Human-Oriented Robotics

Temporal Reasoning

Part 1/3

Kai Arras
Social Robotics Lab, University of Freiburg
Winter term 2014/2015
Temporal Reasoning

Contents

• Introduction
• Temporal Reasoning
• Hidden Markov Models
• Linear Dynamical Systems
• Kalman Filter
• Extended Kalman Filter
• Tracking and Data Association
Introduction: Sequential Data (from Ch. 3)

- Sequential data often arise through measurements of time series, for example:
  - Rainfall measurements on successive days at a particular location
  - Daily currency exchange rates
  - Acoustic features at successive time frames used for speech recognition
  - A human’s arm and hand movements used for sign language understanding

- Other forms of sequential data, e.g. over space, exist as well to which the considered models equally apply

- In applications, we typically wish to be able to predict the next value given observations of the previous values (e.g. in financial forecasting)

- We expect that recent observations are likely to be more informative than more historical observations
**Introduction: Sequential Data** (from Ch. 3)

- This is the case when **successive values** in time series are **correlated**
- **Examples:** spectrogram of the spoken words, weather
Introduction: Markov Models (from Ch. 3)

- Consider a model that postulates dependencies of future observations on all previous observations. Such a model would be impractical because its complexity would grow without limits as the number of observations increases.
- This leads us to consider Markov Models.

Markov models assume that future predictions are independent of all but the most recent observations.
Introduction: Markov Models (from Ch. 3)

- Formally, we recall the chain rule

\[
p(x_1, x_2, \ldots, x_K) = \prod_{k=1}^{K} p(x_k | x_1, \ldots, x_{k-1})
\]

- If we now assume that each of the conditional distributions on the right hand side is independent of all previous observations except the most recent one,

\[
p(x_1, x_2, \ldots, x_K) = \prod_{k=1}^{K} p(x_k | x_1, \ldots, x_{k-1})
= p(x_1) p(x_2 | x_1) p(x_3 | x_1, x_2) p(x_4 | x_1, x_2, x_3) \cdots \\
\quad \cdot p(x_K | x_1, x_2, \ldots, x_{K-1})
\]
Temporal Reasoning

Introduction: Markov Models (from Ch. 3)

we obtain the first-order Markov chain

\[
p(x_1, x_2, \ldots, x_K) = \prod_{k=1}^{K} p(x_k | x_1, \ldots, x_{k-1})
\]

\[
= p(x_1) p(x_2 | x_1) p(x_3 | x_2, x_1) p(x_4 | x_1, x_2, x_3) \cdots
\]

\[
\cdot p(x_K | x_1, x_2, \ldots, x_{K-1})
\]

\[
= p(x_1) \prod_{k=2}^{K} p(x_k | x_{k-1})
\]
Introduction: State Space Model (from Ch. 3)

- Let’s add latent or hidden variables to our model, one for each random variable and let the latent variables form a Markov chain.

- Notice the change in notation: we denote latent variables by $x$ and observations by $z$ (this notation is widely used in particular for LDS).

- It is sometimes common to shade the nodes of latent variables in the graphical representation.
Introduction: State Space Model (from Ch. 3)

- In this model, we view the model to describe a system that evolves on its own, with observations of it occurring in a separate process.
- This model is called state space model or state observation model.

- Hidden variables $x$ are often called states. They are unobservable and typically what we want to estimate.
- Variables $z$ are called observations (or “evidence variables”). It is only through the observations that we can indirectly estimate the $x$’s.
- Observations may be of different type and dimensionality than the states.
- We assume the interval $\Delta t$ to be fixed. Thus we can label time by integer time indices $k$ (or $t$, $i$).
Introduction: State Space Model (from Ch. 3)

- In addition to the independence assumption of the first-order Markov model, we assume that observations at time index $k$ are conditionally independent of the entire state sequence given the state variable at time index $k$.

- The joint distribution of this model is derived as follows:

$$p(x_1, \ldots, x_K, z_1, \ldots, z_K) = p(x_1) p(x_2 | x_1) p(x_3 | x_1, x_2) \cdots$$

$$\cdot p(x_K | x_1, x_2, \ldots, x_{K-1})$$

$$\cdot p(z_1 | x_1, x_2, \ldots, x_K) p(z_2 | x_1, x_2, \ldots, x_K, z_1) \cdots$$

$$\cdot p(z_K | x_1, \ldots, x_K, z_1, \ldots, z_{K-1})$$
Introduction: State Space Model (from Ch. 3)

- In addition to the independence assumption of the first-order Markov model, we assume that observations at time index \( k \) are conditionally independent of the entire state sequence given the state variable at time index \( k \).

- The joint distribution of this model is derived as follows

\[
p(x_1, \ldots, x_K, z_1, \ldots, z_K) = p(x_1) p(x_2|x_1) p(x_3|x_1, x_2) \cdots \\
\quad \cdot p(x_K|x_1, x_2, \ldots, x_{K-1}) \\
\quad \cdot p(z_1|x_1, x_2, \ldots, x_K) p(z_2|x_1, x_2, \ldots, x_K, z_1) \cdots \\
\quad \cdot p(z_K|x_1, \ldots, x_K, z_1, \ldots, z_{K-1})
\]
Introduction: State Space Model (from Ch. 3)

- In addition to the independence assumption of the first-order Markov model, we assume that observations at time index $k$ are conditionally independent of the entire state sequence given the state variable at time index $k$.

- The joint distribution of this model is derived as follows:

$$p(x_1, \ldots, x_K, z_1, \ldots, z_K) = p(x_1) p(x_2 | x_1) p(x_3 | x_1, x_2) \cdots$$

$$\cdot p(x_K | x_1, x_2, \ldots, x_{K-1})$$

$$\cdot p(z_1 | x_1, x_2, \ldots, x_K) p(z_2 | x_1, x_2, \ldots, x_K, z_1) \cdots$$

$$\cdot p(z_K | x_1, \ldots, x_K, z_1, \ldots, z_{K-1})$$

$$= p(x_1) \left[ \prod_{k=2}^{K} p(x_k | x_{k-1}) \right] \prod_{k=1}^{K} p(z_k | x_k)$$
Introduction: State Space Model (from Ch. 3)

- Two important models for sequential data that are described by this graph

1. **Discrete case:** if the latent variables are discrete, we obtain a hidden Markov Model (HMM). Observed variables can either be discrete or continuous in HMMs

