Summary: MAP Estimation

To summarize, we have the following optimization problem:

$$J(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}(\mathbf{w}^T\phi(\mathbf{x}_n) - t_n)^2 + \frac{\lambda}{2}\mathbf{w}^T\mathbf{w} \qquad \phi(\mathbf{x}_n) \in \mathbb{R}^M$$

The same in vector notation:

$$J(\mathbf{w}) = \frac{1}{2}\mathbf{w}^T\Phi^T\Phi\mathbf{w} - \mathbf{w}\Phi^T\mathbf{t} + \frac{1}{2}\mathbf{t}^T\mathbf{t} + \frac{\lambda}{2}\mathbf{w}^T\mathbf{w} \quad \mathbf{t} \in \mathbb{R}^N$$

$$\Phi = \begin{pmatrix} \phi_0(x_1) & \phi_1(x_1) & \dots & \phi_{M-1}(x_1) \\ \phi_0(x_2) & \phi_1(x_2) & \dots & \phi_{M-1}(x_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_0(x_N) & \phi_1(x_N) & \dots & \phi_{M-1}(x_N) \end{pmatrix} \in \mathbb{R}^{N\times M}$$

“Feature Matrix”

Summary: MAP Estimation

To summarize, we have the following optimization problem:

$$J(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}(\mathbf{w}^T\phi(\mathbf{x}_n) - t_n)^2 + \frac{\lambda}{2}\mathbf{w}^T\mathbf{w} \qquad \phi(\mathbf{x}_n) \in \mathbb{R}^M$$

The same in vector notation:

$$J(\mathbf{w}) = \frac{1}{2}\mathbf{w}^T\Phi^T\Phi\mathbf{w} - \mathbf{w}\Phi^T\mathbf{t} + \frac{1}{2}\mathbf{t}^T\mathbf{t} + \frac{\lambda}{2}\mathbf{w}^T\mathbf{w} \quad \mathbf{t} \in \mathbb{R}^N$$

And the solution is

$$\mathbf{w}^* = (\lambda I_M + \Phi^T\Phi)^{-1}\Phi^T\mathbf{t}$$

Identity matrix of size M by M

MLE And MAP

- The benefit of MAP over MLE is that prediction is less sensitive to **overfitting,** i.e. even if there is only little data the model predicts well.
- This is achieved by using **prior information,** i.e. model assumptions that are not based on any observations (= data)
- But: both methods only give the **most likely** model, there is no notion of **uncertainty** yet

Idea 1: Find a **distribution** over model parameters (“parameter posterior”)

MLE And MAP

- The benefit of MAP over MLE is that prediction is less sensitive to **overfitting,** i.e. even if there is only little data the model predicts well.
- This is achieved by using **prior information,** i.e. model assumptions that are not based on any observations (= data)
- But: both methods only give the **most likely** model, there is no notion of **uncertainty** yet

Idea 1: Find a **distribution** over model parameters

Idea 2: Use that distribution to estimate **prediction uncertainty** ("predictive distribution")

When Bayes Meets Gauß

Theorem: If we are given this:

I. $$p(\mathbf{x}) = \mathcal{N}(\mathbf{x} \mid \mu, \Sigma_1)$$

II. $$p(\mathbf{y} \mid \mathbf{x}) = \mathcal{N}(\mathbf{y} \mid A\mathbf{x} + \mathbf{b}, \Sigma_2)$$

linear dependency on $\mathbf{x}$

Then it follows (properties of Gaussians):

III. $$p(\mathbf{y}) = \mathcal{N}(\mathbf{y} \mid A\mu + \mathbf{b}, \Sigma_2 + A\Sigma_1 A^T)$$

IV. $$p(\mathbf{x} \mid \mathbf{y}) = \mathcal{N}(\mathbf{x} \mid \Sigma(A^T\Sigma_2^{-1}(\mathbf{y} - \mathbf{b}) + \Sigma_1^{-1}\mu), \Sigma)$$

where

$$\Sigma = (\Sigma_1^{-1} + A^T\Sigma_2^{-1}A)^{-1}$$

See Bishop's book for the proof!

"Linear Gaussian Model"

When Bayes Meets Gauß

Thus: When using the Bayesian approach, we can do even more than MLE and MAP by using these formulae.

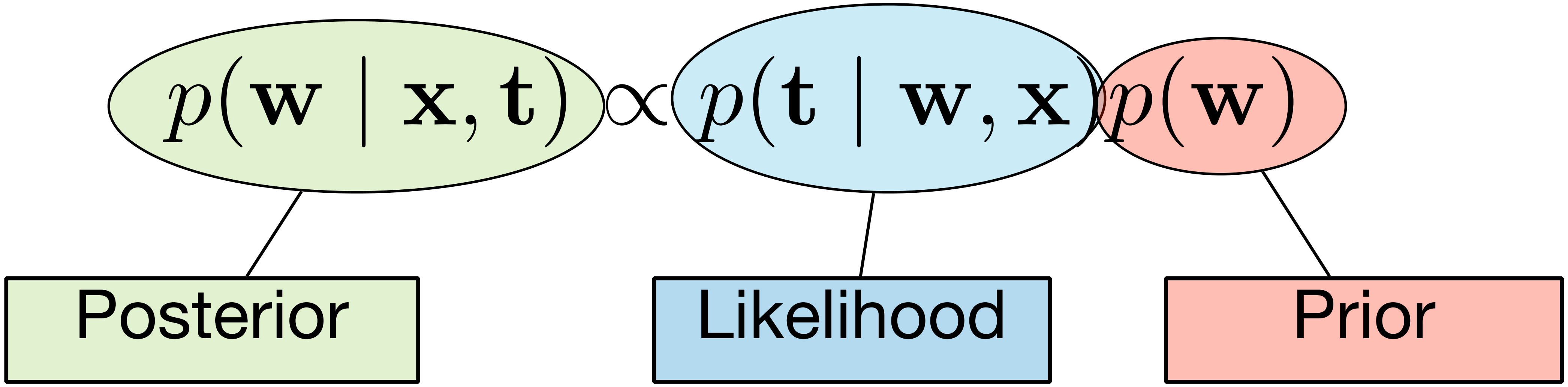
This means:

If the prior and the likelihood are Gaussian then the **posterior** and the **normalizer** are also Gaussian and we can compute them in closed form.

This gives us a natural way to compute uncertainty!

The Posterior Distribution

Remember Bayes Rule:



$$p(\mathbf{w} \mid \mathbf{x}, \mathbf{t}) \propto p(\mathbf{t} \mid \mathbf{w}, \mathbf{x}) p(\mathbf{w})$$

Posterior

Likelihood

Prior

With our theorem, we can compute the posterior in **closed form** (and not just its maximum)!

The posterior is also a Gaussian and its **mean** is the MAP solution.

The Posterior Distribution

We have $p(\mathbf{w}) = \mathcal{N}(\mathbf{w}; \mathbf{0}, \sigma_2^2 I_M)$

and $p(\mathbf{t} \mid \mathbf{w}, \mathbf{x}) = \mathcal{N}(\mathbf{t}; \Phi\mathbf{w}, \sigma_1^2 I_N)$

From this and IV. we get the **posterior covariance:**

$$\Sigma = (\sigma_2^{-2} I_M + \sigma_1^{-2} \Phi^T \Phi)^{-1}$$

$$= \sigma_1^2 (\frac{\sigma_1^2}{\sigma_2^2} I_M + \Phi^T \Phi)^{-1}$$

and the **mean:** $\boldsymbol{\mu} = \sigma_1^{-2} \Sigma \Phi^T \mathbf{t}$

Note: So far we only used the **training** data! $(\mathbf{x}, \mathbf{t})$

So the entire posterior distribution is

$$p(\mathbf{w} \mid \mathbf{t}, \mathbf{x}) = \mathcal{N}(\mathbf{w}; \boldsymbol{\mu}, \Sigma)$$

The Predictive Distribution

We obtain the **predictive distribution** by integrating over all possible model parameters ("inference"):

$$p(\underbrace{t^*|\,x^*}_{\text{Test data}}, \mathbf{t}, \mathbf{x}) = \int \underbrace{p(t^*|\,x^*, \mathbf{w})}_{\text{Test data likelihood}}\underbrace{p(\mathbf{w}\,|\,\mathbf{x}, \mathbf{t})}_{\text{Parameter posterior}}d\mathbf{w}$$

This distribution can be computed in closed form, because both terms on the RHS are Gaussian.

From above we have $\boxed{p(\mathbf{w}\,|\,\mathbf{t}, \mathbf{x})} = \mathcal{N}(\mathbf{w}; \boldsymbol{\mu}, \Sigma)$

where $\boldsymbol{\mu} = \sigma_1^{-2}\Sigma\Phi^T\mathbf{t}$

and $\Sigma = \sigma_1^2(\frac{\sigma_1^2}{\sigma_2^2}I_M + \Phi^T\Phi)^{-1}$

The Predictive Distribution

We obtain the **predictive distribution** by integrating over all possible model parameters (“inference”):

$$p(\underbrace{t^* \mid x^*}_{\text{Test data}}, \mathbf{t}, \mathbf{x}) = \int \underbrace{p(t^* \mid x^*, \mathbf{w})}_{\text{Test data likelihood}} \underbrace{p(\mathbf{w} \mid \mathbf{x}, \mathbf{t})}_{\text{Parameter posterior}} d\mathbf{w}$$

This distribution can be computed in closed form, because both terms on the RHS are Gaussian.

From above we have $\boxed{p(\mathbf{w} \mid \mathbf{t}, \mathbf{x})} = \mathcal{N}(\mathbf{w}; \boldsymbol{\mu}, \Sigma)$

where $\boldsymbol{\mu} = \sigma_1^{-2} \Sigma \Phi^T \mathbf{t}$

and $\Sigma = \sigma_1^2 (\frac{\sigma_1^2}{\sigma_2^2} I_M + \Phi^T \Phi)^{-1}$

$$\Rightarrow \boldsymbol{\mu} = (\lambda I_M + \Phi^T \Phi)^{-1} \Phi^T \mathbf{t}$$

MAP solution

The Predictive Distribution

Using formula III. from above (linear Gaussian),

$$p(t^* \mid x^*, \mathbf{t}, \mathbf{x}) = \int p(t^* \mid x^*, \mathbf{w}) p(\mathbf{w} \mid \mathbf{x}, \mathbf{t}) d\mathbf{w}$$

$$= \int \mathcal{N}(t^*; \phi(x^*)^T \mathbf{w}, \sigma) \mathcal{N}(\mathbf{w}; \boldsymbol{\mu}, \Sigma) d\mathbf{w}$$

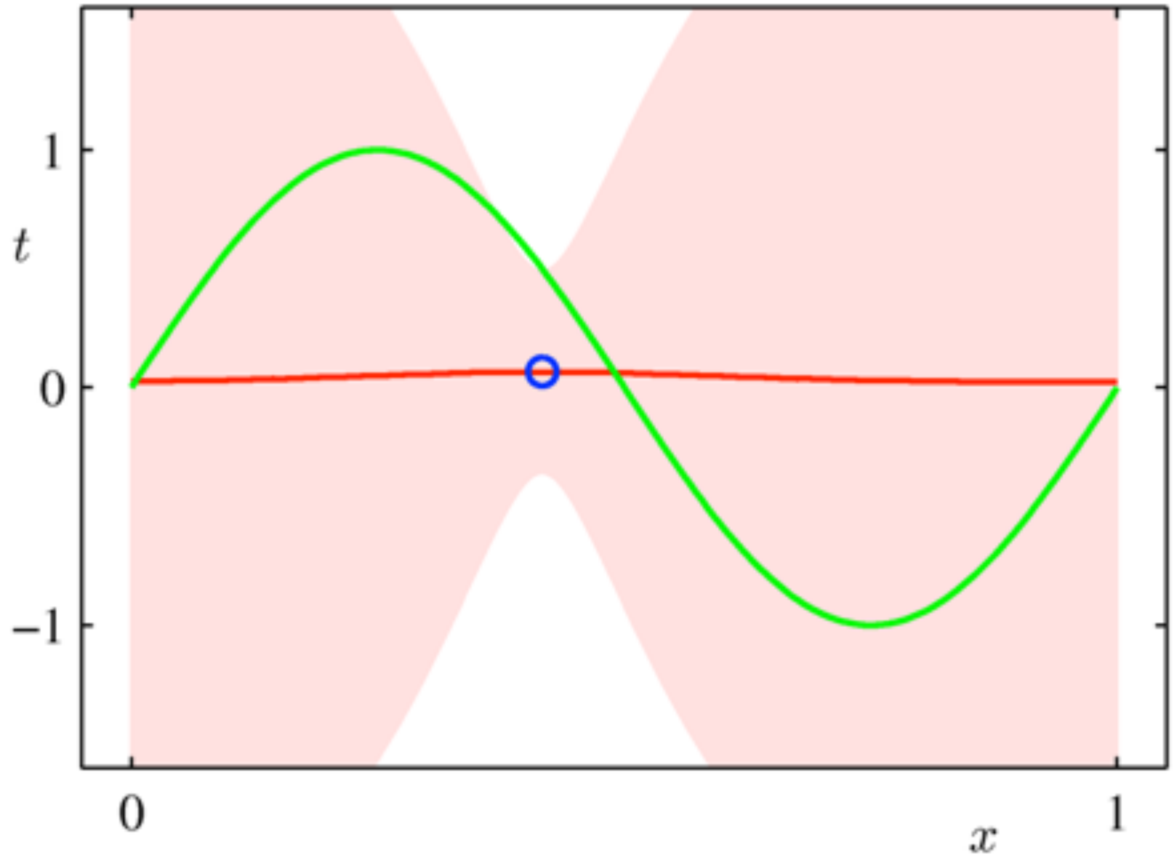
$$= \mathcal{N}(t^*; \phi(x^*)^T \boldsymbol{\mu}, \sigma_N^2(x^*))$$

where

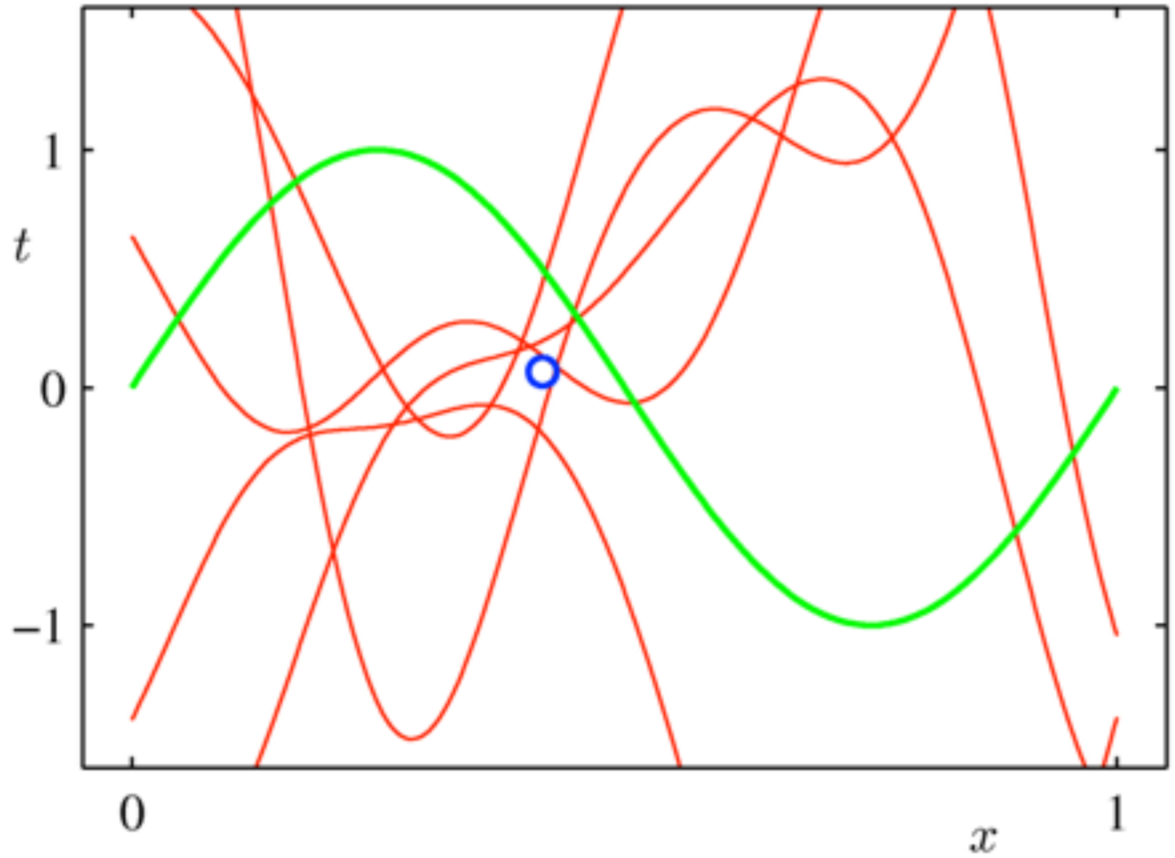
$$\sigma_N^2(x) = \sigma^2 + \phi(x)^T \Sigma \phi(x)$$

The Predictive Distribution (2)

