인공지능

22년 삼성 AI 전문가과정
6월 8일 수요일 3교시
장병탁

12차시 : Temporal Reasoning

서울대학교 컴퓨터공학부
담당 교수: 장병탁

Seoul National University
Byoung-Tak Zhang

Lecture Overview

인공지능

12차시 : Temporal Reasoning

서울대학교 컴퓨터공학부
담당 교수: 장병탁

Seoul National University
Byoung-Tak Zhang




Introduction: Temporal Reasoning

- **Probabilistic Reasoning** (Previous lecture)
  - A systematic way to represent dependency relationships explicitly in Bayesian networks.
  - Probabilistic inference & approximate inference algorithms.
- **Temporal Reasoning** (This lecture)
  - We try to interpret the present, understand the past, and perhaps **predict the future**, even when very little is crystal clear.
    - The methods (search, logical agents, planning etc.) defined belief states in terms of which world states were possible, but could say nothing about which states were ***likely*** or ***unlikely.***
  - In this lecture, we use **probability theory** to quantify the degree of belief in elements of the belief state.
  - We define **basic inference tasks** and describes the general structure of inference algorithms for temporal models.
    - Hidden Markov models, Kalman filters, dynamic Bayesian networks

Motivating Examples: Visual Tracking and Navigation

Videos

- Kalman filter result on real aircraft:
  http://www.youtube.com/watch?v=0GSIKwfkFCA&feature=related
- Robotics - SLAM and localization with a stereo camera:
  http://www.youtube.com/watch?v=m3L8OfbTXH0&feature=related
- Hand tracking using particle filters:
  http://www.youtube.com/watch?v=J3ioMxRI174

Temporal Models

Transition and sensor models

- Markov processes (a.k.a. Markov chains)

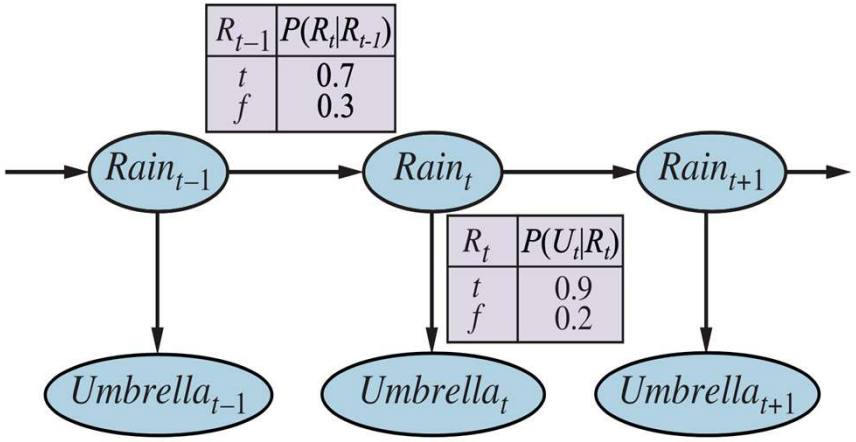
  $\mathbf{P}(\mathbf{X}_t|\mathbf{X}_{0:t-1}) = \mathbf{P}(\mathbf{X}_t|\mathbf{X}_{t-1})$

  ➔ **Transition model**

- Sensor Markov assumption:

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

  ➔ **Sensor model** (or observation model)



- Complete joint distribution:

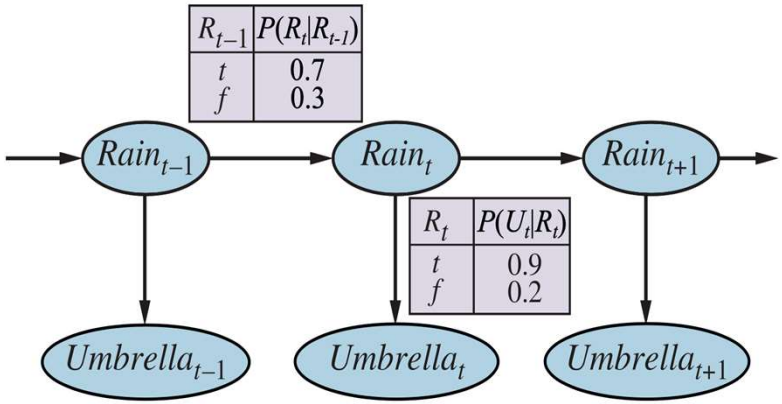
  $\mathbf{P}(\mathbf{X}_{0:t}, \mathbf{E}_{1:t}) = \mathbf{P}(\mathbf{X}_0) \prod_{i=1}^{t} \mathbf{P}(\mathbf{X}_i|\mathbf{X}_{i-1})\mathbf{P}(\mathbf{E}_i|\mathbf{X}_i)$

<출처> Stuart J. Russell and Peter Norvig (2021). Artificial Intelligence: A Modern Approach (4th Edition). Pearson

Inference in Temporal Models

Inference in Temporal Models

- **Filtering** (state estimation): $\mathbf{P}(\mathbf{X}_t|\mathbf{e}_{1:t})$
  - Belief state
- **Prediction**:$\mathbf{P}(\mathbf{X}_{t+k}|\mathbf{e}_{1:t})$ for $k > 0$
  - Evaluation of possible action sequences.
  - Like filtering but without the evidence.



- **Smoothing**:$\mathbf{P}(\mathbf{X}_k|\mathbf{e}_{1:t})$ for $0 \leq k < t$
  - Better estimate of past states, essential for learning.
- **Most likely explanation**: $\text{argmax}_{\mathbf{X}_{1:t}}\mathbf{P}(\mathbf{X}_{1:t}|\mathbf{e}_{1:t})$
  - Speech recognition, decoding with a noisy channel.

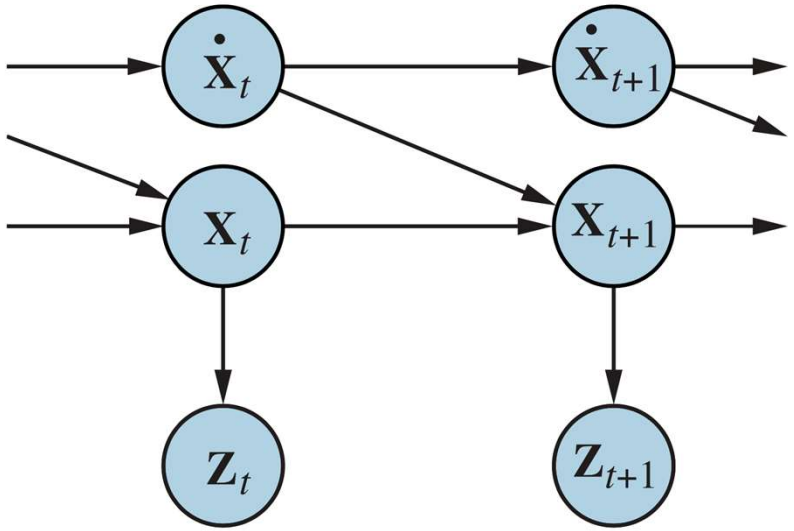
Kalman Filters

Updating Gaussian Distribution

- **Prediction**

$$\mathbf{P}(\mathbf{X}_{t+1}|\mathbf{e}_{1:t}) = \int_{\mathbf{x}_t} \mathbf{P}(\mathbf{X}_{t+1}|\mathbf{x}_t) P(\mathbf{x}_t|\mathbf{e}_{1:t}) d\mathbf{x}_t$$

predicted distribution *transition* *current distribution*



- **Update**

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

updated distribution *sensor model* *prediction*

<출처> Stuart J. Russell and Peter Norvig (2021). Artificial Intelligence: A Modern Approach (4th Edition). Pearson

