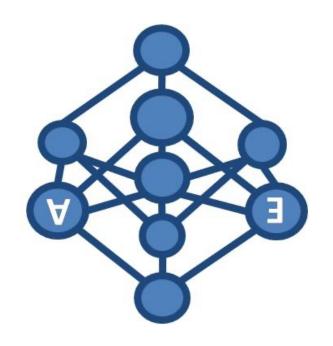
Probabilistic Graphical Models* Dynamic Bayesian Networks





*Thanks to Carlos Guestrin, Pedro Domingos and many others for making their slides publically available





So far

- Bayesian networks
- Inference within Bayesian networks
- Learning Bayesian networks

All rather static situations ...





What's next

- The world is not static but evolves over time
 - Monitoring freeway traffic (from an autonomous driver or for traffic analysis)
 - Monitoring patient's symptoms
- How can we model dynamic environments?
- Hidden Markov Models
- (Bayes Filter and Kalman Filters)
- Dynamic Bayesian Networks





Hidden Markov Models

Let's consider an example: Part-of-speech tagging





Example 1: Parts of Speech

- Part of speech (POS): Noun, verb, adjective, preposition, adverb, article, interjection, pronoun, conjunction, etc.
- POS tagging: The process of assigning (labeling) a part-of-speech or other lexical class marker to each word in a sentence (or a corpus)
 - Decide whether each word is a noun, verb, adjective, ...
- Also called word-category disambiguation





POS Tagging Statistical Approaches: Idea 1

Simply assign each word its most likely POS.

Success rate: 91%!

Word	POS listings in Brown				
heat		noun/89		verb/5	
oil		noun/87			
in	р	rep/2073	1	noun/1	adv/462
а	C	let/22943	3	noun/50	noun-proper/30
large	adj/354		noun/2	adv/5	
pot		noun/27			



Statistical Approaches: Idea 2

For a string of words

$$W = w_1 w_2 w_3 ... w_n$$

find the string of POS tags

$$T = t_1 t_2 t_3 ... t_n$$

which maximizes P(T|W)

 i.e., the probability of tag string T given that the word string was W, i.e., that W was tagged T





The Sparse Data Problem ...

A simple but unrealistic approach to compute P(T|W):

Count instances of the string "heat oil in a large pot" in the training corpus, and pick the most common tag assignment to the string.





So, what can we estimate?

- What parameters can we estimate with a million words of hand tagged training data?
 - Assume a uniform distribution of 5000 words and 40 part of speech tags..

Event	Count	Estimate Quality?
tags	40	Excellent
tag bigrams	1600	Excellent
tag trigrams	64,000	OK
tag 4-grams	2.5M	Poor
words	5000	Very Good
word bigrams	25M	Poor
word x tag pairs	200,000	OK

- Rich Models often require vast amounts of data
- Good estimates of models with bad assumptions often outperform better models which are badly estimated (recall Naïve Bayes!)





A Practical Statistical Tagger

By Bayes' Rule:

$$P(T|W) = \frac{P(W|T) * P(T)}{P(W)}$$

so to maximize P(T|W), need to maximize P(W|T) * P(T).

To compute P(T): By the chain rule,

$$P(T) = P(t_1) * P(t_2|t_1) * P(t_3|t_1t_2) * \dots * P(t_n|t_1 \dots t_{n-1})$$



A Practical Statistical Tagger II

But we cannot accurately estimate more than tag bigrams! Thus, we simply change to a model that we CAN estimate:

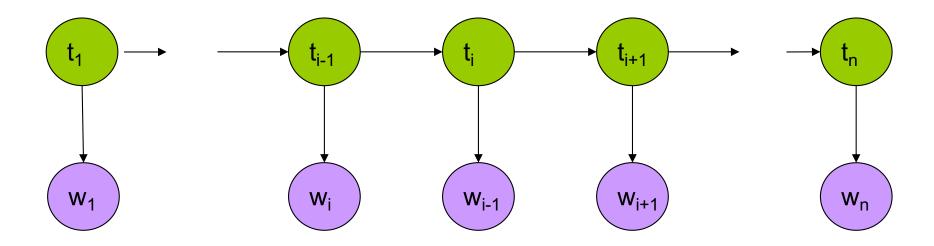
A Markov Assumption: $P(t_i|t_1...t_{i-1}) = P(t_i|t_{i-1})$

By which

$$P(T) = P(t_1) * P(t_2|t_1) * P(t_3|t_2) * \dots * P(t_n|t_{n-1})$$



A Graphical Representation



$$P(T) * P(W|T) = P(t_1) * P(t_2|t_1) * P(t_3|t_2) * \dots * P(t_n|t_{n-1}) * P(w_1|t_1) * P(w_2|t_2) * \dots * P(w_n|t_n)$$

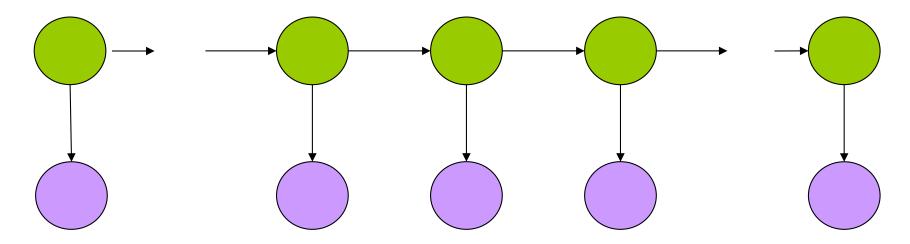


 This is an example of a Hidden Markov Model (HMM)

So, what are HMMs?

What is an HMM?



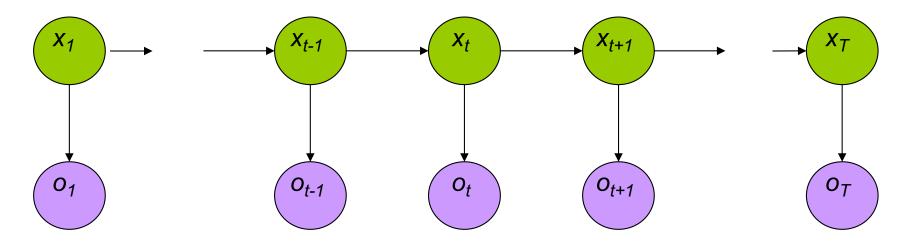


- Graphical Model
- Circles indicate states
- Arrows indicate probabilistic dependencies between states

HMM = Hidden Markov Model

HMMs and Bayesian Nets



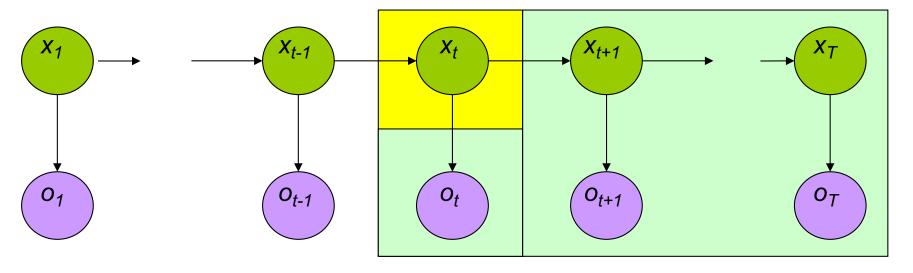


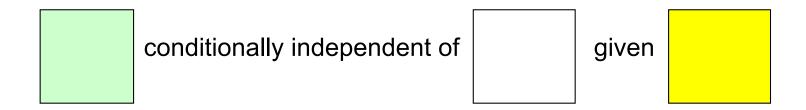
$$P(x_{1}...x_{T}, o_{1}...o_{T}) = P(x_{1})P(o_{1} \mid x_{1}) \prod_{i=1}^{T-1} P(x_{i+1} \mid x_{i}) P(o_{i+1} \mid x_{i+1})$$

$$= \pi_{x_{1}} b_{x_{1}o_{1}} \prod_{t=1}^{T-1} a_{x_{t}x_{t+1}} b_{x_{t+1}o_{t+1}}$$

Independence Assumption





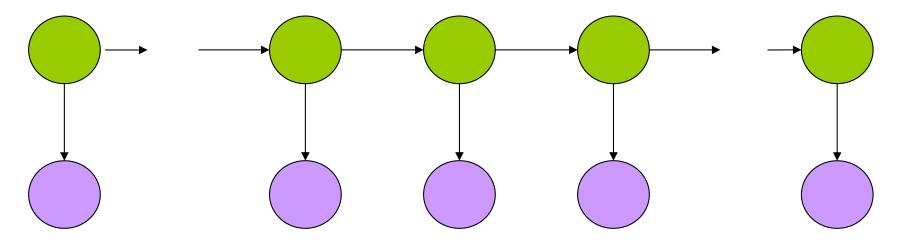


Because of d-separation

"The past is independent of the future given the present."

