4. Probabilistic Graphical Models

Directed Models

The Bayes Filter (Rep.)

$$\boxed{\text{Bel}(x_t)} = p(x_t \mid u_1, z_1, \ldots, u_t, z_t)$$

(Bayes)
$$= \eta\, p(z_t \mid x_t, u_1, z_1, \ldots, u_t) p(x_t \mid u_1, z_1, \ldots, u_t)$$

(Markov)
$$= \eta\, p(z_t \mid x_t) p(x_t \mid u_1, z_1, \ldots, u_t)$$

(Tot. prob.)
$$= \eta\, p(z_t \mid x_t) \int p(x_t \mid u_1, z_1, \ldots, u_t, x_{t-1}) p(x_{t-1} \mid u_1, z_1, \ldots, u_t) dx_{t-1}$$

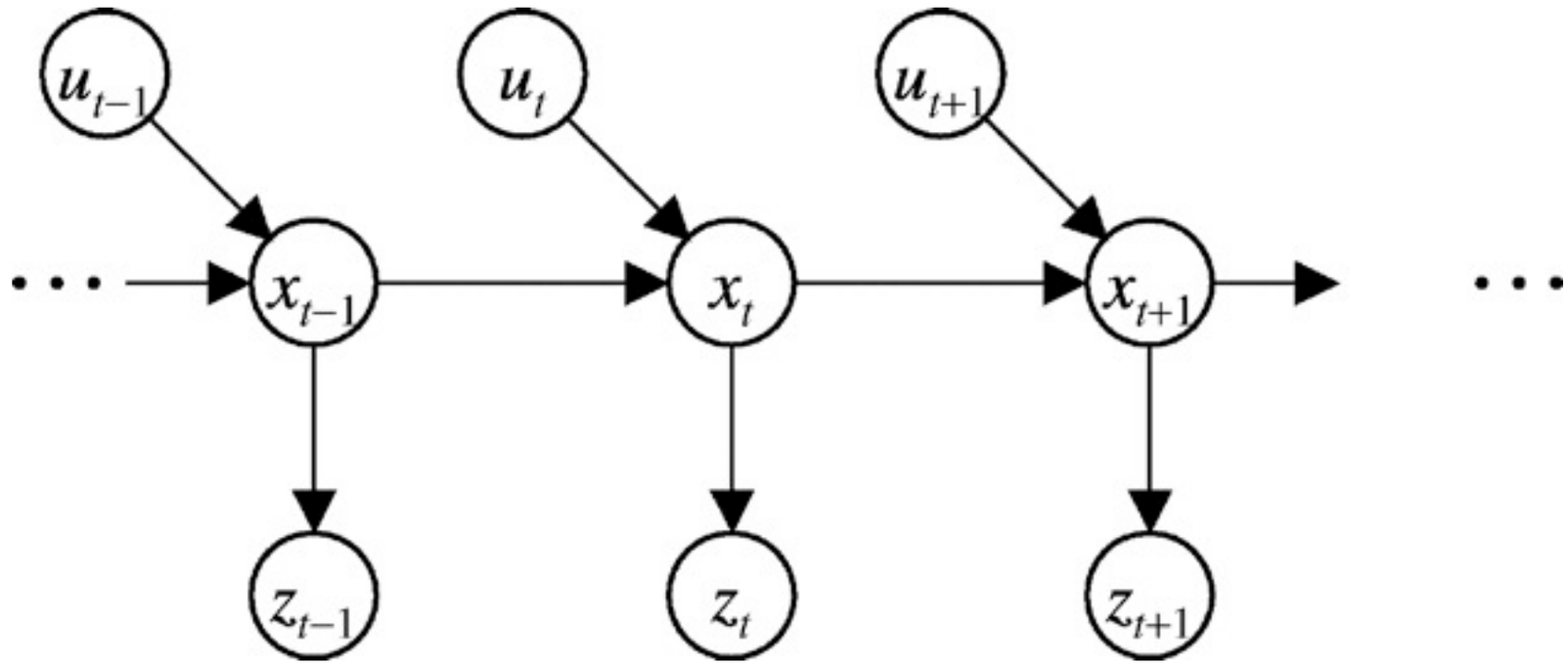
(Markov)
$$= \eta\, p(z_t \mid x_t) \int p(x_t \mid u_t, x_{t-1}) p(x_{t-1} \mid u_1, z_1, \ldots, u_t) dx_{t-1}$$

(Markov)
$$= \eta\, p(z_t \mid x_t) \int p(x_t \mid u_t, x_{t-1}) p(x_{t-1} \mid u_1, z_1, \ldots, z_{t-1}) dx_{t-1}$$

$$\boxed{= \eta\, p(z_t \mid x_t) \int p(x_t \mid u_t, x_{t-1}) \text{Bel}(x_{t-1}) dx_{t-1}}$$

Graphical Representation (Rep.)

We can describe the overall process using a ***Dynamic Bayes Network***:



- This incorporates the following Markov assumptions:

$$p(z_t \mid x_{0:t}, u_{1:t}, z_{1:t}) = p(z_t \mid x_t) \quad \text{(measurement)}$$

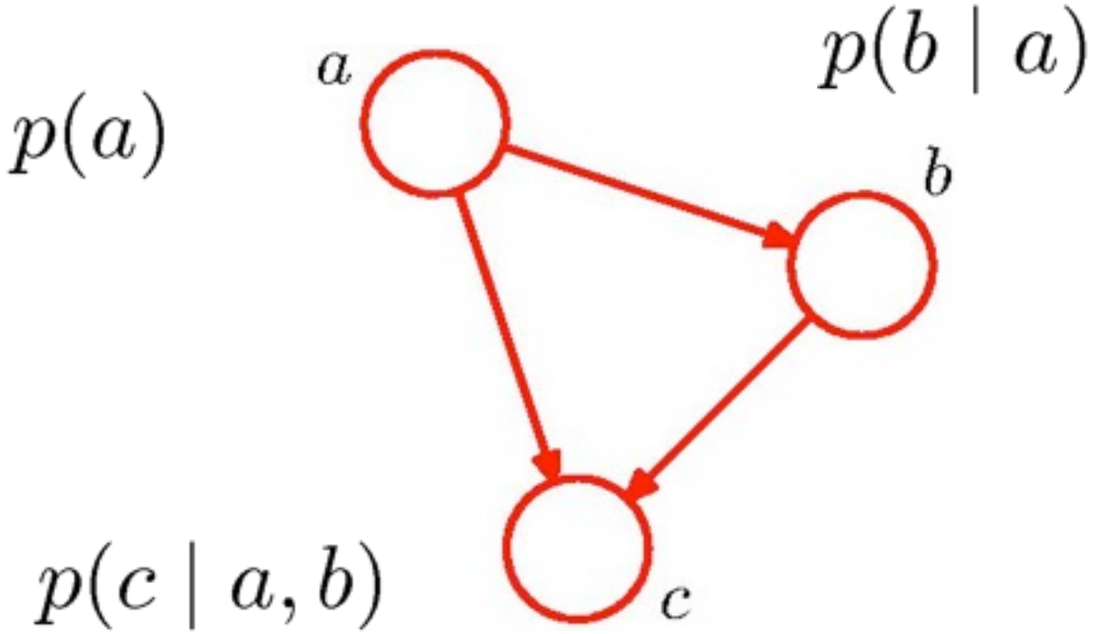
$$p(x_t \mid x_{0:t-1}, u_{1:t}, z_{1:t}) = p(x_t \mid x_{t-1}, u_t) \quad \text{(state)}$$

Definition

- A Probabilistic Graphical Model is a diagrammatic representation of a probability distribution.
- In a Graphical Model, random variables are represented as nodes, and statistical dependencies are represented using edges between the nodes.
- The resulting graph can have the following properties:
- Cyclic / acyclic
- Directed / undirected
- The simplest graphs are Directed Acyclig Graphs (DAG).

Simple Example

- Given: 3 random variables a , b , and c
- Joint prob: $p(a, b, c) = p(c|a, b)p(a, b) = p(c|a, b)p(b|a)p(a)$



Random variables can be discrete or continuous

- A Graphical Model based on a DAG is called a Bayesian Network

Simple Example

- In general: K random variables $x_1, x_2, \ldots, x_K$
- Joint prob:

$$p(x_1, \ldots, x_K) = p(x_K|x_1, \ldots, x_{K-1}) \ldots p(x_2|x_1)p(x_1)$$

- This leads to a fully connected graph.
- Note: The ordering of the nodes in such a fully connected graph is arbitrary. They all represent the joint probability distribution:

$$p(a, b, c) = p(a|b, c)p(b|c)p(c)$$

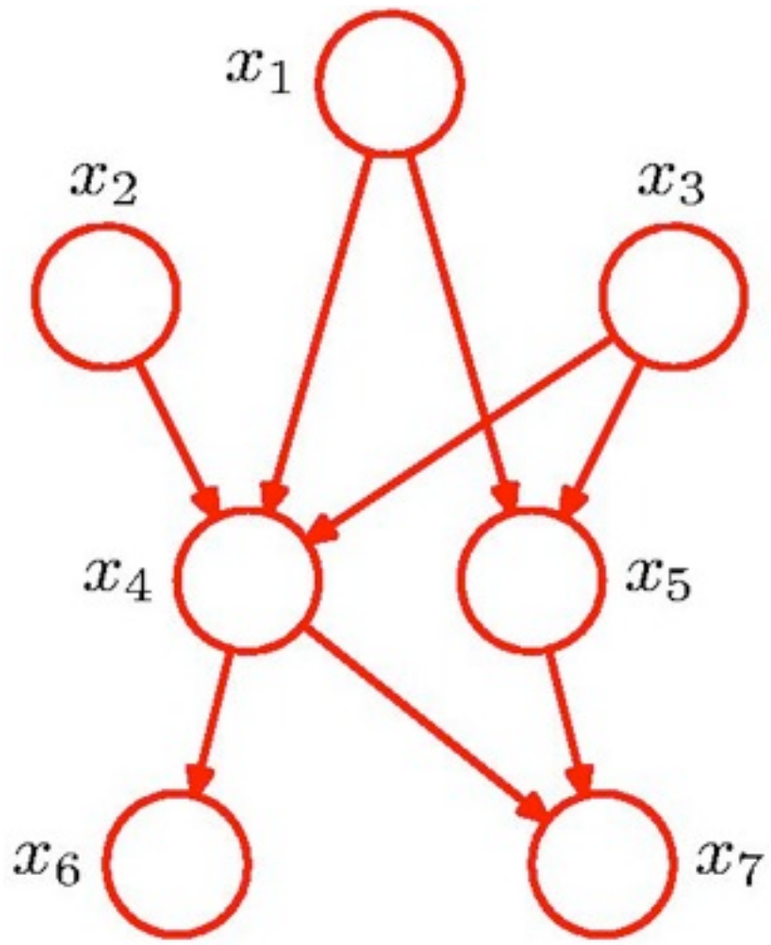
$$p(a, b, c) = p(b|a, c)p(a|c)p(c)$$

$$\vdots$$

Bayesian Networks

- Statistical independence can be represented by the absence of edges. This makes the computation efficient.

