MFC CDT Probability and Statistics
Week 10

O. Deniz Akyildiz

Mathematics for our Future Climate: Theory, Data and Simulation (MFC CDT).

December 9, 2024

IMPERIAL

https://akyildiz.me/

𝕏: @odakyildiz

State-space models

problem definition

$$
\begin{tikzcd}
x_0 \arrow[r] & x_1 \arrow[r] \arrow[d] & x_2 \arrow[r] \arrow[d] & \cdots \arrow[r] & x_t \arrow[d] \\
 & y_1 & y_2 & \cdots & y_t
\end{tikzcd}
$$

The conditional independence structure of a state-space model.

$(x_t)_{t\in\mathbb{N}_+}$: *hidden* signal process, $(y_t)_{t\in\mathbb{N}_+}$ the observation process.

$$
\begin{aligned}
x_0 &\sim \pi_0(\mathrm{d}x_0), && \text{(prior distribution)}\\
x_t|x_{t-1} &\sim \tau_t(\mathrm{d}x_t|x_{t-1}), && \text{(transition model)}\\
y_t|x_t &\sim g_t(y_t|x_t), && \text{(likelihood)}
\end{aligned}
$$

$x_t \in \mathrm{X}$ where X is the state-space. We use: $g_t(x_t) = g_t(y_t|x_t)$.

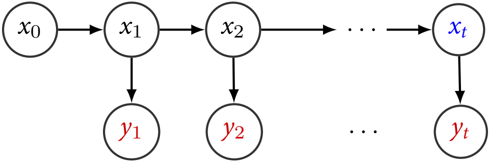
State-space models

Algorithmic principle

We are interested in estimating expectations,

$$(\varphi, \pi_t) = \int \varphi(x_t)\pi_t(x_t|y_{1:t})\mathrm{d}x_t = \int \varphi(x_t)\pi_t(\mathrm{d}x_t),$$

sequentially as new data arrives.



Algorithm:

Predict

$$\xi_t(\mathrm{d}x_t) = \int \pi_{t-1}(\mathrm{d}x_{t-1})\tau_t(\mathrm{d}x_t|x_{t-1})$$

Update

$$\pi_t(\mathrm{d}x_t) = \xi_t(\mathrm{d}x_t)\frac{g_t(y_t|x_t)}{p(y_t|y_{1:t-1})}.$$

Particle filters

Recap

A general algorithm to estimate expectations of any test function $\varphi(x_t)$ given $y_{1:t}$.

- Sampling: draw
$$\bar{x}_t^{(i)} \sim \tau_t(\mathrm{d}x_t | x_{t-1}^{(i)})$$
independently for every $i = 1, \dots, N$.
- Weighting: compute
$$w_t^{(i)} = g_t(\bar{x}_t^{(i)}) / \bar{Z}_t^N$$
for every $i = 1, \dots, N$, where $\bar{Z}_t^N = \sum_{i=1}^N g_t(\bar{x}_t^{(i)})$.
- Resampling: draw independently,
$$x_t^{(i)} \sim \tilde{\pi}_t(\mathrm{d}x) := \sum_i w_t^{(i)} \delta_{\bar{x}_t^{(i)}}(\mathrm{d}x) \quad \text{for } i = 1, \dots, N.$$

$$\pi_{t-1}^N \underbrace{\rightarrow}_{\text{sampling}} \xi_t^N \underbrace{\rightarrow}_{\text{weighting}} \tilde{\pi}_t^N \underbrace{\rightarrow}_{\text{resampling}} \pi_t^N.$$

Consider the following state-space model

$$\begin{aligned} x_0 &\sim \mathcal{N}(x_0; 0, I), \\ x_t | x_{t-1} &\sim \mathcal{N}(x_t; Ax_{t-1}, Q), \\ y_t | x_t &\sim \mathcal{N}(y_t; Hx_t, R). \end{aligned}$$

where

$$A = \begin{pmatrix} 1 & 0 & \kappa & 0 \\ 0 & 1 & 0 & \kappa \\ 0 & 0 & 0.99 & 0 \\ 0 & 0 & 0 & 0.99 \end{pmatrix} \quad \text{and} \quad Q = \begin{pmatrix} \frac{\kappa^3}{3} & 0 & \frac{\kappa^2}{2} & 0 \\ 0 & \frac{\kappa^3}{3} & 0 & \frac{\kappa^2}{2} \\ \frac{\kappa^2}{2} & 0 & \kappa & 0 \\ 0 & \frac{\kappa^2}{2} & 0 & \kappa \end{pmatrix}$$

and

$$H = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \quad \text{and} \quad R = r \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix},$$

where $r = 5$.

Particle filter for this model: Given $x_{1:t-1}^{(i)}$ for $i = 1, \ldots, N$,

- Sample: $\bar{x}_t^{(i)} \sim \mathcal{N}(x_t; Ax_{t-1}^{(i)}, Q)$,
- Compute weights:

$$\mathrm{W}_t^{(i)} = \mathcal{N}(y_t; H\bar{x}_t^{(i)}, R),$$

  Normalise: $\mathrm{w}_t^{(i)} = \mathrm{W}_t^{(i)} / \sum_{i=1}^N \mathrm{W}_t^{(i)}$
- Report

$$\pi_t^N(\mathrm{d}x_t) = \sum_{i=1}^N \mathrm{w}_t^{(i)} \delta_{\bar{x}_t^{(i)}}(\mathrm{d}x_t).$$

- Resample:

$$x_t^{(i)} \sim \sum_{i=1}^N \mathrm{w}_t^{(i)} \delta_{\bar{x}_t^{(i)}}(\mathrm{d}x_t).$$

Let us look the following Lorenz 63 model

$$
\begin{aligned}
x_{1,t} &= x_{1,t-1} - \gamma \mathrm{s}(x_{1,t} - x_{2,t}) + \sqrt{\gamma}\xi_{1,t}, \\
x_{2,t} &= x_{2,t-1} + \gamma(\mathrm{r}x_{1,t} - x_{2,t} - x_{1,t}x_{3,t}) + \sqrt{\gamma}\xi_{2,t}, \\
x_{3,t} &= x_{3,t-1} + \gamma(x_{1,t}x_{2,t} - \mathrm{b}x_{3,t}) + \sqrt{\gamma}\xi_{3,t},
\end{aligned}
$$

where $\gamma = 0.01$, r $= 28$, b $= 8/3$, s $= 10$, and $\xi_{1,t}, \xi_{2,t}, \xi_{3,t} \sim \mathcal{N}(0, 1)$ are independent Gaussian random variables. The observation model is given by

$$
y_t = [1, 0, 0]x_t + \eta_t,
$$

where $\eta_t \sim \mathcal{N}(0, \sigma_y^2)$ is a Gaussian random variable.

Another quantity BPF can estimate is the marginal likelihood:

$$p(y_{1:t}) = \int p(y_{1:t}, x_{0:t}) \mathrm{d}x_{0:t}.$$

This quantity is useful for model selection and model comparison.

