Graphical Models

Mark van der Wilk

Department of Computing
Imperial College London

@markvanderwilk
m.vdwilk@imperial.ac.uk

January 16, 2022

Probabilistic Models

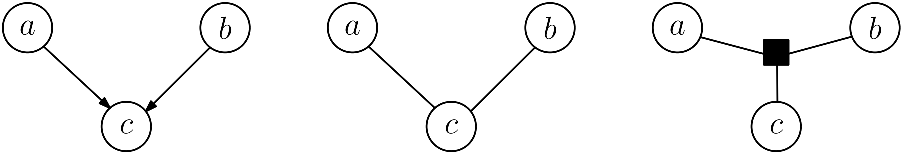
Previously we saw:

- Probabilistic model is a **joint distribution** $p(\mathbf{x}, \mathbf{z})$.
- We make factorisation assumptions to specify the model.
- Factorisation assumptions help simplify the posterior.

Graphical models help us to:

- visualise (conditional) independence,
- specify models with the right structure,
- find (conditional) independence when conditioning,
- do inference automatically and efficiently.

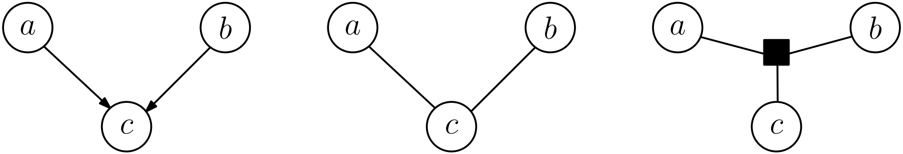
Probabilistic Graphical Models



Three types of probabilistic graphical models:

- Bayesian networks (directed graphical models)
- Markov random fields (undirected graphical models)
- Factor graphs

Probabilistic Graphical Models

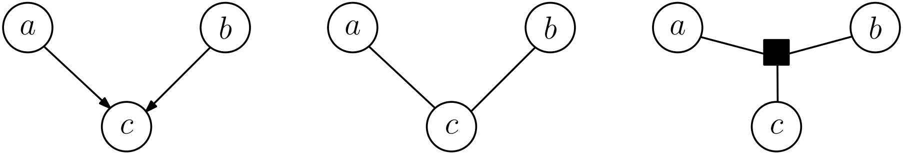


Three types of probabilistic graphical models:

- Bayesian networks (directed graphical models)
- Markov random fields (undirected graphical models)
- Factor graphs

- **Nodes:** (Sets of) random variables
- **Edges:** Probabilistic/functional relations between variables

Probabilistic Graphical Models



Three types of probabilistic graphical models:

- Bayesian networks (directed graphical models)
- Markov random fields (undirected graphical models)
- Factor graphs

- **Nodes:** (Sets of) random variables
- **Edges:** Probabilistic/functional relations between variables

▶▶ Graph captures the way in which the joint distribution over all random variables can be decomposed into a product of factors depending only on a subset of these variables

Importance of Visualization

$$Pr(\{y_g, \gamma_g, t_{gk}, \beta_{gk}, l_d, f_g, z_n, i_{ng}\}|\{w_{nd}\}) = \prod_g^G p(y_g|\rho)p(\gamma_g|\sigma)p(f_g|\alpha)\cdot$$
$$[\prod_k^K p(t_{gk}|\gamma_g)p(\beta_{gk}|t_{gk}, y_g)]p(\kappa|\alpha)\prod_d^D p(l_d|\kappa)p(\pi|\alpha)\prod_n^N p(z_n|\pi)$$
$$\prod_n^N \prod_g^G p(i_{ng}|\beta, z_n)\prod_n^N \prod_d^D p(w_{nd}|i_{ng}, f, l_d)]$$

From Kim et al. (NIPS, 2015)

