Temporal probability models

Chapter 15, Sections 1–5
Outline

♦ Time and uncertainty
♦ Inference: filtering, prediction, smoothing
♦ Hidden Markov models
♦ Dynamic Bayesian networks
Time and uncertainty

The world changes; we need to track and predict it

Diabetes management vs vehicle diagnosis

Basic idea: copy state and evidence variables for each time step

\[ X_t = \text{set of unobservable state variables at time } t \]
  \[ \text{e.g., } \text{BloodSugar}_t, \text{StomachContents}_t, \text{etc.} \]

\[ E_t = \text{set of observable evidence variables at time } t \]
  \[ \text{e.g., } \text{MeasuredBloodSugar}_t, \text{PulseRate}_t, \text{FoodEaten}_t \]

This assumes **discrete time**; step size depends on problem

Notation: \( X_{a:b} = X_a, X_{a+1}, \ldots, X_{b-1}, X_b \)
Markov processes (Markov chains)

Construct a Bayes net from these variables: parents? CPTs?
Markov processes (Markov chains)

Construct a Bayes net from these variables: parents? CPTs?

Markov assumption: \( X_t \) depends on bounded subset of \( X_{0:t-1} \)

First-order Markov process: \( P(X_t|X_{0:t-1}) = P(X_t|X_{t-1}) \)

Second-order Markov process: \( P(X_t|X_{0:t-1}) = P(X_t|X_{t-2}, X_{t-1}) \)

First-order

Second-order

Stationary process: transition model \( P(X_t|X_{t-1}) \) fixed for all \( t \)
Hidden Markov Model (HMM)

Sensor Markov assumption: \( P(E_t|X_{0:t}, E_{1:t-1}) = P(E_t|X_t) \)

Stationary process: transition model \( P(X_t|X_{t-1}) \) and sensor model \( P(E_t|X_t) \) fixed for all \( t \)

HMM is a special type of Bayes net, \( X_t \) is single discrete random variable:

\[
\begin{align*}
X_0 & \rightarrow X_1 \rightarrow \cdots \rightarrow X_k \rightarrow \cdots \rightarrow X_t \\
E_1 & \rightarrow E_k \rightarrow \cdots \rightarrow E_t
\end{align*}
\]

with joint probability distribution

\[ P(X_{0:t}, E_{1:t}) =? \]
Hidden Markov Model (HMM)

Sensor Markov assumption: $P(E_t|X_{0:t}, E_{1:t-1}) = P(E_t|X_t)$

Stationary process: transition model $P(X_t|X_{t-1})$ and sensor model $P(E_t|X_t)$ fixed for all $t$

HMM is a special type of Bayes net, $X_t$ is single discrete random variable:

$$P(X_{0:t}, E_{1:t}) = P(X_0) \prod_{i=1}^{t} P(X_i|X_{i-1})P(E_i|X_i)$$
Example

First-order Markov assumption not exactly true in real world!

Possible fixes:
1. **Increase order** of Markov process
2. **Augment state**, e.g., add $Temp_t$, $Pressure_t$

Example: robot motion.
Augment position and velocity with $Battery_t$
Inference tasks

Filtering: $P(X_t | e_{1:t})$
belief state—input to the decision process of a rational agent

Prediction: $P(X_{t+k} | e_{1:t})$ for $k > 0$
evaluation of possible action sequences;like filtering without the evidence

Smoothing: $P(X_k | e_{1:t})$ for $0 \leq k < t$
better estimate of past states, essential for learning

Most likely explanation: $\arg \max_{x_{1:t}} P(x_{1:t} | e_{1:t})$
speech recognition, decoding with a noisy channel
Aim: devise a **recursive** state estimation algorithm:

\[ P(X_{t+1} | e_{1:t+1}) = f(e_{t+1}, P(X_t | e_{1:t})) \]
Aim: devise a **recursive** state estimation algorithm:

\[ P(X_{t+1}|e_{1:t+1}) = f(e_{t+1}, P(X_t|e_{1:t})) \]

\[ P(X_{t+1}|e_{1:t+1}) = P(X_{t+1}|e_{1:t}, e_{t+1}) \]
\[ = \alpha P(e_{t+1}|X_{t+1}, e_{1:t}) P(X_{t+1}|e_{1:t}) \]
\[ = \alpha P(e_{t+1}|X_{t+1}) P(X_{t+1}|e_{1:t}) \]
Filtering

Aim: devise a **recursive** state estimation algorithm:

\[ P(X_{t+1}|e_{1:t+1}) = f(e_{t+1}, P(X_t|e_{1:t})) \]

\[ P(X_{t+1}|e_{1:t+1}) = P(X_{t+1}|e_{1:t}, e_{t+1}) \]
\[ = \alpha P(e_{t+1}|X_{t+1}, e_{1:t})P(X_{t+1}|e_{1:t}) \]
\[ = \alpha P(e_{t+1}|X_{t+1})P(X_{t+1}|e_{1:t}) \]

