Lecture 7
Models of Sequential Data

University of Amsterdam
Modelling sequential data

Graphical models of sequential data are just graphical models, but:

- The assumption that all examples are drawn i.i.d. from the same distribution does not hold for individual observations anymore
- It does still hold for individual sequences, though
- The data can be sequential over time, space, ...

Example
Speech recognition, weather modelling, DNA sequence modelling
Adapting to the data

Graphical Models

- Give us a way to visualise factorisation of probabilities
- Give us powerful algorithms automatically find the optimal way to compute marginal probabilities, conditional probabilities, . . .
- Therefore allow for efficient algorithms for training
- Cannot, in general, adapt the factorisation to the data

Models of sequential data slightly relax this:

- We now set the general structure of the model
- The exact structure is automatically adapted to the length of the sequence
1. Models of data sequences
   - Independent observations
   - Markov Models

2. Hidden Markov Models
   - Inference: The Baum-Welch Algorithm
   - The Viterbi Algorithm
   - Training
   - Extensions to HMM
   - Dealing with very unlikely events

3. Linear Dynamical System
   - Representation
   - The Kalman Filter
   - Extensions
1. **Models of data sequences**
   - Independent observations
   - Markov Models

2. **Hidden Markov Models**
   - Inference: The Baum-Welch Algorithm
   - The Viterbi Algorithm
   - Training
   - Extensions to HMM
   - Dealing with very unlikely events

3. **Linear Dynamical System**
   - Representation
   - The Kalman Filter
   - Extensions
Assume independence

Simplest solution:

- Assume that all observations are independent
- Problem: fails to exploit sequential patterns in the data

Example: Rainy days

The only information we can obtain from this model is the frequency of rainy days.
Markov Models

If we do not assume any independence, we can factorise the distribution as:

\[ p(x_1, \ldots, x_N) = \prod_{n=1}^{N} p(x_n|x_1, \ldots, x_{n-1}) \]  

(1)

However:

- In general the most recent observations are more likely to affect the current observation.
- If we assume that only the single most recent observation affects the current observation, we obtain the **first order Markov chain**

\[ p(x_1, \ldots, x_N) = p(x_1) \prod_{n=2}^{N} p(x_n|x_{n-1}) \]  

(2)
In most applications, we constrain $p(x_n|x_{n-1})$ to be equal over the length of the chain: **homogeneous markov chain**

Example

This was used to model character occurances in text
In most applications, we constrain $p(x_n|x_{n-1})$ to be equal over the length of the chain: **homogeneous markov chain**

**Example**

This was used to model character occurrences in text.
Higher order Markov chains

First order Markov chains are still very restrictive:

- Modelling trends requires more information from the past
- Second order Markov chain:

\[
p(x_1, \ldots, x_N) = p(x_1)p(x_2|x_1) \prod_{n=3} p(x_n|x_{n-1}, x_{n-2})
\] (3)
Higher order Markov chains

There is a high penalty for the extra flexibility:

- Imagine $K$ discrete states
  - First order chain: $K \times (K - 1)$ independent parameters
  - Second order chain: $K^2 \times (K - 1)$ parameters
  - In general, $M$th order: $K^{M-1} \times (K - 1)$ parameters

- The complexity grows exponentially with $M$

This becomes impractical very quickly.
1 Models of data sequences
   - Independent observations
   - Markov Models

2 Hidden Markov Models
   - Inference: The Baum-Welch Algorithm
   - The Viterbi Algorithm
   - Training
   - Extensions to HMM
   - Dealing with very unlikely events

3 Linear Dynamical System
   - Representation
   - The Kalman Filter
   - Extensions
Hidden Markov Models

How can we keep a longer memory and still restrict the number of parameters?

- Create a latent variable model: the probability of observations depends on the latent variable (see Mixtures of Gaussians)
- The latent variables are part of a latent Markov chain
- D-separation shows that all observations are now dependent
Hidden Markov Models are often viewed as Finite State Machines

- The arrows indicate the transition probabilities, typically denoted by the matrix $A$
- Emission probabilities are associated with each state
- Not depicted here, is the probability of starting a sequence in a given state $k$, typically denoted by $\pi_k$
We can then “unroll” the FSA over time to form a lattice.
Inference in the HMM

Inference allows us to answer the questions:

- What is the probability of each state, for each observation?

**Example**

In speech recognition, what is the probability that a certain phoneme corresponds to a certain audio observation?