2. **Continuous case:** If both the latent and the observed variables are continuous and Gaussian, we have the linear dynamical system (LDS)
State Space Model

- The state space representation is made up by three components:
  1. The **transition model** describes how the world/system evolves. It specifies the probability distribution \( p(x_k \mid x_{0:k-1}) \) over the latest state variable given the previous values. Thanks to the Markov assumption, this is

\[
p(x_k \mid x_{0:k-1}) = p(x_k \mid x_{k-1})
\]

for first-order Markov chains. For second-order Markov chains we have

\[
p(x_k \mid x_{0:k-1}) = p(x_k \mid x_{k-1}, x_{k-2})
\]

where we have introduced the notation \( x_{a:b} \) to denote the sequence of states \( \{x_a, x_{a+1}, \ldots, x_{b-1}, x_b\} \).
State Space Model

- The state space representation is made up by three components:
  
  2. The **observation (or sensor) model** specifies the probability distribution over the observed variable given the previous state and observation sequence

  \[ p(z_k | x_{0:k}, z_{0:k-1}) \]

  This is simplified to be

  \[ p(z_k | x_{0:k}, z_{0:k-1}) = p(z_k | x_k) \]

  which is sometimes called the **sensor Markov assumption**

  3. The **prior probability distribution** over the state at time 0, \( p(x_0) \), sometimes also called initial state model
Temporal Reasoning

State Space Model

- With that, we have specified the complete joint distribution over the domain (= over all our random variables)

$$p(x_0:K, z_1:K) = p(x_0) \prod_{k=1}^{K} p(x_k | x_{k-1}) p(z_k | x_k)$$
State Space Model Example

- Suppose you are a security guard stationed at an underground facility and the only way to know whether it is raining today is by observing the director who comes in with or without an umbrella each morning.

- Let $x_k = Rain_k$ be the binary state variable and $z_k = Umbrella_k$ the binary observation variable, both with values true or false. Then, the state space model is:

  $$
  P(R_t | R_{t-1}) = \begin{cases} 
  0.7 & \text{true} \\
  0.3 & \text{false}
  \end{cases}
  $$

  $$
  P(U_k | R_k) = \begin{cases} 
  0.9 & \text{true} \\
  0.2 & \text{false}
  \end{cases}
  $$

  $$
  P(R_0) = \begin{cases} 
  0.5 & \text{true} \\
  0.5 & \text{false}
  \end{cases}
  $$

  $$
  P(U_0) = \begin{cases} 
  0.5 & \text{true} \\
  0.5 & \text{false}
  \end{cases}
  $$
Note the dependencies between states and sensors: arrows go from actual states to sensor values because the state of the world/system causes the sensors to take on particular values: the rain causes the umbrella to appear.

The inference process goes in the other direction: we seek to estimate the state given observations of the world.
Temporal Reasoning

State Space Model

- The example uses a **first-order** Markov process which implies that the current state variable contains all the information needed to compute the probability of rain for the next time step.

- Whether this assumption is **reasonable** depends on the context/domain. The assumption may be exactly true or may be approximate.

- There are two ways to **improve the accuracy** of approximate models:
  - **Increasing the order** of the Markov process model. E.g. we could add $Rain_{k-2}$ as a parent of $Rain_k$.
  - **Adding more state variables**. For example, we could add *Season* or *Humidity*. However, adding more variables might improve the system’s predictive power but increases the prediction requirements: we have to predict the new variables as well.

- Looking for a self-sufficient set of variables that reflect the physics of the modeled process.
Inference

- Having set up the representation of a generic temporal model consisting of $p(x_0)$, $p(x_k | x_{k-1})$, and $p(z_k | x_k)$, we can formulate the four basic inference tasks:

  - **Filtering**: computing the posterior distribution over the most recent state given all observations to date
    $$p(x_k | z_{1:k})$$
    Keeping track of the system’s current state in an online fashion

  - **Prediction**: computing the posterior distribution over the future state given all observations to date
    $$p(x_{k+t} | z_{1:k}) \quad t > 0$$
    Projecting the state into the future without observations (e.g. weather)
Inference

- **Smoothing**: computing the posterior distribution over a past state given all observations up to the present

\[ p(x_t \mid z_{1:k}) \quad 0 \leq t < k \]

As opposed to filtering incorporates “the future”. Leads to smoother state estimates than filtering

- **Most likely sequence**: given a sequence of observations, finding the most likely state sequence to have generated those observations

\[ \arg \max_{\mathbf{x}_{1:k}} p(\mathbf{x}_{1:k} \mid \mathbf{z}_{1:k}) \]

Highly relevant e.g. in speech recognition where the aim is to find the most likely word given a series of sounds
Examples

- Consider a motion tracking system able to follow human motion

  Interpreting sign language from a sequence of images. Hands are detected and tracked

- Suppose we have two goals:
  - Tracking and state estimation: associating single-frame detections (of the hand, the human) over time and accurately estimating their position from noisy observations
  - Recognizing different movement primitives which may serve later to infer high-level activities such as the sign that is presented or the action the human is engaged in

- The former is a continuous, the latter a discrete state estimation problem. We will first consider the discrete case

  Keeping track of human actions from sequences of laser-based positions

Source [2]
Hidden Markov Models (HMM)

- An HMM is a temporal probabilistic model in which the state of the process is described by a single discrete random variable.
- The random variable \( x_k \) can take \( S \) possible states \( s \in \{1,\ldots,S\} \).
- The HMM is widely used in speech recognition, natural language modeling, human activity recognition, on-line handwriting recognition, analysis of protein/DNA sequences, etc.
- Notice the similarity to mixture models where the latent variables are also discrete, describing which mixture component is responsible for generating the corresponding observation.
- Except that here we have that the probability distribution over \( x_k \) depends on the state of the previous latent variable \( x_{k-1} \) through the transition model \( p(x_k \mid x_{k-1}) \).
Transition Model

- The transition model is represented as a square matrix $A$ with transition probabilities $A_{ij}$ where $0 \leq A_{ij} \leq 1$ and $\sum_j A_{ij} = 1$, that is, outgoing probabilities sum up to 1.

- $A$ has $S(S-1)$ free parameters.

$$A = \begin{pmatrix}
A_{11} & A_{12} & A_{13} \\
A_{21} & A_{22} & A_{23} \\
A_{31} & A_{32} & A_{33}
\end{pmatrix} \quad \sum_j A_{1j} = 1$$

- The state transition diagram is not a graphical model, because the nodes are not separate variables but rather states of a single variable.