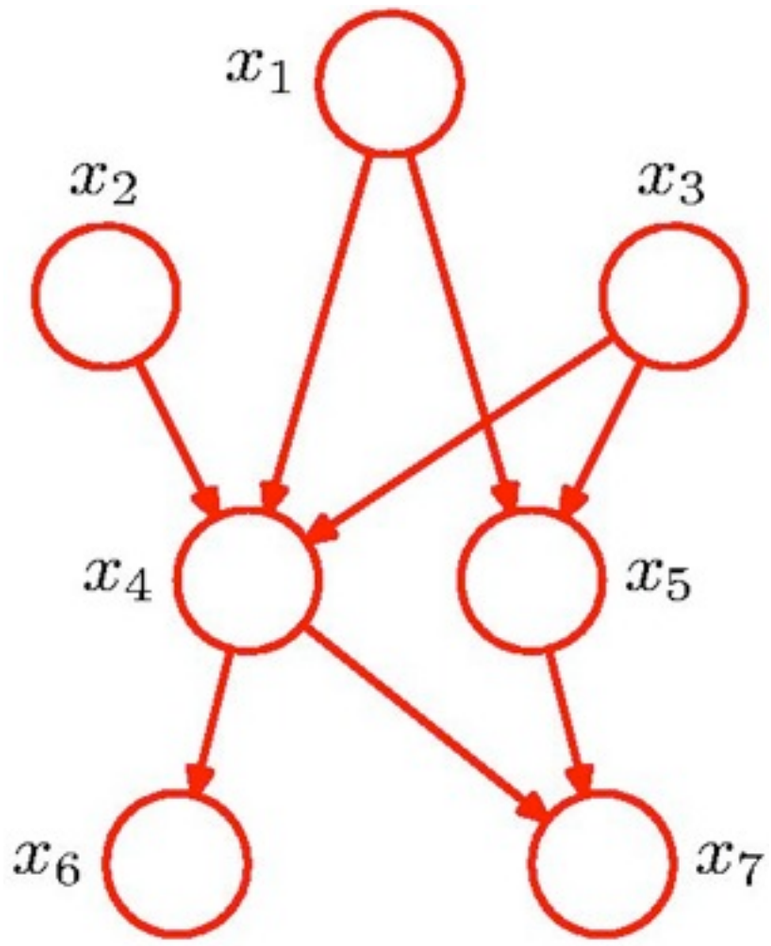
$$p(x_1, \ldots, x_7) = p(x_1)p(x_2)p(x_3)p(x_4|x_1, x_2, x_3) \\ p(x_5|x_1, x_3)p(x_6|x_4)p(x_7|x_4, x_5)$$

- 

-
- Intuitively: only x_1 and x_3 have an influence on x_5

Bayesian Networks

- We can now define a one-to-one mapping from graphical models to probabilistic formulations:



General Factorization

$$p(\mathbf{x}) = \prod_{k=1}^{K} p(x_k|\mathrm{pa}_k)$$

where

$pa_k \triangleq$ ancestors of p_k

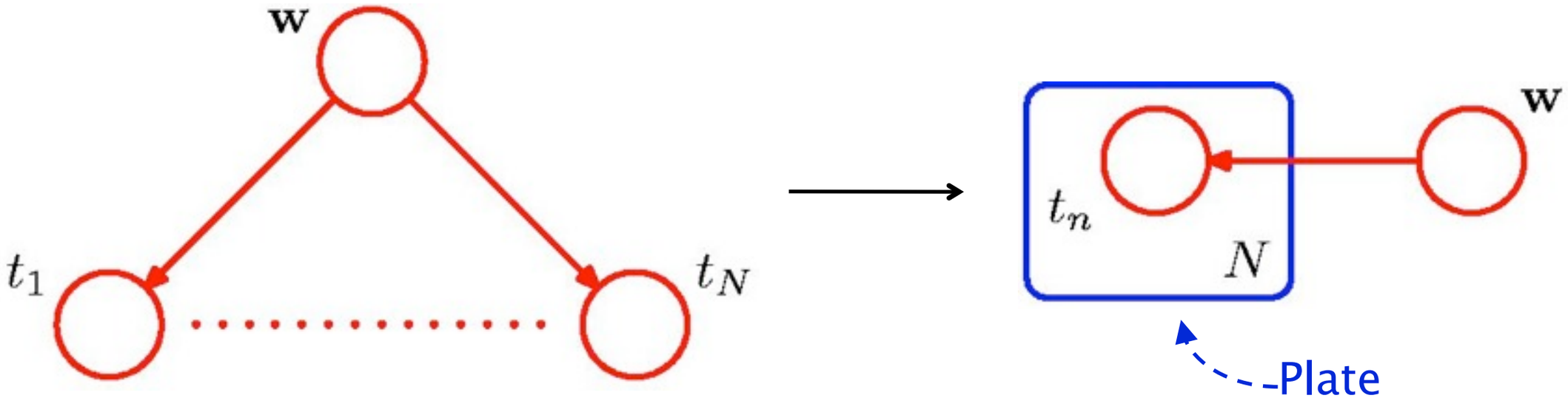
and

$$p(\mathbf{x}) = p(x_1, \ldots, x_K)$$

Elements of Graphical Models

- In case of a series of random variables with equal dependencies, we can subsume them using a plate:

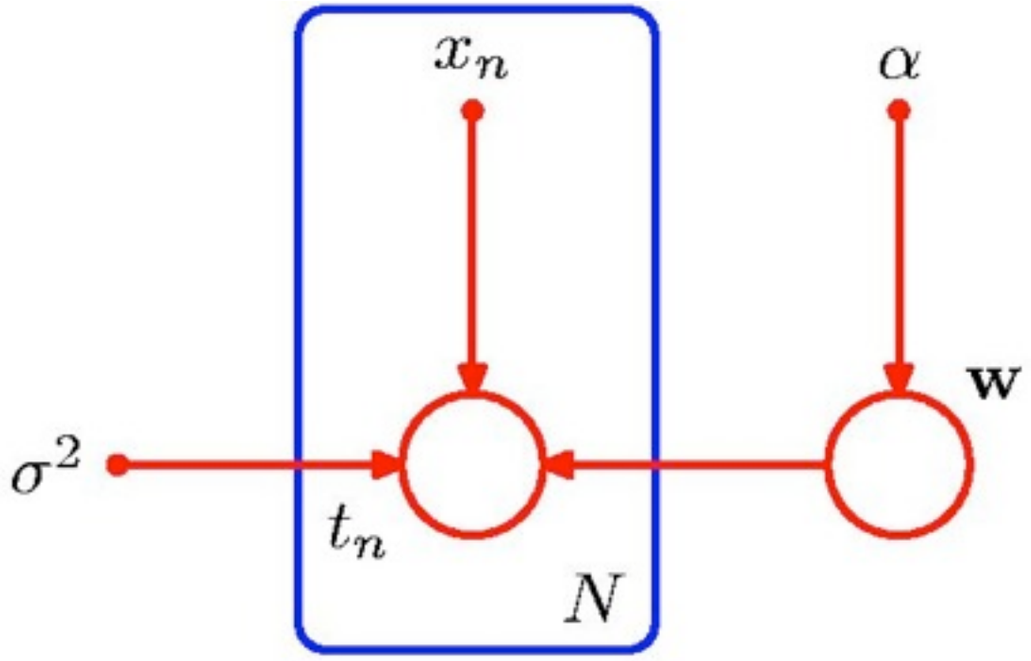
$$p(\mathbf{t}, \mathbf{w}) = p(\mathbf{w}) \prod_{n=1}^{N} p(t_n|\mathbf{w})$$



Elements of Graphical Models (2)

- We distinguish between input variables and explicit hyperparameters:

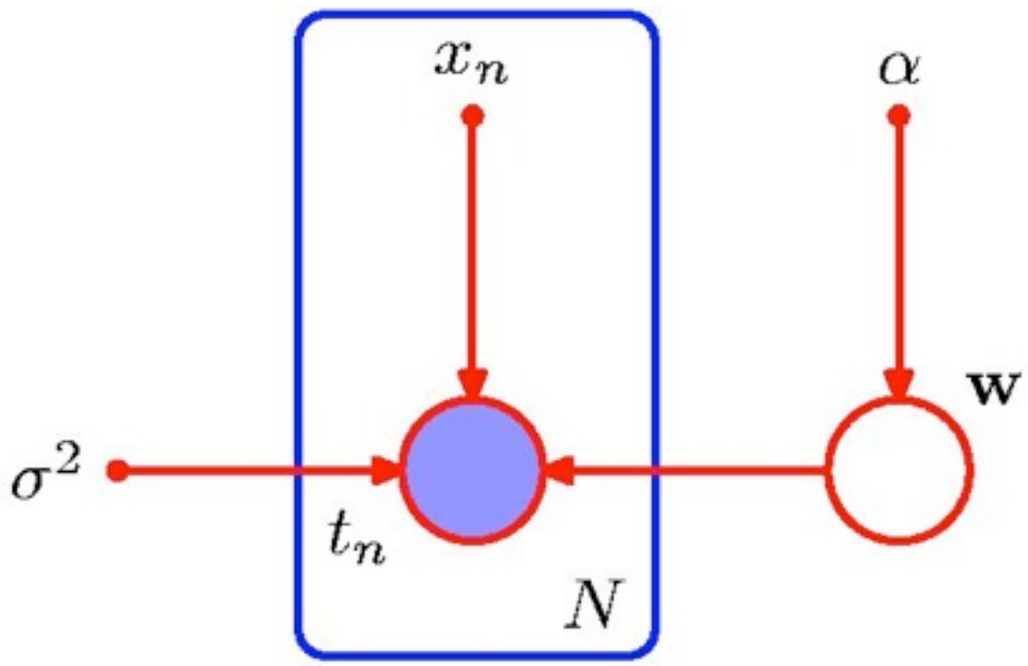
$$p(\mathbf{t}, \mathbf{w}|\mathbf{x}, \alpha, \sigma^2) = p(\mathbf{w}|\alpha) \prod_{n=1}^{N} p(t_n|\mathbf{w}, x_n, \sigma^2).$$