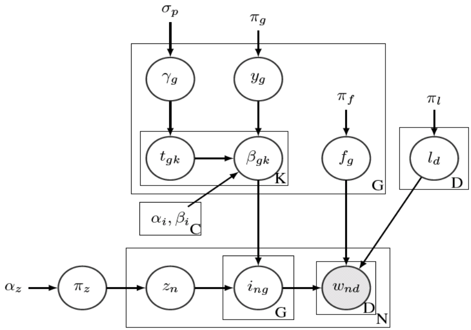
Importance of Visualization

$$Pr(\{y_g, \gamma_g, t_{gk}, \beta_{gk}, l_d, f_g, z_n, i_{ng}\}|\{w_{nd}\}) = \prod_g^G p(y_g|\rho)p(\gamma_g|\sigma)p(f_g|\alpha)\cdot$$

$$[\prod_k^K p(t_{gk}|\gamma_g)p(\beta_{gk}|t_{gk}, y_g)]p(\kappa|\alpha)\prod_d^D p(l_d|\kappa)p(\pi|\alpha)\prod_n^N p(z_n|\pi)$$

$$\prod_n^N \prod_g^G p(i_{ng}|\beta, z_n)\prod_n^N \prod_d^D p(w_{nd}|i_{ng}, f, l_d)]$$

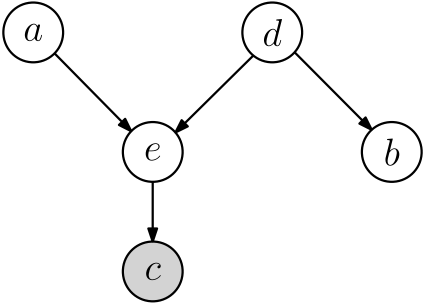
From Kim et al. (NIPS, 2015)



From Kim et al. (NIPS, 2015)

Expressing Factorisations as Graphs

Directed Graphical Models



- Nodes: Random variables
- Shaded nodes: Observed random variables
- Unshaded nodes: Unobserved (latent) random variables
- Directed arrow from a to b: Conditional distribution $p(b|a)$.

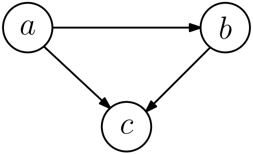
Skill: From Joints to Graphs

Consider the joint distribution

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

Building the corresponding graphical model:

1. Create a node for all random variables



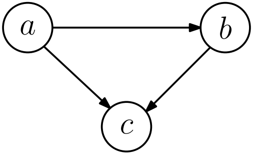
Skill: From Joints to Graphs

Consider the joint distribution

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

Building the corresponding graphical model:

1. Create a node for all random variables
2. For each conditional distribution, we add a directed link (arrow) to the graph from the nodes corresponding to the variables on which the distribution is conditioned on



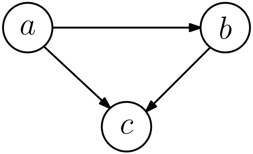
Skill: From Joints to Graphs

Consider the joint distribution

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

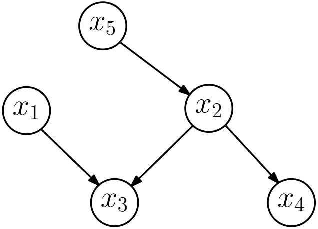
Building the corresponding graphical model:

1. Create a node for all random variables
2. For each conditional distribution, we add a directed link (arrow) to the graph from the nodes corresponding to the variables on which the distribution is conditioned on



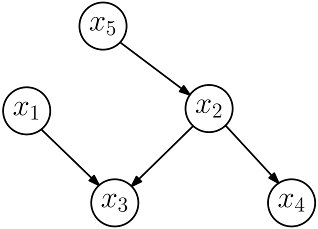
▶▶ Graph layout depends on the choice of factorization

Skill: From Graphs to Joints



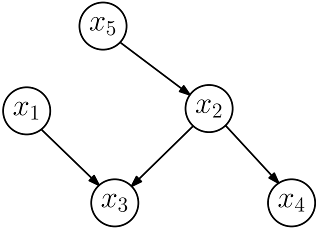
- Joint distribution is the product of a set of conditionals, one for each node in the graph

Skill: From Graphs to Joints



- Joint distribution is the product of a set of conditionals, one for each node in the graph
- Each conditional depends only on the parents of the corresponding node in the graph

