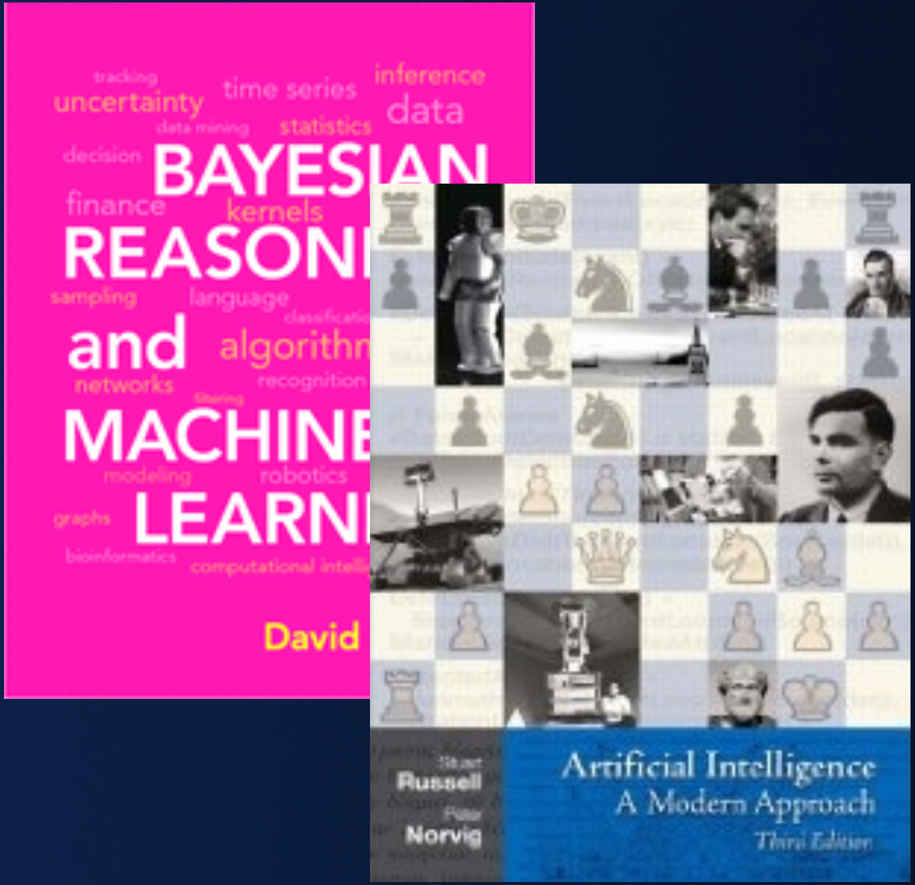


AI Fundamentals: uncertain reasoning

Maria Simi

Probabilistic Reasoning over time
AIMA chapter 15

LESSON 3: TIME AND UNCERTAINTY - INFERENCE IN TEMPORAL MODELS

Reasoning in a changing world

So far, we have been doing uncertain reasoning in a static world.

For reasoning in an evolving world an agent needs:

- A **belief state:** the states of the world that are possible
- A **transition model:** to predict how the world will evolve
- A **sensor model:** to update the belief state from perceptions

A changing world is modeled using a variable for each aspect of the world state *at each point in time* (fluents). We use probabilities to quantify how likely a world is.

The transition and sensor model themselves may be uncertain:

- the *transition model* gives the probability distribution of the variables at time *t*, given the state of the world at past times;
- the *sensor model* describes the probability of each percept at time *t*, given the current state of the world.

Examples

1. **Treating a diabetic patient.** The task is to assess the current state of the patient, including the actual blood sugar level and insulin level from other observable parameters such as recent insulin doses, food intake, blood sugar measurements, and other physical signs.

   Dynamic aspects are essential for a correct prediction and prescription of insulin and food, in this case as in many other real-world scenarios.

2. **The umbrella example, our toy example.**

   You are the security guard stationed at a secret underground installation. You want to know whether it's raining today, but your only access to the outside world occurs each morning when you see the director coming in with, or without, an umbrella.

States and observations

We view the world as a series of *snapshots*, or **time slices**, each of which contains a set of random variables, some observable and some not.

$\mathbf{X}_t$: will denote the set of **state variables**, unobservable (hidden) at time t

$\mathbf{E}_t = \boldsymbol{e}_t$ will denote the set of observations (**evidence variables**) at time t; $\boldsymbol{e}_t$ their values

We will assume that the state sequence starts at t=0. The distance between time slices is fixed.

We will use the notation a:b to denote the sequence of integers from a to b (inclusive), and the notation $\mathbf{X}_{a:b}$ to denote the set of variables from $\mathbf{X}_a$ to $\mathbf{X}_b$

Our umbrella world is represented by a sequence of evidence variables $\mathbf{E}_t = \{U_t\}$ (whether the umbrella appears at time t) and state unobservable variables $\mathbf{X}_t = \{R_t\}$ (it's raining):

state variables: R_0, R_1, R_2, …

evidence variables: U_1, U_2, …

Note: state variable start at 0, evidence variables at time 1.

Transition model and simplifying assumptions

The transition model specifies **how the world evolves**, i.e. the probability distribution over the latest state variables, given the previous values starting from time 0:

$\mathbf{P}(\mathbf{X}_t \mid \mathbf{X}_{0:t-1})$ $\mathbf{X}_{0:t-1} = X_0, X_1, \ldots X_{t-1}$

The sequence of states can become very large, unbounded as t increases.

Markov assumptions: the transition model specifies the probability distribution over the latest state variables, given a **finite fixed number** of previous states:

$\mathbf{P}(\mathbf{X}_t \mid \mathbf{X}_{0:t-1}) = \mathbf{P}(\mathbf{X}_t \mid \mathbf{X}_{t-1})$ *first order* Markov process/chain

$\mathbf{P}(\mathbf{X}_t \mid \mathbf{X}_{0:t-1}) = \mathbf{P}(\mathbf{X}_t \mid \mathbf{X}_{t-1}, \mathbf{X}_{t-2})$ *second order* Markov process/chain

Additionally, we assume a **stationary process**, i.e. the conditional probability distribution is the same for all t.

$\mathbf{P}(\mathbf{X}_t \mid \mathbf{X}_{t-1})$ is the same for all t $\mathbf{P}(R_t \mid r_{t-1}) = \langle 0.7, 0.3 \rangle$

Change is governed by laws that do not themselves change over time.

