Advanced Computational Methods in Statistics
Lecture 3

O. Deniz Akyildiz

LTCC Advanced Course

November 27, 2023

Imperial College London

Recall our basic task:

Recall our basic task:

- We wanted to sample from distributions $\pi(x) \propto \gamma(x)$ given only the knowledge of $\gamma(x)$ and use these samples to estimate an integral

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

for some function φ.

Recall our basic task:

- We wanted to sample from distributions $\pi(x) \propto \gamma(x)$ given only the knowledge of $\gamma(x)$ and use these samples to estimate an integral

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

for some function φ.

We will now consider *dynamic* settings.

- We will have a *sequence* of distributions $(\pi)_{t\geq 0}$ and we will want to estimate the integrals

$$(\varphi, \pi_t) = \int \varphi(x)\pi_t(x)\,\mathrm{d}x$$

for some function φ.

Recall our basic task:

- We wanted to sample from distributions $\pi(x) \propto \gamma(x)$ given only the knowledge of $\gamma(x)$ and use these samples to estimate an integral

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

  for some function φ.

We will now consider *dynamic* settings.

- We will have a *sequence* of distributions $(\pi)_{t\geq 0}$ and we will want to estimate the integrals

  $$(\varphi, \pi_t) = \int \varphi(x)\pi_t(x)\,\mathrm{d}x$$

  for some function φ.

This is called the filtering problem.

State-space models

problem definition

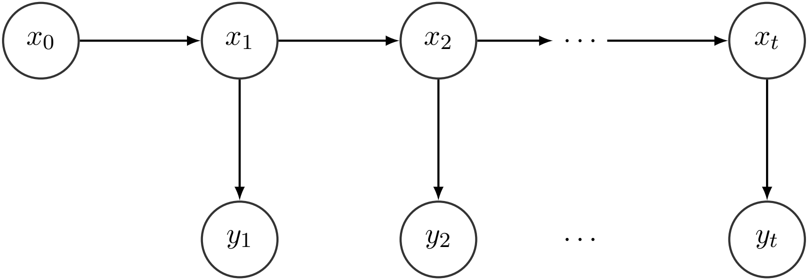


Figure: The conditional independence structure of a state-space model.

$(x_t)_{t\in\mathbb{N}_+}$: *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: $g_t(x_t) = g_t(y_t|x_t)$.

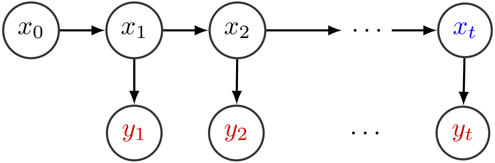
State-space models

problem definition

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.

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. We are interested in estimating expectations,

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

A simpler problem

Sequential inference

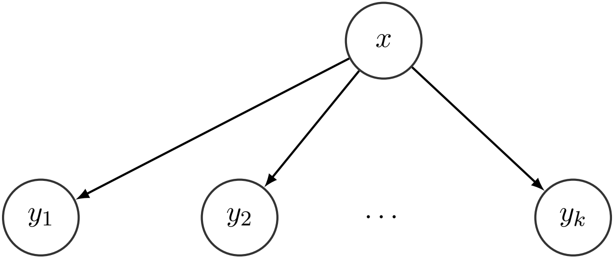
Imperial College London

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,

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



A simpler problem

Sequential inference

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

A simpler problem

Sequential inference

Mini-quiz: 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

Mini-quiz: 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.

A simpler problem

Sequential inference: the Gaussian case

Let us assume that

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

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?

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,*

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

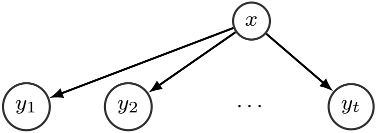
for $t \geq 1$*.*

Static vs. dynamic setting

Static inference: Given a probability model,

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

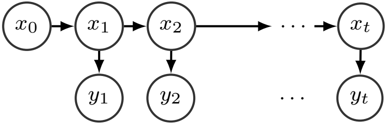
we are interested in static inference: Estimating $\pi(x|y_{1:t})$ sequentially.



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 $\pi_t(x_t|y_{1:t})$.



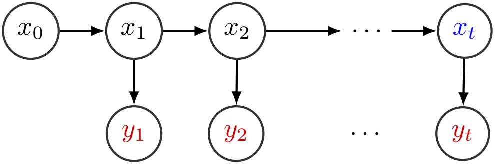
State-space models

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

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

State-space models

Algorithmic principle - prediction

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 $\pi_t(\mathrm{d}x_t|y_{1:t-1})$.

