Introduction to Bayesian Filtering: Theory & Methods

O. Deniz Akyildiz

Department of Mathematics, Imperial College London

May 06, 2024

IMPERIAL

Summer School on Bayesian filtering (SSBF 2024)

https://akyildiz.me/ssbf-2024-intro

State-Space Models and Stochastic Filtering

The Kalman Filter

Monte Carlo methods - an introduction

Particle filters

Smoothing

Background

State-space models

problem definition

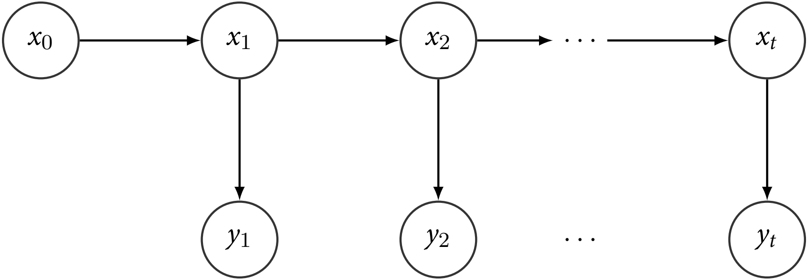


Figure: The conditional independence structure of a state-space model.

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

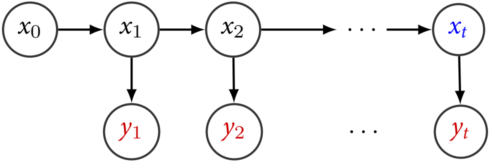
$$
\begin{aligned}
x_0 &\sim \pi_0(\mathrm{d}x_0), && \text{(prior distribution)}\\
x_t|x_{t-1} &\sim \tau_t(\mathrm{d}x_t|x_{t-1}), && \text{(transition model)}\\
y_t|x_t &\sim g_t(y_t|x_t), && \text{(likelihood)}
\end{aligned}
$$

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

We are interested estimating expectations

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

sequentially as new data arrives. This problem is known as *the filtering problem*.



A simpler problem

Sequential inference

Let us first consider a generic probabilistic setting,

$$\pi_0(x) \qquad \text{and} \qquad g_t(y_t|x).$$

for $(y_t)_{t\in\mathbb{N}_+}$ a sequence of observations.

Let us first consider a generic probabilistic setting,

$$\pi_0(x) \qquad \text{and} \qquad g_t(y_t|x).$$

for $(y_t)_{t\in\mathbb{N}_+}$ a sequence of observations. We are interested in estimating expectations,

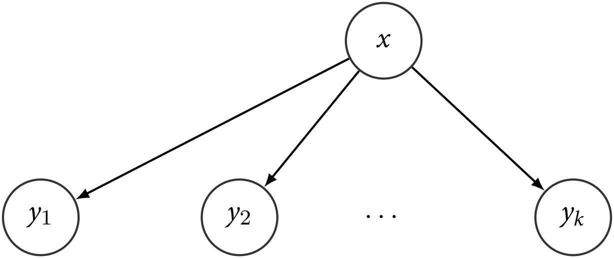
$$(\varphi, \pi_t) = \int \varphi(x)\pi(x|y_{1:t})\mathrm{d}x,$$

Let us first consider a generic probabilistic setting,

$$\pi_0(x) \qquad \text{and} \qquad g_t(y_t|x).$$

for $(y_t)_{t\in\mathbb{N}_+}$ a sequence of observations. We are interested in estimating expectations,

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



How would you obtain $\pi(x|y_{1:t})$?

How would you obtain $\pi(x|y_{1:t})$?

We can use Bayes' rule iteratively

$$\begin{aligned}\pi(x|y_{1:t}) &= \frac{\gamma(x, y_{1:t})}{p(y_{1:t})},\\ &= \frac{g_t(y_t|x)\gamma(x, y_{1:t-1})}{p(y_t|y_{1:t-1})p(y_{1:t-1})},\\ &= \frac{g_t(y_t|x)\pi(x|y_{1:t-1})}{p(y_t|y_{1:t-1})}.\end{aligned}$$

where

$$p(y_t|y_{1:t-1}) = \int g_t(y_t|x)\pi(x|y_{1:t-1})\mathrm{d}x.$$

A simpler problem

Sequential inference

How would you obtain $\pi(x|y_{1:t})$?

We can use Bayes' rule iteratively

$$\begin{aligned}
\pi(x|y_{1:t}) &= \frac{\gamma(x, y_{1:t})}{p(y_{1:t})}, \\
&= \frac{g_t(y_t|x)\gamma(x, y_{1:t-1})}{p(y_t|y_{1:t-1})p(y_{1:t-1})}, \\
&= \frac{g_t(y_t|x)\pi(x|y_{1:t-1})}{p(y_t|y_{1:t-1})}.
\end{aligned}$$

where

$$p(y_t|y_{1:t-1}) = \int g_t(y_t|x)\pi(x|y_{1:t-1})\mathrm{d}x.$$

The previous posterior $\pi(x|y_{1:t-1})$ is used as the prior for the next step.

Let us assume that

$$
\begin{aligned}
\pi_0(x) &= \mathcal{N}(x; \mu_0, V_0), \\
g_t(y_t|x) &= \mathcal{N}(y_t; H_t x, R_t).
\end{aligned}
$$

A simpler problem

Let us assume that

$$\pi_0(x) = \mathcal{N}(x; \mu_0, V_0),$$
$$g_t(y_t|x) = \mathcal{N}(y_t; H_t x, R_t).$$

Can we compute $\pi(x|y_{1:t})$ analytically?

A simpler problem

Sequential inference: the Gaussian case

Let us assume that

$$\pi_0(x) = \mathcal{N}(x; \mu_0, V_0),$$
$$g_t(y_t|x) = \mathcal{N}(y_t; H_t x, R_t).$$

Can we compute $\pi(x|y_{1:t})$ analytically?

Lemma 1

We obtain $\pi(x|y_{1:t}) = \mathcal{N}(x; \mu_t, V_t)$ *where,*

$$\mu_t = \mu_{t-1} + V_{t-1} H_t^\top (R_t + H_t V_{t-1} H_t^\top)^{-1} (y_t - H_t \mu_{t-1}),$$
$$V_t = V_{t-1} - V_{t-1} H_t^\top (R_t + H_t V_{t-1} H_t^\top)^{-1} H_t V_{t-1},$$

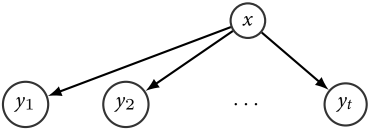
for $t \geq 1$.

