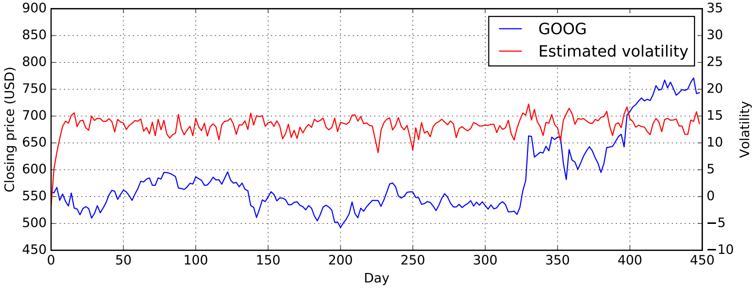

Figure: Estimated volatility of GOOG closing prices, March 2014 to January 2016

Conclusion

- More advanced methods do not necessarily outperform simpler ones
- Good proposal distributions are needed for success
- Choose the right method for your problem!

References I

A. Doucet, A. M. Johansen
A Tutorial on Particle Filtering and Smoothing:
Fifteen years later
2008