Recall tbat we have tbe factorisation:

$$p(y_{1:t}) = \prod_{k=1}^{t} p(y_k|y_{1:k-1}).$$

where

$$p(y_t|y_{1:t-1}) = \int g(y_t|x_t)\xi_t(x_t|y_{1:t-1})\mathrm{d}x_t.$$

Recall that we can obtain the approximation of $\xi_t(x_t|y_{1:t-1})$ by the particle filter using predictive particles $\bar{x}_t^{(i)} \sim \tau(x_t|x_{t-1}^{(i)})$ as

$$p_t^N(\mathrm{d}x_t|y_{1:t-1}) = \frac{1}{N}\sum_{i=1}^{N} \delta_{\bar{x}_t^{(i)}}(\mathrm{d}x_t).$$

Therefore, given

$$p_t^N(\mathrm{d}x_t|y_{1:t-1}) = \frac{1}{N}\sum_{i=1}^{N}\delta_{\bar{x}_t^{(i)}}(\mathrm{d}x_t),$$

we get

$$p^N(y_t|y_{1:t-1}) = \frac{1}{N}\sum_{i=1}^{N} g(y_t|\bar{x}_t^{(i)}).$$

As a result, we can approximate

$$p^N(y_{1:t}) = \prod_{k=1}^{t} p^N(y_k|y_{1:k-1}).$$

Remarkably, this estimate is unbiased:

$$\mathbb{E}[p^N(y_{1:t})] = p(y_{1:t}).$$

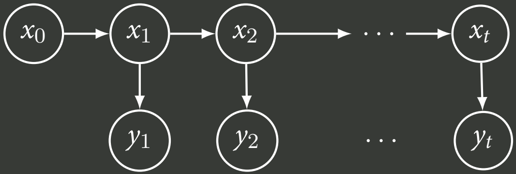
For general (bounded) test functions $\varphi(x_t)$ and filtering measures $\pi_t^N(\mathrm{d}x_t|y_{1:t})$ we have the following L_p bound

$$\|(\varphi, \pi_t^N) - (\varphi, \pi_t)\|_p \leq \frac{c_{t,p}\|\varphi\|_\infty}{\sqrt{N}}.$$

We have seen inference for

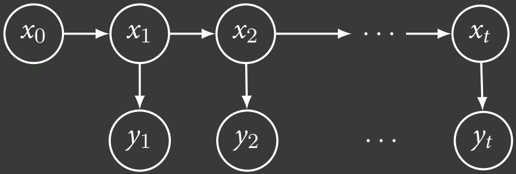
$$\begin{tikzcd}
x_0 \arrow[r] & x_1 \arrow[r] \arrow[d] & x_2 \arrow[r] \arrow[d] & \cdots \arrow[r] & x_t \arrow[d] \\
& y_1 & y_2 & \cdots & y_t
\end{tikzcd}$$

We have seen inference for

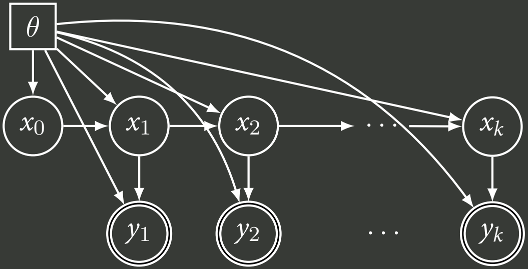


What if the model has parameters θ?

We have seen inference for

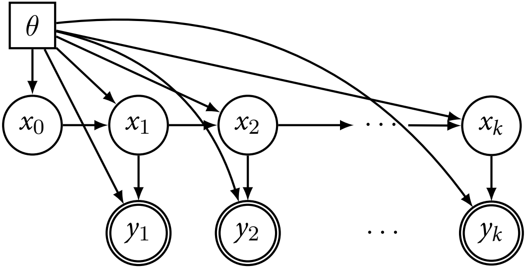


What if the model has parameters θ?



Problem definition

Recap – the model, the notation



We are given the model

$$
\begin{aligned}
x_0 &\sim \mu_\theta(x_0), \\
x_t | x_{t-1} &\sim \tau_\theta(x_t | x_{t-1}), \\
y_t | x_t &\sim g_\theta(y_t | x_t).
\end{aligned}
$$

We aim at estimating θ given $y_{1:T}$.

Problem definition

Marginal likelihood maximization

We are interested in solving the global optimization problem

$$\theta^\star = \underset{\theta \in \Theta}{\operatorname{argmax}} \log p_\theta(y_{1:T}),$$

where

$$p_\theta(y_{1:T}) = \int p_\theta(x_{0:T}, y_{1:T}) \mathrm{d}x_{0:T}.$$

In this lecture, we are interested in gradient-based approaches for maximization of $\log p_\theta(y_{1:T})$.

We have been looking at the filtering problem, i.e., estimating $\pi_t(x_t|y_{1:t})$.

We have been looking at the filtering problem, i.e., estimating $\pi_t(x_t|y_{1:t})$.

What if we want to estimate $\pi_t(x_t|y_{1:T})$ for $T > t$?

We have been looking at the filtering problem, i.e., estimating $\pi_t(x_t|y_{1:t})$.

What if we want to estimate $\pi_t(x_t|y_{1:T})$ for $T > t$?

This is called the *smoothing problem*. These methods are usually implemented backwards in time.

We have smoothing recursions

$$\begin{aligned}\pi(x_{t+1}|y_{1:t}) &= \int \tau(x_{t+1}|x_t)\pi(x_t|y_{1:t})\mathrm{d}x_t,\\ \pi(x_t|y_{1:T}) &= \pi(x_t|y_{1:t})\int \frac{\tau(x_{t+1}|x_t)\pi(x_{t+1}|y_{1:T})}{\pi(x_{t+1}|y_{1:t})}\mathrm{d}x_{t+1}.\end{aligned}$$

Proof: Let us notice

$$
\begin{aligned}
p(x_t|x_{t+1}, y_{1:T}) &= p(x_t|x_{t+1}, y_{1:t}),\\
&= \frac{p(x_t, x_{t+1}|y_{1:t})}{p(x_{t+1}|y_{1:t})},\\
&= \frac{\pi(x_t|y_{1:t})\tau(x_{t+1}|x_t)}{\pi(x_{t+1}|y_{1:t})},
\end{aligned}
$$

where the last equality follows from the Markov property.

Proof: Let us notice

$$\begin{aligned} p(x_t|x_{t+1}, y_{1:T}) &= p(x_t|x_{t+1}, y_{1:t}), \\ &= \frac{p(x_t, x_{t+1}|y_{1:t})}{p(x_{t+1}|y_{1:t})}, \\ &= \frac{\pi(x_t|y_{1:t})\tau(x_{t+1}|x_t)}{\pi(x_{t+1}|y_{1:t})}, \end{aligned}$$

where the last equality follows from the Markov property. Now we construct the joint

$$\begin{aligned} p(x_{t+1}, x_t|y_{1:T}) &= p(x_t|x_{t+1}, y_{1:T})p(x_{t+1}|y_{1:T}), \\ &= \frac{\pi(x_t|y_{1:t})\tau(x_{t+1}|x_t)}{\pi(x_{t+1}|y_{1:t})}\pi(x_{t+1}|y_{1:T}). \end{aligned}$$

By integrating out x_{t+1}, the result follows.

