Overview

Probabilistic Reasoning over Time

- Time and Uncertainty
- Inference in Temporal Models
- Hidden-Markov Models
- Kalman Filters
- Dynamic Bayesian Networks

Time and Uncertainty

- Counterexample-1: Car repair: Observations (e.g. car doesn‘t start, radio silent) and diagnoses (e.g. battery empty) remain constant during process of diagnosis: static world
- Example-2: Treating a diabetes patient: Observations (e.g. recent insulin doses, food intake, blood sugar measurements) not only serve for stationary diagnosis, but also for deciding actions to control future states
- Simplified standard example: The security guard in an underground installation wants to know whether it‘s raining today. The only observation available is, whether the director has an umbrella
  - State with two variables: Rain R (bool; not observable) and Umbrella U (bool; observable)
  - State sequences for different days (time intervals): R_0, R_1, R_2, … and U_1, U_2, …

Notation

- Time slices contain random variables: Partially unobservable state variables (X_t) and partially observable evidence variables (E_t)
- Simplifying assumptions: Equally long time intervals and static sets X_t and E_t over time
- Evidence variable at time t: $E_t = e_t$
- Time sequences are represented by a:b, e.g. $X_{a:b}$

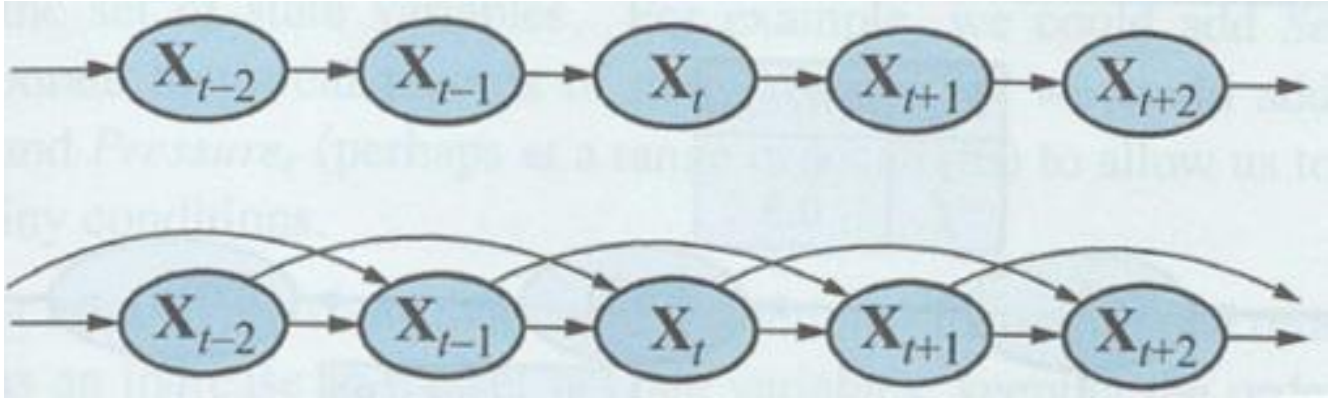
stationary

Transition and Sensor Models (1)

- Representation as **Bayesian network** with causal dependencies of variables over time
- Problems and Solutions:
  - Infinite many conditional probabilities for each variable in each time interval
    - Assumption: Stationary processes whose causality does not change over time, i.e. $P(U_t \mid \text{Parents}(U_t))$ constant for t
  - Each variable has infinite many parents
    - **Markov assumption**: Current state depends on finite subset of variables: **Markov process** or **Markov chain**

Transition and Sensor Models (2)

- **First order Markov process:** Current state depends on predecessor state only: **$P(X_t | X_{0:t-1}) = P(X_t | X_{t-1})$**
  - e.g. battery-based robot: the state of the battery, which depends on all robot actions, must be accessible either by the sum of energy consumption or by a battery-sensor.
  - Illustration: Top row first order Markov process; below second order Markov process

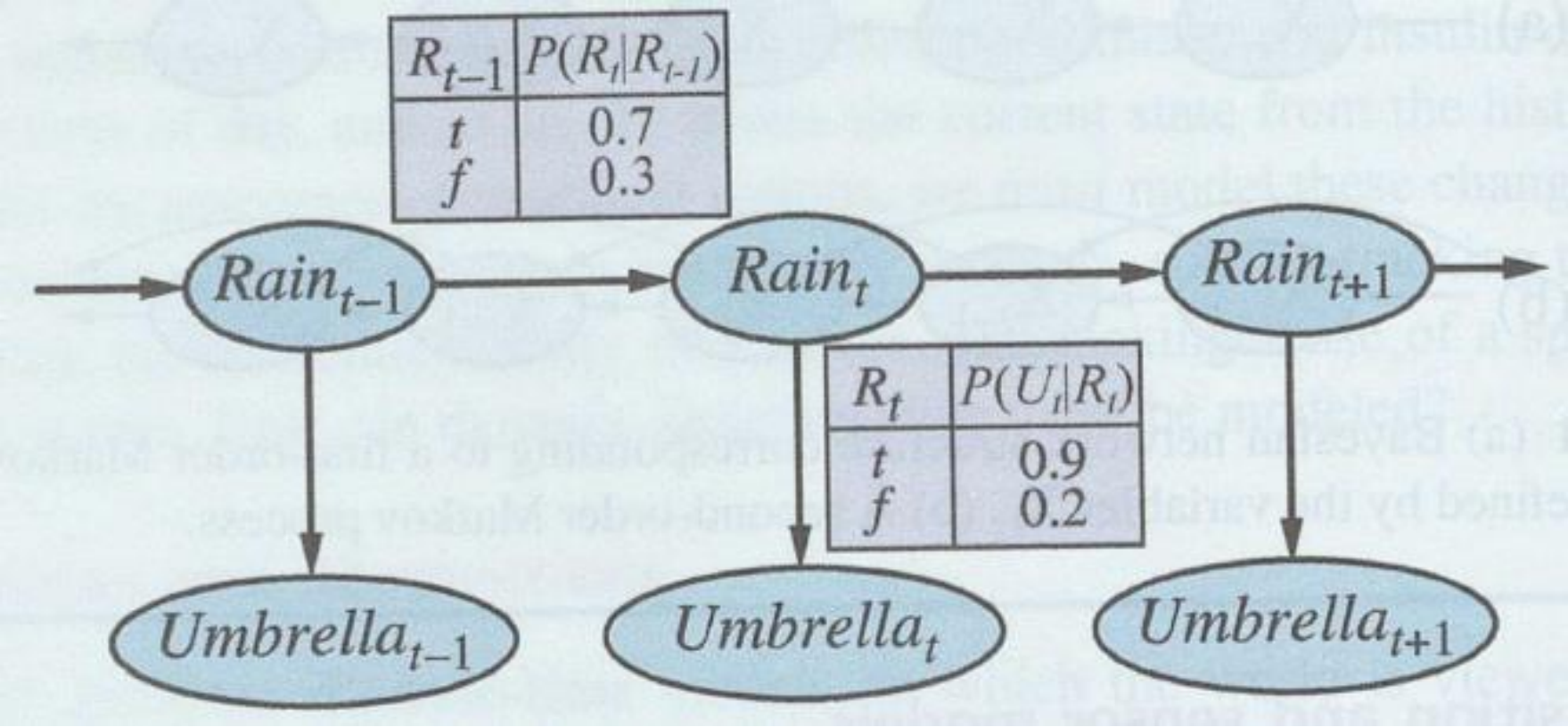


- Evidence variables depend on current state only: $P(E_t | X_{0:t}, E_{0:t-1}) = P(E_t | X_t)$ **("Sensor-model")**

