Introduction to Sequential Monte Carlo Methods

Arnaud Doucet

MLSS 2007
Sequential Monte Carlo (SMC) are a set of methods allowing us to approximate virtually any sequence of probability distributions.
Sequential Monte Carlo (SMC) are a set of methods allowing us to approximate virtually any sequence of probability distributions.

SMC are very popular in physics where they are used to compute eigenvalues of positive operators, the solution of PDEs/integral equations or simulate polymers.
Sequential Monte Carlo (SMC) are a set of methods allowing us to approximate virtually any sequence of probability distributions.

SMC are very popular in physics where they are used to compute eigenvalues of positive operators, the solution of PDEs/integral equations or simulate polymers.

We focus here on Applications of SMC to Hidden Markov Models (HMM) for pedagogical reasons...
Sequential Monte Carlo (SMC) are a set of methods allowing us to approximate virtually any sequence of probability distributions.

SMC are very popular in physics where they are used to compute eigenvalues of positive operators, the solution of PDEs/integral equations or simulate polymers.

We focus here on Applications of SMC to Hidden Markov Models (HMM) for pedagogical reasons...

... and because this is certainly closer to your interests!
Preliminary Remarks

- *Sequential Monte Carlo* (SMC) are a set of methods allowing us to approximate virtually *any sequence of probability distributions*.

- SMC are very popular in physics where they are used to compute eigenvalues of positive operators, the solution of PDEs/integral equations or simulate polymers.

- We focus here on *Applications of SMC to Hidden Markov Models* (HMM) for pedagogical reasons...

- ... and because this is certainly closer to your interests!

- In the HMM framework, SMC are also widely known as Particle Filtering/Smoothing methods.
Filtering, smoothing and parameter estimation in HMM.
Organization of the Lectures

- Filtering, smoothing and parameter estimation in HMM.
- SMC for HMM.
Organization of the Lectures

- Filtering, smoothing and parameter estimation in HMM.
- SMC for HMM.
- Advanced SMC for HMM.
Organization of the Lectures

- Filtering, smoothing and parameter estimation in HMM.
- SMC for HMM.
- Advanced SMC for HMM.
- Recent Developments and Open Problems.
Markov Models

- We model the stochastic processes of interest as a discrete-time Markov process \( \{X_k\}_{k \geq 1} \).
We model the stochastic processes of interest as a discrete-time Markov process \( \{X_k\}_{k \geq 1} \).

\( \{X_k\}_{k \geq 1} \) is characterized by its initial density

\[
X_1 \sim \mu(\cdot)
\]

and its transition density

\[
X_k | (X_{k-1} = x_{k-1}) \sim f(\cdot | x_{k-1}).
\]
Markov Models

- We model the stochastic processes of interest as a discrete-time Markov process \( \{X_k\}_{k \geq 1} \).
- \( \{X_k\}_{k \geq 1} \) is characterized by its initial density \( X_1 \sim \mu(\cdot) \)
  and its transition density
  \[
  \mathbb{P}(X_k | X_{k-1} = x_{k-1}) \sim f(\cdot | x_{k-1}).
  \]
- We introduce the notation \( x_{i:j} = (x_i, x_{i+1}, \ldots, x_j) \) for \( i \leq j \). We have by definition
  \[
  \mathbb{P}(x_{1:n}) = \mathbb{P}(x_1) \prod_{k=2}^{n} \mathbb{P}(x_k | x_{1:k-1})
  = \mu(x_1) \prod_{k=2}^{n} f(x_k | x_{k-1})
  \]
Assume you want to track a target in the XY plane then you can consider the 4-dimensional state

\[ X_k = (X_{k,1}, V_{k,1}, X_{k,2}, V_{k,2})^T \]
Assume you want to track a target in the XY plane then you can consider the 4-dimensional state

$$X_k = (X_{k,1}, V_{k,1}, X_{k,2}, V_{k,2})^T$$

The so-called constant velocity model states that

$$X_k = AX_{k-1} + W_k, \ W_k \sim_{\text{i.i.d.}} \mathcal{N}(0, \Sigma),$$

$$A = \begin{pmatrix} A_{CV} & 0 \\ 0 & A_{CV} \end{pmatrix}, \ A_{CV} = \begin{pmatrix} 1 & T \\ 0 & 1 \end{pmatrix},$$

$$\Sigma = \sigma^2 \begin{pmatrix} \Sigma_{CV} & 0 \\ 0 & \Sigma_{CV} \end{pmatrix}, \ \Sigma_{CV} = \begin{pmatrix} T^3/3 & T^2/2 \\ T^2/2 & T \end{pmatrix}$$
Assume you want to track a target in the XY plane then you can consider the 4-dimensional state

\[ \mathbf{X}_k = (X_{k,1}, V_{k,1}, X_{k,2}, V_{k,2})^T \]

The so-called constant velocity model states that

\[ \mathbf{X}_k = A \mathbf{X}_{k-1} + \mathbf{W}_k, \quad \mathbf{W}_k \overset{i.i.d.}{\sim} \mathcal{N}(0, \Sigma), \]

\[ A = \begin{pmatrix} A_{CV} & 0 \\ 0 & A_{CV} \end{pmatrix}, \quad A_{CV} = \begin{pmatrix} 1 & T \\ 0 & 1 \end{pmatrix}, \]

\[ \Sigma = \sigma^2 \begin{pmatrix} \Sigma_{CV} & 0 \\ 0 & \Sigma_{CV} \end{pmatrix}, \quad \Sigma_{CV} = \begin{pmatrix} T^3/3 & T^2/2 \\ T^2/2 & T \end{pmatrix} \]