Static vs. dynamic setting

Static inference: Given a probability model,

$$x \sim \pi_0(\mathrm{d}x),$$
$$y_t|x_t \sim g_t(y_t|x),$$

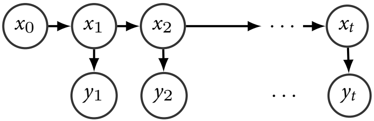
we are interested in static inference: Estimating $\pi(x|y_{1:t})$ sequentially.



Dynamic inference: Given a SSM,

$$x_0 \sim \pi_0(\mathrm{d}x_0),$$
$$x_t|x_{t-1} \sim \tau_t(\mathrm{d}x_t|x_{t-1}),$$
$$y_t|x_t \sim g_t(y_t|x_t),$$

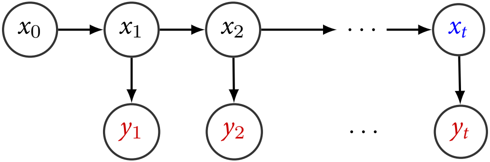
we are interested in *the stochastic filtering problem*: Estimating $\pi_t(x_t|y_{1:t})$.



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

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

Let us look in detail to these steps:

Prediction: Given $\pi_{t-1}(\mathrm{d}x_{t-1}|y_{1:t-1})$, we want to compute $\pi_t(\mathrm{d}x_t|y_{1:t-1})$.

Let us look in detail to these steps:

Prediction: Given $\pi_{t-1}(\mathrm{d}x_{t-1}|y_{1:t-1})$, we want to compute $\pi_t(\mathrm{d}x_t|y_{1:t-1})$.

$$\pi_t(\mathrm{d}x_t|y_{1:t-1}) = \int \pi_{t-1}(\mathrm{d}x_{t-1}|y_{1:t-1})\tau_t(\mathrm{d}x_t|x_{t-1}).$$

Let us look in detail to these steps:

Prediction: Given $\pi_{t-1}(\mathrm{d}x_{t-1}|y_{1:t-1})$, we want to compute $\pi_t(\mathrm{d}x_t|y_{1:t-1})$.

$$\pi_t(\mathrm{d}x_t|y_{1:t-1}) = \int \pi_{t-1}(\mathrm{d}x_{t-1}|y_{1:t-1})\tau_t(\mathrm{d}x_t|x_{t-1}).$$

In terms of densities

$$\pi_t(x_t|y_{1:t-1}) = \int \pi_{t-1}(x_{t-1}|y_{1:t-1})\tau_t(x_t|x_{t-1})\mathrm{d}x_{t-1}.$$

We have already seen the update rule, but we modify this in the dynamic setting: Our prior will now be the predictive distribution $\pi_t(\mathrm{d}x_t|y_{1:t-1})$.

We have already seen the update rule, but we modify this in the dynamic setting: Our prior will now be the predictive distribution $\pi_t(\mathrm{d}x_t|y_{1:t-1})$.

Update: Given $\pi_t(\mathrm{d}x_t|y_{1:t-1})$, we want to compute $\pi_t(\mathrm{d}x_t|y_{1:t})$.

We have already seen the update rule, but we modify this in the dynamic setting: Our prior will now be the predictive distribution $\pi_t(\mathrm{d}x_t|y_{1:t-1})$.

Update: Given $\pi_t(\mathrm{d}x_t|y_{1:t-1})$, we want to compute $\pi_t(\mathrm{d}x_t|y_{1:t})$.

$$\begin{aligned}\pi_t(x_t|y_{1:t}) &= \frac{\gamma(x_t, y_{1:t})}{p(y_{1:t})}, \\ &= \frac{g_t(y_t|x_t)\pi_t(x_t|y_{1:t-1})}{p(y_t|y_{1:t-1})}.\end{aligned}$$

where

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

Let us assume that

$$
\begin{aligned}
\pi_0(x) &= \mathcal{N}(x; \mu_0, V_0), \\
\tau_t(x_t|x_{t-1}) &= \mathcal{N}(x_t; A_t x_{t-1}, Q_t), \\
g_t(y_t|x_t) &= \mathcal{N}(y_t; H_t x_t, R_t).
\end{aligned}
$$

Let us assume that

$$\begin{aligned}
\pi_0(x) &= \mathcal{N}(x; \mu_0, V_0), \\
\tau_t(x_t|x_{t-1}) &= \mathcal{N}(x_t; A_t x_{t-1}, Q_t), \\
g_t(y_t|x_t) &= \mathcal{N}(y_t; H_t x_t, R_t).
\end{aligned}$$

Can we compute $\pi(x_t|y_{1:t})$ analytically?

Lemma 2

Given the optimal filter $\pi_{t-1}(x_{t-1}|y_{1:t-1}) = \mathcal{N}(x_{t-1}; \mu_{t-1}, V_{t-1})$ *at time* $t-1$ *the predictive distribution* $\xi_t(x_t|y_{1:t-1})$ *is given by*

$$\xi_t(x_t|y_{1:t-1}) = \mathcal{N}(x_t; \tilde{\mu}_t, \tilde{V}_t),$$

where,

$$\tilde{\mu}_t = A_t \mu_{t-1}, \tag{1}$$

$$\tilde{V}_t = A_t V_{t-1} A_t^\top + Q_t. \tag{2}$$

```
def kalman_predict(mu, V, A, Q):
    mu_pred = A @ mu
    V_pred = A @ V @ A.T + Q
    return mu_pred, V_pred
```

Lemma 3

Finally, given the predictive distribution $\xi_t(x_t|y_{1:t-1})$, the optimal filter $\pi_t(x_t|y_{1:t})$ is given by

$$\pi_t(x_t|y_{1:t-1}) = \mathcal{N}(x_t; \mu_t, V_t),$$

where,

$$\mu_t = \tilde{\mu}_t + \tilde{V}_t H_t^\top (R_t + H_t \tilde{V}_t H_t^\top)^{-1} (y_t - H_t \tilde{\mu}_t), \tag{3}$$

$$V_t = \tilde{V}_t - \tilde{V}_t H_t^\top (R_t + H_t \tilde{V}_t H_t^\top)^{-1} H_t \tilde{V}_t, \tag{4}$$

using Lemma 1.

```python
def kalman_update(mu_pred, V_pred, H, R, y):
    S = H @ V_pred @ H.T + R
    K = V_pred @ H.T @ np.linalg.inv(S)
    mu = mu_pred + K @ (y - H @ mu_pred)
    V = V_pred - K @ H @ V_pred
    return mu, V
```

Consider the following state-space model

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

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

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

with κ small.