Standard Example: Detecting Rain from Underground

The underground security guard infers whether its raining from observing umbrellas only

- Probability of state variable rain at day t $P(R_t)$ only depends on rain on previous day t-1 (R_{t-1})
- Probability of evidence variable umbrella $P(U_t)$ only depends on rain on same day (R_t)
- Additionally the apriori-probabilitiy $P(R_0)$ of rain from the first day is necessary



Specification of Complete Joint Distribution

… of first order Markov process:

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

with three components:

$P(X_0)$: Initial state model

$P(X_i|X_{i-1})$: Transition model

$P(E_i|X_i)$: Sensor model

Inference in Temporal Models

Basic inference tasks:

- **Filtering** or **state estimation:** Computation of the belief state $\mathbf{P(X_t|e_{1:t})}$, i.e. the probability (posterior distribution) of the current state from the evidence history
  - In underground example: Probability of rain today
- **Prediction:** Computation of the posterior distribution of future states $\mathbf{P(X_{t+k}|e_{1:t})}$, given all evidence for some k>0
  - In underground example: Probability of rain tomorrow
- **Smoothing:** Computation of the posterior distribution over a past state $\mathbf{P(X_k|e_{1:t})}$ for some k with $0 \leq k < t$ with evidence $e_{1:t}$ before and after that state
  - In underground example: Probability of rain yesterday
- **Most likely explanation**: Computation of the most probable sequence of states given all evidence, i.e. the sequence of observations: argmax $\mathbf{P(X_{1-t}|e_{1:t})}$
  - In underground example: If umbrella appears on days 1, 2, 3, and 5 what is the most probable rain sequence?

State Estimation

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

NormConst.	sensor model.	transition model.	Recursion t+1→t

Algorithm: Bottom-up computation

- Start with apriori probability of state
- Compute probability of next state (multiplication of sensor model and transition model)
- … until the current state is reached

Example for State Estimation (1)

- Compute $\mathbf{P(R_2 | u_{1:2})}$ with two observations of umbrella: U_1 = true, U_2 = true
- Apriori-probability of rain $P(R_0)$: 0.5
- Rain probability R_1 after the first observation U_1 = true:

$$P(R_1 | U_1 = \text{true}) = \alpha\, P(U_1 = \text{true} | R_1) * \sum_{r_0} P(R_1 | r_0) * P(r_0)$$
$$= \alpha\, P(U_1 = \text{true} | R_1) * (\langle 0.7\ 0.3\rangle * 0.5 + \langle 0.3\ 0.7\rangle * 0.5)$$
$$= \alpha\, P(U_1 = \text{true} | R_1) * \langle 0.5\ 0.5\rangle$$
$$= \alpha\, \langle 0.9\ 0.2\rangle * \langle 0.5\ 0.5\rangle$$
$$= \alpha\, \langle 0.45\ 0.1\rangle = \mathbf{\langle 0.818\ \ 0.182\rangle}$$

- Rain probability R_2 after the second observation u_2 (i.e. U_2 = true):

$$P(R_2 | U_2 = \text{true}) = \alpha\, P(u_2 | R_2) * \sum_{r_1} P(R_2 | r_1) * P(r_1 | u_1)$$
$$= \alpha\, P(u_2 | R_2) * (\langle 0.7\ 0.3\rangle * 0.818 + \langle 0.3\ 0.7\rangle * 0.182)$$
$$= \alpha\, P(u_2 | R_2) * \langle 0.627\ 0.373\rangle$$
$$= \alpha\, \langle 0.9\ 0.2\rangle * \langle 0.627\ 0.373\rangle$$
$$= \alpha\, \langle 0.565\ 0.075\rangle = \mathbf{\langle 0.883\ \ 0.117\rangle}$$

Example for State Estimation (2)

- Compute $\mathbf{P(R_2 | u_{1:2})}$ with two observations of umbrella: U_1 = true. U_2 = true
- Apriori-probability of rain $P(R_0)$: 0.1
- Rain probability R_1 after the first observation U_1 = true:

$P(R_1 | U_1 = \text{true}) = \alpha\, P(U_1 = \text{true} | R_1) * \sum_{r_0} P(R_1 | r_0) * P(r_0)$

$= \alpha\, P(U_1 = \text{true} | R_1) * (\langle 0.7\ 0.3\rangle * 0.1 + \langle 0.3\ 0.7\rangle * 0.9$

$= \alpha\, P(U_1 = \text{true} | R_1) * \langle 0.34\ 0.66\rangle$

$= \alpha\, \langle 0.9\ 0.2\rangle * \langle 0.34\ 0.66\rangle$

$= \alpha \cdot \langle 0.306\ 0.132\rangle = \mathbf{\langle 0.7\ 0.3\rangle}$

- Rain probability R_2 after the second observation u_2 (i.e. U_2 = true):

$P(R_2 | U_2 = \text{true}) = \alpha\, P(u_2 | R_2) * \sum_{r_1} P(R_2 | r_1) * P(r_1 | u_1)$

$= \alpha\, P(u_2 | R_2) * (\langle 0.7\ 0.3\rangle * 0.7 + \langle 0.3\ 0.7\rangle * 0.3$

$= \alpha\, P(u_2 | R_2) * \langle 0.58\ 0.42\rangle$

$= \alpha\, \langle 0.9\ 0.2\rangle * \langle 0.58\ 0.42\rangle$

$= \alpha \cdot \langle 0.522\ 0.084\rangle = \mathbf{\langle 0.862\ 0.139\rangle}$

Prediction

Predicting the future is like state estimation without new evidence

- $\mathbf{P(X_{t+k+1} \mid e_{1:t}) = \sum_{x_{t+k}} P(X_{t+k+1} \mid x_{t+k})\, P(x_{t+k} \mid e_{1:t})}$

Underground example with : $P(R_t) = 1$ (extreme case!)

- $P(R_{t+1}) = \sum_{r_t} P(R_{t+1} \mid r_t) = 1 * \langle 0.7\ 0.3 \rangle + 0 * \langle 0.3\ 0.7 \rangle = \langle 0.7\ 0.3 \rangle$
- $P(R_{t+2}) = \sum_{r_{t+1}} P(R_{t+2} \mid r_{t+1}) = 0.7 * \langle 0.7\ 0.3 \rangle + 0.3 * \langle 0.3\ 0.7 \rangle = \langle 0.58\ 0.42 \rangle$
- $P(R_{t+3}) = \sum_{r_{t+2}} P(R_{t+3} \mid r_{t+2}) = 0.58 * \langle 0.7\ 0.3 \rangle + 0.42 * \langle 0.3\ 0.7 \rangle \approx \langle 0.53\ 0.47 \rangle$
- Converges after few steps to rain probability of 50%

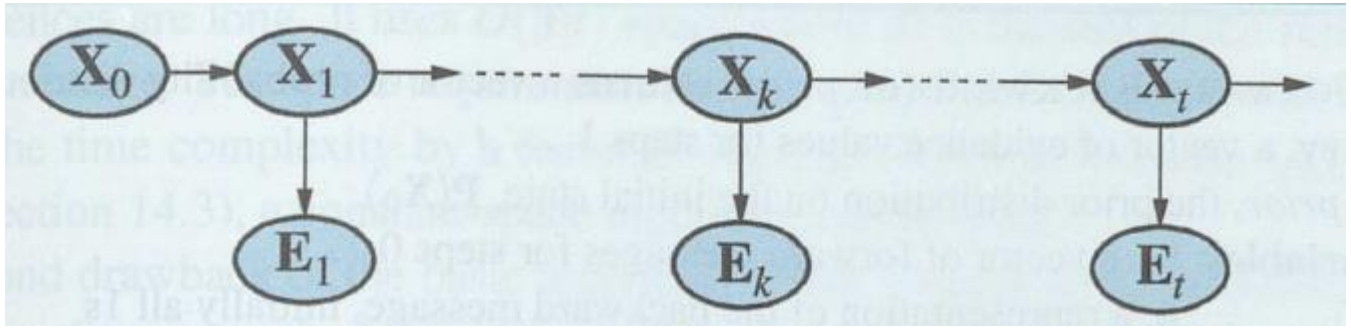
Generalization: Predicting the future depends after a few steps mainly on the transition probabilities from one state to the next and barely on the probability of the initial state