- Example: Sinusoidal data, 9 Gaussian basis functions, 1 data point



The predictive distribution

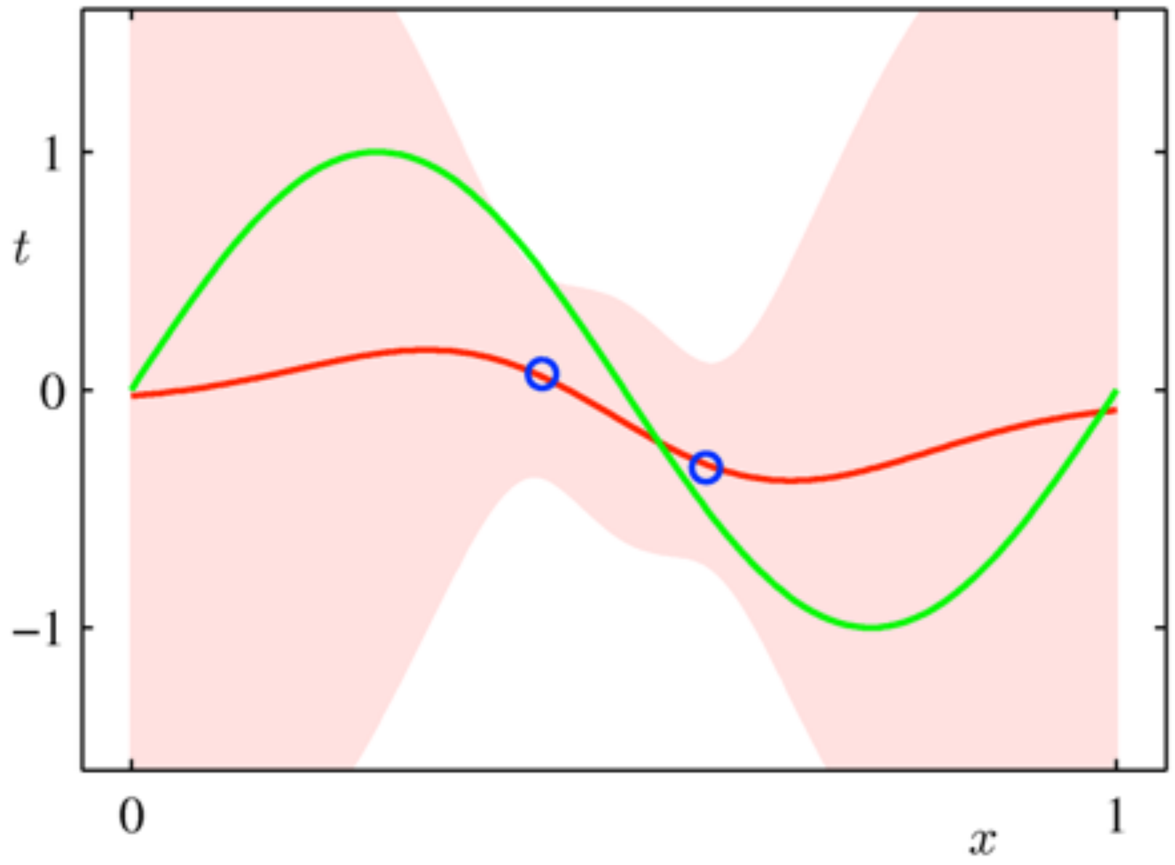


Some samples from the posterior

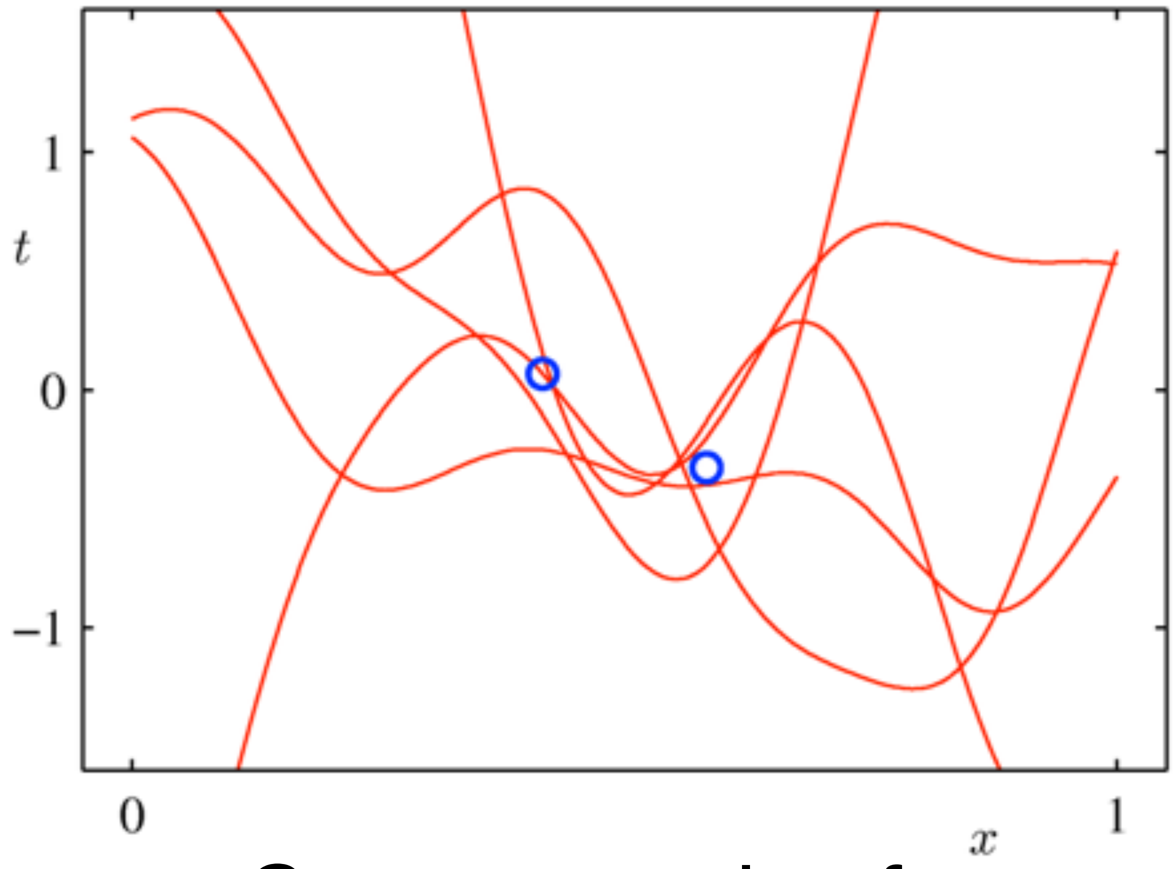
From: C.M. Bishop

Predictive Distribution (3)

- Example: Sinusoidal data, 9 Gaussian basis functions, 2 data points



The predictive distribution

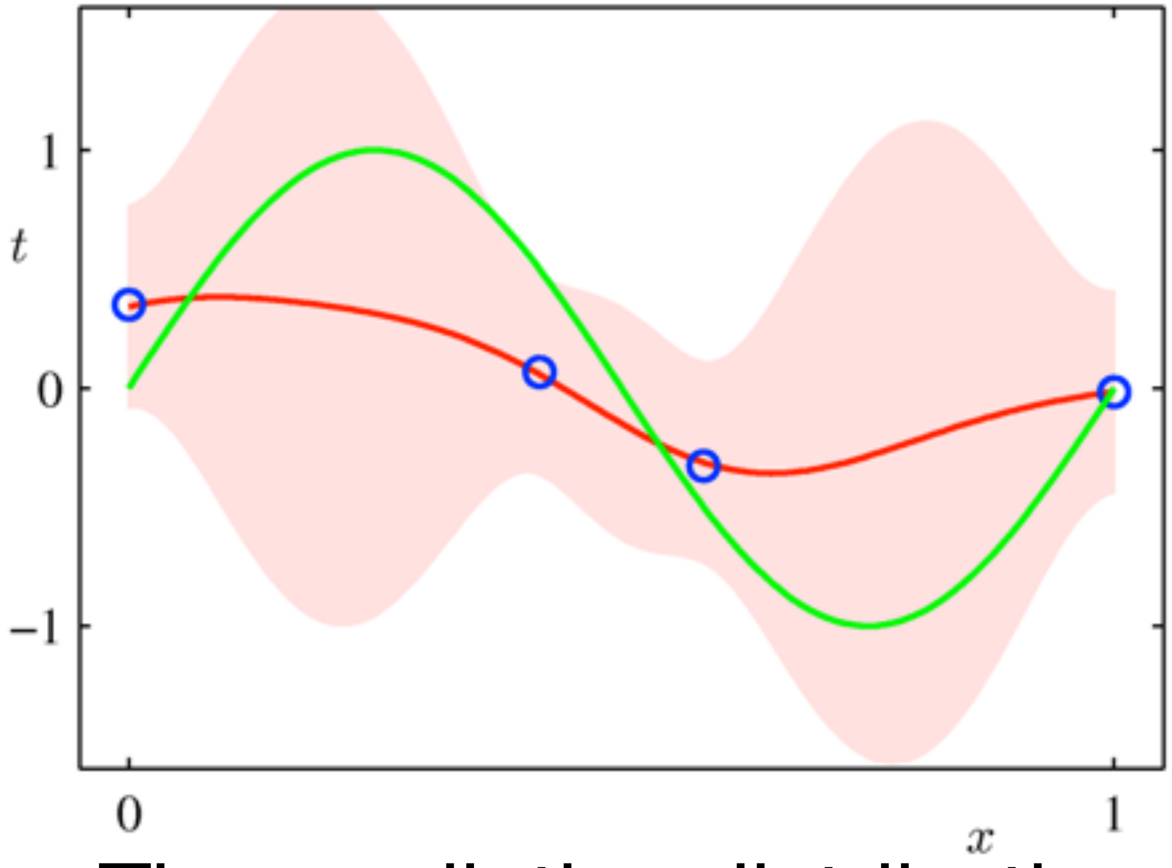


Some samples from the posterior

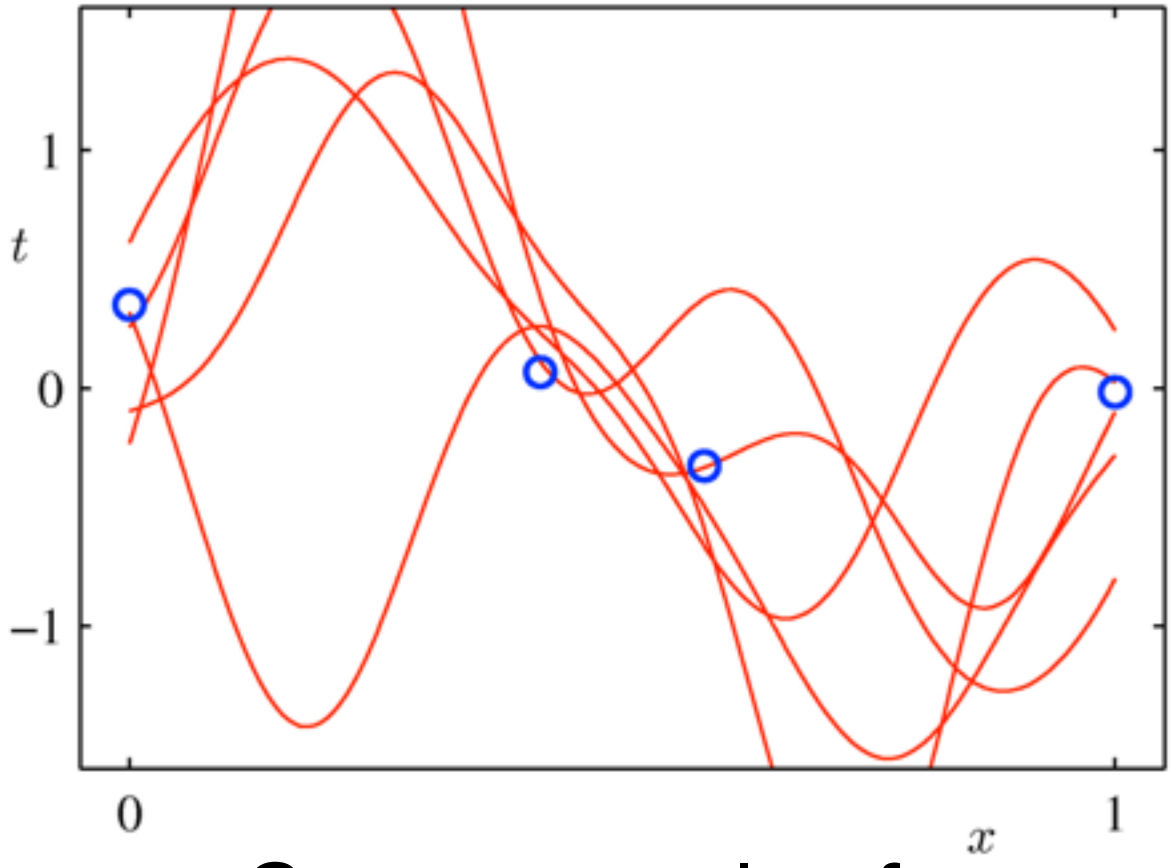
From: C.M. Bishop

Predictive Distribution (4)

- Example: Sinusoidal data, 9 Gaussian basis functions, 4 data points



The predictive distribution

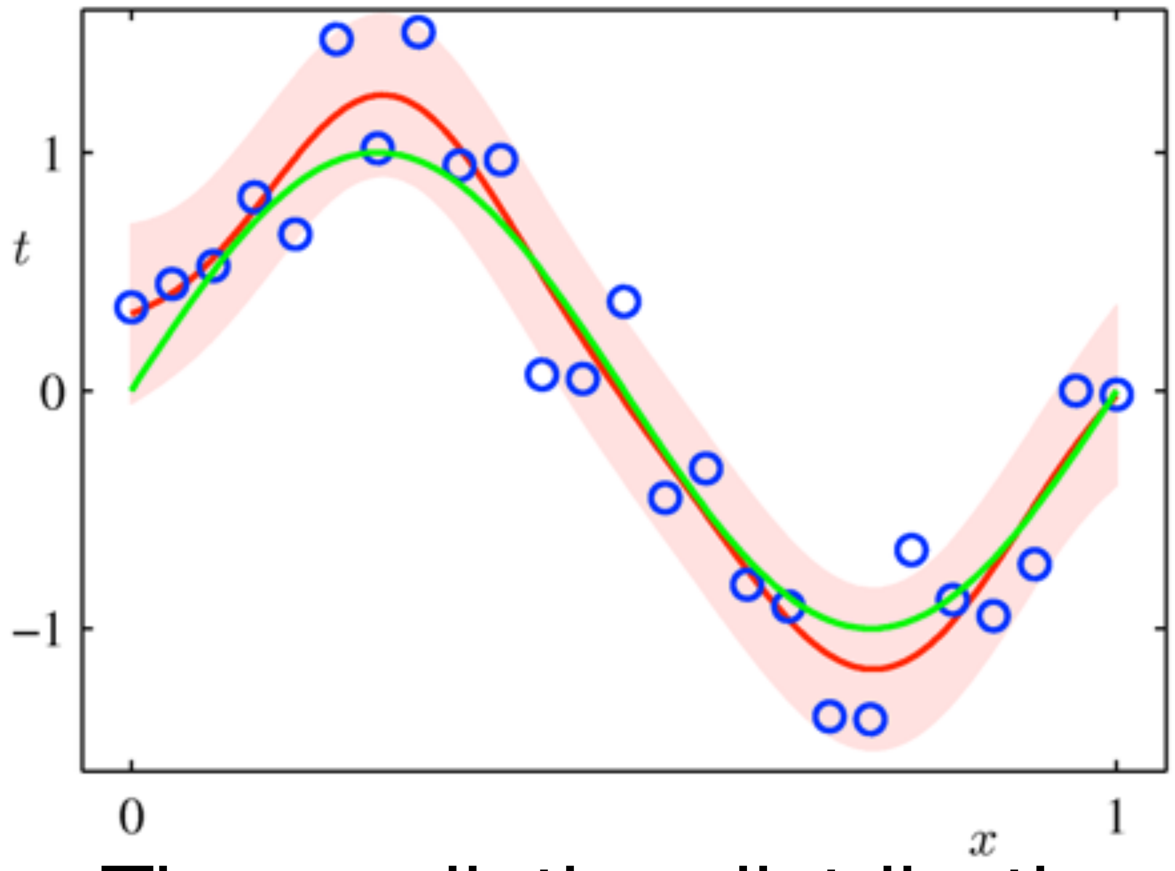


Some samples from the posterior

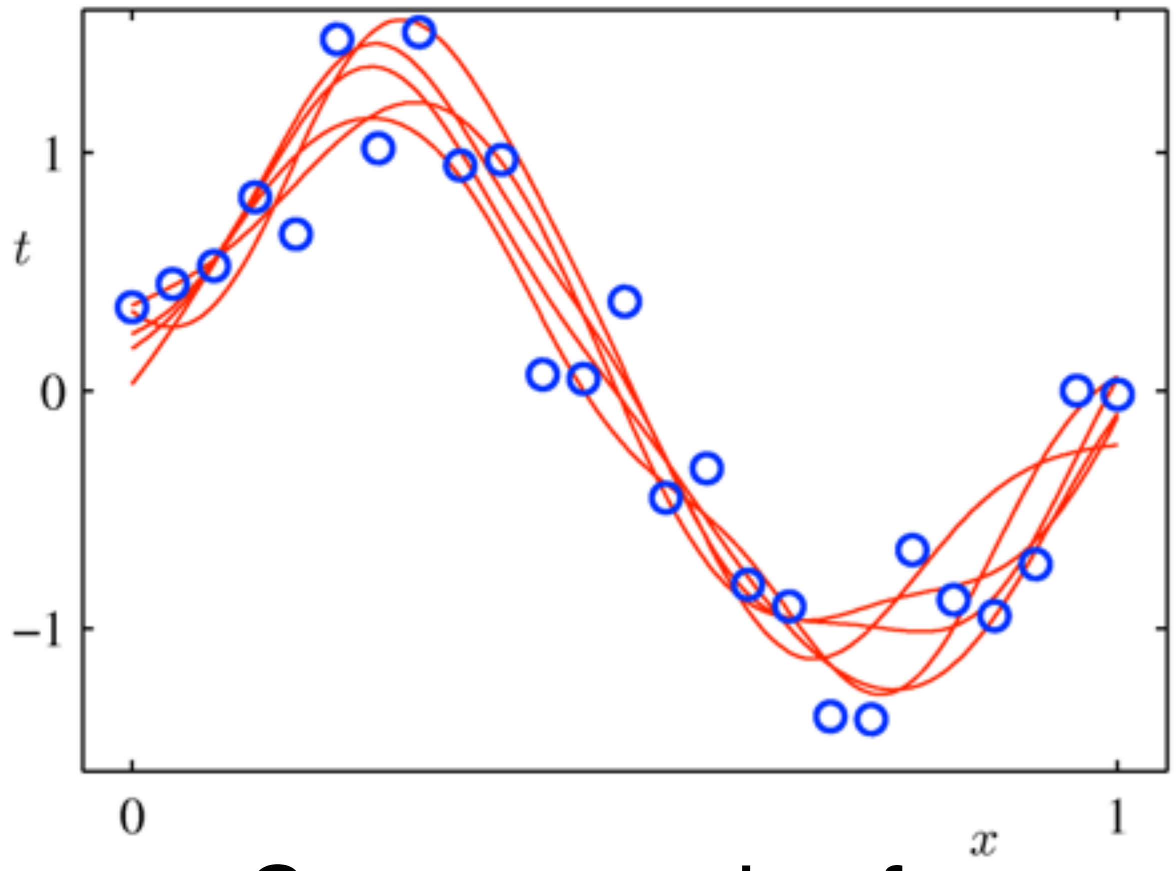
From: C.M. Bishop

Predictive Distribution (5)