Elements of Graphical Models (3)

- We distinguish between observed variables and hidden variables:
- (deterministic parameters omitted)

$$p(\mathbf{w}|\mathbf{t}) \propto p(\mathbf{w}) \prod_{n=1}^{N} p(t_n|\mathbf{w})$$

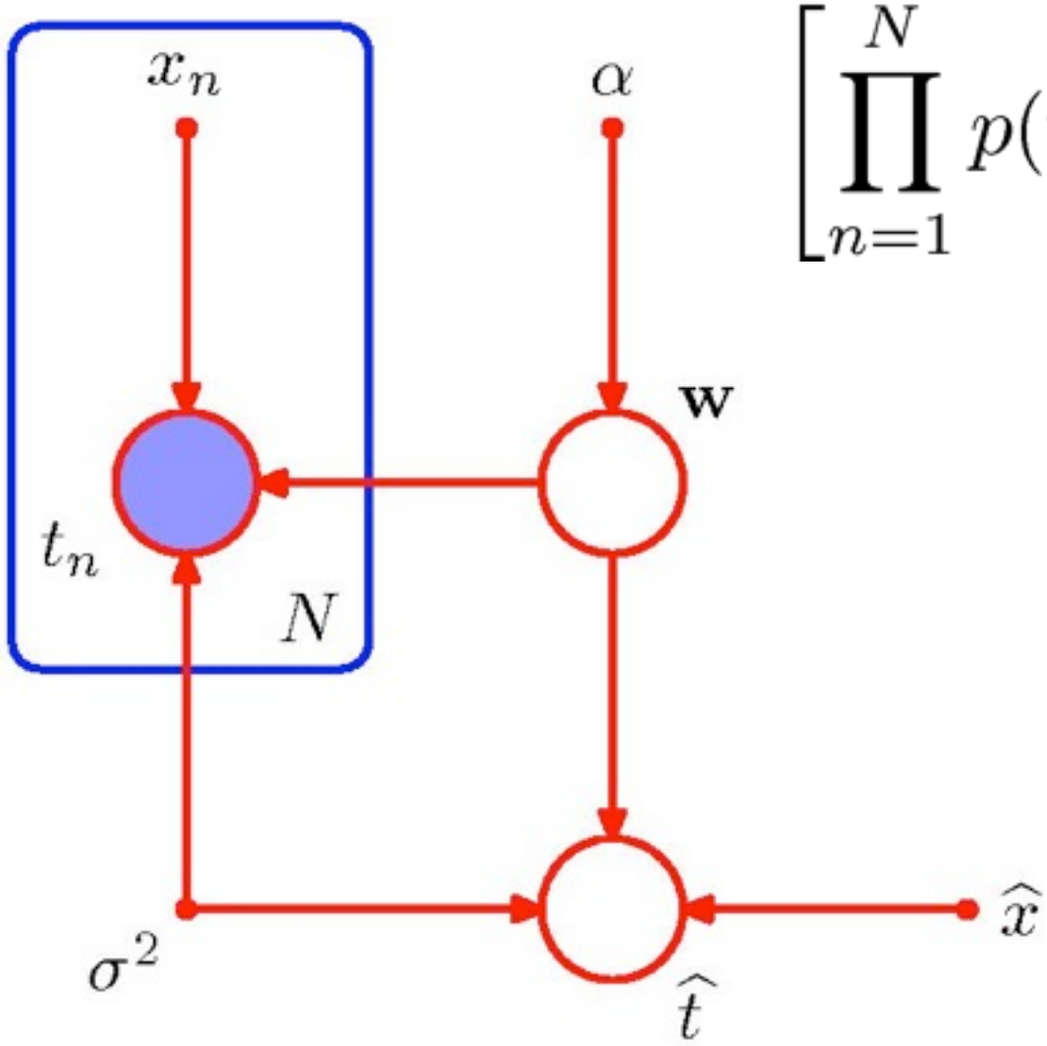


Regression as a Graphical Model

Regression: Prediction of a new target value $\hat{t}$

$$p(\hat{t}, \mathbf{t}, \mathbf{w} \mid \hat{x}, \mathbf{x}, \alpha, \sigma^2) =$$

$$\left[\prod_{n=1}^{N} p(t_n \mid x_n, \mathbf{w}, \sigma^2)\right] p(\mathbf{w} \mid \alpha) p(\hat{t} | \hat{x}, \mathbf{w}, \sigma^2)$$



Here: conditioning on all deterministic parameters

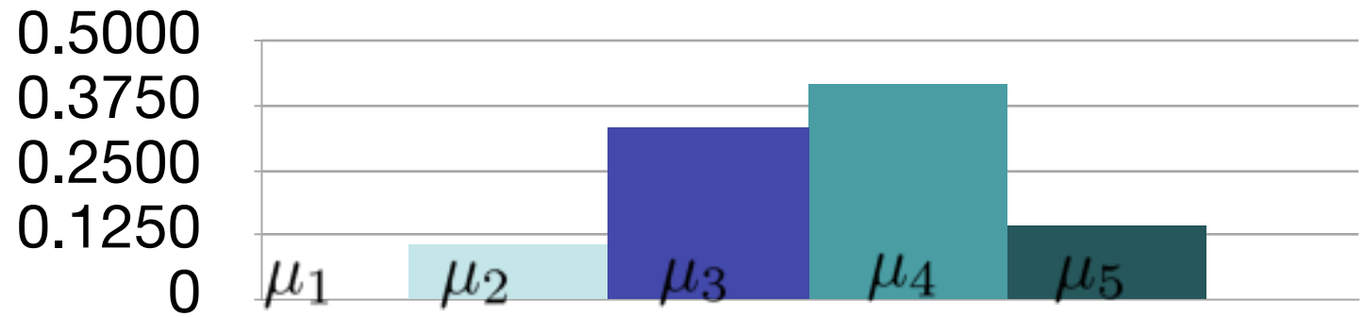
Using this, we can obtain the **Predictive distribution:**

$$p(\widehat{t} | \widehat{x}, \mathbf{x}, \mathbf{t}, \alpha, \sigma^2) \propto \int p(\widehat{t}, \mathbf{t}, \mathbf{w} | \widehat{x}, \mathbf{x}, \alpha, \sigma^2) \, \mathrm{d}\mathbf{w}$$

Two Special Cases

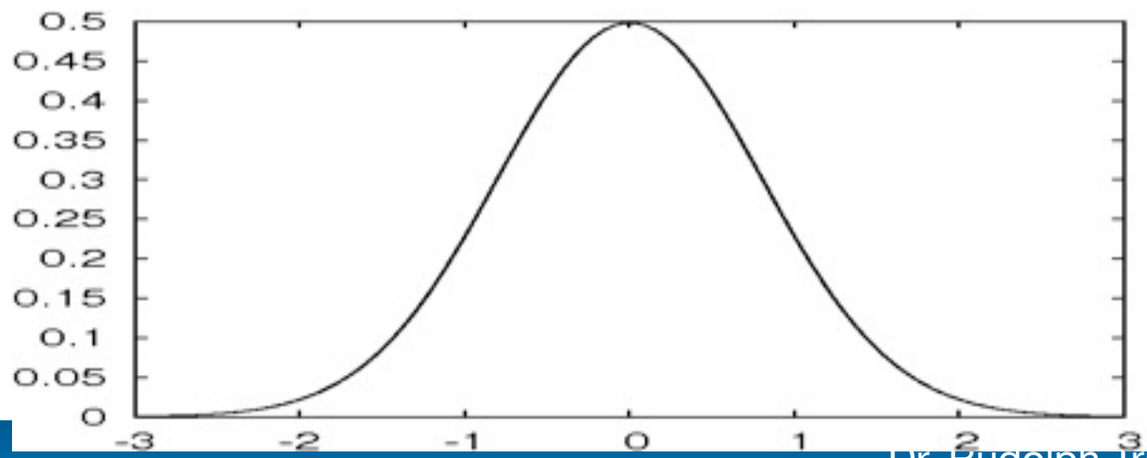
- We consider two special cases:
- All random variables are discrete; i.e. Each x_i is represented by values $\mu_1, \ldots, \mu_K$ where

$$\sum_{j=1}^{K} \mu_j = 1$$



- All random variables are Gaussian

$$x_i \sim \mathcal{N}(.; \mu_i, \sigma_i^2)$$



Discrete Variables: Example

- Two dependent variables: $K^2 - 1$ parameters

Here: K = 2

x_1	$p(x_1)$
1	0.2
2	0.8

$K - 1$

x_1	x_2	$p(x_2 \mid x_1)$
1	1	0.25
1	2	0.75
2	1	0.1
2	2	0.9

$K - 1$, $K - 1$, $K(K-1)$

$$K - 1 + K(K-1) = K^2 - 1$$

- Independent joint distribution: 2(K – 1) parameters

$$K - 1 + K - 1 = 2(K-1)$$

Discrete Variables: General Case

In a general joint distribution with M variables we need to store $K^M - 1$ parameters

If the distribution can be described by this graph:

We have only $K - 1 + (M - 1)\, K(K - 1)$ parameters.

This graph is called a **Markov chain** with M nodes.

The number of parameters grows only **linearly**.

Gaussian Variables

Assume all random variables are Gaussian and we define

$$p(x_i \mid \mathrm{pa}_i) = \mathcal{N}\left(x_i; \sum_{j \in \mathrm{pa}_i} w_{ij} x_j + b_i, v_i\right)$$

Then the joint probability p(x) is a multivariate Gaussian where

- Mean and covariance can be calculated recursively
- The fully connected graph corresponds to a Gaussian with a general symmetric covariance matrix
- The non-connected graph corresponds to a diagonal covariance matrix

Independence (Rep.)

Definition 1.4: Two random variables X and Y are *independent* iff: $p(x, y) = p(x)p(y)$

For independent random variables X and Y we have:

$$p(x \mid y) = \frac{p(x, y)}{p(y)} = \frac{p(x)p(y)}{p(y)} = p(x)$$

Notation: $x \perp\!\!\!\perp y \mid \emptyset$

Independence does not imply conditional independence.