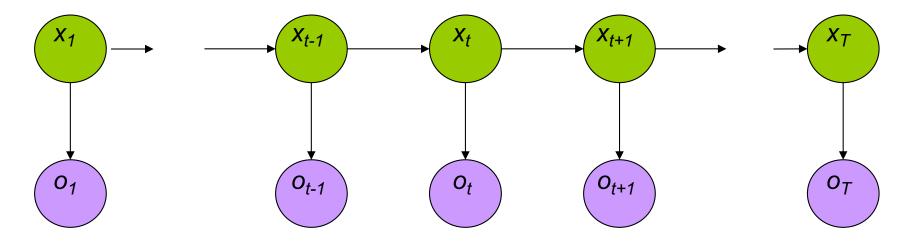
Inference in an HMM





- 1. Compute the probability of a given observation sequence (Decode)
- 2. Given an observation sequence, compute the most likely hidden state sequence (Best State Sequence)
- 3. Given an observation sequence and set of possible models, which model most closely fits the data? (Parameter Estimation)

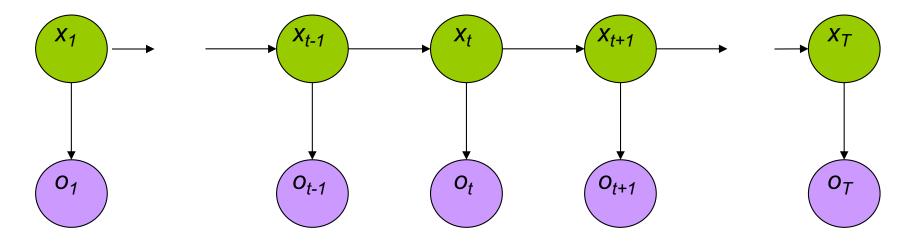




$$P(O | X, \mu) = b_{x_1 o_1} b_{x_2 o_2} ... b_{x_T o_T}$$

μ are the parameters

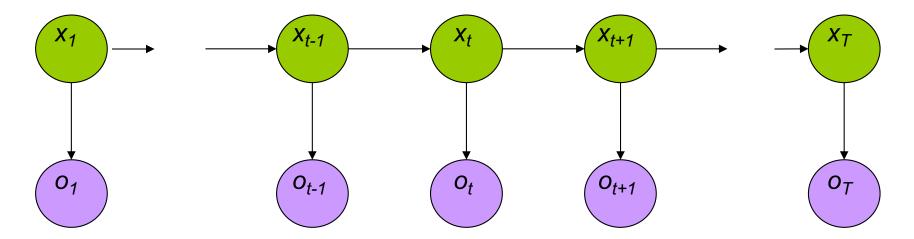




$$P(O | X, \mu) = b_{x_1 o_1} b_{x_2 o_2} ... b_{x_T o_T}$$

$$P(X \mid \mu) = \pi_{x_1} a_{x_1 x_2} a_{x_2 x_3} ... a_{x_{T-1} x_T}$$



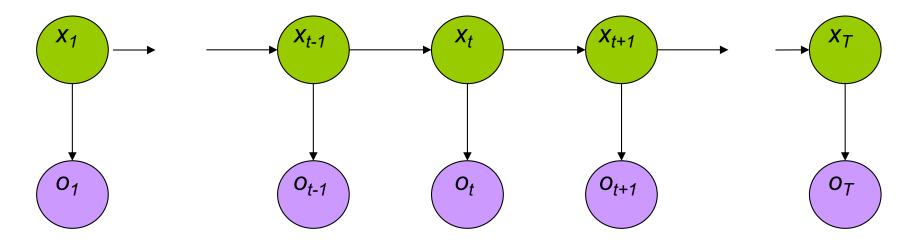


$$P(O \mid X, \mu) = b_{x_1 o_1} b_{x_2 o_2} ... b_{x_T o_T}$$

$$P(X \mid \mu) = \pi_{x_1} a_{x_1 x_2} a_{x_2 x_3} ... a_{x_{T-1} x_T}$$

$$P(O, X \mid \mu) = P(O \mid X, \mu)P(X \mid \mu)$$





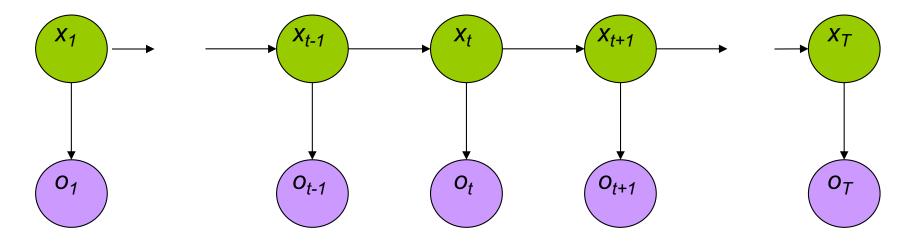
$$P(O \mid X, \mu) = b_{x_1 o_1} b_{x_2 o_2} ... b_{x_T o_T}$$

$$P(X \mid \mu) = \pi_{x_1} a_{x_1 x_2} a_{x_2 x_3} ... a_{x_{T-1} x_T}$$

$$P(O, X \mid \mu) = P(O \mid X, \mu) P(X \mid \mu)$$

$$P(O \mid \mu) = \sum_{X} P(O \mid X, \mu) P(X \mid \mu)$$





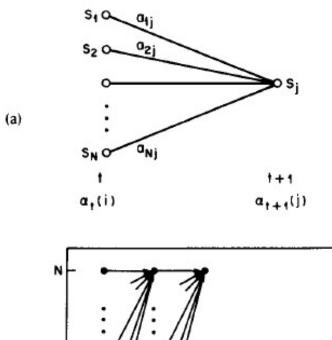
$$P(O \mid \mu) = \sum_{X} \pi_{x_1} b_{x_1 o_1} \prod_{t=1}^{T-1} a_{x_t x_{t+1}} b_{x_{t+1} o_{t+1}}$$

Complexity $O(N^T.2T)$

E.g. N = 5, T = 100 gives $2.100.5^{100} \approx 10^{72}$







Dynamic Programming to the rescue!

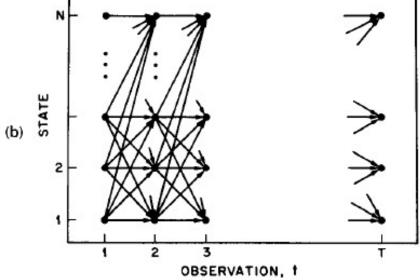
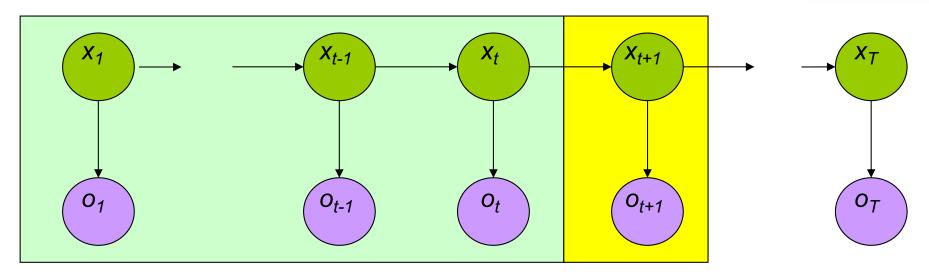


Fig. 4. (a) Illustration of the sequence of operations required for the computation of the forward variable $\alpha_{t+1}(j)$. (b) Implementation of the computation of $\alpha_t(i)$ in terms of a lattice of observations t, and states i.