- Example: Sinusoidal data, 9 Gaussian basis functions, 25 data points



The predictive distribution



Some samples from the posterior

From: C.M. Bishop

Summary

- Regression can be expressed as a **least-squares** problem
- To avoid overfitting, we need to introduce a **regularisation term** with an additional parameter λ
- Regression **without** regularisation is equivalent to Maximum Likelihood Estimation
- Regression **with** regularisation is Maximum A-Posteriori
- When using Gaussian priors (and Gaussian noise), all computations can be done **analytically**
- This gives a closed form of the **parameter posterior** and the **predictive distribution**

3. Probabilistic Graphical Models
Directed Models

The Bayes Filter (Rep.)

$$\mathrm{Bel}(x_t) = p(x_t \mid u_1, z_1, \dots, u_t, z_t)$$

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

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

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

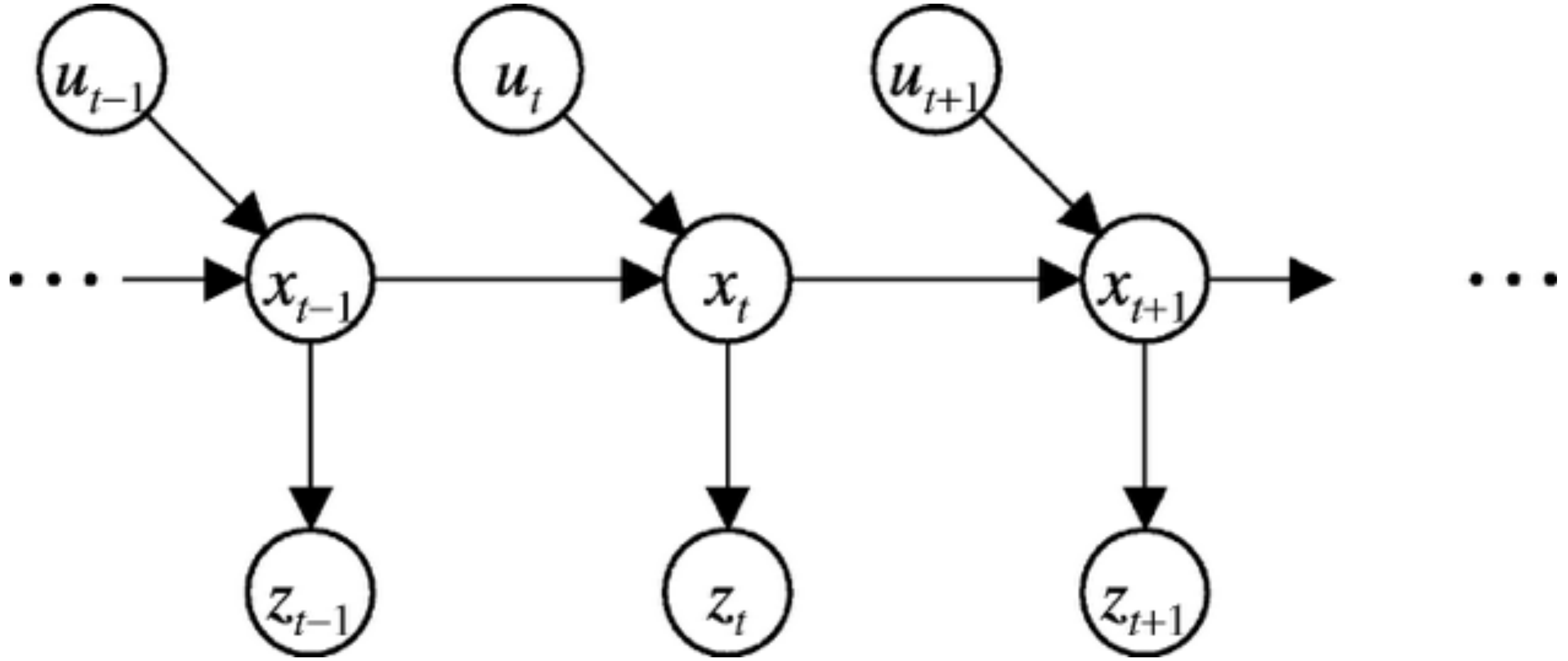
(Markov) $$= \eta\, p(z_t \mid x_t) \int p(x_t \mid u_t, x_{t-1}) p(x_{t-1} \mid u_1, z_1, \dots, u_t) dx_{t-1}$$

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

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

Graphical Representation (Rep.)

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



- This incorporates the following Markov assumptions:

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

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

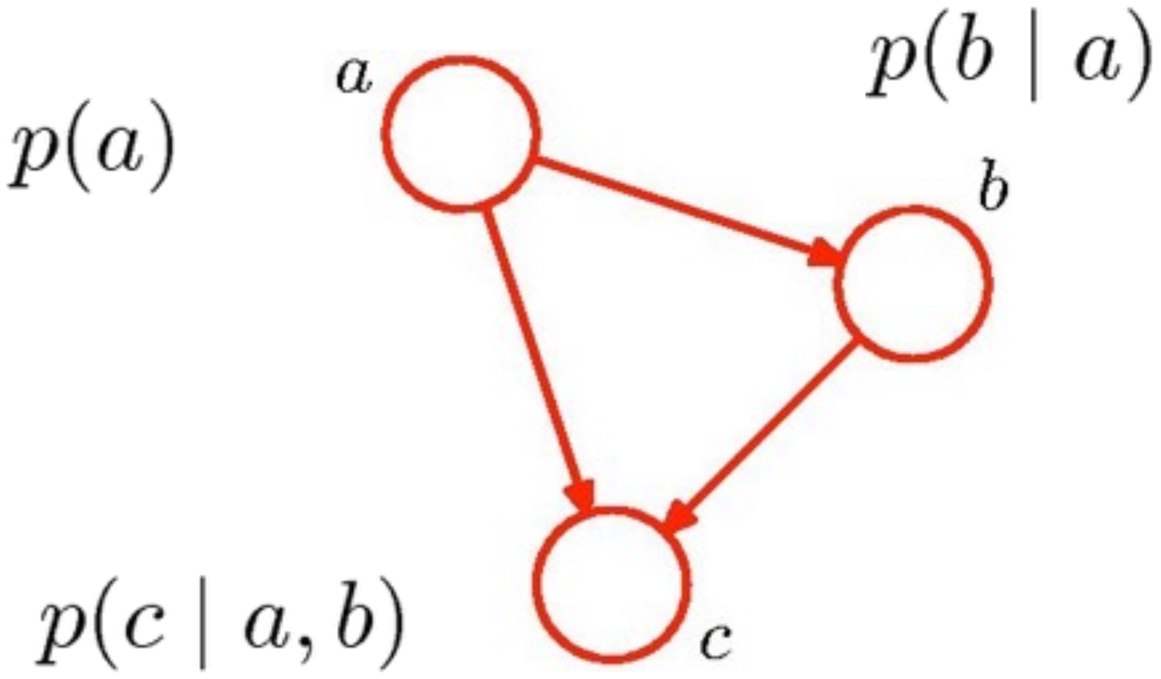
Definition

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:

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

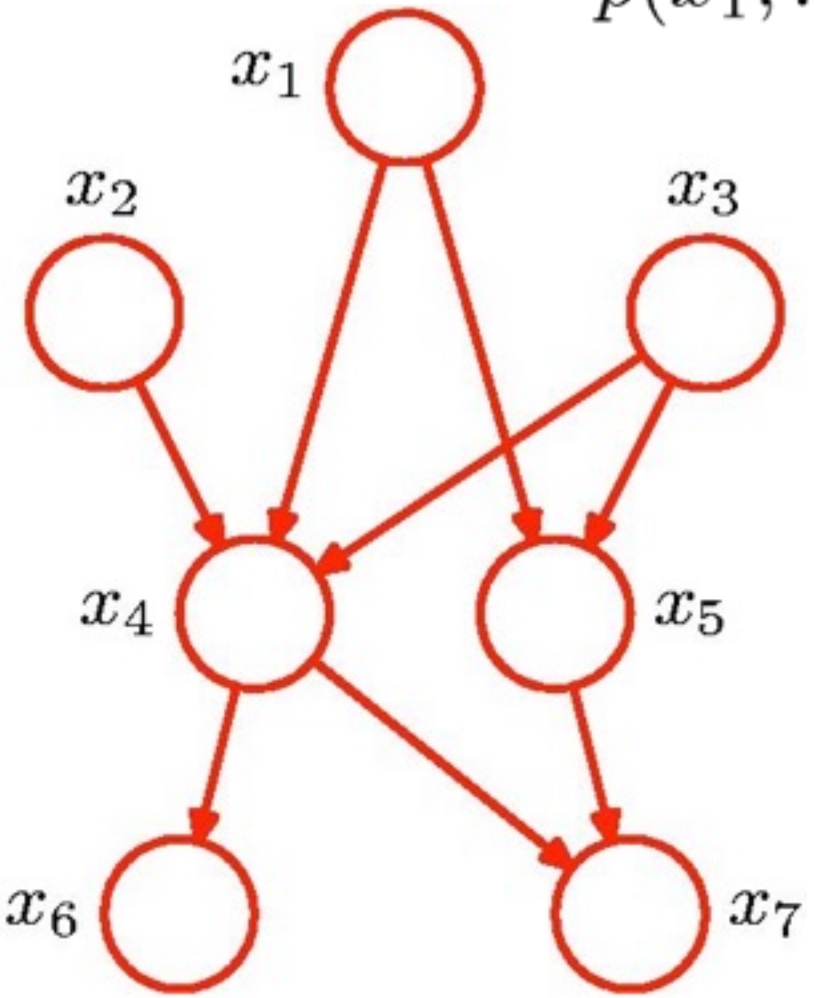
$$p(a, b, c) = p(b|a, c)p(a|c)p(c)$$

$$\vdots$$

Bayesian Networks

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

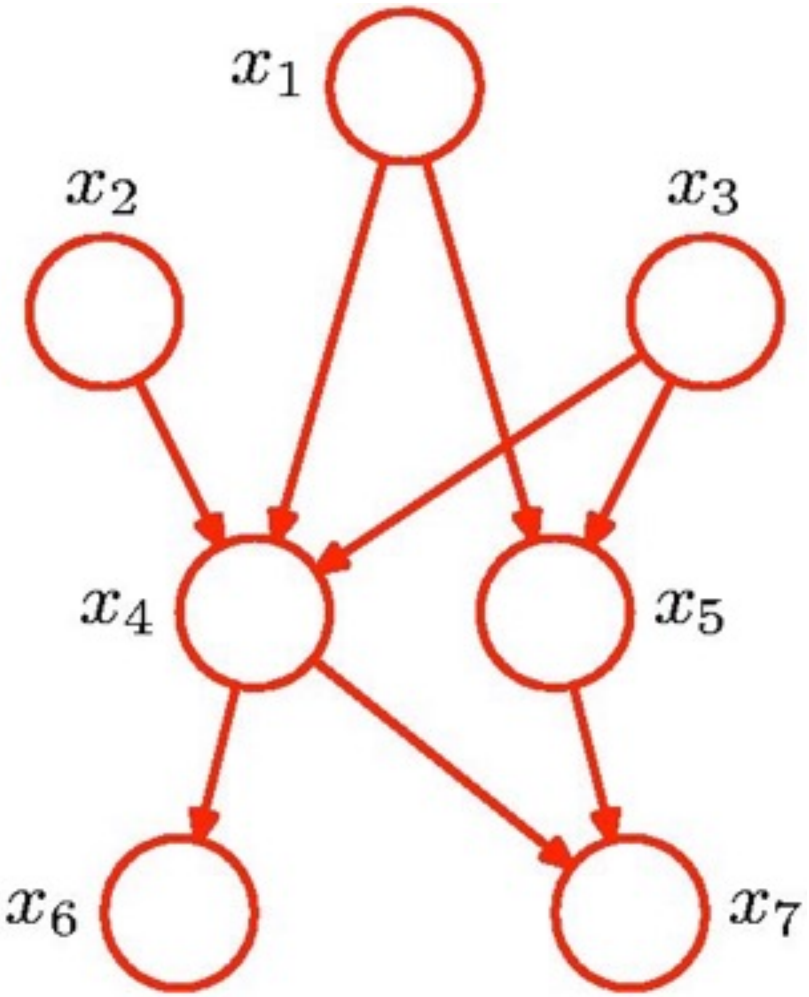
$$p(x_1, \ldots, x_7) = p(x_1)p(x_2)p(x_3)p(x_4|x_1, x_2, x_3)$$
$$p(x_5|x_1, x_3)p(x_6|x_4)p(x_7|x_4, x_5)$$



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

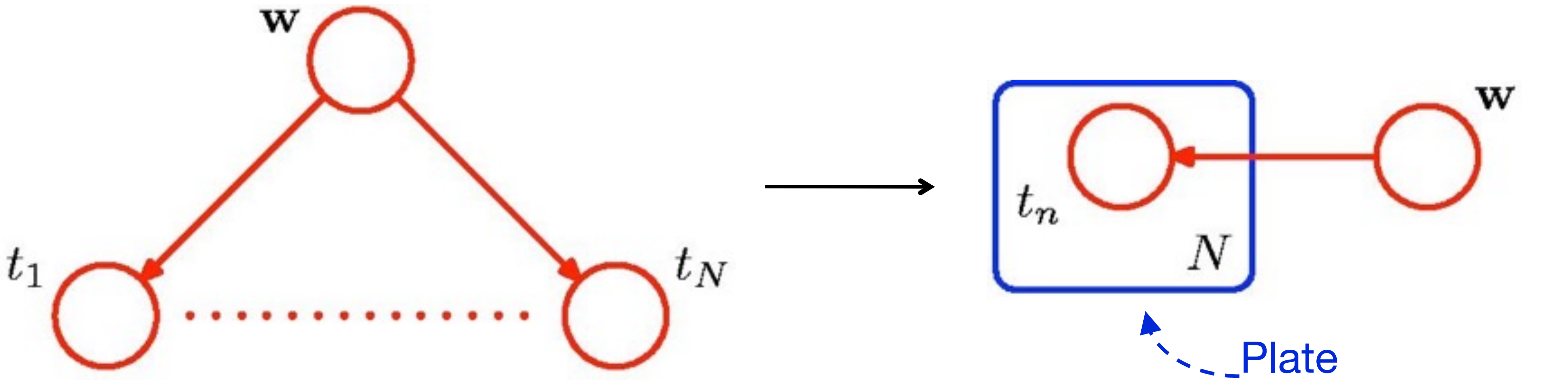
and

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

Elements of Graphical Models

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

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

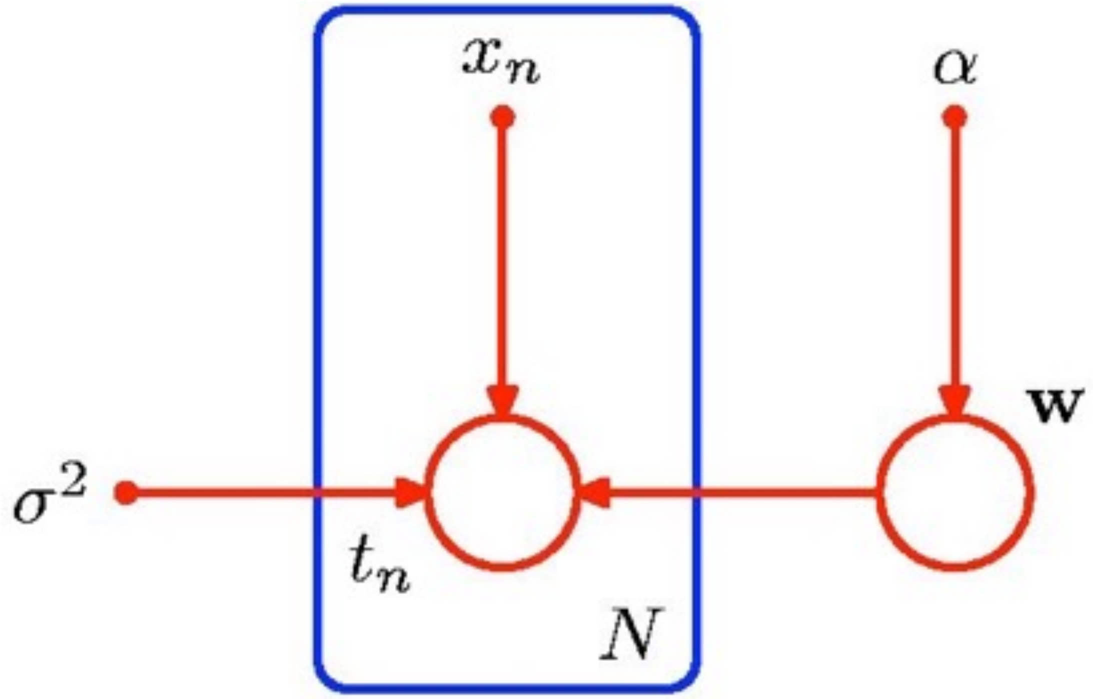


Plate

Elements of Graphical Models (2)

We distinguish between **input** variables and explicit **hyper-parameters**:

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

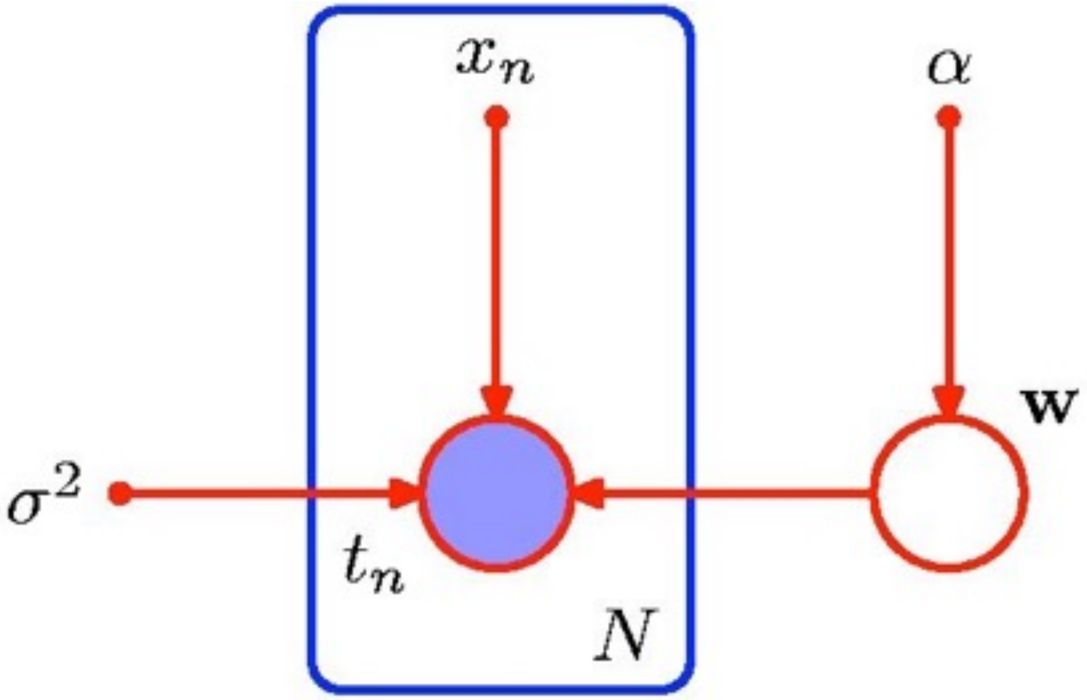


Elements of Graphical Models (3)

