Homework – Advanced Bayesian learning

Thomas Schön and Johan Dahlin

May 13, 2014

These problems constitute a part of the course *Advanced Bayesian Learning* given at Linköping University, Linköping, Sweden in May 2014.

The solutions to all problems should be presented with calculations as well as the code (properly documented) and plots (make sure to describe what variables are plotted and in what units) with clear cross references between these. All solutions should be properly documented and motivated (references to the literature should be clear).

Good luck!

1 **[Bootstrap particle filter for LGSS]** Consider the following scalar LGSS model

$$x_{t+1} = 0.7x_t + v_t, \qquad v_t \sim \mathcal{N}(0, 0.1), \tag{1a}$$

$$y_t = \frac{1}{2}x_t + e_t \qquad e_t \sim \mathcal{N}(0, 0.1). \tag{1b}$$

Let the initial state be distributed according to $x_1 \sim \mathcal{N}(0, 1)$.

(a) Write this model on the form

$$\mathbf{x}_{t+1} \mid (\mathbf{x}_t = x_t) \sim f(x_{t+1} \mid x_t), \tag{2a}$$

$$\mathbf{y}_t \mid (\mathbf{x}_t = x_t) \sim g(y_t \mid x_t). \tag{2b}$$

In other words, find the probability density functions $f(\cdot)$ and $g(\cdot)$ in (2) corresponding to the model (1).

(b) Simulate the model (1) to produce $T = 100$ measurements $y_{1:T}$. Based on these measurements compute the optimal estimate of $\mathbf{x}_t \mid \mathbf{y}_{1:t}$ for $t = 1, \ldots, T$. Implement a bootstrap particle filter and compare to the optimal estimates. You can for example perform this comparison by plotting the root mean square estimate (RMSE) $\varepsilon(N)$ as a function of the number of particles used in the particle filter (also plot the RMSE for the optimal estimator in the same figure). The RMSE is defined according to

$$\varepsilon(N) \triangleq \sqrt{\frac{1}{T}\sum_{t=1}^{T}\left(\widehat{x}^i_{t|t}(N) - x_t\right)^2}, \tag{3}$$

where $\widehat{x}^i_{t|t}(N)$ denotes the estimate of the state produced with a particle filter using N particles and x_t denotes the true state.

(c) **(Optional)** We are free to use more general proposal densities in the propagation step of the particle filter. Change the bootstrap particle filter to propose new particles according to

$$x^i_t \sim p(x_t \mid \tilde{x}^i_{t-1}, y_t), \tag{4}$$

rather than using $x^i_t \sim f(x_t \mid \tilde{x}^i_{t-1})$. Compare the resulting state estimates to the ones obtained in Problem (b). Does this provide better or worse estimates, explain why. *Hint: Start by deriving an explicit expression for $p(x_t \mid x_{t-1}, y_t)$ when the model is given by* (1).

2 **[Stochastic volatility]** Consider the so called stochastic volatility (SV) model

$$x_{t+1}|x_t \sim \mathcal{N}(x_{t+1}; \phi x_t, \sigma^2), \tag{5a}$$

$$y_t|x_t \sim \mathcal{N}(y_t; 0, \beta^2 \exp(x_t)), \tag{5b}$$

where the parameter vector is given by $\theta = \{\phi, \sigma, \beta\}$. Here, x_t denotes the underlying latent volatility (the variations in the asset price) and y_t denotes the observed scaled log-returns from some financial asset. The $T = 500$ observations that we consider in this task are log-returns from the NASDAQ OMX Stockholm 30 Index during a two year period between January 2, 2012 and January 2, 2014. We have calculated the log-returns by $y_t = 100[\log(s_t) - \log(s_{t-1})]$, where s_t denotes the closing price of the index at day t. The data is found in `seOMXlogreturns2012to2014.csv`. For more details on stochastic volatility models, see e.g. [2, 4].

(a) Assume that the parameter vector is given by $\theta = \{0.98, 0.16, 0.70\}$. Estimate the marginal filtering distribution and at each time index $t = 1, \ldots, T$ using the particle filter with $N = 500$ particles. Plot the mean of the distribution at each time step and compare with the observations. Is the estimated volatility reasonable?

(b) Let us now view the particle filter as a way of estimating the JSD $p(x_{1:T} \mid y_{1:T})$ (recall Algorithm 5.4). Plot the particles $\{x_{1:100}\}_{i=1}^N$ using $N = 100$ particles. How well is the JSD approximated?

(c) Compute the distribution of the error in the log-likelihood estimates by $M = 500$ Monte Carlo iterations (this can take a couple of minutes to compute). That is, compute the log-likelihood estimate on the same data M times and compute the histogram and kernel density estimate of the error given that the true log-likelihood is -695.62. What distribution does the error have? Are the estimates biased? What happens when N increases? *Hint: Make use of marginalization to compute the log-likelihood.* $p_\theta(y_{1:T}) = \int p_\theta(x_{1:T}, y_{1:T}) dx_{1:T}$. *For nonlinear state space models this results in that we can compute an estimate of the log-likelihood by making use of the particle weights. The estimator for the log-likelihood is given by*

$$\widehat{\ell}(\theta) = \log \widehat{p}_\theta(y_{1:T}) = \sum_{t=1}^{T} \log \left[\sum_{i=1}^{N} \bar{w}_t^i \right] - T \log N,$$

where $\bar{w}_t^i$ *denotes the unnormalised weight for particle* i *at time* t.