Smoothing

Computation of probability for any state, given complete sequence of observations:

$$\mathbf{P(X_k | e_{1:t})}$$

- Split in forward and backward part



$$\mathbf{P(X_k)|e_{1:t}} = P(X_k | e_{1:k}, e_{k+1:t})$$

$$= \alpha\, P(X_k | e_{1:k})\, P(e_{k+1:t} | X_k, e_{1:k}) \quad \text{(using Bayes' rule, given } e_{1:k})$$

$$= \alpha\, P(X_k | e_{1:k})\, P(e_{k+1:t} | X_k) \quad \text{(using conditional independance)}$$

$$= \alpha\, f_{1:k} \times b_{k+1:t} \quad \text{(vector multiplication of forward and backward path)}$$

Forward path = state estimation

Backward path = see next slide

Backward Path of Smoothing

$$\begin{aligned}
\mathbf{P}(\mathbf{e}_{k+1:t}|\mathbf{X}_k) &= \sum_{\mathbf{x}_{k+1}} \mathbf{P}(\mathbf{e}_{k+1:t}|\mathbf{X}_k, \mathbf{x}_{k+1})\mathbf{P}(\mathbf{x}_{k+1}|\mathbf{X}_k) \quad \text{(conditioning on } \mathbf{X}_{k+1}\text{)} \\
&= \sum_{\mathbf{x}_{k+1}} P(\mathbf{e}_{k+1:t}|\mathbf{x}_{k+1})\mathbf{P}(\mathbf{x}_{k+1}|\mathbf{X}_k) \quad \text{(by conditional independence)} \\
&= \sum_{\mathbf{x}_{k+1}} P(\mathbf{e}_{k+1}, \mathbf{e}_{k+2:t}|\mathbf{x}_{k+1})\mathbf{P}(\mathbf{x}_{k+1}|\mathbf{X}_k) \\
&= \sum_{\mathbf{x}_{k+1}} \underbrace{P(\mathbf{e}_{k+1}|\mathbf{x}_{k+1})}_{\text{sensor model}} \underbrace{P(\mathbf{e}_{k+2:t}|\mathbf{x}_{k+1})}_{\text{recursion}} \underbrace{\mathbf{P}(\mathbf{x}_{k+1}|\mathbf{X}_k)}_{\text{transition model}}, \qquad (14.9)
\end{aligned}$$

- Initialization of backward path: $b_{t+1:t} = P(e_{t+1:t})|X_t) = P(\quad|X_t) = 1$
  - since $e_{t+1:t}$ is the empty sequence which has the probability of 1.

Example for Smoothing

Underground model with goal:

$P(R_1 | u_{1:2}) =$ $\alpha\, P(R_1 | u_1)\, P(u_2 | R_1)$ with U_1 = true and U_2 = true

- $P(R_1 | u_1) = \langle 0.818, 0.182 \rangle$ (see state estimation example)
- $P(u_2 | R_1) = \sum_{r2} P(u_2 | r_2) \quad P(\ \ | r_2) \quad P(r_2 | R_1)$

sensor model rekursion transition model

$= \quad 0.9 \times 1 \times \langle 0.7, 0.3 \rangle +$

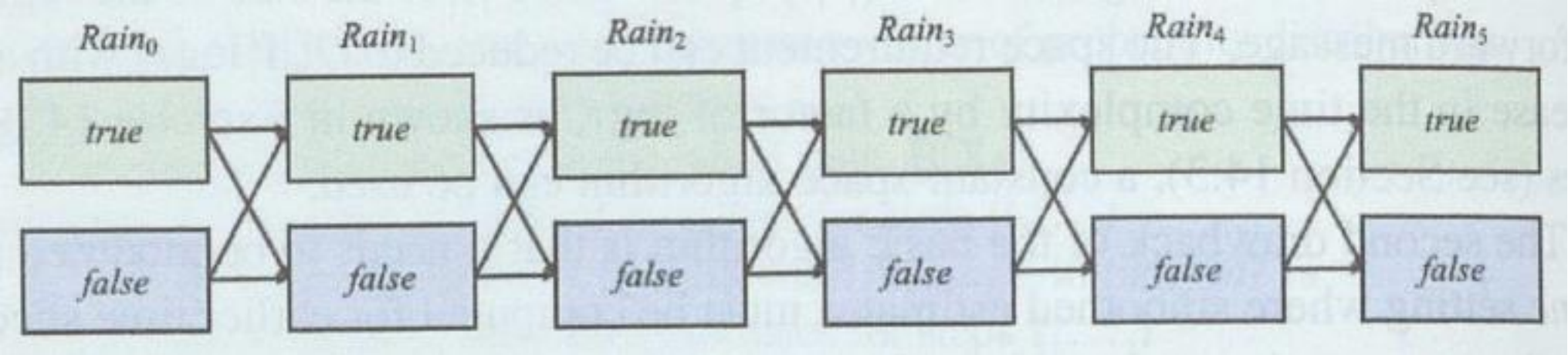
$\quad\quad 0.2 \times 1 \times \langle 0.3, 0.7 \rangle$

$= \quad \langle 0.69, 0.41 \rangle$

$P(R_1 | u_{1:2}) =$ $\alpha \langle 0.818, 0.182 \rangle \times \langle 0.69, 0.41 \rangle \approx \langle 0.883, 0.117 \rangle$

Finding the Most Likely Sequence

- Assume we have the following observations: 5 days with umbrella except day 3
  - What is the most likely sequence for rain for the 5 days?



- Many applications, e.g.
  - Speech recognition: Evidence comes from the record sounds; states are the words

Viterbi Algorithm for Finding the Most Likely Sequence

- **Simple solution:** Compute all paths and choose the most probable (exponential costs)
- **Wrong solution:** Compute with smoothing the probability of each state and choose the best
  - Problem: Approach ignores the transition between states (could be even impossible)
- **Recursive solution (Viterbi algorithm):** Compute most probable path to a state for a time step and expand it to the next time step (linear costs; similar to state estimation):