We distinguish between **observed** variables and **hidden** variables:

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

(deterministic parameters omitted in formula)



Example: Regression as a Graphical Model

Aim: Find a general expression to compute the predictive distribution: $p(\hat{t} \mid \hat{x}, \mathbf{x}, \mathbf{t})$

Notation:

$\hat{t} = t^*$

Bishop vs. Rasmussen

This expression should

- model all conditional independencies
- explicitly incorporate all parameters (also the deterministic ones)

Example: Regression as a Graphical Model

Aim: Find a general expression to compute the predictive distribution: $p(\hat{t} \mid \hat{x}, \mathbf{x}, \mathbf{t})$

Notation:

$\hat{t} = t^*$

Bishop vs. Rasmussen

This expression should

- model all conditional independencies
- explicitly incorporate all parameters (also the deterministic ones)

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

$$= \int \frac{p(\hat{t}, \mathbf{w}, \mathbf{t} \mid \hat{x}, \mathbf{x}, \alpha, \sigma^2)}{p(\mathbf{t} \mid \hat{x}, \mathbf{x}, \alpha, \sigma^2)} d\mathbf{w} \quad \propto \int p(\hat{t}, \mathbf{w}, \mathbf{t} \mid \hat{x}, \mathbf{x}, \alpha, \sigma^2) d\mathbf{w}$$

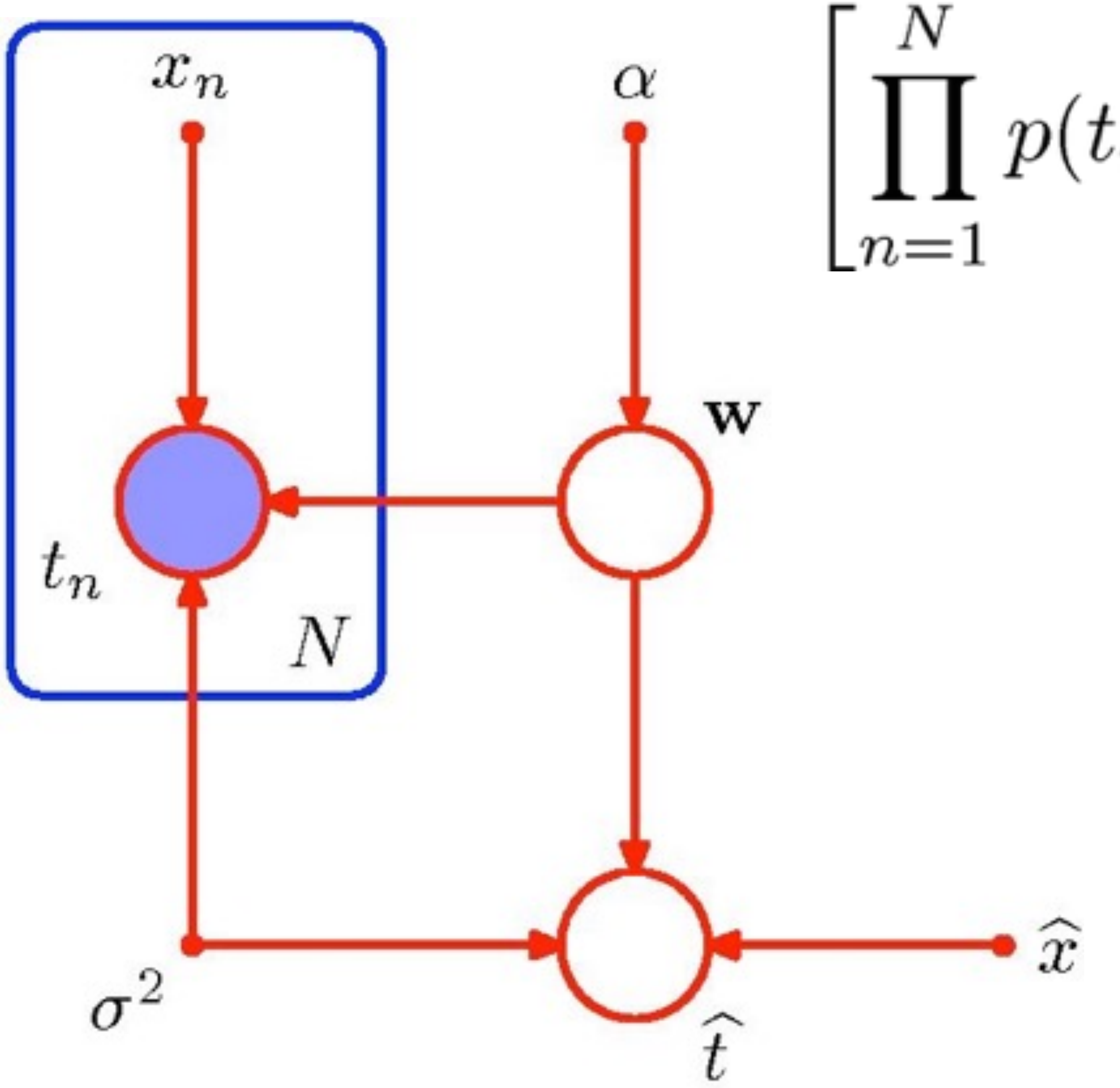
Regression as a Graphical Model

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

Notation:
$\hat{t} = t^*$

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

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



Here: conditioning on all deterministic parameters

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

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

Example: Discrete Variables

- Two dependent variables: K^2 - 1 parameters

Here: $K = 2$

x_1	$p(x_1)$
1	0.2
~~2~~	~~0.8~~

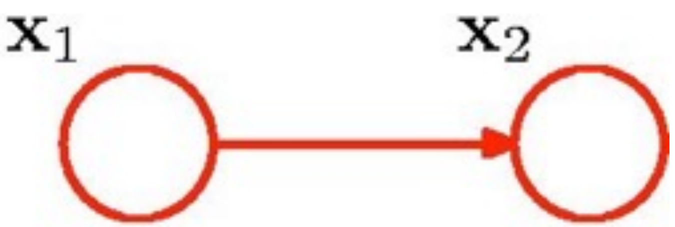
$K-1$

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~~

$K-1$

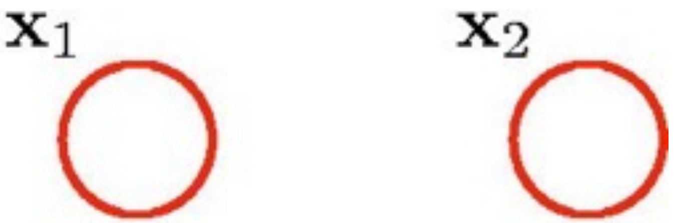
$K-1$

$K(K-1)$



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

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

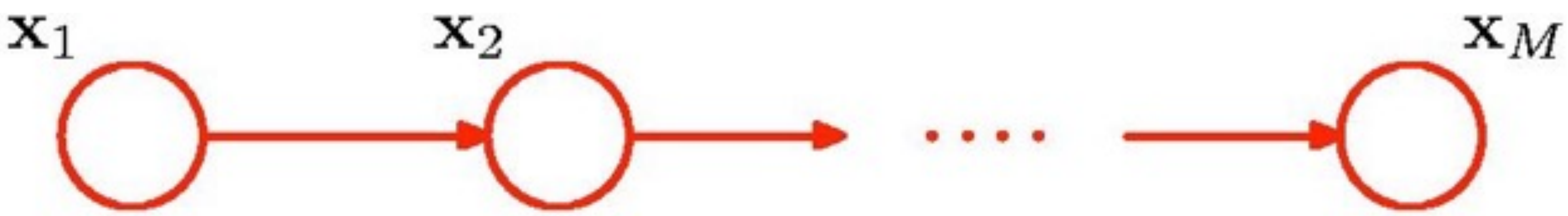


$$K-1+K-1=2(K-1)$$

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:



then 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** with the number of variables.

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:

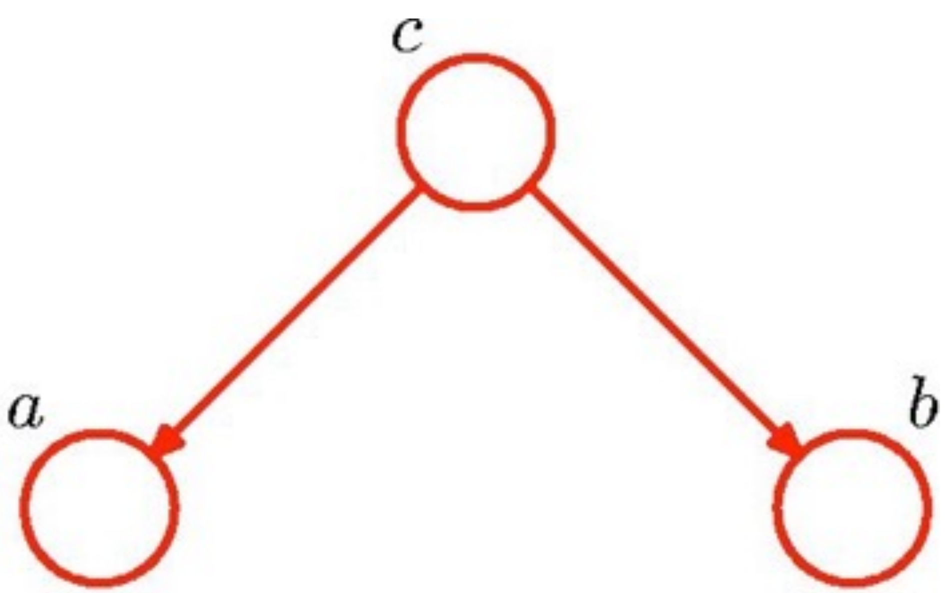
$$p(x, y \mid z) = p(x \mid z)p(y \mid z)$$

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

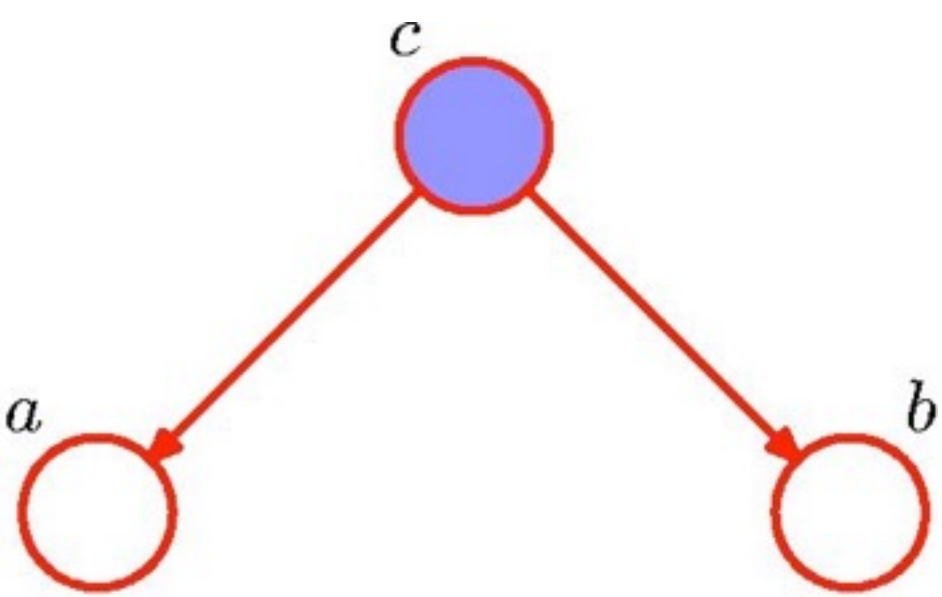
$$p(a,b) = \sum_c p(a|c)p(b|c)p(c)$$

This is in general not equal to $p(a)p(b)$.

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

Conditional Independence: Example 1

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

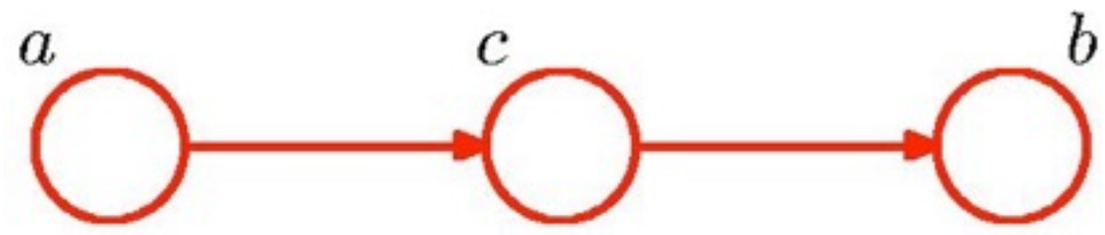


$$
\begin{aligned}
p(a, b|c) &= \frac{p(a, b, c)}{p(c)} \\
&= p(a|c)p(b|c)
\end{aligned}
$$

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:

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

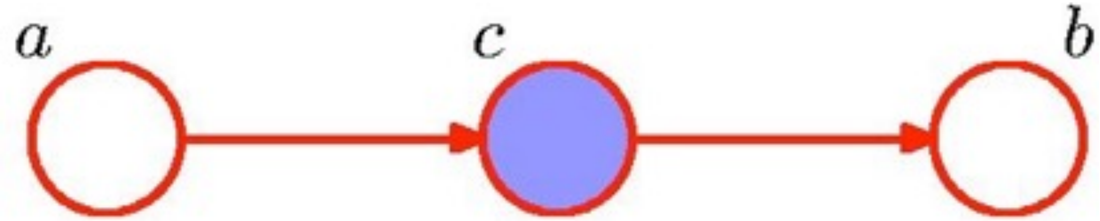
$$= p(a)\sum_c \frac{p(c,a)p(b,c,a)}{p(a)p(c,a)} = p(a)\sum_c p(b,c\mid a)$$

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

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

Conditional Independence: Example 2

As before, now we condition on c :



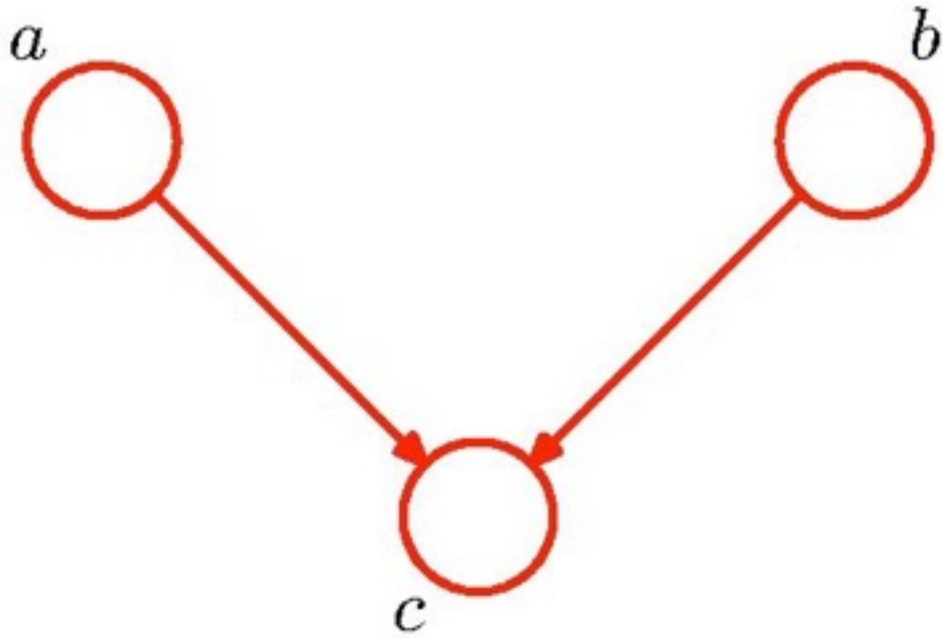
$$\begin{aligned} p(a,b|c) &= \frac{p(a,b,c)}{p(c)} \\ &= \frac{p(a)p(c|a)p(b|c)}{p(c)} \\ &= p(a|c)p(b|c) \end{aligned}$$

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:

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

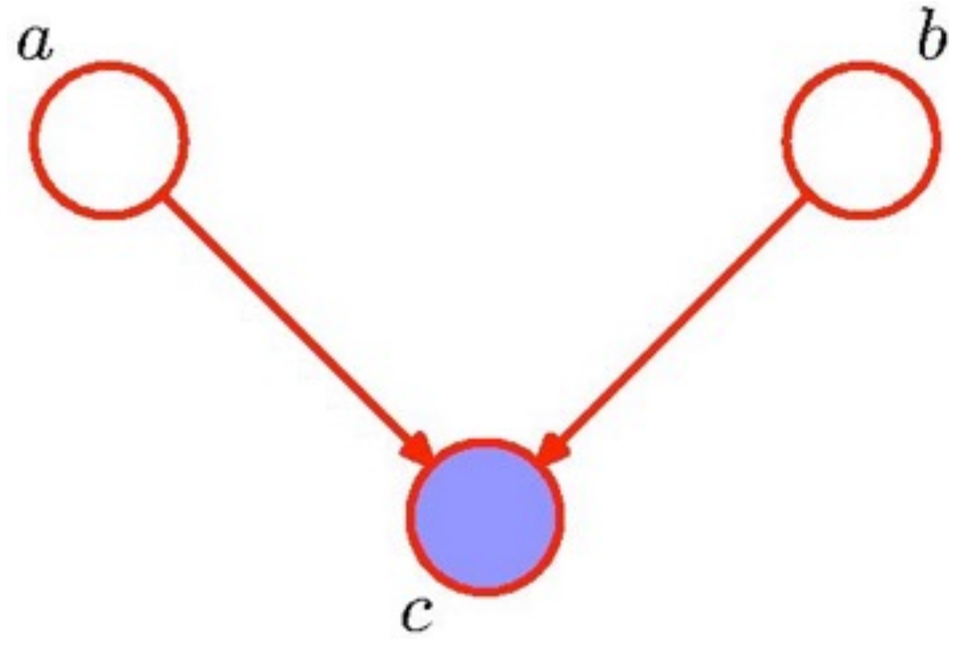
we obtain:

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

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

Conditional Independence: Example 3

Again, we condition on c



$$
\begin{aligned}
p(a, b|c) &= \frac{p(a, b, c)}{p(c)} \\
&= \frac{p(a)p(b)p(c|a, b)}{p(c)}
\end{aligned}
$$

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: $\mathrm{dsep}(A, B|C)$

D-Separation

Say: A, B, and C are non-intersecting subsets of nodes in a directed graph.