The smoothing problem

For the maximum-likelihood parameter estimation methods, we often require an approximation of the smoothing distribution $\pi_\theta(x_{0:T}|y_{1:T})$.

The smoothing problem

For the maximum-likelihood parameter estimation methods, we often require an approximation of the smoothing distribution $\pi_\theta(x_{0:T} | y_{1:T})$.

Wait... Can't we obtain it via the joint sampler we described in the filtering lecture?

The smoothing problem

For the maximum-likelihood parameter estimation methods, we often require an approximation of the smoothing distribution $\pi_\theta(x_{0:T}|y_{1:T})$.

Wait... Can't we obtain it via the joint sampler we described in the filtering lecture?

Yes, but...

- Sample $x_0^{(i)} \sim q(x_0)$ for $i = 1, \dots, N$.
- For $t \geq 1$
  - Sample: $\bar{x}_t^{(i)} \sim q(x_t | x_{t-1}^{(i)})$,
  - Compute weights:

$$\mathrm{W}_t^{(i)} = \frac{\tau(\bar{x}_t^{(i)} | x_{t-1}^{(i)}) g(y_t | \bar{x}_t^{(i)})}{q(\bar{x}_t^{(i)} | x_{t-1}^{(i)})}.$$

    Normalise: $\mathrm{w}_t^{(i)} = \mathrm{W}_t^{(i)} / \sum_{i=1}^N \mathrm{W}_t^{(i)}$
  - Report

$$\tilde{\pi}_t^N(\mathrm{d}x_{0:t}) = \sum_{i=1}^N \mathrm{w}_t^{(i)} \delta_{\bar{x}_{0:t}^{(i)}}(\mathrm{d}x_{0:t}).$$

  - Resample:

$$x_t^{(i)} \sim \sum_{i=1}^N \mathrm{w}_t^{(i)} \delta_{\bar{x}_t^{(i)}}(\mathrm{d}x_t).$$

Recall how we do it: For $t \geq 2$,

- Sample:

$$\bar{x}_t^{(i)} \sim q_t(x_t | x_{t-1}^{(i)}),$$

- Weight

$$w_t^{(i)} \propto \frac{\tau_\theta(\bar{x}_t^{(i)} | x_{t-1}^{(i)}) g_\theta(y_t | \bar{x}_t^{(i)})}{q_t(\bar{x}_t^{(i)} | x_{t-1}^{(i)})},$$

- Resample: Choose $a_t^{(i)}$ where $\mathbb{P}(a_t^{(i)} = j) \propto w_t^j$ and set

$$x_{1:t}^{(i)} = (x_{1:t-1}^{a_t^{(i)}}, \bar{x}_t^{a_t^{(i)}})$$

The entire state history is resampled! What can go wrong?

If we do resampling every step (which is crucial), then we can only do it if we track the genealogy backwards. (?)

- After every resample, we throw away the killed particles' ancestors and replace them with the survivors' ancestors.

Path degeneracy is a big issue.

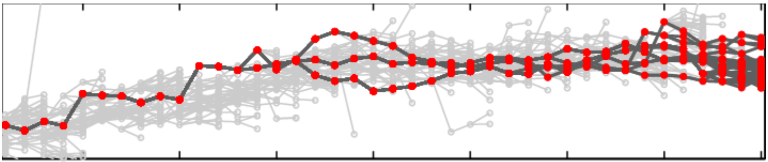


Figure: Source: Svensson, Andreas, Thomas B. Schön, and Manon Kok. "Non-linear state space smoothing using the conditional particle filter." (2015).

Instead, we can consider the following decomposition

$$\begin{aligned}\pi_\theta(x_{0:T}|y_{1:T}) &= \pi_\theta(x_T|y_{0:T}) \prod_{k=0}^{T-1} \pi_\theta(x_k|y_{0:T}, x_{k+1}), \\ &= \pi_\theta(x_T|y_{0:T}) \prod_{k=0}^{T-1} \pi_\theta(x_k|y_{0:k}, x_{k+1}). \qquad (1)\end{aligned}$$

where

$$\begin{aligned}\pi_\theta(x_t|x_{t+1}, y_{1:t}) &= \frac{\pi_\theta(x_t, x_{t+1}|y_{1:t})}{\xi_\theta(x_{t+1}|y_{1:t})}, \qquad (2)\\ &= \frac{\tau_\theta(x_{t+1}|x_t)\pi_\theta(x_t|y_{1:t})}{\xi_\theta(x_{t+1}|y_{1:t})}. \qquad (3)\end{aligned}$$

$$\pi_\theta(x_{0:T}|y_{1:T}) = \pi_\theta(x_T|y_{0:T}) \prod_{k=0}^{T-1} \pi_\theta(x_k|y_{0:k}, x_{k+1}).$$

This recursion suggests sampling $\pi_\theta(x_T|y_{1:T})$ from the filter and sample backwards from $\pi_\theta(x_k|y_{0:k}, x_{k+1})$ by conditioning on the x_{k+1}. This would provide us a sample $x_{0:T}^{(i)}$ from the smoother.

We approximate the backward distribution as

$$\pi_\theta(\mathrm{d}x_t|x_{t+1}, y_{1:t}) = \frac{\tau_\theta(x_{t+1}|x_t)\pi_\theta^N(\mathrm{d}x_t|y_{1:t})}{\xi_\theta^N(x_{t+1}|y_{1:t})}.$$

where π_θ^N and ξ_θ^N approximate filtering and predictive measures (see next slide).

$$\pi_\theta(\mathrm{d}x_t | x_{t+1}, y_{1:t}) = \frac{\tau_\theta(x_{t+1}|x_t)\pi_\theta^N(\mathrm{d}x_t|y_{1:t})}{\int \tau_\theta(x_{t+1}|x_t)\pi_\theta^N(\mathrm{d}x_t|y_{1:t})}$$

Plugging $\pi_\theta^N(\mathrm{d}x_t|y_{1:t}) = \sum_{i=1}^N \mathrm{w}_t^{(i)} \delta_{\bar{x}_t^{(i)}}(\mathrm{d}x_t)$ gives

$$\pi_\theta^N(\mathrm{d}x_t|x_{t+1}, y_{1:t}) = \frac{\sum_{i=1}^N \mathrm{w}_t^{(i)} \tau_\theta(x_{t+1}|\bar{x}_t^{(i)})\delta_{\bar{x}_t^{(i)}}(\mathrm{d}x_t)}{\sum_{i=1}^N \mathrm{w}_t^{(i)} \tau_\theta(x_{t+1}|\bar{x}_t^{(i)})} \tag{4}$$

If we use the weighted approximation then the FFBSa is given by

- At time T, sample $\tilde{x}_T \sim \pi_\theta^N(\mathrm{d}x_T | y_{1:T})$,
- t from $T-1$ to 1:
  - Compute smoothing weights

$$\mathrm{w}_{t+1|t}^{(i)} \propto \mathrm{w}_t^{(i)} \tau_\theta(\tilde{x}_{t+1} | \bar{x}_t^{(i)}).$$

  - Then sample

$$\tilde{x}_t \sim \sum_{i=1}^{N} \mathrm{w}_{t+1|t}^{(i)} \delta_{\bar{x}_t^{(i)}}(\mathrm{d}x_t).$$

The sample $\tilde{x}_{0:T}$ is a sample from the smoother. However, it is just a single sample!