Another crucial quantity in Bayesian computation is the model evidence

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

Another crucial quantity in Bayesian computation is the model evidence

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

Kalman filter provides this quantity as a byproduct.

Note that

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

and

$$p(y_k|y_{1:k-1}) = \int p(y_k|x_k)p(x_k|y_{1:k-1})\mathrm{d}x_k.$$

Note that Then one has

$$p(y_k|y_{1:k-1}) = \mathcal{N}(y_k; H\tilde{\mu}_k, H\tilde{V}_k H^\top + R_k).$$

What if nonlinearities exist in Gaussian models?

$$
\begin{aligned}
\pi_0(x) &= \mathcal{N}(x; \mu_0, V_0), \\
\tau_t(x_t|x_{t-1}) &= \mathcal{N}(x_t; a_t(x_{t-1}), Q_t), \\
g_t(y_t|x_t) &= \mathcal{N}(y_t; h_t(x_t), R_t).
\end{aligned}
$$

Can we still do analytical computations?

What if nonlinearities exist in Gaussian models?

$$\begin{aligned}\pi_0(x) &= \mathcal{N}(x; \mu_0, V_0),\\ \tau_t(x_t|x_{t-1}) &= \mathcal{N}(x_t; a_t(x_{t-1}), Q_t),\\ g_t(y_t|x_t) &= \mathcal{N}(y_t; h_t(x_t), R_t).\end{aligned}$$

Can we still do analytical computations?

Yes! We can use the *extended Kalman filter* (EKF) or the *unscented Kalman filter* (UKF).

Assume that we are given the SSM

$$\begin{aligned}\pi_0(x_0) &= \mathcal{N}(x_0; \mu_0, V_0),\\ \tau_t(x_t|x_{t-1}) &= \mathcal{N}(x_t; a_t(x_{t-1}), Q_t)\\ g_t(y_t|x_t) &= \mathcal{N}(y_t; h_t(x_t), R_t).\end{aligned}$$

where $a_t : \mathsf{X} \to \mathsf{X}$, $h_t : \mathsf{X} \to \mathsf{Y}$, $Q_t \in \mathbb{R}^{d_x \times d_x}$, and $R_t \in \mathbb{R}^{d_y \times d_y}$. Assume that the approximate posterior distribution at time $t-1$ is $\pi_{t-1}^E(x_{t-1}) = \mathcal{N}(x_{t-1}; \mu_{t-1}^E, V_{t-1}^E)$.

If the model is approximately locally linear, one can linearize $a_t(x_t)$ around μ^E_{t-1} and obtain the dynamical model

$$\bar{a}_t(x_t) = a_t(\mu^E_{t-1}) + A_t(x_t - \mu^E_{t-1}) = a_t(\mu^E_{t-1}) + A_t x_t - A_t \mu^E_{t-1}, \tag{5}$$

where

$$A_t = \left. \frac{\partial a_t(x)}{\partial x} \right|_{x=\mu^E_{t-1}}.$$

We can see (5) as a linear model with control inputs. Hence, the prediction step with this linearized model simply becomes

$$\tilde{\mu}^E_t = a_t(\mu^E_{t-1}).$$

The uncertainty is propagated also as in the KF, since (5) is a linear model, hence we obtain

$$\tilde{V}_t^E = A_t V_{t-1}^E A_t^\top + Q_t.$$

Similarly, given $\tilde{\mu}_t^E$, in order to proceed with the observation model we can linearize h_t around $\tilde{\mu}_t^E$, i.e., we construct

$$\bar{h}_t(x_t) = h_t(\tilde{\mu}_t^E) + H_t(x_t - \tilde{\mu}_t^E),$$

where

$$H_t = \left. \frac{\partial h_t(x)}{\partial x} \right|_{x=\tilde{\mu}_t}.$$

Given the linearization, the EKF update step now becomes

$$\begin{aligned} \mu_t^E &= \tilde{\mu}_{t-1}^E + \tilde{V}_t^E H_t^\top (R_t + H_t \tilde{V}_t^E H_t^\top)^{-1} (y_t - h_t(\tilde{\mu}_t^E)), \\ V_t^E &= \tilde{V}_t^E - \tilde{V}_t^E H_t^\top (R_t + H_t \tilde{V}_t^E H_t^\top)^{-1} H_t \tilde{V}_t^E. \end{aligned}$$

Finally, one can compactly summarize the EKF as follows. Given $\pi^E_{t-1}(x_{t-1}) = \mathcal{N}(x_{t-1}; \mu^E_{t-1}, V^E_{t-1})$, the new posterior pdf $\pi^E_t(x_t) = \mathcal{N}(x_t; \mu^E_t, V^E_t)$ is obtained via

$$\tilde{\mu}^E_t = a_t(\mu^E_{t-1}), \tag{6}$$

$$\tilde{V}^E_t = A_t V^E_{t-1} A_t^\top + Q_t, \tag{7}$$

$$\mu^E_t = \tilde{\mu}^E_{t-1} + \tilde{V}^E_t H_t^\top (R_t + H_t \tilde{V}^E_t H_t^\top)^{-1} (y_t - h_t(\tilde{\mu}^E_t)), \tag{8}$$

$$V^E_t = \tilde{V}^E_t - \tilde{V}^E_t H_t^\top (R_t + H_t \tilde{V}^E_t H_t^\top)^{-1} H_t \tilde{V}^E_t. \tag{9}$$

Other kinds of Gaussian approximations are very popular:

- *Unscented Kalman filter* (UKF): The UKF uses a deterministic sampling technique called the *unscented transform* to obtain a Gaussian approximation of the posterior distribution.

Other kinds of Gaussian approximations are very popular:

- *Unscented Kalman filter* (UKF): The UKF uses a deterministic sampling technique called the *unscented transform* to obtain a Gaussian approximation of the posterior distribution.
- *Gaussian sum filter* (GSF): The GSF uses a Gaussian mixture approximation of the posterior distribution.

Other kinds of Gaussian approximations are very popular:

- *Unscented Kalman filter* (UKF): The UKF uses a deterministic sampling technique called the *unscented transform* to obtain a Gaussian approximation of the posterior distribution.
- *Gaussian sum filter* (GSF): The GSF uses a Gaussian mixture approximation of the posterior distribution.
- *ensemble Kalman filter* (EnKF): The EnKF uses a Monte Carlo approximation of the posterior distribution.