Transition model - graphically

Bayes networks under different Markov [independence] assumptions

(a)
$$\begin{tikzcd}
{} \arrow[r] & \mathbf{X}_{t-2} \arrow[r] & \mathbf{X}_{t-1} \arrow[r] & \mathbf{X}_{t} \arrow[r] & \mathbf{X}_{t+1} \arrow[r] & \mathbf{X}_{t+2} \arrow[r] & {}
\end{tikzcd}$$

(b)
$$\begin{tikzcd}
{} \arrow[r] \arrow[rr, bend left] & \mathbf{X}_{t-2} \arrow[r] \arrow[rr, bend left] & \mathbf{X}_{t-1} \arrow[r] \arrow[rr, bend left] & \mathbf{X}_{t} \arrow[r] \arrow[rr, bend left] & \mathbf{X}_{t+1} \arrow[r] \arrow[rr, bend left] & \mathbf{X}_{t+2} \arrow[r] & {}
\end{tikzcd}$$

a. *first order* Markov process/chain
b. *second order* Markov process/chain

Sensor model assumption

The sensor/observation model, under **Markov sensor assumption**, postulates that evidence only depends on the current state

$$\mathbf{P}(\mathbf{E}_t \mid \mathbf{X}_{0:t}, \mathbf{E}_{0:t-1}) = \mathbf{P}(\mathbf{E}_t \mid \mathbf{X}_t)$$

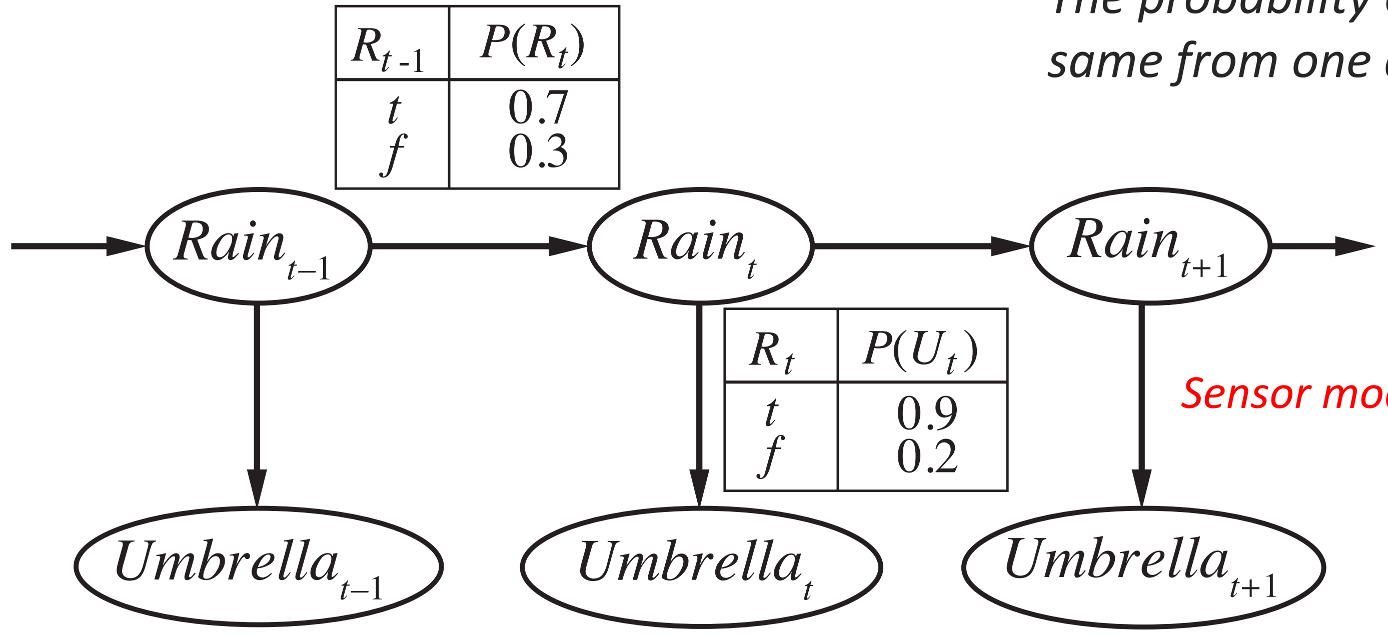
This is a reasonable assumption to make, given the availability of sensors.

Releasing the assumptions:

Assuming that 'Rain' only depends on rain the previous day may be a too strong assumption. There are two ways to improve the accuracy of the approximation:

1. Increasing the order of the Markov process model, for example using a second order assumption.
2. Increasing the set of state variables: we could add *Season, Temperature, Humidity, Pressure* … as state variables. This may imply more computation for predicting state variables or adding new sensors.

The Bayesian net for the *Umbrella* example



R_{t-1}	$P(R_t)$
t	0.7
f	0.3

R_t	$P(U_t)$
t	0.9
f	0.2

The probability of the rain situation being the same from one day to the next is 70%

Transition model, in a 1st order Markov model

Sensor model

The transition model is $P(R_t \mid R_{t-1})$

The sensor model is $P(U_t \mid R_t)$

Rain depends on rain in the previous day
The rain causes the umbrella to appear

Note: This network is a *polytree*

Complete joint distribution

We also need the prior probability distribution at time 0, $\mathbf{P}(\mathbf{X}_0)$

Putting all together, the **complete joint distribution** over all the variables, for any t, computed from the network is:

$$\mathbf{P}(\mathbf{X}_{0:t}, \mathbf{E}_{1:t}) = \mathbf{P}(\mathbf{X}_0) \prod_{i=1}^{t} \mathbf{P}(\mathbf{X}_i | \mathbf{X}_{i-1})\ \mathbf{P}(\mathbf{E}_i | \mathbf{X}_i)$$

(this formula comes from $P(x_1, \dots, x_n) = \prod_{i=1}^{n} P(xi \mid parents(Xi))$)

$\mathbf{P}(\mathbf{X}_0)$	the initial state model over set of variables **X**
$\mathbf{P}(\mathbf{X}_i \| \mathbf{X}_{i-1})$	the transition model under a *first-order Markov assumption*
$\mathbf{P}(\mathbf{E}_i \| \mathbf{X}_i)$	the sensor model under *Markov sensor assumption*

