Probabilistic Reasoning Over Time

Bùi Tiến Lên

2022

KHOA CÔNG NGHỆ THÔNG TIN
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

Contents

Notation

symbol	meaning
$a, b, c, N \ldots$	scalar number
$\boldsymbol{w}, \boldsymbol{v}, \boldsymbol{x}, \boldsymbol{y} \ldots$	column vector
$\boldsymbol{X}, \boldsymbol{Y} \ldots$	matrix
$\mathbb{R}$	set of real numbers
$\mathbb{Z}$	set of integer numbers
$\mathbb{N}$	set of natural numbers
$\mathbb{R}^D$	set of vectors
$\mathcal{D}, \mathcal{X}, \mathcal{Y}, \ldots$	set
$\mathcal{A}$	algorithm

symbol	meaning
$X, Y \ldots$	random variable
$\boldsymbol{X}, \boldsymbol{Y} \ldots$	multivariate random variable
$x, y \ldots$	value
$\boldsymbol{x}, \boldsymbol{y} \ldots$	vector
p, pr, P, Pr	probability

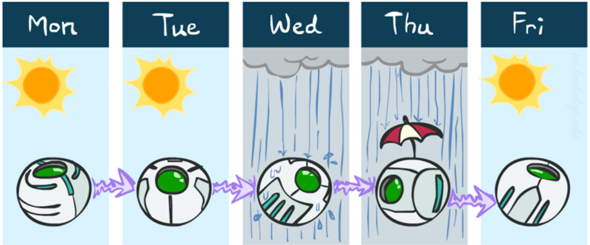
Temporal Probabilistic Model

- Temporal Probabilistic Model
- Markov process

Temporal Probabilistic Model

Why do we need temporal probabilistic model?

- The world changes over time (**random process**), and what happens now impacts what will happen in the future. We need to **track** and **predict** it
  - Stock market
  - Weather



Temporal Probabilistic Model (cont.)

How to model the time?

- View the world as time slices: discrete time steps; step size depends on problem

Temporal Probabilistic Model (cont.)

Concept 1

- **State variables $\boldsymbol{X}_t$** (often hidden): set of **unobservable state variables** at time t
  - State of the environment
  - Not directly observable but defines causal dynamics
- **Evidence variables $\boldsymbol{E}_t$**: set of **observable evidence variables** at time t
  - Caused by the state of the environment

- **Notation**: $\boldsymbol{X}_{a:b} = \boldsymbol{X}_a, \boldsymbol{X}_{a+1}, \ldots, \boldsymbol{X}_{b-1}, \boldsymbol{X}_b$

Temporal Probabilistic Model (cont.)

Concept 2

- **Transition model**: How the world evolves

$$P(\boldsymbol{X}_t \mid \boldsymbol{X}_{0:t-1}) \tag{1}$$

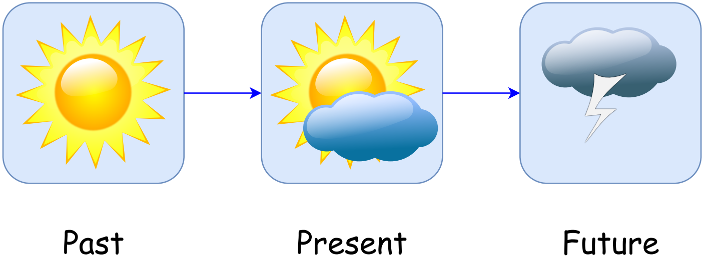
- **Sensor/observation model**: How the evidence variables get their values

$$P(\boldsymbol{E}_t \mid \boldsymbol{X}_{0:t}, \boldsymbol{E}_{0:t-1}) \tag{2}$$

Markov process

Concept 3

Markov process is a random process with **Markov assumption** "$\boldsymbol{X}_t$ depends on *bounded* subset of $\boldsymbol{X}_{0:t-1}$"



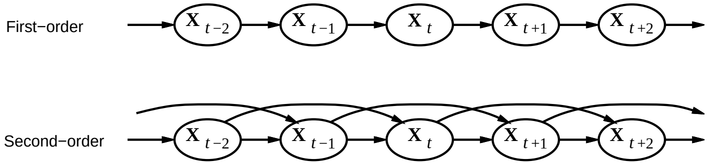
Markov process (cont.)

- **First-order Markov process**:

$$P(\boldsymbol{X}_t \mid \boldsymbol{X}_{0:t-1}) = P(\boldsymbol{X}_t \mid \boldsymbol{X}_{t-1}) \tag{3}$$

- **Second-order Markov process**:

$$P(\boldsymbol{X}_t \mid \boldsymbol{X}_{0:t-1}) = P(\boldsymbol{X}_t \mid \boldsymbol{X}_{t-2}, \boldsymbol{X}_{t-1}) \tag{4}$$



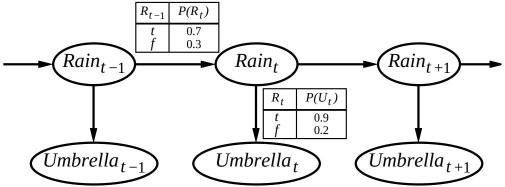
Markov process (cont.)