$$m_{1:t+1} = \max_{x_{1:t}} P(x_{1:t}, X_{t+1}, e_{1:t+1}) =$$

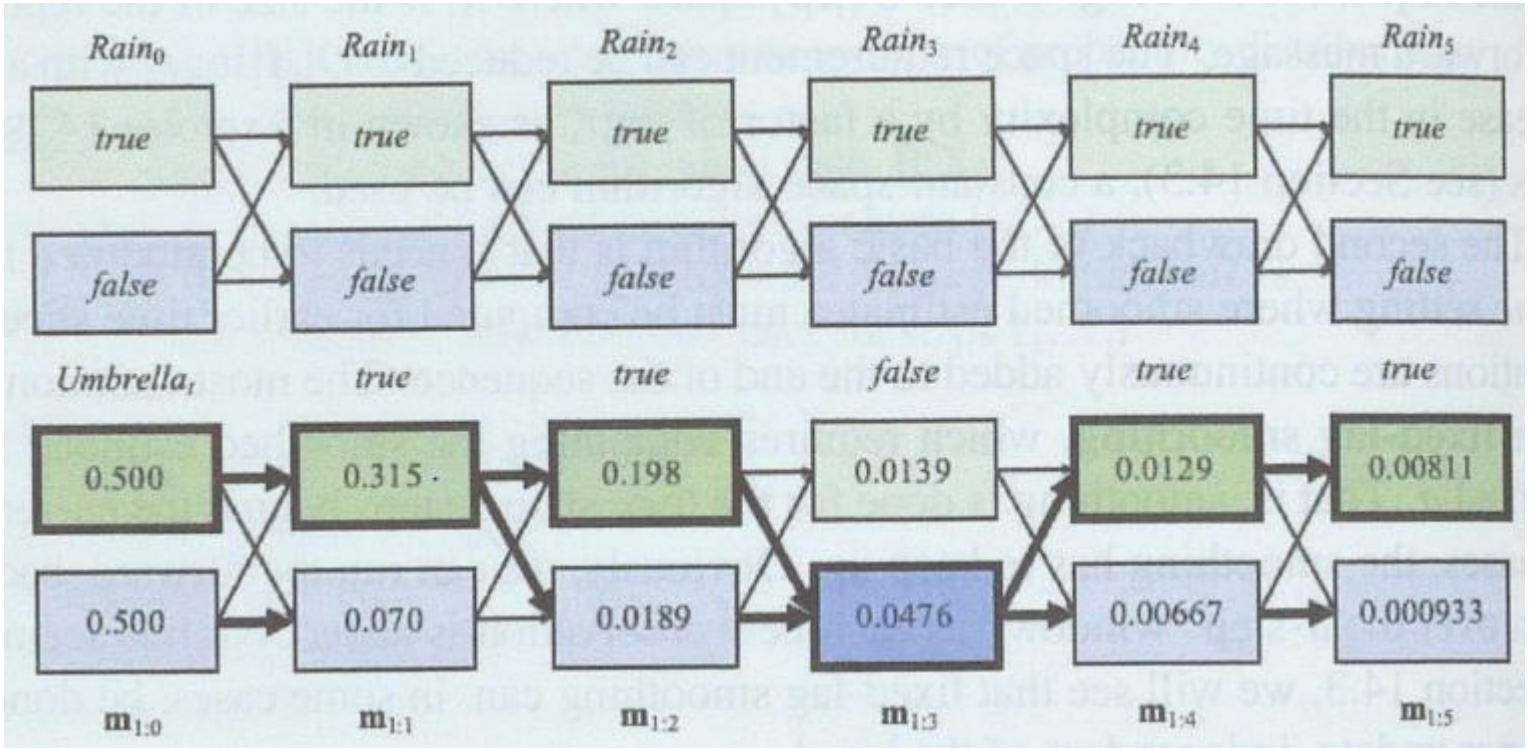
$$\mathbf{P(e_{t+1} | X_{t+1})} \quad \mathbf{\max_{x_t} P(X_{t+1} | x_t)} \quad \mathbf{\max_{x_{1:t-1}} P(x_{1:t-1}, x_t, e_{1:t})}$$

sensor model | transition model | Recursion t+1→t

Example for Viterbi Algorithm

Example continued: 5 days with umbrella except day 3

- What is the most likely sequence for rain for the 5 days?



Transition $m_{1:0}$ to $m_{1:1}$

case distinction (R=Rain; N=NoRain):

$P(R_{t-1})$ * transition * sensor

R→R: 0.5 * 0.7 * 0.9 = **0.315**

R→N: 0.5 * 0.3 * 0.2 = 0.03

N→R: 0.5 * 0.3 * 0.9 = 0.135

N→N: 0.5 * 0.7 * 0.2 = **0.07**

Transition $m_{1:1}$ to $m_{1:2}$

R→R: 0.315 * 0.7 * 0.9 = **0.1985**

R→N: 0.315 * 0.3 * 0.2 = **0.0189**

N→R: 0.07 * 0.3 * 0.9 = 0.0189

N→N: 0.07 * 0.7 * 0.2 = 0.0098

Transition $m_{1:2}$ to $m_{1:3}$

R→R: 0.1985 * 0.7 * 0.1 = **0.0139**

R→N: 0.1985 * 0.3 * 0.8 = **0.0476**

N→R: 0.0189 * 0.3 * 0.1 = 0.0006

N→N: 0.0189 * 0.7 * 0.8 = 0.0106

Hidden Markov Models (HMM)

- HMMs contain only one state variable at one time-interval
  - If there are several variables, they can be aggregated to one „mega variable“.
- HMMs may contain several evidence variables, since they are always observed (if not, they can be dropped for that time step).
- HMMs allow to use matrix operations for the inference tasks:
  - Transition model $P(X_t | X_{t-1})$ becomes an S x S matrix T with
    - $T_{ij} = P(X_t = j \mid X_{t-1} = i)$
    - Example for underground world: $T = P(X_t | X_{t-1}) = \begin{pmatrix} 0.7 & 0.3 \\ 0.3 & 0.7 \end{pmatrix}$

HMM Sensor Model

- Sensor model is also put in matrix form
  - Because the value of the evidence variable E_t at time t (e_t) is known in each time step
    - We need only $P(e_t | X_t=i)$ for each state i
    - For mathematical convenience we place these values into an S x S observation matrix O_t, one for each time step
    - The i-th diagnosis entry of O_t is $P(e_t | X_t=i)$, the other entries are 0
    - Example: Day 1 in underground world (U_1 = true) and day 3 (U_3 = false):

$$O_1 = \begin{pmatrix} 0.9 & 0 \\ 0 & 0.2 \end{pmatrix} \qquad O_3 = \begin{pmatrix} 0.1 & 0 \\ 0 & 0.8 \end{pmatrix}$$

HMM formulas

Computations like state estimation or smoothing become simple matrix vector multiplications:

- **State estimation (forward path):**

$f_{1:t+1} = \mathbf{P}(X_{t+1} | e_{1:t+1}) = \quad \alpha \quad \mathbf{P}(e_{t+1} | X_{t+1}) \quad \sum_{x_t} \mathbf{P}(X_{t+1} | x_t) \quad \mathbf{P}(x_t | e_{1:t})$

NormConst sensor model transition model Recursion t+1→t

$f_{1:t+1} = \quad \alpha \quad \mathbf{O}_{t+1} \quad \mathbf{T}^T \quad \mathbf{f}_{1:t}$