Lecture 12. Temporal Reasoning

- How the agents interpret the *present*, understand the *past*, and predict the *future*, even when very little is clear. A changing world is modeled using a time variable. The transition and sensor models may be uncertain.
- **Inference tasks and inference algorithms** for models with transition and sensor models:
  - Filtering
  - Prediction
  - Smoothing
  - Most likely sequence
- **Specific temporal probability models:**
  - Hidden Markov Models
  - Kalman Filters
  - Dynamic Bayesian Networks
  - Particle Filters

Outline (Lecture 12)

12.1 Time and Uncertainty




12.1 Time and Uncertainty (1/4)

Time and Uncertainty: The world changes; we need to track and predict it.

States and observations

- Assume discrete-time models → world viewed as a series of time slices.
- Time interval between slices is assumed to be same for every interval.
- For simplicity, this chapter assume that the same subset of variables is observable in each time slice.
  - $\mathbf{X}_t$: Set of state variables at time t.
  - $\mathbf{E}_t$: Set of observable evidence variables.
  - $\mathbf{X}_{a:b}$: set of variables from $\mathbf{X}_a$ to $\mathbf{X}_b$ inclusive.
- The observation at time t is $\mathbf{E}_t = \mathbf{e}_t$ for some set of values $\mathbf{e}_t$.

12.1 Time and Uncertainty (2/4)

Transition and sensor models

- Markov assumption: $\mathbf{X}_t$ depends on bounded subset of $\mathbf{X}_{0:t-1}$
- Markov processes
  - First-order Markov process: $\mathbf{P}(\mathbf{X}_t|\mathbf{X}_{0:t-1}) = \mathbf{P}(\mathbf{X}_t|\mathbf{X}_{t-1})$ → Transition model
  - Second-order Markov process: $\mathbf{P}(\mathbf{X}_t|\mathbf{X}_{0:t-1}) = \mathbf{P}(\mathbf{X}_t|\mathbf{X}_{t-2}, \mathbf{X}_{t-1})$

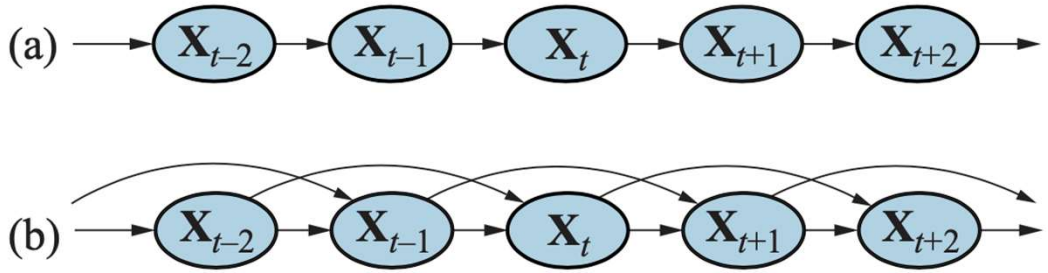


Figure 14.1 (a) Bayesian network structure corresponding to a first-order Markov process with state defined by the variables $\mathbf{X}_t$. (b) A second-order Markov process.

<출처> Stuart J. Russell and Peter Norvig (2021). Artificial Intelligence: A Modern Approach (4th Edition). Pearson

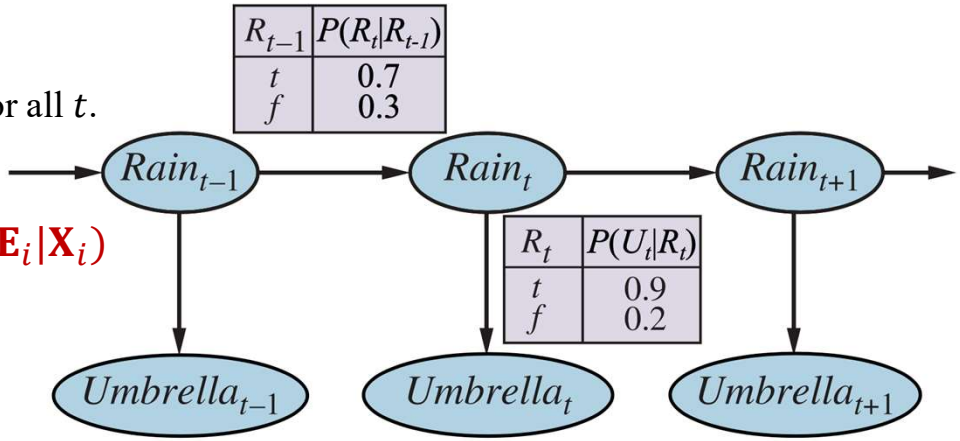
12.1 Time and Uncertainty (3/4)

Transition and sensor models

- Assume that changes in the world state are caused by a time-homogenous process.
  - A process of change that is governed by laws that do not themselves change over time.
- Sensor Markov assumption:

  $\mathbf{P}(\mathbf{E}_t|\mathbf{X}_{0:t}, \mathbf{E}_{1:t-1}) = \mathbf{P}(\mathbf{E}_t|\mathbf{X}_t)$→ Sensor model or observation model
- Stationary process:
  - Transition model and sensor model fixed for all t.
- Complete joint distribution:

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



<출처> Stuart J. Russell and Peter Norvig (2021). Artificial Intelligence: A Modern Approach (4th Edition). Pearson

12.1 Time and Uncertainty (4/4)

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

Possible fixes

- Increase order of Markov process
- Augment state (e.g. Temperature, Pressure, etc.)
- Restore the Markov property by including charge level Battery.

Example: Robot motion

- Augment position and velocity with Battery.