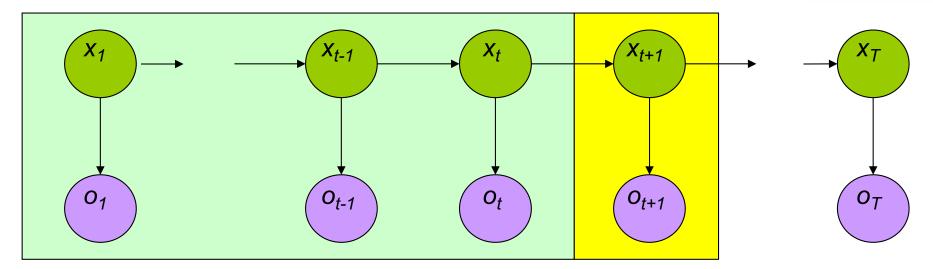


- Special structure gives us an efficient solution using dynamic programming
- **Intuition**: Probability of the first *t* observations is the same for all possible *t*+1 length state sequences
- Define:

$$\alpha_i(t) = P(o_1...o_t, x_t = i \mid \mu)$$

$$\alpha_i(1) = P(o_1, x_1 = i \mid \mu)$$
$$= \pi_i \cdot b_{io_1}$$





$$\alpha_j(t+1)$$

$= P(o_1...o_{t+1}, x_{t+1} = j)$

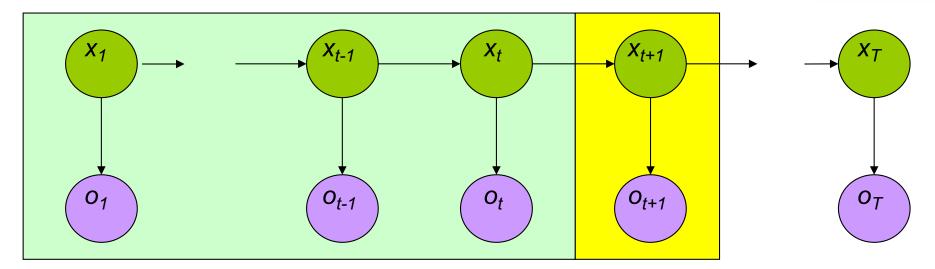
definition (µ omitted)

$$= P(o_{1}...o_{t+1} | x_{t+1} = j)P(x_{t+1} = j)$$

$$= P(o_{1}...o_{t} | x_{t+1} = j)P(o_{t+1} | x_{t+1} = j)P(x_{t+1} = j)$$

$$= P(o_{1}...o_{t}, x_{t+1} = j)P(o_{t+1} | x_{t+1} = j)$$





$$\alpha_j(t+1)$$

$$= P(o_1...o_{t+1}, x_{t+1} = j)$$

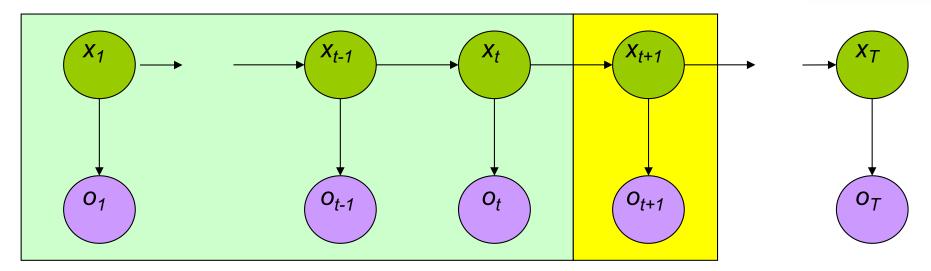
$$= P(o_1...o_{t+1} \mid x_{t+1} = j)P(x_{t+1} = j)$$

chain rule

$$= P(o_1...o_t \mid x_{t+1} = j)P(o_{t+1} \mid x_{t+1} = j)P(x_{t+1} = j)$$

=
$$P(o_1...o_t, x_{t+1} = j)P(o_{t+1} | x_{t+1} = j)$$





$$\alpha_j(t+1)$$

$$= P(o_1...o_{t+1}, x_{t+1} = j)$$

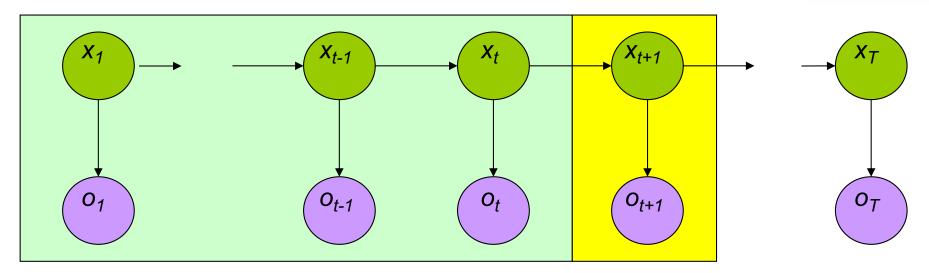
=
$$P(o_1...o_{t+1} | x_{t+1} = j)P(x_{t+1} = j)$$

$$= P(o_1...o_t \mid x_{t+1} = j)P(o_{t+1} \mid x_{t+1} = j)P(x_{t+1} = j)$$

=
$$P(o_1...o_t, x_{t+1} = j)P(o_{t+1} | x_{t+1} = j)$$

chain rule





$$\alpha_j(t+1)$$

$$= P(o_{1}...o_{t+1}, x_{t+1} = j)$$

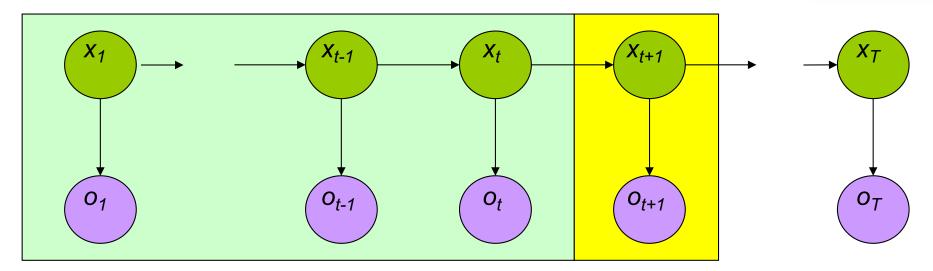
$$= P(o_{1}...o_{t+1} | x_{t+1} = j)P(x_{t+1} = j)$$

$$= P(o_{1}...o_{t} | x_{t+1} = j)P(o_{t+1} | x_{t+1} = j)P(x_{t+1} = j)$$

$$= P(o_1...o_t, x_{t+1} = j)P(o_{t+1} | x_{t+1} = j)$$

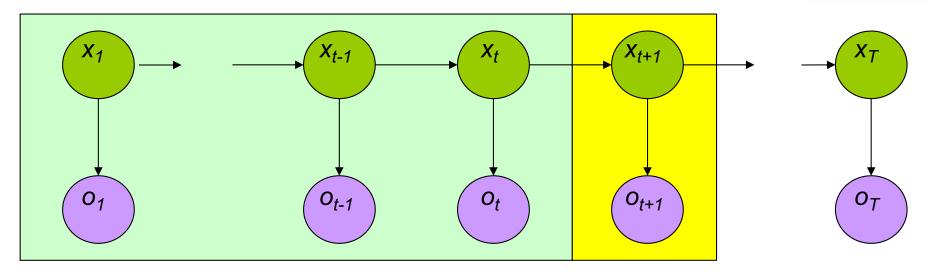
chain rule





$$\begin{split} &= \sum_{i=1\dots N} P(o_1 \dots o_t, x_t = i, x_{t+1} = j) P(o_{t+1} \mid x_{t+1} = j) & \text{Total prob} \\ &= \sum_{i=1\dots N} P(o_1 \dots o_t, x_{t+1} = j \mid x_t = i) P(x_t = i) P(o_{t+1} \mid x_{t+1} = j) \\ &= \sum_{i=1\dots N} P(o_1 \dots o_t, x_t = i) P(x_{t+1} = j \mid x_t = i) P(o_{t+1} \mid x_{t+1} = j) \\ &= \sum_{i=1\dots N} \alpha_i(t) a_{ij} b_{jo_{t+1}} \end{split}$$