State-space models

Algorithmic principle - prediction

Imperial College London

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 $\pi_t(\mathrm{d}x_t|y_{1:t-1})$.

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

State-space models

Algorithmic principle - prediction

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

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

State-space models

Algorithmic principle - update

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

State-space models

Algorithmic principle - update

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 $\pi_t(\mathrm{d}x_t|y_{1:t})$.

State-space models

Algorithmic principle - update

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

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

State-space models

The Kalman filter: Linear-Gaussian case

Imperial College London

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

State-space models

The Kalman filter: Linear-Gaussian case

Yes, we can!

State-space models

The Kalman filter: Linear-Gaussian case

Yes, we can!

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,

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

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

Lemma 3

Finally, 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,

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

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

from Lemma 1.

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?

State-space models

Kalmanesque filters

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).

State-space models

Kalmanesque filters - EKF

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 : X $\to$ X, h_t : X $\to$ 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 $\pi^E_{t-1}(x_{t-1}) = \mathcal{N}(x_{t-1}; \mu^E_{t-1}, V^E_{t-1})$.

State-space models

Kalmanesque filters - EKF

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

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

where

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

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

$$\tilde{\mu}_t^E = a_t(\mu_{t-1}^E).$$

State-space models

Kalmanesque filters - EKF

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

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

State-space models

Kalmanesque filters - EKF

Finally, one can compactly summarize the EKF as follows. Given $\pi_{t-1}^E(x_{t-1}) = \mathcal{N}(x_{t-1}; \mu_{t-1}^E, V_{t-1}^E)$, the new posterior pdf $\pi_t^E(x_t) = \mathcal{N}(x_t; \mu_t^E, V_t^E)$ is obtained via

$$\tilde{\mu}_t^E = a_t(\mu_{t-1}^E), \tag{6}$$

$$\tilde{V}_t^E = A_t V_{t-1}^E A_t^\top + Q_t, \tag{7}$$

$$\mu_t^E = \tilde{\mu}_{t-1}^E + \tilde{V}_t^E H_t^\top (R_t + H_t \tilde{V}_t^E H_t^\top)^{-1} (y_t - h_t(\tilde{\mu}_t^E)), \tag{8}$$

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

State-space models

Kalmanesque filters

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.

State-space models

Kalmanesque filters

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.

State-space models

Kalmanesque filters

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.

State-space models

Kalmanesque filters II - UKF

We can use the *unscented transform* to obtain a Gaussian approximation of the posterior distribution.

We can use the *unscented transform* to obtain a Gaussian approximation of the posterior distribution.

Let X be a Gaussian random variable $X \sim \mathcal{N}(x; \mu, \Sigma)$.

State-space models

Kalmanesque filters II - UKF

We can use the *unscented transform* to obtain a Gaussian approximation of the posterior distribution.

Let X be a Gaussian random variable $X \sim \mathcal{N}(x; \mu, \Sigma)$.

Say we would like to compute moments of $g(X)$ where g is a non-linear function.

The unscented transform precisely approximates the moments of $g(X)$ by the moments of a Gaussian random variable.

The unscented transform precisely approximates the moments of $g(X)$ by the moments of a Gaussian random variable.

The unscented transform is based on the idea of *sigma points*, which are chosen deterministically:

$$\begin{aligned}
\sigma_0 &= \mu, \\
\sigma_i &= \mu + \left(\sqrt{(n+\lambda)\Sigma}\right)_i, \quad i = 1, \ldots, n, \\
\sigma_i &= \mu - \left(\sqrt{(n+\lambda)\Sigma}\right)_{i-n}, \quad i = n+1, \ldots, 2n,
\end{aligned}$$

where $\lambda = \alpha^2(n+\kappa) - n$ and α and κ are parameters that can be chosen freely.

How is this idea used given:

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

State-space models

Kalmanesque filters II - UKF

How is this idea used given:

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

Given μ_{t-1}, V_{t-1}, use the nonlinearity $a_t(\cdot)$ using unscented transform to compute the moments of prediction.

State-space models

Kalmanesque filters II - UKF

How is this idea used given:

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

Given μ_{t-1}, V_{t-1}, use the nonlinearity $a_t(\cdot)$ using unscented transform to compute the moments of prediction.