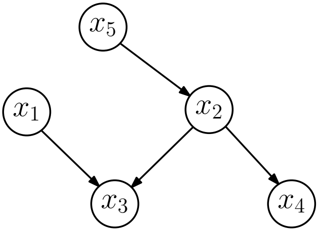
Skill: From Graphs to Joints



- Joint distribution is the product of a set of conditionals, one for each node in the graph
- Each conditional depends only on the parents of the corresponding node in the graph

$$p(x_1, x_2, x_3, x_4, x_5) = p(x_1)p(x_5)p(x_2|x_5)p(x_3|x_1, x_2)p(x_4|x_2)$$

Skill: From Graphs to Joints



- Joint distribution is the product of a set of conditionals, one for each node in the graph
- Each conditional depends only on the parents of the corresponding node in the graph

$$p(x_1, x_2, x_3, x_4, x_5) = p(x_1)p(x_5)p(x_2|x_5)p(x_3|x_1, x_2)p(x_4|x_2)$$

In general: $p(\boldsymbol{x}) = p(x_1, \ldots, x_K) = \prod_{k=1}^{K} p(x_k|\text{parents}(x_k))$

What is **the** graphical model?

Remember, a model is defined simply by its joint distribution, which often is just between data and a latent variable:

$$p(\mathbf{x}, \mathbf{z}) \tag{1}$$

What is **the** graphical model?

Remember, a model is defined simply by its joint distribution, which often is just between data and a latent variable:

$$p(\mathbf{x}, \mathbf{z}) \tag{1}$$

We can factorise this in two distinct, but equally valid ways:

$$p(\mathbf{x}, \mathbf{z}) = p(\mathbf{x}|\mathbf{z})p(\mathbf{z}) = p(\mathbf{x})p(\mathbf{z}|\mathbf{x}) \tag{2}$$

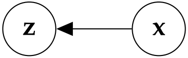
What is **the** graphical model?

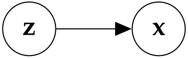
Remember, a model is defined simply by its joint distribution, which often is just between data and a latent variable:

$$p(\mathbf{x}, \mathbf{z}) \tag{1}$$

We can factorise this in two distinct, but equally valid ways:

$$p(\mathbf{x}, \mathbf{z}) = p(\mathbf{x}|\mathbf{z})p(\mathbf{z}) = p(\mathbf{x})p(\mathbf{z}|\mathbf{x}) \tag{2}$$





Which one is correct?

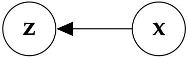
What is **the** graphical model?

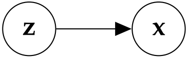
Remember, a model is defined simply by its joint distribution, which often is just between data and a latent variable:

$$p(\mathbf{x}, \mathbf{z}) \tag{1}$$

We can factorise this in two distinct, but equally valid ways:

$$p(\mathbf{x}, \mathbf{z}) = p(\mathbf{x}|\mathbf{z})p(\mathbf{z}) = p(\mathbf{x})p(\mathbf{z}|\mathbf{x}) \tag{2}$$

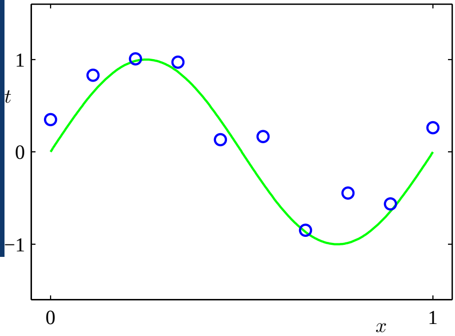




Which one is correct? Depends on which conditional you specified!

$$p(\mathbf{x}, \mathbf{z}) = p(\mathbf{x}|\mathbf{z})\, p(\mathbf{z}) = p(\mathbf{x})\, p(\mathbf{z}|\mathbf{x}) \tag{3}$$