- A path … ntains a nod…

a) the a… tail-to-tail at t…

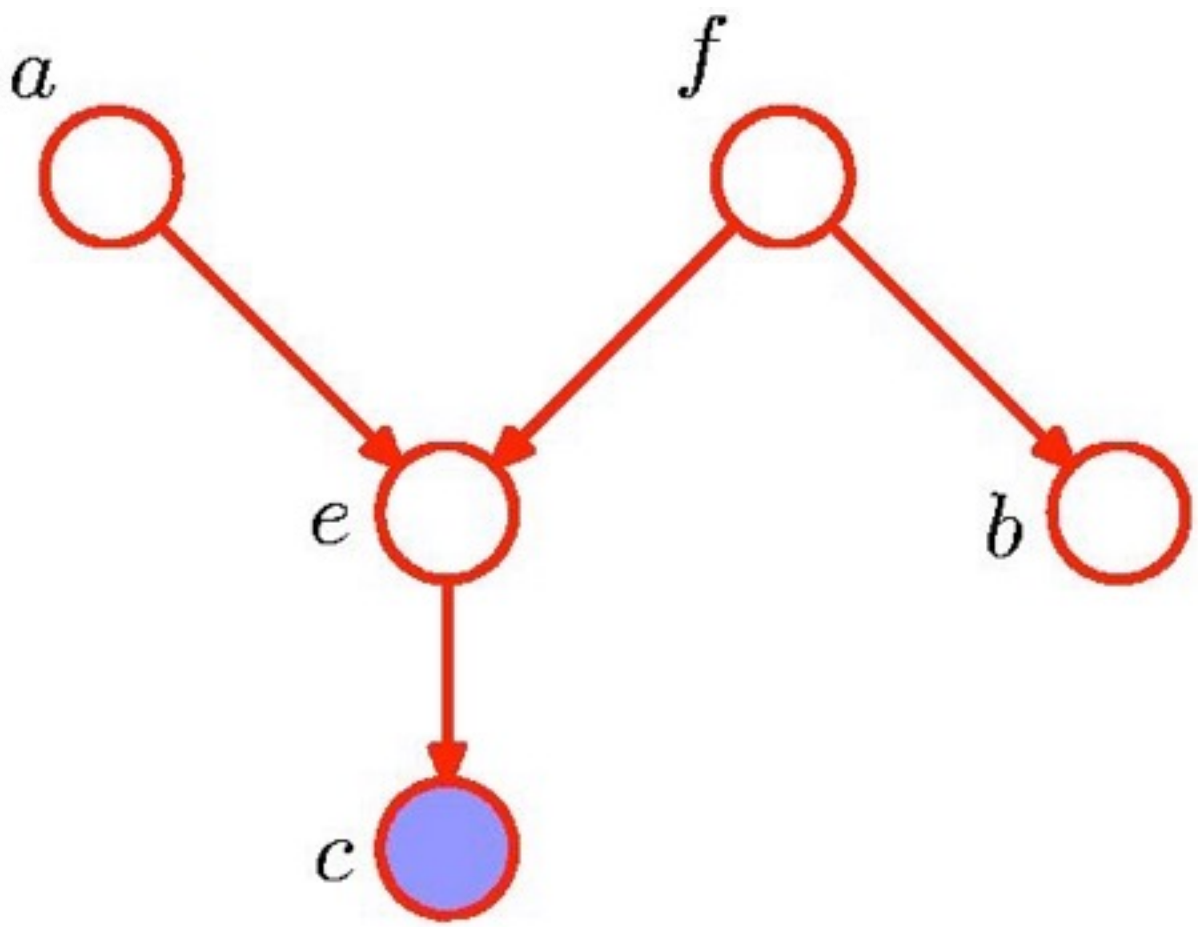
b) the a… neither the nod… C.

- If all p… aid to be **d-separated** from B by C.

Notation: $\mathrm{dsep}(A, B|C)$

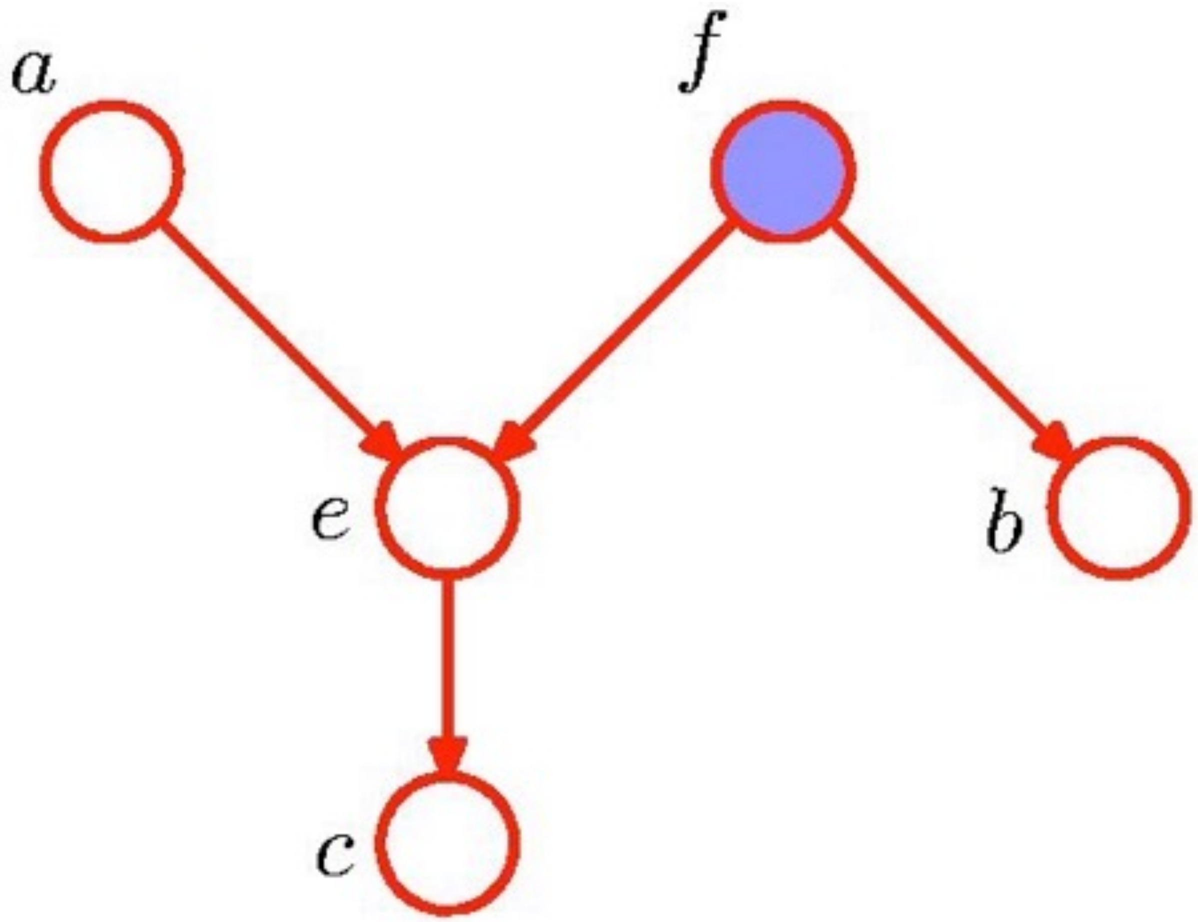
D-Separation is a property of graphs and not of probability distributions

D-Separation: Example



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

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



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

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.

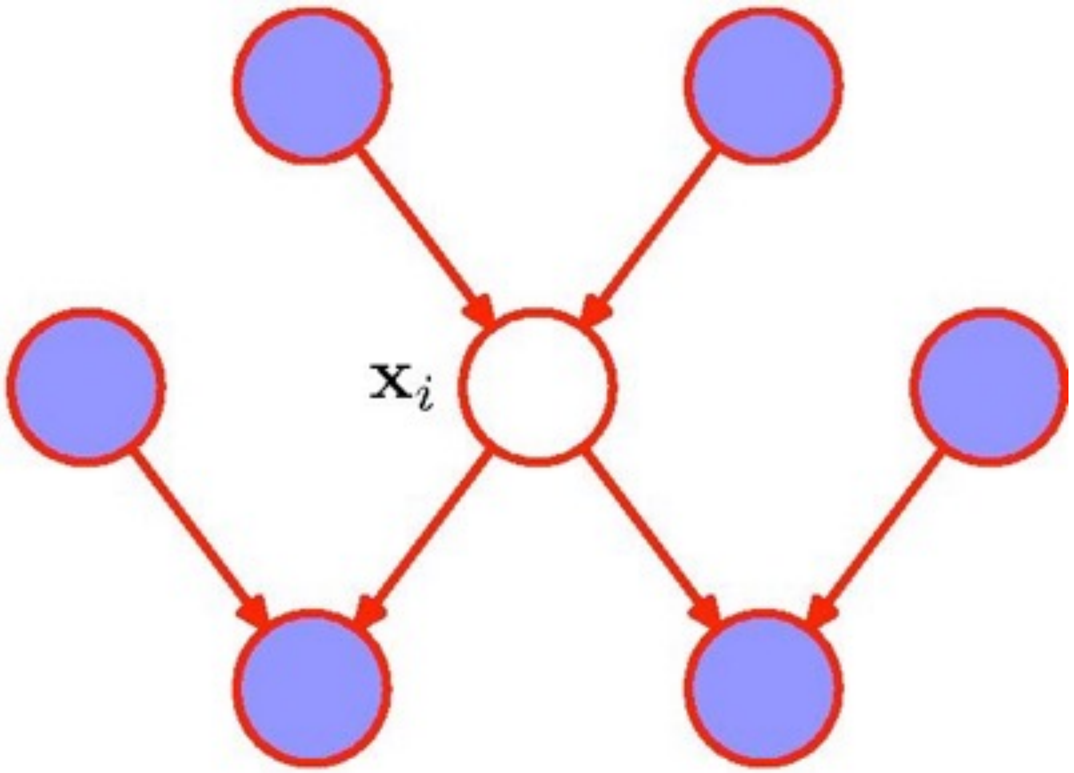
$$\forall A, B, C : A \perp\!\!\!\perp B \mid C \Leftrightarrow \mathrm{dsep}(A, B, C)$$

A perfect map uniquely defines a probability distribution.

The Markov Blanket

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

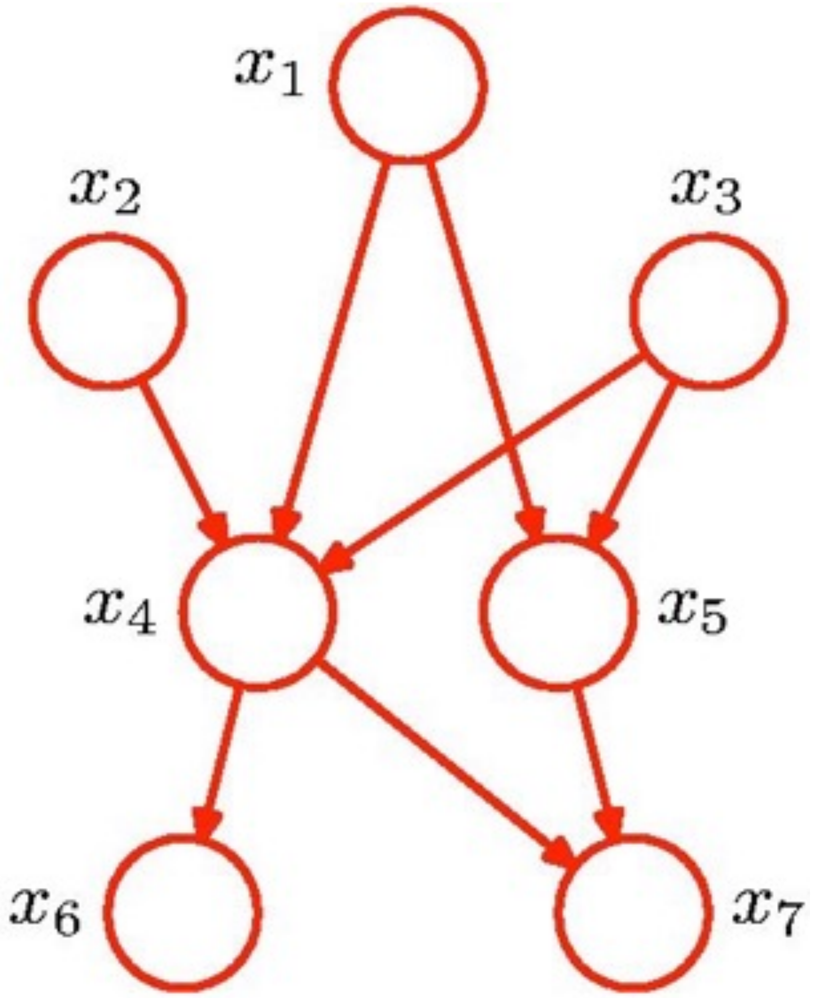
$$p(\mathbf{x}_i|\mathbf{x}_{\{j\neq i\}}) = \frac{p(\mathbf{x}_1,\ldots,\mathbf{x}_M)}{\int p(\mathbf{x}_1,\ldots,\mathbf{x}_M)d\mathbf{x}_i} = \frac{\prod_k p(\mathbf{x}_k|\mathrm{pa}_k)}{\int \prod_k p(\mathbf{x}_k|\mathrm{pa}_k)d\mathbf{x}_i} = p(\mathbf{x}_i \mid \mathbf{x}_{\mathcal{M}_i})$$



Factors independent of $\mathbf{x}_i$ cancel between numerator and denominator.

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

Repetition: Bayesian Networks

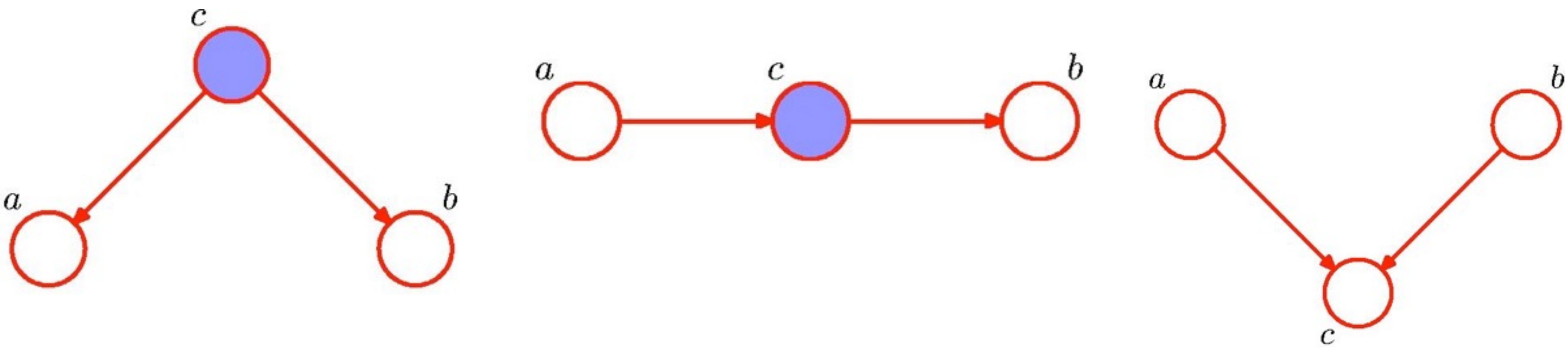


Directed graphical models can be used to represent **probability distributions**

This is useful to do **inference** and to **generate samples** from the distribution efficiently

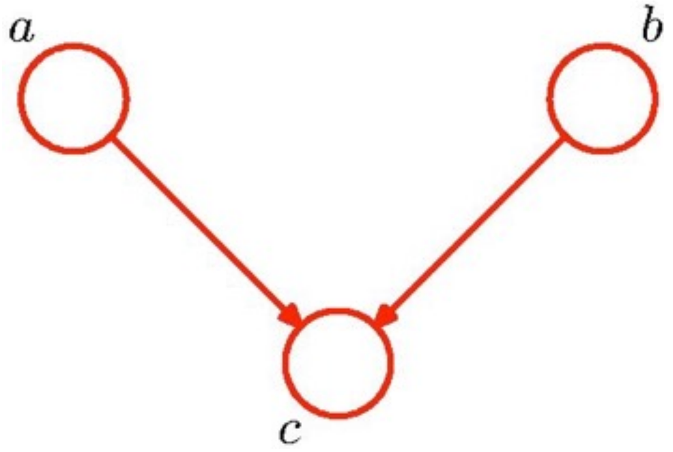
$$
\begin{aligned}
p(x_1,\ldots,x_7) \quad = \quad & p(x_1)p(x_2)p(x_3)p(x_4|x_1,x_2,x_3) \\
& p(x_5|x_1,x_3)p(x_6|x_4)p(x_7|x_4,x_5)
\end{aligned}
$$

Repetition: D-Separation



- D-separation is a property of graphs that can be easily determined
- An I-map assigns every d-separation a c.i. rel
- A D-map assigns every c.i. rel a d-separation
- Every Bayes net determines a unique prob. dist.