12.2 Inference in Temporal Models

12.2 Inference in Temporal Models (1/7)

Inference in Temporal Models

- **Filtering (state estimation)**: $\mathbf{P}(\mathbf{X}_t|\mathbf{e}_{1:t})$
  - Belief state
- **Prediction**: $\mathbf{P}(\mathbf{X}_{t+k}|\mathbf{e}_{1:t})$ for $k > 0$
  - Evaluation of possible action sequences.
  - Like filtering but without the evidence.
- **Smoothing**: $\mathbf{P}(\mathbf{X}_k|\mathbf{e}_{1:t})$ for $0 \leq k < t$
  - Better estimate of past states, essential for learning.
- **Most likely explanation**: $\text{argmax}_{\mathbf{X}_{1:t}}\mathbf{P}(\mathbf{X}_{1:t}|\mathbf{e}_{1:t})$
  - Speech recognition, decoding with a noisy channel.

12.2 Inference in Temporal Models (2/7)

Filtering

- Aim: Devise a recursive state estimation algorithm, $\mathbf{P}(\mathbf{X}_{t+1}|\mathbf{e}_{1:t+1}) = f(\mathbf{e}_{t+1}, \mathbf{P}(\mathbf{X}_t|\mathbf{e}_{1:t}))$
- $\mathbf{P}(\mathbf{X}_{t+1}|\mathbf{e}_{1:t+1}) = \mathbf{P}(\mathbf{X}_{t+1}|\mathbf{e}_{1:t}, \mathbf{e}_{t+1})$ (dividing up the evidence)

  $= \alpha\mathbf{P}(\mathbf{e}_{t+1}|\mathbf{X}_{t+1}, \mathbf{e}_{1:t})\mathbf{P}(\mathbf{X}_{t+1}|\mathbf{e}_{1:t})$ (using Bayes' rule, given $\mathbf{e}_{1:t}$)

  $= \alpha\mathbf{P}(\mathbf{e}_{t+1}|\mathbf{X}_{t+1})\mathbf{P}(\mathbf{X}_{t+1}|\mathbf{e}_{1:t})$ (by the sensor Markov assumption)

  update *prediction*
- Message is propagated **forward** along the sequence modified by each transition and updated by each new observation.
  - $\boldsymbol{f}_{1:t+1} = \text{FORWARD}(\boldsymbol{f}_{1:t}, \mathbf{e}_{t+1})$ where $\boldsymbol{f}_{1:t} = \mathbf{P}(\mathbf{X}_t|\mathbf{e}_{1:t})$
  - Time and space constant (independent of t)
  - $\mathbf{P}(\mathbf{X}_{t+1}|\mathbf{e}_{1:t+1}) = \alpha\mathbf{P}(\mathbf{e}_{t+1}|\mathbf{X}_{t+1}) \sum_{\mathbf{x}_t} \mathbf{P}(\mathbf{X}_{t+1}|\mathbf{x}_t, \mathbf{e}_{1:t})\mathbf{P}(\mathbf{x}_t|\mathbf{e}_{1:t})$

    $= \alpha\mathbf{P}(\mathbf{e}_{t+1}|\mathbf{X}_{t+1}) \sum_{\mathbf{x}_t} \mathbf{P}(\mathbf{X}_{t+1}|\mathbf{x}_t)\mathbf{P}(\mathbf{x}_t|\mathbf{e}_{1:t})$

    sensor model *transition model* *recursion*

12.2 Inference in Temporal Models (3/7)

Prediction

- Aim: Filtering without the addition of new evidence.
- Filtering process already incorporates a one-step prediction.
- $\mathbf{P}(\mathbf{X}_{t+k+1}|\mathbf{e}_{1:t}) = \sum_{\mathbf{x}_{t+k}} \mathbf{P}(\mathbf{X}_{t+k+1}|\mathbf{x}_{t+k})\mathbf{P}(\mathbf{x}_{t+k}|\mathbf{e}_{1:t})$

transition model *recursion*

Likelihood

- Likelihood message: $\ell_{1:t}(\mathbf{X}_t) = \mathbf{P}(\mathbf{X}_t, \mathbf{e}_{1:t})$
- Useful quantity if we want to compare different temporal models that might have produced the same evidence sequence.
- Having computed $\ell_{1:t}$, we obtain the actual likelihood by summing out $\mathbf{X}_t$
- $L_{1:t} = \mathbf{P}(\mathbf{e}_{1:t}) = \sum_{\mathbf{x}_t} \ell_{1:t}(\mathbf{x}_t)$

12.2 Inference in Temporal Models (4/7)

Smoothing

- Split the computation into two parts – evidence up to k and evidence from $k+1$ to t.
- $\mathbf{P}(\mathbf{X}_k|\mathbf{e}_{1:t}) = \mathbf{P}(\mathbf{X}_k|\mathbf{e}_{1:k}, \mathbf{e}_{k+1:t})$
  $= \alpha\mathbf{P}(\mathbf{X}_k|\mathbf{e}_{1:k})\mathbf{P}(\mathbf{e}_{k+1:t}|\mathbf{X}_k, \mathbf{e}_{1:k})$ (using Bayes' rule, given $\mathbf{e}_{1:k}$)
  $= \alpha\mathbf{P}(\mathbf{X}_k|\mathbf{e}_{1:k})\mathbf{P}(\mathbf{e}_{k+1:t}|\mathbf{X}_k) = \alpha f_{1:k} \times \mathbf{b}_{k+1:t}$ (using conditional independence)
- **Backward message** analogous to the forward message.
- The backward message $\mathbf{b}_{k+1:t}$ can be computed by a recursive process that runs *backward* from t:
  - $\mathbf{P}(\mathbf{e}_{k+1:t}|\mathbf{X}_t) = \sum_{\mathbf{x}_{k+1}} \mathbf{P}(\mathbf{e}_{k+1:t}|\mathbf{X}_k, \mathbf{x}_{k+1})\mathbf{P}(\mathbf{x}_{k+1}|\mathbf{X}_k)$
    $= \sum_{\mathbf{x}_{k+1}} \mathbf{P}(\mathbf{e}_{k+1:t}|\mathbf{x}_{k+1})\mathbf{P}(\mathbf{x}_{k+1}|\mathbf{X}_k) = \sum_{\mathbf{x}_{k+1}} \mathbf{P}(\mathbf{e}_{k+1}, \mathbf{e}_{k+2:t}|\mathbf{x}_{k+1})\mathbf{P}(\mathbf{x}_{k+1}|\mathbf{X}_k)$
    $= \sum_{\mathbf{x}_{k+1}} \mathbf{P}(\mathbf{e}_{k+1}|\mathbf{x}_{k+1})\mathbf{P}(\mathbf{e}_{k+2}|\mathbf{x}_{k+1})\mathbf{P}(\mathbf{x}_{k+1}|\mathbf{X}_k)$

sensor model *recursion* *transition model*

12.2 Inference in Temporal Models (5/7)

Forward-backward Algorithm for smoothing

- Cache forward messages along the way
- Time linear in t (polytree inference), space $O(t|\boldsymbol{f}|)$