Do the same N times. Reduces path degeneracy, but $\mathcal{O}(N^2(T+1))$.

Recall the original smoothing recursions we discussed:

$$\begin{aligned}
\pi_\theta(x_t|y_{1:T}) &= \int \pi_\theta(x_t, x_{t+1}|y_{1:T})\mathrm{d}x_{t+1}, \\
&= \int \pi_\theta(x_t|x_{t+1}, y_{1:t})\pi_\theta(x_{t+1}|y_{1:T})\mathrm{d}x_{t+1}, \\
&= \int \frac{\tau_\theta(x_{t+1}|x_t)\pi_\theta(x_t|y_{1:t})}{\xi_\theta(x_{t+1}|y_{1:t})}\pi_\theta(x_{t+1}|y_{1:T})\mathrm{d}x_{t+1}.
\end{aligned}$$

Recall the original smoothing recursions we discussed:

$$\begin{aligned}
\pi_\theta(x_t|y_{1:T}) &= \int \pi_\theta(x_t, x_{t+1}|y_{1:T})\mathrm{d}x_{t+1}, \\
&= \int \pi_\theta(x_t|x_{t+1}, y_{1:t})\pi_\theta(x_{t+1}|y_{1:T})\mathrm{d}x_{t+1}, \\
&= \int \frac{\tau_\theta(x_{t+1}|x_t)\pi_\theta(x_t|y_{1:t})}{\xi_\theta(x_{t+1}|y_{1:t})}\pi_\theta(x_{t+1}|y_{1:T})\mathrm{d}x_{t+1}.
\end{aligned}$$

Can we use these to build a particle approximation?

Recall the original smoothing recursions we discussed:

$$\begin{aligned}\pi_\theta(x_t|y_{1:T}) &= \int \pi_\theta(x_t, x_{t+1}|y_{1:T})\mathrm{d}x_{t+1},\\ &= \int \pi_\theta(x_t|x_{t+1}, y_{1:t})\pi_\theta(x_{t+1}|y_{1:T})\mathrm{d}x_{t+1},\\ &= \int \frac{\tau_\theta(x_{t+1}|x_t)\pi_\theta(x_t|y_{1:t})}{\xi_\theta(x_{t+1}|y_{1:t})}\pi_\theta(x_{t+1}|y_{1:T})\mathrm{d}x_{t+1}.\end{aligned}$$

Can we use these to build a particle approximation? Recall measure theoretic form

$$\pi_\theta(\mathrm{d}x_t|y_{1:T}) = \pi_\theta(\mathrm{d}x_t|y_{1:t}) \int \frac{\tau_\theta(x_{t+1}|x_t)}{\xi_\theta(x_{t+1}|y_{1:t})}\pi_\theta(x_{t+1}|y_{1:T})\mathrm{d}x_{t+1}.$$

Backward recursion

$$\pi_\theta(\mathrm{d}x_t|y_{1:T}) = \pi_\theta(\mathrm{d}x_t|y_{1:t}) \int \frac{\tau_\theta(x_{t+1}|x_t)}{\int \tau_\theta(x_{t+1}|x_t)\pi_\theta(\mathrm{d}x_t|y_{1:t})} \pi_\theta(\mathrm{d}x_{t+1}|y_{1:T}).$$

Backward recursion

$$\pi_\theta(\mathrm{d}x_t|y_{1:T}) = \pi_\theta(\mathrm{d}x_t|y_{1:t}) \int \frac{\tau_\theta(x_{t+1}|x_t)}{\int \tau_\theta(x_{t+1}|x_t)\pi_\theta(\mathrm{d}x_t|y_{1:t})} \pi_\theta(\mathrm{d}x_{t+1}|y_{1:T}).$$

This means that we can use approximations $\{\pi_\theta^N(\mathrm{d}x_t|y_{1:t})\}_{t=1}^T$ again to recursively update the smoother backwards in time and construct the smoother update

$$\pi_\theta(\mathrm{d}x_{t+1}|y_{1:T}) \mapsto \pi_\theta(\mathrm{d}x_t|y_{1:T}).$$

Assume we have an approximation

$$\pi_\theta^N(\mathrm{d}x_{t+1}|y_{1:T}) = \sum_{i=1}^{N} \mathrm{w}_{t+1|T}^{(i)} \delta_{\bar{x}_{t+1}^{(i)}}(\mathrm{d}x_{t+1}).$$

where $\mathrm{w}_{T|T}^{(i)} = \mathrm{w}_T^{(i)}$. We can use the recursion in the previous slide to obtain

$$\pi_\theta(\mathrm{d}x_t|y_{1:T}) = \sum_{i=1}^{N} \mathrm{w}_{t|T}^{(i)} \delta_{\bar{x}_t^{(i)}}(\mathrm{d}x_t),$$

where

$$\mathrm{w}_{t|T}^{(i)} = \mathrm{w}_t^{(i)} \sum_{j=1}^{N} \frac{\mathrm{w}_{t+1|T}^{(j)} \tau_\theta(\bar{x}_{t+1}^{(j)}|\bar{x}_t^{(i)})}{\sum_{l=1}^{N} \mathrm{w}_t^{(l)} \tau_\theta(\bar{x}_{t+1}^{(j)}|\bar{x}_t^{(l)})}$$

Recall we are interested in solving the global optimization problem

$$\theta^\star = \underset{\theta \in \Theta}{\operatorname{argmax}} \log p_\theta(y_{1:T}),$$

where

$$p_\theta(y_{1:T}) = \int p_\theta(x_{0:T}, y_{1:T}) \mathrm{d}x_{0:T}.$$

A generic way to do this would be to run

$$\theta_{i+1} = \theta_i + \gamma \nabla \log p_\theta(y_{1:T}).$$

- Well understood gradient scheme,
- Can be also replaced by an adaptive gradient scheme. (Adam, your favourite one...)

However, the gradient is not computable...

For this maximization, we will be interested in computing

$$\nabla_\theta \log p_\theta(y_{1:T}).$$

For this, we use Fisher's identity.

The parameter estimation problem

How to compute the gradient?

Proposition 1 (Fisher's identity)

Under appropriate regularity conditions, we have

$$\nabla_\theta \log p_\theta(y_{1:T}) = \int \nabla_\theta \log p_\theta(x_{0:T}, y_{1:T}) p_\theta(x_{0:T} | y_{1:T}) dx_{0:T}.$$

The parameter estimation problem

How to compute the gradient?

Proof.