- Probability of a transition from state 3 to state 2
Hidden Markov Models

Transition Model

- Unfolded over time, we obtain a lattice (or trellis) diagram
Transition Model

- States in a simple human action recognition example

- Ground truth state sequence:
Hidden Markov Models

Observation Model

- Observed variables can either be discrete or continuous in HMMs
- They may be of different type and dimensionality than the states
- The observation model $p(z_k | x_k)$ “maps” the space of observations to the space of discrete HMM states. Concretely, for each $z_k$ it computes a probability distribution over the states $s \in \{1, ..., S\}$ specifying the emission probability that state $s$ caused observation $z_k$ to appear
- In the continuous case, a parametric distribution with parameters $\theta$, such as a Gaussian, is typically chosen. We may also write $p(z_k | x_k, \theta)$
- We then need a conditional probability $p(z_k | x_k = s, \theta)$ for each state $s$
Observation Model and Prior

- In the **discrete case**, the observation model is represented as a matrix $E$ of emission probabilities $E_{ij}$ where $0 \leq E_{ij} \leq 1$ and $\sum_j E_{ij} = 1$

$$
E = \begin{pmatrix}
E_{11} & E_{12} & E_{13} & E_{14} & E_{15} \\
E_{21} & E_{22} & E_{23} & E_{24} & E_{25} \\
E_{31} & E_{32} & E_{33} & E_{34} & E_{35}
\end{pmatrix}
$$

Every row of $E$ describes a distribution $p(z_t | x_t = s)$

- Let observations be from a finite set of $O$ symbols. A special case is when $E$ is squared, that is, $S = O$

- Finally, we also have to specify the **prior distribution** $p(x_0)$ over states $s$, using, for example, a categorial distribution
Hidden Markov Models

Observation Model

- States and observations in our running example: $S = O = 3$
- Suppose we have a noisy “sensor” for movement primitives, for example, a poorly trained 3-way classifier

- Ground truth sequence of state

- Sequence of observations
Hidden Markov Models

Inference

• Having set up the representation of an HMM with parameters \( p(x_0) \), 
\( p(x_k | x_{k-1}) \) and \( p(z_k | x_k) \) we recall the four basic inference tasks:

• **Filtering:** computing the posterior distribution \( p(x_k | z_{1:k}) \) over the most recent state given all observations to date

• **Prediction:** computing the posterior distribution over the future state given all observations to date: \( p(x_{k+t} | z_{1:k}) \quad t > 0 \)

• **Smoothing:** computing the posterior distribution over a past state given all observations up to the present, i.e. \( p(x_t | z_{1:k}) \quad 0 \leq t < k \)

• **Most likely sequence:** given a sequence of observations, finding the most likely state sequence to have generated those observations. Formally, \( \arg \max_{x_{1:k}} p(x_{1:k} | z_{1:k}) \)
Inference: Smoothing

• Let us start with the more general smoothing which contains filtering and prediction as a special case

• We want to compute \( p(x_t \mid z_{1:k}) \quad 0 \leq t < k \)

\[
p(x_t \mid z_{1:k}) &= \eta p(x_t, z_{1:k}) \\
&= \eta p(x_t, z_{1:t}, z_{t+1:k}) \\
&= \eta p(z_{t+1:k} \mid x_t, z_{1:t}) \ p(x_t, z_{1:t}) \\
&= \eta p(z_{t+1:k} \mid x_t) \ p(x_t, z_{1:t})
\]

conditional probability  
“dividing up the evidence”  
chain rule  
sensor Markov assumption  

backward  
forward

• We will see next that the first term can be computed in a backward recursion and the second term in a forward recursion through the chain rule.
Hidden Markov Models

Inference: Smoothing

- Computing \( p(x_t, z_{1:t}) \) is called **forward step**

\[
p(x_t, z_{1:t}) = \sum_{x_{t-1}=1}^S p(x_t, x_{t-1}, z_{1:t})
\]

- Computing \( p(x_t, z_{1:t}) \) is called **forward step**

\[
= \sum_{x_{t-1}} p(x_t, x_{t-1}, z_{1:t-1}, z_t)
\]

- Computing \( p(x_t, z_{1:t}) \) is called **forward step**

\[
= \sum_{x_{t-1}} p(z_t | x_t, x_{t-1}, z_{1:t-1}) p(x_t | x_{t-1}, z_{1:t-1}) p(x_{t-1}, z_{1:t-1})
\]

- Computing \( p(x_t, z_{1:t}) \) is called **forward step**

\[
= \sum_{x_{t-1}} p(z_t | x_t) p(x_t | x_{t-1}) p(x_{t-1}, z_{1:t-1})
\]
Inference: Smoothing

- Computing \( p(x_t, z_{1:t}) \) is called **forward step**

\[
p(x_t, z_{1:t}) = \sum_{x_{t-1}=1}^S p(x_t, x_{t-1}, z_{1:t})
\]

\[
= \sum_{x_{t-1}} p(x_t, x_{t-1}, z_{1:t-1}, z_t)
\]

\[
= \sum_{x_{t-1}} p(z_t | x_t, x_{t-1}, z_{1:t-1}) \ p(x_t | x_{t-1}, z_{1:t-1}) \ p(x_{t-1}, z_{1:t-1})
\]

“dividing up the evidence”

chain rule

\[
= \sum_{x_{t-1}} p(z_t | x_t) \ p(x_t | x_{t-1}) \ p(x_{t-1}, z_{1:t-1})
\]

conditional independence

We have found a recursion!
Inference: Smoothing

- All terms in $p(x_t, z_{1:t})$ are known HMM parameters (emission and transition probabilities). Let the recursive term be $\alpha_{t-1}$, then we can write

$$\alpha_t = p(z_t | x_t) \sum_{x_{t-1}} p(x_t | x_{t-1}) \alpha_{t-1}$$

- How to compute $\alpha_1$? We find

$$\alpha_1 = p(x_1, z_{1:1}) = p(x_1, z_1) = p(x_1) p(z_1 | x_1)$$

Again, this depends only on known parameters (priors/emission probs.)
Inference: Filtering

- Let us rewrite the $\alpha$-term using conditional probability

$$\alpha_t = p(x_t, z_{1:t}) = \eta p(x_t | z_{1:t})$$

- We recognize the **wanted posterior probability for filtering**
  (up to normalization)