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 $1, \ldots, t$

  $\mathbf{fv}[0] \leftarrow prior$
  for $i = 1$ **to** t **do**
    $\mathbf{fv}[i] \leftarrow \text{FORWARD}(\mathbf{fv}[i-1], \mathbf{ev}[i])$
  for $i = t$ **down to** 1 **do**
    $\mathbf{sv}[i] \leftarrow \text{NORMALIZE}(\mathbf{fv}[i] \times \mathbf{b})$
    $\mathbf{b} \leftarrow \text{BACKWARD}(\mathbf{b}, \mathbf{ev}[i])$
  return sv

<출처> Stuart J. Russell and Peter Norvig (2021). Artificial Intelligence: A Modern Approach (4th Edition). Pearson

12.2 Inference in Temporal Models (6/7)

Finding the most likely sequence

- Most likely sequence ≠ sequence of most likely states.
- Most likely path to each $\mathbf{x}_{t+1}$ = most likely path to some $\mathbf{x}_t$ with another step
- $\max_{\mathbf{x}_1 \ldots \mathbf{x}_t} \mathbf{P}(\mathbf{x}_1, \ldots, \mathbf{x}_t, \mathbf{X}_{t+1} | \mathbf{e}_{1:t+1})$

  $= \alpha \mathbf{P}(\mathbf{e}_{t+1} | \mathbf{X}_{t+1}) \max_{\mathbf{x}_t} [\mathbf{P}(\mathbf{X}_{t+1} | \mathbf{x}_t) \max_{\mathbf{x}_1 \ldots \mathbf{x}_{t-1}} \mathbf{P}(\mathbf{x}_1, \ldots, \mathbf{x}_{t-1}, \mathbf{x}_t | \mathbf{e}_{1:t})]$
- Identical to filtering, except $\boldsymbol{f}_{1:t}$ replaced by the message

$$\mathbf{m}_{1:t} = \max_{\mathbf{x}_1 \ldots \mathbf{x}_{t-1}} \mathbf{P}(\mathbf{x}_1, \ldots, \mathbf{x}_{t-1}, \mathbf{X}_t | \mathbf{e}_{1:t})$$

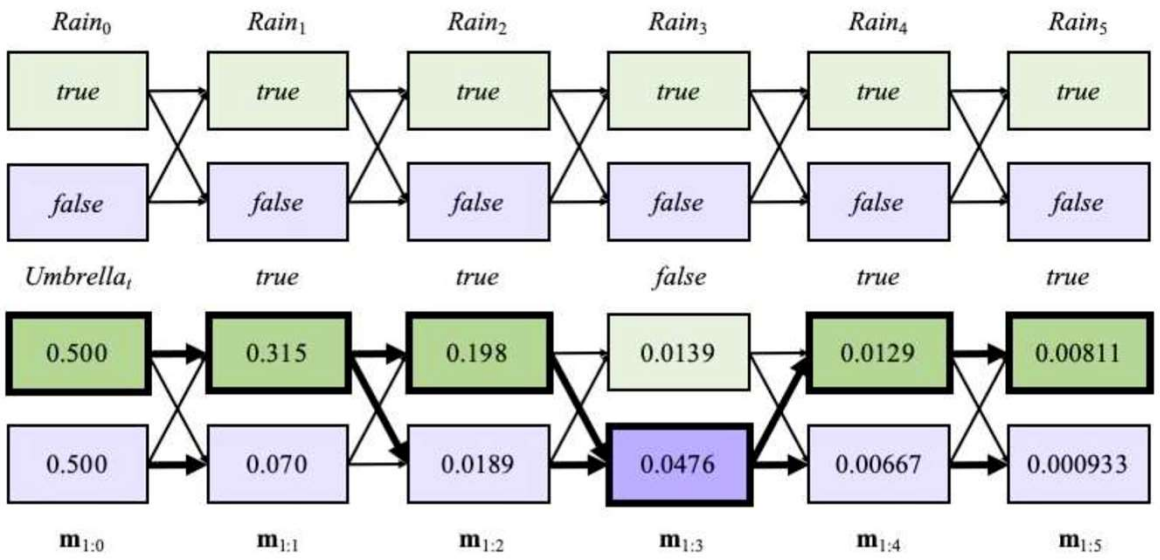
- Update has sum replaced by max, giving the Viterbi algorithm:

$$\mathbf{m}_{1:t+1} = \mathbf{P}(\mathbf{e}_{t+1} | \mathbf{X}_{t+1}) \max_{\mathbf{x}_t} \mathbf{P}(\mathbf{X}_{t+1} | \mathbf{x}_t) \, \mathbf{m}_{1:t}$$

12.2 Inference in Temporal Models (7/7)

Finding the most likely sequence

- $\mathbf{m}_{1:t} = \max_{\mathbf{x}_1 \ldots \mathbf{x}_{t-1}} \mathbf{P}(\mathbf{x}_1, \ldots, \mathbf{x}_{t-1}, \mathbf{X}_t | \mathbf{e}_{1:t})$
- $\mathbf{m}_{1:t+1} = \mathbf{P}(\mathbf{e}_{t+1} | \mathbf{X}_{t+1}) \max_{\mathbf{x}_t} \mathbf{P}(\mathbf{X}_{t+1} | \mathbf{x}_t) \mathbf{m}_{1:t}$



Operation of the Viterbi algorithm for the umbrella observation sequence. [true,true,false,true,true]

<출처> Stuart J. Russell and Peter Norvig (2021). Artificial Intelligence: A Modern Approach (4th Edition). Pearson

12.3 Hidden Markov Models




12.3 Hidden Markov Models (1/3)

Hidden Markov Model (HMM)

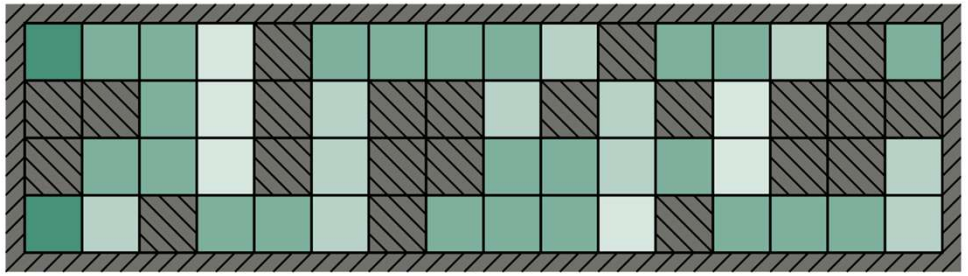
- X_t is a single, discrete variable and X_t have values denoted by integers $1, \ldots, S$
- **Transition matrix** $\mathbf{T} = \mathbf{P}(X_t|X_{t-1}) = \begin{pmatrix} 0.7 & 0.3 \\ 0.3 & 0.7 \end{pmatrix}$ ($S \times S$ Matrix)
  - $\mathbf{T}_{ij} = P(X_t = j|X_{t-1} = i)$
- **Sensor matrix**
  - $\mathbf{O}_t$ (observation matrix) for each time step, diagonal elements $P(e_t|X_t = i)$
- **Forward and backward messages**
  - $\mathbf{f}_{1:t+1} = \alpha \mathbf{O}_{t+1} \mathbf{T}^\top \mathbf{f}_{1:t}$
  - $\mathbf{b}_{1:t+1} = \mathbf{T} \mathbf{O}_{t+1} \mathbf{b}_{k+2:t}$