- **Sensor Markov assumption**:

$$P(\boldsymbol{E}_t \mid \boldsymbol{X}_{0:t}, \boldsymbol{E}_{0:t-1}) = P(\boldsymbol{E}_t \mid \boldsymbol{X}_t) \tag{5}$$

- **Stationary process**:
  - transition model $P(\boldsymbol{X}_t \mid \boldsymbol{X}_{t-1})$ fixed for all t
  - sensor model $P(\boldsymbol{E}_t \mid \boldsymbol{X}_t)$ fixed for all t

Example



- First-order Markov assumption may not be exactly true in real world!
- Possible fixes:
  1. **Increase order** of Markov process
  2. **Augment state**, e.g., add $Temp_t$, $Pressure_t$

Inference in Temporal Models

- Filtering
- Smoothing
- Most likely explanation

Inference tasks

- **Filtering**: computing the **belief state**, the posterior distribution over the most recent state given all evidence to date

$$P(X_t \mid e_{1:t})$$

- **Prediction**: computing the posterior distribution over the future state, given all evidence to date

$$P(X_{t+k} \mid e_{1:t}) \text{ for } k > 0$$

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

$$P(X_k \mid e_{1:t}) \text{ for } 0 \leq k < t$$

- **Most likely explanation**: given a sequence of observations, find the sequence of states that is most likely to have generated those observations

$$\arg\max_{x_{1:t}} P(x_{1:t} \mid e_{1:t})$$

Filtering

- Given the result of filtering up to time t, we needs to compute the result for $t+1$ from the new evidence e_{t+1},

$$\begin{aligned} P(X_{t+1} \mid e_{1:t+1}) &= P(X_{t+1} \mid e_{1:t}, e_{t+1}) \\ &= \alpha P(e_{t+1} \mid X_{t+1}, e_{1:t}) P(X_{t+1} \mid e_{1:t}) \\ &= \alpha P(e_{t+1} \mid X_{t+1}) P(X_{t+1} \mid e_{1:t}) \end{aligned}$$

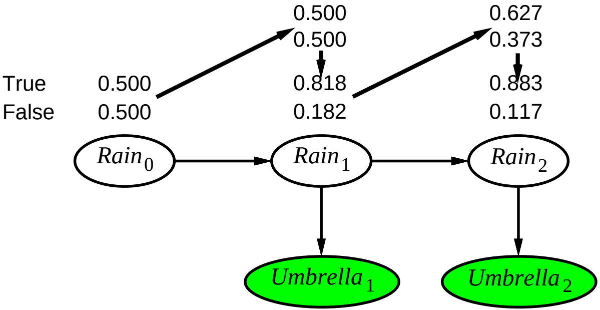
- We obtain the one-step prediction for the next state by conditioning on the current state X_t:

$$\begin{aligned} P(X_{t+1} \mid e_{1:t+1}) &= \alpha P(e_{t+1} \mid X_{t+1}) \sum_{x_t} P(X_{t+1} \mid x_t, e_{1:t}) P(x_t \mid e_{1:t}) \\ &= \alpha P(e_{t+1} \mid X_{t+1}) \sum_{x_t} P(X_{t+1} \mid x_t) P(x_t \mid e_{1:t}) \qquad (6) \end{aligned}$$

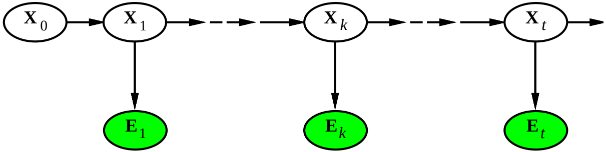
- The process is given by

$$f_{1:t+1} = \textsc{Forward}(f_{1:t}, e_{t+1}) \text{ where } f_{1:t} = P(X_t \mid e_{1:t})$$

Filtering example



Smoothing



- Divide evidence $e_{1:t}$ into $e_{1:k}$, $e_{k+1:t}$:

$$\begin{aligned} P(X_k \mid e_{1:t}) &= P(X_k \mid e_{1:k}, e_{k+1:t}) \\ &= \alpha P(X_k \mid e_{1:k}) P(e_{k+1:t} \mid X_k, e_{1:k}) \\ &= \alpha P(X_k \mid e_{1:k}) P(e_{k+1:t} \mid X_k) \\ &= \alpha f_{1:k} b_{k+1:t} \end{aligned} \tag{7}$$

Smoothing (cont.)

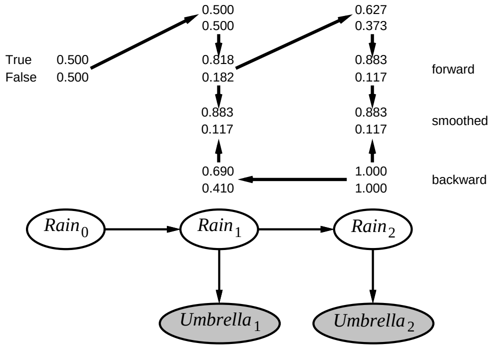
- Backward message computed by a backwards recursion:

$$\begin{aligned} P(e_{k+1:t} \mid X_k) &= \sum_{x_{k+1}} P(e_{k+1:t} \mid X_k, x_{k+1}) P(x_{k+1} \mid X_k) \\ &= \sum_{x_{k+1}} P(e_{k+1:t} \mid x_{k+1}) P(x_{k+1} \mid X_k) \\ &= \sum_{x_{k+1}} P(e_{k+1} \mid x_{k+1}) P(e_{k+2:t} \mid x_{k+1}) P(x_{k+1} \mid X_k) \quad (8) \end{aligned}$$

- The process is given by

$$b_{k+1:t} = \textsc{Backward}(b_{k+2:t}, e_{t+1}) \text{ where } b_{k+1:t} = P(e_{k+1:t} \mid X_k)$$