- The **recursiveness** to compute $\alpha$ is an important property of **filtering algorithms**: the update only depends on the previous estimate and the “new” observation,

$$p(x_t | z_{1:t}) = f(z_t, p(x_{t-1} | z_{1:t-1}))$$

rather than going back over the entire observation history every time
Inference: Filtering

- Considering the rewritten $\alpha$-term:

$$p(x_t \mid z_{1:t}) = \eta \ p(z_t \mid x_t) \sum_{x_{t-1}} p(x_t \mid x_{t-1}) \ p(x_{t-1} \mid z_{1:t-1})$$

leads us immediately – after dropping the conditioning on the observation history – to an important result: the recursive Bayes filter

$$p(x_t \mid z_t) = \eta \ p(z_t \mid x_t) \sum_{x_{t-1}} p(x_t \mid x_{t-1}) \ p(x_{t-1} \mid z_{t-1})$$

- The Bayes filter is a general recursive state estimation scheme (in the continuous case, the sum becomes an integral)
- Various applications in control and robotics: localization, SLAM, tracking, …
Inference: Filtering

- Filtering has a \textit{prediction–update scheme}
- For this to become evident, we transform the sum on the right hand side

\[
\sum_{x_{t-1}} p(x_t | x_{t-1}) p(x_{t-1}, z_{1:t-1}) =
\]

\[
= \sum_{x_{t-1}} p(x_t | x_{t-1}, z_{1:t-1}) p(x_{t-1}, z_{1:t-1}) \quad \text{conditional independence}
\]

\[
= \sum_{x_{t-1}} p(x_t, x_{t-1}, z_{1:t-1}) \quad \text{chain rule}
\]

\[
= p(x_t, z_{1:t-1}) \quad \text{marginalization}
\]

\[
= \eta p(x_t | z_{1:t-1}) \quad \text{conditional probability}
\]

- This represent a \textbf{one-step prediction} of the next state
Inference: Filtering

- Filtering has a prediction–update scheme

\[
p(x_t | z_t) = \eta \ p(z_t | x_t) \sum_{x_{t-1}} p(x_t | x_{t-1}) \ p(x_{t-1} | z_{t-1})
\]

\[= \eta' \ p(z_t | x_t) \ p(x_t | z_{t-1})
\]

- Kalman filter and particle filter are two continuous instances of the recursive Bayes filter. They have explicit prediction and update steps (later in this course)
Inference: Filtering

- Back to HMMs, let us look at the forward step
- In filtering, we compute $\alpha_t = p(x_t, z_{1:t})$ as

$$\alpha_t = p(z_t \mid x_t) \sum_{x_{t-1}} p(x_t \mid x_{t-1}) \alpha_{t-1}$$

in each step by taking the previous $\alpha_{t,s}$ values for every state $s$, summing them up with weights given by the transition matrix $A$, and then multiplying by the observation model for state $s$, $p(z_t \mid x_t = s)$

- This is called the **forward algorithm**. We start at the first node of the chain, work along the chain and evaluate $\alpha_t$ for every latent node

- $O(S^2)$ complexity per step, $O(S^2K)$ for a chain of length $K$
Inference: Smoothing

Let us get back to smoothing and the **backward term**.

We recall

\[
p(x_t \mid z_{1:k}) = \eta \ p(x_t, z_{1:k})
\]

\[
= \eta \ p(x_t, z_{1:t}, z_{t+1:k})
\]

\[
= \eta \ p(z_{t+1:k} \mid x_t, z_{1:t}) \ p(x_t, z_{1:t})
\]

\[
= \eta \ p(z_{t+1:k} \mid x_t) \ p(x_t, z_{1:t})
\]

As we will see next, the backward term has a very similar derivation than the forward term.
Inference: Smoothing

- Computing $p(z_{k+1:k} | x_t)$ is called **backward step**

\[
p(z_{t+1:k} | x_t) = \sum_{x_{t+1}=1}^{S} p(z_{t+1:k}, x_{t+1} | x_t)
\]

\[
= \sum_{x_{t+1}} p(z_{t+1}, z_{t+2:k}, x_{t+1} | x_t)
\]

\[
= \sum_{x_{t+1}} p(z_{t+2:k} | z_{t+1}, x_{t+1}, x_t) p(z_{t+1} | x_{t+1}, x_t) p(x_{t+1} | x_t)
\]

chain rule

\[
= \sum_{x_{t+1}} p(z_{t+2:k} | x_{t+1}) p(z_{t+1} | x_{t+1}) p(x_{t+1} | x_t)
\]

conditional independence

marginalization

“dividing up the evidence”
Hidden Markov Models

Inference: Smoothing

- Computing \( p(z_{k+1:k} \mid x_t) \) is called **backward step**

\[
p(z_{t+1:k} \mid x_t) = \sum_{x_{t+1}=1}^{S} p(z_{t+1:k}, x_{t+1} \mid x_t)
\]

\[
= \sum_{x_{t+1}} p(z_{t+1}, z_{t+2:k}, x_{t+1} \mid x_t)
\]

\[
= \sum_{x_{t+1}} p(z_{t+2:k} \mid z_{t+1}, x_{t+1}, x_t) \ p(z_{t+1} \mid x_{t+1}, x_t) \ p(x_{t+1} \mid x_t)
\]

Again a recursion, this time in a backward sense!
Inference: Smoothing

Again, all terms in

\[
p(z_{t+1:k} \mid x_t) = \sum_{x_{t+1}=1}^{S} p(z_{t+2:k} \mid x_{t+1}) p(z_{t+1} \mid x_{t+1}) p(x_{t+1} \mid x_t)
\]

are known HMM parameters (emission and transition probabilities). Let the recursive term be \( \beta_t \), then we write

\[
\beta_t = \sum_{x_{t+1}} \beta_{t+1} p(z_{t+1} \mid x_{t+1}) p(x_{t+1} \mid x_t)
\]

This is valid for \( t = 1, \ldots, k - 1 \). But what about \( \beta_k \)? Let us consider the second last step \( t = k - 1 \).
Inference: Smoothing