12.3 Hidden Markov Models (2/3)

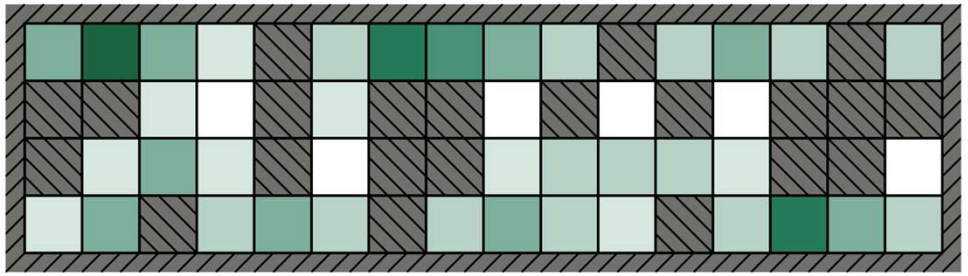
HMM example: Localization

$$P(X_{t+1} = j | X_t = i) = \mathbf{T}_{ij} = \begin{cases} \frac{1}{N(i)} & \text{if } j \in \text{NEIGHBORS}(i) \\ 0 & \text{otherwise.} \end{cases}$$

- $P(E_t = e_t | X_t = i) = (\mathbf{O}_t)_{ii}$
  $= (1-\epsilon)^{4-d_{it}} \epsilon^{d_{it}}$
- $\mathbf{P}(X_1 | E_1 = 1011)$
- $\mathbf{P}(X_2 | E_1 = 1011, E_2 = 1010)$



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



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

Figure 14.7 Posterior distribution over robot location: (a) after one observation $E_1 = 1011$ (i.e., obstacles to the north, south, and west); (b) after a random move to an adjacent location and a second observation $E_2 = 1010$ (i.e., obstacles to the north and south). The darkness of each square corresponds to the probability that the robot is at that location. The sensor error rate for each bit is $\epsilon = 0.2$.

<출처> Stuart J. Russell and Peter Norvig (2021). Artificial Intelligence: A Modern Approach (4th Edition). Pearson

12.3 Hidden Markov Models (3/3)

Performance of HMM Localization

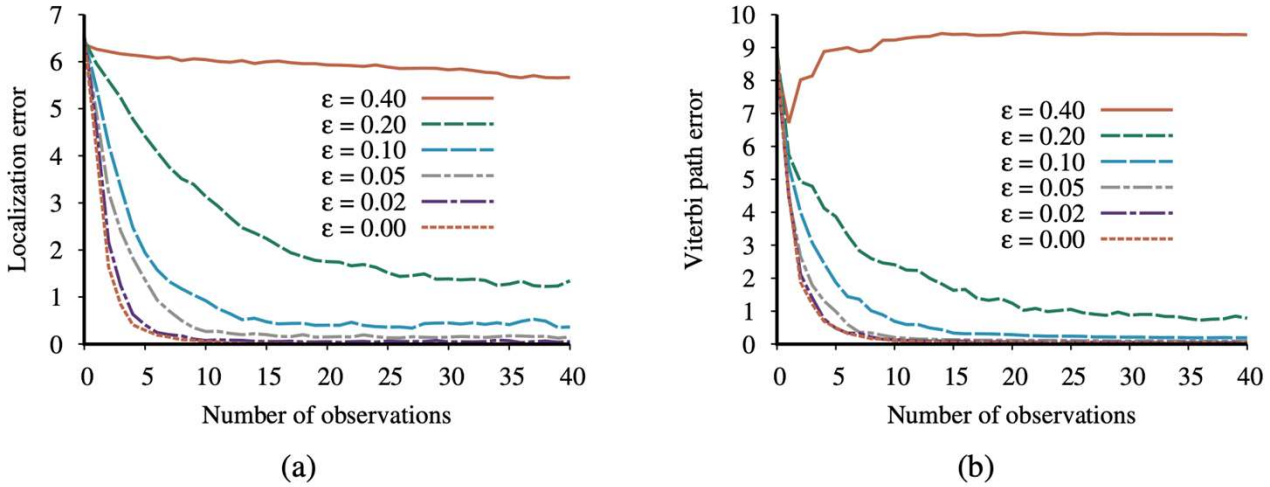


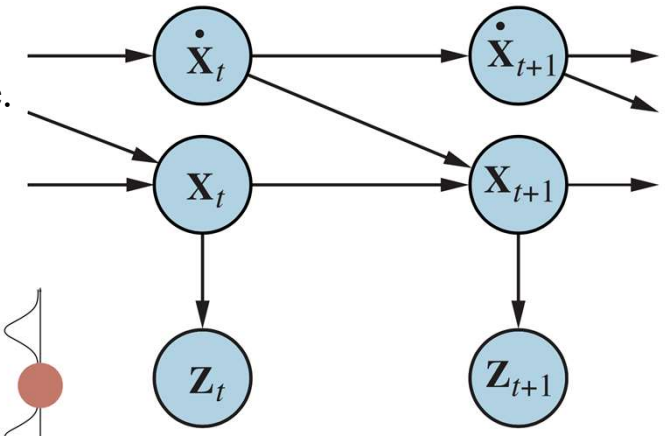
Figure 14.8 Performance of HMM localization as a function of the length of the observation sequence for various different values of the sensor error probability ϵ; data averaged over 400 runs. (a) The localization error, defined as the Manhattan distance from the true location. (b) The Viterbi path error, defined as the average Manhattan distance of states on the Viterbi path from corresponding states on the true path.