- What is the probability of seeing an observation sequence, according to the model?

**Example**

In elderly care, it is useful to monitor how well patients perform activities at home. It is very hard to have a model of the problems they may have, but it is possible to detect when their activities become very unlikely according to the model.
Inference in the HMM

Inference allows us to answer the questions:

- What is the probability of each state, for each observation?

Example

In speech recognition, what is the probability that a certain phoneme corresponds to a certain audio observation?

- What is the probability of seeing an observation sequence, according to the model?

Example

In elderly care, it is useful to monitor how well patients perform activities at home. It is very hard to have a model of the problems they may have, but it is possible to detect when their activities become very unlikely according to the model.
The Forward-Backward Algorithm

HMM have been around for much longer than graphical models

- Specific versions of the sum-product and max-product were developed for the HMM
- The sum-product rule for HMM is known as the Forward-Backward or Baum-Welch algorithm
- It consists of a single forward and a single backward pass through the chain
The Forward pass

The message leaving the first node is

\[ \mu_{z_1 \rightarrow \psi_1} = p(x_1 | z_1) p(z_1) = p(x_1, z_1) \]

The message leaving the second node is

\[ \mu_{z_2 \rightarrow \psi_3} = p(x_2 | z_2) \sum_{z_1} p(x_1, z_1) p(z_2 | z_1) \]

\[ = p(x_1, x_2, z_2) \]

In general:

\[ \mu_{\psi_{n-1} \rightarrow z_n} = p(x_1, \ldots, x_n, z_n) \]

\[ = \alpha(z_n) \]
The Backward pass

The message entering the penultimate node is

$$\mu_{\psi_{N-1} \rightarrow z_{N-1}} = \sum_{z_N} p(z_N | z_{N-1}) p(x_N | z_N)$$

$$= p(x_N | z_{N-1})$$

The message entering the antepenultimate node is

$$\mu_{\psi_{N-2} \rightarrow z_{N-2}} = \sum_{z_{N-1}} p(z_{N-1} | z_{N-2}) p(x_{N-1} | z_{N-1}) p(x_N | z_{N-1})$$

$$= p(x_{N-1}, x_N | z_{N-2})$$

In general, the “backward” message is:

$$\mu_{\psi_n \rightarrow z_n} = p(x_{n+1}, \ldots, x_N | z_n)$$

$$= \beta(z_n)$$
The Viterbi Algorithm

The sequence of hidden states often has a real, physical explanation.

**Example**

In speech recognition, the sequence of phonemes for a sequence of sound observations.

It is therefore useful to have the most likely single sequence of observations, rather than the sequence of most likely observations.

- We obtain this using the max-sum algorithm
- For the HMM, this is called the Viterbi Algorithm
Training of the HMM

We can train the HMM using the EM algorithm, optimising

\[ Q(\theta, \theta^{\text{old}}) = \sum_z p(Z|X, \theta^{\text{old}}) \ln p(X, Z|\theta) \]  

(4)

We introduce

\[ \gamma(z_{nk}) = p(z_{nk}|X, \theta^{\text{old}}) \]  

(5)

\[ \xi(z_{n-1j}, z_{nk}) = p(z_{n-1j}, z_{nk}|X, \theta^{\text{old}}) \]  

(6)

so that the expectation of the complete log-likelihood becomes

\[ Q(\theta, \theta^{\text{old}}) = \sum_k \gamma(z_{1k}) \ln \pi_k + \sum_{n=2}^{N} \sum_{j=1}^{K} \sum_{k=1}^{K} \xi(z_{n-1j}, z_{nk}) \ln A_{jk} + \]

\[ \sum_{n=1}^{N} \sum_{k=1}^{K} \gamma(z_{nk}) \ln p(x_n|z_k) \]
The responsibilities $\gamma(z_{nk})$ can be computed from the forward and backward probabilities:

$$
\gamma(z_n) = p(z_n|\mathbf{X}, \theta^{\text{old}}) \\
= \frac{\alpha(z_n)\beta(z_n)}{p(\mathbf{X})}
$$

and similarly for the pairwise responsibilities:

$$
\xi(z_{n-1}, z_n) = p(z_{n-1}, z_n|\mathbf{X}) \\
= \frac{\alpha(z_{n-1})p(z_n|z_{n-1})p(\mathbf{x}_n|z_n)\beta(z_n)}{p(\mathbf{X})}
$$
The M step