Other kinds of Gaussian approximations are very popular:

- *Unscented Kalman filter* (UKF): The UKF uses a deterministic sampling technique called the *unscented transform* to obtain a Gaussian approximation of the posterior distribution.
- *Gaussian sum filter* (GSF): The GSF uses a Gaussian mixture approximation of the posterior distribution.
- *ensemble Kalman filter* (EnKF): The EnKF uses a Monte Carlo approximation of the posterior distribution.

Many other variants, very popular in fields like robotics, navigation, guidance, aerospace, finance, vision, etc.

Other kinds of Gaussian approximations are very popular:

- *Unscented Kalman filter* (UKF): The UKF uses a deterministic sampling technique called the *unscented transform* to obtain a Gaussian approximation of the posterior distribution.
- *Gaussian sum filter* (GSF): The GSF uses a Gaussian mixture approximation of the posterior distribution.
- *ensemble Kalman filter* (EnKF): The EnKF uses a Monte Carlo approximation of the posterior distribution.

Many other variants, very popular in fields like robotics, navigation, guidance, aerospace, finance, vision, etc.

A great reference on all things practical about filtering: Särkkä (2013): *Bayesian Filtering and Smoothing*.

Deterministic approximations are only useful in certain settings where we can ensure

- Exact or approximate linearity
- Gaussianity

Deterministic approximations are only useful in certain settings where we can ensure

- Exact or approximate linearity
- Gaussianity

For more general cases, one option is to use Monte Carlo methods.

Deterministic approximations are only useful in certain settings where we can ensure

- Exact or approximate linearity
- Gaussianity

For more general cases, one option is to use Monte Carlo methods.

Next: A general Monte Carlo approach to estimate expectations w.r.t. posterior distributions $\pi_t^N(\mathrm{d}x_t|y_{1:t})$.

Consider a target measure $\pi(x)\mathrm{d}x$ and a function $\varphi(x)$. If we have access to i.i.d samples from $X_i \sim \pi(x)$, then

$$(\varphi, \pi) := \int \varphi(x)\pi(x)\mathrm{d}x \approx \frac{1}{N}\sum_{i=1}^{N} \varphi(X_i),$$

using a particle approximation

$$\pi^N(\mathrm{d}x) = \frac{1}{N}\sum_{i=1}^{N} \delta_{X_i}(\mathrm{d}x),$$

since by definition of the Dirac measure, we have

$$\varphi(y) = \int \varphi(x)\delta_y(\mathrm{d}x).$$

Perfect Monte Carlo

Theorem 1 (Perfect Monte Carlo)

Let φ be a bounded function. Then, for any $N \geq 1$,

$$\|(\varphi, \pi) - (\varphi, \pi^N)\|_2 \leq \frac{2\|\varphi\|_\infty}{\sqrt{N}}.$$

Typically, we do not have access to i.i.d samples from π.

Typically, we do not have access to i.i.d samples from π.

A well-known approach to compute expectations (φ, π) is called *importance sampling*.

Typically, we do not have access to i.i.d samples from π.

A well-known approach to compute expectations (φ, π) is called *importance sampling*.

Assume, π is absolutely continuous w.r.t. q, denoted as $\pi \ll q$, meaning $\pi(x) = 0 \implies q(x) = 0$.

Typically, we do not have access to i.i.d samples from π.

A well-known approach to compute expectations (φ, π) is called *importance sampling*.

Assume, π is absolutely continuous w.r.t. q, denoted as $\pi \ll q$, meaning $\pi(x) = 0 \implies q(x) = 0$.

Then, we can write

$$(\varphi, \pi) = \int \varphi(x)\pi(\mathrm{d}x) = \int \varphi(x)\frac{\mathrm{d}\pi}{\mathrm{d}q}(x)q(x)\mathrm{d}x.$$

When π and q admit densities,

$$(\varphi, \pi) = \int \varphi(x)\pi(x)\mathrm{d}x = \int \varphi(x)\frac{\pi(x)}{q(x)}q(x)\mathrm{d}x.$$

Given

$$(\varphi, \pi) = \int \varphi(x) \frac{\pi(x)}{q(x)} q(x) \mathrm{d}x,$$

we can employ standard Monte Carlo by sampling $X_i \sim q$ and then constructing (by setting $w = \pi/q$)

$$\begin{aligned}(\varphi, \tilde{\pi}^N) &= \frac{1}{N} \sum_{i=1}^{N} \varphi(X_i) w(X_i), \\ &= \frac{1}{N} \sum_{i=1}^{N} \mathrm{w}_i \varphi(X_i).\end{aligned}$$

where $\mathrm{w}_i = w(X_i)$. We will call this estimator the importance sampling (IS) estimator.

Mini-quiz: Is this estimator unbiased?

Mini-quiz: Is this estimator unbiased?

Yes.

$$\begin{aligned}
\mathbb{E}_q[(\varphi, \tilde{\pi}^N)] &= \mathbb{E}_q\left[\frac{1}{N}\sum_{i=1}^{N} \mathrm{w}_i \varphi(X_i)\right], \\
&= \frac{1}{N}\sum_{i=1}^{N} \mathbb{E}_q\left[\frac{\pi(X_i)}{q(X_i)}\varphi(X_i)\right] \\
&= \frac{1}{N}\sum_{i=1}^{N} \int \frac{\pi(x)}{q(x)}\varphi(x) q(x) \mathrm{d}x \\
&= \int \varphi(x)\pi(x)\mathrm{d}x = (\varphi, \pi).
\end{aligned}$$

What is the variance?

$$
\begin{aligned}
\mathrm{var}_q[(\varphi, \tilde{\pi}^N)] &= \mathrm{var}_q\left[\frac{1}{N}\sum_{i=1}^{N} \mathrm{w}_i \varphi(X_i)\right] \\
&= \frac{1}{N^2}\mathrm{var}_q\left[\sum_{i=1}^{N} w(X_i)\varphi(X_i)\right] \\
&= \frac{1}{N}\mathrm{var}_q\left[w(X)\varphi(X)\right] \qquad \text{where } X \sim q(x) \\
&= \frac{1}{N}\left(\mathbb{E}_q\left[w^2(X)\varphi^2(X)\right] - \mathbb{E}_q\left[w(X)\varphi(X)\right]^2\right) \\
&= \frac{1}{N}\left(\mathbb{E}_q\left[w^2(X)\varphi^2(X)\right] - \bar{\varphi}^2\right).
\end{aligned}
$$

What if we only have access to $\gamma(x) \propto \pi(x)$?