<출처> Stuart J. Russell and Peter Norvig (2021). Artificial Intelligence: A Modern Approach (4th Edition). Pearson

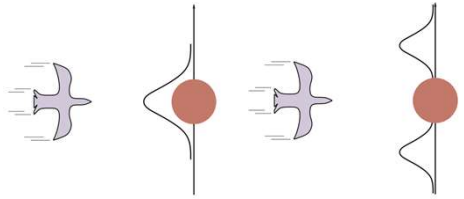
12.4 Kalman Filters

12.4 Kalman Filters (1/4)

The Problem - Filtering



- Estimating state variables from noisy observations over time.
  - Discrete variables: **HMM**
  - Continuous variables: **Kalman filtering**
- Flying bird example
  - Position $\mathbf{x}_t = (X_t, Y_t, Z_t)$
  - Velocity $\dot{\mathbf{X}}_t = (\dot{X}_t, \dot{Y}_t, \dot{Z}_t)$



  - Let time interval between observations be Δ, and assume constant velocity during the interval; then the position update is given by $X_{t+\Delta} = X_t + \dot{X}\Delta$.
  - Adding Gaussian noise, we obtain a linear-Gaussian transition model:
  - $P(X_{t+\Delta} = x_{t+\Delta} | X_t = x_t, \dot{X}_t = \dot{x}_t) = \mathcal{N}(x_{t+\Delta};, x_t + \dot{x}_t\Delta, \sigma^2)$

<출처> Stuart J. Russell and Peter Norvig (2021). Artificial Intelligence: A Modern Approach (4th Edition). Pearson

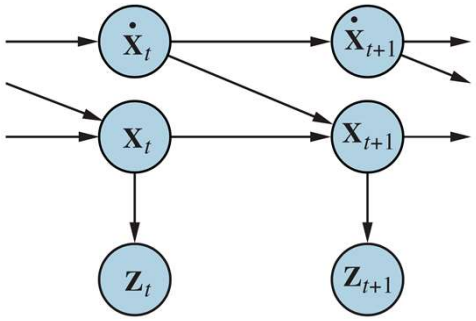
12.4 Kalman Filters (2/4)

Updating Gaussian Distribution

- If the current distribution $\mathbf{P}(\mathbf{X}|\mathbf{e}_{1:t})$ is Gaussian and the transition model $\mathbf{P}(\mathbf{X}_{t+1}|\mathbf{x}_t)$ is linear Gaussian, then the one-step predicted distribution given by the following equation is also a Gaussian distribution.

$$\mathbf{P}(\mathbf{X}_{t+1}|\mathbf{e}_{1:t}) = \int_{\mathbf{x}_t} \mathbf{P}(\mathbf{X}_{t+1}|\mathbf{x}_t) P(\mathbf{x}_t|\mathbf{e}_{1:t}) d\mathbf{x}_t$$

predicted distribution *transition* *current distribution*



- If the predicted is Gaussian and the sensor model is linear-Gaussian, then, after conditioning on the new evidence, the updated distribution is also a Gaussian distribution.

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

updated distribution *sensor model* *prediction*

<출처> Stuart J. Russell and Peter Norvig (2021). Artificial Intelligence: A Modern Approach (4th Edition). Pearson

12.4 Kalman Filters (3/4)

Kalman filter update cycle

- Full multivariate Gaussian distribution

$$\mathcal{N}(\mathbf{x};\mu,\Sigma)(x) = \alpha e^{-\frac{1}{2}((\mathbf{x}-\mu)^\top \Sigma^{-1}(\mathbf{x}-\mu))}$$

- Transition and sensor model

$$P(\mathbf{x}_{t+1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t+1};\mathbf{F}\mathbf{x}_t,\Sigma_\mathbf{x})$$
$$P(\mathbf{z}_t|\mathbf{x}_t) = \mathcal{N}(\mathbf{z}_t;\mathbf{H}\mathbf{x}_t,\Sigma_z)$$

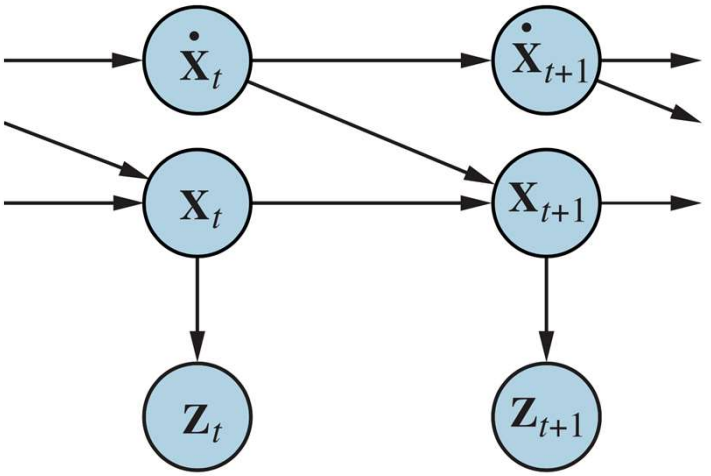
- Update for mean and covariance

$$\mu_{t+1} = \mathbf{F}\mu_t + \mathbf{K}_{t+1}(\mathbf{z}_{t+1} - \mathbf{H}\mathbf{F}_{\mu_t})$$
$$\Sigma_{t+1} = (\mathbf{I} - \mathbf{K}_{t+1}\mathbf{H})(\mathbf{F}\Sigma_t\mathbf{F}^\top + \Sigma_x)$$

- Kalman gain matrix:

$$\mathbf{K}_{t+1} = (\mathbf{F}\Sigma_t\mathbf{F}^\top + \Sigma_x)\mathbf{H}^\top(\mathbf{H}(\mathbf{F}\Sigma_t\mathbf{F}^\top + \Sigma_x)\mathbf{H}^\top + \Sigma_z)^{-1}$$

<출처> Stuart J. Russell and Peter Norvig (2021). Artificial Intelligence: A Modern Approach (4th Edition). Pearson