- Taking the first derivative of $Q(\theta, \theta^{old})$ with respect to the parameters and setting it to zero gives us the update rules:

$$\pi_k = \frac{\gamma(z_{1k})}{\sum_{j=1}^{K} \gamma(z_{1j})}$$ \hspace{2cm} (11)

$$A_{jk} = \frac{\sum_{n=2}^{N} \xi(z_{n-1,j}, z_{nk})}{\sum_{l=2}^{K} \sum_{n=2}^{N} \xi(z_{n-1,j}, z_{nl})}$$ \hspace{2cm} (12)

- optimising the emission probabilities depends on the specific form of these, and is identical to a mixture model.
The M-Step

Example: Gaussian observations

If the observations have a Gaussian distribution, the maximum likelihood parameters are given by:

\[
\mu_k = \frac{\sum_{n=1}^{N} \gamma(z_{nk}) x_n}{\sum_{n=1}^{N} \gamma(z_{nk})} \tag{13}
\]

\[
\Sigma_k = \frac{\sum_{n=1}^{N} \gamma(z_{nk}) (x_n - \mu_k)(x_n - \mu_k)^\top}{\sum_{n=1}^{N} \gamma(z_{nk})} \tag{14}
\]

(See Gaussian Mixture Models)
Extensions

The basic HMM framework can be extended in many ways.

- Autoregressive HMM
- Input-output HMM
- Factorial HMM
- Explicit time model (Hidden semi-Markov model)
The basic HMM framework can be extended in many ways

- Autoregressive HMM
- Input-output HMM
- Factorial HMM
- Explicit time model (Hidden semi-Markov model)
The basic HMM framework can be extended in many ways

- Autoregressive HMM
- Input-output HMM
- Factorial HMM
- Explicit time model (Hidden semi-Markov model)
Extensions

The basic HMM framework can be extended in many ways:

- Autoregressive HMM
- Input-output HMM
- Factorial HMM
- Explicit time model (Hidden semi-Markov model)

\[ p(z_n, \ldots, z_{n+t} = z) = A_{zz}(1 - A_{zz}) \]
Dealing with very low probabilities

The probability of a chain of observations quickly becomes small.

**Example: Throwing a coin**

If we throw a fair die 1000 times, what is the probability of observing the exact sequence that we got? \((\frac{1}{6})^{1000}\)

We cannot easily represent such low probabilities.
Dealing with very low probabilities

The probability of a chain of observations quickly becomes small.

**Example: Throwing a coin**

If we throw a fair die 1000 times, what is the probability of observing the exact sequence that we got? \( \left( \frac{1}{6} \right)^{1000} \)

We cannot easily represent such low probabilities.
Dealing with very low probabilities

The probability of a chain of observations quickly becomes small.

**Example: Throwing a coin**

If we throw a fair die 1000 times, what is the probability of observing the exact sequence that we got? \((\frac{1}{6})^{1000}\)

We cannot easily represent such low probabilities.
Probabilities in chains

Problem:

- (Double-precision) Floating point arithmetic cannot represent probabilities associated with long chains

Solution:

1. Rescale probabilities along the chain (and keep track of the scaling factors)
   - Unwieldy
   - Not a general solution (e.g., high-dimensional Gaussian)
2. Use log-probabilities
Problem:

- (Double-precision) Floating point arithmetic cannot represent probabilities associated with long chains

Solution:

1. Rescale probabilities along the chain (and keep track of the scaling factors)
   - Unwieldy
   - Not a general solution (e.g., high-dimensional Gaussian)
2. Use log-probabilities
Problem:

- (Double-precision) Floating point arithmetic cannot represent probabilities associated with long chains

Solution:

1. Rescale probabilities along the chain (and keep track of the scaling factors)
   - Unwieldy
   - Not a general solution (e.g., high-dimensional Gaussian)
2. Use log-probabilities
### Using log-probabilities

<table>
<thead>
<tr>
<th>Range</th>
<th>Probabilities</th>
<th>log-probabilities</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p(A \land B)$</td>
<td>$p(A) \cdot p(B</td>
<td>A)$</td>
</tr>
<tr>
<td>$p(A</td>
<td>B)$</td>
<td>$\frac{p(A,B)}{p(B)}$</td>
</tr>
<tr>
<td>$p(A \lor B)$</td>
<td>$p(A) + P(B</td>
<td>A)$</td>
</tr>
</tbody>
</table>