Inference in temporal models

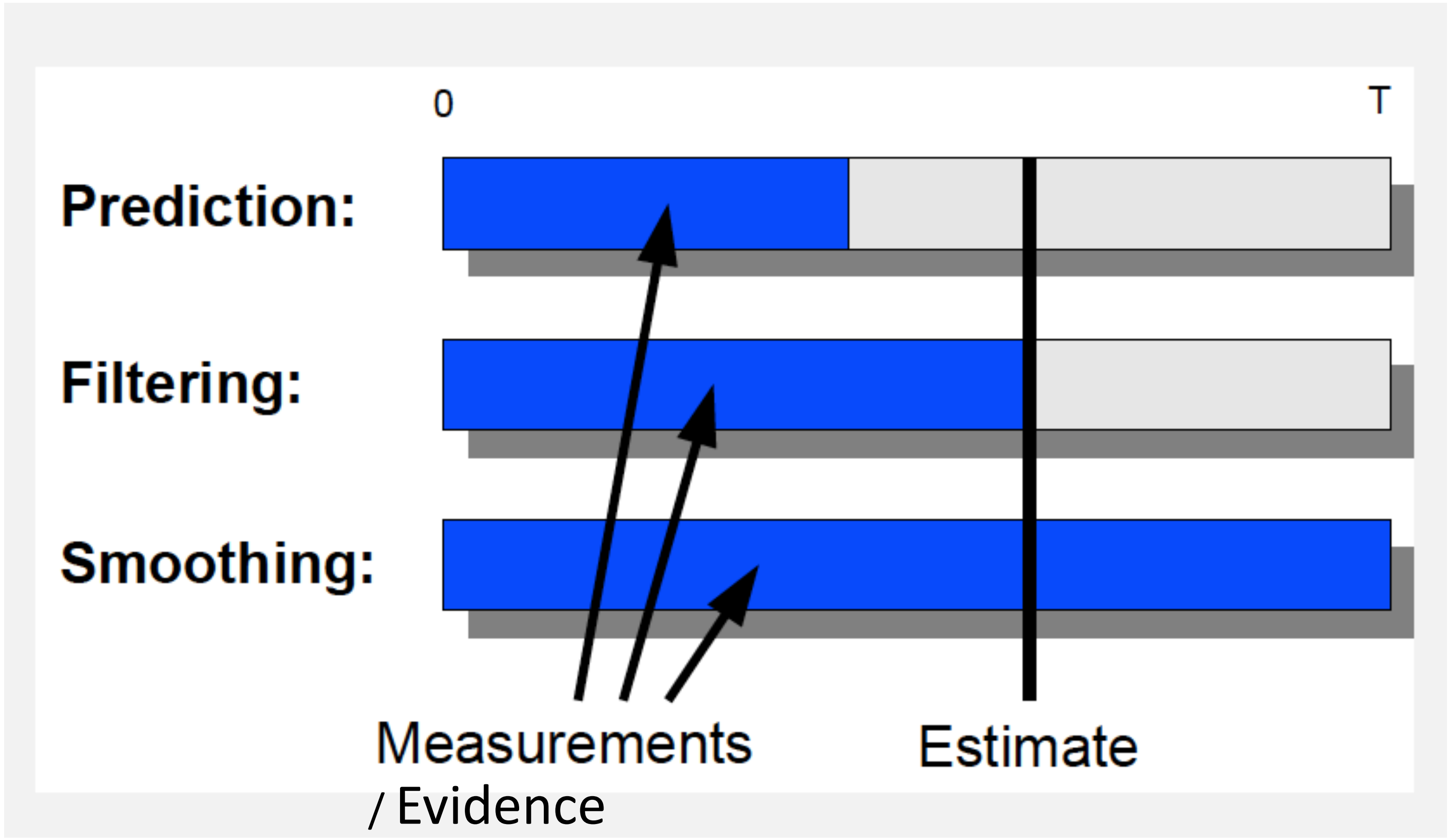
FILTERING, PREDICTION, SMOOTHING, MOST LIKELY EXPLANATION

Inference in temporal models

Basic inference tasks based on the temporal model:

- **Filtering**, or **state estimation**: computing the belief state (*posterior probability distribution* of state variables) given evidence from previous and current states. A subtask is the **likelihood of the evidence sequence**. $\mathbf{P}(\boldsymbol{X}_t|\boldsymbol{e}_{1:t})$
- **Prediction:** computing the posterior distribution over a **future** state, given all evidence to date. $\mathbf{P}(\boldsymbol{X}_{t+\mathrm{k}}|\boldsymbol{e}_{1:t})$ with $k > 1$.
- **Smoothing:** computing the posterior distribution over a **past** state, given all evidence up to the present. Looking back with the knowledge of today provides a more accurate estimate. $\mathbf{P}(\boldsymbol{X}_{\mathrm{k}}|\boldsymbol{e}_{1:t})$ with $0 \leq k < t$.
- **Most likely explanation/sequence**: given a sequence of observations, we might wish to find the sequence of states that is most likely to have generated those observations.
- **Learning**: The transition and sensor models, can be learned from observations.

Filtering vs other inference tasks



From a tutorial by Simo Särkkä

Filtering: estimation of next state

A good filtering algorithm maintains a **current state estimate** and updates it, rather than going back over the entire history of percepts for each time

The filtering function f takes into account **the state estimation computed up to the present** and the **new evidence.**

$$\mathbf{P}(\boldsymbol{X}_{t+1}|\boldsymbol{e}_{1:t+1}) = f(\boldsymbol{e}_{t+1}, \mathbf{P}(\boldsymbol{X}_t|\boldsymbol{e}_{1:t}))$$

This process is called **recursive estimation** and is made of two parts:

1. **Prediction**: the current state distribution is projected forward from t to $t+1$: $\mathbf{P}(\boldsymbol{X}_{t+1}|\boldsymbol{e}_{1:t})$
2. **Update**: then it is updated using the new evidence e_{t+1}: $\mathbf{P}(\boldsymbol{e}_{t+1}|\boldsymbol{X}_{t+1})$

Computing filtering

It can be computed as follows:

$\mathbf{P}(\boldsymbol{X}_{t+1} \mid \boldsymbol{e}_{1:t+1}) = \mathbf{P}(\boldsymbol{X}_{t+1} \mid \boldsymbol{e}_{1:t}, \boldsymbol{e}_{t+1}) =$	*Dividing up the evidence*
$= \alpha\, \mathbf{P}(\boldsymbol{e}_{t+1} \mid \boldsymbol{X}_{t+1}\, \boldsymbol{e}_{1:t})\, \mathbf{P}(\boldsymbol{X}_{t+1} \mid \boldsymbol{e}_{1:t})$	*Bayes's rule, focusing on* $\boldsymbol{e}_{t+1}$ $\rightarrow$
$= \alpha\, \mathbf{P}(\boldsymbol{e}_{t+1} \mid \boldsymbol{X}_{t+1})\, \mathbf{P}(\boldsymbol{X}_{t+1} \mid \boldsymbol{e}_{1:t})$	*Markov's sensor assumption*
$= \alpha\, \mathbf{P}(\boldsymbol{e}_{t+1} \mid \boldsymbol{X}_{t+1}) \sum_{\boldsymbol{x}_t} \mathbf{P}(\boldsymbol{X}_{t+1} \mid \boldsymbol{x}_t, \boldsymbol{e}_{1:t})\, \mathbf{P}(\boldsymbol{x}_t \mid \boldsymbol{e}_{1:t})$	*By conditioning on the current state* $\boldsymbol{X}_t$ $\rightarrow$
$= \alpha\, \mathbf{P}(\boldsymbol{e}_{t+1} \mid \boldsymbol{X}_{t+1}) \sum_{\boldsymbol{x}_t} \mathbf{P}(\boldsymbol{X}_{t+1} \mid \boldsymbol{x}_t)\, \mathbf{P}(\boldsymbol{x}_t \mid \boldsymbol{e}_{1:t})$	*Markov's sensor assumption*

Finally: $\mathbf{P}(\boldsymbol{X}_{t+1} \mid \boldsymbol{e}_{1:t+1}) = \alpha\, \underbrace{\mathbf{P}(\boldsymbol{e}_{t+1} \mid \boldsymbol{X}_{t+1})}_{\text{----update---}} \underbrace{\sum_{\boldsymbol{x}_t} \mathbf{P}(\boldsymbol{X}_{t+1} \mid \boldsymbol{x}_t)\, \mathbf{P}(\boldsymbol{x}_t \mid \boldsymbol{e}_{1:t})}_{\text{---one-step prediction----}}$ (*Filtering*)

We can think of $\mathbf{P}(\boldsymbol{X}_t \mid \boldsymbol{e}_{1:t})$ as a *message* $f_{1:t}$ that is propagated forward in the sequence. This makes evident the recursive structure:

$f_0 = \mathbf{P}(X_0)$

$f_{1:t+1} = \alpha\, \mathit{Forward}(f_{1:t}, \boldsymbol{e}_{t+1})$

where *Forward* implements the *filtering* update in **costant time**

Reminder

$$\mathbf{P}(X \mid Y, Z) = \frac{\mathbf{P}(Y \mid X, Z)\,\mathbf{P}(Y \mid Z)}{\mathbf{P}(X \mid Z)}$$ (*Bayes rule with background knowledge Z*)

In our case:

$$\mathbf{P}(\boldsymbol{X}_{t+1} \mid \boldsymbol{e}_{1:t}, \boldsymbol{e}_{t+1}) = \alpha\, \mathbf{P}(\boldsymbol{e}_{t+1} \mid \boldsymbol{X}_{t+1}\, \boldsymbol{e}_{1:t})\, \mathbf{P}(\boldsymbol{X}_{t+1} \mid \boldsymbol{e}_{1:t})$$

Conditioning is marginalization applied to conditional probabilities

--- product rule ---

$$\mathbf{P}(\mathbf{Y}) = \sum_{z\,\in \mathbf{Z}} \mathbf{P}(\mathbf{Y}, z) = \sum_{z \in \mathbf{Z}} \mathbf{P}(\mathbf{Y} \mid z)\, P(z)$$ (*Conditioning*)

In our case:

$$\mathbf{P}(\boldsymbol{X}_{t+1} \mid \boldsymbol{e}_{1:t}) = \sum_{\boldsymbol{x}_t} \mathbf{P}(\boldsymbol{X}_{t+1} \mid \boldsymbol{x}_t, \boldsymbol{e}_{1:t})\ \mathbf{P}(\boldsymbol{x}_t \mid \boldsymbol{e}_{1:t})$$

Computing filtering for umbrellas

The example shows the consequences of observing umbrellas for two consecutive days, $\mathbf{P}(R_2|u_1,u_2)$

Day 0: $\mathbf{P}(R_0) = \langle 0.5, 0.5\rangle$ $\quad f_0$ no observation, a 50% chance of rain

Day 1:

Predict: $\mathbf{P}(R_1) = \sum_{r_0} \mathbf{P}(R_1 | r_0)\, P(r_0) = \mathbf{P}(R_1|r_0) \times P(r_0) + \mathbf{P}(R_1|\neg r_0) \times P(\neg r_0) =$

$= \langle 0.7, 0.3\rangle \times 0.5 + \langle 0.3, 0.7\rangle \times 0.5 = \langle 0.5, 0.5 \rangle$

Update, observing the evidence u_1 ($U_1 = \text{true}$)

$\mathbf{P}(R_1|u_1)=\alpha\, \mathbf{P}(u_1|R_1)\, \mathbf{P}(R_1)=\alpha \langle 0.9, 0.2\rangle \langle 0.5, 0.5\rangle= \alpha \langle 0.45, 0.1\rangle \approx \overset{f_{1:1}}{\langle 0.818, 0.182\rangle}$

Day 2:

Predict: $\mathbf{P}(R_2) = \sum_{r_1} \mathbf{P}(R_2 | r_1)\, P(r_1) = \langle 0.7, 0.3\rangle \times 0.818 + \langle 0.3, 0.7\rangle \times 0.182 = \langle 0.627, 0.373\rangle$

Update, observing the evidence u_2

$\mathbf{P}(R_2|u_1,u_2)=\alpha\, \mathbf{P}(u_2|R_2)\, \mathbf{P}(R_2|u_1)=\alpha \langle 0.9, 0.2\rangle \langle 0.627, 0.373\rangle= \alpha \langle 0.565, 0.075\rangle \approx \overset{f_{1:2}}{\langle 0.883, 0.117\rangle}$

The probability of rain increases from day 1 to day 2 because rain persists.

Prediction

The task of prediction can be seen simply as filtering without the contribution of new evidence. The filtering process already incorporates a **one-step** prediction.

In general, looking ahead k steps, at time $t+k+1$, given evidence up to t:

$$\mathbf{P}(\boldsymbol{X}_{t+k+1}|\boldsymbol{e}_{1:t}) = \sum_{x_{t+k}} \mathbf{P}(\boldsymbol{X}_{t+k+1}|x_{t+k})\ \mathbf{P}(x_{t+k}|\boldsymbol{e}_{1:t})$$ *… looking k steps ahead*

This computation involves only *the transition model* and **not** the *sensor model*.

We can show that the predicted distribution for $Rain$ converges to a fixed point $\langle 0.5, 0.5\rangle$, after which it remains constant for all time (the **stationary distribution** of the Markov process). The ***mixing time*** is the time to reach the fixed point.

The more uncertainty there is in the transition model, the shorter will be the *mixing time* and the more the future is obscure.

Likelihood of evidence sequence (skip)

We can use a forward recursion also to compute the likelihood of an evidence sequence $P(\boldsymbol{e}_{1:t})$, useful if we want to compare two models producing the same evidence sequence.

We can derive a recursive equation similar to filtering and a similar likelihood message

$$\ell_{1:\mathrm{t}}(X_{\mathrm{t}}) = P(X_{\mathrm{t}}, e_{1:\mathrm{t}})$$

Once we have $\ell_{1:\mathrm{t}}(X_{\mathrm{t}})$ we can compute the likelihood of the evidence sequence by summing out on the values of X_t

$$\mathrm{L}_{1:\mathrm{t}} = P(\boldsymbol{e}_{1:\mathrm{t}}) = \sum_{x_t} \ell_{1:\mathrm{t}}(x_t)$$

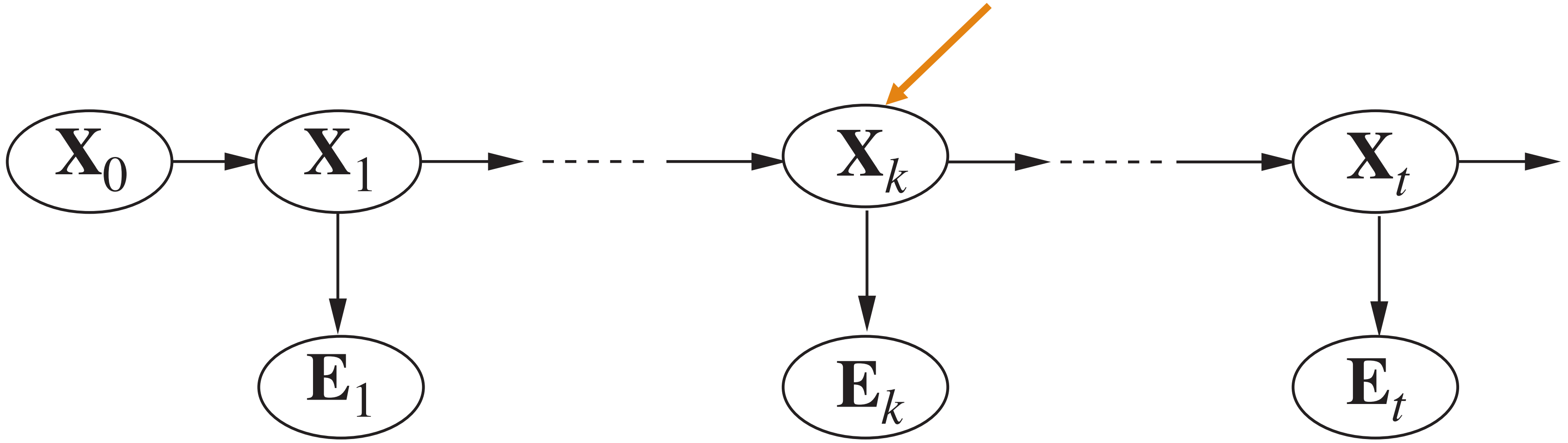
Note: the likelihood becomes smaller and smaller as t increases.