Smoothing example



- **Forward–backward** algorithm: cache forward messages along the way
  Time linear in t (polytree inference), space $O(t\,|f|)$

Most likely explanation

- Most likely sequence $\neq$ sequence of most likely states!
- Most likely path to each x_{t+1} = most likely path to *some* x_t plus one more step

$$\max P(x_1, \ldots, x_t, X_{t+1} \mid e_{1:t+1})$$
$$= P(e_{t+1} \mid X_{t+1}) \max_{x_t} \left(P(X_{t+1} \mid x_t) \max_{x_1 \ldots x_{t-1}} P(x_1, \ldots, x_{t-1}, x_t \mid e_{1:t}) \right)$$

- Identical to filtering, except $f_{1:t}$ replaced by

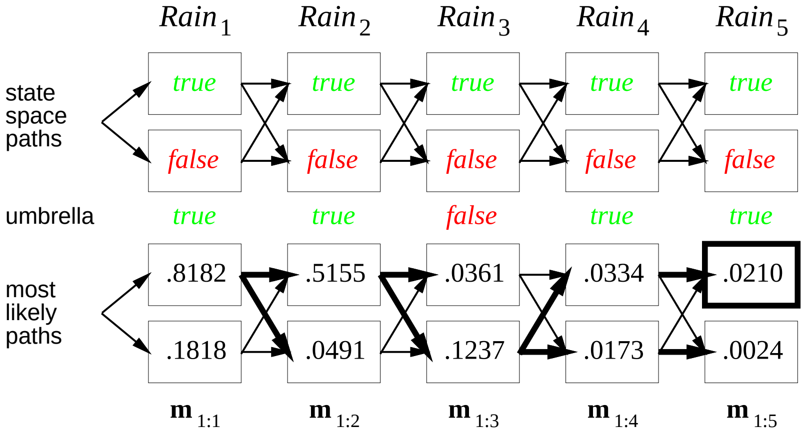
$$m_{1:t} = \max_{x_1 \ldots x_{t-1}} P(x_1, \ldots, x_{t-1}, X_t \mid e_{1:t}),$$

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

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

$$m_{1:t+1} = P(e_{t+1} \mid X_{t+1}) \max_{x_t} \left(P(X_{t+1} \mid x_t) m_{1:t} \right)$$

Viterbi example



Hidden Markov Models

- Example
- Hidden Markov Models

Problem

Consider a temporal probabilistic model with two random variables

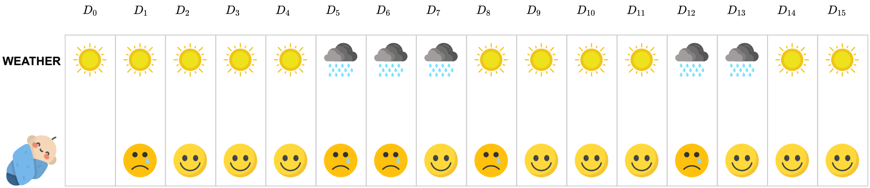
- Weather: *sunny* or *rainny*
- Baby: *happy* or *sad*

WEATHER

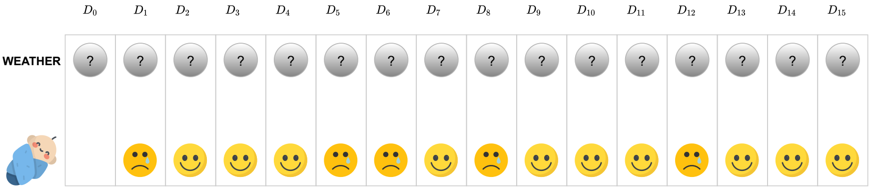
BABY

Data

- Complete observations in 15 days



- Partial observations in 15 days



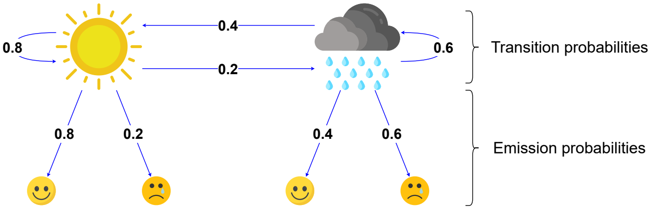
Markov Model

- Markov model with the following assumptions
  - Transition probabilities

$$P(weather_{t+1} \mid weather_t)$$

  - Emission probabilities

$$P(baby_t \mid weather_t)$$



Questions

Learning

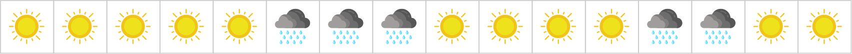
1. How did we find these probabilities?

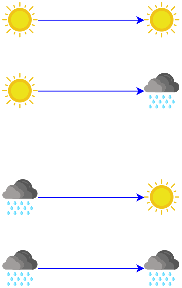
Inference

1. What's the probability that a random day is Sunny or Rainy?
2. If Baby is Happy today, what's the probability that it's Sunny or Rainy?
3. If for three days Baby is Happy, Sad, Happy, what was the weather?

How did we find these probabilities?