Given μ_t, V_t, use the nonlinearity $h_t(\cdot)$ using unscented transform to compute the moments of updated posterior.

State-space models

The smoothing problem

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

State-space models

The smoothing problem

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 $T > t$?

State-space models

The smoothing problem

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.

State-space models

The smoothing problem

We have smoothing recursions

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

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

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.

State-space models

The smoothing problem

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.

State-space models

The smoothing problem

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.

State-space models

The smoothing problem

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.

State-space models

The smoothing problem

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

State-space models

The smoothing problem

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

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

State-space models

The smoothing problem

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

State-space models

The smoothing problem

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

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

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

We will now introduce a general Monte Carlo approach to estimate posterior distributions $\pi_t^N(\mathrm{d}x_t|y_{1:t})$.

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

- Exact or approximate linearity
- Gaussianity

We will now introduce a general Monte Carlo approach to estimate posterior distributions $\pi_t^N(\mathrm{d}x_t|y_{1:t})$.

Particle filters.

Particle filters

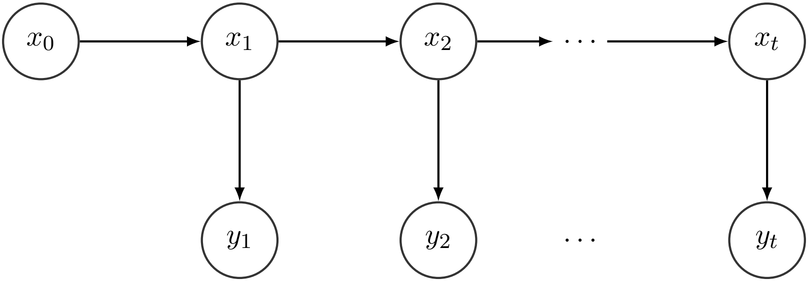


Figure: The conditional independence structure of a state-space model.

$(x_t)_{t\in\mathbb{N}_+}$: *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: $g_t(x_t) = g_t(y_t|x_t)$.

Particle filters

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

Particle filters

Before we go into the details of the derivation, let us directly look at the algorithm. 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 $i = 1, \ldots, N$.
- 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,
$$x_t^{(i)} \sim \tilde{\pi}_t(\mathrm{d}x) := \sum_i w_t^{(i)} \delta_{\bar{x}_t^{(i)}}(\mathrm{d}x) \quad \text{for } i = 1, ..., N.$$

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

Particle filters

Derivation

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.

Particle filters

Derivation

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.

Particle filters

Derivation

The key recursion on the path distributions:

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

Particle filters

Derivation

Recall importance sampling: Assume that we aim at estimating expectations of a given density π, i.e., we would like to compute

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

We also assume that sampling from this density is not possible and we can only evaluate the *unnormalised* density $\gamma(x)$.

Particle filters

Derivation

One way to estimate this expectation is to sample from a proposal measure q and rewrite the integral as

$$
\begin{aligned}
(\varphi, \pi) &= \int \varphi(x)\pi(x)\mathrm{d}x, \\
&= \frac{\int \varphi(x)\frac{\gamma(x)}{q(x)}q(x)\mathrm{d}x}{\int \frac{\gamma(x)}{q(x)}q(x)\mathrm{d}x}, \\
&\approx \frac{\frac{1}{N}\sum_{i=1}^{N} \varphi(x^{(i)})\frac{\gamma(x^{(i)})}{q(x^{(i)})}}{\frac{1}{N}\sum_{i=1}^{N} \frac{\gamma(x^{(i)})}{q(x^{(i)})}}, \qquad x^{(i)} \sim q, \quad i = 1, \ldots, N.
\end{aligned}
\tag{10}
$$

Particle filters

Derivation

Let us now introduce the unnormalised weight function

$$W(x) = \frac{\gamma(x)}{q(x)}. \tag{11}$$

With this, the Eq. (10) becomes

$$\begin{aligned}(\varphi, \pi^N) &= \frac{\frac{1}{N}\sum_{i=1}^N \varphi(x^{(i)})W(x^{(i)})}{\frac{1}{N}\sum_{i=1}^N W(x^{(i)})}, \qquad x^{(i)} \sim q, \quad i = 1, \dots, N, \\ &= \frac{\sum_{i=1}^N \varphi(x^{(i)})\mathsf{W}^{(i)}}{\sum_{i=1}^N \mathsf{W}^{(i)}}, \qquad x^{(i)} \sim q, \quad i = 1, \dots, N,\end{aligned}$$

where $\mathsf{W}^{(i)} = W(x^{(i)})$ are called *the unnormalised weights*.