Smoothing

Smoothing is the process of computing the posterior distribution of the state at **some past time** k given a complete sequence of observations up to the present t

$$\mathbf{P}(\boldsymbol{X}_k | \boldsymbol{e}_{1:t}) \qquad \text{for } 0 \leq k < t$$

The additional evidence is expected to provide more information and more accurate predictions on the past … The Bayesian network is:



Computing smoothing

$\mathbf{P}(\boldsymbol{X}_k|\boldsymbol{e}_{1:t}) = \mathbf{P}(\boldsymbol{X}_k|\boldsymbol{e}_{1:k}, \boldsymbol{e}_{k+1:t})$ *Splitting the evidence*

$= \alpha\, \mathbf{P}(\boldsymbol{X}_k|\boldsymbol{e}_{1:k})\, \mathbf{P}(\boldsymbol{e}_{k+1:t}|\boldsymbol{X}_k, \boldsymbol{e}_{1:k})$ *Bayes's rule*

$= \alpha\, \mathbf{P}(\boldsymbol{X}_k|\boldsymbol{e}_{1:k})\, \mathbf{P}(\boldsymbol{e}_{k+1:t}|\boldsymbol{X}_k)$ *Conditional independence*

$= \alpha\, f_{1:k} \times b_{k+1:t}$ $\times$ is pointwise multiplication

$\mathbf{P}(\boldsymbol{X}_k|\boldsymbol{e}_{1:k})$ corresponda to the *forward* message $f_{1:k}$, computed up to *t,* as before in filtering.

We define another *message* $b_{k+1:t} = \mathbf{P}(\boldsymbol{e}_{k+1:t}|\, \boldsymbol{X}_k)$ that can be computed by a recursive process that runs **backward** from t (next slide).

The terms in $f_{1:k}$and $b_{k+1:t}$ can be implemented by two recursive calls

- one running forward from 1 to k and using the *filtering equation*
- the other running backward from t to $k+1$ for computing $\mathbf{P}(\boldsymbol{e}_{k+1:t}|\boldsymbol{X}_k)$

Computing smoothing going backwards

$\mathbf{P}(\mathbf{e}_{k+1:t} \mid \mathbf{X}_k) = \sum_{\mathbf{x}_{k+1}} \mathbf{P}(\mathbf{e}_{k+1:t} \mid \mathbf{X}_k, \mathbf{x}_{k+1})\, \mathbf{P}(\mathbf{x}_{k+1} \mid \mathbf{X}_k)$ *conditioning on* $\mathbf{X}_{k+1}$

$= \sum_{\mathbf{x}_{k+1}} \mathbf{P}(\mathbf{e}_{k+1:t} \mid x_{k+1})\, \mathbf{P}(\mathbf{x}_{k+1} \mid \mathbf{X}_k)$ *the evidence only depends on* $\mathbf{x}_{k+1}$

$= \sum_{\mathbf{x}_{k+1}} \mathbf{P}(\mathbf{e}_{k+1}, \mathbf{e}_{k+2:t} \mid x_{k+1})\, \mathbf{P}(\mathbf{x}_{k+1} \mid \mathbf{X}_k)$ *splitting the evidence*

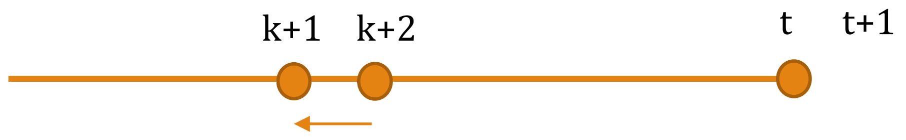
$= \sum_{\mathbf{x}_{k+1}} P(\mathbf{e}_{k+1} \mid x_{k+1})\, P(\mathbf{e}_{k+2:t} \mid x_{k+1})\, \boldsymbol{P}(\mathbf{x}_{k+1} \mid \mathbf{X}_k)$ *conditional independence*

$b_{k+1:t} = P(\mathbf{e}_{k+2:t} \mid x_{k+1})$ is defined by the previous equation computing backword.

The backward phase starts with $\mathbf{P}(\mathbf{e}_{t+1:t} \mid \mathbf{X}_t) = \mathbf{1}$ **(a vector of 1's)**, because $(t+1:t)$ is an empty sequence of evidence with probability 1 of observing it.

$b_{t+1:t} = \mathbf{1}$

$b_{k+1:t} = Backward(b_{k+2:t}, \mathbf{e}_{k+1})$



where *Backward* implements the update defined above

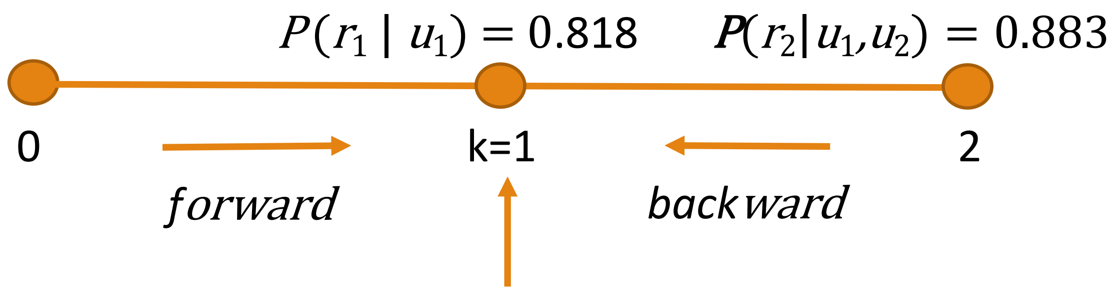
Computing the smoothed estimate for *rain*

Let's compute the smoothed estimate at time k=1 for the probability of rain. Given the observations at time 1 and 2.

$$\mathbf{P}(R_1 \mid u_1, \underline{u}_2) = \alpha\, \mathbf{P}(R_1 \mid u_1) \times \mathbf{P}(u_2 \mid R_1)$$

$$\mathbf{P}(R_1 \mid u_1) = \langle 0.818, 0.182\rangle$$

already computed in the filtering stage.



$$\mathbf{P}(u_2 \mid R_1) = \textstyle\sum_{r_2} P(u_2 \mid r_2)\, P(U_{3:2} \mid r_2)\, \mathbf{P}(r_2 \mid R_1)$$

$$= (P(u_2 \mid r_2) \times \mathbf{1} \times \mathbf{P}(r_2 \mid R_1)) + (P(u_2 \mid \neg r_2) \times \mathbf{1} \times \mathbf{P}(\neg r_2 \mid R_1))$$

$$= (0.9 \times 1 \times \langle 0.7, 0.3\rangle) + (0.2 \times 1 \times \langle 0.3, 0.7\rangle) = \langle 0.69, 0.41\rangle$$

$$\mathbf{P}(R_1 \mid u_1, \underline{u}_2) = \alpha \langle 0.818, 0.182\rangle \times \langle 0.69, 0.41\rangle \approx \langle 0.883, 0.117\rangle$$

The **smoothed estimate** for rain on day 1 is higher than the *filtered estimate*. Rain persists.