- For the second last step, \( t = k - 1 \), we redo the derivation

\[
\beta_{k-1} = p(z_{k:k} \mid x_{k-1}) \\
= p(z_k \mid x_{k-1}) \\
= \sum_{x_k} p(z_k, x_k \mid x_{k-1}) \\
= \sum_{x_k} p(z_k \mid x_k, x_{k-1}) p(x_k \mid x_{k-1}) \\
= \sum_{x_k} p(z_k \mid x_k) p(x_k \mid x_{k-1})
\]

- It follows that \( \beta_k = 1 \) for this to be true
Inference: Smoothing

- Let us look at the backward step
- We compute $\beta_t = p(z_{t+1:k} \mid x_t)$ as

$$\beta_t = \sum_{x_{t+1}} \beta_{t+1} p(z_{t+1} \mid x_{t+1}) p(x_{t+1} \mid x_t)$$

in each step by taking the future $\beta_{t+1,s}$ values for every state $s$, multiplying the corresponding elements of $A$ and observation model for state $s$, $p(z_{t+1} \mid x_{t+1} = s)$, and summing up

- This is called the **backward algorithm**. We start at the last node of the chain, work backwards, and evaluate $\beta_t$ for every latent node
- $O(S^2)$ complexity per step, $O(S^2K)$ for a chain of length $K$
Inference: Smoothing

- Let us wrap up: in our derivation we have found the forward algorithm and the backward algorithm to compute the two individual terms of

\[ p(x_t \mid z_{1:k}) = \eta \ p(z_{t+1:k} \mid x_t) \ p(x_t, z_{1:t}) \]

\[ = \eta \ \beta_t \ \alpha_t \]

- The \( \alpha \) term is computed by filtering, running forward in time from 1 to \( t \), the \( \beta \) term is computed by running backward in time from \( k \) down to \( t+1 \)

- The smoothing result is finally obtained by multiplication and normalization of the terms. The corresponding algorithm is the forward-backward algorithm
Hidden Markov Models

Inference: Smoothing

- The forward-backward algorithm for smoothing

**Algorithm:** Forward-Backward Algorithm

<table>
<thead>
<tr>
<th>In:</th>
<th>( z ), vector of observations</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>prior, prior distribution on initial state ( p(x_0) )</td>
</tr>
<tr>
<td>Out:</td>
<td>( x ), vector of smoothed probability distributions</td>
</tr>
<tr>
<td>Local:</td>
<td>( f ), vector of forward probabilities</td>
</tr>
<tr>
<td></td>
<td>( b ), vector of backward probabilities, initially all 1</td>
</tr>
</tbody>
</table>

\[
\begin{align*}
f[0] & \leftarrow \text{prior} \\
\text{for } t = 1 \ldots k \text{ do} & \\
& \quad f[t] \leftarrow \alpha_t(f[t-1], z[t]) \\
\text{end} \\
\text{for } t = k \ldots 1 \text{ do} & \\
& \quad x[t] \leftarrow \text{normalize}(f[t] \times b) \\
& \quad b \leftarrow \beta_t(b, z[t]) \\
\text{end} \\
\text{return } x
\end{align*}
\]
Inference: Smoothing

- The forward-backward algorithm (and variants thereof) is the backbone for many applications that deal with sequences of noisy observations.

- However, it is an off-line algorithm and does not work in an online setting where smoothed estimates are required as new observations are continuously added to the end of the sequence.

- Fixed-lag smoothing is able to do this. It involves computing the smoothed estimated

\[ p(x_{t-d} \mid z_{1:t}) \]

for fixed \( d \). That is, smoothing is done for the time index \( d \) steps behind the current time \( t \).

- Efficient fixed-lag smoothing algorithms with constant time updates exist (see literature).
Inference: Filtering Example

- Let us illustrate filtering and smoothing in the umbrella example

\[ A = \begin{pmatrix} 0.7 & 0.3 \\ 0.3 & 0.7 \end{pmatrix} \]

Prior \[ p(R_0) = \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix} \]

Transition model

\[ E = \begin{pmatrix} 0.9 & 0.1 \\ 0.2 & 0.8 \end{pmatrix} \]

Observation model

- **Day 0**: we have no observation, only the security guard’s prior beliefs. Let us assume uniformity, that is, \[ p(R_0) = (0.5, 0.5)^T \]
Hidden Markov Models

Inference: Filtering Example

- **Day 1**: $U_1 = \text{true}$, the umbrella appears

- The prediction from $k = 0$ to $k = 1$ is

\[
p(R_1) = \sum_{r_0=\{t,f\}} p(R_1|R_0 = r_0) p(R_0 = r_0)
\]
\[
= \begin{pmatrix} 0.7 \\ 0.3 \end{pmatrix} \cdot 0.5 + \begin{pmatrix} 0.3 \\ 0.7 \end{pmatrix} \cdot 0.5 = \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix}
\]

- Then the update step simply multiplies by the probability of the observation for $k = 1$ and normalizes

\[
p(R_1|U_1 = u_1) = \eta \ p(U_1 = u_1|R_1) \ p(R_1)
\]
\[
= \eta \ \begin{pmatrix} 0.9 \\ 0.2 \end{pmatrix} \cdot \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix} = \eta \ \begin{pmatrix} 0.45 \\ 0.1 \end{pmatrix} = \begin{pmatrix} 0.818 \\ 0.182 \end{pmatrix}
\]
Inference: Filtering Example

- **Day 2**: $U_2 = \text{true},$ umbrella appears again

- The prediction from $k = 1$ to $k = 2$ is

\[
p(R_2|U_1 = u_1) = \sum_{r_1 = \{t, f\}} p(R_2|R_1 = r_1) p(R_1 = r_1|U_1 = u_1)
\]
\[
= \left( \begin{array}{c} 0.7 \\ 0.3 \end{array} \right) \cdot 0.818 + \left( \begin{array}{c} 0.3 \\ 0.7 \end{array} \right) \cdot 0.182 = \left( \begin{array}{c} 0.627 \\ 0.373 \end{array} \right)
\]

- The update step for $k = 2$ is

\[
p(R_2|U_2 = u_2) = \eta \ p(U_2 = u_2|R_2) \ p(R_2|U_1 = u_1)
\]
\[
= \eta \left( \begin{array}{c} 0.9 \\ 0.2 \end{array} \right) \cdot \left( \begin{array}{c} 0.627 \\ 0.373 \end{array} \right) = \eta \left( \begin{array}{c} 0.565 \\ 0.075 \end{array} \right) = \left( \begin{array}{c} 0.883 \\ 0.117 \end{array} \right)
\]
Hidden Markov Models