I.e., prediction + estimation. Prediction by summing out \( X_t \):

\[ P(X_{t+1}|e_{1:t+1}) = \alpha P(e_{t+1}|X_{t+1})\sum_{x_t}P(X_{t+1}, x_t|e_{1:t}) \]
\[ = \alpha P(e_{t+1}|X_{t+1})\sum_{x_t}P(X_{t+1}|x_t, e_{1:t})P(x_t|e_{1:t}) \]
\[ = \alpha P(e_{t+1}|X_{t+1})\sum_{x_t}P(X_{t+1}|x_t)P(x_t|e_{1:t}) \]
Filtering

Aim: devise a recursive state estimation algorithm:

\[ P(X_{t+1}|e_{1:t+1}) = f(e_{t+1}, P(X_t|e_{1:t})) \]

\[ P(X_{t+1}|e_{1:t+1}) = P(X_{t+1}|e_{1:t}, e_{t+1}) \]
\[ = \alpha P(e_{t+1}|X_{t+1}, e_{1:t}) P(X_{t+1}|e_{1:t}) \]
\[ = \alpha P(e_{t+1}|X_{t+1}) P(X_{t+1}|e_{1:t}) \]

i.e., prediction + estimation. Prediction by summing out \( X_t \):

\[ P(X_{t+1}|e_{1:t+1}) = \alpha P(e_{t+1}|X_{t+1}) \sum_{x_t} P(X_{t+1}, x_t|e_{1:t}) \]
\[ = \alpha P(e_{t+1}|X_{t+1}) \sum_{x_t} P(X_{t+1}|x_t, e_{1:t}) P(x_t|e_{1:t}) \]
\[ = \alpha P(e_{t+1}|X_{t+1}) \sum_{x_t} P(X_{t+1}|x_t) P(x_t|e_{1:t}) \]

\[ f_{1:t+1} = \text{FORWARD}(f_{1:t}, e_{t+1}) \text{ where } f_{1:t} = P(X_t|e_{1:t}) \]

Time and space constant (independent of \( t \))
Filtering example

\[ P(X_{t+1}|e_{1:t+1}) = \alpha P(e_{t+1}|X_{t+1}) \sum_{x_t} P(X_{t+1}|x_t)P(x_t|e_{1:t}) \]

<table>
<thead>
<tr>
<th>(R_{t-1})</th>
<th>(P(R_t))</th>
<th>(R_t)</th>
<th>(P(U_t))</th>
</tr>
</thead>
<tbody>
<tr>
<td>t</td>
<td>0.7</td>
<td>t</td>
<td>0.9</td>
</tr>
<tr>
<td>f</td>
<td>0.3</td>
<td>f</td>
<td>0.2</td>
</tr>
</tbody>
</table>
Most likely explanation
Most likely sequence \( \neq \) sequence of most likely states!!!!

Most likely path to each \( x_{t+1} \)
\[ = \text{most likely path to some } x_t \text{ plus one more step} \]
\[
\max_{x_1 \ldots x_t} P(x_1, \ldots, x_t, X_{t+1}|e_{1:t+1}) \\
= P(e_{t+1}|X_{t+1}) \max_{x_t} \left( P(X_{t+1}|x_t) \max_{x_1 \ldots x_{t-1}} P(x_1, \ldots, x_{t-1}, x_t|e_{1:t}) \right)
\]

Identical to filtering, except \( f_{1:t} \) replaced by
\[
m_{1:t} = \max_{x_1 \ldots x_{t-1}} P(x_1, \ldots, x_{t-1}, X_t|e_{1:t}),
\]

I.e., \( m_{1:t}(i) \) gives the probability of the most likely path to state \( i \).

Update has sum replaced by max, giving the **Viterbi algorithm**:
\[
m_{1:t+1} = P(e_{t+1}|X_{t+1}) \max_{x_t} (P(X_{t+1}|x_t)m_{1:t})
\]
Viterbi example

<table>
<thead>
<tr>
<th>Rain₁</th>
<th>Rain₂</th>
<th>Rain₃</th>
<th>Rain₄</th>
<th>Rain₅</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true</td>
<td>true</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

State space paths

Umbrella

Most likely paths

\[ m_{1:1} = .8182 \]
\[ m_{1:2} = .5155 \]
\[ m_{1:3} = .0361 \]
\[ m_{1:4} = .0334 \]
\[ m_{1:5} = .0210 \]
Implementation Issues

Viterbi message: \( m_{1:t+1} = P(e_{t+1}|X_{t+1}) \max_{x_t} (P(X_{t+1}|x_t)m_{1:t}) \)

or filtering update: \( P(X_{t+1}|e_{1:t+1}) = \alpha P(e_{t+1}|X_{t+1}) \sum_{x_t} P(X_{t+1}|x_t) P(x_t|e_{1:t}) \)

What is \( 10^{-6} \cdot 10^{-6} \cdot 10^{-6} \)?
Implementation Issues

Viterbi message: $m_{1:t+1} = P(e_{t+1}|X_{t+1}) \max_{x_t} (P(X_{t+1}|x_t)m_{1:t})$

or filtering update: $P(X_{t+1}|e_{1:t+1}) = \alpha P(e_{t+1}|X_{t+1}) \sum_{x_t} P(X_{t+1}|x_t)P(x_t|e_{1:t})$

What is $10^{-6} \cdot 10^{-6} \cdot 10^{-6}$?