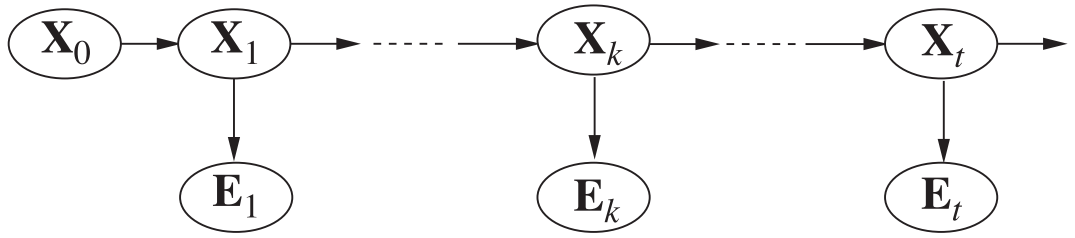
The complexity for smoothing at a specific time k with respect to evidence $e_{1:t}$ is O(t). For smoothing a sequence O(t^2). A better linear approach extsts.

Improving efficiency

The algorithm, called **forward–backward**, uses dynamic programming to reduce the complexity of running the algorithm over the whole sequence.

The trick is to record the results computed during the forward filtering phase, over the whole sequence, and reuse them during the backward phase.

The complexity is O(t) and is consistent with the fact that the Bayesian network structure is a **polytree**.

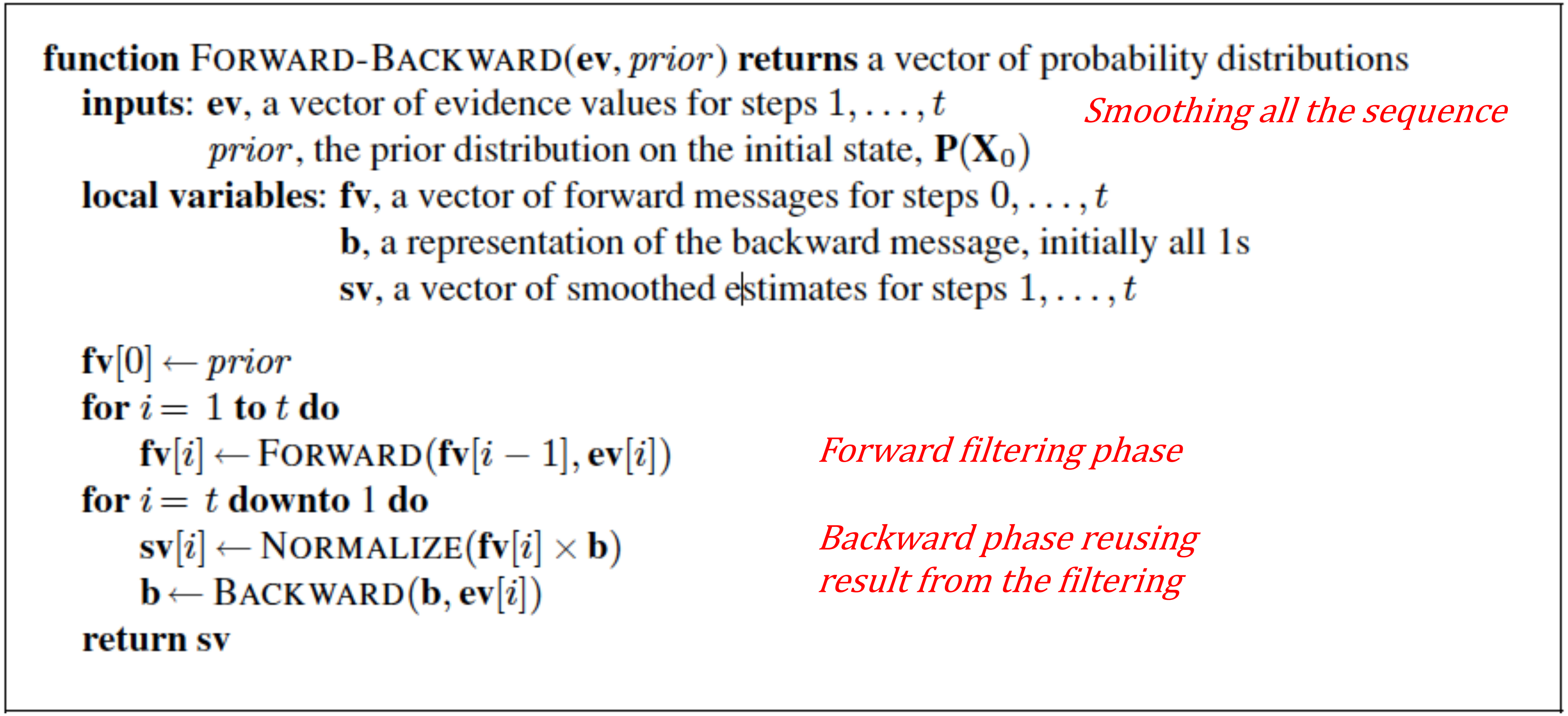


The *forward–backward* algorithm is the basis for many applications that deal with sequences of noisy observations.

Improvements are required to deal with

1. *space complexity* for long sequences
2. *online computations*, where new observations continuously arrive (**fixed-lag smoothing**)

Forward–backward algorithm for smoothing



```
function FORWARD-BACKWARD(ev, prior) returns a vector of probability distributions
   inputs: ev, a vector of evidence values for steps $1, \ldots, t$
           prior, the prior distribution on the initial state, $\mathbf{P}(\mathbf{X}_0)$
   local variables: fv, a vector of forward messages for steps $0, \ldots, t$
                    b, a representation of the backward message, initially all 1s
                    sv, a vector of smoothed estimates for steps $1, \ldots, t$

   fv[0] ← prior
   for $i = 1$ to $t$ do
      fv[i] ← FORWARD(fv[i − 1], ev[i])
   for $i = t$ downto 1 do
      sv[i] ← NORMALIZE(fv[i] × b)
      b ← BACKWARD(b, ev[i])
   return sv
```

Smoothing all the sequence

Forward filtering phase

Backward phase reusing result from the filtering

Finding the most likely sequence

Suppose we observe [*true, true, false, true, true*] for the *Umbrella* variable.