- **Backward Path:**

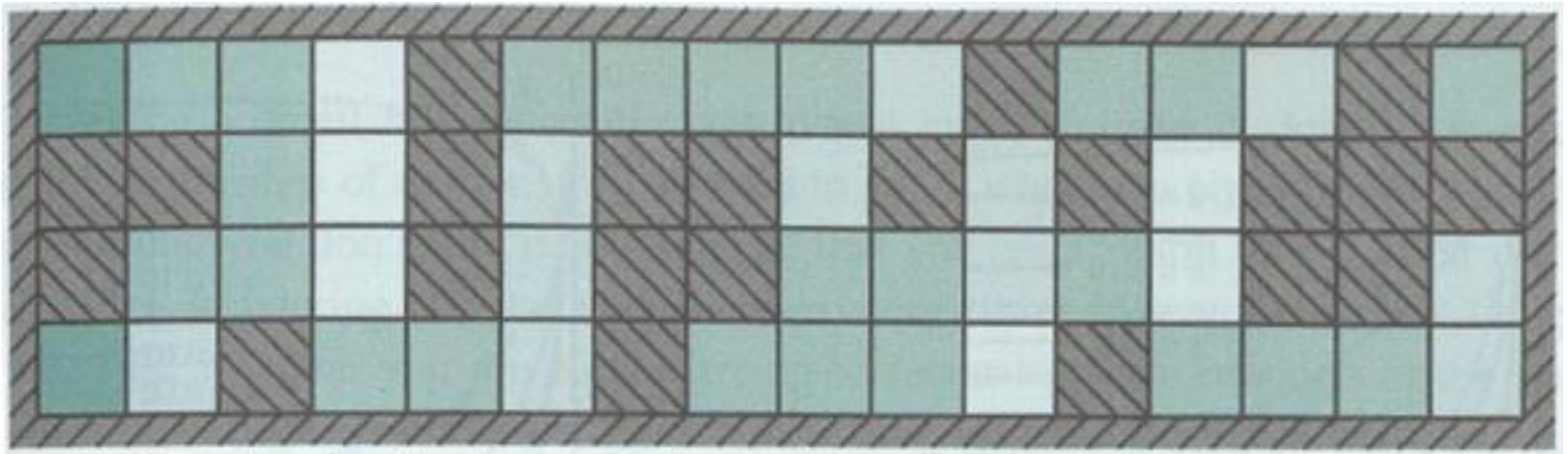
$b_{k+1:t} =$

$$\sum_{x_{k+1}} \underbrace{P(e_{k+1} | x_{k+1})}_{\text{sensor model}} \underbrace{P(e_{k+2:t} | x_{k+1})}_{\text{recursion}} \underbrace{\mathbf{P}(x_{k+1} | X_k)}_{\text{transition model}}$$

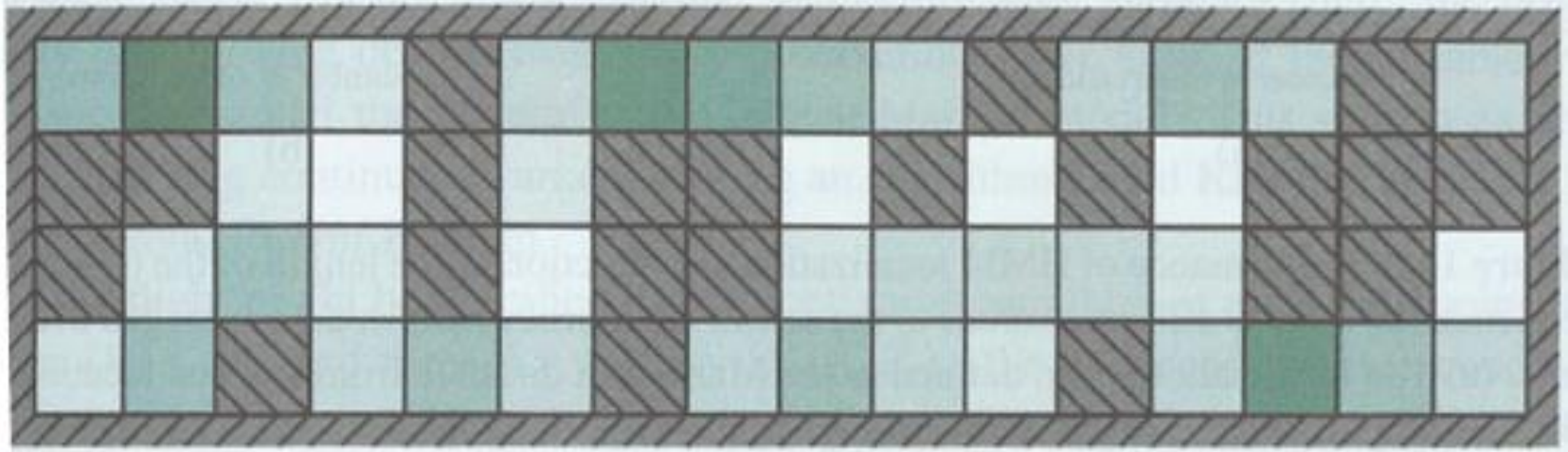
$b_{k+1:t} = \quad \mathbf{O}_{k+1} \quad \mathbf{b}_{k+2:t} \quad \mathbf{T}$

HMM Example: Localization

- A robot has 4 sonar sensors telling him, whether there is a wall, e.g. 1011 means North: Wall; East: No wall; South and West: Wall.
  - Sensors are unreliable with probability 0.2
- Robot has 4 actions to move in each direction
  - However, move is non-deterministically, so the robot may end on any adjacent field
- 42 states (empty squares)



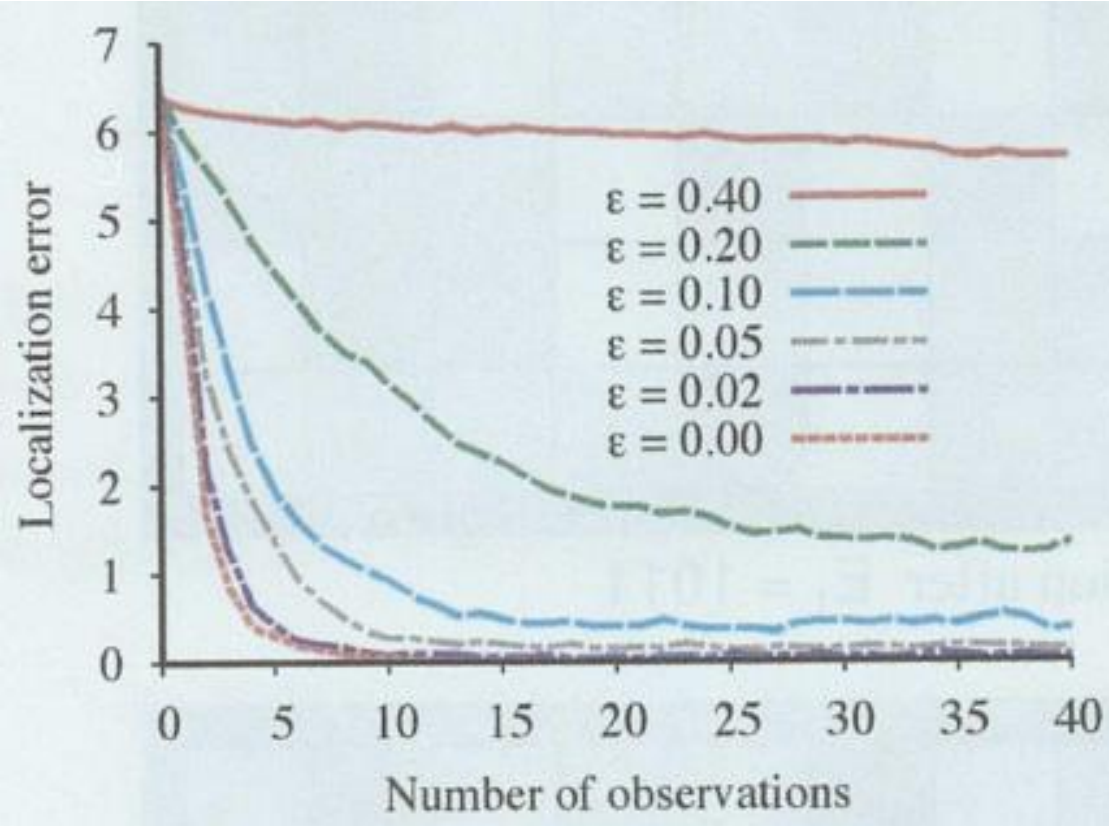
(a) Posterior distribution over robot location after $E_1 = 1011$