Let us note that

$$
\begin{aligned}
\nabla_\theta \log p_\theta(y_{1:T}) &= \frac{\nabla_\theta p_\theta(y_{1:T})}{p_\theta(y_{1:T})}, \\
&= \frac{\nabla \int p_\theta(x_{0:T}, y_{1:T}) \mathrm{d}x_{0:T}}{p_\theta(y_{1:T})}, \\
&= \int \frac{\nabla p_\theta(x_{0:T}, y_{1:T})}{p_\theta(y_{1:T})} \mathrm{d}x_{0:T}, \\
&= \int \frac{\nabla \log p_\theta(x_{0:T}, y_{1:T}) p_\theta(x_{0:T}, y_{1:T})}{p_\theta(y_{1:T})} \mathrm{d}x_{0:T}, \\
&= \int \nabla \log p_\theta(x_{0:T}, y_{1:T}) \pi_\theta(x_{0:T} | y_{1:T}) \mathrm{d}x_{0:T}.
\end{aligned}
$$

■

Given Fisher's identity,

$$\nabla_\theta \log p_\theta(y_{1:T}) = \int \nabla_\theta \log p_\theta(x_{0:T}, y_{1:T}) \pi_\theta(x_{0:T}|y_{1:T}) \mathrm{d}x_{0:T}.$$

and

$$\log p_\theta(x_{0:T}, y_{1:T}) = \log \mu_\theta(x_0) + \sum_{t=1}^{T} \log \tau_\theta(x_t|x_{t-1}) + \sum_{t=1}^{T} \log g_\theta(y_t|x_t),$$

Given

$$\log p_\theta(x_{0:T}, y_{1:T}) = \log \mu_\theta(x_0) + \sum_{t=1}^{T} \log \tau_\theta(x_t|x_{t-1}) + \sum_{t=1}^{T} \log g_\theta(y_t|x_t),$$

Some shortcut notation:

$$\begin{aligned} s_1^\theta(x_{-1}, x_0) &= s_0^\theta(x_0) = \nabla \log \mu_\theta(x_0), \\ s_{\theta,t}(x_{t-1}, x_t) &= \nabla \log g_\theta(y_t|x_t) + \nabla \log \tau_\theta(x_t|x_{t-1}). \end{aligned}$$

So finally the gradient can be written as an expectation

$$\nabla_\theta \log p_\theta(y_{1:T}) = \int \nabla_\theta \log p_\theta(x_{0:T}, y_{1:T}) p_\theta(x_{0:T}|y_{1:T}) \mathrm{d}x_{0:T}.$$

We identify the marginal likelihood as an additive functional

$$\begin{aligned}\nabla_\theta \log p_\theta(y_{1:T}) &= S_T^\theta(x_{1:T}), \\ &= \int_{\mathsf{X}^{T+1}} \left(\sum_{t=1}^{T} s_t^\theta(x_{t-1}, x_t) \right) \pi_\theta(x_{0:T}|y_{1:T}) \mathrm{d}x_{0:T}.\end{aligned}$$

But how do we compute? Recall

$$s_t^\theta(x_{t-1}, x_t) = \nabla \log g_\theta(y_t|x_t) + \nabla \log \tau_\theta(x_t|x_{t-1}).$$

The BPF with parameter gradient computation. Fix θ and assume $\{X_{1:t-1}^{(i)}, \alpha_{t-1}^{(i)}$
are given.

- Sample: $\bar{x}_t^{(i)} \sim \tau_\theta(x_t|x_{t-1}^{(i)})$.
- Weight $\mathrm{w}_t^{(i)} \propto g(y_t|\bar{x}_t^{(i)})$.
- Resample:

$$x_t^{(i)} \sim \sum_{i=1}^N \mathrm{w}_t^{(i)} \delta_{\bar{x}_t^{(i)}}(\mathrm{d}x_t),$$

i.e. $x_t^{(i)} = \bar{x}_t^{a_t^{(i)}}$ with $\mathbb{P}(a_t^{(i)} = j) = \mathrm{w}_t^j$ and construct the estimate

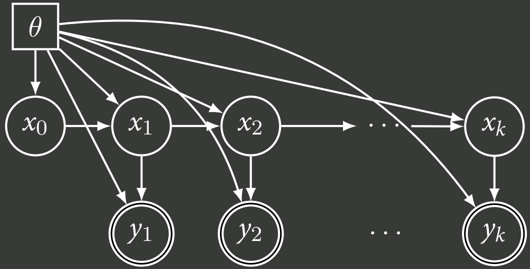
$$\alpha_t^{(i)} = \alpha_{t-1}^{a_t^{(i)}} + s_t^\theta(x_{t-1}^{a_t^{(i)}}, x_t^{(i)})$$

Then

$$S_T^{\theta,N} = \frac{1}{N}\sum_{i=1}^{N}\alpha_T^{(i)}$$

However, as this naive "forward smoother" $\mathcal{O}(N)$ iteration complexity) suffers from path degeneracy as we discussed before, therefore the estimates will not be reliable.

Use FFBS described before however the computation won't be recursive (it is offline) and $\mathcal{O}(N^2)$ complexity - but has better properties.



We are given the model

$$
\begin{aligned}
x_0 &\sim \mu_\theta(x_0), \\
x_t | x_{t-1} &\sim \tau_\theta(x_t | x_{t-1}), \\
y_t | x_t &\sim g_\theta(y_t | x_t).
\end{aligned}
$$

We looked at estimating θ given $y_{1:T}$.

- We have seen maximum likelihood approaches

$$\theta^\star \in \operatorname*{argmax}_{\theta \in \Theta} \log p(y_{1:T}|\theta).$$

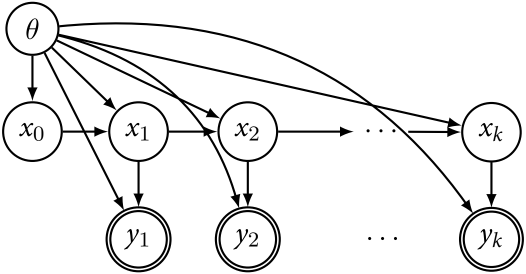
where

$$p(y_{1:T}|\theta) = \int p(y_{1:T}, x_{0:T}|\theta) \mathrm{d}x_{0:T}.$$

We will now look at the Bayesian approach to this problem.

Problem definition

Recap – the model, the notation



We are given the model

$$\begin{aligned} \theta &\sim p(\theta), \\ x_0 &\sim \mu_\theta(x_0), \\ x_t|x_{t-1} &\sim \tau(x_t|x_{t-1}, \theta), \\ y_t|x_t &\sim g(y_t|x_t, \theta). \end{aligned}$$

We aim at sampling from $p(\theta|y_{1:T})$.

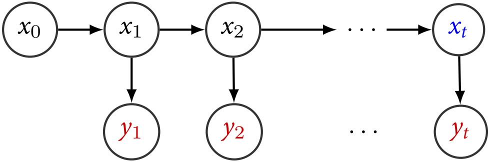
State-space models

Recall: Algorithmic principle

We are interested in estimating expectations,

$$(\varphi, \pi_t) = \int \varphi(x_t)\pi_t(\color{blue}{x_t}\color{black}|\color{red}{y_{1:t}}\color{black})\mathrm{d}x_t = \int \varphi(x_t)\pi_t(\mathrm{d}x_t),$$

sequentially as new data arrives.