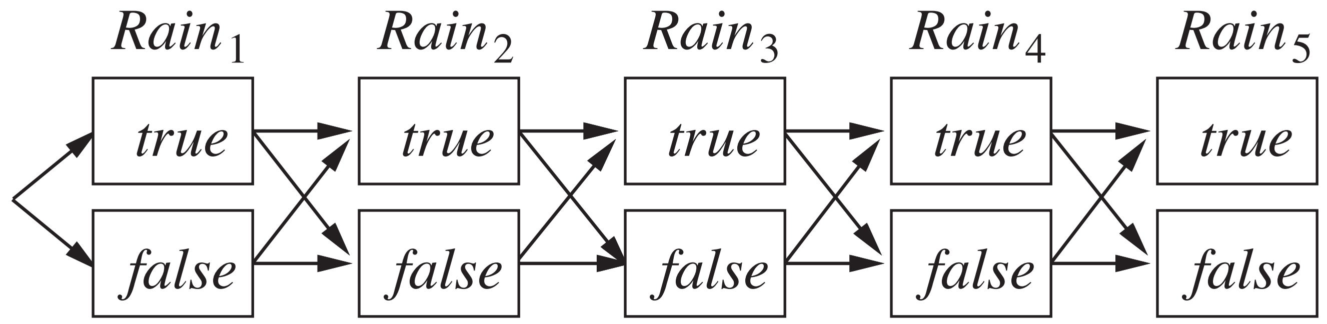
What is the most likely sequence for *Rain* given these oservations?

There are 2^5 possible *Rain* sequences …Each sequence is a path through a graph whose nodes are the possible states at each time step.

We want to discover which is the one maximizing the likelihood, **in linear time**.

Likelihood of a path: product of the transition probabilities along the path and the probabilities of the given observations at each state

(a)



Trellis diagram

Square boxes are states

Computing the most likely sequence

There is a recursive relationship between the most likely path to each state X_{t+1} and most likely paths to each previous state x_t. We can write a recursive equation, similar to the one for filtering:

$$max_{x_1 \dots xt} \mathbf{P}(\mathbf{x}_1, \dots, \mathbf{x}_t, X_{t+1} \mid \mathbf{e}_{1:t+1}) =$$

$$= \alpha\, \mathbf{P}(\mathbf{e}_{t+1} \mid \mathbf{X}_{t+1})\, max_{x_t} (\mathbf{P}(\mathbf{X}_{t+1} \mid \mathbf{x}_t)\, max_{x_1 \dots x_{t-1}} P(\mathbf{x}_1, \dots, \mathbf{x}_{t-1}, \mathbf{x}_t \mid \mathbf{e}_{1:t}))$$

The forward message in this case is

$$\mathbf{m}_{1:t} = max_{x_1 \dots x_{t-1}} \mathbf{P}(\mathbf{x}_1, \dots, \mathbf{x}_{t-1}, \mathbf{X}_t \mid \mathbf{e}_{1:t})$$

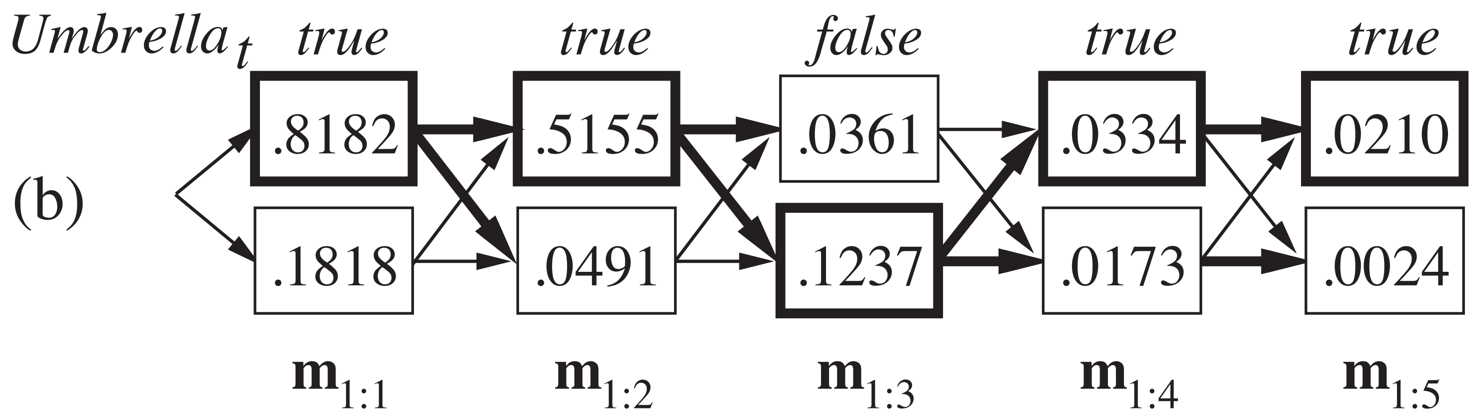
the probabilities of the best path to each state X_t; from those we can compute the probabilities of the extended paths at time t+1 and take the max.

The most likely sequence overall can be computed in one pass.

For each state, the best state that leads to it is recorded (marked as black arrows in the example) so that the optimal sequence is identified by following black arrows backwards from the best final state.

This algorithm is the famous **Viterbi algorithm**, named after A. Viterbi [1967]

Viterbi algorithm on the Umbrella example



$\mathbf{m}_{1:1} = \langle 0.8182, 0.1818 \rangle$ as computed in filtering

$\mathbf{m}_{1:2} = \langle \max[P(u_2 \mid r_2) \times 0.8182 \times P(r_2 \mid r_1),\ P(u_2 \mid r_2) \times 0.1818 \times P(r_2 \mid \neg r_1)],$ *rain at time 2*

$\max[P(u_2 \mid \neg r_2) \times 0.8182 \times P(\neg r_2 \mid r_1), P(u_2 \mid \neg r_2) \times 0.1818 \times P(\neg r_2 \mid \neg r_1)] \rangle$ *not rain at time 2*

$= \langle \max[0.9 \times 0.8182 \times 0.7, 0.9 \times 0.1818 \times 0.3], \max[0.2 \times 0.8182 \times 0.3, 0.2 \times 0.1818 \times 0.7] \rangle$

$= \langle \max[0.5155, 0.0491], \max[0.0491, 0.0255] \rangle = \langle 0.5155, 0.0491 \rangle$

$\mathbf{m}_{1:3} = P(\neg u_3 \mid R_3) \times \max_{r2}[P(R_3 \mid r_2) \times \mathbf{m}_{1:2}] = \langle 0.361, 0.1237 \rangle$

$\mathbf{m}_{1:4} = P(u_4 \mid R_4) \times \max_{r3} [P(R_4 \mid r_3) \times \mathbf{m}_{1:3}] = \langle 0.334, 0.173 \rangle$

$\mathbf{m}_{1:5} = P(u_5 \mid R_5) \times \max_{r4} [P(R_5 \mid r_4) \times \mathbf{m}_{1:4}] = \langle 0.0210, 0.0024 \rangle$

Applications of the Viterbi algorithm

- The original application of Viterbi was in telecommunications: Viterbi decoders are used for decoding a bitstream encoded using a technique called **convolutional code** or **trellis code**.
- In natural language processing:
  - Different kinds of “sequence tagger”: PoS taggers, NE taggers …
  - Dependency parsing
  - Speech recognition (speech-to-text), speech synthesis, speech diarization (recognizing sequences from the same speaker), keyword spotting …
- …

Other approaches to uncertain reasoning

An overview of other approaches

Probability theory and Bayesian networks are the dominant approaches today, despite the disadvantage of having to specify many probability values (lots of numbers).

