An Brief Overview of Particle Filtering

Adam M. Johansen

a.m.johansen@warwick.ac.uk
www2.warwick.ac.uk/fac/sci/statistics/staff/academic/johansen/talks/

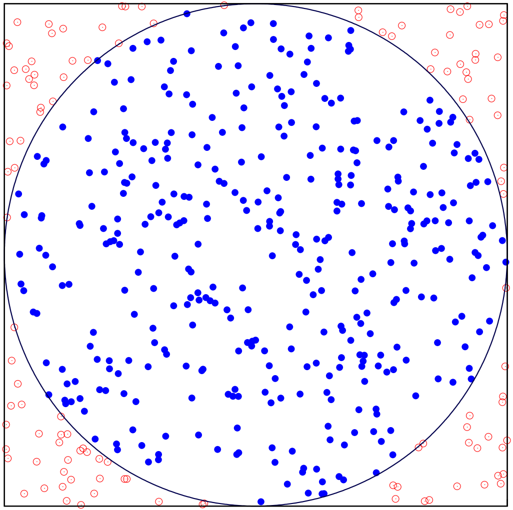
May 11th, 2010
Warwick University Centre for Systems Biology

Outline

- Background
  - Monte Carlo Methods
  - Hidden Markov Models / State Space Models
- Sequential Monte Carlo
  - Filtering
    - Sequential Importance Sampling
    - Sequential Importance Resampling
    - Advanced methods
  - Smoothing
  - Parameter Estimation

Background

Estimating π



- Rain is uniform.
- Circle is inscribed in square.
- $A_{\text{square}} = 4r^2$.
- $A_{\text{circle}} = \pi r^2$.
- $p = \frac{A_{\text{circle}}}{A_{\text{square}}} = \frac{\pi}{4}$.
- 383 of 500 "successes".
- $\hat{\pi} = 4\frac{383}{500} = 3.06$.
- Also obtain confidence intervals.

The Monte Carlo Method

- Given a probability density, p,

$$I = \int_E \varphi(x)p(x)dx$$

- Simple Monte Carlo solution:
  - Sample $X_1, \ldots, X_N \overset{iid}{\sim} p$.
  - Estimate $\hat{I} = \frac{1}{N}\sum_{i=1}^{N} \varphi(X_N)$.
- Justified by the law of large numbers…
- and the central limit theorem.
- Can also be viewed as approximating $\pi(dx) = p(x)dx$ with

$$\hat{\pi}^N(dx) = \frac{1}{N}\sum_{i=1}^{N} \delta_{X_i}(dx).$$

Importance Sampling

- Given q, such that
  - $p(x) > 0 \Rightarrow q(x) > 0$
  - and $p(x)/q(x) < \infty$,

  define $w(x) = p(x)/q(x)$ and:

$$I = \int \varphi(x)p(x)dx = \int \varphi(x)w(x)q(x)dx.$$

- This suggests the importance sampling estimator:
  - Sample $X_1, \ldots, X_N \overset{iid}{\sim} q$.
  - Estimate $\hat{I} = \frac{1}{N} \sum_{i=1}^{N} w(X_i)\varphi(X_i)$.
- Can also be viewed as approximating $\pi(dx) = p(x)dx$ with

$$\hat{\pi}^n(dx) = \frac{1}{n} \sum_{i=1}^{n} w(x_i)\delta_{X_i}(dx).$$

Variance and Importance Sampling

- Variance depends upon proposal:

$$\begin{aligned}
\mathsf{Var}(\hat{I}) =& \mathbb{E}\left[\left(\frac{1}{N}\sum_{i=1}^{N} w(X_i)\varphi(X_i) - I\right)^2\right] \\
=& \frac{1}{N^2}\sum_{i=1}^{N} \mathbb{E}\left[\left(w(X_i)^2\varphi(X_i)^2 - I^2\right)\right] \\
=& \frac{1}{N}\left(\mathbb{E}_q\left[w(X)^2\varphi(X)^2\right] - I^2\right) \\
=& \frac{1}{N}\left(\mathbb{E}_p\left[w(X)\varphi(X)^2\right] - I^2\right)
\end{aligned}$$