Algorithm:

Predict

$$\xi_t(\mathrm{d}x_t) = \int \pi_{t-1}(\mathrm{d}x_{t-1})\tau_t(\mathrm{d}x_t|x_{t-1})$$

Update

$$\pi_t(\mathrm{d}x_t) = \xi_t(\mathrm{d}x_t)\frac{g_t(y_t|x_t)}{p(y_t|y_{1:t-1})}.$$

A general algorithm to estimate expectations of any test function $\varphi(x_t)$ given $y_{1:t}$.

- Sampling: draw
$$\bar{x}_t^{(i)} \sim \tau_\theta(\mathrm{d}x_t | x_{t-1}^{(i)})$$
independently for every $i = 1, \ldots, N$.
- Weighting: compute
$$w_t^{(i)} = g_\theta(\bar{x}_t^{(i)}) / \bar{Z}_t^N$$
for every $i = 1, \ldots, N$, where $\bar{Z}_t^N = \sum_{i=1}^N g_\theta(\bar{x}_t^{(i)})$.
- Resampling: draw independently,
$$x_t^{(i)} \sim \tilde{\pi}_t(\mathrm{d}x) := \sum_i w_t^{(i)} \delta_{\bar{x}_t^{(i)}}(\mathrm{d}x) \quad \text{for } i = 1, \ldots, N.$$

$$\pi_{t-1}^N \underbrace{\rightarrow}_{\text{sampling}} \xi_t^N \underbrace{\rightarrow}_{\text{weighting}} \tilde{\pi}_t^N \underbrace{\rightarrow}_{\text{resampling}} \pi_t^N.$$

Another quantity BPF can estimate is the marginal likelihood:

$$p(y_{1:t}|\theta) = \int p(y_{1:t}, x_{0:t}|\theta)\mathrm{d}x_{0:t}.$$

This quantity is useful for model selection and model comparison.

Recall that we have tbe factorisation:

$$p(y_{1:t}|\theta) = \prod_{k=1}^{t} p(y_k|y_{1:k-1}, \theta).$$

where

$$p(y_t|y_{1:t-1}, \theta) = \int g(y_t|x_t, \theta)\xi(x_t|y_{1:t-1}, \theta)\mathrm{d}x_t.$$

Recall that we can obtain the approximation of $\xi(x_t|y_{1:t-1}, \theta)$ by the particle filter using predictive particles $\bar{x}_t^{(i)} \sim \tau(x_t|x_{t-1}^{(i)}, \theta)$ as

$$p^N(\mathrm{d}x_t|y_{1:t-1}, \theta) = \frac{1}{N}\sum_{i=1}^{N} \delta_{\bar{x}_t^{(i)}}(\mathrm{d}x_t).$$

Therefore, given

$$p_\theta^N(\mathrm{d}x_t|y_{1:t-1},\theta) = \frac{1}{N}\sum_{i=1}^{N}\delta_{\bar{x}_t^{(i)}}(\mathrm{d}x_t),$$

we get

$$p^N(y_t|y_{1:t-1},\theta) = \frac{1}{N}\sum_{i=1}^{N} g(y_t|\bar{x}_t^{(i)},\theta).$$

As a result, we can approximate

$$p^N(y_{1:t}|\theta) = \prod_{k=1}^{t} p^N(y_k|y_{1:k-1},\theta).$$

Remarkably, this estimate is unbiased:

$$\mathbb{E}[p^N(y_{1:t}|\theta)] = p(y_{1:t}|\theta),$$

for every fixed θ.

Let us assume that we would like to sample from $p(\theta|y_{1:t})$

Let us assume that we would like to sample from $p(\theta|y_{1:t})$

- We would normally use the factorisation

$$p(\theta|y_{1:t}) \propto p(y_{1:t}|\theta)p(\theta).$$

Let us assume that we would like to sample from $p(\theta|y_{1:t})$

- We would normally use the factorisation

$$p(\theta|y_{1:t}) \propto p(y_{1:t}|\theta)p(\theta).$$

- Based on this, we could design a Metropolis-Hastings algorithm (with any proposal).

Let us assume that we would like to sample from $p(\theta|y_{1:t})$

- We would normally use the factorisation

$$p(\theta|y_{1:t}) \propto p(y_{1:t}|\theta)p(\theta).$$

- Based on this, we could design a Metropolis-Hastings algorithm (with any proposal).

Recall the Metropolis-Hastings algorithm for this case:

- Given $\theta^{(i)}$, sample $\theta' \sim q(\theta'|\theta^{(i)})$.

Recall the Metropolis-Hastings algorithm for this case:

- Given $\theta^{(i)}$, sample $\theta' \sim q(\theta'|\theta^{(i)})$.
- Compute the acceptance ratio

$$r(\theta^{(i)}, \theta') = \frac{p(y_{1:t}|\theta')p(\theta')q(\theta^{(i)}|\theta')}{p(y_{1:t}|\theta^{(i)})p(\theta^{(i)})q(\theta'|\theta^{(i)})}.$$

Recall the Metropolis-Hastings algorithm for this case:

- Given $\theta^{(i)}$, sample $\theta' \sim q(\theta'|\theta^{(i)})$.
- Compute the acceptance ratio

$$r(\theta^{(i)}, \theta') = \frac{p(y_{1:t}|\theta')p(\theta')q(\theta^{(i)}|\theta')}{p(y_{1:t}|\theta^{(i)})p(\theta^{(i)})q(\theta'|\theta^{(i)})}.$$

- Accept θ' with probability $\min\{1, r(\theta^{(i)}, \theta')\}$ and set $\theta^{(i+1)} = \theta'$.

Recall the Metropolis-Hastings algorithm for this case:

- Given $\theta^{(i)}$, sample $\theta' \sim q(\theta'|\theta^{(i)})$.
- Compute the acceptance ratio

$$r(\theta^{(i)}, \theta') = \frac{p(y_{1:t}|\theta')p(\theta')q(\theta^{(i)}|\theta')}{p(y_{1:t}|\theta^{(i)})p(\theta^{(i)})q(\theta'|\theta^{(i)})}.$$

- Accept θ' with probability $\min\{1, r(\theta^{(i)}, \theta')\}$ and set $\theta^{(i+1)} = \theta'$.
- Otherwise, set $\theta^{(i+1)} = \theta^{(i)}$.

Recall the Metropolis-Hastings algorithm for this case:

- Given $\theta^{(i)}$, sample $\theta' \sim q(\theta'|\theta^{(i)})$.
- Compute the acceptance ratio

$$r(\theta^{(i)}, \theta') = \frac{p(y_{1:t}|\theta')p(\theta')q(\theta^{(i)}|\theta')}{p(y_{1:t}|\theta^{(i)})p(\theta^{(i)})q(\theta'|\theta^{(i)})}.$$

- Accept θ' with probability $\min\{1, r(\theta^{(i)}, \theta')\}$ and set $\theta^{(i+1)} = \theta'$.
- Otherwise, set $\theta^{(i+1)} = \theta^{(i)}$.