What is floating point arithmetic precision?
Implementation Issues

Viterbi message: \( m_{1:t+1} = P(e_{t+1}|X_{t+1}) \max_{x_t} (P(X_{t+1}|x_t)m_{1:t}) \)

or filtering update: \( P(X_{t+1}|e_{1:t+1}) = \alpha P(e_{t+1}|X_{t+1}) \sum_{x_t} P(X_{t+1}|x_t) P(x_t|e_{1:t}) \)

What is \( 10^{-6} \cdot 10^{-6} \cdot 10^{-6} \)?

What is floating point arithmetic precision?

\( 10^{-6} \cdot 10^{-6} \cdot 10^{-6} = 0 \)
Answer?

Use either:
- Rescaling, multiply values by a (large) constant
- \( \log \text{sum trick (Assignment 5)} \)

\( \log \) is monotone increasing, so:
\[
\arg \max f(x) = \arg \max \log f(x)
\]

Also,
\[
\log(a \cdot b) = \log a + \log b
\]

Therefore, work with sums of logarithms of probabilities, rather than products of probabilities:
\[
\begin{align*}
\mathbf{m}_{1:t+1} &= P(e_{t+1} | X_{t+1}) \max_{x_t} (P(X_{t+1} | x_t) m_{1:t}) \\
\rightarrow \log \mathbf{m}_{1:t+1} &= \log P(e_{t+1} | X_{t+1}) + \max_{x_t} (\log P(X_{t+1} | x_t) + \log m_{1:t})
\end{align*}
\]
Hidden Markov models

\( X_t \) is a single, discrete variable (usually \( E_t \) is too)
Domain of \( X_t \) is \( \{1, \ldots, S\} \)

Transition matrix \( T_{ij} = P(X_t = j|X_{t-1} = i) \), e.g., \( \begin{pmatrix} 0.7 & 0.3 \\ 0.3 & 0.7 \end{pmatrix} \)

Sensor matrix \( O_t \) for each time step, diagonal elements \( P(e_t|X_t = i) \)
e.g., with \( U_1 = \text{true}, \ O_1 = \begin{pmatrix} 0.9 & 0 \\ 0 & 0.2 \end{pmatrix} \)

Forward messages as column vectors:
\[ f_{1:t+1} = \alpha O_{t+1} T^T f_{1:t} \]
Dynamic Bayesian networks

$X_t, E_t$ contain arbitrarily many variables in a replicated Bayes net
Summary

Temporal models use state and sensor variables replicated over time

Markov assumptions and stationarity assumption, so we need
  – transition model $P(X_t|X_{t-1})$
  – sensor model $P(E_t|X_t)$

Tasks are filtering, prediction, smoothing, most likely sequence; all done recursively with constant cost per time step

Hidden Markov models have a single discrete state variable; used for speech recognition

Dynamic Bayes nets subsume HMMs; exact update intractable
Example Umbrella Problems

Filtering:

\[ f_{1:t+1} := P(X_{t+1}|e_{1:t+1}) = \alpha P(e_{t+1}|X_{t+1}) \sum_{x_t} P(X_{t+1}|x_t) P(x_t|e_{1:t}) \]

Viterbi:

\[ m_{1:t+1} = P(e_{t+1}|X_{t+1}) \max_{x_t} (P(X_{t+1}|x_t)m_{1:t}) \]

<table>
<thead>
<tr>
<th>( R_{t-1} )</th>
<th>( P(R_t = t) )</th>
<th>( P(R_t = f) )</th>
<th>( R_t )</th>
<th>( P(U_t = t) )</th>
<th>( P(U_t = f) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>t</td>
<td>0.7</td>
<td>0.3</td>
<td>t</td>
<td>0.9</td>
<td>0.1</td>
</tr>
<tr>
<td>f</td>
<td>0.3</td>
<td>0.7</td>
<td>f</td>
<td>0.2</td>
<td>0.8</td>
</tr>
</tbody>
</table>

\[ P(R_3|\neg u_1, u_2, \neg u_3) = ? \]

\[ \arg \max_{R_{1:3}} P(R_{1:3}|\neg u_1, u_2, \neg u_3) = ? \]