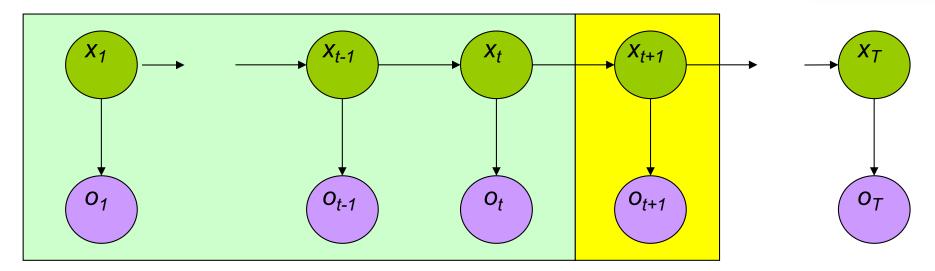
$$= \sum_{i=1...N} P(o_1...o_t, x_t = i, x_{t+1} = j) P(o_{t+1} \mid x_{t+1} = j)$$

$$= \sum_{i=1...N} P(o_1...o_t, x_{t+1} = j \mid x_t = i) P(x_t = i) P(o_{t+1} \mid x_{t+1} = j)$$
 Chain rule

$$= \sum_{i=1...N} P(o_1...o_t, x_t = i)P(x_{t+1} = j \mid x_t = i)P(o_{t+1} \mid x_{t+1} = j)$$

$$=\sum_{i=1}^{N}\alpha_{i}(t)a_{ij}b_{jo_{t+1}}$$





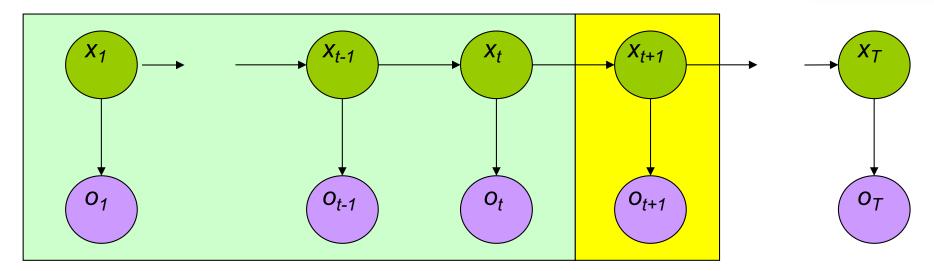
$$= \sum_{i=1...N} P(o_1...o_t, x_t = i, x_{t+1} = j) P(o_{t+1} \mid x_{t+1} = j)$$

$$= \sum_{i=1}^{N} P(o_1...o_t, x_{t+1} = j \mid x_t = i) P(x_t = i) P(o_{t+1} \mid x_{t+1} = j)$$

$$= \sum_{i=1...N} P(o_1...o_t, x_t = i) P(x_{t+1} = j \mid x_t = i) P(o_{t+1} \mid x_{t+1} = j)$$
2x Chain rule

$$=\sum_{i=1...N} \alpha_{i}(t)a_{ij}b_{jo_{t+1}}$$





$$= \sum_{i=1...N} P(o_1...o_t, x_t = i, x_{t+1} = j) P(o_{t+1} \mid x_{t+1} = j)$$

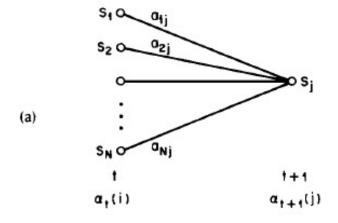
$$= \sum_{i=1...N} P(o_1...o_t, x_{t+1} = j \mid x_t = i) P(x_t = i) P(o_{t+1} \mid x_{t+1} = j)$$

$$= \sum_{i=1...N} P(o_1...o_t, x_t = i)P(x_{t+1} = j \mid x_t = i)P(o_{t+1} \mid x_{t+1} = j)$$

$$=\sum_{i=1,N}\alpha_i(t)a_{ij}b_{jo_{t+1}}$$

Definition + given parameters





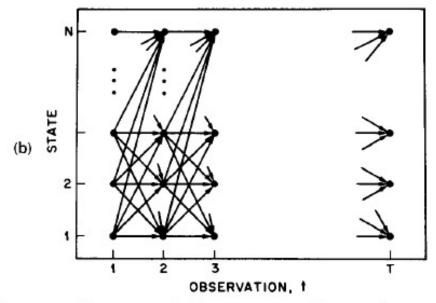


Fig. 4. (a) Illustration of the sequence of operations required for the computation of the forward variable $\alpha_{t+1}(j)$. (b) Implementation of the computation of $\alpha_t(i)$ in terms of a lattice of observations t, and states i.

Dynamic Programming

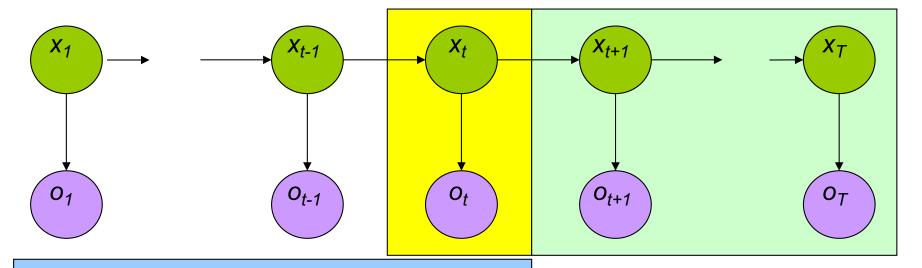
$$\alpha_{j}(t+1) = \sum_{i=1,N} \alpha_{i}(t) a_{ij} b_{jo_{t+1}}$$

Complexity $O(N^2 T)$ $E.g. N = 5, T = 100 \text{ gives } \approx 3000$



Similar, Backward Procedure





$$\beta_i(T) = 1$$

$$\beta_i(t) = P(o_{t+1}...o_T \mid x_t = i)$$

$$\beta_{i}(t) = \sum_{j=1...N} a_{ij} b_{io_{t+1}} \beta_{j}(t+1)$$

Probability of the rest of the states given the first state



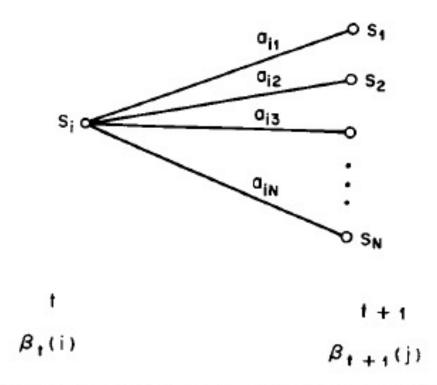
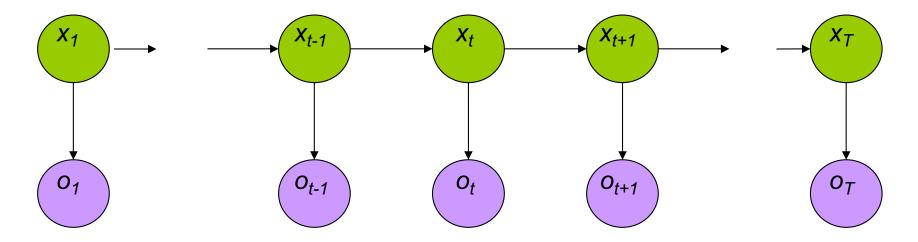


Fig. 5. Illustration of the sequence of operations required for the computation of the backward variable $\beta_t(i)$.

(1) Decoding: Solution





$$P(O \mid \mu) = \sum_{i=1}^{N} \alpha_i(T)$$

Forward Procedure

$$P(O | \mu) = \sum_{i=1}^{N} \pi_i \beta_i(1)$$

Backward Procedure

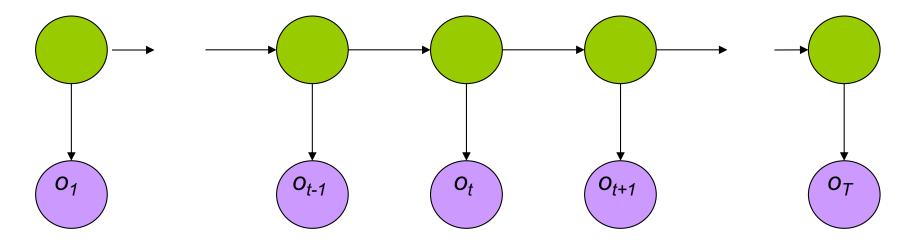
$$P(O \mid \mu) = \sum_{i=1}^{N} \alpha_i(t) \beta_i(t)$$

Combination

Important for (2) and (3)

(2) Best State Sequence





Find the state sequence that best explains the observations

$$arg \max_{X} P(X \mid O)$$

- Two approaches
 - Individually most likely states
 - Most likely sequence (Viterbi)



Best State Sequence (Variant A)

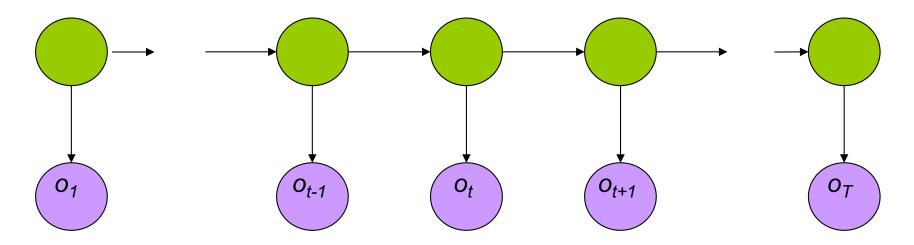
$$\begin{split} \gamma_i(t) &= P(X_t = i \,|\, O, \mu) & \text{target} \\ &= \frac{P(X_t = i, O \,|\, \mu)}{P(O \,|\, \mu)} & \text{definition of conditional prob} \\ &= \frac{\alpha_i(t).\beta_i(t)}{\sum\limits_{j=1}^n \alpha_j(t).\beta_j(t)} & \text{use what we know already} \end{split}$$