Can this be applicable for state-space models?

The issue:

- We do not know $p(y_{1:t}|\theta)$ as this is an integral over $x_{0:t}$:

$$p(y_{1:t}|\theta) = \int p(y_{1:t}, x_{0:t}|\theta)\mathrm{d}x_{0:t}.$$

The issue:

- We do not know $p(y_{1:t}|\theta)$ as this is an integral over $x_{0:t}$:

$$p(y_{1:t}|\theta) = \int p(y_{1:t}, x_{0:t}|\theta)\mathrm{d}x_{0:t}.$$

- We can approximate this integral using the particle filter:

$$p^N(y_{1:t}|\theta) = \frac{1}{N}\sum_{i=1}^{N} g(y_t|\bar{x}_t^{(i)}, \theta).$$

The issue:

- We do not know $p(y_{1:t}|\theta)$ as this is an integral over $x_{0:t}$:

$$p(y_{1:t}|\theta) = \int p(y_{1:t}, x_{0:t}|\theta)\mathrm{d}x_{0:t}.$$

- We can approximate this integral using the particle filter:

$$p^N(y_{1:t}|\theta) = \frac{1}{N}\sum_{i=1}^{N} g(y_t|\bar{x}_t^{(i)}, \theta).$$

- Remarkably, plugging in unbiased estimates in Metropolis-Hastings ratios preserves the stationary measure (Andrieu et al., 2010).

Recall the Metropolis-Hastings algorithm for this case:

- Given $\theta^{(i)}$, sample $\theta' \sim q(\theta'|\theta^{(i)})$.

Recall the Metropolis-Hastings algorithm for this case:

- Given $\theta^{(i)}$, sample $\theta' \sim q(\theta'|\theta^{(i)})$.
- Compute the acceptance ratio

$$r(\theta^{(i)}, \theta') = \frac{p^N(y_{1:t}|\theta')p(\theta')q(\theta^{(i)}|\theta')}{p^N(y_{1:t}|\theta^{(i)})p(\theta^{(i)})q(\theta'|\theta^{(i)})}.$$

Recall the Metropolis-Hastings algorithm for this case:

- Given $\theta^{(i)}$, sample $\theta' \sim q(\theta'|\theta^{(i)})$.
- Compute the acceptance ratio

$$r(\theta^{(i)}, \theta') = \frac{p^N(y_{1:t}|\theta')p(\theta')q(\theta^{(i)}|\theta')}{p^N(y_{1:t}|\theta^{(i)})p(\theta^{(i)})q(\theta'|\theta^{(i)})}.$$

- Accept θ' with probability $\min\{1, r(\theta^{(i)}, \theta')\}$ and set $\theta^{(i+1)} = \theta'$.

Recall the Metropolis-Hastings algorithm for this case:

- Given $\theta^{(i)}$, sample $\theta' \sim q(\theta'|\theta^{(i)})$.
- Compute the acceptance ratio

$$r(\theta^{(i)}, \theta') = \frac{p^N(y_{1:t}|\theta')p(\theta')q(\theta^{(i)}|\theta')}{p^N(y_{1:t}|\theta^{(i)})p(\theta^{(i)})q(\theta'|\theta^{(i)})}.$$

- Accept θ' with probability $\min\{1, r(\theta^{(i)}, \theta')\}$ and set $\theta^{(i+1)} = \theta'$.
- Otherwise, set $\theta^{(i+1)} = \theta^{(i)}$.

Parameter inference

particle Metropolis-Hastings

Recall the Metropolis-Hastings algorithm for this case:

- Given $\theta^{(i)}$, sample $\theta' \sim q(\theta'|\theta^{(i)})$.
- Compute the acceptance ratio

$$r(\theta^{(i)}, \theta') = \frac{p^N(y_{1:t}|\theta')p(\theta')q(\theta^{(i)}|\theta')}{p^N(y_{1:t}|\theta^{(i)})p(\theta^{(i)})q(\theta'|\theta^{(i)})}.$$

- Accept θ' with probability $\min\{1, r(\theta^{(i)}, \theta')\}$ and set $\theta^{(i+1)} = \theta'$.
- Otherwise, set $\theta^{(i+1)} = \theta^{(i)}$.

This is called the particle Metropolis-Hastings algorithm.

A few drawbacks of this approach:

- The algorithm is not very efficient as it requires a large number of particles to obtain a good approximation of $p(y_{1:t}|\theta)$.

A few drawbacks of this approach:

- The algorithm is not very efficient as it requires a large number of particles to obtain a good approximation of $p(y_{1:t}|\theta)$.
- Also, for every parameter sample $\theta^{(i)}$, a fresh run of the particle filter is required.

A few drawbacks of this approach:

- The algorithm is not very efficient as it requires a large number of particles to obtain a good approximation of $p(y_{1:t}|\theta)$.
- Also, for every parameter sample $\theta^{(i)}$, a fresh run of the particle filter is required.

We will now look at a completely online approach.

Let us discuss a meta-sampler that can be used to sample from $p(\theta|y_{1:t})$. First, let us try to use a naive importance sampler to sample from $p(\theta|y_{1:t})$ (forget for now about latents $x_{1:t}$).

How to develop an importance sampler for evolving $p(\theta|y_{1:t})$?

Let us recall the recursions:

$$p(\theta|y_{1:t}) = \frac{p(y_t|\theta)p(\theta|y_{1:t-1})}{p(y_t|y_{1:t-1})}.$$

Let us recall the recursions:

$$p(\theta|y_{1:t}) = \frac{p(y_t|\theta)p(\theta|y_{1:t-1})}{p(y_t|y_{1:t-1})}.$$

With these recursions in mind, we can indeed naively try to develop an importance sampler.

Let us choose a proposal: $q(\theta)$ and then perform importance sampling:

- Sample $\theta^{(i)} \sim q(\theta)$ for $i = 1, \ldots, N$.

Let us choose a proposal: $q(\theta)$ and then perform importance sampling:

- Sample $\theta^{(i)} \sim q(\theta)$ for $i = 1, \ldots, N$.
- Compute the importance weights:

$$\mathrm{W}_t^{(i)} = \frac{p(y_{1:t}|\theta^{(i)})p(\theta^{(i)})}{q(\theta^{(i)})}.$$

Let us choose a proposal: $q(\theta)$ and then perform importance sampling:

- Sample $\theta^{(i)} \sim q(\theta)$ for $i = 1, \ldots, N$.
- Compute the importance weights:

$$\mathrm{W}_t^{(i)} = \frac{p(y_{1:t}|\theta^{(i)})p(\theta^{(i)})}{q(\theta^{(i)})}.$$

- Normalise the weights:

$$\mathrm{w}_t^{(i)} = \frac{\mathrm{W}_t^{(i)}}{\sum_{j=1}^{N} \mathrm{W}_t^{(j)}}.$$

Let us choose a proposal: $q(\theta)$ and then perform importance sampling:

- Sample $\theta^{(i)} \sim q(\theta)$ for $i = 1, \ldots, N$.
- Compute the importance weights:

$$\mathrm{W}_t^{(i)} = \frac{p(y_{1:t}|\theta^{(i)})p(\theta^{(i)})}{q(\theta^{(i)})}.$$

- Normalise the weights:

$$\mathrm{w}_t^{(i)} = \frac{\mathrm{W}_t^{(i)}}{\sum_{j=1}^{N} \mathrm{W}_t^{(j)}}.$$

Can we get a sequential structure in weights as in the particle filter case?