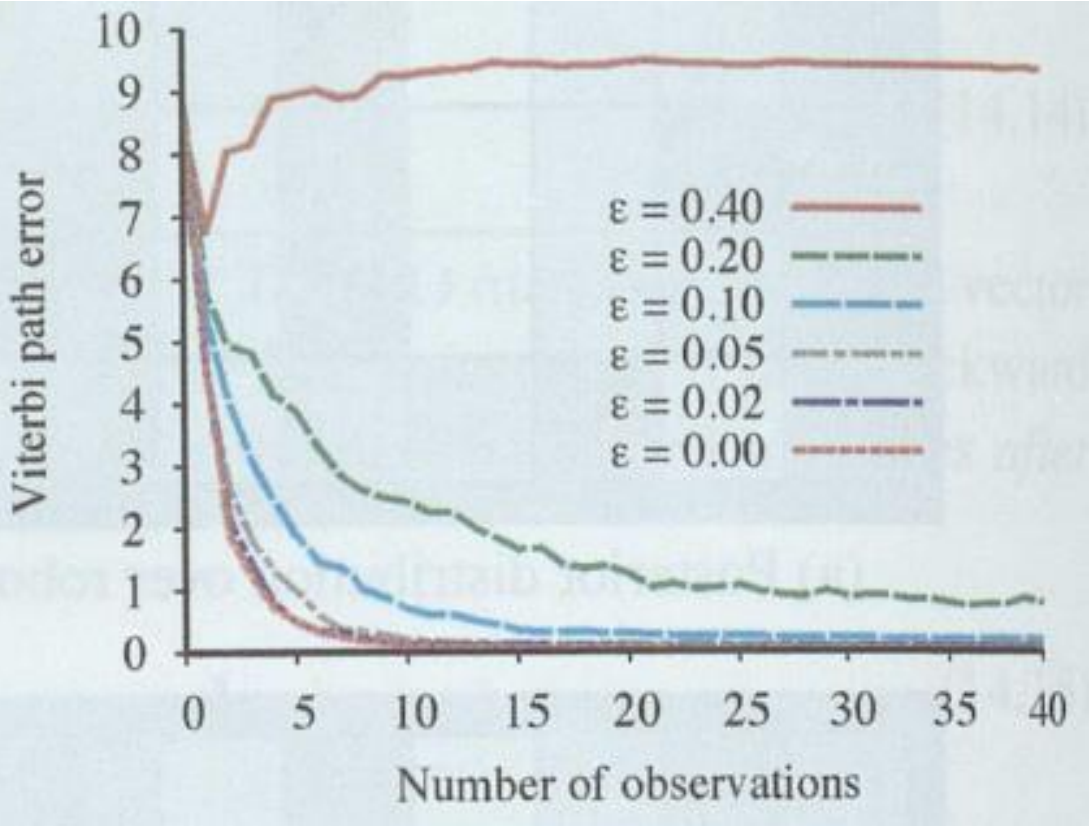


(b) Posterior distribution over robot location after $E_1 = 1011, E_2 = 1010$

Evaluation Results

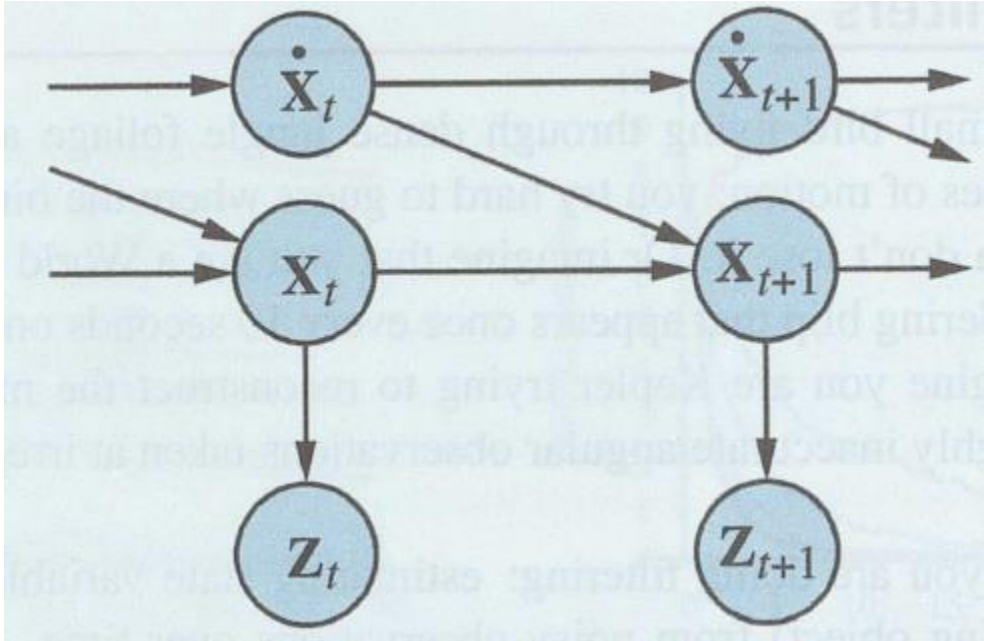
- Robot localization errors depending on length of observation and sensor error rate
  - Left: Localization error defined as Manhattan distance from true localization
  - Right: Path error defined as average Manhattan distance from true path