- For positive φ, minimized by:

$$q(x) \propto p(x)\varphi(x) \Rightarrow w(x)\varphi(x) \propto 1$$

Self-Normalised Importance Sampling

- Often, p is known only up to a normalising constant.
- As $\mathbb{E}_q(Cw\varphi) = C\mathbb{E}_p(\varphi)\ldots$
- If $v(x) = Cw(x)$, then

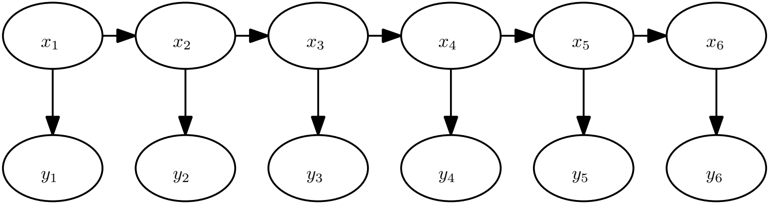
$$\frac{\mathbb{E}_q(v\varphi)}{\mathbb{E}_q(v\mathbf{1})} = \frac{\mathbb{E}_q(Cw\varphi)}{\mathbb{E}_q(Cw\mathbf{1})} = \frac{C\mathbb{E}_p(\varphi)}{C\mathbb{E}_p(\mathbf{1})} = \mathbb{E}_p(\varphi).$$

- Estimate the numerator and denominator with the same sample:

$$\hat{I} = \frac{\sum_{i=1}^{N} v(X_i)\varphi(X_i)}{\sum_{i=1}^{N} v(X_i)}.$$

- Biased for finite samples, but consistent.
- Typically reduces variance.

Hidden Markov Models / State Space Models



- Unobserved Markov chain $\{X_n\}$ transition f.
- Observed process $\{Y_n\}$ conditional density g.
- Density:

$$p(x_{1:n}, y_{1:n}) = f_1(x_1)g(y_1|x_1)\prod_{i=2}^{n} f(x_i|x_{i-1})g(y_i|x_i).$$

Inference in HMMs

- Given $y_{1:n}$:
  - What is $x_{1:n}$,
  - in particular, what is x_n,
  - and what about x_{n+1}?
- f and g may have unknown parameters θ.
- We wish to obtain estimates for $n = 1, 2, \ldots$.

Some Terminology

Three main problems:

- Given known parameters θ, what are:
  - The *filtering and prediction* distributions:

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

$$p(x_{n+1}|y_{1:n}) = \int p(x_{n+1}|x_n)p(x_n|y_{1:n})dx_n$$

  - The *smoothing* distributions:

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

- And how can we *estimate static parameters*:

$$p(\theta|y_{1:n}) \text{ and } p(x_{1:n}, \theta|y_{1:n})?$$

Formal Solutions

- Filtering and Prediction Recursions:

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

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

- Smoothing:

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

Importance Sampling in This Setting

- Given $p(x_{1:n}|y_{1:n})$ for $n = 1, 2, \ldots$.
- We could sample from a sequence $q_n(x_{1:n})$ for each n.
- Or we could let $q_n(x_{1:n}) = q_n(x_n|x_{1:n-1})q_{n-1}(x_{1:n-1})$ and re-use our samples.
- The importance weight decomposes:

$$
\begin{aligned}
w_n(x_{1:n}) &\propto \frac{p(x_{1:n}|y_{1:n})}{q_n(x_n|x_{1:n-1})q_{n-1}(x_{1:n-1})} \\
&= \frac{p(x_{1:n}|y_{1:n})}{q_n(x_n|x_{1:n-1})p(x_{1:n-1}|y_{1:n-1})} w_{n-1}(x_{1:n-1}) \\
&= \frac{f(x_n|x_{n-1})g(y_n|x_n)}{q_n(x_n|x_{n-1})p(y_n|y_{1:n-1})} w_{n-1}(x_{1:n-1})
\end{aligned}
$$

Sequential Importance Sampling – Prediction & Update

- A first "particle filter":
  - Simple default: $q_n(x_n|x_{n-1}) = f(x_n|x_{n-1})$.
  - Importance weighting becomes:

$$w_n(x_{1:n}) = w_{n-1}(x_{1:n-1}) \times g(y_n|x_n)$$

- Algorithmically, at iteration n:
  - Given $\{W_{n-1}^i, X_{1:n-1}^i\}$ for $i = 1, \ldots, N$:
    - Sample $X_n^i \sim f(\cdot|X_{n-1}^i)$ *(prediction)*
    - Weight $W_n^i \propto W_{n-1}^i g(y_n|X_n^i)$ *(update)*