We obtain that

\[ f \left( x_k \mid x_{k-1} \right) = \mathcal{N}(x_k; Ax_{k-1}, \Sigma). \]
A basic model for speech signals consists of modelling them as autoregressive (AR) processes; i.e.

\[ S_k = \sum_{i=1}^{d} \alpha_i S_{k-i} + V_k, \quad V_k \sim \mathcal{N}(0, \sigma_s^2) \]
Speech Enhancement

- A basic model for speech signals consists of modelling them as autoregressive (AR) processes; i.e.
\[
S_k = \sum_{i=1}^{d} \alpha_i S_{k-i} + V_k, \quad V_k \sim i.i.d. \mathcal{N}(0, \sigma_s^2)
\]

- If we write \(U_k = (S_k, \ldots, S_{k-d})^T\) then we have equivalently
\[
U_k = AU_{k-1} + BV_k
\]
where
\[
A = \begin{pmatrix}
\alpha_1 & \alpha_2 & \cdots & \alpha_d \\
1 &  &  & \\
& \ddots & & \\
& & 1 & \\
\end{pmatrix}, \quad B = \begin{pmatrix}
1 \\
0 \\
\vdots \\
0
\end{pmatrix}.
\]
A basic model for speech signals consists of modelling them as autoregressive (AR) processes; i.e.

\[ S_k = \sum_{i=1}^{d} \alpha_i S_{k-i} + V_k, \quad V_k \sim \mathcal{N}(0, \sigma_s^2) \]

If we write \( U_k = (S_k, \ldots, S_{k-d})^T \) then we have equivalently

\[ U_k = AU_{k-1} + BV_k \]

where

\[
A = \begin{pmatrix}
\alpha_1 & \alpha_2 & \cdots & \alpha_d \\
1 & \ddots & & \\
& & \ddots & \ddots \\
& & & 1
\end{pmatrix}, \quad B = \begin{pmatrix} 1 \\ \vdots \\ 0 \\ 0 \end{pmatrix}.
\]

We have

\[ f_U(u_k | u_{k-1}) = \mathcal{N}(u_k; (Au_{k-1})_{1:d}, \sigma_s^2) \delta_{(u_{k-1})_{1:d-1}}((u_k)_{2:d}) \]
This model could be not flexible enough and we might want additionally to make the AR coefficient time-varying.
This model could be not flexible enough and we might want additionally to make the AR coefficient time-varying.

Defining $\alpha_k = (\alpha_{k,1}, \alpha_{k,1}, \ldots, \alpha_{k,d})$, we could consider

$$\alpha_k = \alpha_{k-1} + W_k \text{ where } W_k \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, \sigma^2_{\alpha} I_d)$$

which implies that

$$f_\alpha(\alpha_k | \alpha_{k-1}) = \mathcal{N}(\alpha_k; \alpha_{k-1}, \sigma^2_{\alpha} I_d).$$
• This model could be not flexible enough and we might want additionally to make the AR coefficient time-varying.

• Defining $\alpha_k = (\alpha_{k,1}, \alpha_{k,1}, \ldots, \alpha_{k,d})$, we could consider

\[
\alpha_k = \alpha_{k-1} + W_k \quad \text{where} \quad W_k \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, \sigma^2_\alpha I_d)
\]

which implies that

\[
f_{\alpha_k}(\alpha_k | \alpha_{k-1}) = \mathcal{N}(\alpha_k; \alpha_{k-1}, \sigma^2_\alpha I_d).
\]

• The process $X_k = (\alpha_k, U_k)$ is Markov with transition density

\[
f(x_k | x_{k-1}) = \mathcal{N}(\alpha_k; \alpha_{k-1}, \sigma^2_\alpha I_d) \mathcal{N}(u_k; (A_k u_{k-1})_1, \sigma^2_s) \\
\times \delta((u_{k-1})_1:d-1)((u_k)_{2:d})
\]

where $(A_k)_1 = \alpha_k$. 
The (simplified) Heston model (1993) is used to described the dynamics of an asset price $S_t$ using the following model for $X_t = \log (S_t)$

$$dX_t = \mu dt + dW_t + dZ_t$$

where $Z_t$ is a jump process.
The (simplified) Heston model (1993) is used to described the dynamics of an asset price $S_t$ using the following model for $X_t = \log (S_t)$

$$dX_t = \mu dt + dW_t + dZ_t$$

where $Z_t$ is a jump process.

We can approximate this process by a discrete-time Markov process using an Euler scheme

$$X_{t+\delta} = X_t + \delta \mu + W_{t+\delta,t} + Z_{t+\delta,t}.$$
The (simplified) Heston model (1993) is used to described the dynamics of an asset price $S_t$ using the following model for $X_t = \log (S_t)$

$$dX_t = \mu dt + dW_t + dZ_t$$

where $Z_t$ is a jump process.

We can approximate this process by a discrete-time Markov process using an Euler scheme

$$X_{t+\delta} = X_t + \delta \mu + W_{t+\delta,t} + Z_{t+\delta,t}.$$  

Similar discretization schemes are used for biochemical networks (e.g. D. Wilkinson, Stochastic modelling for systems biology, CRC, 2006), disease dynamics (e.g. E.L. Ionides, PNAS, 2006) or population dynamics.
We do not observe $\{X_k\}_{k \geq 1}$; the process is *hidden*. We only have access to another related process $\{Y_k\}_{k \geq 1}$.
Observation Model

- We do not observe $\{X_k\}_{k \geq 1}$; the process is *hidden*. We only have access to another related process $\{Y_k\}_{k \geq 1}$.
- We assume that, conditional on $\{X_k\}_{k \geq 1}$, the observations $\{Y_k\}_{k \geq 1}$ are independent and marginally distributed according to

$$Y_k \mid (X_k = x_k) \sim g(\cdot \mid x_k).$$
We do not observe \( \{X_k\}_{k \geq 1} \); the process is *hidden*. We only have access to another related process \( \{Y_k\}_{k \geq 1} \).