12.4 Kalman Filters (4/4)

Result of Kalman filtering for an object moving on the X-Y plane

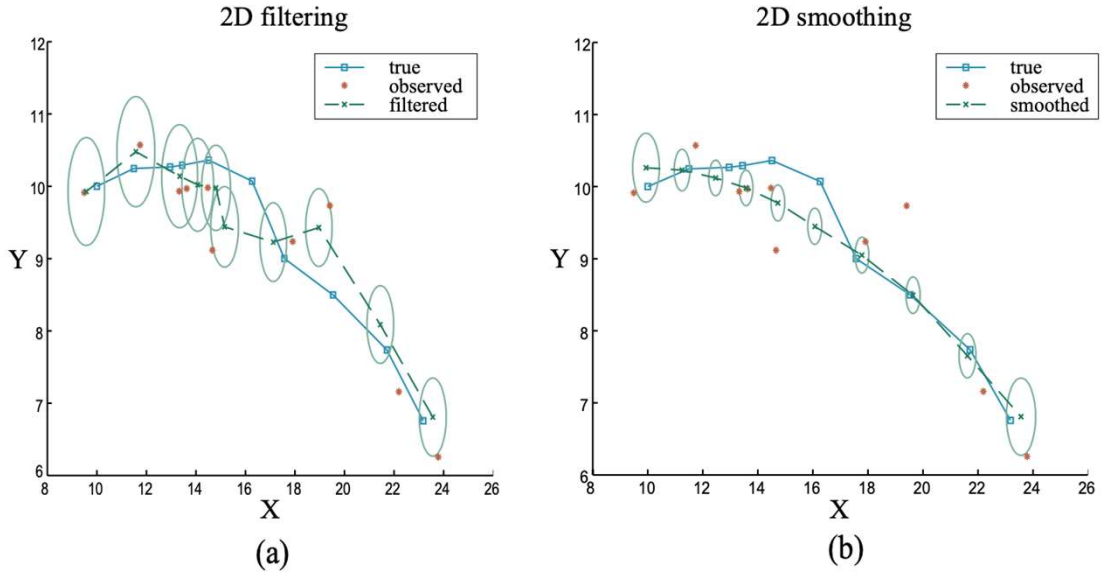


Figure 14.11 (a) Results of Kalman filtering for an object moving on the X–Y plane, showing the true trajectory (left to right), a series of noisy observations, and the trajectory estimated by Kalman filtering. Variance in the position estimate is indicated by the ovals. (b) The results of Kalman smoothing for the same observation sequence.

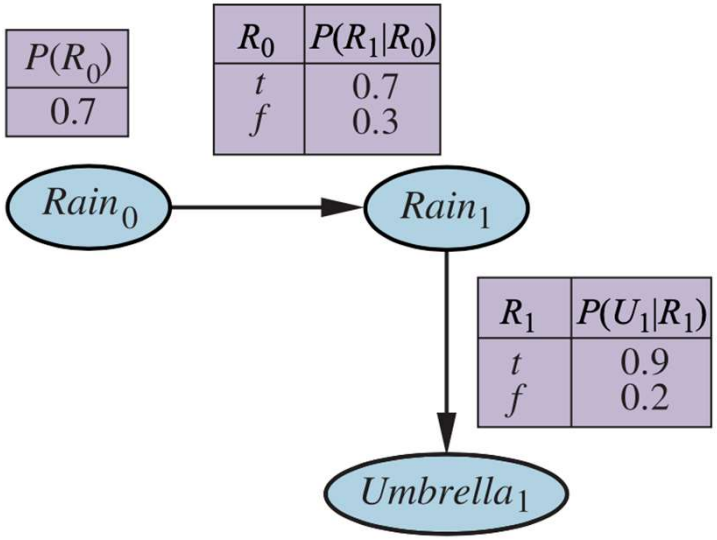
<출처> Stuart J. Russell and Peter Norvig (2021). Artificial Intelligence: A Modern Approach (4th Edition). Pearson

12.5 Dynamic Bayesian Networks

12.5 Dynamic Bayesian Networks (1/7)

Dynamic Bayesian Networks (DBNs)

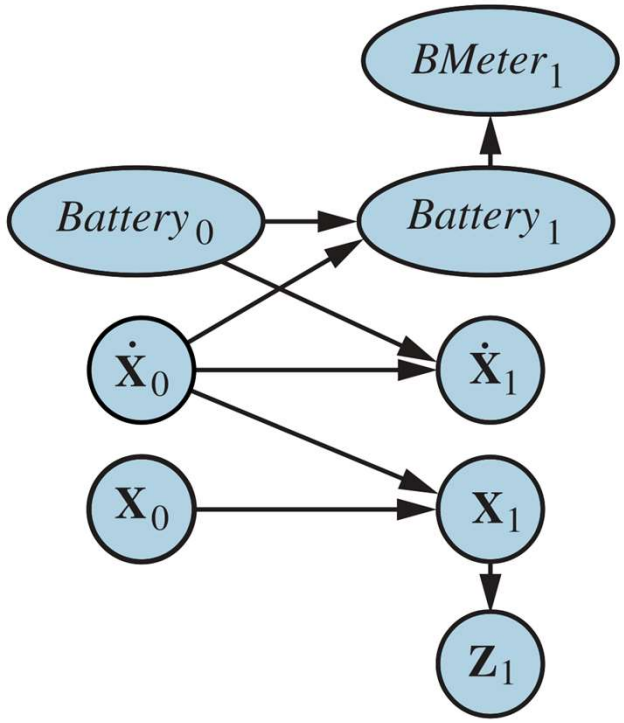
- Bayesian network that represents a temporal probability model.
- Assumptions and variables
  - Markov process assumption.
  - State variable $\mathbf{X}_t$, evidence variable $\mathbf{E}_t$.
- Formulation
  - Transition model: $\mathbf{P}(\mathbf{X}_{t+1}|\mathbf{X}_t)$
  - Sensor model: $\mathbf{P}(\mathbf{E}_t|\mathbf{X}_t)$
- Applying to continuous variables
  - Use derivates of variables with variables.



<출처> Stuart J. Russell and Peter Norvig (2021). Artificial Intelligence: A Modern Approach (4th Edition). Pearson

12.5 Dynamic Bayesian Networks (2/7)

Dynamic Bayesian Network for robot motion in X-Y plane



<출처> Stuart J. Russell and Peter Norvig (2021). Artificial Intelligence: A Modern Approach (4th Edition). Pearson

12.5 Dynamic Bayesian Networks (3/7)

Unrolling a Dynamic Bayesian Network

- **Unrolling**: Transition and sensor models shared, so we can repeat recursively. Slices are replicated to accommodate the observation sequence.
  - Once unrolled, exact inferences for Bayesian networks can be used.

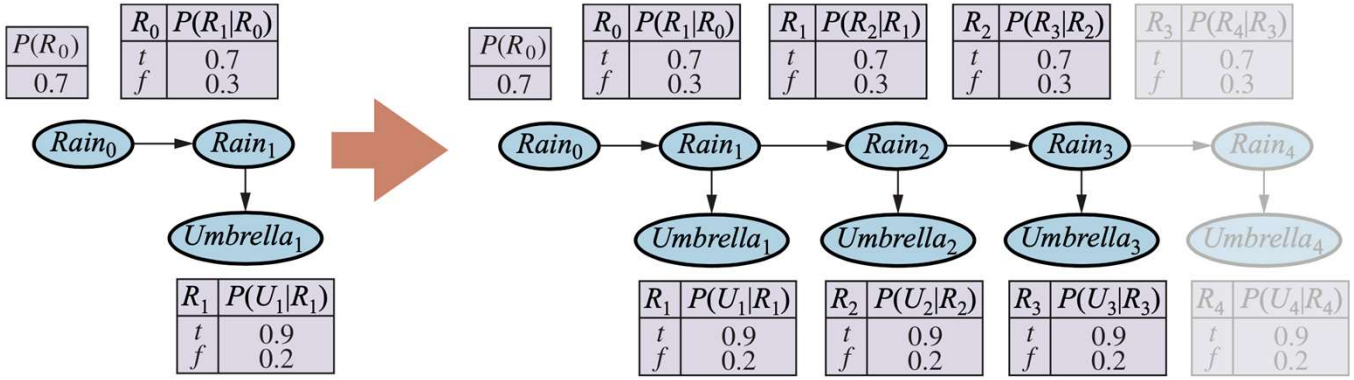


Figure 14.16 Unrolling a dynamic Bayesian network: slices are replicated to accommodate the observation sequence $Umbrella_{1:3}$. Further slices have no effect on inferences within the observation period.