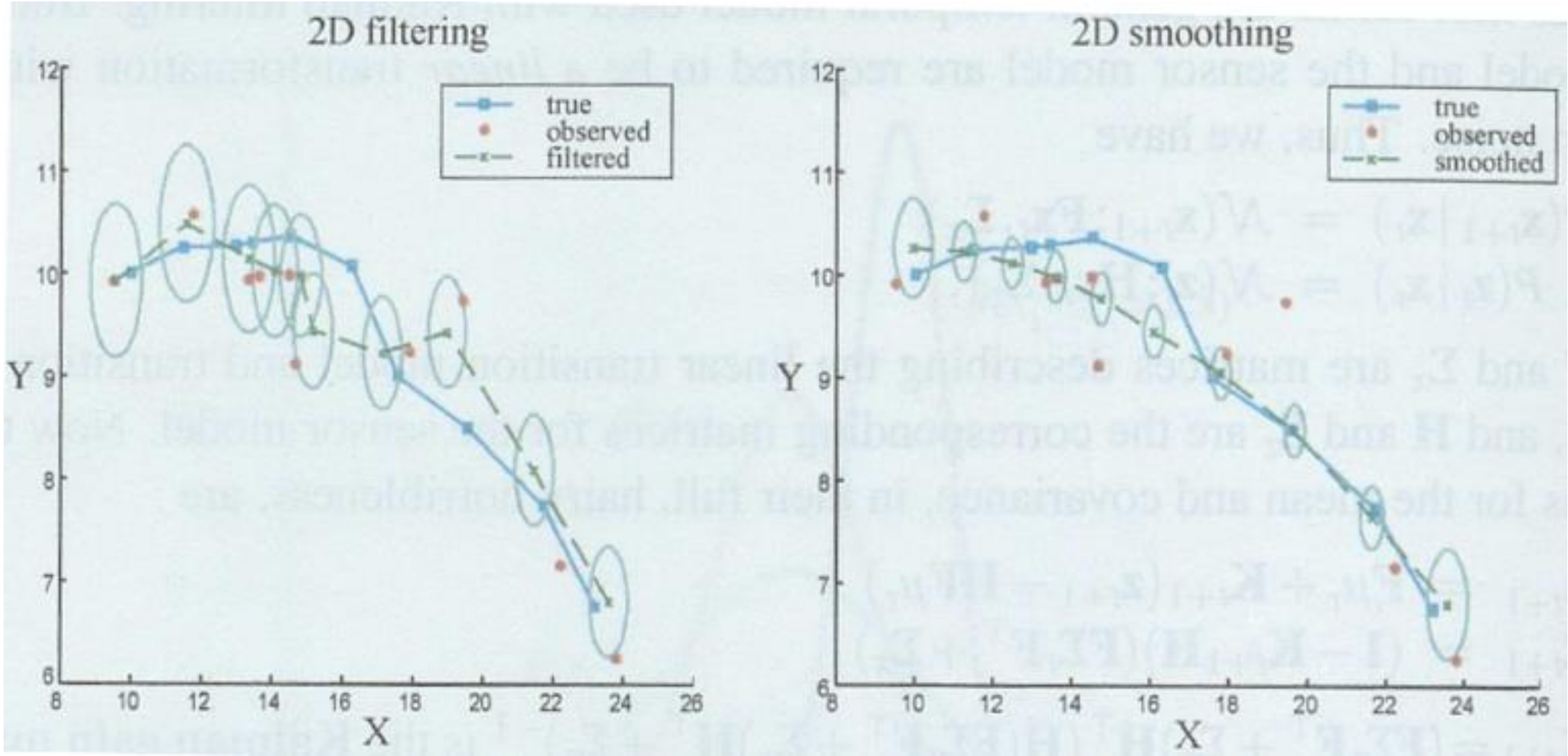
Kalman Filter

- Example application: Tracing objects based on noisy observations, e.g. a bird or an object on a radar screen
- Problem: Dealing with **continuous variables** (observatations and states) like the position and velocity of a moving object in three dimensions
  - If variables would be discrete, we could use HMMs
- Solution: **Kalman filtering algorithm**
  - Requirement: Suitable conditional densities to represent transition and sensor models
    - Use linear-Gaussian distributions (i.e. the next state X_{t+1} must be a linear function of the current state X_t plus some Gaussian noise)
      - Example (right): Next X-position of a flying object depends on current X-position and X-velocity and on observations Z



Example for Kalman Filter

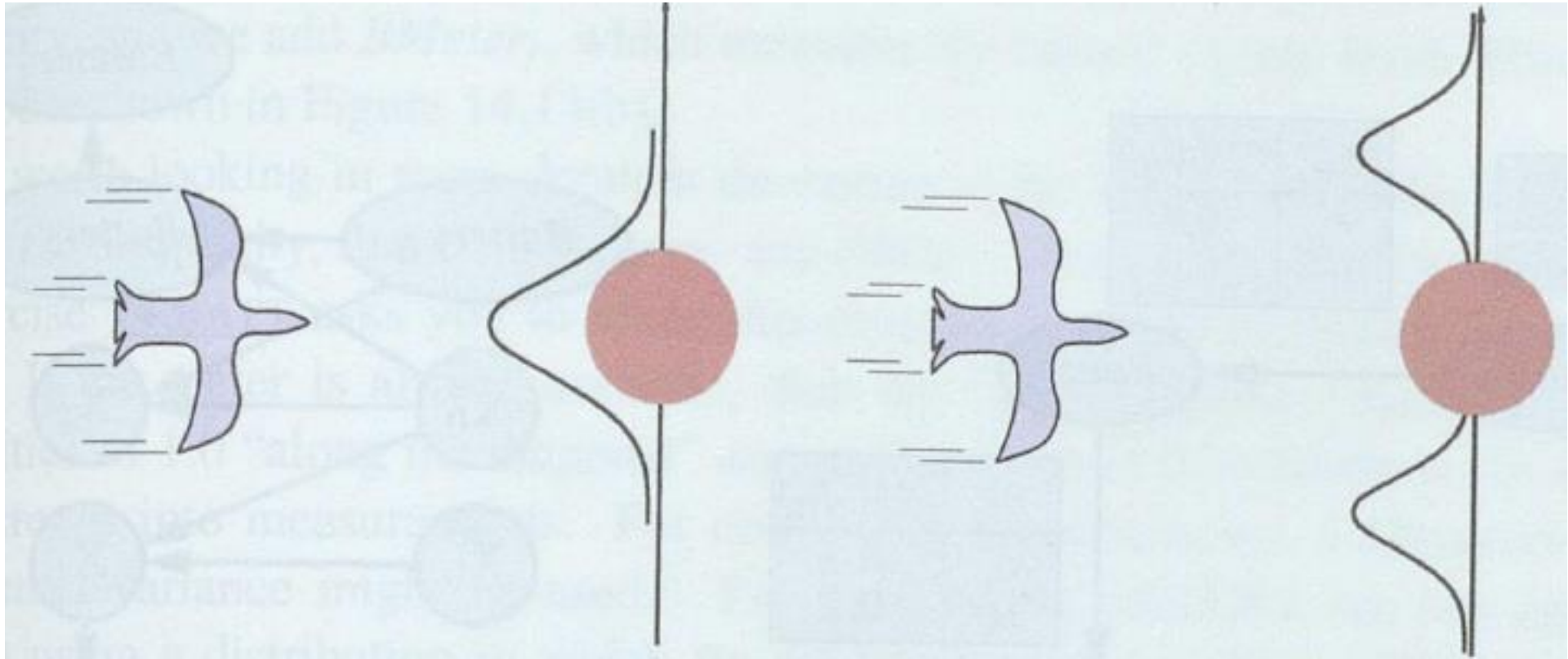
- Estimation of tracectory of a moving object in X-Y plane with noisy observations (red dots)
- Left: Results of Kalman filtering; Right: Results of Kalman smoothing (green line)
  - True movement: blue line



Frank Puppe

Problem with Kalman Filter

- Kalman filter will predict the location of the bird fyling toward the tree using a single Gaussian distribution centered on the obstacle (left).
- To achieve a more realistic prediction, a highly nonlinear model is necessary, e.g. achieved by switching Kalman filter, using and combining different models of the system (straight flight, left turns, right turns)
- Alternative: Dynamic Bayesian Networks