What if we only have access to $\gamma(x) \propto \pi(x)$?

Assume $\gamma \ll q$ and both abs. cont w.r.t. to the Lebesgue measure. Then we can write

$$\begin{aligned}(\varphi, \pi) &= \int \varphi(x)\pi(x)\mathrm{d}x \\ &= \frac{\int \varphi(x)\frac{\gamma(x)}{q(x)}q(x)\mathrm{d}x}{\int \frac{\gamma(x)}{q(x)}q(x)\mathrm{d}x}.\end{aligned}$$

We can then perform the same Monte Carlo integration idea but now both for the numerator and denominator.

We have

$$(\varphi, \pi) = \int \varphi(x)\pi(x)\mathrm{d}x$$
$$= \frac{\int \varphi(x)\frac{\gamma(x)}{q(x)}q(x)\mathrm{d}x}{\int \frac{\gamma(x)}{q(x)}q(x)\mathrm{d}x}.$$

Define $W(x) = \gamma(x)/q(x)$ and the SNIS approximation is given as

$$(\varphi, \pi) = \frac{\int \varphi(x)W(x)q(x)\mathrm{d}x}{\int W(x)q(x)\mathrm{d}x} \approx \frac{\frac{1}{N}\sum_{i=1}^{N}\varphi(X_i)W(X_i)}{\frac{1}{N}\sum_{i=j}^{N}W(X_j)}.$$

where $X_i \sim q(x)$. Let us write $\mathrm{W}_i = W(X_i)$ and $\mathrm{w}_i = \mathrm{W}_i / \sum_{j=1}^{N}\mathrm{W}_j$. Then the final estimator is

$$(\varphi, \tilde{\pi}^N) = \sum_{i=1}^{N}\mathrm{w}_i\varphi(X_i)$$

Mini-quiz: Is this estimator unbiased?

Mini-quiz: Is this estimator unbiased?

No.

Mini-quiz: Is this estimator unbiased?

No.

The estimator is a ratio of two unbiased estimators. However, this ratio is *not* unbiased.

However, one can prove that

$$\|(\varphi, \pi) - (\varphi, \tilde{\pi}^N)\|_p \leq \frac{\tilde{c}_p \|\varphi\|_\infty}{\sqrt{N}},$$

where $\tilde{c}_p$ is a constant depending on p and q and φ is bounded.

Theorem 2

Assume $\|W\|_\infty < \infty$. The L_2 error (i.e., set $p = 2$) is bounded by

$$\|(\varphi, \pi) - (\varphi, \tilde{\pi}^N)\|_2 \leq \frac{c_2 \|\varphi\|_\infty}{\sqrt{N}}$$

where

$$c_2 = \frac{4\|W\|_\infty}{(W, q)}.$$

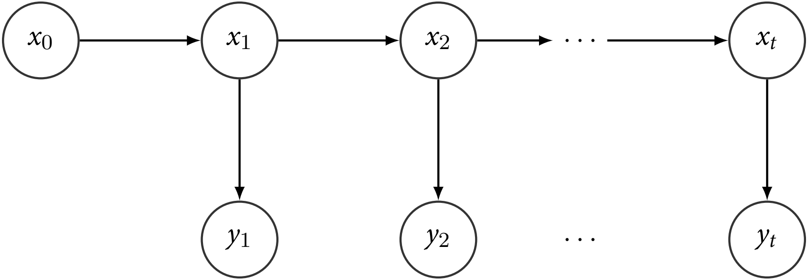


Figure: The conditional independence structure of a state-space model.

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

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

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

Before we go into the details of the derivation, let us directly look at the algorithm.

Before we go into the details of the derivation, let us directly look at the algorithm.

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

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

Where does the algorithm come from?

Where does the algorithm come from?

Surprisingly, we will not use the prediction-update recursions directly unlike in the Kalman filter.

Where does the algorithm come from?

Surprisingly, we will not use the prediction-update recursions directly unlike in the Kalman filter.

We will instead develop an importance sampler on the path space.

The key recursion on the path distributions:

$$
\begin{aligned}
\pi_t(x_{0:t}|y_{1:t}) &= \frac{\gamma(x_{0:t}, y_{1:t})}{p(y_{1:t})} \\
&= \frac{\gamma(x_{0:t-1}, y_{1:t-1})}{p(y_{1:t-1})} \frac{\tau(x_t|x_{t-1}) g(y_t|x_t)}{p(y_t|y_{1:t-1})} \\
&= \pi_t(x_{0:t-1}|y_{1:t-1}) \frac{\tau(x_t|x_{t-1}) g(y_t|x_t)}{p(y_t|y_{1:t-1})}.
\end{aligned}
$$

Let us denote the proposal over the entire path space with $q(x_{0:t}|y_{1:t})$. Note the "unnormalised target"

$$\gamma(x_{0:t}, y_{1:t}) = \mu(x_0) \prod_{k=1}^{t} \tau(x_k|x_{k-1}) g(y_k|x_k). \tag{10}$$

Let us denote the proposal over the entire path space with $q(x_{0:t}|y_{1:t})$. Note the "unnormalised target"

$$\gamma(x_{0:t}, y_{1:t}) = \mu(x_0) \prod_{k=1}^{t} \tau(x_k|x_{k-1}) g(y_k|x_k). \tag{10}$$

This simply the joint distribution of all variables $(x_{0:t}, y_{1:t})$. Just as in the regular importance sampling

$$W_{0:t}(x_{0:t}) = \frac{\gamma(x_{0:t}, y_{1:t})}{q(x_{0:t}|y_{1:t})}.$$

Obviously, given samples from the proposal $x_{0:t}^{(i)} \sim q(x_{0:t}|y_{1:t})$, by evaluating the weight $\mathrm{W}_{0:t}^{(i)} = W_{0:t}(x_{0:t}^{(i)})$ for $i = 1, \ldots, N$ and building a particle approximation, we can get

$$\pi^N(\mathrm{d}x_{0:t}) = \sum_{i=1}^{N} \mathrm{W}_{0:t}^{(i)} \delta_{x_{0:t}^{(i)}}(\mathrm{d}x_{0:t}).$$