In-depth: The Head-to-Head Node



$p(a) = 0.9 \qquad p(b) = 0.9$

a	*b*	*p(c)*
1	1	0.8
1	0	0.2
0	1	0.2
0	0	0.1

Example:

a: Battery charged (0 or 1)

b: Fuel tank full (0 or 1)

c: Fuel gauge says full (0 or 1)

We can compute $p(\neg c) = 0.315$

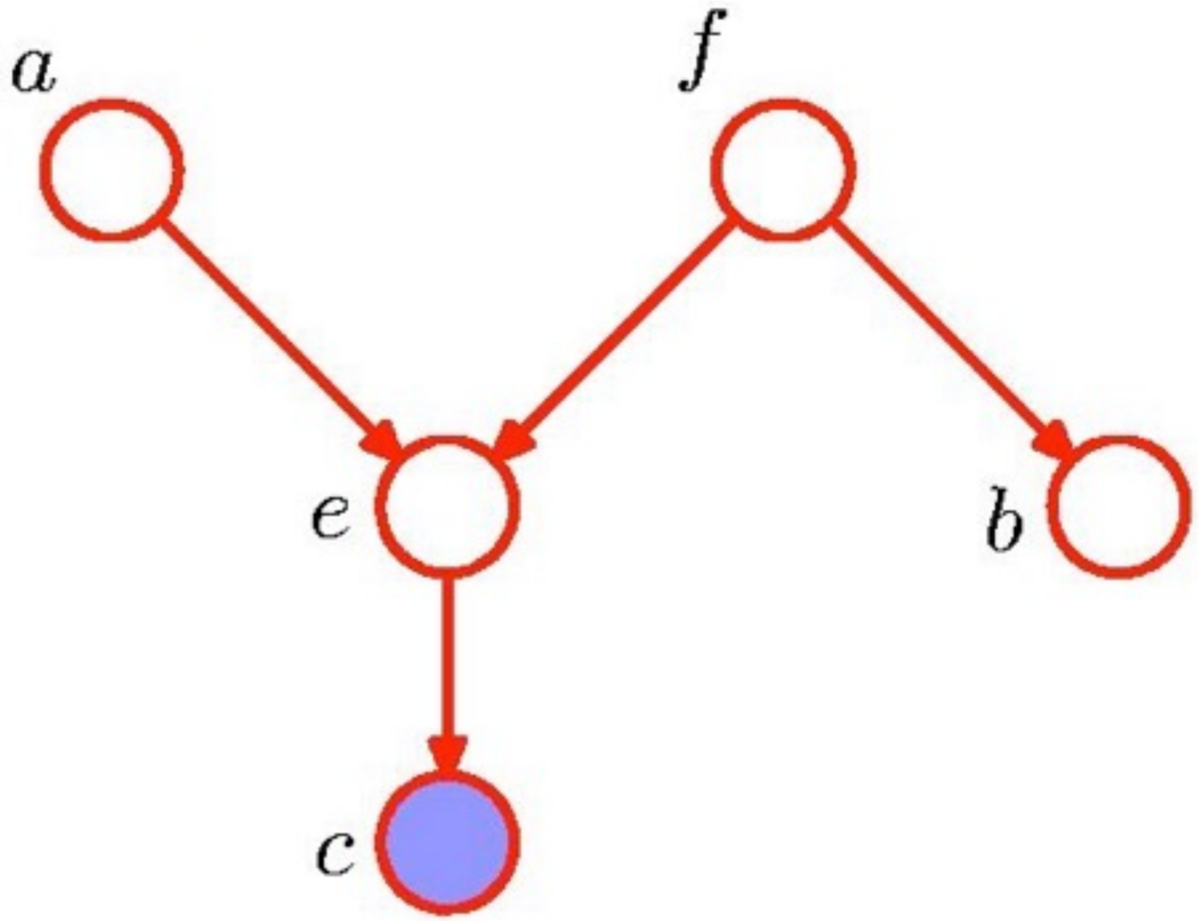
and $p(\neg c \mid \neg b) = 0.81$

and obtain $p(\neg b \mid \neg c) \approx 0.257$

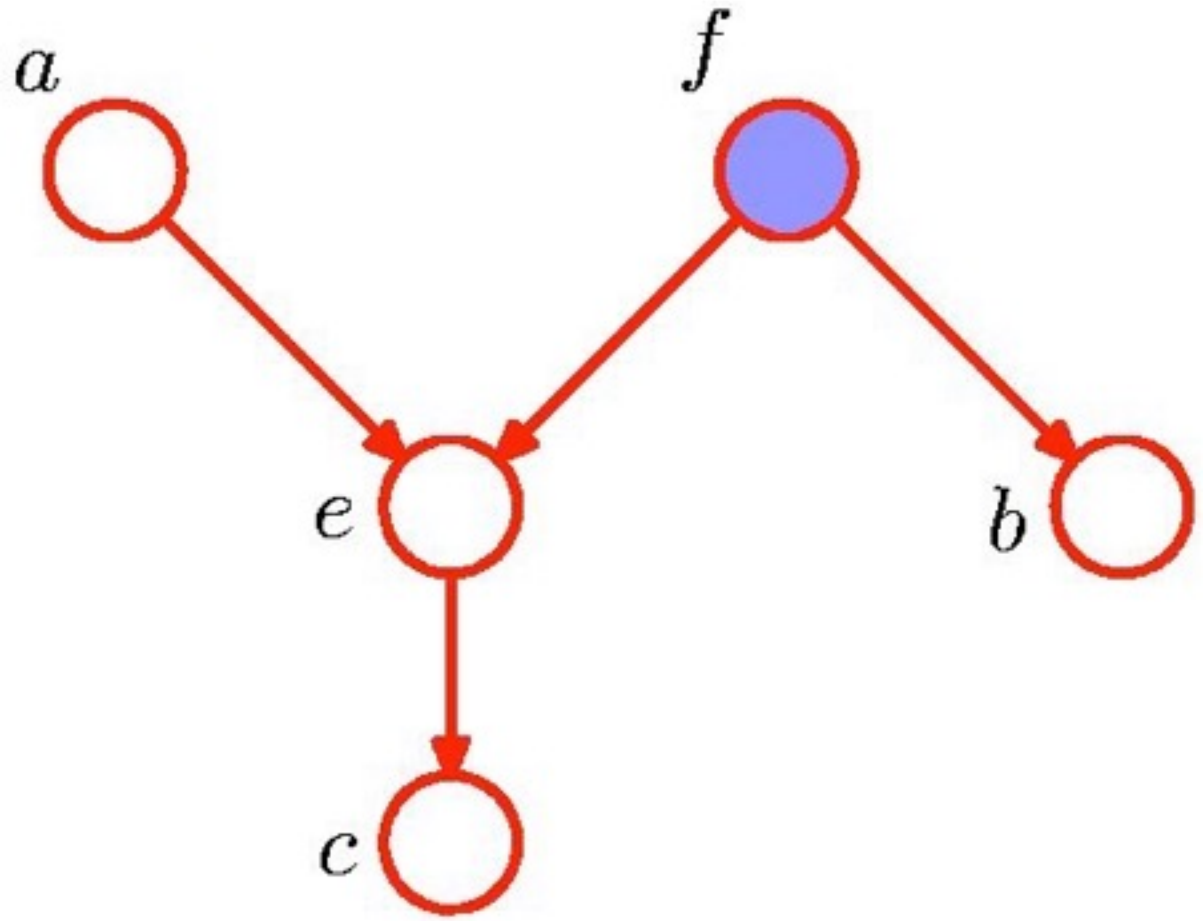
similarly: $p(\neg b \mid \neg c, \neg a) \approx 0.111$

“a **explains** c **away**”

Repetition: D-Separation



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



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

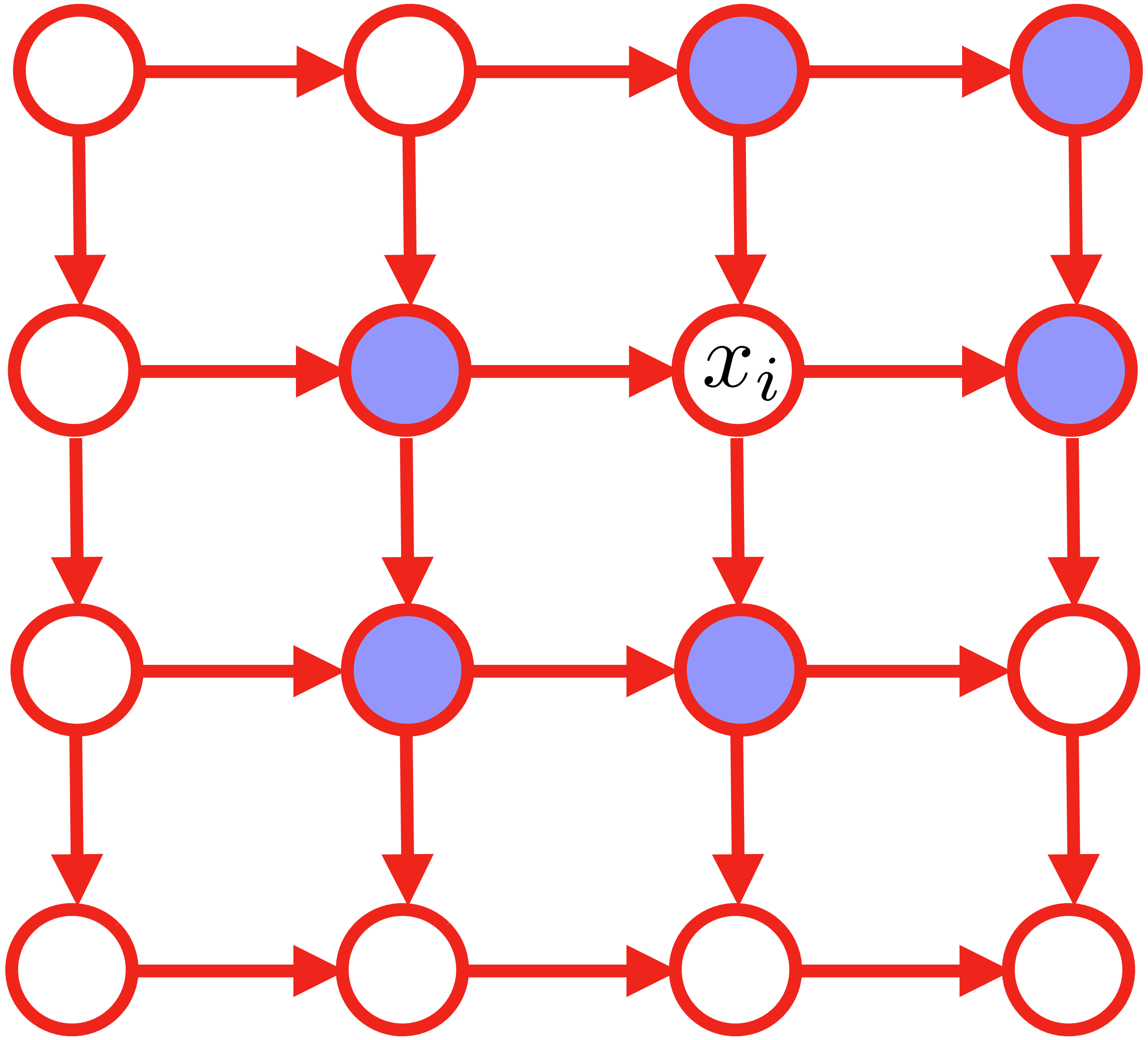
Directed vs. Undirected Graphs

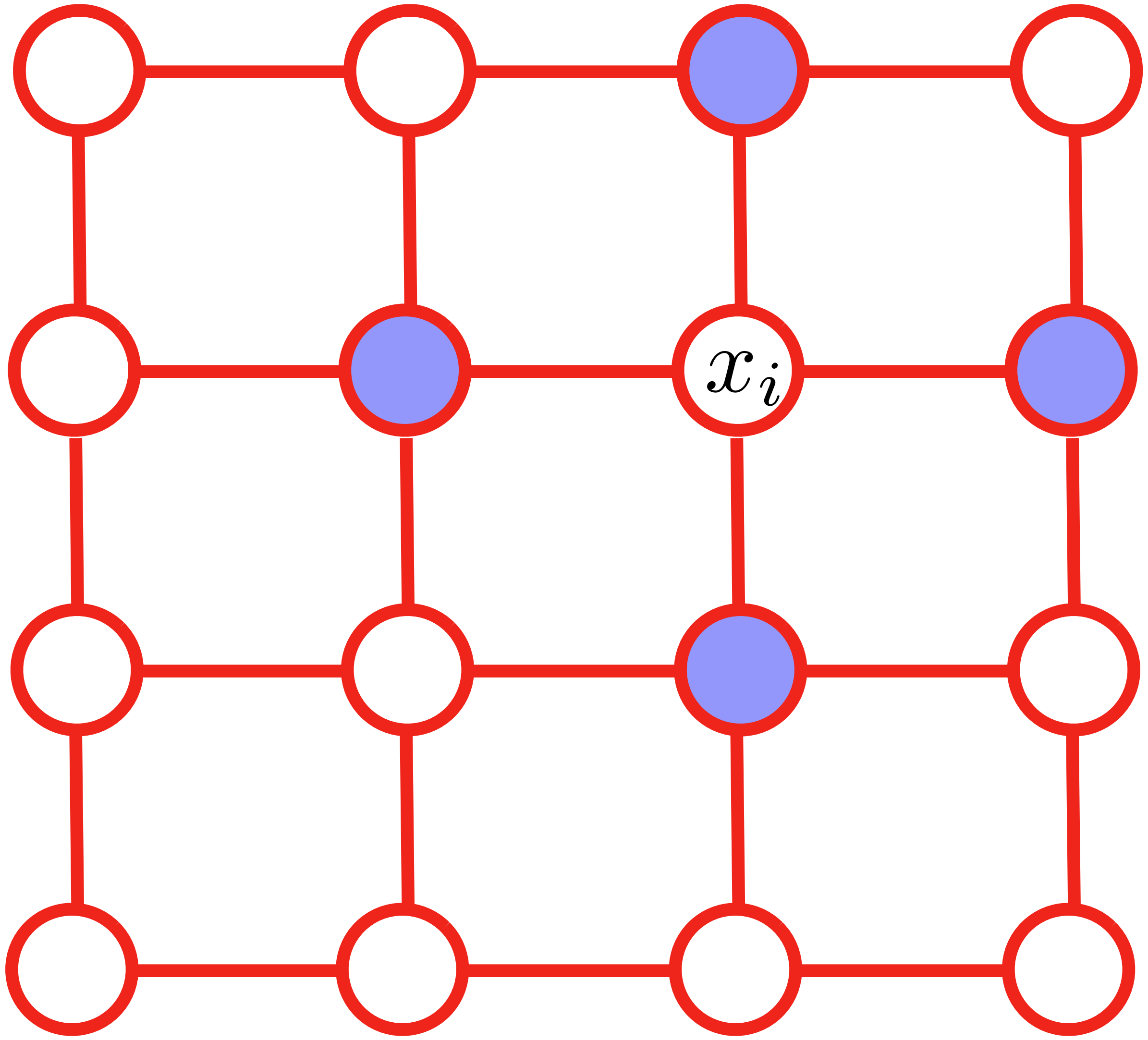
Using D-separation we can identify conditional independencies in directed graphical models, but:

- Is there a simpler, more intuitive way to express conditional independence in a graph?
- Can we find a representation for cases where an „ordering“ of the random variables is inappropriate (e.g. the pixels in a camera image)?

Yes, we can: by removing the directions of the edges we obtain an Undirected Graphical Model, also known as a **Markov Random Field**

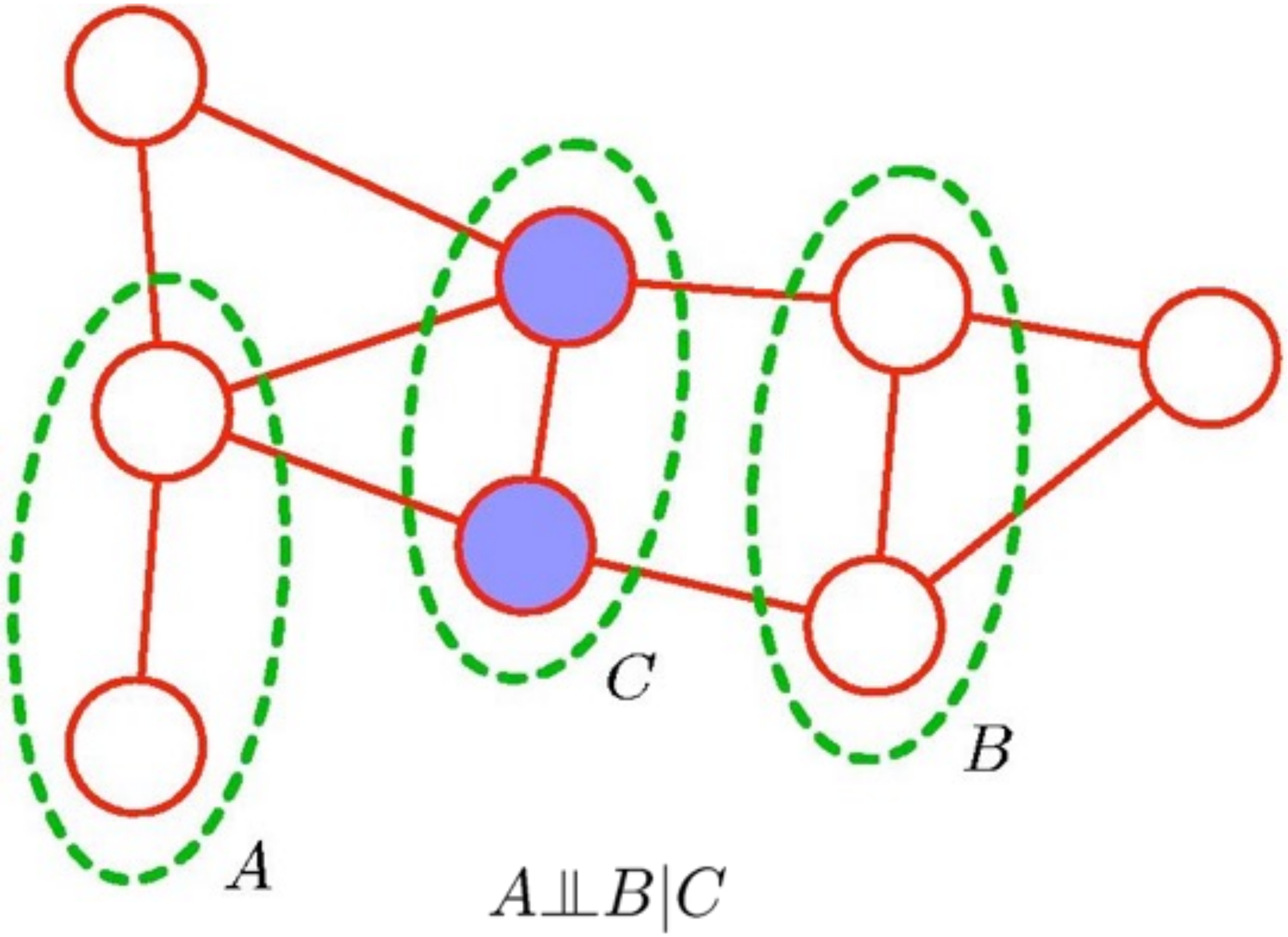
Example: Camera Image



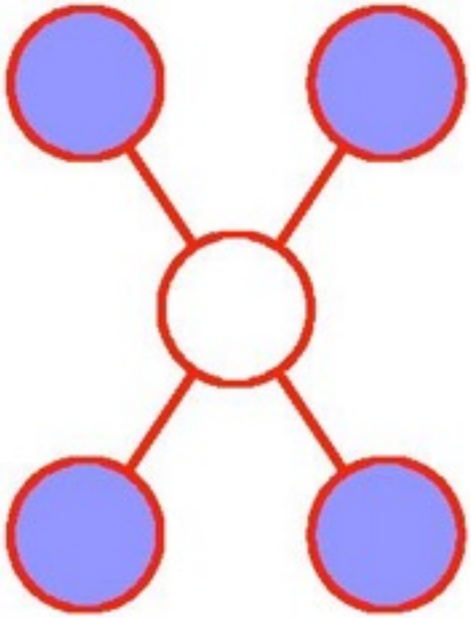


- directions are counter-intuitive for images
- Markov blanket is not just the direct neighbors when using a directed model

Markov Random Fields



Markov Blanket



All paths from A to B go through C, i.e. C blocks all paths.