Graphical Model for (Bayesian) Linear Regression



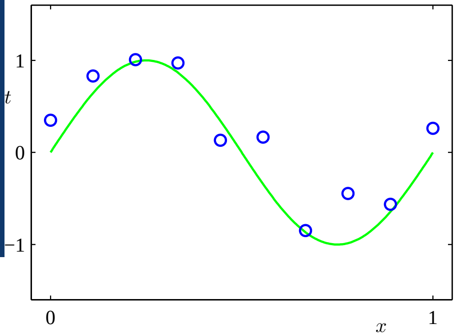
From PRML (Bishop, 2006)

We are given a data set $(x_1, y_1), \ldots, (x_N, y_N)$ where

$$y_i = f(x_i) + \varepsilon, \quad \varepsilon \sim \mathcal{N}\big(0, \sigma^2\big)$$

with f unknown.
▶▶ Find a (regression) model that explains the data

Graphical Model for (Bayesian) Linear Regression



From PRML (Bishop, 2006)

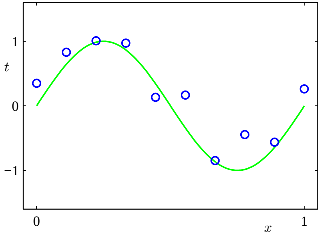
We are given a data set $(x_1, y_1), \ldots, (x_N, y_N)$ where

$$y_i = f(x_i) + \varepsilon, \quad \varepsilon \sim \mathcal{N}\left(0,\, \sigma^2\right)$$

with f unknown.
▶▶ Find a (regression) model that explains the data

- Consider polynomials $f(x) = \sum_{j=0}^{M} w_j x^j$ with parameters $\boldsymbol{w} = [w_0, \ldots, w_M]^\top$.
- Bayesian linear regression: Place a conjugate Gaussian prior on the parameters: $p(\boldsymbol{w}) = \mathcal{N}\left(\mathbf{0},\, \alpha^2 \boldsymbol{I}\right)$

Graphical Model for Linear Regression



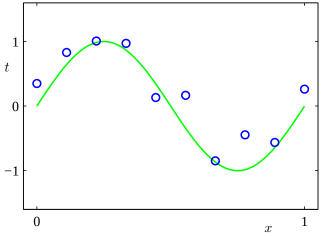
From PRML (Bishop, 2006)

$$p(y_i|\boldsymbol{w}, x_i) = \mathcal{N}\left(y_i \mid f_{\boldsymbol{w}}(x_i), \sigma^2\right)$$

$$f(x) = \sum_{j=0}^{M} w_j x^j$$

$$p(\boldsymbol{w}) = \mathcal{N}\left(\boldsymbol{0}, \alpha^2 \boldsymbol{I}\right)$$

Graphical Model for Linear Regression

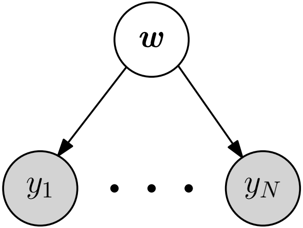


From PRML (Bishop, 2006)

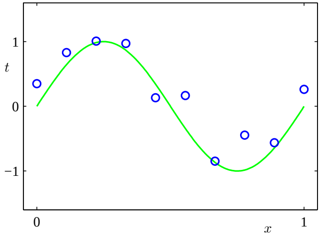
$$p(y_i|\boldsymbol{w}, x_i) = \mathcal{N}\left(y_i \mid f_{\boldsymbol{w}}(x_i),\, \sigma^2\right)$$

$$f(x) = \sum_{j=0}^{M} w_j x^j$$

$$p(\boldsymbol{w}) = \mathcal{N}\left(\boldsymbol{0},\, \alpha^2 \boldsymbol{I}\right)$$



Graphical Model for Linear Regression

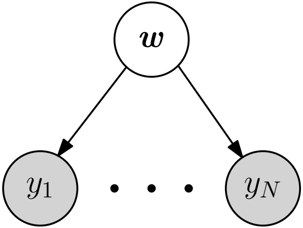


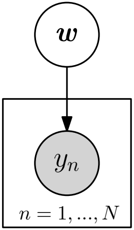
From PRML (Bishop, 2006)

$$p(y_i|\boldsymbol{w}, x_i) = \mathcal{N}\left(y_i \mid f_{\boldsymbol{w}}(x_i), \sigma^2\right)$$