Dealing with very unlikely events

log-probabilities: range $[\log \infty, 0]$
Using log-probabilities

<table>
<thead>
<tr>
<th>Range</th>
<th>Probabilities</th>
<th>log-probabilities</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p(A \land B)$</td>
<td>$p(A) \cdot p(B</td>
<td>A)$</td>
</tr>
<tr>
<td>$p(A</td>
<td>B)$</td>
<td>$\frac{p(A,B)}{p(B)}$</td>
</tr>
<tr>
<td>$p(A \lor B)$</td>
<td>$p(A) + P(B</td>
<td>A)$</td>
</tr>
</tbody>
</table>
Using log-probabilities

<table>
<thead>
<tr>
<th>Range</th>
<th>Probabilities</th>
<th>log-probabilities</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p(A \land B)$</td>
<td>$p(A) \cdot p(B</td>
<td>A)$</td>
</tr>
<tr>
<td>$p(A \mid B)$</td>
<td>$\frac{p(A,B)}{p(B)}$</td>
<td>$\log p(A, B) - \log p(B)$</td>
</tr>
<tr>
<td>$p(A \lor B)$</td>
<td>$p(A) + P(B</td>
<td>A)$</td>
</tr>
</tbody>
</table>

Dealing with very unlikely events

Using log-probabilities

**Range**

- $[0, 1]$
- $[\infty, 0]$
### Using log-probabilities

<table>
<thead>
<tr>
<th>Range</th>
<th>Probabilities</th>
<th>log-probabilities</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p(A \land B)$</td>
<td>$p(A) \cdot p(B</td>
<td>A)$</td>
</tr>
<tr>
<td>$p(A</td>
<td>B)$</td>
<td>$\frac{p(A,B)}{p(B)}$</td>
</tr>
<tr>
<td>$p(A \lor B)$</td>
<td>$p(A) + P(B</td>
<td>A)$</td>
</tr>
</tbody>
</table>

- $[0, 1]$ range for probabilities
- $[\infty, 0]$ range for log-probabilities
## Using log-probabilities

<table>
<thead>
<tr>
<th>Range</th>
<th>Probabilities</th>
<th>log-probabilities</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p(A \land B)$</td>
<td>$p(A) \cdot p(B</td>
<td>A)$</td>
</tr>
<tr>
<td>$p(A</td>
<td>B)$</td>
<td>$\frac{p(A,B)}{p(B)}$</td>
</tr>
<tr>
<td>$p(A \lor B)$</td>
<td>$p(A) + P(B</td>
<td>A)$</td>
</tr>
</tbody>
</table>
Log of sum of exponentials

\[
\log(a + b) \text{ when given } \log(a) \text{ and } \log(b)
\]

\[
\log(a + b) = \log(\exp \log a + \exp \log b)
\]

\[
= \log \left( \exp \log a \left[ 1 + \frac{\exp \log b}{\exp \log a} \right] \right)
\]

\[
= \log a + \log(1 + \exp(\log b - \log a))
\]

Define a new function:

\[
lse(a, b) = \log(\exp a + \exp b)
\]
Dealing with very unlikely events

Log of sum of exponentials

\[ \log(a + b) \text{ when given } \log(a) \text{ and } \log(b) \]

\[
\log(a + b) = \log(\exp \log a + \exp \log b)
\]

\[
= \log \left( \exp \log a \left[ 1 + \frac{\exp \log b}{\exp \log a} \right] \right)
\]

\[
= \log a + \log(1 + \exp(\log b - \log a))
\]

Define a new function:

\[ \text{lse}(a, b) = \log(\exp a + \exp b) \]
Dealing with very unlikely events

\[ y = \log(1 + e^x) \]
Dealing with very unlikely events

Dealing with log-probabilities

**log-of-sum-of-exponentials — lse**

\[
\log(a + b) = \log a + \log(1 + \exp(\log b - \log a))
\]

Some implementation details:

- Keep the exponent small (choose \(a > b\))
- \(\log(1 + x)\) is a special function: \texttt{log1p}
- Handle infinity as a special case
- For summations:

\[
a + b + c + \ldots = ((a + b) + c) + \ldots \\
lse(a, b, c, \ldots) = lse(lse(lse(a, b), c), \ldots)
\]
Dealing with log-probabilities

**log-of-sum-of-exponentials — lse**

\[
\log(a + b) = \log a + \log(1 + \exp(\log b - \log a))
\]

Some implementation details:

- Keep the exponent small (choose \( a > b \))
- \( \log(1 + x) \) is a special function: \( \log1p \)
- Handle infinity as a special case
- For summations:

\[
a + b + c + \ldots = ((a + b) + c) + \ldots
\]

\[
lse(a, b, c, \ldots) = lse(lse(lse(a, b), c), \ldots)
\]
Dealing with log-probabilities

**log-of-sum-of-exponentials — lse**

\[
\log(a + b) = \log a + \log(1 + \exp(\log b - \log a))
\]

Some implementation details:

- Keep the exponent small (choose \(a > b\))
- \(\log(1 + x)\) is a special function: \(\log1p\)
- Handle infinity as a special case
- For summations:

  \[
  a + b + c + \ldots = ((a + b) + c) + \ldots
  \]

  \[
  \text{lse}(a, b, c, \ldots) = \text{lse}(\text{lse}(\text{lse}(a, b), c), \ldots)
  \]
Dealing with log-probabilities

\[ \log(a + b) = \log a + \log(1 + \exp(\log b - \log a)) \]

Some implementation details:

- Keep the exponent small (choose \( a > b \))
- \( \log(1 + x) \) is a special function: \( \log 1p \)
- Handle infinity as a special case
- For summations:

\[ a + b + c + \ldots = ((a + b) + c) + \ldots \]
\[ \text{lse}(a, b, c, \ldots) = \text{lse}(\text{lse}(\text{lse}(a, b), c), \ldots) \]
Dealing with log-probabilities

**log-of-sum-of-exponentials — lse**

\[
\log(a + b) = \log a + \log(1 + \exp(\log b - \log a))
\]

Some implementation details:
- Keep the exponent small (choose \(a > b\))
- \(\log(1 + x)\) is a special function: \texttt{log1p}
- Handle infinity as a special case
- For summations:
  \[
  a + b + c + \ldots = ((a + b) + c) + \ldots
  
  \text{lse}(a, b, c, \ldots) = \text{lse}(\text{lse}(\text{lse}(a, b), c), \ldots)
  \]
1. Models of data sequences
   - Independent observations
   - Markov Models

2. Hidden Markov Models
   - Inference: The Baum-Welch Algorithm
   - The Viterbi Algorithm
   - Training
   - Extensions to HMM
   - Dealing with very unlikely events

3. Linear Dynamical System
   - Representation
   - The Kalman Filter
   - Extensions
In the HMM, we assume that the latent variables are discrete.

- The observations (emission probabilities) can have any form (Bernoulli, Gaussian, Mixture of Gaussians, etc.)
- This is not a limitation of the graphical model per se.

One well-known model that allows for continuous latent variables is the Linear Dynamical System.
Linear-Gaussian system

\[ \mathcal{N}(x; \mu, \Lambda) \]
Linear-Gaussian system

\[ N(x; \mu, \Lambda) \]
Linear-Gaussian system

\[ \mathcal{N}(x; \mu, \Lambda) \]

\[ x_i \]

\[ Ax_i + b \]
Linear-Gaussian system

\[ \mathcal{N}(x; \mu, \Lambda) \]

\[ x_i \]

\[ \mathcal{N}(y; Ax_i + b, L) \]
Linear-Gaussian system

\[ N(x; \mu, \Lambda) \]

\[ x_i \]

\[ Ax_i + b \]

\[ N(y; A\mu + b, L + AA^\top) \]

\[ y_i \]
Linear-Gaussian system

\[ \mathcal{N}(x; \mu, \Lambda) \]

\[ Ax_i + b \]

\[ \mathcal{N}(y; A\mu + b, L + A\Lambda A^T) \]

\[ x_i \]

\[ y_i \]
Linear-Gaussian system
Linear-Gaussian system
Linear-Gaussian system
The mean of the next latent variable is a linear function of the previous latent variable:

\[ p(z_n|z_{n-1}) = \mathcal{N}(z_n|Az_{n-1}, \Gamma) \]  \hspace{1cm} (15)

- The current observation has a Gaussian distribution centred around the current latent variable

\[ p(x_n|z_n) = \mathcal{N}(x_n|Cz_n, \Sigma) \]  \hspace{1cm} (16)

- Assuming Gaussian distributions ensures that the complexity of the messages does not increase along the chain
- The prior \( p(z_0) \) can be a mixture of \( K \) Gaussians, however
The mean of the next latent variable is a linear function of the previous latent variable:

$$p(z_n|z_{n-1}) = \mathcal{N}(z_n|Az_{n-1} + Bu, \Gamma)$$  \hspace{1cm} (15)$$

The current observation has a Gaussian distribution centred around the current latent variable

$$p(x_n|z_n) = \mathcal{N}(x_n|Cz_n, \Sigma)$$ \hspace{1cm} (16)$$

Assuming Gaussian distributions ensures that the complexity of the messages does not increase along the chain