Example: Almost Constant Velocity Model

- States: $x_n = [s_n^x \; u_n^x \; s_n^y \; u_n^y]^T$
- Dynamics: $x_n = Ax_{n-1} + \epsilon_n$

$$\begin{bmatrix} s_n^x \\ u_n^x \\ s_n^y \\ u_n^y \end{bmatrix} = \begin{bmatrix} 1 & \Delta t & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \Delta t \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} s_{n-1}^x \\ u_{n-1}^x \\ s_{n-1}^y \\ u_{n-1}^y \end{bmatrix} + \epsilon_n$$

- Observation: $y_n = Bx_n + \nu_n$

$$\begin{bmatrix} r_n^x \\ r_n^y \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} s_n^x \\ u_n^x \\ s_n^y \\ u_n^y \end{bmatrix} + \nu_n$$

Better Proposal Distributions

- The importance weighting is:

$$w_n(x_{1:n}) \propto \frac{f(x_n|x_{n-1})g(y_n|x_n)}{q_n(x_n|x_{n-1})} w_{n-1}(x_{1:n-1})$$

- Optimally, we'd choose:

$$q_n(x_n|x_{n-1}) \propto f(x_n|x_{n-1})g(y_n|x_n)$$

- Typically impossible; but good approximation is possible.
  - Can obtain *much* better performance.
  - But, eventually the approximation variance is too large.

Resampling

- Given $\{W_n^i, X_{1:n}^i\}$ targetting $p(x_{1:n}|y_{1:n})$.
- Sample

$$\widetilde{X}_{1:n}^i \sim \sum_{j=1}^{N} W_n^i \delta_{X_{1:n}^j}.$$

- And $\{1/N, \widetilde{X}_{1:n}^i\}$ also targets $p(x_{1:n}|y_{1:n})$.
- Such steps can be incorporated into the SIS algorithm.

Sequential Importance Resampling

- Algorithmically, at iteration n:
  - Given $\{W_{n-1}^i, X_{n-1}^i\}$ for $i = 1, \ldots, N$:
  - Resample, obtaining $\{1/N, \widetilde{X}_{n-1}^i\}$.
    - Sample $X_n^i \sim q(\cdot|\widetilde{X}_{n-1}^i)$
    - Weight $W_n^i \propto \frac{f(X_n^i|\widetilde{X}_{n-1}^i)g(y_n|X_n^i)}{q(X_n^i|\widetilde{X}_{n-1}^i)}$
- Actually:
  - You can resample more efficiently.
  - It's only necessary to resample some of the time.

What else can we do?

Some common improvements:

- Incorporate MCMC (7)
- Use the next observation before resampling – auxiliary particle filters (11, 9)
- Sample new values for recent states (6)
- Explicitly approximate the marginal filter (10)

Smoothing:

- Resampling helps filtering.
- But it eliminates unique particle values.
- Approximating $p(x_{1:n}|y_{1:n})$ or $p(x_m|y_{1:n})$ fails for large n.

Smoothing Strategies

Some approaches:

- Fixed-lag smoothing:

$$p(x_{n-L}|y_{1:n}) \approx p(x_{n-L}|y_{1:n-1})$$

  for large enough L.

- Particle implementation of forward-backward algorithm:

$$p(x_{1:n}|y_{1:n}) = p(x_n|y_{1:n}) \prod_{m=1}^{n-1} p(x_{n-1}|x_n, y_{1:n})$$

  Doucet et al. have developed a forward-only variant.

- Particle implementation of two-filter formula:

$$p(x_m|y_{1:n}) \propto p(x_m|y_{1:m})p(y_{m+1:n}|x_m)$$

Static Parameters

If $f(x_n|x_{n-1})$ and $g(y_n|x_n)$ depend upon θ:

- We could set $x'_n = (x_n, \theta_n)$ and

$$f'(x'_n|x'_{n-1}) = f(x_n|x_{n-1})\delta_{\theta_{n-1}}(\theta_n)$$

- Degeneracy causes difficulties:
  - Resampling eliminates values.
  - We never propose new values.
  - MCMC transitions don't mix well.