We have

$$W_{0:t}(\theta) = \frac{p(y_{1:t}|\theta)p(\theta)}{q(\theta)}.$$

We have

$$W_{0:t}(\theta) = \frac{p(y_{1:t}|\theta)p(\theta)}{q(\theta)}.$$

Unlike the particle filter case, we do not have a sequential structure in the weights. One can try

$$W_{0:t}(\theta) = p(y_t|y_{1:t-1}, \theta)W_{0:t-1}(\theta).$$

This means that we have to unroll it back to time zero:

$$W_{0:t}(\theta) = p(y_t|y_{1:t-1}, \theta)p(y_{t-1}|y_{1:t-2}, \theta) \cdots \frac{p(\theta)}{q(\theta)}.$$

Given

$$W_{0:t}(\theta) = p(y_t|y_{1:t-1},\theta)p(y_{t-1}|y_{1:t-2},\theta)\cdots\frac{p(\theta)}{q(\theta)}.$$

the practical weight computation would be:

$$\mathrm{W}_0^{(i)} = \frac{p(\theta^{(i)})}{q(\theta^{(i)})},$$

and

$$\mathrm{W}_t^{(i)} = p(y_t|y_{1:t-1},\theta^{(i)})\mathrm{W}_{t-1}^{(i)}.$$

This would cause multiple issues:

- The algorithm is essentially putting samples into the space and just recomputing weights.

Parameter inference

Nested particle filter

This would cause multiple issues:

- The algorithm is essentially putting samples into the space and just recomputing weights.
  - Samples do not move!

This would cause multiple issues:

- The algorithm is essentially putting samples into the space and just recomputing weights.
  - Samples do not move!
- Even if we introduce resampling at every stage, then still have the same problem.

Parameter inference

Nested particle filter

This would cause multiple issues:

- The algorithm is essentially putting samples into the space and just recomputing weights.
  - Samples do not move!
- Even if we introduce resampling at every stage, then still have the same problem.
  - Samples do not move + are resampled.

This would cause multiple issues:

- The algorithm is essentially putting samples into the space and just recomputing weights.
  - Samples do not move!
- Even if we introduce resampling at every stage, then still have the same problem.
  - Samples do not move + are resampled.
  - Only one sample will survive.

This would cause multiple issues:

- The algorithm is essentially putting samples into the space and just recomputing weights.
  - Samples do not move!
- Even if we introduce resampling at every stage, then still have the same problem.
  - Samples do not move + are resampled.
  - Only one sample will survive.
- We need to introduce a new mechanism to move the samples around.

We need a way to *shake* the particles, without introducing too much error.

- Use a jittering kernel (Crisan, Míguez, et al., 2014):

$$\kappa(\mathrm{d}\theta|\theta') = (1-\epsilon_N)\delta_{\theta'}(\mathrm{d}\theta) + \epsilon_N \tau(\mathrm{d}\theta|\theta'), \tag{5}$$

  to sample new particles $\theta_t^{(i)} \sim \kappa(\cdot|\theta_{t-1}^{(i)})$.
- We usually choose $\epsilon_N \leq \frac{1}{\sqrt{N}}$.
- τ can be simple, i.e., multivariate Gaussian or multivariate t distribution.

Parameter inference

Nested particle filter

The jittered sampler:

- Sample $\bar{\theta}_t^{(i)} \sim \kappa(\cdot|\theta_{t-1}^{(i)})$ for $i = 1, \ldots, N$.

The jittered sampler:

- Sample $\bar{\theta}_t^{(i)} \sim \kappa(\cdot|\theta_{t-1}^{(i)})$ for $i = 1, \ldots, N$.
- Compute the importance weights:

$$W_t^{(i)} = p(y_t|y_{1:t-1}, \bar{\theta}_t^{(i)}),$$

The jittered sampler:

- Sample $\bar{\theta}_t^{(i)} \sim \kappa(\cdot|\theta_{t-1}^{(i)})$ for $i = 1, \ldots, N$.
- Compute the importance weights:

$$\mathrm{W}_t^{(i)} = p(y_t|y_{1:t-1}, \bar{\theta}_t^{(i)}),$$

- Normalise the weights:

$$\mathrm{w}_t^{(i)} = \frac{\mathrm{W}_t^{(i)}}{\sum_{j=1}^N \mathrm{W}_t^{(j)}}.$$

- Resample:

$$\theta_t^{(i)} \sim \sum_{j=1}^N \mathrm{w}_t^{(j)} \delta_{\bar{\theta}_t^{(j)}}(\mathrm{d}\theta).$$

As you could guess, "compute the importance weights" step should be done using a particle filter.

- Sample $\bar{\theta}_t^{(i)} \sim \kappa(\cdot|\theta_{t-1}^{(i)})$ for $i = 1, \ldots, N$.

As you could guess, "compute the importance weights" step should be done using a particle filter.

- Sample $\bar{\theta}_t^{(i)} \sim \kappa(\cdot|\theta_{t-1}^{(i)})$ for $i = 1, \ldots, N$.
- Compute the importance weights:

$$\mathrm{W}_t^{(i)} = p^M(y_t|y_{1:t-1}, \bar{\theta}_t^{(i)}),$$

  using a particle filter with M particles.

As you could guess, "compute the importance weights" step should be done using a particle filter.

- Sample $\bar{\theta}_t^{(i)} \sim \kappa(\cdot|\theta_{t-1}^{(i)})$ for $i = 1, \dots, N$.
- Compute the importance weights:

$$\mathrm{W}_t^{(i)} = p^M(y_t | y_{1:t-1}, \bar{\theta}_t^{(i)}),$$

  using a particle filter with M particles.
- Normalise the weights:

$$\mathrm{w}_t^{(i)} = \frac{\mathrm{W}_t^{(i)}}{\sum_{j=1}^N \mathrm{W}_t^{(j)}}.$$

- Resample:

$$\theta_t^{(i)} \sim \sum_{j=1}^N \mathrm{w}_t^{(j)} \delta_{\bar{\theta}_t^{(j)}}(\mathrm{d}\theta).$$

This algorithm is purely online.

Both approaches (pMCMC and nested PF) rely on unbiased marginal likelihoods.

Both approaches (pMCMC and nested PF) rely on unbiased marginal likelihoods.

Therefore, the unbiasedness property of PFs are crucial.

References I

Andrieu, Christophe, Arnaud Doucet, and Roman Holenstein (2010). "Particle Markov chain Monte Carlo methods". In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 72.3, pp. 269–342.

Crisan, Dan, Joaquín Míguez, et al. (2014). "Particle-kernel estimation of the filter density in state-space models". In: *Bernoulli* 20.4, pp. 1879–1929.