- Transition probabilities

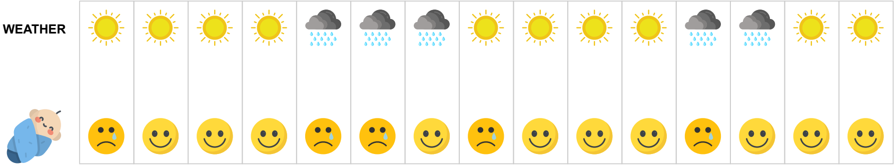
WEATHER 



counts	probabilities
8	0.8
2	0.2
2	0.4
3	0.6

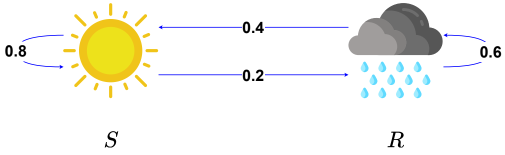
How did we find these probabilities? (cont.)

- Emission probabilities



counts	8	2	2	3
probabilities	0.8	0.2	0.4	0.6

Sunny or Rainny



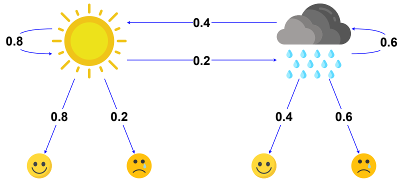
- We have the equations

$$S = 0.8S + 0.4R$$
$$R = 0.2S + 0.6R$$

- Solve the equations

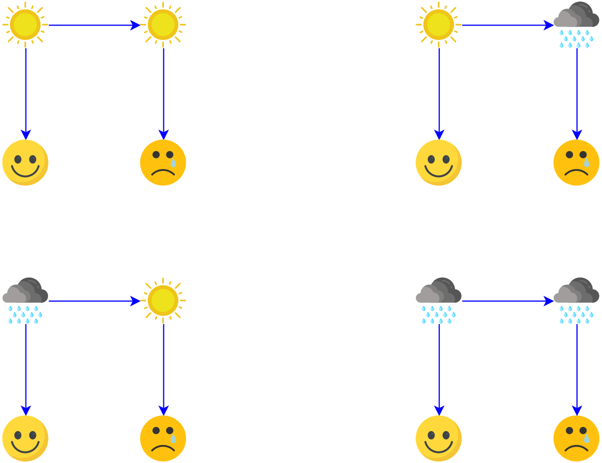
$$S = 2/3$$
$$R = 1/3$$

Today, Baby is happy



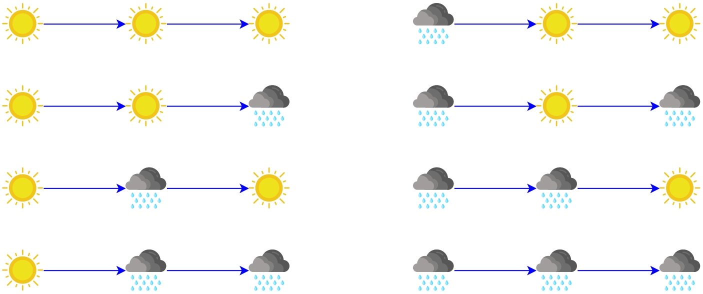
What was the weather?

- If for two days Baby is Happy, Sad, what was the weather?

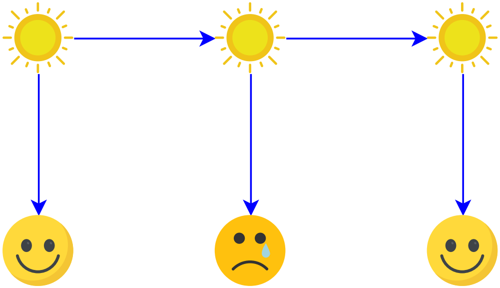


What was the weather? (cont.)

- If for three days Baby is Happy, Sad, Happy, what was the weather?

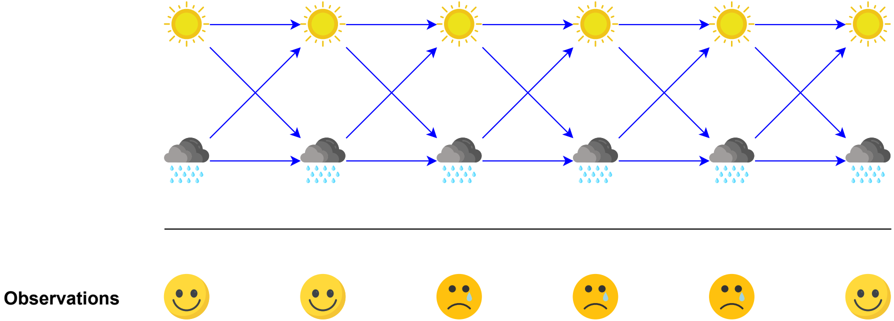


What was the weather? (cont.)



Viterbi algorithm

- If for three days Baby is Happy, Happy, Sad, Sad, Sad and Happy, what was **the most likely weather**?



Hidden Markov models

Concept 4

Hidden Markov model is a temporal probabilistic model where X_t is a single, discrete variable (usually E_t is too) and domain of X_t is $\{1, \ldots, S\}$

- **Transition matrix** T_{ij} for each time step

$$T_{ij} = P(X_t = j \mid X_{t-1} = i)$$

- **Sensor matrix** O_t for each time step, diagonal elements

$$P(e_t \mid X_t = i)$$

- For example, $T = \begin{pmatrix} 0.7 & 0.3 \\ 0.3 & 0.7 \end{pmatrix}$, $O = \begin{pmatrix} 0.9 & 0 \\ 0 & 0.2 \end{pmatrix}$ or $\begin{pmatrix} 0.1 & 0 \\ 0 & 0.8 \end{pmatrix}$