The same is true for the opposite case.

Conditional Independence (Rep.)

Definition 1.5: Two random variables X and Y are ***conditional independent*** given a third random variable Z iff:

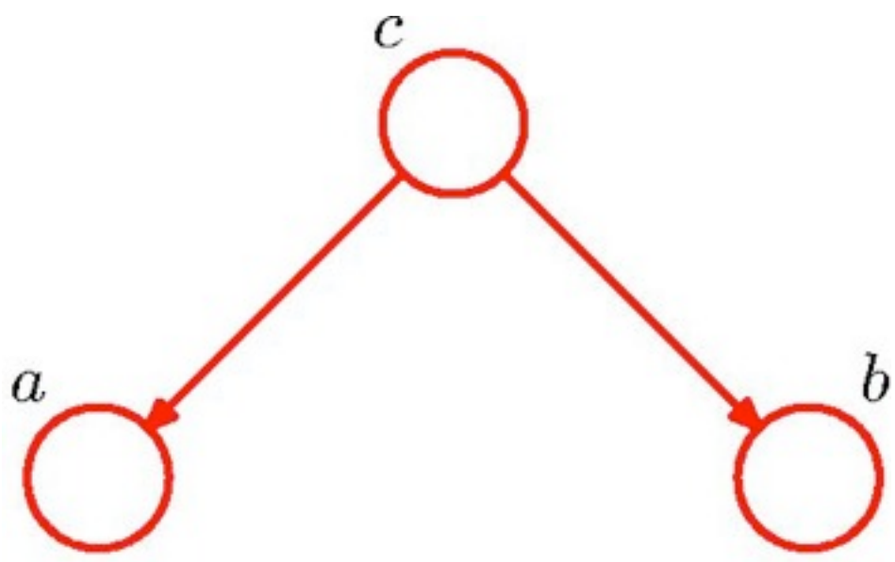
$$p(x, y \mid z) = p(x \mid z)p(y \mid z)$$

This is equivalent to:

$$p(x \mid z) = p(x \mid y, z) \quad \text{and}$$
$$p(y \mid z) = p(y \mid x, z)$$

Notation: $x \perp\!\!\!\perp y \mid z$

Conditional Independence: Example 1



- This graph represents the probability distribution:

$$p(a, b, c) = p(a|c)p(b|c)p(c)$$

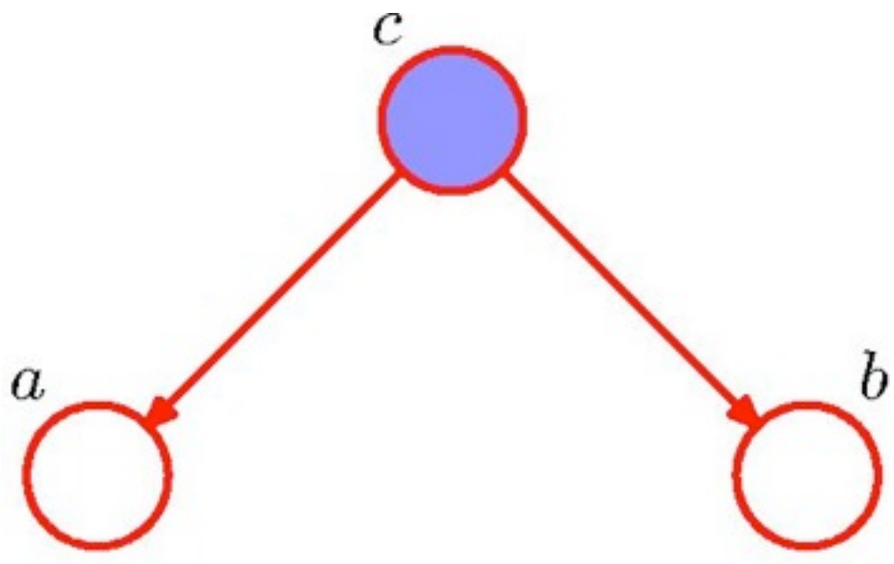
- Marginalizing out *c* on
- both sides gives

$$p(a, b) = \sum_c p(a|c)p(b|c)p(c)$$

Thus: a and b are not independent: $a \not\perp b \mid \emptyset$

Conditional Independence: Example 1

- Now, we condition on c (it is assumed to be known):

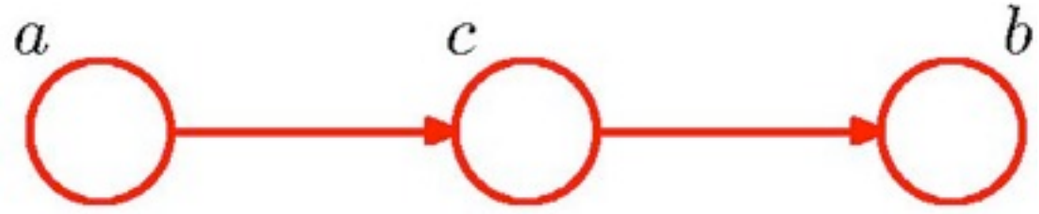


$$p(a,b|c) = \frac{p(a,b,c)}{p(c)}$$
$$= p(a|c)p(b|c)$$

Thus: a and b are conditionally independent given c : $a \perp\!\!\!\perp b \mid c$

We say that the node at c is a **tail-to-tail node** on the path between a and b

Conditional Independence: Example 2



This graph represents the distribution:

$$p(a, b, c) = p(a)p(c|a)p(b|c)$$

Again, we marginalize over c:

$$p(a, b) = p(a) \sum_c p(c|a)p(b|c) = p(a) \sum_c p(c|a)p(b|c, a)$$

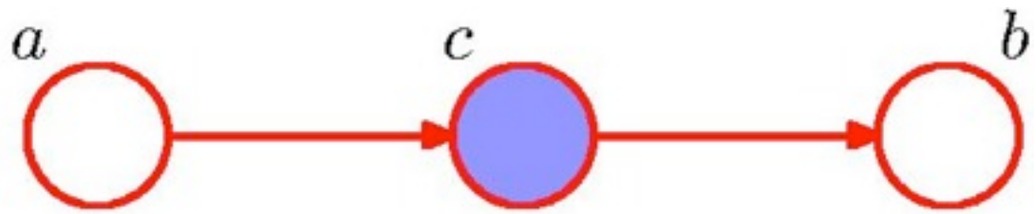
$$= p(a) \sum_c \frac{p(c, a)p(b, c, a)}{p(a)p(c, a)} = p(a) \sum_c p(b, c \mid a)$$

$$= p(a)p(b|a)$$

And we obtain: $a \not\perp b \mid \emptyset$

Conditional Independence: Example 2

- As before, now we condition on c :



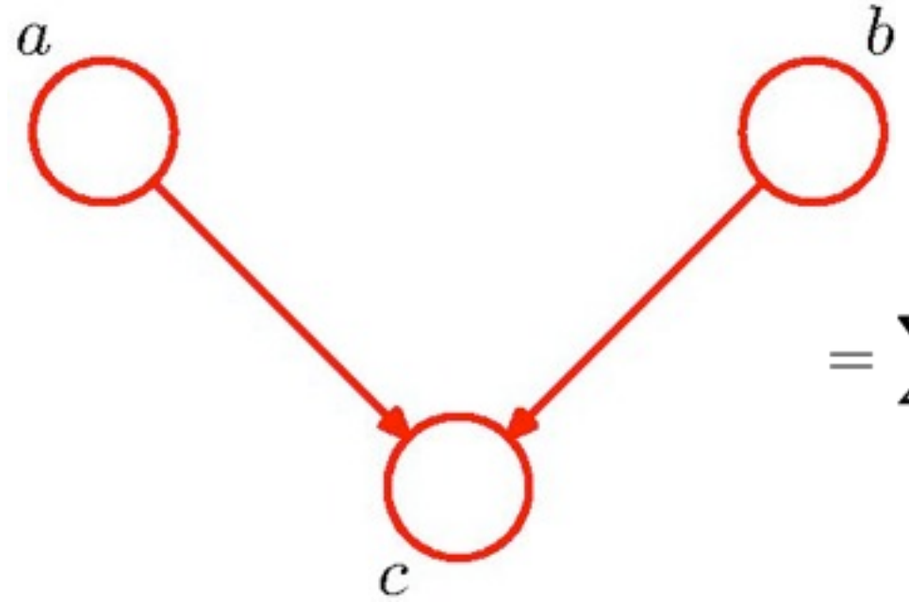
$$p(a,b|c) = \frac{p(a,b,c)}{p(c)}$$
$$= \frac{p(a)p(c|a)p(b|c)}{p(c)}$$
$$= p(a|c)p(b|c)$$

And we obtain: $a \perp\!\!\!\perp b \mid c$

We say that the node at c is a **head-to-tail node** on the path between a and b.

Conditional Independence: Example 3

Now consider this graph:



$$p(a, b, c) = p(a)p(b)p(c|a, b)$$

using:

$$\sum_c p(c \mid a, b) = \sum_c \frac{p(a \mid b, c)p(c \mid b)}{p(a \mid b)}$$

$$= \sum_c \frac{p(a \mid b, c)p(b \mid c)p(c)}{p(a \mid b)p(b)} = \sum_c \frac{p(a, b, c)}{p(a \mid b)p(b)} = 1$$

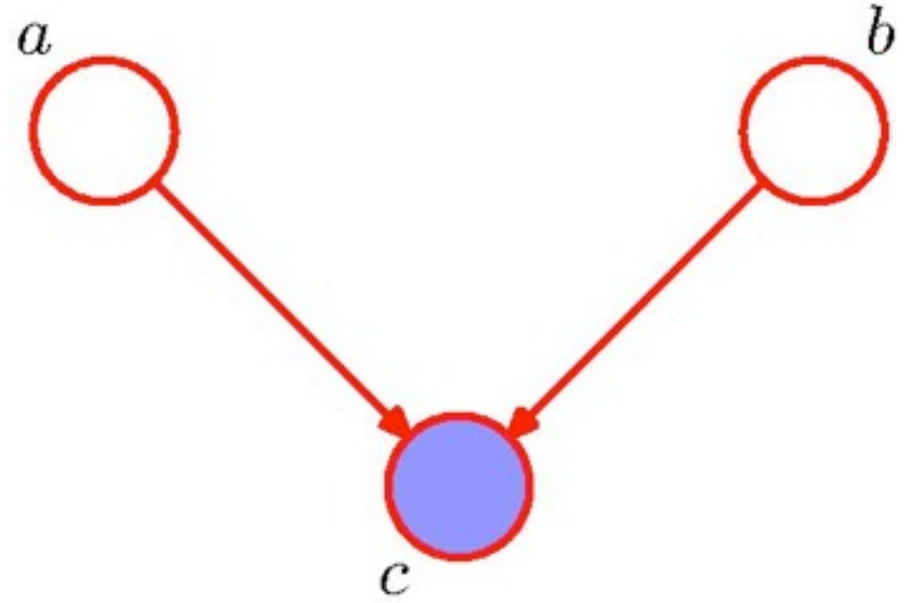
we obtain:

$$p(a, b) = p(a)p(b)$$

And the result is: $a \perp\!\!\!\perp b \mid \emptyset$

Conditional Independence: Example 3

- Again, we condition on c



$$p(a,b|c) = \frac{p(a,b,c)}{p(c)}$$

$$= \frac{p(a)p(b)p(c|a,b)}{p(c)}$$

This results in: $a \not\perp\!\!\!\perp b \mid c$

We say that the node at c is a **head-to-head node** on the path between a and b.