Particle filters

Derivation

Finally, we can obtain the estimator in a more convenient form,

$$(\varphi, \pi^N) = \sum_{i=1}^{N} \mathsf{w}^{(i)} \varphi(x^{(i)}).$$

by introducing the *normalised importance weights*

$$\mathsf{w}^{(i)} = \frac{\mathsf{w}^{(i)}}{\sum_{i=1}^{N} \mathsf{w}^{(i)}}, \tag{12}$$

for $i = 1, \ldots, N$. We note that the particle approximation of π in this case is given as

$$\pi^N(\mathrm{d}x) = \sum_{i=1}^{N} \mathsf{w}^{(i)} \delta_{x^{(i)}}(\mathrm{d}x). \tag{13}$$

In the following, we will derive the importance sampler aiming at building particle approximations of $\pi_t(x_{0:t}|y_{1:t})$ for a state-space model.

Particle filters

Derivation

The proposal over the entire path space $x_{0:t}$ denoted $q(x_{0:t})$. Note

$$\gamma(x_{0:t}, y_{1:t}) = \mu(x_0) \prod_{k=1}^{t} \tau(x_k|x_{k-1}) g(y_k|x_k). \tag{14}$$

The proposal over the entire path space $x_{0:t}$ denoted $q(x_{0:t})$. Note

$$\gamma(x_{0:t}, y_{1:t}) = \mu(x_0) \prod_{k=1}^{t} \tau(x_k|x_{k-1}) g(y_k|x_k). \tag{14}$$

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

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

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

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

Let us consider a decomposition of the proposal

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

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

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

Particle filters

Derivation - sequential approach

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

Particle filters

Derivation - sequential approach

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

$$\begin{aligned} W_{0:t}(x_{0:t}) &= \frac{\gamma(x_{0:t}, y_{1:t})}{q(x_{0:t})}, && (16)\\ &= \frac{\gamma(x_{0:t-1}, y_{1:t-1})}{q(x_{0:t-1})} \frac{\tau(x_t|x_{t-1}) g(y_t|x_t)}{q(x_t|x_{t-1})}, && (17)\\ &= W_{0:t-1}(x_{0:t-1}) \frac{\tau(x_t|x_{t-1}) g(y_t|x_t)}{q(x_t|x_{t-1})}, && (18)\\ &= W_{0:t-1}(x_{0:t-1}) W_t(x_t, x_{t-1}), && (19) \end{aligned}$$

Particle filters

Sequential Importance Sampling (SIS)

Let us implement it. Assume that we have computed the unnormalised weights $\mathsf{W}_{1:t-1}^{(i)} = W(x_{0:t-1}^{(i)})$ 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 (19). And also note that $\mathsf{W}_{1:t-1}^{(i)}$ for $i = 1, \ldots, N$ are just numbers, they do not need the storage of previous samples. 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

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

where

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

Particle filters

Sequential Importance Sampling (SIS)

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

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

and update

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

These are unnormalised weights and we normalise them to obtain,

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

which finally leads to the empirical measure,

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

Particle filters

Sequential Importance Sampling (SIS)

- 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:

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

    and update

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

    Normalise weights,

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

  - Report

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

Particle filters

Sequential Importance Sampling (SIS)

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

Particle filters

Sequential Importance Sampling (SIS)

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

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

Particle filters

Sequential Importance Sampling - Resampling (SISR)

- 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:

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

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

  - Report

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

  - Resample:

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

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

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

Particle filters

Bootstrap particle filter

- 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:
    $$\mathsf{W}_t^{(i)} = g(y_t | x_t^{(i)}),$$
    Normalise: $\mathsf{w}_{1:t}^{(i)} = \mathsf{W}_{1:t}^{(i)} / \sum_{i=1}^{N} \mathsf{W}_{1:t}^{(i)}$
  - Report
    $$\pi_t^N(\mathrm{d}x_{0:t}) = \sum_{i=1}^{N} \mathsf{w}_{1:t}^{(i)} \delta_{x_{0:t}^{(i)}}(\mathrm{d}x_{0:t}).$$
  - Resample:
    $$x_t^{(i)} \sim \sum_{i=1}^{N} \mathsf{w}_t^{(i)} \delta_{\tilde{x}_t^{(i)}}(\mathrm{d}x_t).$$

References I