We assume that, conditional on \( \{X_k\}_{k \geq 1} \), the observations \( \{Y_k\}_{k \geq 1} \) are independent and marginally distributed according to

\[
Y_k \mid (X_k = x_k) \sim g(\cdot | x_k).
\]

Formally this means that

\[
p(y_{1:n} \mid x_{1:n}) = \prod_{k=1}^{n} g(y_k | x_k).
\]
**Figure:** Graphical model representation of HMM
The observation equation is dependent on the sensor.
The observation equation is dependent on the sensor.

**Simple case**

\[ Y_k = CX_k + DE_k, \quad E_k \overset{i.i.d.}{\sim} \mathcal{N}(0, \Sigma_e) \]

so

\[ g(y_k | x_k) = \mathcal{N}(y_k; Cx_k, \Sigma_e). \]
The observation equation is dependent on the sensor.

**Simple case**

\[ Y_k = CX_k + DE_k, \quad E_k \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, \Sigma_e) \]

so

\[ g(y_k | x_k) = \mathcal{N}(y_k; CX_k, \Sigma_e). \]

**Complex realistic case** (Bearings-only-tracking)

\[ Y_k = \tan^{-1}\left(\frac{X_{k,2}}{X_{k,1}}\right) + E_k, \quad E_k \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, \sigma^2) \]

so

\[ g(y_k | x_k) = \mathcal{N}\left(y_k; \tan^{-1}\left(\frac{X_{k,2}}{X_{k,1}}\right), \sigma^2\right). \]
Stochastic Volatility

- We have the following standard model

\[ X_k = \phi X_{k-1} + V_k, \quad V_k \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, \sigma^2) \]

so that

\[ f(x_k | x_{k-1}) = \mathcal{N}(x_k; \phi x_{k-1}, \sigma^2). \]
Stochastic Volatility

- We have the following standard model

\[ X_k = \phi X_{k-1} + V_k, \quad V_k \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, \sigma^2) \]

so that

\[ f(x_k | x_{k-1}) = \mathcal{N}(x_k; \phi x_{k-1}, \sigma^2). \]

- We observe

\[ Y_k = \beta \exp\left(\frac{X_k}{2}\right) \mathcal{W}_k, \quad \mathcal{W}_k \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, 1) \]

so that

\[ g(y_k | x_k) = \mathcal{N}(y_k; \beta \exp(x_k), 1). \]
Given a realization of the observations $Y_{1:n} = y_{1:n}$, we are interested in inferring the states $X_{1:n}$.
Inference in HMM

- Given a realization of the observations \( Y_{1:n} = y_{1:n} \), we are interested in inferring the states \( X_{1:n} \).
- We are in a Bayesian framework where

\[
\text{Prior: } p(x_{1:n}) = \mu(x_1) \prod_{k=2}^{n} f(x_k | x_{k-1}),
\]

\[
\text{Likelihood: } p(y_{1:n}|x_{1:n}) = \prod_{k=1}^{n} g(y_k | x_k)
\]
Inference in HMM

- Given a realization of the observations $Y_{1:n} = y_{1:n}$, we are interested in inferring the states $X_{1:n}$.
- We are in a Bayesian framework where

$$
Prior: \quad p(x_{1:n}) = \mu(x_1) \prod_{k=2}^{n} f(x_k | x_{k-1}),
$$

$$
Likelihood: \quad p(y_{1:n} | x_{1:n}) = \prod_{k=1}^{n} g(y_k | x_k)
$$

- Using Bayes’ rule, we obtain

$$
p(x_{1:n} | y_{1:n}) = \frac{p(y_{1:n} | x_{1:n}) p(x_{1:n})}{p(y_{1:n})}
$$

where the marginal likelihood is given by

$$
p(y_{1:n}) = \int p(y_{1:n} | x_{1:n}) p(x_{1:n}) \, dx_{1:n}.
$$
From this posterior distribution, we can compute any point estimate.
From this posterior distribution, we can compute any point estimate. The joint Maximum a Posteriori (MAP) sequence is given by

$$\arg\max p(x_{1:n} | y_{1:n})$$
Point Estimates

- From this posterior distribution, we can compute any point estimate.
  - The joint Maximum a Posteriori (MAP) sequence is given by
    \[
    \text{arg max } p (x_{1:n} \mid y_{1:n})
    \]
  - The marginal MAP is given for \( k \leq n \) by
    \[
    \text{arg max } p (x_k \mid y_{1:n})
    \]
    where the marginal smoothing distribution is
    \[
    p (x_k \mid y_{1:n}) = \int p (x_{1:n} \mid y_{1:n}) \, dx_{1:k-1} \, dx_{k+1:n}
    \]
From this posterior distribution, we can compute any point estimate.