We only need to condition on the **direct neighbors** of $\mathbf{x}$ to get c.i., because these already block every path from $\mathbf{x}$ to any other node.

Factorization of MRFs

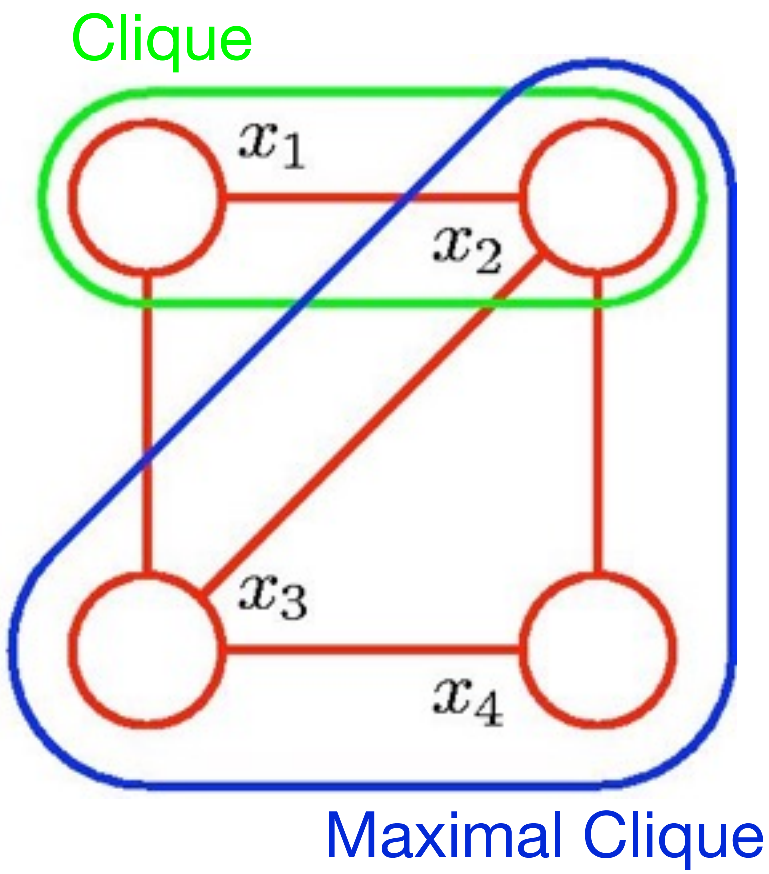
Any two nodes x_i and x_j that are not connected in an MRF are conditionally independent given all other nodes:

$$p(x_i, x_j \mid \mathbf{x}_{\setminus\{i,j\}}) = p(x_i \mid \mathbf{x}_{\setminus\{i,j\}}) p(x_j \mid \mathbf{x}_{\setminus\{i,j\}})$$

In turn: each factor contains only nodes that are connected

This motivates the consideration of cliques in the graph:

- A **clique** is a fully connected subgraph.
- A **maximal** clique can not be extended with another node without loosing the property of full connectivity.



Factorization of MRFs

In general, a Markov Random Field is factorized as

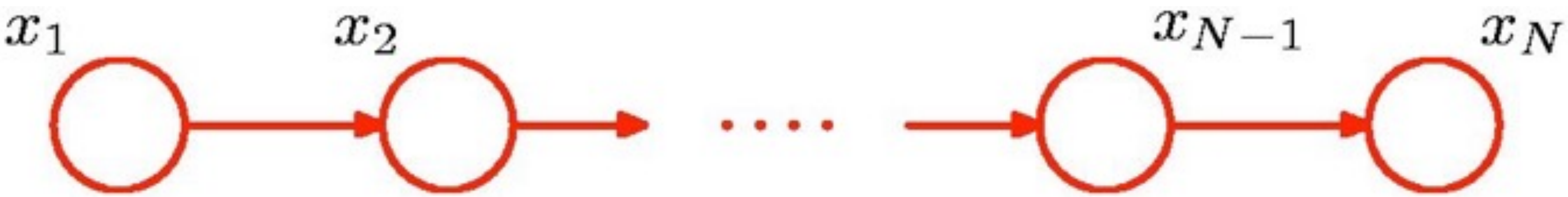
$$p(\mathbf{x}) = \frac{\prod_C \phi_C(\mathbf{x}_C)}{\sum_{\mathbf{x}'} \prod_C \phi_C(\mathbf{x}'_C)} = \frac{1}{Z} \prod_C \phi_C(\mathbf{x}_C) \qquad (4.1)$$

where C is the set of all (maximal) cliques and Φ_C is a positive function of a given clique $\mathbf{x}_C$ of nodes, called the **clique potential**. Z is called the **partition function**.

Theorem (Hammersley/Clifford): Any undirected model with associated clique potentials Φ_C is a perfect map for the probability distribution defined by Equation (4.1).

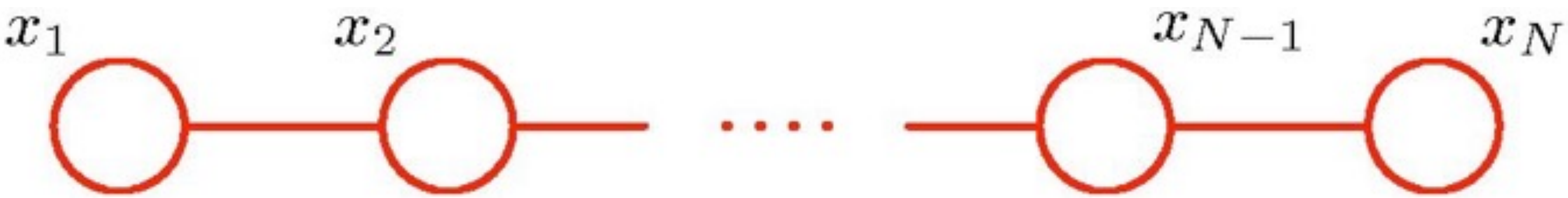
As a conclusion, all probability distributions that can be factorized as in (4.1), can be represented as an MRF.

Converting Directed to Undirected Graphs (1)



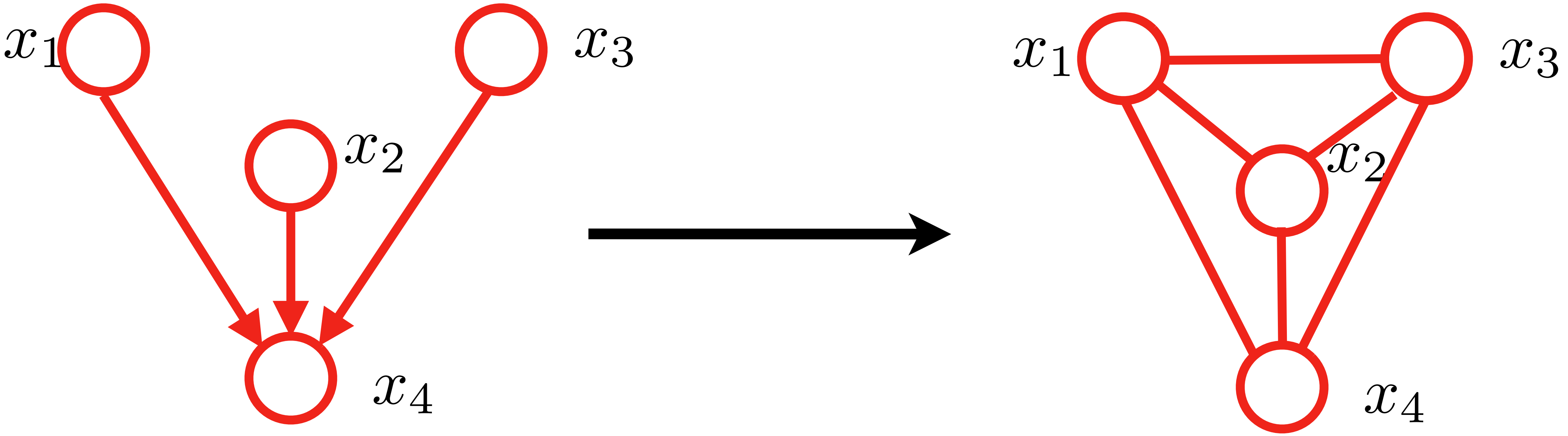
$$p(\mathbf{x}) = \underbrace{p(x_1)p(x_2|x_1)}\; p(x_3|x_2) \cdots p(x_N|x_{N-1})$$

$$p(\mathbf{x}) = \frac{1}{Z}\, \psi_{1,2}(x_1, x_2)\; \psi_{2,3}(x_2, x_3) \cdots \psi_{N-1,N}(x_{N-1}, x_N)$$



In this case: Z=1

Converting Directed to Undirected Graphs (2)



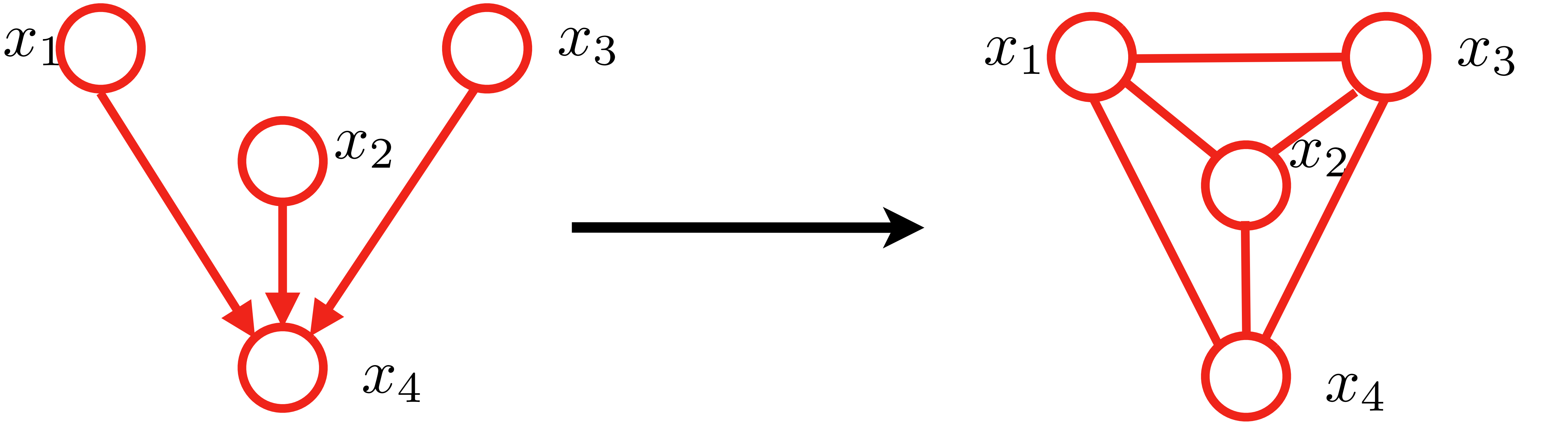
$$p(\mathbf{x}) = p(x_1)p(x_2)p(x_2)p(x_4 \mid x_1, x_2, x_3)$$

In general: conditional distributions in the directed graph are mapped to cliques in the undirected graph

However: the variables are **not** conditionally independent given the head-to-head node

Therefore: Connect all parents of head-to-head nodes with each other (**moralization**)

Converting Directed to Undirected Graphs (2)



$$p(\mathbf{x}) = p(x_1)p(x_2)p(x_2)p(x_4 \mid x_1, x_2, x_3)$$

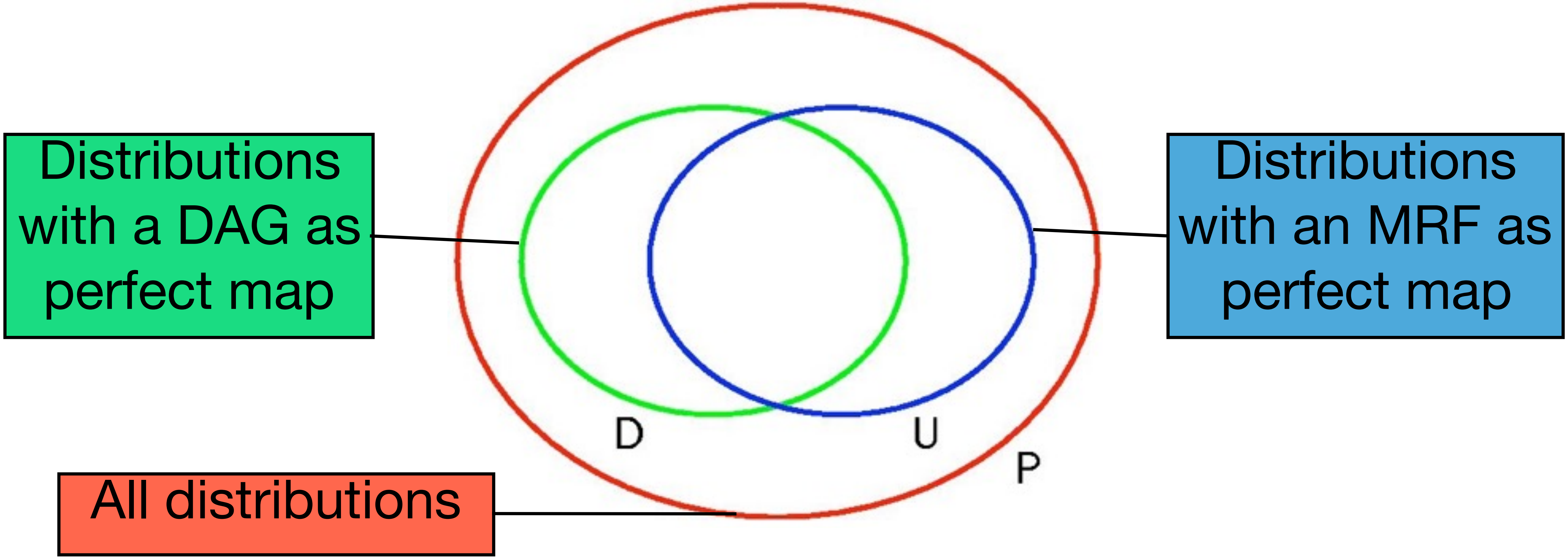
$$p(\mathbf{x}) = \phi(x_1, x_2, x_3, x_4)$$

Problem: This process can remove conditional independence relations (inefficient)

Generally: There is no one-to-one mapping between the distributions represented by directed and by undirected graphs.

Representability

- As for DAGs, we can define an I-map, a D-map and a perfect map for MRFs.
- The set of all distributions for which a DAG exists that is a perfect map is different from that for MRFs.



Distributions with a DAG as perfect map

Distributions with an MRF as perfect map

All distributions

Using Graphical Models

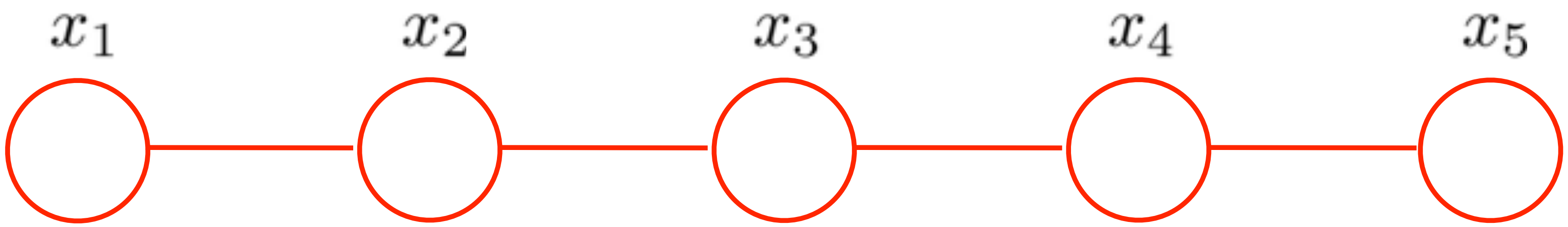
We can use a graphical model to do **inference**:

- Some nodes in the graph are **observed**, for others we want to find the posterior distribution
- Also, computing the local **marginal distribution** $p(x_n)$ at any node x_n can be done using inference.

Question: How can inference be done with a graphical model?

We will see that when exploiting conditional independences we can do efficient inference.