Forward-backward messages

- Forward and backward messages as column vectors:

$$\begin{aligned} f_{1:t+1} &= \alpha O_{t+1} T^{\mathsf{T}} f_{1:t} \\ b_{k+1:t} &= T O_{k+1} b_{k+2:t} \end{aligned}$$

- Forward-backward algorithm needs time $O(S^2 t)$ and space $O(St)$

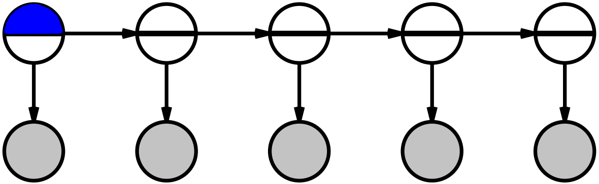
Country dance algorithm

Can avoid storing all forward messages in smoothing by running forward algorithm backwards:

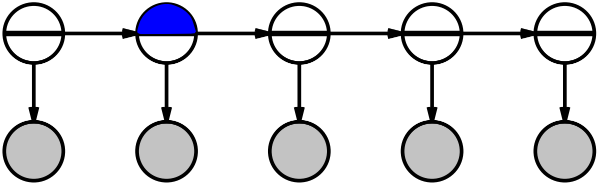
$$\begin{aligned} f_{1:t+1} &= \alpha O_{t+1} T^{\intercal} f_{1:t} \\ O_{t+1}^{-1} f_{1:t+1} &= \alpha T^{\intercal} f_{1:t} \\ \alpha' (T^{\intercal})^{-1} O_{t+1}^{-1} f_{1:t+1} &= f_{1:t} \end{aligned}$$

Algorithm: forward pass computes f_t, backward pass does f_i, b_i

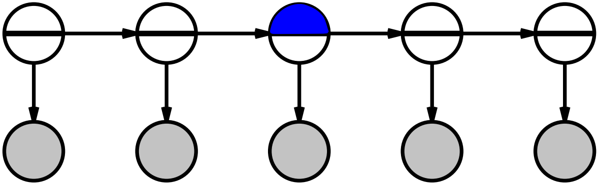
Country dance algorithm (cont.)



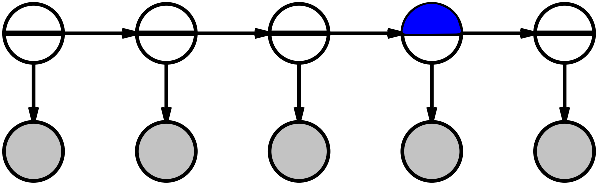
Country dance algorithm (cont.)



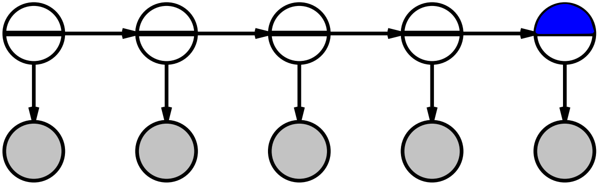
Country dance algorithm (cont.)



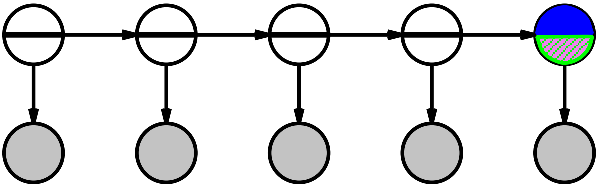
Country dance algorithm (cont.)



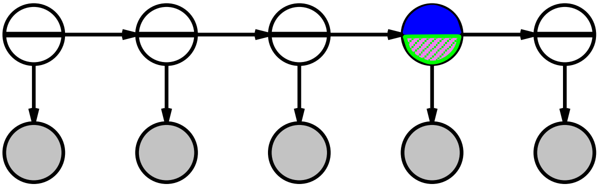
Country dance algorithm (cont.)



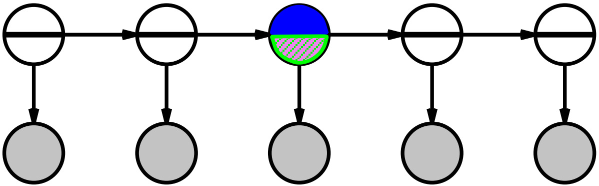
Country dance algorithm (cont.)



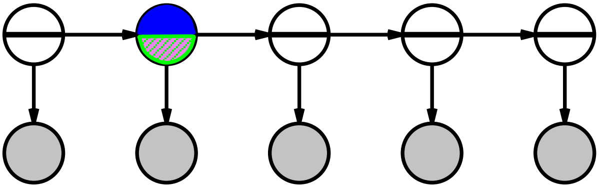
Country dance algorithm (cont.)



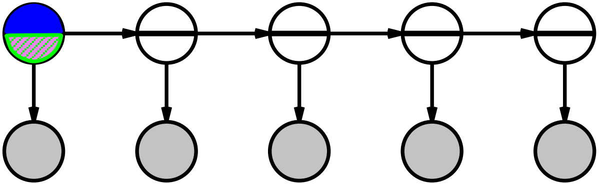
Country dance algorithm (cont.)



Country dance algorithm (cont.)



Country dance algorithm (cont.)