Most likely state at each point in time

$$\hat{X}_t = \arg\max \gamma_i(t)$$

Best State Sequence (Variant B)



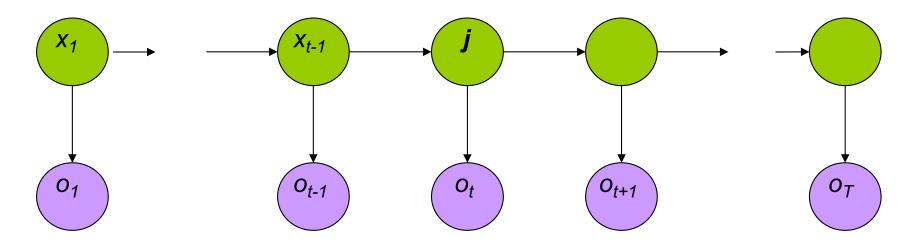


- Find the state sequence that best explains the observations, i.e., MPE
- Viterbi algorithm

$$arg \max_{X} P(X \mid O)$$

Viterbi Algorithm



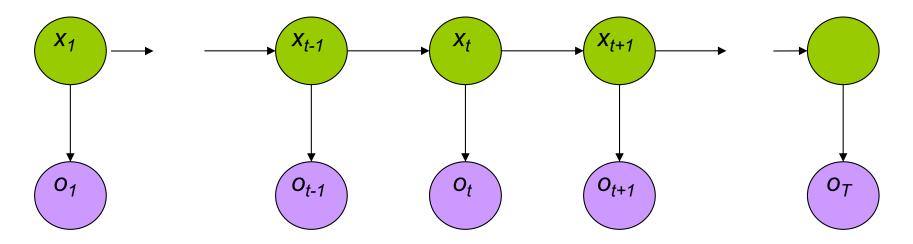


$$\delta_{j}(t) = \max_{x_{1}...x_{t-1}} P(x_{1}...x_{t-1}, o_{1}...o_{t-1}, x_{t} = j, o_{t})$$

The state sequence which maximizes the probability of seeing the observations to time t-1, landing in state j, and seeing the observation at time t

Viterbi Algorithm





$$\delta_{j}(t) = \max_{x_{1}...x_{t-1}} P(x_{1}...x_{t-1}, o_{1}...o_{t-1}, x_{t} = j, o_{t})$$

$$\delta_{j}(t+1) = \max_{i} \delta_{i}(t) a_{ij} b_{jo_{t+1}}$$

$$\psi_{j}(t+1) = \arg\max_{i} \delta_{i}(t) a_{ij} b_{jo_{t+1}}$$

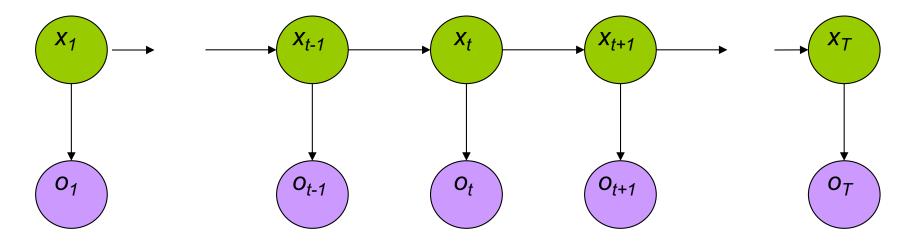
Initialization

$$\delta_1(i) = \pi_i b_{io_1}$$

$$\psi_1(i) = 0$$

Viterbi Algorithm





$$\hat{X}_{T} = \underset{i}{\operatorname{argmax}} \delta_{i}(T)$$

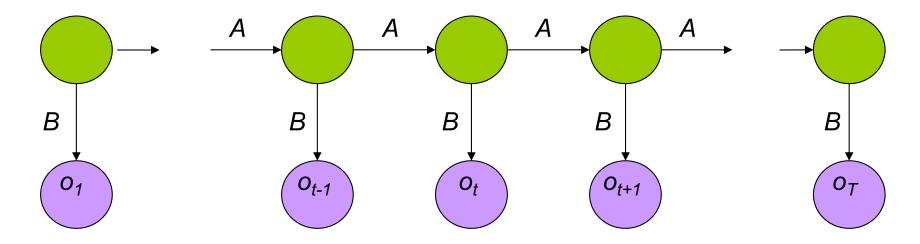
$$\hat{X}_{t} = \psi_{\hat{X}_{t+1}}(t+1)$$

$$P(\hat{X}) = \underset{i}{\operatorname{argmax}} \delta_{i}(T)$$

Compute the most likely state sequence by working backwards

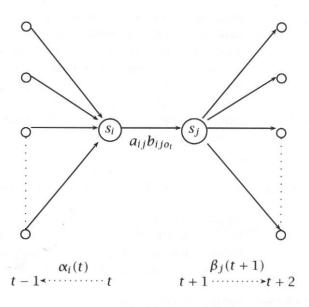
(3) Parameter Estimation using EM





- Given an observation sequence, find the model that is most likely to produce that sequence.
- No analytic method $\arg \max P(O_{training} \mid \mu)$
- Given a model and observation sequence, update the model parameters to better fit the observations.





$$\alpha_{i}(t) = P(o_{1}...o_{t}, x_{t} = i \mid \mu)$$
 $\beta_{i}(t) = P(o_{t+1}...o_{T} \mid x_{t} = i)$

$$P(O \mid \mu) = \sum_{i=1}^{N} \alpha_{i}(t)\beta_{i}(t)$$

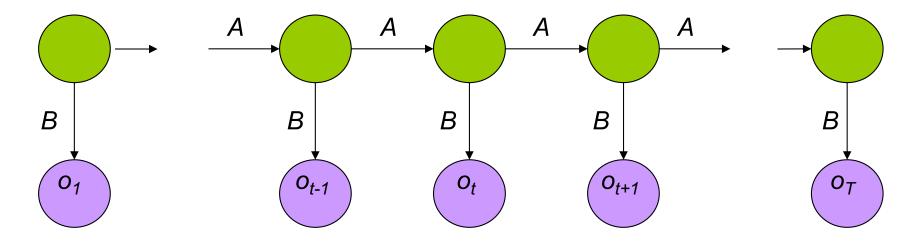
Figure 9.7 The probability of traversing an arc. Given an observation sequence and a model, we can work out the probability that the Markov process went from state s_i to s_j at time t.

Probability of making a transition from i to j at time t

$$\begin{split} p_t(i,j) &= P(X_t = i, X_{t+1} = j \mid O, \mu) \quad \text{target} \\ &= \frac{P(X_t = i, X_{t+1} = j, O \mid \mu)}{P(O \mid \mu)} \quad \text{def of cond prob} \end{split}$$

Parameter Estimation





$$p_{t}(i,j) = \frac{\alpha_{i}(t)a_{ij}b_{jo_{t+1}}\beta_{j}(t+1)}{\sum_{m=1...N}\alpha_{m}(t)\beta_{m}(t)}$$

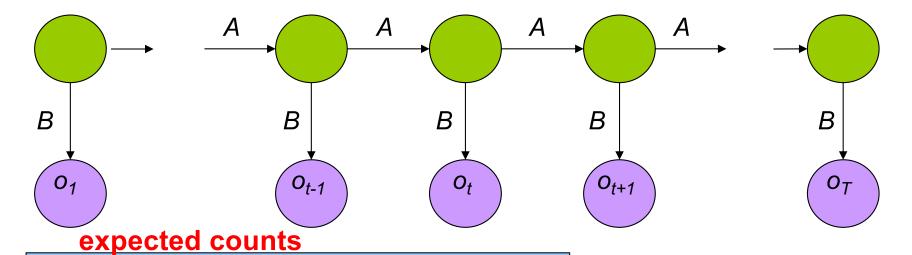
Probability of traversing an arc

$$\gamma_i(t) = \sum_{j=1...N} p_t(i,j)$$

Probability of being in state *i*

Parameter Estimation





$$\hat{\pi}_i = \gamma_i(1)$$

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T-1} p_t(i,j)}{\sum_{t=1}^{T-1} \gamma_i(t)}$$

$$\hat{b}_{ik} = \frac{\sum_{\{t:o_t = k\}} \gamma_i(t)}{\sum_{t=1}^{T} \gamma_i(t)}$$

Now we can compute the new estimates of the model parameters.