Let us consider the decomposition of the proposal

$$q(x_{0:t}) = q(x_0) \prod_{k=1}^{t} q(x_k | x_{0:k-1}, y_{1:k}).$$

Note that, based on this, we can build a recursion for the function $W(x_{0:t})$ by writing

$$\begin{aligned} W_{0:t}(x_{0:t}) &= \frac{\gamma(x_{0:t}, y_{1:t})}{q(x_{0:t})}, \\ &= \frac{\gamma(x_{0:t-1}, y_{1:t-1})}{q(x_{0:t-1}|y_{1:t-1})} \frac{\tau(x_t|x_{t-1}) g(y_t|x_t)}{q(x_t|x_{0:t-1}, y_{1:t})}, \\ &= W_{0:t-1}(x_{0:t-1}) \frac{\tau(x_t|x_{t-1}) g(y_t|x_t)}{q(x_t|x_{0:t-1}, y_{1:t})}, \\ &= W_{0:t-1}(x_{0:t-1}) W_t(x_{0:t}). \end{aligned} \tag{11}$$

Particle filters

This is still not optimal, as we still need to store the whole path.

This is still not optimal, as we still need to store the whole path.

We can further simplify our proposal by assuming a Markov structure.

$$q(x_{0:t}) = q(x_0) \prod_{k=1}^{t} q(x_k|x_{k-1}).$$

This allows us to obtain purely recursive weight computation

$$\begin{aligned} W_{0:t}(x_{0:t}) &= \frac{\gamma(x_{0:t}, y_{1:t})}{q(x_{0:t})}, && (12) \\ &= \frac{\gamma(x_{0:t-1}, y_{1:t-1})}{q(x_{0:t-1})} \frac{\tau(x_t|x_{t-1})g(y_t|x_t)}{q(x_t|x_{t-1})}, && (13) \\ &= W_{0:t-1}(x_{0:t-1}) \frac{\tau(x_t|x_{t-1})g(y_t|x_t)}{q(x_t|x_{t-1})}, && (14) \\ &= W_{0:t-1}(x_{0:t-1}) W_t(x_t, x_{t-1}), && (15) \end{aligned}$$

Assume that we have computed the unnormalised weights $\mathrm{W}_{0:t-1}^{(i)} = W(x_{0:t-1}^{(i)})$ recursively and obtained samples $x_{0:t-1}^{(i)}$. We only need the last sample $x_{t-1}^{(i)}$ to obtain the weight update given in (15). We can now sample from the Markov proposal

$$x_t^{(i)} \sim q(x_t | x_{t-1}^{(i)})$$

and compute the weights of the path sampler at time t as

$$\mathrm{W}_{1:t}^{(i)} = \mathrm{W}_{1:t-1}^{(i)} \times \mathrm{W}_t^{(i)},$$

where

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

Given the samples $x_{t-1}^{(i)}$, we first perform sampling step

$$x_t^{(i)} \sim q(x_t | x_{t-1})$$

and then compute

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

and update

$$\mathrm{W}_{1:t}^{(i)} = \mathrm{W}_{1:t-1}^{(i)} \times \mathrm{W}_t^{(i)}.$$

These are unnormalised weights and we normalise them to obtain,

$$\mathrm{w}_{1:t}^{(i)} = \frac{\mathrm{W}_{1:t}^{(i)}}{\sum_{i=1}^{N} \mathrm{W}_{1:t}^{(i)}},$$

which finally leads to the empirical measure,

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

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

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

    and update

$$\mathrm{W}_{1:t}^{(i)} = \mathrm{W}_{1:t-1}^{(i)} \times \mathrm{W}_t^{(i)}.$$

    Normalise weights,

$$\mathrm{w}_{1:t}^{(i)} = \frac{\mathrm{W}_{1:t}^{(i)}}{\sum_{i=1}^N \mathrm{W}_{1:t}^{(i)}}.$$

  - Report

$$\pi_t^N(\mathrm{d}x_{0:t}) = \sum_{i=1}^N \mathrm{w}_{1:t}^{(i)} \delta_{x_{0:t}^{(i)}}(\mathrm{d}x_{0:t}).$$

There is a well-known problem with this scheme: *Weight degeneracy*.

There is a well-known problem with this scheme: *Weight degeneracy*.

To resolve this, the approach is to introduce resampling steps.

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

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

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

$$\pi_t^N(\mathrm{d}x_{0:t}) = \sum_{i=1}^N \mathrm{w}_{1:t}^{(i)} \delta_{x_{0:t}^{(i)}}(\mathrm{d}x_{0:t}).$$

  - Resample:

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

The bootstrap particle filter (BPF) is the SISR algorithm with the following choices:

$$q(x_t|x_{t-1}) = \tau(x_t|x_{t-1}),$$

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

$$\mathrm{W}_t^{(i)} = g(y_t | x_t^{(i)}),$$

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

$$\pi_t^N(\mathrm{d}x_{0:t}) = \sum_{i=1}^{N} \mathrm{w}_{1:t}^{(i)} \delta_{x_{0:t}^{(i)}}(\mathrm{d}x_{0:t}).$$

  - Resample:

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

Theorem 3

Assume that the likelihood function is positive and bounded

$$g_t(x_t) > 0 \qquad \text{and} \qquad \|g_t\|_\infty = \sup_{x_t \in \mathsf{X}} g_t(x_t) < \infty,$$

for all $t \geq 1$. *Let* φ *be a bounded function and* π_t^N *be particle filter approximations of* π_t. *Then, for any* $N \geq 1$,

$$\|(\varphi, \pi_t) - (\varphi, \pi_t^N)\|_2 \leq \frac{c_t \|\varphi\|_\infty}{\sqrt{N}}.$$

where $c_t < \infty$ *is a constant independent of* 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 $i = 1, \dots, N$,

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

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

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

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

- Resample:

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

Let us look the following Lorenz 63 model

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

where $\gamma = 0.01$, $\mathrm{r} = 28$, $\mathrm{b} = 8/3$, $\mathrm{s} = 10$, 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.

Bootstrap particle filter
Marginal likelihoods

Recall that we have tbe factorisation:

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

where

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

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

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

Bootstrap particle filter

Marginal likelihoods

Therefore, given

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

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:

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

Remarkably, this estimate is unbiased:

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

This is of crucial importance for methods like particle MCMC.

Remarkably, this estimate is unbiased:

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

This is of crucial importance for methods like particle MCMC.

Exercise: Estimate this for the linear Gaussian model and compare it to Kalman filter's estimate over M Monte Carlo iterations.