(d) Implement an importance sampler (IS) to estimate the parameter posterior distribution given by

$$p(\theta|y_{1:T}) \propto p(y_{1:T} \mid \theta) p(\theta),$$

where $p(y_{1:T} \mid \theta)$ and $p(\theta)$ denotes the likelihood function and the parameter prior, respectively. Here, we make use of an uniform prior over $[0, 1] \in \mathbb{R}$ for each of the three parameters, respectively.

Choose an appropriate proposal distribution for ϕ and calculate the estimated parameter posterior for ϕ (keeping the other parameters fixed to the values given above) using 500 samples from the log-likelihood and $N = 500$ particles. Present the result as a histogram of the posterior estimate together with a kernel density estimate and the proposal distribution.

(e) **(Optional)** Repeat the previous task for all the parameters $\{\phi, \sigma, \beta\}$ with 2 000 samples and $N = 1\,000$ particles (this can take some time, i.e. an hour to compute). Check the number of weights that are non-zero. What problems might the use of IS lead to if the number of parameters are large?

(f) **(Optional)** If you know about the Metropolis-Hastings (HW) algorithm, it is fairly simple to implement the particle marginal MH (PMMH) algorithm for parameter inference in the SV model. The main problem is that the intractable likelihood is required for computing the acceptance probability given by

$$\alpha(\theta'', \theta') = \min\left(1, \frac{p_{\theta''}(y_{1:T})}{p_{\theta'}(y_{1:T})} \frac{p(\theta'')}{p(\theta')} \frac{q(\theta'|\theta'')}{q(\theta''|\theta')}\right). \tag{6}$$

However, it is possible to show that we can replace the analytically intractable likelihood with the estimate from the particle filter and still obtain a valid MH algorithm, which gives a Markov chain that converges to the target distribution (the parameter posterior) as its stationary distribution. To implement the PMMH, you need to code the following algorithm:

- Initialise with θ_0 and compute $\widehat{p}_{\theta_0}(y_{1:T})$ using the particle filter.
- FOR $k = 1$ to K
  - Propose a new parameter from a random walk proposal by sampling $\theta' \sim \mathcal{N}(\theta_{k-1}, \epsilon^2)$.
  - Run the particle filter to obtain $\widehat{p}_{\theta'}(y_{1:T})$
  - Calculate the acceptance probability $\alpha(\theta', \theta_{k-1})$ in (6) by replacing $p_{\theta''}(y_{1:T})$ with $\widehat{p}_{\theta'}(y_{1:T})$ and $p_{\theta'}(y_{1:T})$ with $\widehat{p}_{\theta_{k-1}}(y_{1:T})$. Note that the random walk proposal is symmetric and the ratio between the proposal distributions cancels.
  - Make a standard accept/reject decision. If the parameter is accepted, set $\{\theta_k, \widehat{p}_{\theta_k}(y_{1:T})\} \leftarrow \{\theta', \widehat{p}_{\theta'}(y_{1:T})\}$, otherwise keep the old parameter and likelihood estimate.
- END FOR

In practice, we usually compute the log-likelihood and compute the log of the acceptance probability for numerical stability. This is done by computing

$$\alpha(\theta'', \theta') = \exp\left[\widehat{\ell}(\theta'') - \widehat{\ell}(\theta') + \log p(\theta'') - \log p(\theta')\right].$$

Implement the PMMH algorithm for inferring ϕ in the same setting as above, i.e. using the same number of particles and prior distributions. Select a reasonable value for ϵ and K and plot the resulting posterior estimate using a histogram and kernel density estimate.

Hint: It is numerically beneficial to work with the log-weights in the particle filter. That is compute the weight by $v_t^i = \log \mathcal{N}(y_t; 0, \beta^2 \exp(x_t^i))$ and use the following transformation to compute the corresponding particle weights: (i) compute $v_{\max,t} = \max\{v_t^i\}_{i=1}^N$ and (ii) compute $\widetilde{w}_t^i = \exp(v_t^i - v_{\max,t})$. The particle weights can now be normalised as usual. Also, make use of this procedure in the IS algorithm for parameter inference.

Background: The PMMH algorithms is one member of the Particle Markov chain Monte Carlo (PMCMC) family of algorithms introduced by [1]. See also Chapter 5 in [3] for an introduction to these algorithms.

References

[1] C. Andrieu, A. Doucet, and R. Holenstein. Particle Markov chain Monte Carlo methods. *Journal of the Royal Statistical Society. Series B (Methodological)*, 72(2):1–33, 2010.

[2] M. Chesney and L. Scott. Pricing European currency options: A comparison of the modifies Black-Scholes model and a random variance model. *Journal of Financial and Quantitative Analysis*, 24(3):267–284, 1989.

[3] F. Lindsten and T. B. Schön. Backward simulation methods for Monte Carlo statistical inference. *Foundations and Trends in Machine Learning*, 6(1):1–143, 2013.

[4] A. Melino and S.M. Turnbull. Pricing foreign currency options with stochastic volatility. *Journal of Econometrics*, 45(1–2):239–265, 1990.