Instance of EM

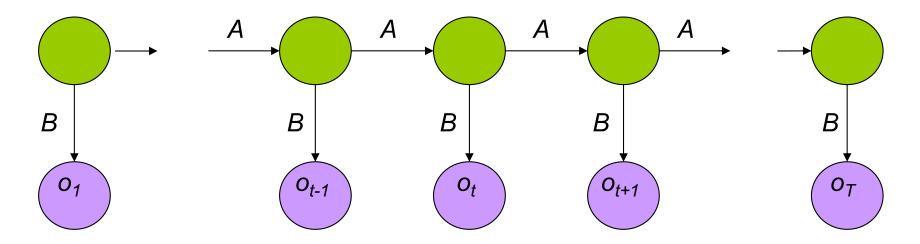
We have that

$$P(O \mid \hat{\mu}) \ge P(O \mid \mu)$$

- We may get stuck in local maximum (or even saddle point)
- Nevertheless, Baum-Welch usually effective

The Most Important Thing





We can use the special structure of this model to do a lot of neat math and solve problems that are otherwise not solvable.



Kalman Filers

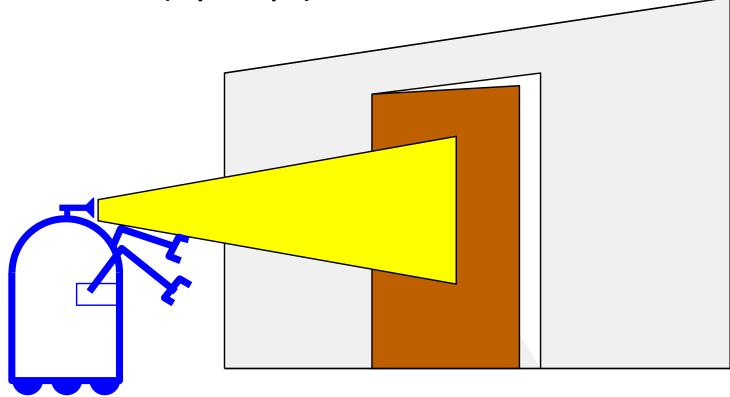
What about continuous states as e.g. encountered in robotics?

 The Kalman filter is a recursive filter that estimates the state of a linear dynamic system from a series of noisy measurements



Simple Example of State Estimation

- Suppose a robot obtains measurement z
- What is P(open | z)?







Example

•
$$P(z|open) = 0.6$$
 $P(z|\neg open) = 0.3$

• $P(open) = P(\neg open) = 0.5$

$$P(open \mid z) = \frac{P(z \mid open)P(open)}{P(z \mid open)p(open) + P(z \mid \neg open)p(\neg open)}$$

$$P(open \mid z) = \frac{0.6 \cdot 0.5}{0.6 \cdot 0.5 + 0.3 \cdot 0.5} = \frac{2}{3} = 0.67$$

Observing z raises the probability that the door is open.



Combining Evidence

- Suppose our robot obtains another observation z_2 .
- How can we integrate this new information?
- More generally, how can we estimate $P(x|z_1...z_n)$?



Recursive Bayesian Updating

$$P(x \mid z_1,...,z_n) = \frac{P(z_n \mid x,z_1,...,z_{n-1}) P(x \mid z_1,...,z_{n-1})}{P(z_n \mid z_1,...,z_{n-1})}$$

Markov assumption: z_n is independent of $z_1, ..., z_{n-1}$ if we know x

$$P(x \mid z_{1},...,z_{n}) = \frac{P(z_{n} \mid x) P(x \mid z_{1},...,z_{n-1})}{P(z_{n} \mid z_{1},...,z_{n-1})}$$

$$= \eta P(z_{n} \mid x) P(x \mid z_{1},...,z_{n-1})$$

$$= \eta_{1...n} \prod_{i=1...n} P(z_{i} \mid x) P(x)$$



Example: Second Measurement



•
$$P(z_2|open) = 0.5$$
 $P(z_2|\neg open) = 0.6$

$$P(z_2|\neg open) = 0.6$$

• $P(open|z_1)=2/3$

$$P(open | z_2, z_1) = \frac{P(z_2 | open) P(open | z_1)}{P(z_2 | open) P(open | z_1) + P(z_2 | \neg open) P(\neg open | z_1)}$$

$$= \frac{\frac{1}{2} \cdot \frac{2}{3}}{\frac{1}{2} \cdot \frac{2}{3} + \frac{3}{5} \cdot \frac{1}{3}} = \frac{5}{8} = 0.625$$

z_2 lowers the probability that the door is open.



Actions

- As you know already, often the world is dynamic since
 - actions carried out by the robot,
 - actions carried out by other agents,
 - or just the time is passing by change the world.
- How can we incorporate such actions?



Typical Actions for a Robot

- The robot turns its wheels to move
- The robot uses its manipulator to grasp an object
- Plants grow over time...

- Actions are never carried out with absolute certainty.
- In contrast to measurements, actions generally increase the uncertainty.





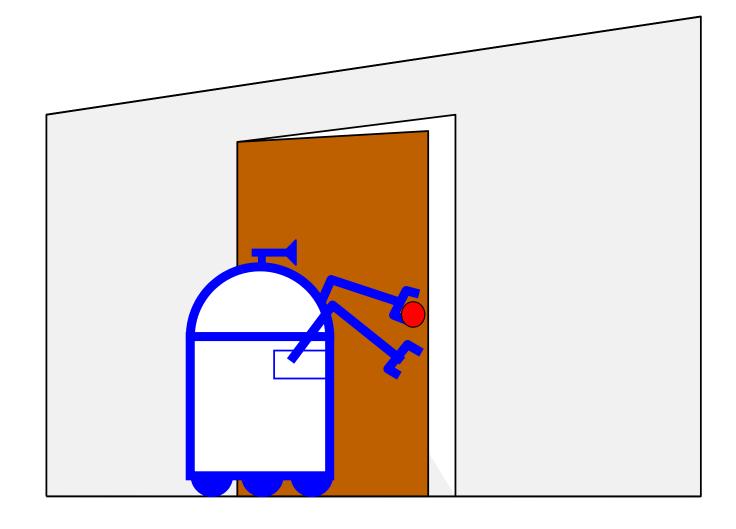
Modeling Action

 To incorporate the outcome of an action u into the current "belief", we use the conditional pdf

This term specifies the pdf that executing u changes the state from x' to x.



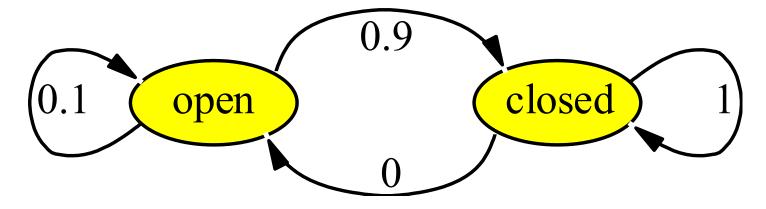
Example: Closing the door



State Transitions



P(x|u,x') for u = "close door":



If the door is open, the action "close door" succeeds in 90% of all cases.

Integrating the Outcome of Actions



Continuous case:

$$P(x \mid u) = \int P(x \mid u, x') P(x') dx'$$

Discrete case:

$$P(x | u) = \sum_{x'} P(x | u, x') P(x')$$

Example: The Resulting Belief



$$P(closed | u) = \sum P(closed | u, x')P(x')$$

$$= P(closed | u, open)P(open)$$

$$+ P(closed | u, closed)P(closed)$$

$$= \frac{9}{10} * \frac{5}{8} + \frac{1}{1} * \frac{3}{8} = \frac{15}{16}$$

$$P(open | u) = \sum P(open | u, x')P(x')$$

$$= P(open | u, open)P(open)$$

$$+ P(open | u, closed)P(closed)$$

$$= \frac{1}{10} * \frac{5}{8} + \frac{0}{1} * \frac{3}{8} = \frac{1}{16}$$

$$= 1 - P(closed | u)$$



Bayes Filters: Framework

Given:

Stream of observations z and action data u:

$$d_t = \{u_1, z_2, ..., u_{t-1}, z_t\}$$

- Sensor model P(z|x).
- Action model P(x|u,x').
- Prior probability of the system state P(x).