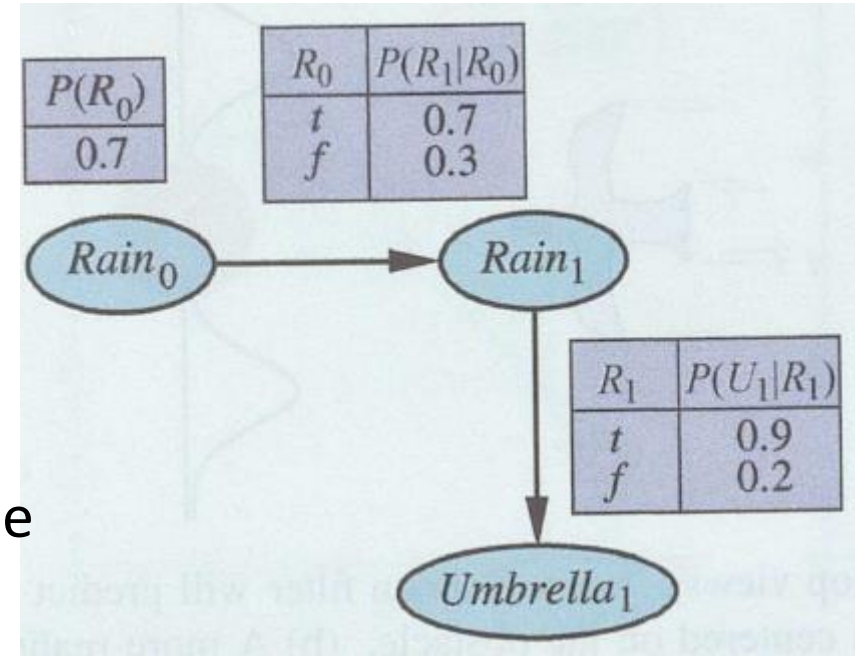


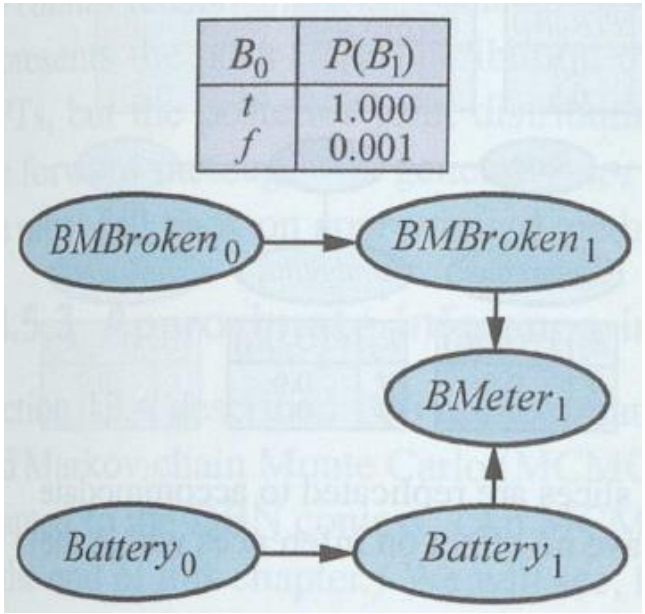
Dynamic Bayesian Networks (DBN)

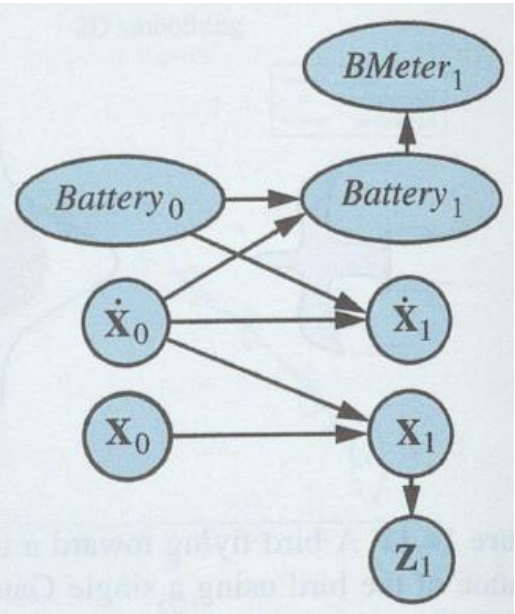
- Every HMM can be represented as discrete DBN and vice versa
  - However, the state represetation in a HMM might be exponential in the number of variables
- Every Kalman Filter can be represented as continuous DBN but not vice versa
  - Density distribution in Kalman filter is a Gaussian distribution, which cannot deal with non-linearities (e.g. bird example or probabilty of location of a lost object: it might be at one of the last visited places, but probably not in between)
  - Aspects of the real world like purposive agents, obstacles and preferences introduce „non-linearities“ that require combinations of discrete and continuous variables

Construction DBNs

- To construct a DBN, 3 kinds of information are necessary:
  - Prior distribution over state variables: $\mathbf{P(X_0)}$
  - Transition model $\mathbf{P(X_{t+1} | X_t)}$
  - Sensor model $\mathbf{P(E_t | X_t)}$
- Examples:
  - Known underground world: 1 state and 1 evidence variable
  - Simple robot motion (3 state variables for position, velocity, battery charge and 2 evidence variables for battery charge level and position; without probability tables)
  - Extended robot motion with an additional sensor model „BMBroken“ stating that a broken sensor remains broken forever

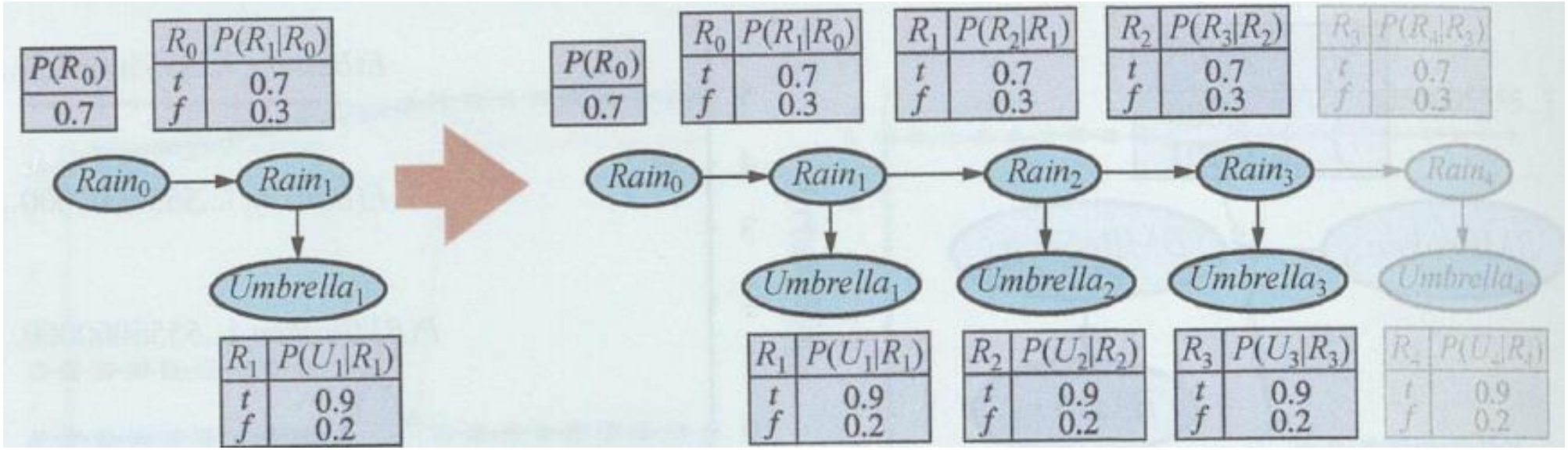






Exact Inference in DBNs

- General technique to deal with many observations: **Unrolling** the network
- Any exact inference algorithm for Bayesian network usable



- Naive application of unrolling inefficent, since space and inference time increase linearly with number of observations
- Recursive algorithm (like state estimation algorithm) possible with constant space (two slices) and constant time per update.
- However, the constant is usually exponential in number of state variables (similar to HMM)

Approximate Inference in DBNs

- Two candidates: Likelihood weighting and Markow chain Monte Carlo (MCMC)
  - Likelihood weighting with two modifications:
    - Avoid unrolling by recursive computation (with only two states in memory) by sampling N examples together: „**Sequential importance sampling**“
    - Remaining problem: Likelihood weighting sets evidence variables and weights each run with the probability of setting the evidence variables given its parents. If evidence variable appear late (as in DBNs), this does not help very much
      - Necessary improvement: set samples in high-probability regions of the state space
      - Realization by throwing away samples with very low weights: The population of samples is used to generate a new population of samples, each new sample is selected based on a sample of the current population with probability proportional to its weight: „**Particle filtering**“ (**„sequential importance sampling with resampling“)**
  - Particle filtering is efficient in many cases, but with exceptions. For some special cases, better specialized algorithm exist

Comparison

Comparison of particle filtering (PF) and sequential importance sampling (SIS) for the grid-world location problem with the gold standard computed by an exact inference algorithm