Inference on a Chain



The joint probability is given by

$$p(\mathbf{x}) = \frac{1}{Z}\psi_{1,2}(x_1, x_2)\psi_{2,3}(x_2, x_3)\psi_{3,4}(x_3, x_4)\psi_{4,5}(x_4, x_5)$$

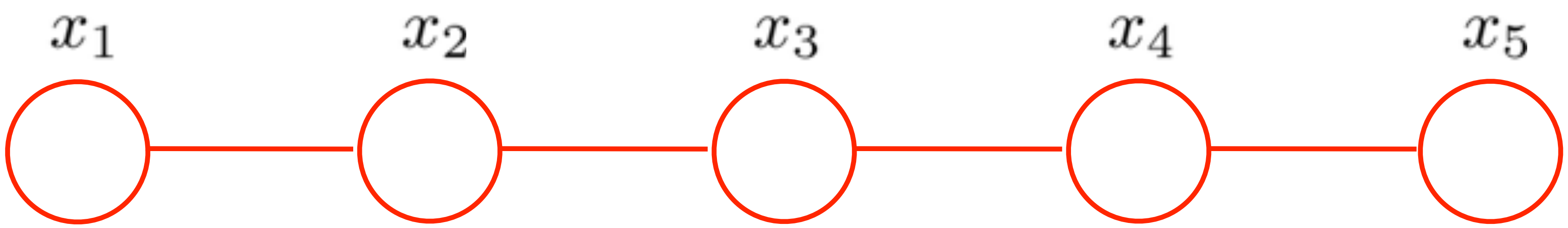
The marginal at x_3 is $$p(x_3) = \sum_{x_1}\sum_{x_2}\sum_{x_4}\sum_{x_5} p(\mathbf{x})$$

In the general case with N nodes we have

$$p(\mathbf{x}) = \frac{1}{Z}\psi_{1,2}(x_1, x_2)\psi_{2,3}(x_2, x_3)\cdots\psi_{N-1,N}(x_{N-1}, x_N)$$

and $$p(x_n) = \sum_{x_1}\cdots\sum_{x_{n-1}}\sum_{x_{n+1}}\cdots\sum_{x_N} p(\mathbf{x})$$

Inference on a Chain



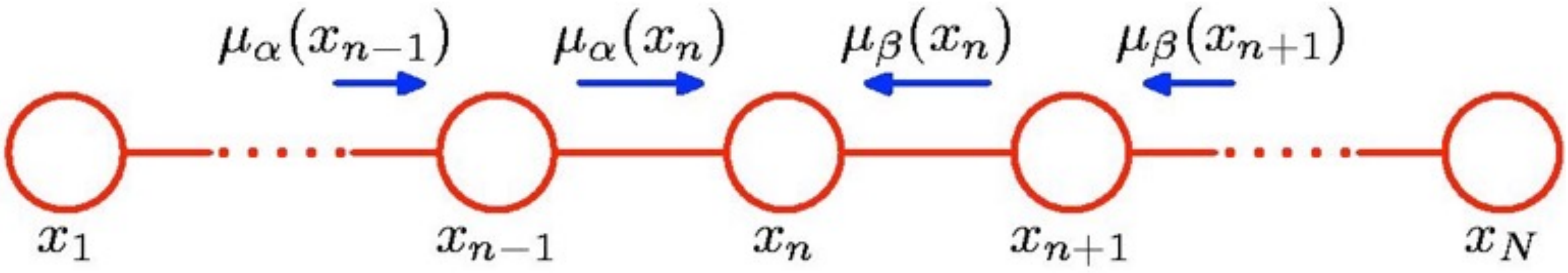
$$p(x_3) = \sum_{x_1}\sum_{x_2}\sum_{x_4}\sum_{x_5} p(\mathbf{x})$$

- This would mean K^N computations! A more efficient way is obtained by rearranging:

$$\begin{aligned} p(x_3) &= \frac{1}{Z}\sum_{x_1}\sum_{x_2}\sum_{x_4}\sum_{x_5}\psi_{1,2}(x_1,x_2)\psi_{2,3}(x_2,x_3)\psi_{3,4}(x_3,x_4)\psi_{4,5}(x_4,x_5) \\ &= \frac{1}{Z}\sum_{x_2}\sum_{x_1}\sum_{x_4}\sum_{x_5}\psi_{1,2}(x_1,x_2)\psi_{2,3}(x_2,x_3)\psi_{3,4}(x_3,x_4)\psi_{4,5}(x_4,x_5) \\ &= \frac{1}{Z}\underbrace{\sum_{x_2}\psi_{2,3}(x_2,x_3)\sum_{x_1}\psi_{1,2}(x_1,x_2)}_{\mu_\alpha(x_3)}\underbrace{\sum_{x_4}\psi_{3,4}(x_3,x_4)\sum_{x_5}\psi_{4,5}(x_4,x_5)}_{\mu_\beta(x_3)} \end{aligned}$$

Vectors of size K

Inference on a Chain



In general, we have

$$p(x_n) = \frac{1}{Z} \underbrace{\left[\sum_{x_{n-1}} \psi_{n-1,n}(x_{n-1}, x_n) \cdots \left[\sum_{x_1} \psi_{1,2}(x_1, x_2) \right] \cdots \right]}_{\mu_\alpha(x_n)}$$

$$\underbrace{\left[\sum_{x_{n+1}} \psi_{n,n+1}(x_n, x_{n+1}) \cdots \left[\sum_{x_N} \psi_{N-1,N}(x_{N-1}, x_N) \right] \cdots \right]}_{\mu_\beta(x_n)}$$

Inference on a Chain

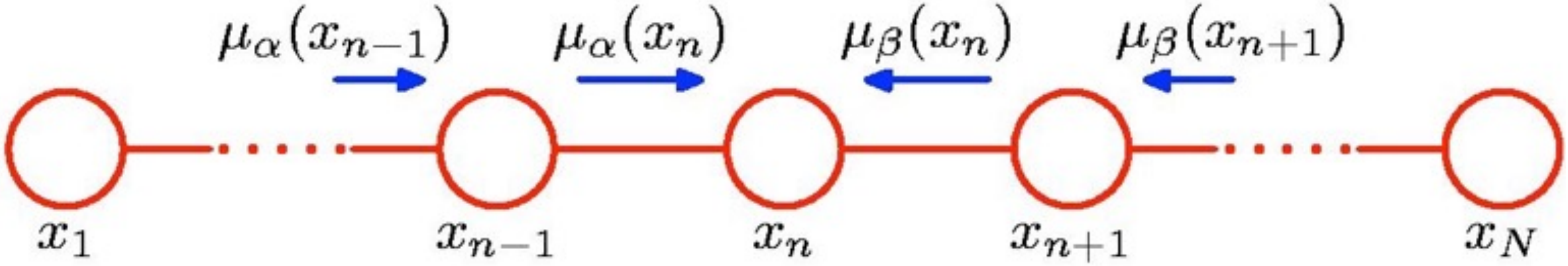
The messages μ_α and μ_β can be computed recursively:

$$\begin{aligned}\mu_\alpha(x_n) &= \sum_{x_{n-1}} \psi_{n-1,n}(x_{n-1}, x_n) \left[\sum_{x_{n-2}} \cdots \right] \\ &= \sum_{x_{n-1}} \psi_{n-1,n}(x_{n-1}, x_n) \mu_\alpha(x_{n-1}).\end{aligned}$$

$$\begin{aligned}\mu_\beta(x_n) &= \sum_{x_{n+1}} \psi_{n,n+1}(x_n, x_{n+1}) \left[\sum_{x_{n+2}} \cdots \right] \\ &= \sum_{x_{n+1}} \psi_{n,n+1}(x_n, x_{n+1}) \mu_\beta(x_{n+1}).\end{aligned}$$

Computation of μ_α starts at the first node and computation of μ_β starts at the last node.

Inference on a Chain



- The first values of μ_α and μ_β are:

$$\mu_\alpha(x_2) = \sum_{x_1} \psi_{1,2}(x_1, x_2) \qquad \mu_\beta(x_{N-1}) = \sum_{x_N} \psi_{N-1,N}(x_{N-1}, x_N)$$

- The partition function can be computed at any node:

$$Z = \sum_{x_n} \mu_\alpha(x_n)\mu_\beta(x_n)$$

- Overall, we have $O(NK^2)$ operations to compute the marginal $p(x_n)$

Inference on a Chain

To compute local marginals:

- Compute and store all forward messages, $\mu_\alpha(x_n)$.
- Compute and store all backward messages, $\mu_\beta(x_n)$
- Compute Z **once** at a node x_m: $Z = \sum_{x_m} \mu_\alpha(x_m)\mu_\beta(x_m)$
- Compute

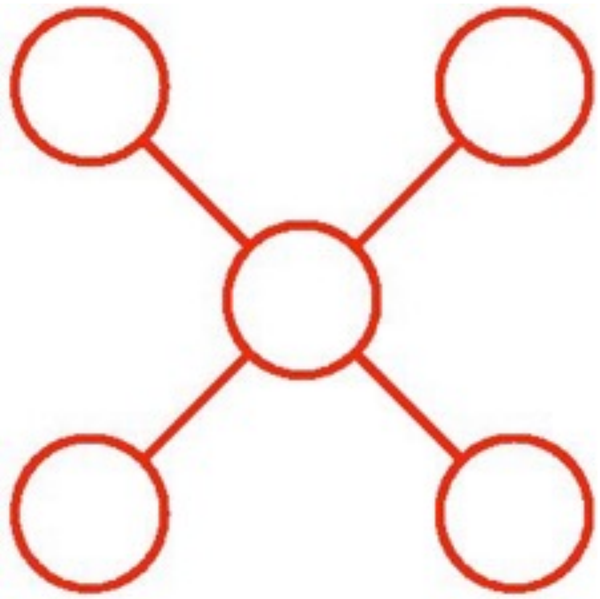
$$p(x_n) = \frac{1}{Z}\mu_\alpha(x_n)\mu_\beta(x_n)$$

for all variables required.

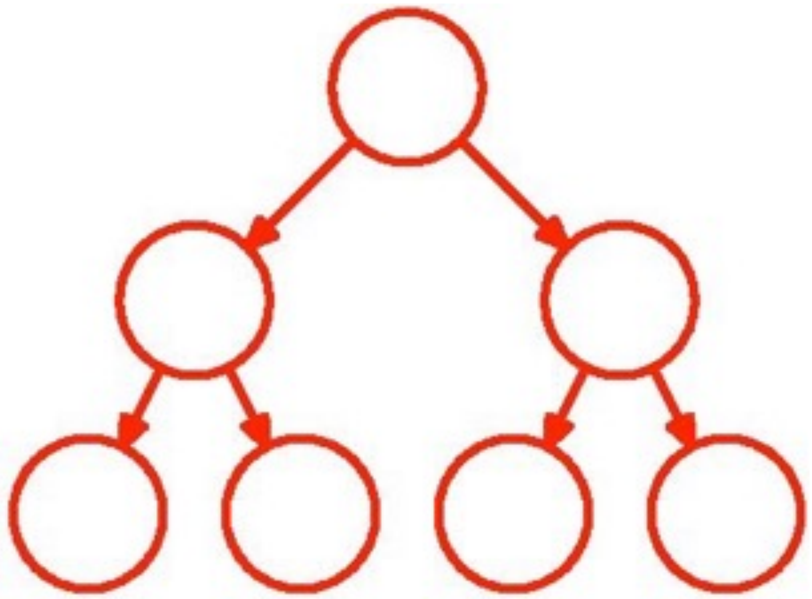
More General Graphs

The message-passing algorithm can be extended to more general graphs:

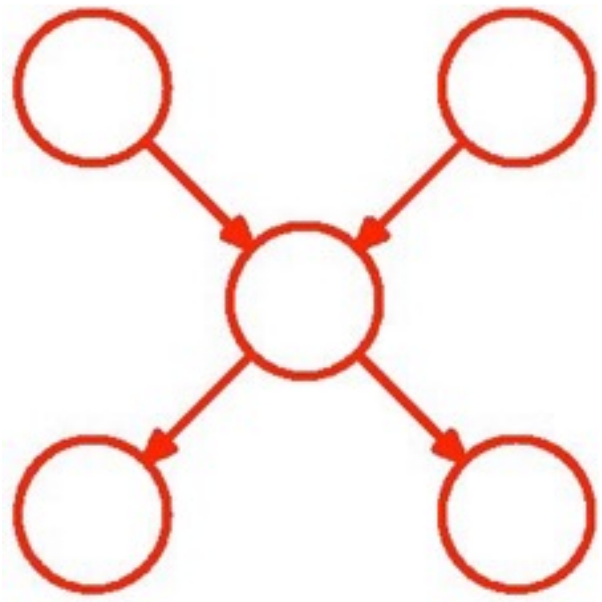
Undirected Tree



Directed Tree



Polytree

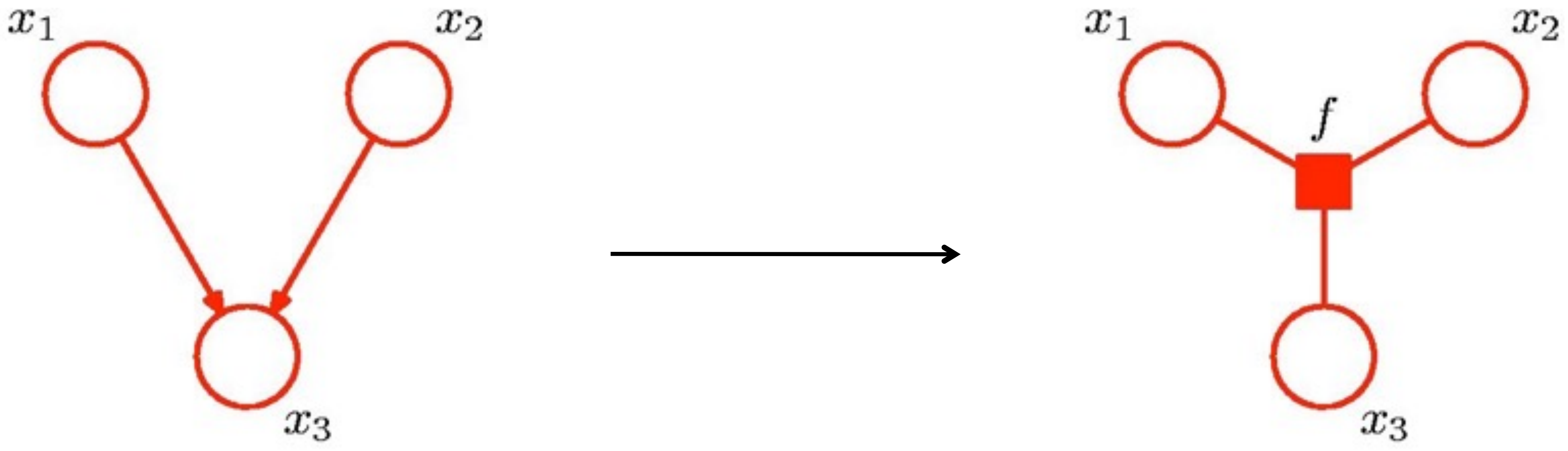


It is then known as the **sum-product algorithm.**

A special case of this is **belief propagation**.

Factor Graphs

- The Sum-product algorithm can be used to do inference on undirected and directed graphs.
- A representation that generalizes directed and undirected models is the **factor graph**.



$p(\mathbf{x}) = p(x_1)p(x_2)p(x_3|x_1, x_2)$

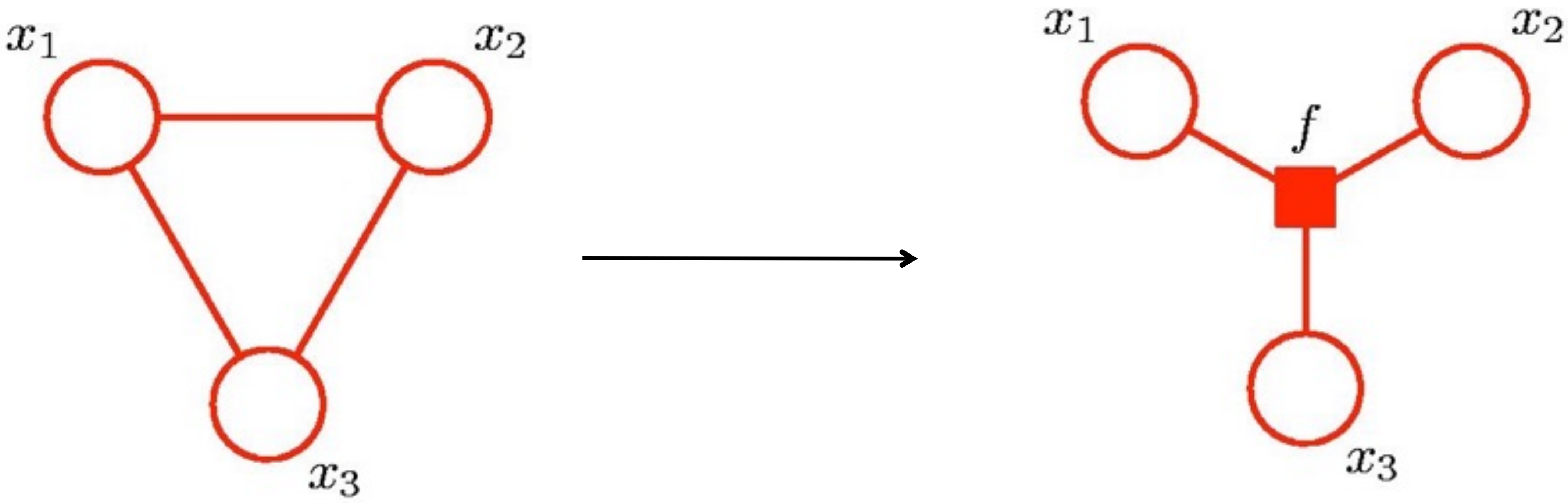
$f(x_1, x_2, x_3) = p(x_1)p(x_2)p(x_3 \mid x_1, x_2)$

Directed graph

Factor graph

Factor Graphs

- The Sum-product algorithm can be used to do inference on undirected and directed graphs.
- A representation that generalizes directed and undirected models is the **factor graph**.



$\psi(x_1, x_2, x_3)$ $\quad$ $f(x_1, x_2, x_3) = \psi(x_1, x_2, x_3)$

Undirected graph $\quad$ Factor graph

Sum-Product Inference in General Graphical Models

1. Convert graph (directed or undirected) into a **factor graph** (there are no cycles)

2. If the goal is to **marginalize** at node x, then consider x as a root node

3. Initialize the recursion at the leaf nodes as:

   $\mu_{f \to x}(x) = 1$ (var) or $\mu_{x \to f}(x) = f(x)$ (fac)

4. Propagate messages from the leaves to x

5. Propagate messages from x to the leaves

6. Obtain marginals at every node by multiplying all incoming messages

Further Topics on Graphical Models

Other inference algorithms:

- Max-Sum algorithm: used to **maximize** the joint probability of all variables (no marginalization)
- Junction Tree algorithm: exact inference for general graphs (even with loops)
- Loopy belief propagation: approximate inference on general graphs (more efficient)

Special kind of undirected GM:

- Conditional Random fields (e.g.: classification)

More details: see class of Dr. Domokos

http://vision.in.tum.de/teaching/ss2016/lecture_graphical_models

Summary

- Undirected models (aka Markov random fields) provide an intuitive representation of conditional independence
- An MRF is defined as a **factorization** over clique potentials and normalized globally
- Directed and undirected models have different representative power (no simple “containment”)
- Inference on undirected Markov chains is efficient using message passing
- Factor graphs are more general; exact inference can be done efficiently using sum-product