Wanted:

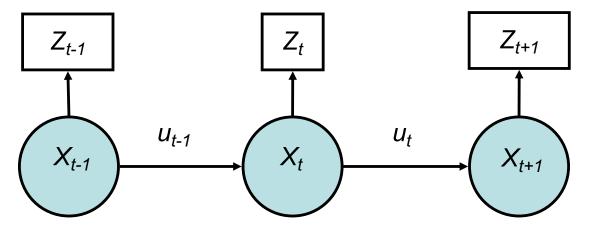
- Estimate of the state X of a dynamical system.
- The posterior of the state is also called Belief:

$$Bel(x_t) = P(x_t | u_1, z_2 ..., u_{t-1}, z_t)$$



Markov Assumption





$$p(d_{t}, d_{t-1}, ..., d_{0} | x_{t}, d_{t+1}, d_{t+2}, ...) = p(d_{t}, d_{t-1}, ..., d_{0} | x_{t})$$

$$p(d_{t}, d_{t+1}, ... | x_{t}, d_{1}, d_{2}, ..., d_{t-1}) = p(d_{t}, d_{t+1}, ... | x_{t})$$

$$p(x_{t} | u_{t-1}, x_{t-1}, d_{t-2}, ..., d_{0}) = p(x_{t} | u_{t-1}, x_{t-1})$$

Underlying Assumptions

Static world; Independent noise; Perfect model, no approximation errors



u = action

x = state

$$\begin{split} & \underbrace{Bel(x_t)} = P(x_t \mid u_1, z_2 \, ..., u_{t-1}, z_t) \\ & \text{Bayes} & = \eta \, P(z_t \mid x_t, u_1, z_2, ..., u_{t-1}) \, P(x_t \mid u_1, z_2, ..., u_{t-1}) \\ & \text{Markov} & = \eta \, P(z_t \mid x_t) \, P(x_t \mid u_1, z_2, ..., u_{t-1}) \\ & \text{Total prob.} & = \eta \, P(z_t \mid x_t) \int P(x_t \mid u_1, z_2, ..., u_{t-1}, x_{t-1}) \\ & \qquad \qquad P(x_{t-1} \mid u_1, z_2, ..., u_{t-1}) \, dx_{t-1} \\ & \text{Markov} & = \eta \, P(z_t \mid x_t) \int P(x_t \mid u_{t-1}, x_{t-1}) \, P(x_{t-1} \mid u_1, z_2, ..., u_{t-1}) \, dx_{t-1} \\ & = \eta \, P(z_t \mid x_t) \int P(x_t \mid u_{t-1}, x_{t-1}) \, Bel(x_{t-1}) \, dx_{t-1} \end{split}$$

$Bel(x_t) = \eta P(z_t | x_t) \int P(x_t | u_{t-1}, x_{t-1}) Bel(x_{t-1}) dx_{t-1}$

HNISCHE IVERSITÄT RMSTADT

- 1. Algorithm **Bayes_filter**(*Bel(x),d*):
- $\eta=0$
- 3. if d is a perceptual data item z then
- 4. For all x do
- 5. $Bel'(x) = P(z \mid x)Bel(x)$
- 6. $\eta = \eta + Bel'(x)$
- 7. For all *x* do
- 8. $Bel'(x) = \eta^{-1}Bel'(x)$
- 9. else if *d* is an action data item *u* then
- 10. For all *x* do
- 11. $Bel'(x) = \int P(x \mid u, x') Bel(x') dx'$
- 12. return Bel'(x)



Bayes Filters are Familiar!

$$Bel(x_t) = \eta \ P(z_t \mid x_t) \int P(x_t \mid u_{t-1}, x_{t-1}) \ Bel(x_{t-1}) \ dx_{t-1}$$

- Kalman filters
- Particle filters
- Hidden Markov models
- Dynamic Bayes networks (later this class)
- Partially Observable Markov Decision Processes (POMDPs; maybe we touch them later this class)





Gaussian

$$p(x) \sim N(\mu, \sigma^2)$$
:

$$p(x) \sim N(\mu, \sigma^2):$$

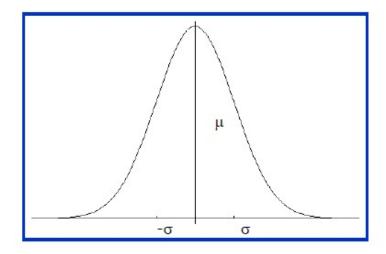
$$p(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{1(x-\mu)^2}{2\sigma^2}}$$

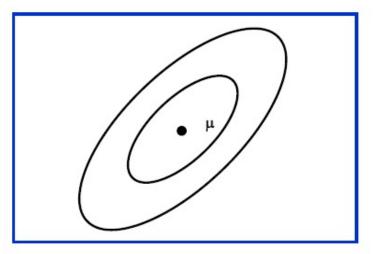
Univariate

$$p(\mathbf{x}) \sim N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$$
:

$$p(\mathbf{x}) = \frac{1}{(2\pi)^{d/2} |\mathbf{\Sigma}|^{1/2}} e^{\frac{1}{2} (\mathbf{x} - \mathbf{\mu})^t \mathbf{\Sigma}^{-1} (\mathbf{x} - \mathbf{\mu})}$$

Multivariate







Propertis of Gaussians

$$\left. \begin{array}{l} X \sim N(\mu, \Sigma) \\ Y = AX + B \end{array} \right\} \quad \Rightarrow \quad Y \sim N(A\mu + B, A\Sigma A^T)$$

$$\frac{X_1 \sim N(\mu_1, \Sigma_1)}{X_2 \sim N(\mu_2, \Sigma_2)} \Rightarrow p(X_1) \cdot p(X_2) \sim N \left(\frac{\Sigma_2}{\Sigma_1 + \Sigma_2} \mu_1 + \frac{\Sigma_1}{\Sigma_1 + \Sigma_2} \mu_2, \frac{1}{\Sigma_1^{-1} + \Sigma_2^{-1}} \right)$$

 We stay in the "Gaussian world" as long as we start with Gaussians and perform only linear transformations.





Discrete Time Kalman Filter

Estimates the state x of a discrete-time controlled process that is governed by the linear stochastic difference equation

$$x_t = A_t x_{t-1} + B_t u_t + \varepsilon_t$$

with a measurement

$$z_t = C_t x_t + \delta_t$$



Components of a Kalman Filter

- Matrix (nxn) that describes how the state evolves from t to t-1 without controls or noise.
- B_t Matrix (nxl) that describes how the control u_t changes the state from t to t-1.
- C_t Matrix (kxn) that describes how to map the state x_t to an observation z_t .
- Random variables representing the process and measurement noise that are assumed to be independent and normally distributed with covariance R_t and Q_t respectively.

Initialization



Initial belief is normally distributed:

$$bel(x_0) = N(x_0; \mu_0, \Sigma_0)$$



Dynamics

 Dynamics are linear function of state and control plus additive noise:

$$x_t = A_t x_{t-1} + B_t u_t + \varepsilon_t$$

$$p(x_{t} | u_{t}, x_{t-1}) = N(x_{t}; A_{t}x_{t-1} + B_{t}u_{t}, R_{t})$$

$$\overline{bel}(x_t) = \int p(x_t \mid u_t, x_{t-1}) \qquad bel(x_{t-1}) dx_{t-1}$$

$$\downarrow \qquad \qquad \downarrow \qquad \qquad \downarrow$$

$$\sim N(x_t; A_t x_{t-1} + B_t u_t, R_t) \sim N(x_{t-1}; \mu_{t-1}, \Sigma_{t-1})$$



Belief Update: Prediction

$$\overline{bel}(x_{t}) = \int p(x_{t} \mid u_{t}, x_{t-1}) \qquad bel(x_{t-1}) dx_{t-1}$$

$$\downarrow \qquad \qquad \downarrow \qquad \qquad \downarrow \qquad \qquad \downarrow \qquad \qquad \downarrow \qquad \qquad \qquad N(x_{t}; A_{t}x_{t-1} + B_{t}u_{t}, R_{t}) \sim N(x_{t-1}; \mu_{t-1}, \Sigma_{t-1})$$

$$\downarrow \qquad \qquad \downarrow \qquad \qquad$$



Observations

 Observations are linear function of state plus additive noise:

$$z_t = C_t x_t + \delta_t$$

$$p(z_t \mid x_t) = N(z_t; C_t x_t, Q_t)$$

$$bel(x_t) = \eta \quad p(z_t \mid x_t) \qquad \overline{bel}(x_t)$$

$$\downarrow \qquad \qquad \downarrow$$

$$\sim N(z_t; C_t x_t, Q_t) \qquad \sim N(x_t; \overline{\mu}_t, \overline{\Sigma}_t)$$



Belief Update: Correction

$$bel(x_t) = \eta \quad p(z_t \mid x_t) \qquad \overline{bel}(x_t)$$

$$\downarrow \qquad \qquad \downarrow \qquad \qquad \downarrow$$

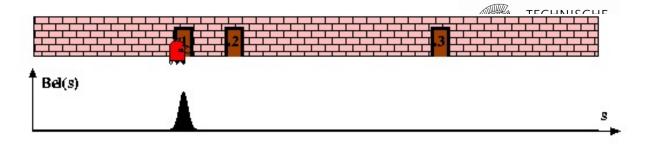
$$\sim N(z_t; C_t x_t, Q_t) \qquad \sim N(x_t; \overline{\mu}_t, \overline{\Sigma}_t)$$