To Summarize

- When does the graph represent (conditional) independence?
- Tail-to-tail case: if we condition on the tail-to-tail node
- Head-to-tail case: if we cond. on the head-to-tail node
- Head-to-head case: if we do not condition on the head-to-head node (and neither on any of its descendants)

- In general, this leads to the notion of D-separation for directed graphical models.

D-Separation

- Say: A, B, and C are non-intersecting subsets of nodes in a directed graph.
- A path from A to B is blocked by C if it contains a node such that either

  a) the arrows on the path meet either head-to-tail or tail-to-tail at the node, and the node is in the set C, or

  b) the arrows meet head-to-head at the node, and neither the node, nor any of its descendants, are in the set C.

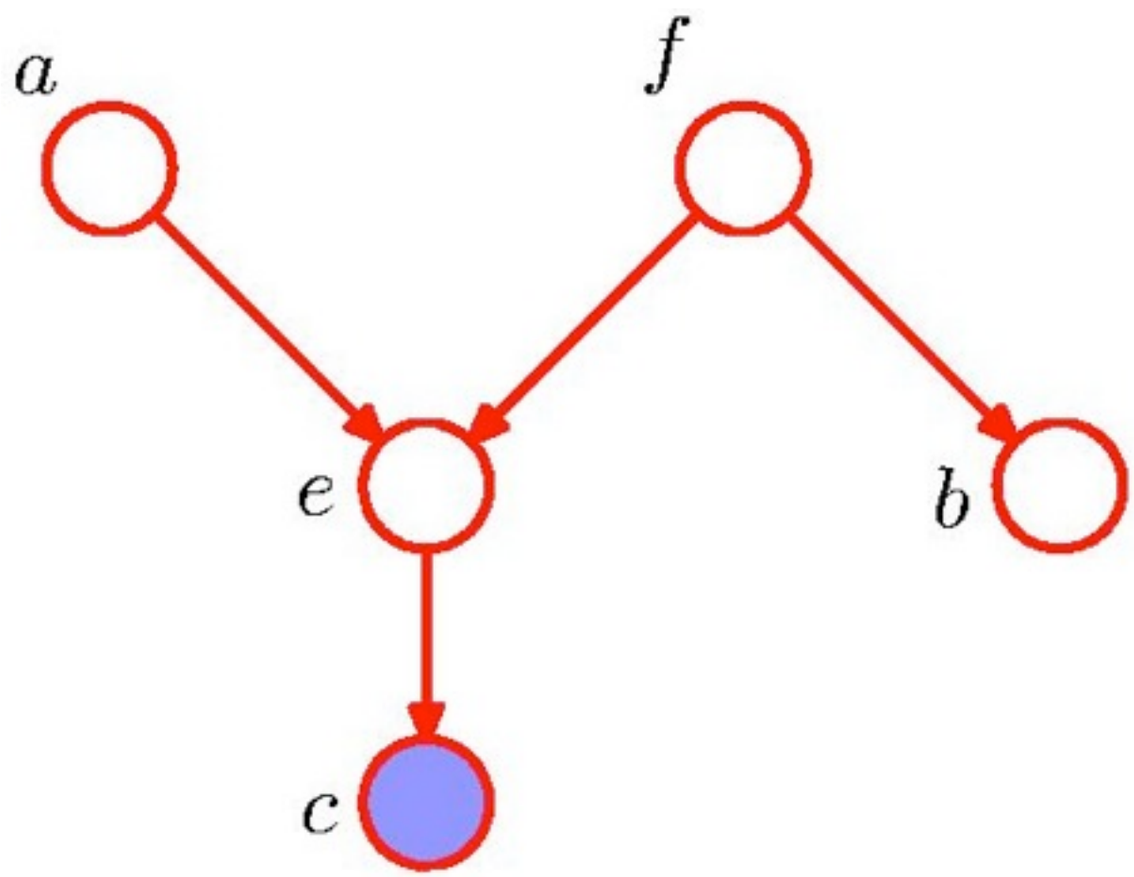
- If all paths from A to B are blocked, A is said to be d-separated from B by C. Notation: $\mathrm{dsep}(A, B \mid C)$

D-Separation

- Say: A, B, and C are non-intersecting subsets of nodes in a directed graph.
- A path
  a)the
  tail a
  b)the
  the
- If all p
  be d-s

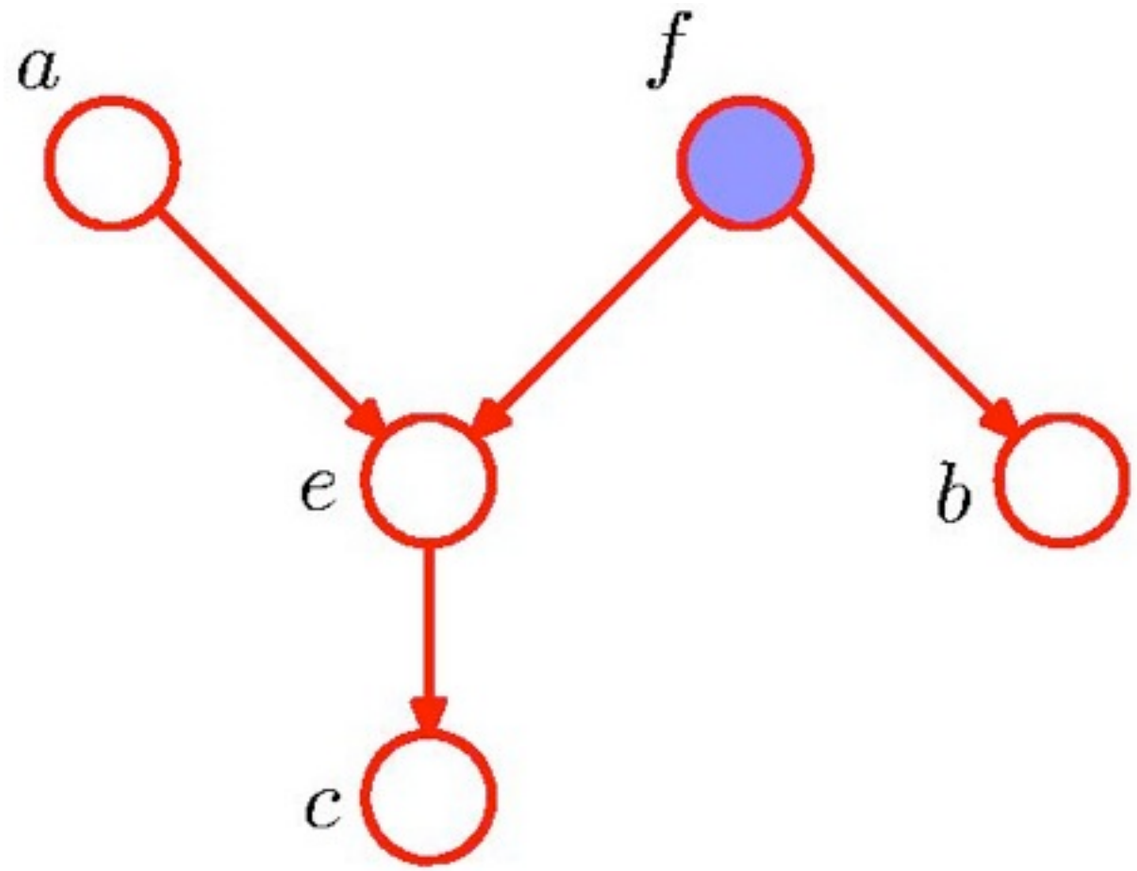
D-Separation is a property of graphs and not of probability distributions

D-Separation: Example



$$\neg \text{dsep}(a, b|c)$$

We condition on a descendant of e, i.e. it does not block the path from a to b.



$$\text{dsep}(a, b|f)$$

We condition on a tail-to-tail node on the only path from a to b, i.e f blocks the path.

I-Map

Definition 4.1: A graph G is called an **I-map** for a distribution p if every D-separation of G corresponds to a conditional independence relation satisfied by p:

$$\forall A, B, C : \mathrm{dsep}(A, B, C) \Rightarrow A \perp\!\!\!\perp B \mid C$$

Example: The fully connected graph is an I-map for any distribution, as there are no D-separations in that graph.

D-Map

Definition 4.2: A graph G is called an **D-map** for a distribution p if for every conditional independence relation satisfied by p there is a D-separation in G :

$$\forall A, B, C : A \perp\!\!\!\perp B \mid C \Rightarrow \mathrm{dsep}(A, B, C)$$

Example: The graph without any edges is a D-map for any distribution, as all pairs of subsets of nodes are D-separated in that graph.