Inference: Filtering Example

- **Day 2**: \( U_2 = \text{true} \), umbrella appears again

- The prediction from \( k = 1 \) to \( k = 2 \) is

  \[
  p(R_2 | U_1 = u_1) = \sum_{r_1 = \{t, f\}} p(R_2 | R_1 = r_1) p(R_1 = r_1 | U_1 = u_1)
  \]

  \[
  = \begin{pmatrix} 0.7 \\ 0.3 \end{pmatrix} \cdot 0.818 + \begin{pmatrix} 0.3 \\ 0.7 \end{pmatrix} \cdot 0.182 = \begin{pmatrix} 0.627 \\ 0.373 \end{pmatrix}
  \]

- The update step for \( k = 2 \) is

  \[
  p(R_2 | U_2 = u_2) = \eta \ p(U_2 = u_2 | R_2) p(R_2 | U_1 = u_1)
  \]

  \[
  = \eta \ \begin{pmatrix} 0.9 \\ 0.2 \end{pmatrix} \cdot \begin{pmatrix} 0.627 \\ 0.373 \end{pmatrix} = \eta \begin{pmatrix} 0.565 \\ 0.075 \end{pmatrix} = \begin{pmatrix} 0.883 \\ 0.117 \end{pmatrix}
  \]

  Increases because rain persists
Inference: Smoothing Example

Let us compute the **smoothed estimate** for **day 1**, given the umbrella observations for day 1 and day 2: $U_1 = \text{true}$, $U_2 = \text{true}$

$$p(R_1|U_1 = u_1, U_2 = u_2) = \eta \ p(R_1|U_1 = u_1) \ p(U_2 = u_2|R_1)$$

The first term is known from the filtering pass, the second term can be computed by applying the backward recursion

$$p(U_2 = u_2|R_1) = \sum_{r_2 = \{t, f\}} \beta_2 p(U_2 = u_2|R_2 = r_2) p(R_2 = r_2|R_1)$$

$$= 1 \cdot 0.9 \cdot \begin{pmatrix} 0.7 \\ 0.3 \end{pmatrix} + 1 \cdot 0.2 \cdot \begin{pmatrix} 0.3 \\ 0.7 \end{pmatrix} = \begin{pmatrix} 0.69 \\ 0.41 \end{pmatrix}$$

Plugging this back in yields the smoothed estimate

$$p(R_1|U_1 = u_1, U_2 = u_2) = \eta \begin{pmatrix} 0.818 \\ 0.182 \end{pmatrix} \times \begin{pmatrix} 0.69 \\ 0.41 \end{pmatrix} = \begin{pmatrix} 0.883 \\ 0.117 \end{pmatrix}$$
Hidden Markov Models

Inference: Smoothing Example

- Let us compute the **smoothed estimate** for day 1, given the umbrella observations for day 1 and day 2: $U_1 = \text{true}, U_2 = \text{true}$

\[ p(R_1|U_1 = u_1, U_2 = u_2) = \eta \ p(R_1|U_1 = u_1) \ p(U_2 = u_2|R_1) \]

- The first term is known from the filtering pass, the second term can be computed by applying the backward recursion

\[ p(U_2 = u_2|R_1) = \sum_{r_2=\{t,f\}} \beta_2 \ p(U_2 = u_2|R_2 = r_2) \ p(R_2 = r_2|R_1) \]

\[ = 1 \cdot 0.9 \cdot \begin{pmatrix} 0.7 \\ 0.3 \end{pmatrix} + 1 \cdot 0.2 \cdot \begin{pmatrix} 0.3 \\ 0.7 \end{pmatrix} = \begin{pmatrix} 0.69 \\ 0.41 \end{pmatrix} \]

- Plugging this back in yields the smoothed estimate

\[ p(R_1|U_1 = u_1, U_2 = u_2) = \eta \ \begin{pmatrix} 0.818 \\ 0.182 \end{pmatrix} \times \begin{pmatrix} 0.69 \\ 0.41 \end{pmatrix} = \begin{pmatrix} 0.883 \\ 0.117 \end{pmatrix} \]

Higher than filtering for $k = 1$
Inference: Filtering Example

- In our simple human action recognition example, the sequence of observations is

- Let us assume the following HMM parameters

  \[
  A = \begin{pmatrix}
  0.98 & 0.01 & 0.01 \\
  0.04 & 0.95 & 0.01 \\
  0.025 & 0.025 & 0.95 \\
  \end{pmatrix}
  \quad
  E = \begin{pmatrix}
  0.7 & 0.2 & 0.1 \\
  0.1 & 0.8 & 0.1 \\
  0.1 & 0.3 & 0.6 \\
  \end{pmatrix}
  \quad
  p(x_0) = \begin{pmatrix}
  0.33 \\
  0.33 \\
  \end{pmatrix}
  \]

- The filtered probabilities are
Inference: Smoothing Example

- In our simple human action recognition example, the sequence of observations is

- Let us assume the following HMM parameters

\[ A = \begin{pmatrix} 0.98 & 0.01 & 0.01 \\ 0.04 & 0.95 & 0.01 \\ 0.025 & 0.025 & 0.95 \end{pmatrix} \quad E = \begin{pmatrix} 0.7 & 0.2 & 0.1 \\ 0.1 & 0.8 & 0.1 \\ 0.1 & 0.3 & 0.6 \end{pmatrix} \quad p(x_0) = \begin{pmatrix} 0.33 \\ 0.33 \end{pmatrix} \]

- The **smoothed** probabilities are
Inference: Smoothing Example

- In our simple human action recognition example, the sequence of observations is

- Let us assume the following HMM parameters

\[
A = \begin{pmatrix}
0.98 & 0.01 & 0.01 \\
0.04 & 0.95 & 0.01 \\
0.025 & 0.025 & 0.95
\end{pmatrix}
\quad E = \begin{pmatrix}
0.7 & 0.2 & 0.1 \\
0.1 & 0.8 & 0.1 \\
0.1 & 0.3 & 0.6
\end{pmatrix}
\quad p(x_0) = \begin{pmatrix}
0.33 \\
0.33
\end{pmatrix}
\]

- The smoothed probabilities are

Smoothing provides much better state estimates!
Inference: Prediction

- The task of prediction can be seen as **filtering without new evidence**
- The filtering process already incorporates a **one-step prediction**. A general prediction of the state at time $t + i + 1$ from a prediction for $t + i$ (with evidence only up to time $t$) is

$$
p(x_{t+i+1} \mid z_{1:t}) = \sum_{x_{t+i}} p(x_{t+i+1} \mid x_{t+i}) p(x_{t+i} \mid z_{1:t})
$$

