UPPSALA UNIVERSITET

Sequential Monte Carlo methods

Lecture 9 – Maximum likelihood parameter estimation

Andreas Lindholm, Uppsala University

2019-08-27

Aim: Open up for using the particle filter for inference about parameters θ (and not only states X_t) in state-space models.

Outline:

1. The particle filter as likelihood estimator
2. Maximum likelihood estimation of state-space models
    a. Direct optimization
    b. Expectation maximization

State-space models

From lecture 2:

$$X_t = f(X_{t-1}, \theta) + V_t,$$
$$Y_t = g(X_t, \theta) + E_t,$$

where X_t are the states and θ the model parameters.

Only (but important!) **difference: X_t depends on t, whereas θ doesn't.**

The particle filter assumes θ is known and computes $p(x_t \mid y_{1:t}, \theta)$.

The particle filter assumes θ is known and computes $p(x_t \mid y_{1:t}, \theta)$.

Inference about θ requires

$$p(\theta \mid y_{1:T}) \text{ (Posterior; Bayesian inference)}$$

or

$$p(y_{1:T} \mid \theta) \text{ (Likelihood function; Fisherian inference/maximum likelihood).}$$

$$p(\theta \mid y_{1:T}) = \frac{p(y_{1:T} \mid \theta)p(\theta)}{p(y_{1:T})}$$

This lecture: Focus on maximum likelihood. More on the Bayesian setting in later lectures.

Maximum likelihood problem: Select θ such that the observed data $y_{1:T}$ is as likely as possible to have been observed, i.e.,

$$\widehat{\theta} = \arg\max_{\theta} p(y_{1:T} \mid \theta)$$

Particle filter as likelihood estimator

$$
\begin{aligned}
p(y_{1:T} \mid \theta) &= \prod_{t=1}^{T} p(y_t \mid y_{1:t-1}, \theta), \\
p(y_t \mid y_{1:t-1}, \theta) &= \int p(y_t, x_t \mid y_{1:t-1}, \theta) dx_t = \\
= \int p(y_t \mid x_t, \theta) & \underbrace{p(x_t \mid y_{1:t-1}, \theta)}_{\overset{\text{bPF}}{\approx} \sum_{i=1}^{N} \frac{1}{N} \delta_{x_t^i}(x_t)} dx_t \approx \\
\approx \frac{1}{N} \sum_{i=1}^{N} & p(y_t \mid x_t^i, \theta) = \frac{1}{N} \sum_{i=1}^{N} \widetilde{w}_t^i \\
\Rightarrow p(y_{1:T} \mid \theta) &\approx \prod_{t=1}^{T} \left(\frac{1}{N} \sum_{i=1}^{N} \widetilde{w}_t^i \right)
\end{aligned}
$$

(w_t^i are the *unnormalized* weights)

UPPSALA UNIVERSITET

Algorithm 1 Bootstrap particle filter (for $i = 1, \ldots, N$)

1. **Initialization ($t = 0$):**
   (a) Sample $x_0^i \sim p(x_0 \mid \theta)$.
   (b) Set initial weights: $w_0^i = 1/N$.
2. **for** $t = 1$ **to** T **do**
   (a) Resample: sample ancestor indices $a_t^i \sim \mathcal{C}(\{w_{t-1}^j\}_{j=1}^N)$.
   (b) Propagate: sample $x_t^i \sim p(x_t \mid x_{t-1}^{a_t^i}, \theta)$.
   (c) Weight: compute $\widetilde{w}_t^i = p(y_t \mid x_t^i, \theta)$ and normalize $w_t^i = \widetilde{w}_t^i / \sum_{j=1}^N \widetilde{w}_t^j$.

$$p(y_{1:T} \mid \theta) \approx \prod_{t=1}^{T} \left(\frac{1}{N} \sum_{i=1}^{N} \widetilde{w}_t^i \right)$$

Log-weights: an important practical aspect

For realistic problems, $\widetilde{w}_t^i$ might be smaller than machine precision $\to \widetilde{w}_t^i = 0$ on your computer.

Use **shifted log-weights** v_t^i!

$$v_t^i = \log \widetilde{w}_t^i - c_t, \quad c_t = \max\{\log \widetilde{w}_t^1, \dots, \log \widetilde{w}_t^N\}$$

Implement your particle filter using shifted log-weights! Store $\{v_t^i\}_{i=1}^N$ and c_t.