Kalman Filters

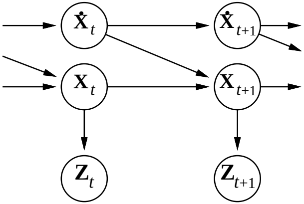
Kalman filters

Concept 5

Kalman Filter is a temporal probabilistic model that represents systems described by a set of continuous variables with **assumptions**: Gaussian prior, linear Gaussian transition model and sensor model

- Tracking a bird flying

$$\boldsymbol{X}_t = (position, velocity) = (x, y, z, \dot{x}, \dot{y}, \dot{z})$$



Updating Gaussian distributions

Prediction step:

- If $P(X_t \mid e_{1:t})$ is Gaussian and the transition model $P(X_{t+1} \mid x_t)$ is linear Gaussian, then prediction

$$P(X_{t+1} \mid e_{1:t}) = \int_{x_t} P(X_{t+1} \mid x_t) P(x_t \mid e_{1:t})\, dx_t$$

  is Gaussian.
- If $P(X_{t+1} \mid e_{1:t})$ is Gaussian and the sensor model $P(e_{t+1}|X_{t+1})$ is linear Gaussian, then the updated distribution

$$P(X_{t+1} \mid e_{1:t+1}) = \alpha P(e_{t+1} \mid X_{t+1}) P(X_{t+1} \mid e_{1:t})$$

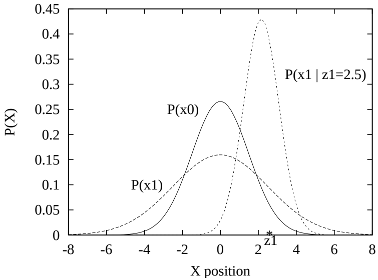
  is Gaussian
- Hence $P(X_t \mid e_{1:t})$ is multivariate Gaussian $\mathcal{N}(\boldsymbol{\mu}_t, \Sigma_t)$ for all t

Simple 1-D example

- Gaussian random walk on X–axis, transition s.d. σ_x, sensor s.d. σ_z

$$\mu_{t+1} = \frac{(\sigma_t^2 + \sigma_x^2) z_{t+1} + \sigma_z^2 \mu_t}{\sigma_t^2 + \sigma_x^2 + \sigma_z^2}$$

$$\sigma_{t+1}^2 = \frac{(\sigma_t^2 + \sigma_x^2)\sigma_z^2}{\sigma_t^2 + \sigma_x^2 + \sigma_z^2}$$



General Kalman update

- Transition and sensor models:

$$\begin{aligned} P(\boldsymbol{x}_{t+1} \mid \boldsymbol{x}_t) &= \mathcal{N}(F\boldsymbol{x}_t, \Sigma_x)(\boldsymbol{x}_{t+1}) \\ P(\boldsymbol{z}_t \mid \boldsymbol{x}_t) &= \mathcal{N}(H\boldsymbol{x}_t, \Sigma_z)(\boldsymbol{z}_t) \end{aligned} \tag{9}$$

  F is the matrix for the transition; Σ_x the transition noise covariance
  H is the matrix for the sensors; Σ_z the sensor noise covariance

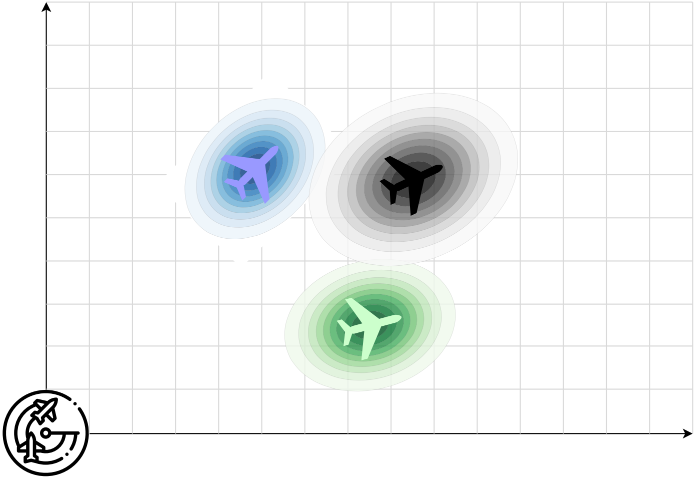
- Filter computes the following update:

$$\begin{aligned} \boldsymbol{\mu}_{t+1} &= F\boldsymbol{\mu}_t + K_{t+1}(\boldsymbol{z}_{t+1} - HF\boldsymbol{\mu}_t) \\ \Sigma_{t+1} &= (I - K_{t+1})(F\Sigma_t F^\intercal + \Sigma_x) \end{aligned} \tag{10}$$

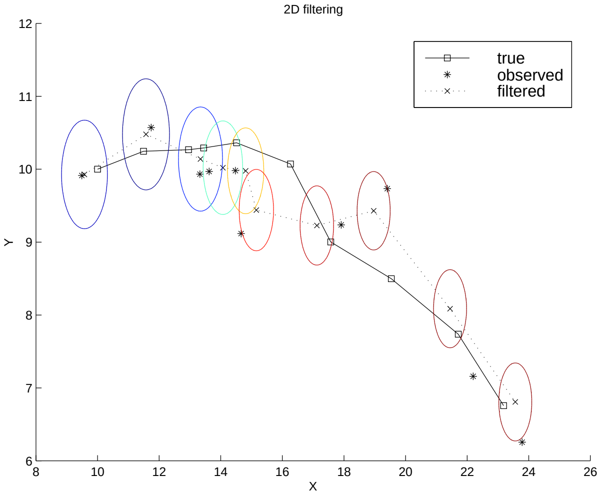
  where $K_{t+1} = (F\Sigma_t F^\intercal + \Sigma_x)H^\intercal(H(F\Sigma_t F^\intercal + \Sigma_x)H^\intercal + \Sigma_z)^{-1}$ is the **Kalman gain matrix**