Approaches to Parameter Estimation

- Artificial dynamics:

$$\theta_n = \theta_{n-1} + \epsilon_n.$$

- Iterated filtering.
- Sufficient statistics.
- Various online likelihood approaches.
- Practical filtering.

$$\vdots$$

Summary

SMCTC: C++ Template Class for SMC Algorithms

- Implementing SMC algorithms in C/C++ isn't hard.

- Software for implementing general SMC algorithms.
- C++ element largely confined to the library.
- Available (under a GPL-3 license from)

  `www2.warwick.ac.uk/fac/sci/statistics/staff/academic/johansen/smctc/`

  or type "smctc" into google.
- Example code includes simple particle filter.

In Conclusion (with references)

- Sequential Monte Carlo methods ("Particle Filters") can approximate filtering and smoothing distributions (5).
- Proposal distributions matter.
- Resampling:
  - is necessary (2),
  - but not every iteration ,
  - and need not be multinomial (4).
- Smoothing & parameter estimation are harder.
- Software for easy implementation exists:
  - PFLib (1)
  - SMCTC (8)
- Actually, similar techniques apply elsewhere (3)

References

[1] L. Chen, C. Lee, A. Budhiraja, and R. K. Mehra. PFlib: An object oriented matlab toolbox for particle filtering. In *Proceedings of SPIE Signal Processing, Sensor Fusion and Target Recognition XVI*, volume 6567, 2007.

[2] P. Del Moral. *Feynman-Kac formulae: genealogical and interacting particle systems with applications*. Probability and Its Applications. Springer Verlag, New York, 2004.

[3] P. Del Moral, A. Doucet, and A. Jasra. Sequential Monte Carlo samplers. *Journal of the Royal Statistical Society B*, 63(3):411–436, 2006.

[4] R. Douc and O. Cappé. Comparison of resampling schemes for particle filters. In *Proceedings of the 4th International Symposium on Image and Signal Processing and Analysis*, volume I, pages 64–69. IEEE, 2005.

[5] A. Doucet and A. M. Johansen. A tutorial on particle filtering and smoothing: Fiteen years later. In D. Crisan and B. Rozovsky, editors, *The Oxford Handbook of Nonlinear Filtering*. Oxford University Press, 2010. To appear.

[6] A. Doucet, M. Briers, and S. Sénécal. Efficient block sampling strategies for sequential Monte Carlo methods. *Journal of Computational and Graphical Statistics*, 15(3):693–711, 2006.

[7] W. R. Gilks and C. Berzuini. Following a moving target – Monte Carlo inference for dynamic Bayesian models. *Journal of the Royal Statistical Society B*, 63:127–146, 2001.

[8] A. M. Johansen. SMCTC: Sequential Monte Carlo in C++. *Journal of Statistical Software*, 30(6):1–41, April 2009.

[9] A. M. Johansen and A. Doucet. A note on the auxiliary particle filter. *Statistics and Probability Letters*, 78(12):1498–1504, September 2008. URL `http://dx.doi.org/10.1016/j.spl.2008.01.032`.

[10] M. Klass, N. de Freitas, and A. Doucet. Towards practical n^2 Monte Carlo: The marginal particle filter. In *Proceedings of Uncertainty in Artificial Intelligence*, 2005.

[11] M. K. Pitt and N. Shephard. Filtering via simulation: Auxiliary particle filters. *Journal of the American Statistical Association*, 94(446):590–599, 1999.

SMC Sampler Outline

- Given a sample $\{X_{1:n-1}^{(i)}\}_{i=1}^{N}$ targeting $\widetilde{\eta}_{n-1}$,
- sample $X_n^{(i)} \sim K_n(X_{n-1}^{(i)}, \cdot)$,
- calculate

$$W_n(X_{1:n}^{(i)}) = \frac{\eta_n(X_n^{(i)}) L_{n-1}(X_n^{(i)}, X_{n-1}^{(i)})}{\eta_{n-1}(X_{n-1}^{(i)}) K_n(X_{n-1}^{(i)}, X_n^{(i)})}.$$

- Resample, yielding: $\{X_{1:n}^{(i)}\}_{i=1}^{N}$ targeting $\widetilde{\eta}_n$.
- Hints that we'd like to use

$$L_{n-1}(x_n, x_{n-1}) = \frac{\eta_{n-1}(x_{n-1}) K_n(x_{n-1}, x_n)}{\int \eta_{n-1}(x'_{n-1}) K_n(x'_{n-1}, x_n)}.$$