<출처> Stuart J. Russell and Peter Norvig (2021). Artificial Intelligence: A Modern Approach (4th Edition). Pearson

12.5 Dynamic Bayesian Networks (4/7)

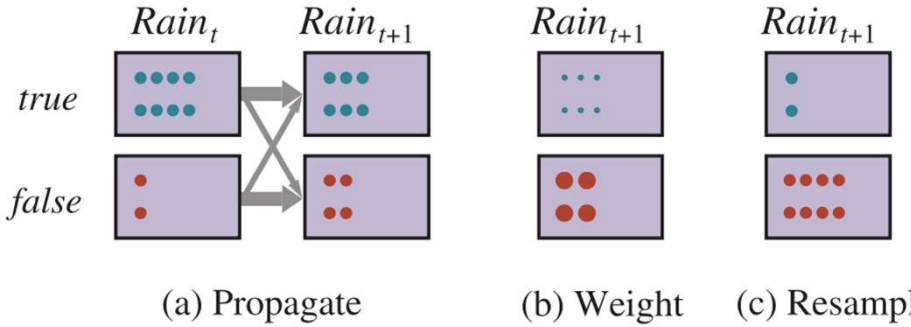
Appropriate inference in DBNs (by Particle Filtering)

- MCMC, Likelihood weighting

- Particle Filtering (a.k.a. Sequential importance sampling with resampling)
  - Use the samples themselves as an approximate representation of the current state distribution.
  - Focus the set of samples on the high-probability regions of the state space.

12.5 Dynamic Bayesian Networks (5/7)

- **Particle Filtering** works as follows: First we generate a population of N samples from the prior distribution $\mathbf{P}(\mathbf{X}_0)$. Then the **update cycle** is repeated for each time step:
- 1. Each sample is propagated forward by sampling the next state value $\mathbf{x}_{t+1}$ give the current state $\mathbf{x}_t$, for the sample, based on the transition model $\mathbf{P}(\mathbf{X}_{t+1}|\mathbf{x}_t)$
- 2. Each sample is weighted by the likelihood it assigns to the new evidence, $\mathbf{P}(\mathbf{e}_{t+1})$.
- 3. The population is *resampled* to generate a new population of N samples. Each new sample is selected from the current population; this probability that a particular sample is selected is proportional to its weight. New samples are unweighted.



<출처> Stuart J. Russell and Peter Norvig (2021). Artificial Intelligence: A Modern Approach (4th Edition). Pearson

12.5 Dynamic Bayesian Networks (6/7)

Performance of the standard particle filtering algorithm

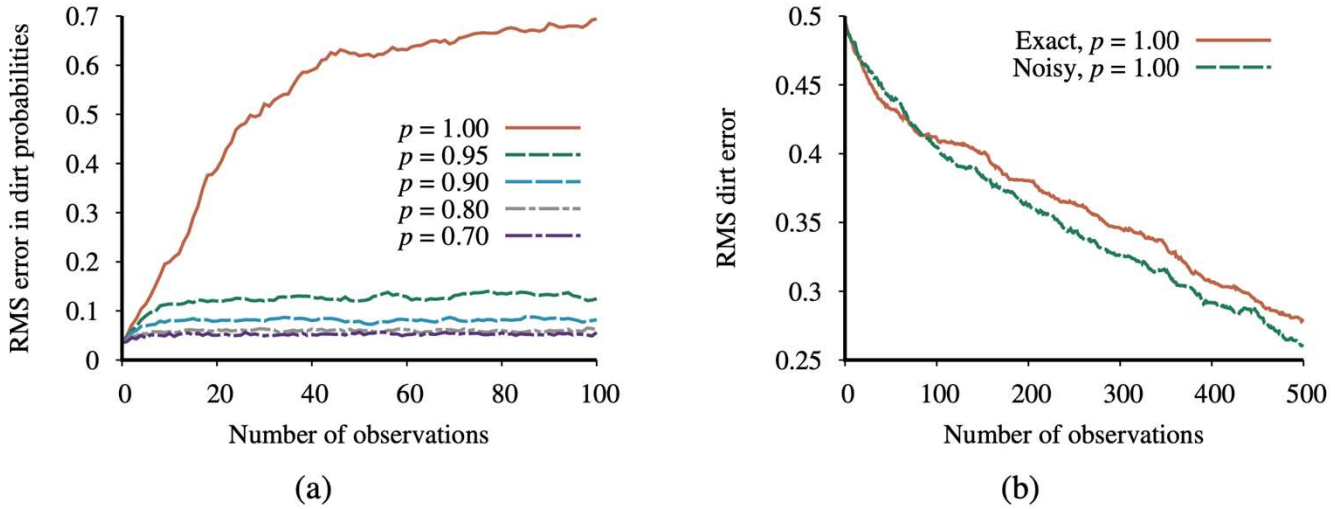


Figure 14.21 (a) Performance of the standard particle filtering algorithm with 1,000 particles, showing RMS error in marginal dirt probabilities compared to exact inference for different values of the dirt persistence p. (b) Performance of Rao-Blackwellized particle filtering (100 particles) compared to ground truth, for both exact location sensing and noisy wall sensing and with deterministic dirt. Data averaged over 20 runs.

<출처> Stuart J. Russell and Peter Norvig (2021). Artificial Intelligence: A Modern Approach (4th Edition). Pearson

12.5 Dynamic Bayesian Networks (7/7)

Rao-Blackwellization

- Based on the idea that exact inference is always more accurate than sampling.

- For the SLAM problem, we run particle filtering on the robot location and then, for each particle, we run exact HMM inference for each dirt square independently, conditioned on the location sequence in that particle.

Summary

1. Representations can be designed to satisfy the Markov property, so that the future is independent of the past given the present. Combined with the assumption that the process is time-homogenous, this greatly simplifies the representation.
2. A temporal probability model contains a transition model describing the state evolution and a sensor model describing the observation process.
3. The principal inference tasks in temporal models are filtering (state estimation), prediction, smoothing, and computing the most likely explanation.
4. These families of temporal models were studied in more depth: Hidden Markov Models, Kalman Filters, and Dynamic Bayesian Networks.
5. Unless special assumptions are made, as in Kalman filters, exact inference with many state variables in intractable.
6. In practice, the particle filtering algorithm and its descendants are an effective family approximate algorithms.