From this, the likelihood estimate is obtained

$$\prod_{t=1}^T \left(\frac{1}{N} \sum_{i=1}^N \widetilde{w}_t^i \right) = \prod_{t=1}^T \exp\left(c_t + \log \sum_{i=1}^N e^{v_t^i} - \log N \right)$$

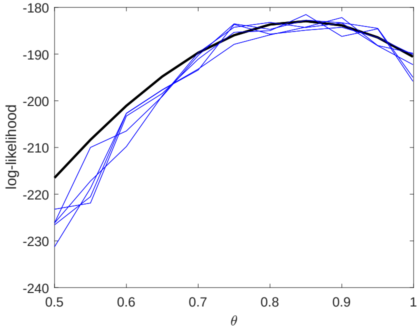
Also the normalized weights $\{w_t^i\}_{i=1}^N$ can be computed from $\{v_t^i\}_{i=1}^N$,

$$w_t^i = \frac{\widetilde{w}_t^i}{\sum_{j=1}^N \widetilde{w}_t^j} = \frac{e^{v_t^i + c_t}}{\sum_{j=1}^N e^{v_t^j + c_t}} = \frac{e^{v_t^i}}{\sum_{j=1}^N e^{v_t^j}}$$

Simple LG-SSM,

$$X_t = \theta X_{t-1} + V_t, \qquad V_t \sim \mathcal{N}(0,1),$$
$$Y_t = X_t + E_t, \qquad E_t \sim \mathcal{N}(0,1).$$

Task: estimate $p(y_{1:100} \mid \theta)$ for a simulated data set. True $\theta^\star = 0.9$.



Black line – true likelihood computed using the Kalman filter.

Blue thin lines – 5 different likelihood estimates $\widehat{p}^N(y_{1:100} \mid \theta)$ computed using a bootstrap particle filter with $N = 100$ particles.

The particle filter as likelihood estimator

- **Good news:** Each run of the particle filter returns an estimate of $p(y_{1:T} \mid \theta)$ — in addition to the state estimates!
- **Challenge:** The particle filter contains randomness $\rightarrow$ the estimate of $p(y_{1:T} \mid \theta)$ contains randomness or 'noise'.
- More on its stochastic properties in the next lecture.

Direct optimization

$$\widehat{\theta} = \arg\max_{\theta} p(y_{1:T} \mid \theta)$$

Can we use standard optimization routines?

Say, `scipy.optimize.minimize(fun=-my_BPF_function,x0 = .2)`

No. The evaluation of the cost function is 'noisy'.

Solution: Use (or design) **probabilistic optimization** methods that can work with noisy cost functions.

For example using Gaussian processes

Estimating likelihood gradients

We can also get noisy approximations for the gradient of the likelihood.

Fisher's identity $\nabla_\theta \log p(y_{1:T} \mid \theta) = \mathbb{E}_\theta[\nabla_\theta \log p(x_{1:T}, y_{1:T} \mid \theta) \mid y_{1:T}]$,

where

$$\nabla_\theta \log p(x_{1:T}, y_{1:T} \mid \theta) = \sum_{t=1}^{T} \nabla_\theta \log p(x_t \mid x_{t-1}, \theta) + \nabla_\theta \log p(y_t \mid x_t, \theta),$$

$$\Rightarrow \nabla_\theta \log p(y_{1:T} \mid \theta) =$$
$$\sum_{t=1}^{T} \int [\nabla_\theta \log p(x_t \mid x_{t-1}, \theta) + \nabla_\theta \log p(y_t \mid x_t, \theta)]\, p(x_{t-1:t} \mid y_{1:T}, \theta) \mathrm{d}x_{t-1:t}.$$

Here, $p(x_{t-1:t} \mid y_{1:T}, \theta)$ requires a particle *smoother*. Several SMC-based alternative exists, but are not in this course.

As an alternative to direct optimization of $p(y_{1:T} \mid \theta)$, we can use the **Expectation Maximization** (EM) method.