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 - they are crucial for parameter estimation.

Next: Smoothing algorithms.

In this talk, we rely on key smoothing recursions (there are others):

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

where

$$\pi(x_{t+1}|y_{1:t}) = \int \tau(x_{t+1}|x_t)\pi(x_t|y_{1:t})\mathrm{d}x_t.$$

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.

Let us consider our linear-Gaussian model again

$$\begin{aligned} \pi_0(x) &= \mathcal{N}(x; \mu_0, V_0), \\ \tau_t(x_t|x_{t-1}) &= \mathcal{N}(x_t; A_t x_{t-1}, Q_t), \\ g_t(y_t|x_t) &= \mathcal{N}(y_t; H_t x_t, R_t). \end{aligned}$$

In this setting, smoothing can be exactly implemented too.

Let us consider our linear-Gaussian model again

$$\begin{aligned}
\pi_0(x) &= \mathcal{N}(x; \mu_0, V_0), \\
\tau_t(x_t|x_{t-1}) &= \mathcal{N}(x_t; A_t x_{t-1}, Q_t), \\
g_t(y_t|x_t) &= \mathcal{N}(y_t; H_t x_t, R_t).
\end{aligned}$$

In this setting, smoothing can be exactly implemented too.

The resulting algorithm is called the *Rauch-Tung-Striebel* (RTS) smoother.

Assume we have computed filter moments $(\mu_t, V_t)_{t=0}^{T}$.

Assume we have computed filter moments $(\mu_t, V_t)_{t=0}^{T}$. The smoother is then given as

$$\begin{aligned} \mu_T^s &= \mu_T, \\ V_T^s &= V_T, \\ \mu_t^s &= \mu_t + J_t(\mu_{t+1}^s - A_t\mu_t), \\ V_t^s &= V_t + J_t(V_{t+1}^s - V_t)J_t^\top, \end{aligned}$$

where

$$J_t = V_t A_t^\top \hat{V}_{t+1}^{-1}.$$

Assume we have computed filter moments $(\mu_t, V_t)_{t=0}^T$.

Assume we have computed filter moments $(\mu_t, V_t)_{t=0}^T$. The smoother is then given as

$$\begin{aligned} \mu_T^s &= \mu_T, \\ V_T^s &= V_T, \\ \mu_t^s &= \mu_t + J_t(\mu_{t+1}^s - A_t\mu_t), \\ V_t^s &= V_t + J_t(V_{t+1}^s - V_t)J_t^\top, \end{aligned}$$

where

$$J_t = V_t A_t^\top \hat{V}_{t+1}^{-1}.$$

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

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

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

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

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

Yes, but...

The smoothing problem

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

- Sample:

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

- Weight

$$\mathrm{w}_t^{(i)} \propto \frac{\tau(\bar{x}_t^{(i)} | x_{t-1}^{(i)}) g(y_t | \bar{x}_t^{(i)})}{q_t(\bar{x}_t^{(i)} | x_{t-1}^{(i)})},$$

- Resample: Choose $a_t^{(i)}$ where $\mathbb{P}(a_t^{(i)} = j) \propto \mathrm{w}_t^j$ 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.

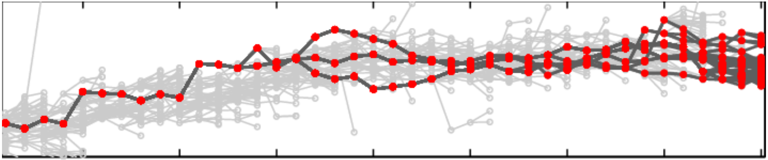


Figure: Source: Svensson, Andreas, Thomas B. Schön, and Manon Kok. "Non-linear state space smoothing using the conditional particle filter." (2015).

Instead, we can consider the following decomposition

$$\pi(x_{0:T}|y_{1:T}) = \pi(x_T|y_{0:T}) \prod_{k=0}^{T-1} \pi(x_k|y_{0:T}, x_{k+1}),$$
$$= \pi(x_T|y_{0:T}) \prod_{k=0}^{T-1} \pi(x_k|y_{0:k}, x_{k+1}). \quad (16)$$

where

$$\pi(x_t|x_{t+1}, y_{1:t}) = \frac{\pi(x_t, x_{t+1}|y_{1:t})}{\xi(x_{t+1}|y_{1:t})}, \quad (17)$$
$$= \frac{\tau(x_{t+1}|x_t)\pi(x_t|y_{1:t})}{\xi(x_{t+1}|y_{1:t})}. \quad (18)$$

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

This recursion suggests sampling $\pi(x_T|y_{1:T})$ from the filter and sample backwards from $\pi(x_k|y_{0:k}, x_{k+1})$ 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

$$\pi(\mathrm{d}x_t|x_{t+1}, y_{1:t}) = \frac{\tau(x_{t+1}|x_t)\pi^N(\mathrm{d}x_t|y_{1:t})}{\xi^N(x_{t+1}|y_{1:t})}.$$

where π^N and ξ^N approximate filtering and predictive measures (see next slide).

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

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

$$\pi^N(\mathrm{d}x_t|x_{t+1}, y_{1:t}) = \frac{\sum_{i=1}^N \mathrm{w}_t^{(i)}\tau(x_{t+1}|\bar{x}_t^{(i)})\delta_{\bar{x}_t^{(i)}}(\mathrm{d}x_t)}{\sum_{i=1}^N \mathrm{w}_t^{(i)}\tau(x_{t+1}|\bar{x}_t^{(i)})} \quad (19)$$

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

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

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

  - 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:

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

Recall the original smoothing recursions we discussed:

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

Can we use these to build a particle approximation?