- The joint Maximum a Posteriori (MAP) sequence is given by
  \[
  \arg \max_{x_1:n} p(x_1:n | y_1:n)
  \]

- The marginal MAP is given for \( k \leq n \) by
  \[
  \arg \max_{x_k} p(x_k | y_1:n)
  \]

where the marginal smoothing distribution is

\[
p(x_k | y_1:n) = \int p(x_1:n | y_1:n) \, dx_{1:k-1} \, dx_{k+1:n}
\]

- We have also the minimum mean square estimate
  \[
  \mathbb{E} [X_k | y_1:n] = \int x_k p(x_k | y_1:n) \, dx_k.
  \]
From this posterior distribution, we can compute any point estimate.

- The joint Maximum a Posteriori (MAP) sequence is given by
  \[
  \arg\max \ p(x_{1:n} \mid y_{1:n})
  \]

- The marginal MAP is given for \( k \leq n \) by
  \[
  \arg\max \ p(x_k \mid y_{1:n})
  \]
  where the marginal smoothing distribution is
  \[
  p(x_k \mid y_{1:n}) = \int p(x_{1:n} \mid y_{1:n}) \, dx_{1:k-1} \, dx_{k+1:n}
  \]

- We have also the minimum mean square estimate
  \[
  \mathbb{E}[X_k \mid y_{1:n}] = \int x_k \, p(x_k \mid y_{1:n}) \, dx_k.
  \]

Conceptually, there is no problem whatsoever.
In particular, we will focus here on the sequential estimation of $p(x_{1:n}|y_{1:n})$ and $p(y_{1:n})$; that is at each time $n$ we want update our knowledge of the hidden process in light of $y_n$. 

There is a simple recursion relating $p(x_{1:n}|y_{1:n})$ to $p(x_{1:n}|y_{1:n})$ given by 

$$p(x_{1:n}|y_{1:n}) = p(x_{1:n}|y_{1:n})f(x_n|x_{1:n})g(y_n|x_n)\ p(y_{1:n}|y_{1:n})dx_{1:n}.$$ 

We will also simply write $p(x_{1:n}|y_{1:n}) \propto p(x_{1:n}|y_{1:n})f(x_n|x_{1:n})g(y_n|x_n)$.
In particular, we will focus here on the sequential estimation of $p(x_{1:n} \mid y_{1:n})$ and $p(y_{1:n})$; that is at each time $n$ we want update our knowledge of the hidden process in light of $y_n$.

There is a simple recursion relating $p(x_{1:n-1} \mid y_{1:n-1})$ to $p(x_{1:n} \mid y_{1:n})$ given by

$$p(x_{1:n} \mid y_{1:n}) = p(x_{1:n-1} \mid y_{1:n-1}) \frac{f(x_n \mid x_{n-1}) g(y_n \mid x_n)}{p(y_n \mid y_{1:n-1})}$$

where

$$p(y_n \mid y_{1:n-1}) = \int g(y_n \mid x_n) f(x_n \mid x_{n-1}) p(x_{n-1} \mid y_{1:n-1}) \, dx_{n-1:n}.$$
In particular, we will focus here on the *sequential estimation* of $p(\mathbf{x}_{1:n} | \mathbf{y}_{1:n})$ and $p(\mathbf{y}_{1:n})$; that is at each time $n$ we want update our knowledge of the hidden process in light of $y_n$.

There is a simple recursion relating $p(\mathbf{x}_{1:n-1} | \mathbf{y}_{1:n-1})$ to $p(\mathbf{x}_{1:n} | \mathbf{y}_{1:n})$ given by

$$p(\mathbf{x}_{1:n} | \mathbf{y}_{1:n}) = p(\mathbf{x}_{1:n-1} | \mathbf{y}_{1:n-1}) \frac{f(\mathbf{x}_n | \mathbf{x}_{n-1}) g(\mathbf{y}_n | \mathbf{x}_n)}{p(\mathbf{y}_n | \mathbf{y}_{1:n-1})}$$

where

$$p(\mathbf{y}_n | \mathbf{y}_{1:n-1}) = \int g(\mathbf{y}_n | \mathbf{x}_n) f(\mathbf{x}_n | \mathbf{x}_{n-1}) p(\mathbf{x}_{n-1} | \mathbf{y}_{1:n-1}) \, d\mathbf{x}_{n-1:n}.$$

We will also simply write

$$p(\mathbf{x}_{1:n} | \mathbf{y}_{1:n}) \propto p(\mathbf{x}_{1:n-1} | \mathbf{y}_{1:n-1}) f(\mathbf{x}_n | \mathbf{x}_{n-1}) g(\mathbf{y}_n | \mathbf{x}_n).$$
The "proof" is trivial and only involves rewriting

\[ p( x_{1:n} | y_{1:n} ) = \frac{ p( x_{1:n} | y_{1:n} ) }{ p( x_{1:n-1} | y_{1:n-1} ) } p( x_{1:n-1} | y_{1:n-1} ) \]

\[ = \frac{ p( x_{1:n}, y_{1:n} ) }{ p( x_{1:n-1}, y_{1:n-1} ) } / p( y_{1:n} ) p( x_{1:n-1} | y_{1:n-1} ) \]
The "proof" is trivial and only involves rewriting

\[ p(x_{1:n} | y_{1:n}) = \frac{p(x_{1:n} | y_{1:n})}{p(x_{1:n-1} | y_{1:n-1})} p(x_{1:n-1} | y_{1:n-1}) \]

\[ = \frac{p(x_{1:n}, y_{1:n})}{p(x_{1:n-1}, y_{1:n-1})} \frac{p(y_{1:n})}{p(y_{1:n-1})} p(x_{1:n-1} | y_{1:n-1}) \]

Now we have

\[ \frac{p(x_{1:n}, y_{1:n})}{p(x_{1:n-1}, y_{1:n-1})} = f(x_n | x_{n-1}) g(y_n | x_n) \]

and

\[ \frac{p(y_{1:n})}{p(y_{1:n-1})} = p(y_n | y_{1:n-1}) \]

and the result follows.
In many papers/books in the literature, you will find the following two-step prediction-updating recursion for the marginals so-called filtering distributions $p(x_n|y_{1:n})$ which is a direct consequence.
• In many papers/books in the literature, you will find the following two-step prediction-updating recursion for the marginals so-called filtering distributions \( p(x_n|y_{1:n}) \) which is a direct consequence.