- Σ_t and K_t are independent of observation sequence, so compute offline

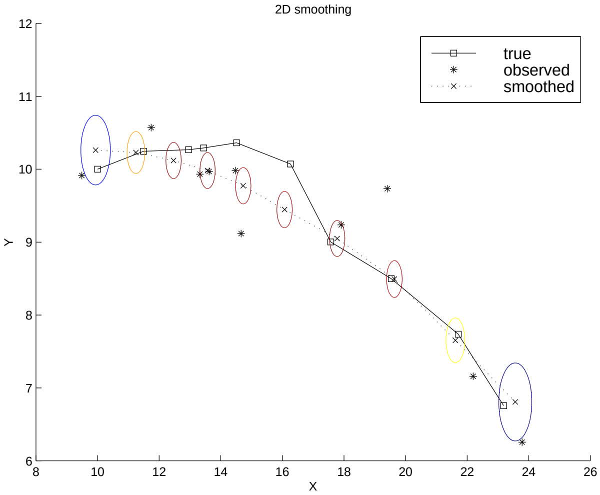
2-D tracking example



2-D tracking example: filtering

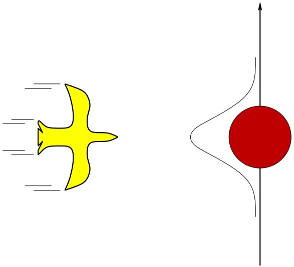


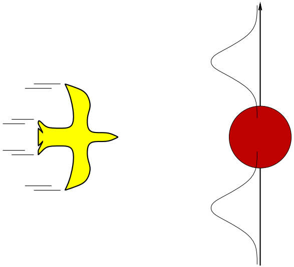
2-D tracking example: smoothing



Where it breaks

- Cannot be applied if the transition model is nonlinear
- **Extended Kalman Filter** models transition as *locally linear* around $x_t = \mu_t$
  - Fails if systems is locally unsmooth



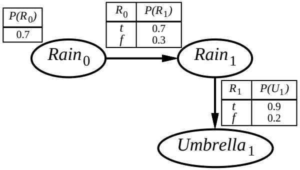


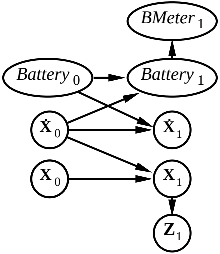
Dynamic Bayesian Networks

- Particle Filtering

Dynamic Bayesian networks

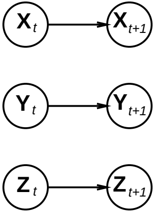
- X_t, E_t contain arbitrarily many variables in a replicated Bayes net

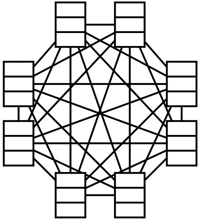




DBNs vs. HMMs

- Every HMM is a single-variable DBN; every discrete DBN is an HMM

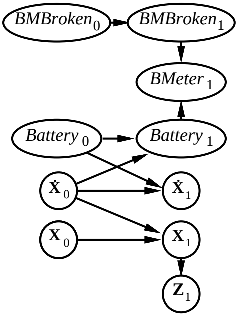


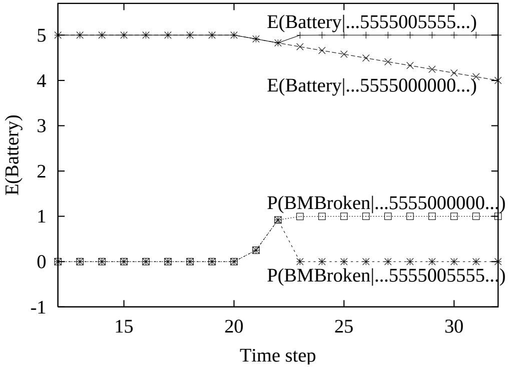


- Sparse dependencies $\Rightarrow$ exponentially fewer parameters; e.g., 20 state variables, three parents each
- DBN has $20 \times 2^3 = 160$ parameters, HMM has $2^{20} \times 2^{20} \approx 10^{12}$

DBNs vs Kalman filters

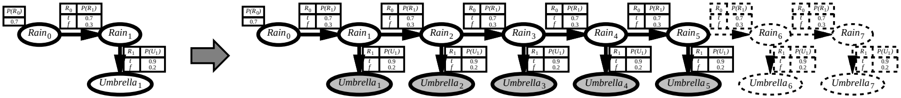
- Every Kalman filter model is a DBN, but few DBNs are KFs; real world requires non-Gaussian posteriors
  E.g., where are bin Laden and my keys? What's the battery charge?