Perfect Map

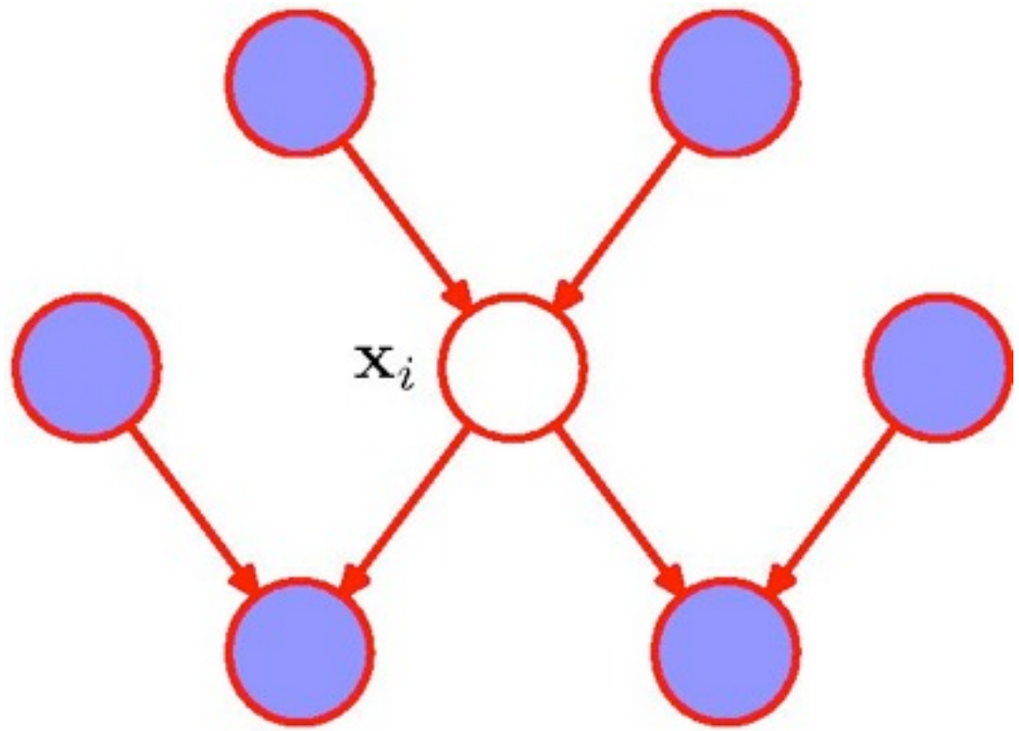
Definition 4.3: A graph G is called a **perfect map** for a distribution p if it is a D-map and an I-map of p.

$$\forall A, B, C : A \perp\!\!\!\perp B \mid C \Leftrightarrow \mathrm{dsep}(A, B, C)$$

A perfect map uniquely defines a probability distribution.

The Markov Blanket

- Consider a distribution of a node x_i conditioned on all other nodes:



$$
\begin{aligned}
p(\mathbf{x}_i|\mathbf{x}_{\{j\neq i\}}) &= \frac{p(\mathbf{x}_1,\ldots,\mathbf{x}_M)}{\int p(\mathbf{x}_1,\ldots,\mathbf{x}_M)d\mathbf{x}_i} \\
&= \frac{\prod_k p(\mathbf{x}_k|\mathrm{pa}_k)}{\int \prod_k p(\mathbf{x}_k|\mathrm{pa}_k)d\mathbf{x}_i} \\
&= p(\mathbf{x}_i \mid \mathbf{x}_{\mathcal{M}_i})
\end{aligned}
$$

Factors independent of x_i cancel between numerator and denominator.

Markov blanket $\mathcal{M}_i$ at x_i : all parents, children and co-parents of x_i.

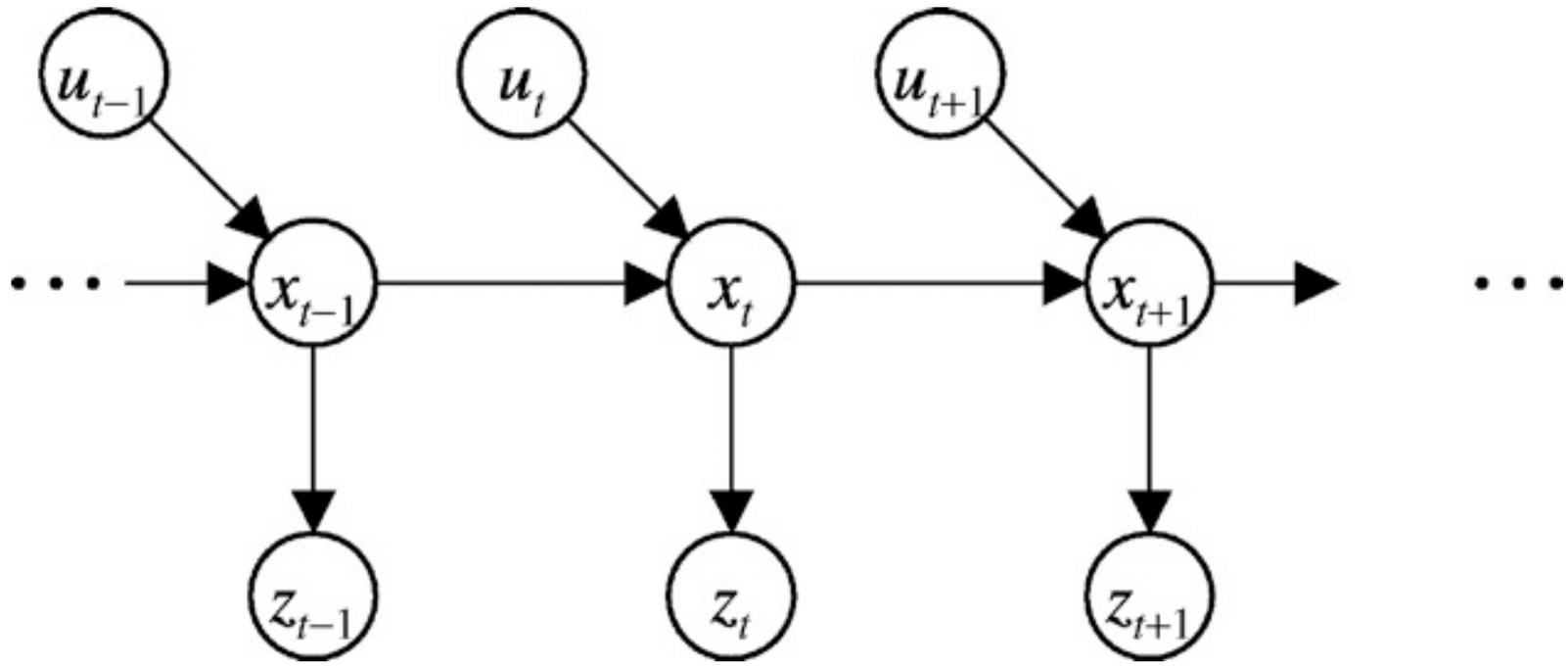
Summary

- Graphical models represent joint probability distributions using nodes for the random variables and edges to express (conditional) (in)dependence
- A prob. dist. can always be represented using a fully connected graph, but this is inefficient
- In a directed acyclic graph, conditional indepen-dence is determined using D-separation
- A perfect map implies a one-to-one mapping between c.i. relations and D-separations
- The Markov blanket is the minimal set of observed nodes to obtain conditional independence

Hidden Markov Models

Graphical Representation (Rep.)

We can describe the overall process using a ***Dynamic Bayes Network***:



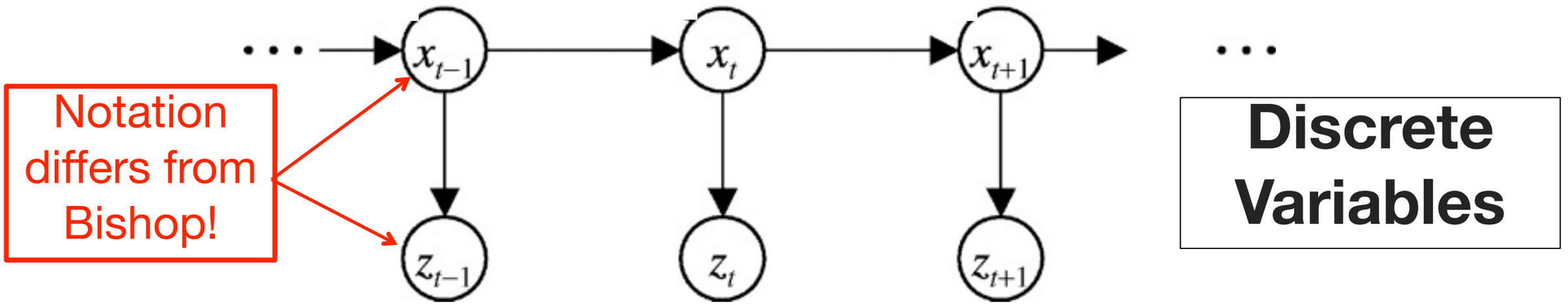
- This incorporates the following Markov assumptions:

$$p(z_t \mid x_{0:t}, u_{1:t}, z_{1:t}) = p(z_t \mid x_t) \quad \text{(measurement)}$$

$$p(x_t \mid x_{0:t-1}, u_{1:t}, z_{1:t}) = p(x_t \mid x_{t-1}, u_t) \quad \text{(state)}$$

Graphical Representation

We can describe the overall process using a ***Markov chain of latent variables***:



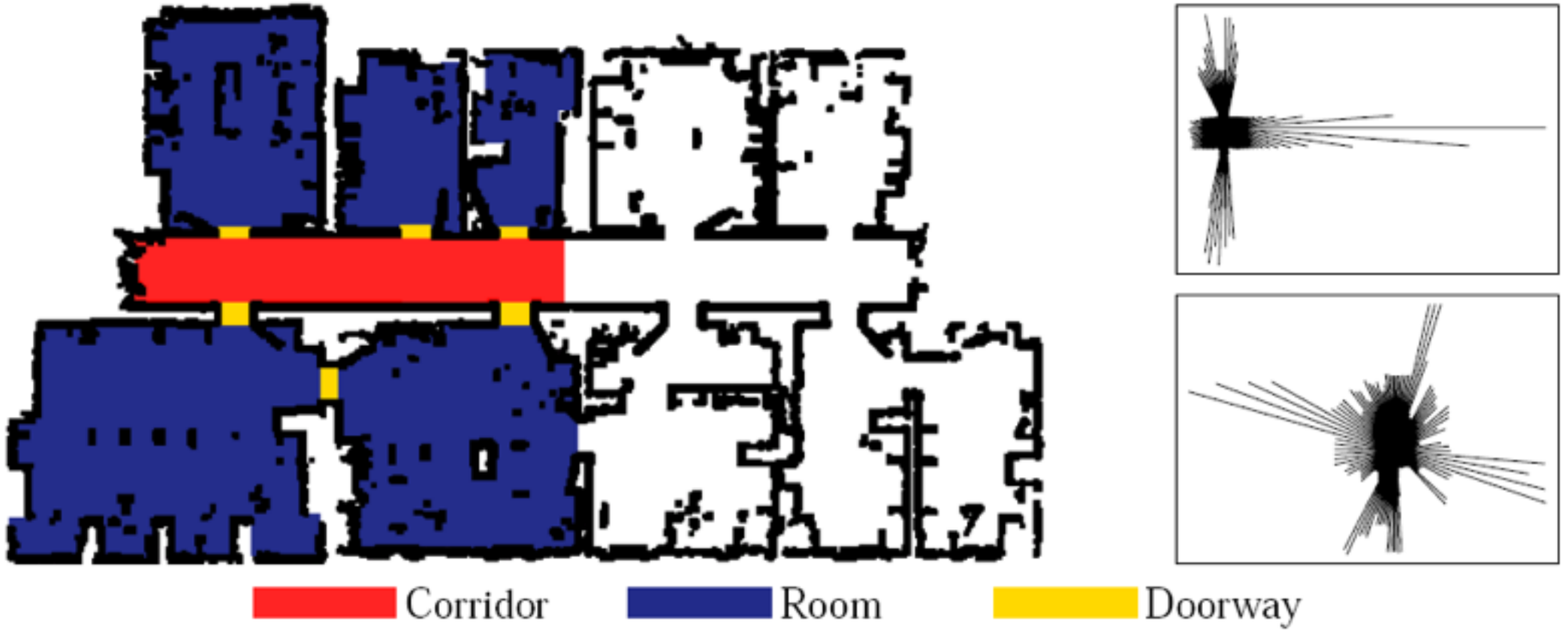
- This incorporates the following Markov assumptions:

$$p(z_t \mid x_{0:t}, z_{1:t}) = p(z_t \mid x_t) \quad \text{(measurement)}$$

$$p(x_t \mid x_{0:t-1}, z_{1:t}) = p(x_t \mid x_{t-1}) \quad \text{(state)}$$

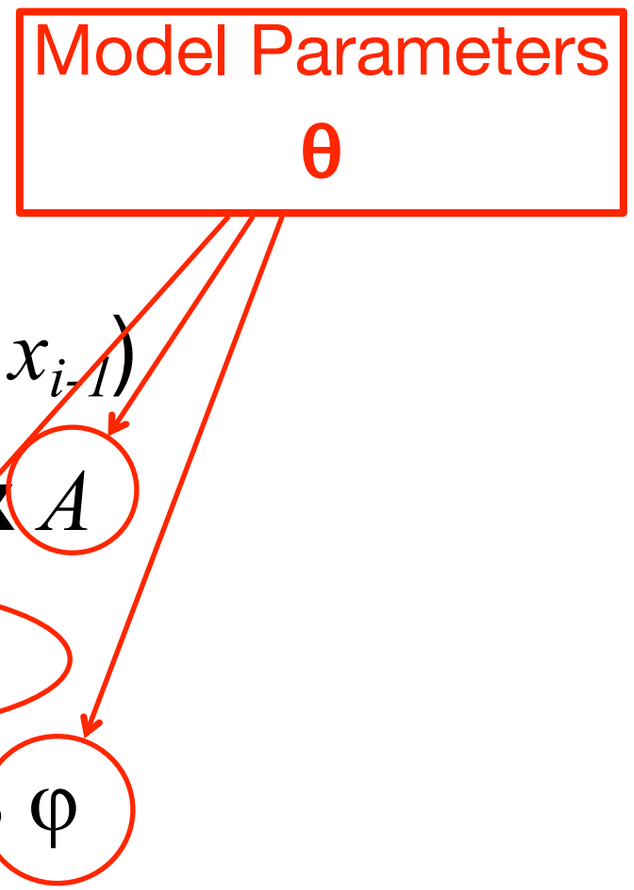
Example

- Place recognition for mobile robots
- 3 different states: corridor, room, doorway
- Problem: misclassifications
- Idea: use information from previous time step

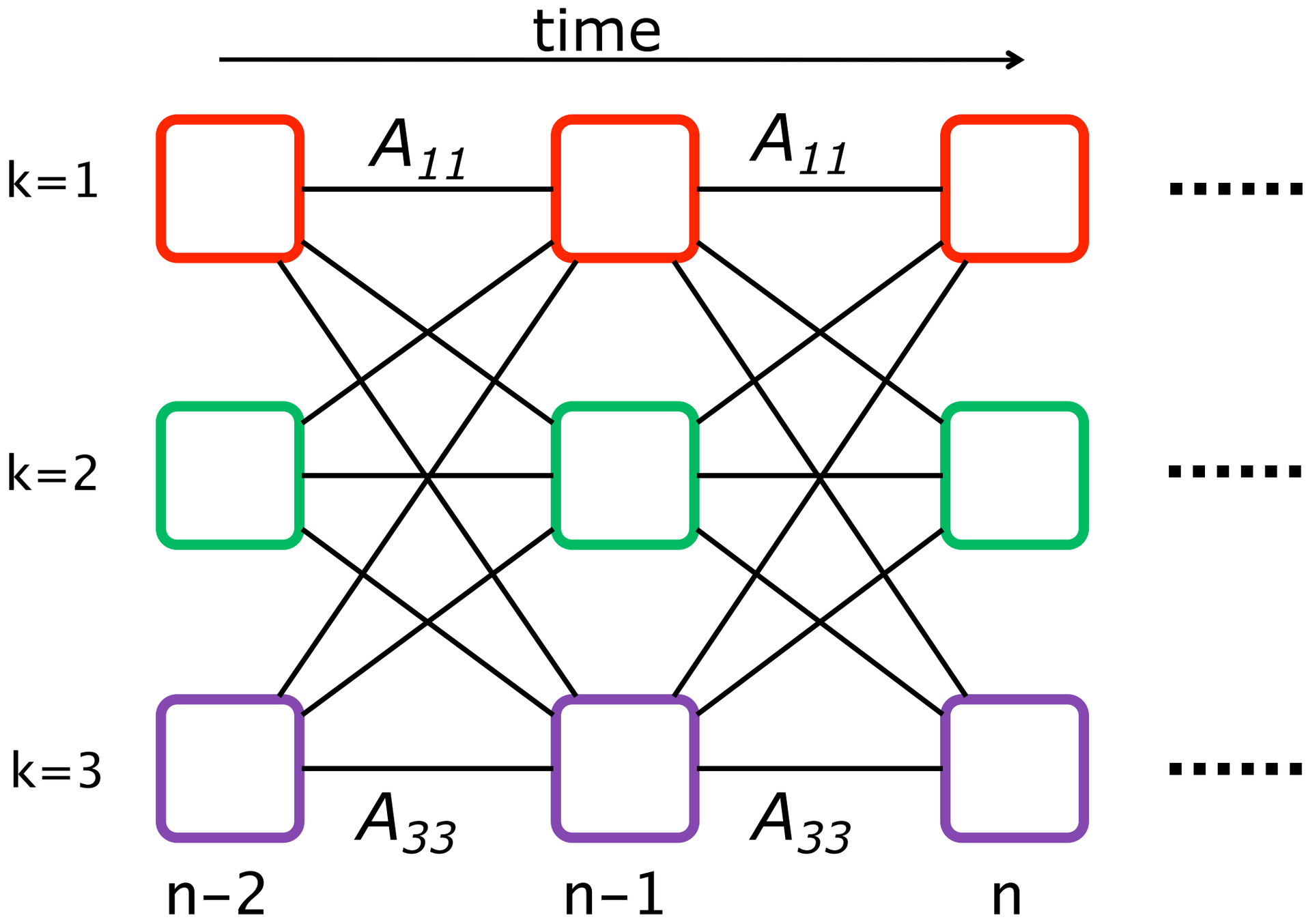


Formulation as HMM

1. Discrete random variables
   - Observation variables: $\{z_n\}$, $n = 1..N$
   - State variables (unobservable): $\{x_n\}$, $n = 1..N$
   - Number of states K: $x_n \epsilon \{1..K\}$
2. Transition model $p(x_i \mid x_{i-1})$
   - Markov assumption (x_i only depends on x_{i-1})
   - Represented as a $K \times K$ **transition matrix** A
   - Initial probability: $p(x_0)$ repr. as π_1, π_2, π_3
3. Observation model $p(z_i \mid x_i)$ with parameters φ
   - Observation only depends on the current state
   - Example: output of a "local" place classifier



The Trellis Representation



Application Example (1)

- Given an observation sequence $z_1, z_2, z_3 \ldots$
- Assume that the model parameters $\theta = (A, \pi, \varphi)$ are known
- What is the probability that the given observation sequence is actually observed under this model, i.e. $p(Z|\ \theta)$?
- If we are given several different models, we can choose the one with highest probability
- Expressed as a **supervised learning problem**, this can be interpreted as the inference step (classification step)

Application Example (2)

- Given an observation sequence $z_1, z_2, z_3 \ldots$
- Assume that the model parameters $\theta = (A, \pi, \varphi)$ are known
- What is the state sequence $x_1, x_2, x_3 \ldots$ that explains best the given observation sequence?
- In the case of place recognition: which is the sequence of truly visited places that explains best the sequence of obtained place labels (classifications)?