• Prediction Step

\[
p(x_n|y_{1:n-1}) = \int p(x_{n-1:n}|y_{1:n-1}) \, dx_{n-1} \\
= \int p(x_n|x_{n-1}, y_{1:n-1}) \, p(x_{n-1}|y_{1:n-1}) \, dx_{n-1} \\
= \int f(x_n|x_{n-1}) \, p(x_{n-1}|y_{1:n-1}) \, dx_{n-1}.
\]
In many papers/books in the literature, you will find the following two-step prediction-updating recursion for the marginals so-called filtering distributions $p(x_n|y_{1:n})$ which is a direct consequence.

**Prediction Step**

\[
p(x_n|y_{1:n-1}) = \int p(x_{n-1:n}|y_{1:n-1}) \, dx_{n-1} \\
= \int p(x_n|x_{n-1}, y_{1:n-1}) \, p(x_{n-1}|y_{1:n-1}) \, dx_{n-1} \\
= \int f(x_n|x_{n-1}) \, p(x_{n-1}|y_{1:n-1}) \, dx_{n-1}.
\]

**Updating Step**

\[
p(x_n|y_{1:n}) = \frac{g(y_n|x_n) \, p(x_n|y_{1:n-1})}{p(y_n|y_{1:n-1})}
\]
In many papers/books in the literature, you will find the following two-step prediction-updating recursion for the marginals so-called filtering distributions \( p(x_{n|y_{1:n}}) \) which is a direct consequence.

**Prediction Step**

\[
p(x_{n|y_{1:n-1}}) = \int p(x_{n-1:n|y_{1:n-1}}) \, dx_{n-1}
\]

\[
= \int p(x_{n|x_{n-1},y_{1:n-1}}) \, p(x_{n-1|y_{1:n-1}}) \, dx_{n-1}
\]

\[
= \int f(x_{n|x_{n-1}}) \, p(x_{n-1|y_{1:n-1}}) \, dx_{n-1}.
\]

**Updating Step**

\[
p(x_{n|y_{1:n}}) = \frac{g(y_{n|x_{n}}) \, p(x_{n|y_{1:n-1}})}{p(y_{n|y_{1:n-1}})}
\]

Although we will not use directly the filtering recursion for SMC, the filtering distributions will also prove useful.
We have seen that

\[ p (y_{1:n}) = \int p (y_{1:n} | x_{1:n}) p (x_{1:n}) \, dx_{1:n}. \]
(Marginal) Likelihood Evaluation

- We have seen that

\[ p(y_{1:n}) = \int p(y_{1:n} \mid x_{1:n}) p(x_{1:n}) \, dx_{1:n}. \]

- We also have the following decomposition

\[ p(y_{1:n}) = p(y_1) \prod_{k=2}^{n} p(y_k \mid y_{1:k-1}) \]

where

\[ p(y_k \mid y_{1:k-1}) = \int p(y_k, x_k \mid y_{1:k-1}) \, dx_k \]

\[ = \int g(y_k \mid x_k) \, p(x_k \mid y_{1:k-1}) \, dx_k \]

\[ = \int g(y_k \mid x_k) \, f(x_n \mid x_{n-1}) \, p(x_{k-1} \mid y_{1:k-1}) \, dx_{k-1} \]
We have seen that
\[ p(y_{1:n}) = \int p(y_{1:n} \mid x_{1:n}) p(x_{1:n}) \, dx_{1:n}. \]

We also have the following decomposition
\[
p(y_{1:n}) = p(y_1) \prod_{k=2}^{n} p(y_k \mid y_{1:k-1})
\]

where
\[
p(y_k \mid y_{1:k-1}) = \int p(y_k, x_k \mid y_{1:k-1}) \, dx_k
\]
\[= \int g(y_k \mid x_k) p(x_k \mid y_{1:k-1}) \, dx_k
\]
\[= \int g(y_k \mid x_k) f(x_n \mid x_{n-1}) p(x_{k-1} \mid y_{1:k-1}) \, dx_{k-1}
\]

We have “broken” an high dimensional integral into the product of lower dimensional integrals.
Assume given $n$ data, you are interested in estimating the marginal smoothing distributions $p(x_k|y_{1:n})$ for $k = 1, \ldots, n$. 
Assume given $n$ data, you are interested in estimating the marginal smoothing distributions $p(x_k | y_1:n)$ for $k = 1, \ldots, n$.

**Forward pass:** compute and store $p(x_k | y_1:k)$ and $p(x_{k+1} | y_1:k)$ for $k = 1, \ldots, n$ using the updating recursion.

**Remark:** Surprisingly, this recursion is almost never used for finite state-space HMM.
Assume given $n$ data, you are interested in estimating the marginal smoothing distributions $p(x_k | y_{1:n})$ for $k = 1, ..., n$.

**Forward pass:** compute and store $p(x_k | y_{1:k})$ and $p(x_{k+1} | y_{1:k})$ for $k = 1, ..., n$ using the updating recursion.