Recall the original smoothing recursions we discussed:

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

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

$$
\pi(\mathrm{d}x_t|y_{1:T}) = \pi(\mathrm{d}x_t|y_{1:t}) \int \frac{\tau(x_{t+1}|x_t)}{\xi(x_{t+1}|y_{1:t})} \pi(x_{t+1}|y_{1:T}) \mathrm{d}x_{t+1}.
$$

Backward recursion

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

Backward recursion

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

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

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

Assume we have an approximation

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

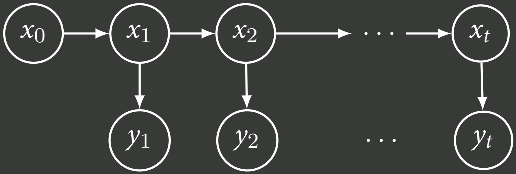
where $\mathrm{w}_{T|T}^{(i)} = \mathrm{w}_T^{(i)}$. We can use the recursion in the previous slide to obtain

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

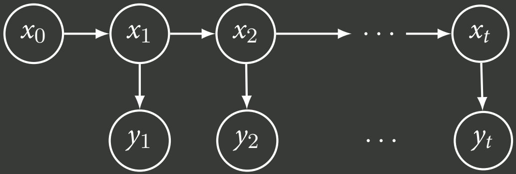
where

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

We have seen inference for

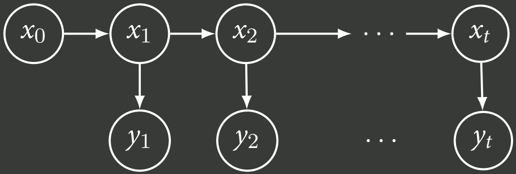


We have seen inference for

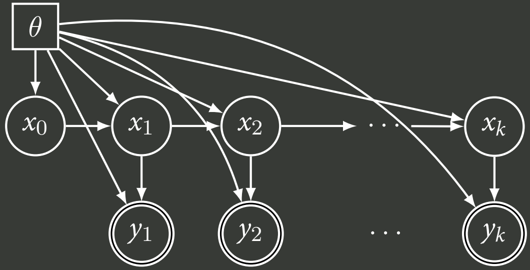


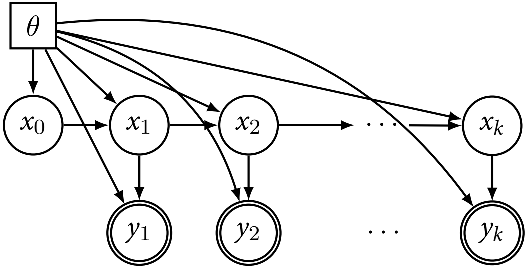
What if the model has parameters θ?

We have seen inference for



What if the model has parameters θ?





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

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

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

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

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

Proof.

Let us note that

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

■

Given Fisher's identity,

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

and

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

The parameter estimation problem

How to compute the gradient?

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

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

The parameter estimation problem

How to compute the gradient?

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.

There is a method called forward smoothing, which can build the smoothed additive functional expectations *online*. Let us go back and write, for $n < T$,

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

where

$$V_n^\theta(x_n) = \int \left(\sum_{k=1}^n s_k(x_{k-1}, x_k)\right) p_\theta(x_{0:n-1}|y_{0:n-1}, x_n)\mathrm{d}x_{0:n-1}.$$

The key recursion, note that

$$
\begin{aligned}
V_{n+1}^{\theta}(x_{n+1}) &= \int \left(\sum_{k=1}^{n+1} s_k(x_{k-1}, x_k) \right) p_\theta(x_{0:n}|y_{0:n}, x_{n+1}) \mathrm{d}x_{0:n}, \\
&= \int \left(\sum_{k=1}^{n} s_k(x_{k-1}, x_k) + s_n(x_{n-1}, x_n) \right) \\
&\quad p_\theta(x_{0:n-1}|y_{0:n-1}, x_n) \mathrm{d}x_{0:n-1} p_\theta(x_n|y_{0:n}, x_{n+1}) \mathrm{d}x_n, \\
&= \int \left(V_n^{\theta}(x_n) + s_n(x_{n-1}, x_n) \right) p_\theta(x_n|y_{0:n}, x_{n+1}) \mathrm{d}x_n.
\end{aligned}
$$

We have a recursion for $(V_n^{\theta})_{n \geq 1}$ that can be estimated online using $(x_t^{(i)}, x_{t+1}^{(i)})$.

The parameter estimation problem

How to compute the gradient?

How do compute things only forward pass? Recall FFBS

- 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

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

Forward only smoothing: Assume we have a good approximation of $V_t^\theta(x_t^{(i)})$.

- Sample $\bar{x}_{t+1}^{(i)} \sim f(\cdot|x_t^{(i)})$,
- Use it to compute FFBS smoothing weights (with predictive particles)

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

and

$$V_{t+1}^\theta(\bar{x}_{t+1}^{(i)}) = \sum_{j=1}^N \mathrm{w}_{t+1|t}^{(i)} \left(V_t^\theta(x_t^{(i)}) + s_{t+1}(x_t^{(i)}, x_{t+1}^{(i)}) \right).$$

and build

$$S_{t+1}^{\theta,N} = \sum_{j=1}^N \mathrm{w}_{t+1}^{(i)} V_t^\theta(x_{t+1}^{(i)}).$$

Forward only smoothing: Assume we have a good approximation of $V_t^\theta(x_t^{(i)})$.

- Sample $\bar{x}_{t+1}^{(i)} \sim f(\cdot|x_t^{(i)})$,
- Use it to compute FFBS smoothing weights (with predictive particles)

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

and

$$V_{t+1}^\theta(\bar{x}_{t+1}^{(i)}) = \sum_{j=1}^N \mathrm{w}_{t+1|t}^{(i)} \left(V_t^\theta(x_t^{(i)}) + s_{t+1}(x_t^{(i)}, x_{t+1}^{(i)}) \right).$$

and build

$$S_{t+1}^{\theta,N} = \sum_{j=1}^N \mathrm{w}_{t+1}^{(i)} V_t^\theta(x_{t+1}^{(i)}).$$

Forward smoothing.

We have discussed MLE approach.

We have discussed MLE approach.

We can also look at the Bayesian estimation in SSMs, i.e., for the model where we have $p(\theta)$ as the prior of θ.

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.

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.

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 and Míguez, 2014):

$$\kappa(\mathrm{d}\theta|\theta') = (1-\epsilon_N)\delta_{\theta'}(\mathrm{d}\theta) + \epsilon_N\tau(\mathrm{d}\theta|\theta'), \tag{20}$$

  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.

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 $i = 1, \dots, N$.
- 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:

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

  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 $i = 1, \ldots, N$.
- 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.
- 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.

This machinery and much more was built for the last 30 years for filtering and solving other problems.

This machinery and much more was built for the last 30 years for filtering and solving other problems.

There are lots of exciting directions available (discussion).

Thanks!

References I

- Crisan, Dan and Joaquín Míguez (2014). "Particle-kernel estimation of the filter density in state-space models". In: *Bernoulli* 20.4, pp. 1879–1929.
- Särkkä, Simo (2013). *Bayesian filtering and smoothing*. Cambridge University Press.