Exact inference in DBNs

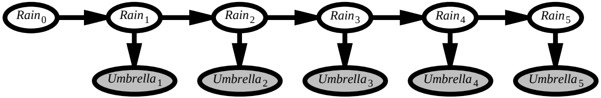
- Naive method: **unroll** the network and run any exact algorithm



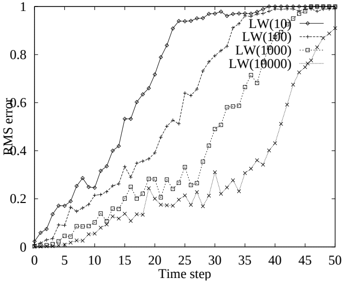
- Problem: inference cost for each update grows with t
- **Rollup filtering**: add slice $t+1$, "sum out" slice t using variable elimination
- Largest factor is $O(d^{n+1})$, update cost $O(d^{n+2})$ (cf. HMM update cost $O(d^{2n})$)

Likelihood weighting for DBNs

- Set of weighted samples approximates the belief state



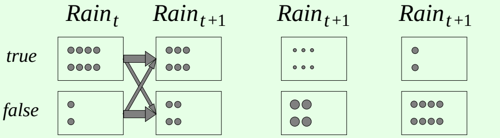
- LW samples pay no attention to the evidence!
  $\Rightarrow$ fraction "agreeing" falls exponentially with t
  $\Rightarrow$ number of samples required grows exponentially with t



Particle filtering

Concept 6

Particle filtering is a model to focus on the population of samples (“particles”) that tracks the high-likelihood regions of the state-space



(a) Propagate (b) Weight (c) Resample

- Widely used for tracking nonlinear systems, esp. in vision
- Also used for simultaneous localization and mapping in mobile robots 10^5-dimensional state space

Filter task

Writing $N(\boldsymbol{x}_t \mid \boldsymbol{e}_{1:t})$ for the number of samples occupying state $\boldsymbol{x}_t$ after observations $\boldsymbol{e}_{1:t}$ have been processed. Assume consistent at time t:

$$N(\boldsymbol{x}_t \mid \boldsymbol{e}_{1:t})/N = P(\boldsymbol{x}_t \mid \boldsymbol{e}_{1:t}) \tag{11}$$

- **Propagate** forward populations of $\boldsymbol{x}_{t+1}$:

$$N(\boldsymbol{x}_{t+1} \mid \boldsymbol{e}_{1:t}) = \sum_{x_t} P(\boldsymbol{x}_{t+1} \mid \boldsymbol{x}_t) N(\boldsymbol{x}_t \mid \boldsymbol{e}_{1:t}) \tag{12}$$

- **Weight** samples by their likelihood for $\boldsymbol{e}_{t+1}$:

$$W(\boldsymbol{x}_{t+1} \mid \boldsymbol{e}_{1:t+1}) = P(\boldsymbol{e}_{t+1} \mid \boldsymbol{x}_{t+1}) N(\boldsymbol{x}_{t+1} \mid \boldsymbol{e}_{1:t}) \tag{13}$$

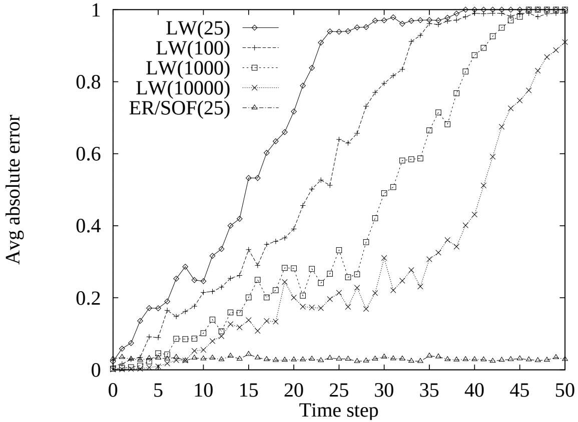
Filter task (cont.)

- **Resample** to obtain populations proportional to W:

$$
\begin{aligned}
N(\mathbf{x}_{t+1} \mid \mathbf{e}_{1:t+1})/N &= \alpha W(\mathbf{x}_{t+1} \mid \mathbf{e}_{1:t+1}) \\
&= \alpha P(\mathbf{e}_{t+1} \mid \mathbf{x}_{t+1}) N(\mathbf{x}_{t+1} \mid \mathbf{e}_{1:t}) \\
&= \alpha P(\mathbf{e}_{t+1} \mid \mathbf{x}_{t+1}) \sum_{x_t} P(\mathbf{x}_{t+1} \mid x_t) N(\mathbf{x}_t \mid \mathbf{e}_{1:t}) \\
&= \alpha P(\mathbf{e}_{t+1} \mid \mathbf{x}_{t+1}) \sum_{x_t} P(\mathbf{x}_{t+1} \mid \mathbf{x}_t) P(\mathbf{x}_t \mid \mathbf{e}_{1:t}) \\
&= P(\mathbf{x}_{t+1} \mid \mathbf{e}_{1:t+1}) \qquad (14)
\end{aligned}
$$

Particle filtering performance

- Approximation error of particle filtering remains bounded over time, at least empirically—theoretical analysis is difficult



Keeping Track of Many Objects

Summary

- Temporal models use state and sensor variables replicated over time
- Markov assumptions and stationarity assumption, so we need
  - transition model $P(X_t \mid X_{t-1})$
  - sensor model $P(E_t \mid X_t)$
- Tasks are filtering, prediction, smoothing, most likely sequence; *all done recursively with constant cost per time step*
- Hidden Markov models have a single discrete state variable; used for speech recognition
- Kalman filters allow n state variables, linear Gaussian, $O(n^3)$ update
- Dynamic Bayes nets subsume HMMs, Kalman filters; exact update intractable
- Particle filtering is a good approximate filtering algorithm for DBNs

References

Goodfellow, I., Bengio, Y., and Courville, A. (2016).
Deep learning.
MIT press.

Lê, B. and Tô, V. (2014).
Cở sở trí tuệ nhân tạo.
Nhà xuất bản Khoa học và Kỹ thuật.

Russell, S. and Norvig, P. (2021).
Artificial intelligence: a modern approach.
Pearson Education Limited.