**Backward pass:** use for $k = n - 1, n - 2, ..., 1$ the following recursion

$$p(x_k | y_{1:n}) = \int \frac{f(x_{k+1} | x_k) p(x_k | y_{1:k})}{p(x_{k+1} | y_{1:k})} p(x_{k+1} | y_{1:n}) \, dx_{k+1}.$$
Assume given \( n \) data, you are interested in estimating the marginal smoothing distributions \( p(x_k | y_{1:n}) \) for \( k = 1, \ldots, n \).

**Forward pass:** compute and store \( p(x_k | y_{1:k}) \) and \( p(x_{k+1} | y_{1:k}) \) for \( k = 1, \ldots, n \) using the updating recursion.

**Backward pass:** use for \( k = n - 1, n - 2, \ldots, 1 \) the following recursion

\[
p(x_k | y_{1:n}) = \int \frac{f(x_{k+1} | x_k) p(x_k | y_{1:k})}{p(x_{k+1} | y_{1:k})} p(x_{k+1} | y_{1:n}) \, dx_{k+1}.
\]

Remark: Surprisingly, this recursion is almost never used for finite state-space HMM.
Proof.

\[
p(x_k | y_{1:n}) = \int p(x_k, x_{k+1} | y_{1:n}) \, dx_{k+1}
\]

\[
= \int p(x_k | x_{k+1}, y_{1:n}) \, dx_{k+1} \cdot p(x_{k+1} | y_{1:n}) \, dx_{k+1}
\]

\[
= \int p(x_k | x_{k+1}, y_{1:k}) \, dx_{k+1} \cdot p(x_{k+1} | y_{1:n}) \, dx_{k+1}
\]

\[
= \int \frac{f(x_{k+1} | x_k) \, p(x_k | y_{1:k})}{p(x_{k+1} | y_{1:k})} \, p(x_{k+1} | y_{1:n}) \, dx_{k+1}
\]
Two-Filter Smoothing

- An alternative approach consists of noting that

\[ p(x_k | y_{1:n}) = \frac{p(x_k | y_{1:k}) p(y_{k+1:n} | x_k)}{p(y_{k+1:n} | y_{1:k})} \]
Two-Filter Smoothing

- An alternative approach consists of noting that

\[
p(x_k | y_{1:n}) = \frac{p(x_k | y_{1:k}) p(y_{k+1:n} | x_k)}{p(y_{k+1:n} | y_{1:k})}
\]

- In this case, the smoothing distribution is the combination of the standard forward filter and the so-called backward information filter given by

\[
p(y_{k+1:n} | x_k) = \int p(y_{k+1:n}, x_{k+1} | x_k) \, dx_{k+1}
\]

\[
= \int p(y_{k+1:n} | x_{k+1}, x_k) f(x_{k+1} | x_k) \, dx_{k+1}
\]

\[
= \int p(y_{k+2:n} | x_{k+1}) g(y_{k+1} | x_{k+1}) f(x_{k+1} | x_k) \, dx_{k+1}
\]
Two-Filter Smoothing

- An alternative approach consists of noting that

\[
p(x_k \mid y_{1:n}) = \frac{p(x_k \mid y_{1:k}) p(y_{k+1:n} \mid x_k)}{p(y_{k+1:n} \mid y_{1:k})}
\]

- In this case, the smoothing distribution is the combination of the standard forward filter and the so-called backward information filter given by

\[
p(y_{k+1:n} \mid x_k) = \int p(y_{k+1:n}, x_{k+1} \mid x_k) \, dx_{k+1}
\]

\[
= \int p(y_{k+1:n} \mid x_{k+1}, x_k) f(x_{k+1} \mid x_k) \, dx_{k+1}
\]

\[
= \int p(y_{k+2:n} \mid x_{k+1}) g(y_{k+1} \mid x_{k+1}) f(x_{k+1} \mid x_k) \, dx_{k+1}
\]

- We can have \( \int p(y_{k+1:n} \mid x_k) \, dx_k = \infty \), this has led to numerous wrong algorithms in the literature.

Remark: The two-filter smoother is known as the forward-backward smoother for finite state-space HMM!
Two-Filter Smoothing

- An alternative approach consists of noting that

\[ p(x_k \mid y_{1:n}) = \frac{p(x_k \mid y_{1:k}) p(y_{k+1:n} \mid x_k)}{p(y_{k+1:n} \mid y_{1:k})} \]

- In this case, the smoothing distribution is the combination of the standard forward filter and the so-called backward information filter given by

\[ p(y_{k+1:n} \mid x_k) = \int p(y_{k+1:n}, x_{k+1} \mid x_k) \, dx_{k+1} \]

\[ = \int p(y_{k+1:n} \mid x_{k+1}, x_k) f(x_{k+1} \mid x_k) \, dx_{k+1} \]

\[ = \int p(y_{k+2:n} \mid x_{k+1}) g(y_{k+1} \mid x_{k+1}) f(x_{k+1} \mid x_k) \, dx_{k+1} \]

- We can have \( \int p(y_{k+1:n} \mid x_k) \, dx_k = \infty \), this has led to numerous wrong algorithms in the literature.