Application Example (3)

- Given an observation sequence $z_1, z_2, z_3 \ldots$
- What are the optimal model parameters $\theta = (A, \pi, \varphi)$?
- This can be interpreted as the **training step**
- Is in general the most difficult problem

Summary: 3 Operations on HMMs

1. Compute data likelihood $p(Z|\theta)$ from a known model
   - Can be computed with the **forward-backward** algorithm

2. Compute optimal state sequence with a known model
   - Can be computed with the **Viterbi**-Algorithm

1. Learn model parameters for an observation sequence
   - Can be computed using **Expectation-Maximization** (or Baum-Welch)

1. Computing the Data Likelihood

- Assume: given a state sequence $x_1, x_2, x_3 \ldots$
- Compute $p(Z|X, \theta) = \prod_{n=1}^{N} p(z_n|x_n, \theta)$
- The probability of this state sequence is

$$p(X|\theta) = p(x_0)\, p(x_1|x_0)\, p(x_2|x_1)\ldots$$

- Thus, we have $p(Z,X|\ \theta) = p(Z|X, \theta)\ p(X|\theta)$
- We need $p(Z|\ \theta) = \prod_{\text{all } X} p(Z,X|\ \theta)$, but this is intractable

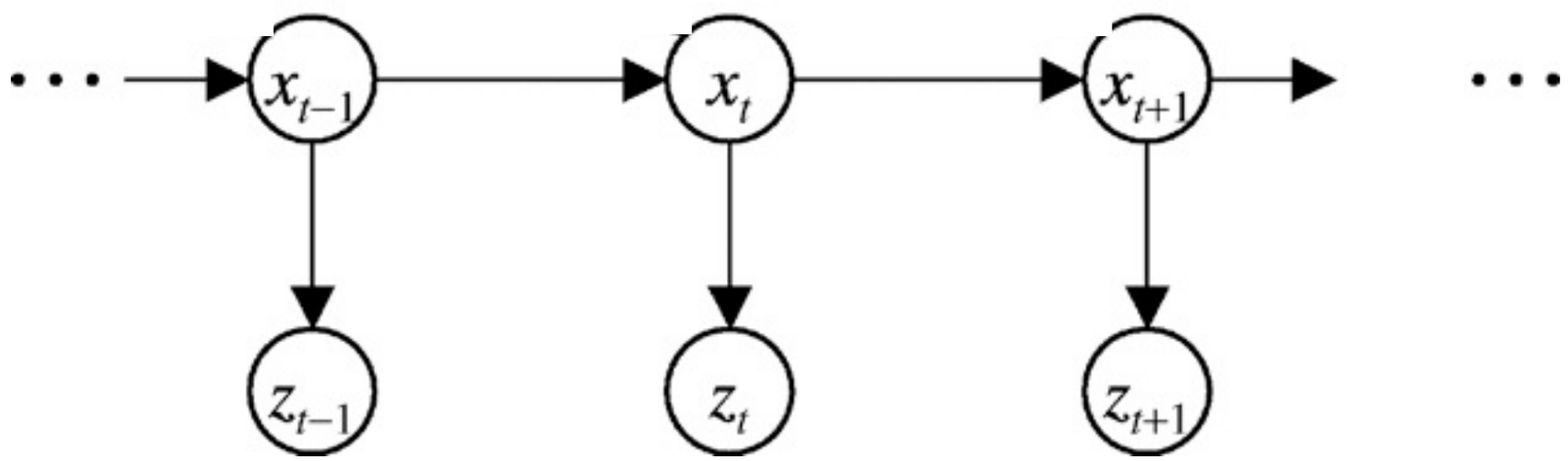
The Forward-Backward Algorithm

- Define $\alpha(x_n) = p(z_1, z_2, \ldots, z_n, x_n)$

The Forward-Backward Algorithm

- Define $\alpha(x_n) = p(z_1, z_2, \ldots, z_n, x_n)$
- This can be recursively computed:

$$\alpha(x_n) = p(z_n|x_n) \sum_{x_{n-1}} \alpha(x_{n-1})\, p(x_n|x_{n-1})$$



The Forward-Backward Algorithm

- Define $\alpha(x_n) = p(z_1, z_2, \ldots, z_n, x_n)$
- This can be recursively computed:

$$\alpha(x_n) = p(z_n|x_n) \sum_{x_{n-1}} \alpha(x_{n-1})\, p(x_n|x_{n-1})$$

- Then we have:

$$p(Z|\theta) = \sum_{x_N} \alpha(x_N)$$

Solution to first problem!

The Forward-Backward Algorithm

- Define $\alpha(x_n) = p(z_1, z_2, \ldots, z_n, x_n)$
- This can be recursively computed:

$$\alpha(x_n) = p(z_n|x_n) \sum_{x_{n-1}} \alpha(x_{n-1})\, p(x_n|x_{n-1})$$

- Then we have:

$$p(Z|\theta) = \sum_{x_N} \alpha(x_N)$$

Solution to first problem!

- Similarly, we define $\beta(x_n) = p(z_{n+1}, z_{n+2}, \ldots, z_N|x_n)$
- This can also be recursively computed:

$$\beta(x_n) = \sum_{x_{n+1}} \beta(x_{n+1})\, p(z_{n+1}|x_{n+1})\, p(x_{n+1}|x_n)$$

The Forward-Backward Algorithm

- this can be used to computey computed:

2. Computing the Most Likely States

- Goal: find a state sequence $x_1,x_2,x_3\ldots$ that maximizes the probability $\mathrm{p}(\mathrm{X},\mathrm{Z}|\theta)$

- Define $\delta(x_n)= \max_{x_1,\ldots,x_{n-1}} \mathrm{p}(x_1,x_2,\ldots,x_n,z_1,z_2,\ldots,z_n)$

2. Computing the Most Likely States

- Goal: find a state sequence $x_1, x_2, x_3 \ldots$ that maximizes the probability $p(X,Z|\theta)$

- Define $\delta(x_n) = \max_{x_1,\ldots,x_{n-1}} p(x_1, x_2, \ldots, x_n, z_1, z_2, \ldots, z_n)$

- This can be recursively computed:

$$\delta(x_n) = p(z_n|x_n) \max_{x_{n-1}} [\delta(x_{n-1}) \, p(x_n|x_{n-1})]$$

2. Computing the Most Likely States

- Goal: find a state sequence $x_1, x_2, x_3 \ldots$ that maximizes the probability $p(X,Z|\theta)$

- Define $\delta(x_n) = \max_{x_1,\ldots,x_{n-1}} p(x_1, x_2, \ldots, x_n, z_1, z_2, \ldots, z_n)$

- This can be recursively computed:

$$\delta(x_n) = p(z_n|x_n) \max_{x_{n-1}} [\delta(x_{n-1})\, p(x_n|x_{n-1})]$$

- But we also need the argmax:

$$\psi(x_n) = \text{argmax} [\delta(x_{n-1})\, p(x_n|x_{n-1})]$$

The Viterbi algorithm

- Initialize:
  - $\delta(x_0) = p(x_0)\, p(z_0 \mid x_0)$
  - $\psi(x_0) = 0$
- Compute recursively for $n=1 \ldots N$:
  - $\delta(x_n) = p(z_n|x_n) \max_{x_{n-1}} [\delta(x_{n-1})\, p(x_n|x_{n-1})]$
  - $\psi(x_n) = \operatorname{argmax}_{x_{n-1}} [\delta(x_{n-1})\, p(x_n|x_{n-1})]$
- On termination:
  - $p(Z,X|\theta) = \max_{x_N} \delta(x_N)$
  - $x_N^* = \operatorname{argmax}_{x_N} \delta(x_N)$

3. Learning the Model Parameters

- Given an observation sequence $z_1, z_2, z_3 \ldots$
- Find optimal model parameters θ
- We need to maximize the likelihood $p(Z|\theta)$
- Can not be solved in closed form
- Iterative algorithm:
  Expectation Maximization (EM) or for the case of HMMs: Baum-Welch algorithm

Expectation maximisation

- Objective: Find the model parameters knowing the observations: π,A,ϕ
- Result:
  - Train the HMM to recognize sequences of input
  - Train the HMM to generate sequences of input
- Technique: Expectation Maximisation
  - E: Find the best state sequence given the parameters
  - M: Find the parameters using the state sequence
  - Maximisation of the log-likelihood:

$$\arg\max_{pi,A,\phi} -\log\left(P\left(\{Z_i\}^{..}\pi, A, \phi\right)\right)$$

The Baum-Welsh algorithm

- E-Step (assuming we know π,A,ϕ, i.e. θ^{old})
- Define the posterior probability of being in state i at step k:
- Define $\gamma(x_n) = p(x_n|Z)$

The Baum-Welsh algorithm

- E-Step (assuming we know π,A,ϕ, i.e. θ^{old})
- Define the posterior probability of being in state i at step k:
- Define $\gamma(x_n) = p(x_n|Z)$
- It follows that $\gamma(x_n) = \alpha(x_n)\,\beta(x_n) / p(Z)$

The Baum-Welsh algorithm

- E-Step (assuming we know π,A,ϕ, i.e. θ^{old})
- Define the posterior probability of being in state i at step k:
- Define $\gamma(x_n) = p(x_n|Z)$
- It follows that $\gamma(x_n) = \alpha(x_n)\,\beta(x_n) / p(Z)$
- Define $\xi(x_{n-1}, x_n) = p(x_{n-1}, x_n|Z)$
- It follows that $\xi(x_{n-1}, x_n) =$

$$\alpha(x_{n-1})p(z_n|x_n)p(x_n|x_{n-1})\beta(x_n) / p(Z)$$

The Baum-Welsh algorithm

- Maximizing Q also maximizes the likelihood:
  $p(Z|\theta) \geq p(Z|\theta^{old})$
- M-Step:
  - $\pi_k = \gamma(x_1=k) / \sum_j \gamma(x_1=j)$

here, we need forward and backward step!

  - $A_{ij} = \sum_n \xi(x_{n-1}=i, x_n=j) / \sum_k \sum_n \xi(x_{n-1}=i, x_n=k)$
- With these new values, Q is recomputed
- This is done until the likelihood does not increase anymore (convergence)

The Baum-Welsh algorithm - summary

- Start with an initial estimate of $\theta=(\pi,A,\varphi)$
  e.g. uniformly and k-means for φ
- Compute $Q(\theta,\theta^{old})$ (E-Step)
- Maximize Q (M-step)
- Iterate E and M until convergence
- In each iteration one full application of the forward-backward algorithm is performed
- Result gives a **local** optimum
- For other local optima, the algorithm needs to be started again with new initialization

The Scaling problem

- Probability of sequences

$$\prod_i P(X_i \ldots) \to 1$$

<1

  - Probabilities are very small
  - The product of the terms soon is very small

- Usually: convert to log-space works
- But: we have sums of products!
- Solution: Rescale/Normalise the probability during the computation, e.g.:

$$\hat{\alpha}(x_n) = \alpha(x_n) \,/\, p(z_1, z_2, \ldots, z_n)$$

Summary

- HMMs are a way to model sequential data
- They assume discrete states
- Three possible operations can be performed with HMMs:
  - Data likelihood, given a model and an observation
  - Most likely state sequence, given a model and an observation
  - Optimal Model parameters, given an observation
- Appropriate scaling solves numerical problems
- HMMs are widely used, e.g. in speech recognition