Open issue: Humans reason with numbers? Or their reasoning is more "qualitative"?

Other approaches have been used:

1. Rule-based methods for uncertain reasoning in expert systems
2. Representing ignorance: Dempster–Shafer theory
3. Representing vagueness: fuzzy sets and fuzzy logic

Rule-based methods for uncertain reasoning

Three good properties of classical logic-based rules:

1. **Locality**: In logical systems, from A and A ⇒ B, we can conclude B, without worrying about any other rules. In probabilistic systems, we need to consider all the evidence.
2. **Detachment**: Once B is proved, it can be used regardless of how it was derived. It can be *detached* from its justification. Probabilities cannot be detached from evidence.
3. **Truth-functionality**: the truth of complex sentences can be computed from the truth of the components. Probability combination does not work this way.

   Ex $P(H_1) = 0.5$, $P(H_2)=0.5$ but $P(H_1 \vee H_2) = 0.75$

The idea is to attach **degree of belief** to facts and rules and to combine and propagate them

The most famous example is the **certainty factors model**, which was developed for the MYCIN medical diagnosis program. The system was carefully engineered to avoid mixing different kind of rules (diagnostic vs causal), trying to control non plausible results.

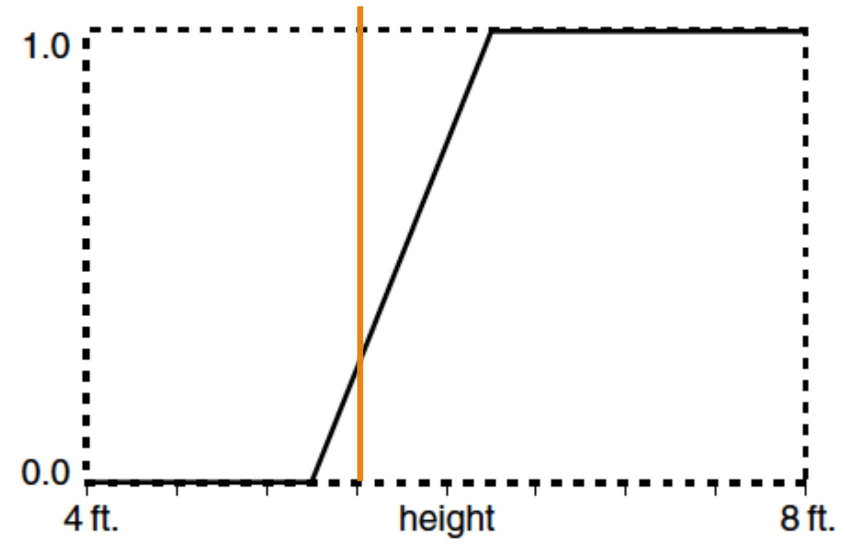
Part of the mechanism was later explained in terms of a limited form of probabilistic reasoning.

Fuzzy set theory: representing vagueness

The theory of **fuzzy sets** is a way to specify **how well** an object satisfies a vague description (non categorical) property, like "being tall".

- ✓ Crispy sets: Nate is tall is True or False. Membership function: $\boldsymbol{\mu}$(S) in {0, 1}
- ✓ Fuzzy sets: Nate is 1.75m. Is Nate tall? Sort of … the are degrees of tallness.
  - Membership function: $\boldsymbol{\mu}$(S) in [0, 1].
  - 0 means definetely not tall; 1 definetely tall.
  - The set does not have sharp boundaries.

Fuzziness id not uncertainty in the world, uncertainty in the use of qualifiers/properties.



Relation between degree of 'tallness' and height

Fuzzy logic

Fuzzy logic is way of reasoning about membership in fuzzy sets.

A fuzzy predicate implicitly defines a **fuzzy set**.

Fuzzy logic is a method for reasoning with logical expressions describing membership in fuzzy sets.

Standard rules:

$\mathrm{T}(A \wedge B) = min(\mathrm{T}(A), \mathrm{T}(B))$ *A, B are assertions or properties*

$\mathrm{T}(A \vee B) = max(\mathrm{T}(A), \mathrm{T}(B))$ *they denote degree of membership in a set*

$\mathrm{T}(\neg A) = 1 - \mathrm{T}(A)$

Example. $\mathrm{T}(Tall(john) \wedge Heavy(john)) = \mathrm{MIN}(\mathrm{T}(Tall(john)), \mathrm{T}(Heavy(john)))$

Fuzzy control

Fuzzy control is a methodology for constructing control systems in which the mapping between real-valued input and output parameters is represented by fuzzy rules.

1. **Fuzzification**: continous variables are mapped into a set of linguistic variables with fuzzy values; *temperature* can be mapped in fuzzy variables such as *low, medium, high* and their membership functions

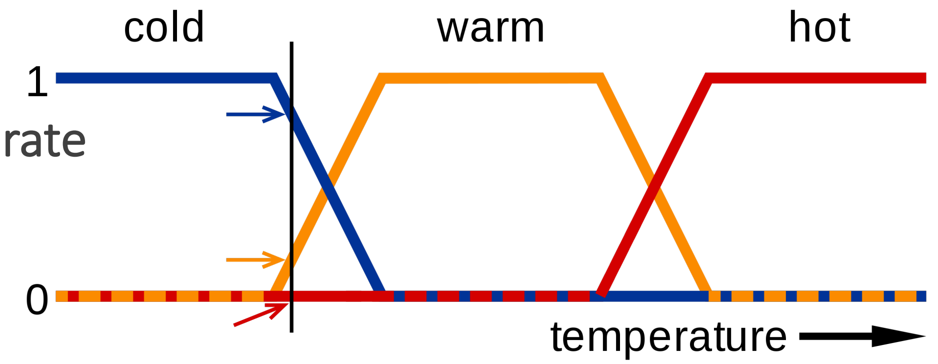
2. **Reasoning** with rules expressed in terms of these fuzzy variables, reaching fuzzy conclusions.

   IF temperature IS cold THEN fan_speed is slow

   IF temperature IS warm THEN fan_speed is moderate

   IF temperature IS hot THEN fan_speed is high

1. **Defuzzification**: map the results into numerical values



Conclusions

- ✓ A dynamic world can be represented using a set of random variables to represent the state at each point in time. A special case of Bayesian network.
- ✓ A temporal probability model can be thought of as containing a **transition model** describing the state evolution and a **sensor model** describing the observation process.
- ✓ Assuming the **Markov property** and **stationary processes** greatly simplifies the representation and the computation.
- ✓ The principal inference tasks in temporal models are: **filtering**, **prediction**, and **computing the most likely explanation sequence**. Simple, recursive algorithms exist *linear* in the length of the sequence.
- ✓ Algorithms can be further simplified assuming **Hidden Markov Models**, i.e. states are represented by a single discrete variable.
- ✓ **Hidden Markov models**, **Kalman filters**, and **dynamic Bayesian networks** are left out.

References

✓ Stuart J. Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach* (3rd edition). Pearson Education 2010 [Cap 15 – Probabilistic reasoning over time]