- Remark: The two-filter smoother is known as the forward-backward smoother for finite state-space HMM!
In most applications of interest, we have the initial distribution $\mu(x_1)$, the transition density $f(x_k|x_{k-1})$ and observation density $g(y_k|x_k)$ dependent on some hyperparameters $\theta$ and we write $\mu_\theta(x_1)$, $f_\theta(x_k|x_{k-1})$ and $g_\theta(y_k|x_k)$.
In most applications of interest, we have the initial distribution $\mu(x_1)$, the transition density $f(x_k | x_{k-1})$ and observation density $g(y_k | x_k)$ dependent on some hyperparameters $\theta$ and we write $\mu_\theta(x_1)$, $f_\theta(x_k | x_{k-1})$ and $g_\theta(y_k | x_k)$.

For example, in the tracking example, the variances of both the dynamic noise and observation noise might be unknown.
In most applications of interest, we have the initial distribution $\mu (x_1)$, the transition density $f (x_k | x_{k-1})$ and observation density $g (y_k | x_k)$ dependent on some hyperparameters $\theta$ and we write $\mu_\theta (x_1)$, $f_\theta (x_k | x_{k-1})$ and $g_\theta (y_k | x_k)$.

For example, in the tracking example, the variances of both the dynamic noise and observation noise might be unknown.

In a *full Bayesian framework*, we set a prior $p (\theta)$ on $\theta$. If we define the extended state $Z_k = (Z^1_k, Z^2_k) = (\theta, X_k)$, we can rewrite everything as a standard HMM where

$$Z_1 \sim p (z^1_1) \mu_{z^1_1} (z^2_1),$$

$$Z_k \mid (Z_{k-1} = z_{k-1}) \sim \delta_{z^1_{k-1}} (z^1_k) f_{z^1_k} (z^2_k \mid z^2_{k-1}),$$

$$Y_k \mid (Z_k = z_k) \sim g_{z^1_k} (y_k \mid z^2_k).$$
In most applications of interest, we have the initial distribution \( \mu (x_1) \), the transition density \( f (x_k | x_{k-1}) \) and observation density \( g (y_k | x_k) \) dependent on some hyperparameters \( \theta \) and we write \( \mu_\theta (x_1) \), \( f_\theta (x_k | x_{k-1}) \) and \( g_\theta (y_k | x_k) \).

For example, in the tracking example, the variances of both the dynamic noise and observation noise might be unknown.

In a full Bayesian framework, we set a prior \( p (\theta) \) on \( \theta \). If we define the extended state \( Z_k = (Z^1_k, Z^2_k) = (\theta, X_k) \), we can rewrite everything as a standard HMM where

\[
Z_1 \sim p (z^1_1) \mu_{z^1_1} (z^2_1),
\]

\[
Z_k | (Z_{k-1} = z_{k-1}) \sim \delta_{z^1_{k-1}} (z^1_k) f_{z^1_k} (z^2_k | z^2_{k-1}),
\]

\[
Y_k | (Z_k = z_k) \sim g_{z^2_k} (y_k | z^2_k).
\]

Conceptually, this solution is correct. Practically, the degeneracy of the transition kernel of \( \{Z_k\}_{k \geq 1} \) can cause serious numerical problems for approximation methods.
Standard approaches for parameter estimation consists of computing the Maximum Likelihood (ML) estimate

\[ \theta_{ML} = \arg \max \log p_{\theta}(y_{1:n}) \]
Standard approaches for parameter estimation consists of computing the Maximum Likelihood (ML) estimate

\[ \theta_{ML} = \arg \max \log p_\theta(y_{1:n}) \]

The likelihood function can be multimodal and there is no guarantee to find its global optimum.
Standard approaches for parameter estimation consists of computing the Maximum Likelihood (ML) estimate

\[ \theta_{ML} = \arg \max \log p_{\theta} (y_{1:n}) \]

The likelihood function can be multimodal and there is no guarantee to find its global optimum.

Standard (stochastic) gradient algorithms can be used based for example on Fisher’s identity

\[ \nabla \log p_{\theta} (y_{1:n}) = \int \nabla \log p_{\theta} (x_{1:n}, y_{1:n}) \cdot p_{\theta} (x_{1:n} | y_{1:n}) \, dx_{1:n}. \]
Standard approaches for parameter estimation consists of computing the Maximum Likelihood (ML) estimate

$$\theta_{ML} = \arg \max \log p_\theta (y_{1:n})$$

The likelihood function can be multimodal and there is no guarantee to find its global optimum.

Standard (stochastic) gradient algorithms can be used based for example on Fisher’s identity

$$\nabla \log p_\theta (y_{1:n}) = \int \nabla \log p_\theta (x_{1:n}, y_{1:n}) \cdot p_\theta (x_{1:n} | y_{1:n}) \, dx_{1:n}.$$
We can use as an alternative the popular Expectation-Maximization algorithm

\[ \theta^{(i)} = \arg \max \quad Q \left( \theta^{(i)}, \theta \right) \]

where

\[
Q \left( \theta^{(i)}, \theta \right) = \int \log p_{\theta} (x_{1:n}, y_{1:n}) \cdot p_{\theta^{(i-1)}} (x_{1:n} \mid y_{1:n}) \, dx_{1:n} \\
= \int \log \left( \mu (x_1) g (y_1 \mid x_1) \right) \cdot p_{\theta^{(i-1)}} (x_1 \mid y_{1:n}) \, dx_1 \\
+ \sum_{k=2}^{n} \int \log \left( f (x_k \mid x_{k-1}) g (y_k \mid x_k) \right) \cdot p_{\theta^{(i-1)}} (x_{k-1:k} \mid y_{1:n}) \, dx_{k-1:k}.
\]
We can use as an alternative the popular Expectation-Maximization algorithm

\[ \theta^{(i)} = \arg \max \ Q \left( \theta^{(i)}, \theta \right) \]

where

\[
Q \left( \theta^{(i)}, \theta \right) = \int \log p_{\theta} (x_{1:n}, y_{1:n}) \cdot p_{\theta^{(i-1)}} (x_{1:n} | y_{1:n}) \, dx_{1:n}
\]

\[
= \int \log \left( \mu (x_{1}) \cdot g (y_{1} \mid x_{1}) \right) \cdot p_{\theta^{(i-1)}} (x_{1} \mid y_{1:n}) \, dx_{1}
\]

\[
+ \sum_{k=2}^{n} \int \log \left( f (x_{k} \mid x_{k-1}) \cdot g (y_{k} \mid x_{k}) \right) \cdot p_{\theta^{(i-1)}} (x_{k-1:k} \mid y_{1:n}) \, dx_{k-1:k}.
\]