- Of course, this computation only involves the transition model, not the observation model
- Predicting further and further into the future makes the predicted distribution converge towards a **stationary distribution**, which is only determined by the transition matrix regardless the starting state
Inference: Prediction

- Original transition model

\[ A = \begin{pmatrix} 0.98 & 0.01 & 0.01 \\ 0.04 & 0.95 & 0.01 \\ 0.025 & 0.025 & 0.95 \end{pmatrix} \]

- Different priors

\[ p(x_0) = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \]
Inference: Prediction

- Uncertain second state

\[ A = \begin{pmatrix} 0.92 & 0.08 & 0.0 \\ 0.3 & 0.6 & 0.1 \\ 0.009 & 0.001 & 0.99 \end{pmatrix} \]

- The more uncertain (less peaked) a row distribution \( p(x_t \mid x_{t-1} = s) \) in \( A \) is, the less likely it is to converge to state \( s \) in the stationary distribution.

- The mixing time is the time it takes for the Markov process to converge.

- The more uncertainty there is in the transition model (e.g. many nearly uniform row distributions), the shorter will be the mixing time and the more the future is obscured (stationary distribution nearly uniform).
Inference: Most Likely Sequence

- Given a sequence of observations, we want to find the **most likely state sequence** to have generated those observations.
- Applications of this inference task include (among many others) **speech recognition** or **handwriting recognition** where we seek to find a word from a noisy sequence of recognized phonemes or letters.
- Phonemes or letters are typically obtained by probabilistic classification based on features that characterize portions of the original signal (e.g. small time frames).
Inference: Most Likely Sequence

• For example, suppose the security guard observes the umbrella sequence \{true, true, false, true, true\}. What is the weather sequence most likely to explain this?

  • Does the absence of the umbrella on day 3 mean that it was not raining, or did the director forget to bring it?
  • If it didn’t rain on day 3, perhaps – as weather tends to persist – it did not rain on day 4 either but the director brought it just in case. And so on...

• In all, there are $2^5$ possible weather sequences. Is there a smart way to find the most likely, better than enumerating all of them?

• What about taking the smoothing result and choose the state with the highest posterior probability?

  • Nope! Those probabilities are distributions over single time steps. To find the most likely sequence we must consider joint probabilities over all time steps.
Inference: Most Likely Sequence

- Let us view each sequence as a path through the trellis diagram.

- Recall, we want to find the state sequence $x^*$ that maximizes the probability along its path, i.e.

$$x^* = \arg \max_{x_{1:k}} p(x_{1:k} \mid z_{1:k})$$
Inference: Most Likely Sequence

- Let us consider the **second last step** of a sequence of length $k$ and see if we get an idea for an algorithm by induction
- Assume we already have the maximizing paths for time $k-1$

Can we **reuse** the results computed so far when extending the paths to time $k$?
Inference: Most Likely Sequence

- This would be the case if we were able to simply **append** the most likely path from time $k-1$ to time $k$ to the computed ones, limiting the focus of our maximization onto a **single transition**

- In other words: if there were a **recursive relationship** between most likely paths to each state in $x_k$ and most likely paths to each state in $x_{k-1}$
Inference: Most Likely Sequence

- Let us first consider the maximum

\[
\max_{\mathbf{x}_{1:k-1}} p(\mathbf{x}_{1:k} \mid \mathbf{z}_{1:k})
\]

instead of the \( \arg \max_{\mathbf{x}_{1:k}} p(\mathbf{x}_{1:k} \mid \mathbf{z}_{1:k}) \), and see if we can later recover the state sequence from the maximum.

- And consider the following useful relationship: given two nonnegative functions \( f(a) \geq 0, g(a, b) \geq 0 \ \forall a, b \)

\[
\max_{a, b} f(a) g(a, b) = \max_{a} \left( f(a) \max_{b} g(a, b) \right)
\]

- This can be verified by first ignoring the maximization over \( a \) making \( f(a) \) a constant. Thus, we can “pull” the maximization over \( b \) into the product.
Inference: Most Likely Sequence

- Developing the maximization term

\[
\max_{\mathbf{x}_{1:k-1}} p(\mathbf{x}_{1:k} | \mathbf{z}_{1:k})
\]

\[
= \eta \max_{\mathbf{x}_{1:k-1}} p(\mathbf{x}_{1:k}, \mathbf{z}_{1:k})
\]

\[
= \eta \max_{\mathbf{x}_{1:k-1}} p(\mathbf{x}_{1:k-1}, \mathbf{z}_{1:k-1}, \mathbf{x}_k, \mathbf{z}_k)
\]

\[
= \eta \max_{\mathbf{x}_{1:k-1}} p(\mathbf{x}_{1:k-1}, \mathbf{z}_{1:k-1}) p(\mathbf{x}_k | \mathbf{x}_{1:k-1}, \mathbf{z}_{1:k-1}) p(\mathbf{z}_k | \mathbf{x}_{1:k-1}, \mathbf{z}_{1:k-1}, \mathbf{x}_k)
\]

\[
= \eta \max_{\mathbf{x}_{1:k-1}} p(\mathbf{x}_{1:k-1}, \mathbf{z}_{1:k-1}) p(\mathbf{x}_k | \mathbf{x}_{k-1}) p(\mathbf{z}_k | \mathbf{x}_k)
\]

\[
= \eta p(\mathbf{z}_k | \mathbf{x}_k) \max_{\mathbf{x}_{1:k-1}} p(\mathbf{x}_k | \mathbf{x}_{k-1}) p(\mathbf{x}_{1:k-1}, \mathbf{z}_{1:k-1})
\]

\[
= \eta p(\mathbf{z}_k | \mathbf{x}_k) \max_{\mathbf{x}_{k-1}} \left( p(\mathbf{x}_k | \mathbf{x}_{k-1}) \max_{\mathbf{x}_{1:k-2}} p(\mathbf{x}_{1:k-1}, \mathbf{z}_{1:k-1}) \right)
\]
Inference: Most Likely Sequence