Dempster, Arthur P., Nan M. Laird, and Donald B. Rubin. **Maximum likelihood from incomplete data via the EM algorithm.** *Journal of the Royal Statistical Society: Series B (Methodological)*, 39.1 (1977): 1-22..

Idea:

(E) Let $\mathcal{Q}(\theta, \theta_{k-1}) = \int \log p(y_{1:T}, x_{0:T} \mid \theta) p(x_{0:T} \mid y_{1:T}, \theta_{k-1}) dx_{0:T}$

(M) Solve $\theta_k \leftarrow \operatorname{argmax}_\theta \mathcal{Q}_k(\theta, \theta_{k-1})$

Iterate until convergence.

Note: Does not make use of the particle filter as a likelihood estimator, but uses a particle smoother (again: not in this course).

Computing $\mathcal{Q}$

Inserting

$$\log p(x_{0:T}, y_{1:T} \mid \theta) = \log\left(\prod_{t=1}^{T} p(y_t \mid x_t, \theta) \prod_{t=1}^{T} p(x_t \mid x_{t-1}, \theta) p(x_0 \mid \theta)\right)$$

$$= \sum_{t=1}^{T} \log p(y_t \mid x_t, \theta) + \sum_{t=1}^{T} \log p(x_t \mid x_{t-1}, \theta) + \log p(x_0 \mid \theta)$$

into the expression for $\mathcal{Q}(\theta, \theta_k)$ results in

$$\mathcal{Q}(\theta, \theta_k) = \int \sum_{t=1}^{T} \log p(y_t \mid x_t, \theta) p(x_t \mid y_{1:T}, \theta_k) \mathrm{d}x_t$$

$$+ \int \sum_{t=1}^{T} \log p(x_t \mid x_{t-1}, \theta) p(x_{t-1:t} \mid y_{1:T}, \theta_k) \mathrm{d}x_{t-1:t}$$

$$+ \int \log p(x_0 \mid \theta) p(x_0 \mid y_{1:T}, \theta_k) \mathrm{d}x_0.$$

Final EM algorithm

UPPSALA UNIVERSITET

Inserting particle smoothing approximations now allows for straightforward approximation of $\mathcal{Q}(\theta, \theta_k)$,

$$\widehat{\mathcal{Q}}(\theta, \theta_k) = \sum_{t=1}^{T} \sum_{i=1}^{N} \log p(y_t \,|\, x^i_{t\,|\,T}, \theta) + \sum_{t=1}^{T} \sum_{i=1}^{N} \log p(x^i_{t\,|\,T} \,|\, x^i_{t-1\,|\,T}, \theta) + \log \sum_{i=1}^{N} p(x^i_{0\,|\,T} \,|\, \theta).$$

---

1. Initialize θ_0 and run a particle smoother conditional on θ_0.
2. Use the result from previous step to compute $\widehat{\mathcal{Q}}(\theta, \theta_0)$.
3. Solve $\theta_1 = \arg\max_{\theta} \widehat{\mathcal{Q}}(\theta, \theta_0)$.
4. Run a particle smoother conditional on θ_1.
5. ....

Requires $N \to \infty$ **and** infinitely many iterations. There are more intricate solutions.

Further reading

UPPSALA UNIVERSITET

Fairly recent survey/tutorial papers:

Nikolas Kantas, Arnaud Doucet, Sumeetpal S. Singh, Jan Maciejowski and Nicolas Chopin. **On particle methods for parameter estimation in general state-space models.** *Statistical Science*, 30(3):328-351, 2015.

Thomas B. Schön, Fredrik Lindsten, Johan Dahlin, Johan Wagberg, Christian A. Naesseth, Andreas Svensson and Liang Dai. **Sequential Monte Carlo methods for system identification.** *Proceedings of the 17th IFAC Symposium on System Identification (SYSID)*, Beijing, China, October 2015.

Maximum likelihood inference using the Gaussian process:

Adrian G. Wills and Thomas B. Schön. **On the construction of probabilistic Newton-type algorithms.** *Proceedings of the 56th IEEE Conference on Decision and Control (CDC)*, Melbourne, Australia, December 2017.

Maximum likelihood inference using EM:

Andreas Lindholm and Fredrik Lindsten. **Learning dynamical systems with particle stochastic approximation EM.** *arXiv:1806.09548*, 2018.