Implementing this algorithm requires being able to compute expectations with respect to the smoothing distributions

\[ p_{\theta^{(i-1)}} (x_{k-1:k} \mid y_{1:n}) \].
Closed-form Inference in HMM

- We have closed-form solutions for

\[ E = f_1, \ldots, f_p \] as all integrals are finite sums

Linear Gaussian models; all the posterior distributions are Gaussian; e.g. the celebrated Kalman filter.

A whole reverse engineering literature exists for closed-form solutions in alternative cases...

In many cases of interest, it is impossible to compute the solution in closed-form and we need approximations,
We have closed-form solutions for

- Finite state-space HMM; i.e. \( E = \{e_1, \ldots, e_p\} \) as all integrals are becoming finite sums
Closed-form Inference in HMM

- We have closed-form solutions for
  - Finite state-space HMM; i.e. \( E = \{ e_1, ..., e_p \} \) as all integrals are becoming finite sums
  - Linear Gaussian models; all the posterior distributions are Gaussian; e.g. the celebrated Kalman filter.
Closed-form Inference in HMM

- We have closed-form solutions for
  - Finite state-space HMM; i.e. $E = \{e_1, \ldots, e_p\}$ as all integrals are becoming finite sums
  - Linear Gaussian models; all the posterior distributions are Gaussian; e.g. the celebrated Kalman filter.
  - A whole reverse engineering literature exists for closed-form solutions in alternative cases...
Closed-form Inference in HMM

- We have closed-form solutions for
  - Finite state-space HMM; i.e. \( E = \{e_1, ..., e_p\} \) as all integrals are becoming finite sums
  - Linear Gaussian models; all the posterior distributions are Gaussian; e.g. the celebrated Kalman filter.
  - A whole reverse engineering literature exists for closed-form solutions in alternative cases...

- In many cases of interest, it is impossible to compute the solution in closed-form and we need approximations,
Aim of the Course

- Present generic numerical approximation techniques to be able to perform optimal state and parameter estimation in general non-linear non-Gaussian models.
Aim of the Course

- Present generic numerical approximation techniques to be able to perform optimal state and parameter estimation in general non-linear non-Gaussian models.
- These methods are in some sense ‘asymptotically consistent’; i.e. if my computational efforts increase without bounds, then the approximations will converge towards the ground thruth.
Aim of the Course

- Present generic numerical approximation techniques to be able to perform optimal state and parameter estimation in general non-linear non-Gaussian models.

- These methods are in some sense ‘asymptotically consistent’; i.e. if my computational efforts increase without bounds, then the approximations will converge towards the ground thruth.

- Most approximation methods are not ‘asymptotically consistent’ and they might work better for a fixed computational complexity.
Gaussian approximations: Extended Kalman filter, Unscented Kalman filter.
Standard Approximations for Filtering Distributions

- Gaussian approximations: Extended Kalman filter, Unscented Kalman filter.
- Gaussian sum approximations.
Gaussian approximations: Extended Kalman filter, Unscented Kalman filter.

Gaussian sum approximations.

Projection filters, Variational approximations.
Standard Approximations for Filtering Distributions

- Gaussian approximations: Extended Kalman filter, Unscented Kalman filter.
- Gaussian sum approximations.
- Projection filters, Variational approximations.
- Simple discretization of the state-space.
Standard Approximations for Filtering Distributions

- Gaussian approximations: Extended Kalman filter, Unscented Kalman filter.
- Gaussian sum approximations.
- Projection filters, Variational approximations.
- Simple discretization of the state-space.
- Analytical methods work in simple cases but are not reliable and it is difficult to diagnose when they fail.
Gaussian approximations: Extended Kalman filter, Unscented Kalman filter.

Gaussian sum approximations.

Projection filters, Variational approximations.

Simple discretization of the state-space.

Analytical methods work in simple cases but are not reliable and it is difficult to diagnose when they fail.

Standard discretization of the space is expensive and difficult to implement in high-dimensional scenarios.
At the beginning of the 90’s, the optimal filtering area was considered virtually dead; there had not been any significant progress for years then...
At the beginning of the 90’s, the optimal filtering area was considered virtually dead; there had not been any significant progress for years then...

At the beginning of the 90’s, the optimal filtering area was considered virtually dead; there had not been any significant progress for years then...


This article introduces a simple method which relies neither on a functional approximation nor a deterministic grid.

This paper was ignored by most researchers for a few years until its rediscovery in 1996 by Isard & Blake in the field of computer vision.
At the beginning of the 90’s, the optimal filtering area was considered virtually dead; there had not been any significant progress for years then...


This article introduces a simple method which relies neither on a functional approximation nor a deterministic grid.

This paper was ignored by most researchers for a few years until its rediscovery in 1996 by Isard & Blake in the field of computer vision.