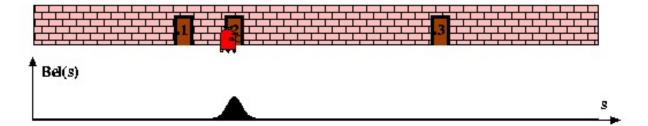
$$\downarrow \qquad \qquad \downarrow$$

$$bel(x_t) = \eta \exp\left\{-\frac{1}{2}(z_t - C_t x_t)^T Q_t^{-1}(z_t - C_t x_t)\right\} \exp\left\{-\frac{1}{2}(x_t - \overline{\mu}_t)^T \overline{\Sigma}_t^{-1}(x_t - \overline{\mu}_t)\right\}$$

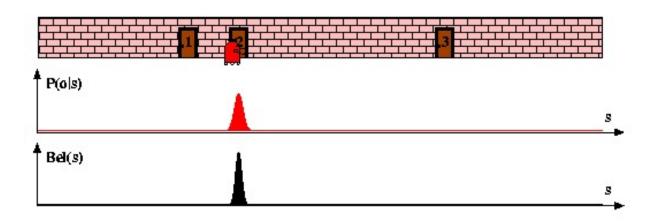
$$bel(x_t) = \begin{cases} \mu_t = \overline{\mu}_t + K_t(z_t - C_t \overline{\mu}_t) \\ \Sigma_t = (I - K, C_t) \overline{\Sigma}_t \end{cases} \quad \text{with} \quad K_t = \overline{\Sigma}_t C_t^T (C_t \overline{\Sigma}_t C_t^T + Q_t)^{-1}$$

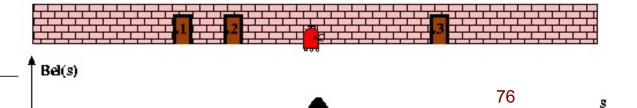
Kalman Filters





We stopped here







So far

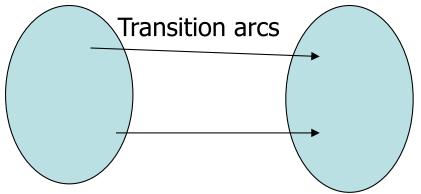
- Sometimes, belief states admit compact representations & manipulations:
 - E.g. Kalman filters (assuming Gaussian processes)

What about Dynamic Bayesian Networks, i.e., the general case?



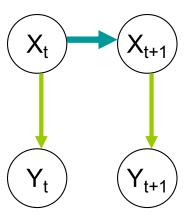


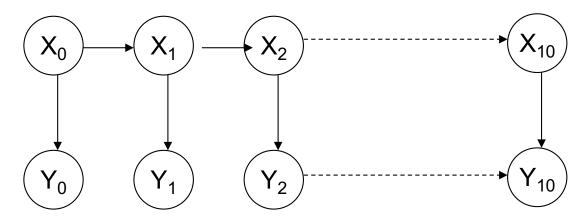
Dynamic Bayesian Networks (DBNs)



Arbitrary Bayesian Network at time t

The same Bayesian Network at time t+1





Unrolled DBN for t=0 to t=10



Types of Inference Tasks

Filtering: $\mathbf{P}(\mathbf{X}_t|\mathbf{e}_{1:t})$ belief state—input to the decision process of a rational agent

Prediction: $P(X_{t+k}|e_{1:t})$ for k > 0 evaluation of possible action sequences; like filtering without the evidence

Smoothing: $P(X_k|e_{1:t})$ for $0 \le k < t$ better estimate of past states, essential for learning

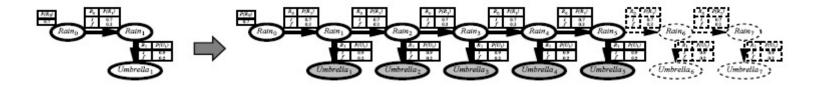
Most likely explanation: $\arg \max_{\mathbf{x}_{1:t}} P(\mathbf{x}_{1:t}|\mathbf{e}_{1:t})$ speech recognition, decoding with a noisy channel

filtering P(X(t) | y(1:t)) prediction P(X(t+H) | y(1:t)) fixed-lag smoothing p(X(t-L) | y(1:t)) fixed interval smoothing P(X(t) | y(1:t))



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

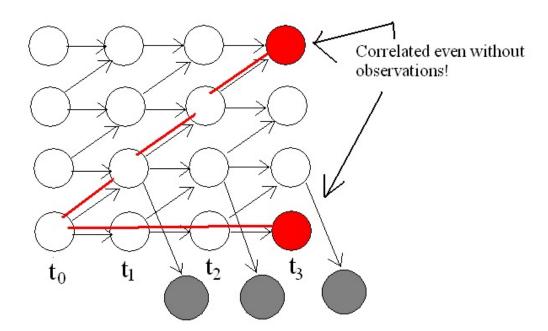
Largest factor is $O(d^{n+1})$, update cost $O(d^{n+2})$ (cf. HMM update cost $O(d^{2n})$)



Problem



- Bayesian Network: a decomposed structure to represent the full joint distribution and to do efficient inference
- Does it imply easy decomposition for the belief state?



No!





Tractable, approximate representation

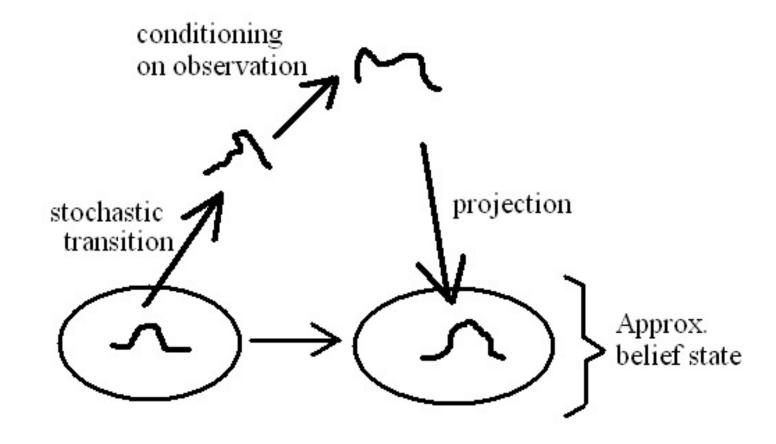
- Exact inference in DBN is intractable
- Need approximation
 - Maintain an approximate belief state
- E.g. Factored belief state



Assumed Density Filtering: Idea



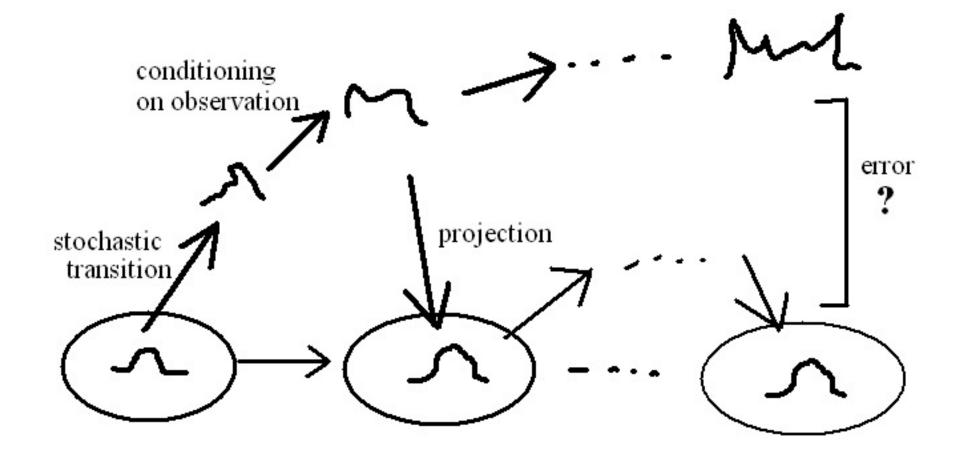
Use a decomposable representation for the belief state (pre-assume some independency)



Is there a Problem?



- What about the approximation errors?
 - It might accumulate and grow unbounded...





Contraction property

- Boyen and Koller have proven that
 - Under reasonable assumptions about the stochasticity of the process, every state transition results in a contraction of the distance between the two distributions by a constant factor
 - Since approximation errors from previous steps decrease exponentially, the overall error remains bounded indefinitely