The prior $p(z_0)$ can be a mixture of $K$ Gaussians, however
The mean of the next latent variable is a linear function of the previous latent variable:

$$p(z_n|z_{n-1}) = \mathcal{N}(z_n|Az_{n-1} + Bu, \Gamma)$$  \hspace{1cm} (15)

The current observation has a Gaussian distribution centred around the current latent variable

$$p(x_n|z_n) = \mathcal{N}(x_n|Cz_n, \Sigma)$$  \hspace{1cm} (16)

Assuming Gaussian distributions ensures that the complexity of the messages does not increase along the chain.

The prior $p(z_0)$ can be a mixture of $K$ Gaussians, however...
The mean of the next latent variable is a linear function of the previous latent variable:

$$p(z_n|z_{n-1}) = \mathcal{N}(z_n|Az_{n-1} + Bu, \Gamma)$$  \hspace{1cm} (15)

The current observation has a Gaussian distribution centred around the current latent variable

$$p(x_n|z_n) = \mathcal{N}(x_n|Cz_n, \Sigma)$$ \hspace{1cm} (16)

Assuming Gaussian distributions ensures that the complexity of the messages does not increase along the chain

The prior $p(z_0)$ can be a mixture of $K$ Gaussians, however
Again, this is a model that was developed long before graphical models

- If only the forward pass is done, it is called the *Kalman Filter*
- It was developed for optimal tracking of rockets and satellites
- If the backwards pass is performed as well, it is called the *Kalman Smoother*. The corresponding equations are called the *Rauch-Tung-Striebel (RTS)* equations.
Kalman Filter Updates

- Distribution over $z_{n-1}$
- Prediction over $z_n$
- Probability of the observation $x_n$ given $z_n$, and updated distribution over $z_n$
Kalman Filter Updates

- Distribution over $z_{n-1}$
- Prediction over $z_n$
- Probability of the observation $x_n$ given $z_n$, and updated distribution over $z_n$
Kalman Filter Updates

- Distribution over $z_{n-1}$
- Prediction over $z_n$
- Probability of the observation $x_n$ given $z_n$, and updated distribution over $z_n$
Learning in the LDS

The Kalman filter is often used for tracking

- The matrices $A$ and $C$ are then known *a priori* — e.g.

$$z = \begin{bmatrix} x \\ y \\ v_x \\ v_y \end{bmatrix}$$

$$x = \begin{bmatrix} x \\ y \end{bmatrix}$$

$$A = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$  \hspace{1cm} (17)

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$  \hspace{1cm} (18)

- Covariance matrices $\Gamma$ and $\Sigma$ must be estimated

...Sometimes process must be learnt as well
As always, we can use EM to learn the model parameters

- The latent variables are continuous, posterior PDF is a Gaussian
- The expectation of the complete log-likelihood can be optimised in closed form
  - With respect to the distribution parameters \((\mu, \Sigma, \Gamma)\)
  - With respect to the deterministic model parameters \((A, C)\)
Sometimes, Gaussian distributions are too limited for the task at hand.
Sometimes, Gaussian distributions are too limited for the task at hand.
Sometimes, Gaussian distributions are too limited for the task at hand.
Particle filters

Sometimes, Gaussian distributions are too limited for the task at hand

- The complexity of the distributions is then exponential in sequence length
- Approximations are required
- Lecture 11: sampling
Switching Linear Dynamical System

- Additional categorical variable $s$
- Switch between multiple processes
- Exact inference is intractable!
1 Models of data sequences
   - Independent observations
   - Markov Models

2 Hidden Markov Models
   - Inference: The Baum-Welch Algorithm
   - The Viterbi Algorithm
   - Training
   - Extensions to HMM
   - Dealing with very unlikely events

3 Linear Dynamical System
   - Representation
   - The Kalman Filter
   - Extensions
Wrap up

Today, we’ve looked at models of sequential data:

- Markov Models (Bishop, p. 605-610)
- HMM (Bishop, p. 610-630)
- Introduction to the LDS (Bishop, p. 635-637)

Exercise:
- EM for mixtures of Bernoulli distributions

Lab:
- The EM algorithm for GMM