- Developing the maximization term

\[
\max_{\mathbf{x}_{1:k-1}} p(\mathbf{x}_{1:k} \mid \mathbf{z}_{1:k})
\]

\[
= \eta \max_{\mathbf{x}_{1:k-1}} p(\mathbf{x}_{1:k}, \mathbf{z}_{1:k})
\]

\[
= \eta \max_{\mathbf{x}_{1:k-1}} p(\mathbf{x}_{1:k-1}, \mathbf{z}_{1:k-1}, \mathbf{x}_k, \mathbf{z}_k)
\]

\[
= \eta \max_{\mathbf{x}_{1:k-1}} p(\mathbf{x}_{1:k-1}, \mathbf{z}_{1:k-1}) p(\mathbf{x}_k \mid \mathbf{x}_{1:k-1}, \mathbf{z}_{1:k-1}) p(\mathbf{z}_k \mid \mathbf{x}_{1:k-1}, \mathbf{z}_{1:k-1}, \mathbf{x}_k)
\]

\[
= \eta \max_{\mathbf{x}_{1:k-1}} p(\mathbf{x}_{1:k-1}, \mathbf{z}_{1:k-1}) p(\mathbf{x}_k \mid \mathbf{x}_{1:k-1}) p(\mathbf{z}_k \mid \mathbf{x}_k)
\]

\[
= \eta p(\mathbf{z}_k \mid \mathbf{x}_k) \max_{\mathbf{x}_{1:k-1}} p(\mathbf{x}_k \mid \mathbf{x}_{1:k-1}) p(\mathbf{x}_{1:k-1}, \mathbf{z}_{1:k-1})
\]

\[
= \eta p(\mathbf{z}_k \mid \mathbf{x}_k) \max_{\mathbf{x}_{k-1}} \left( p(\mathbf{x}_k \mid \mathbf{x}_{k-1}) \max_{\mathbf{x}_{1:k-2}} p(\mathbf{x}_{1:k-1}, \mathbf{z}_{1:k-1}) \right)
\]

- Pulling in the max

A recursion!
Inference: Most Likely Sequence

- Summarizing

\[
\max_{x_{1:k-1}} p(x_{1:k}, z_{1:k}) = p(z_k \mid x_k) \max_{x_{k-1}} \left( p(x_k \mid x_{k-1}) \max_{x_{1:k-2}} p(x_{1:k-1}, z_{1:k-1}) \right)
\]

- Let \( \mu_k = \max_{x_{1:k-1}} p(x_{1:k}, z_{1:k}) \), we have

\[
\mu_k = p(z_k \mid x_k) \max_{x_{k-1}} \left( p(x_k \mid x_{k-1}) \mu_{k-1} \right)
\]

- This result is very similar to the filtering equation

\[
\alpha_k = p(z_k \mid x_k) \sum_{x_{k-1}} p(x_k \mid x_{k-1}) \alpha_{k-1}
\]

where the summation over \( x \) is replaced by the maximization over \( x \)
Inference: Most Likely Sequence

- Thus, the algorithm for computing the most likely sequence is **similar to filtering**: it runs forward along the sequence and computes $\mu$ at each time step.

- At the end, it will have the probability for the most likely sequence reaching each of the final states.

- We also require the **initial** state

\[
\mu_1 = \max_{x_{1:0}} p(x_{1:1}, z_{1:1}) = p(x_1, z_1) = p(x_1) p(z_1 \mid x_1)
\]

which only depends on known HMM parameters.

- This procedure computes the probability of the most likely sequence. The sought **state sequence**, $x^*$, is obtained by storing pointers that, for each state, record the best state that leads to it. Finally, $x^*$ is obtained through **backtracking**.
Inference: Most Likely Sequence

- The algorithm computes
  \[ \mu_k = \max_{x_{1:k-1}} p(x_{1:k}, z_{1:k}) \] 

\[ \mu_k = p(z_k | x_k) \max_{x_{k-1}} \left( p(x_k | x_{k-1}) \mu_{k-1} \right) \]

in each step by maximizing over the previous \( \mu_{k,s} \) values for every state \( s \), weighted by the transition matrix \( A \), and then multiplying by the observation model for state \( s \), \( p(z_t | x_t = s) \)

- This is called the Viterbi algorithm after its inventor
- Like the filtering algorithm, its time complexity is also linear in the length of the sequence
Hidden Markov Models

Inference: Most Likely Sequence Example

- Let us illustrate the Viterbi algorithm in our umbrella example

\[ U_k = \text{true} \]
\[ \text{true} \]
\[ \text{false} \]
\[ \text{true} \]

Values of \( \mu_{k,s} \) which give the probability of the best sequence reaching each state at time \( k \) are shown in the boxes.

- Bold arrows indicate a state’s best predecessor as measured by the product of the preceding \( \mu_{k,s} \) probability and the transition probability.

- Backtracking: following the bold arrows back from the most likely state in \( \mu_5 \) gives the most likely sequence.
Inference: Most Likely Sequence Example

- In our simple human action recognition example, the **sequence of observations** is

- The **smoothed** probabilities

- The **ground truth**
Inference: Most Likely Sequence Example

- In our simple human action recognition example, the sequence of observations is

- The smoothed probabilities

- The ground truth

- The most likely sequence
Summary

- **Temporal reasoning** deals with the representation, inference and learning of and with sequential data.
- The **state space model** describes systems that evolve on their own, with observations of it occurring in a separate process.
- State space models possess **three parameters**: the transition model, the observation model, and the prior distribution.
- An HMM is a temporal probabilistic model in which the state of the process is described by a **single discrete** random variable.
- HMM is a very popular method and widely used in speech recognition, natural language modeling, human activity recognition, on-line handwriting recognition, analysis of protein/DNA sequences, etc.
- Many extensions of the basic HMMs exist.
Summary

- We have considered four HMM inference tasks: filtering, smoothing, prediction, and most likely sequence
  - Filtering is done with the forward algorithm
  - Smoothing is done with the forward-backward algorithm
  - Prediction is like filtering without new evidence
  - Computing the most likely sequence is done with the Viterbi algorithm

- They are all linear-time algorithms

- Smoothing provides better state estimates than filtering due to its incorporation of the “future”
  - Better in terms of less noise
  - Better in terms of shorter reaction times to state changes
Sources and Further Readings

These slides follow three sources: the books by Russell and Norvig [1] (chapter 15), Bishop [2] (chapter 13), and the videos by mathematicalmonk [3]. The seminal tutorial by Rabiner [4] is also highly recommendable.