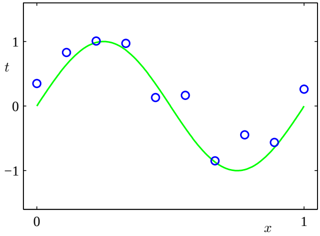
$$f(x) = \sum_{j=0}^{M} w_j x^j$$

$$p(\boldsymbol{w}) = \mathcal{N}\left(\boldsymbol{0}, \alpha^2 \boldsymbol{I}\right)$$





Graphical Model for Linear Regression

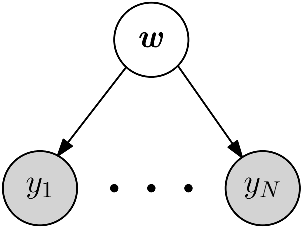


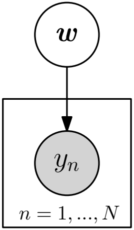
From PRML (Bishop, 2006)

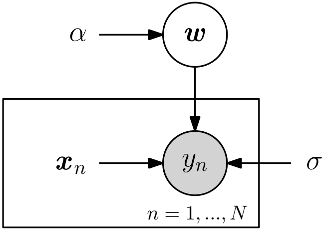
$$p(y_i|\boldsymbol{w}, x_i) = \mathcal{N}\left(y_i \mid f_{\boldsymbol{w}}(x_i), \sigma^2\right)$$

$$f(x) = \sum_{j=0}^{M} w_j x^j$$

$$p(\boldsymbol{w}) = \mathcal{N}\left(\mathbf{0}, \alpha^2 \boldsymbol{I}\right)$$

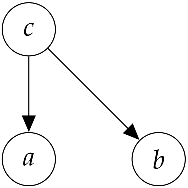






Finding Conditional Independence

Conditional Independence



$$a \perp\!\!\!\perp b|c \iff p(a,b|c) = p(a|c)p(b|c)$$
$$\iff p(a|b,c) = p(a|c)$$

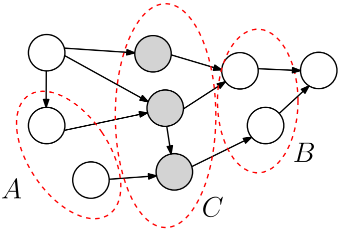
- (Conditional) independence allows for a factorization of the joint distribution ⏩ More efficient inference

$$\mathbb{E}_{p(a,b|c)}[f(a)g(b)] = \mathbb{E}_{p(a|c)}[f(a)] \cdot \mathbb{E}_{p(b|c)}[g(b)] \tag{4}$$

- Conditional independence properties of the joint distribution can be read directly from the graph without analytical manipulations!

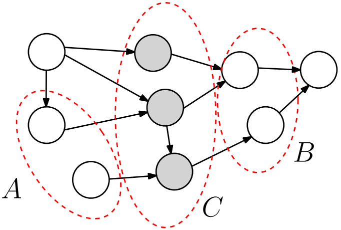
⏩ **d-separation** (Pearl, 1988)

D-Separation (Directed Graphs)



Directed, acyclic graph in which A, B, C are arbitrary, non-intersecting sets of nodes. Does $A \perp\!\!\!\perp B | C$ hold?
Note: C is observed if we condition on it (and the nodes in the GM are shaded)

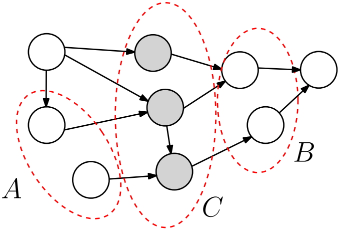
D-Separation (Directed Graphs)



Directed, acyclic graph in which A, B, C are arbitrary, non-intersecting sets of nodes. Does $A \perp\!\!\!\perp B | C$ hold?
Note: C is observed if we condition on it (and the nodes in the GM are shaded)

- Consider all possible paths from any node in A to any node in B.

D-Separation (Directed Graphs)



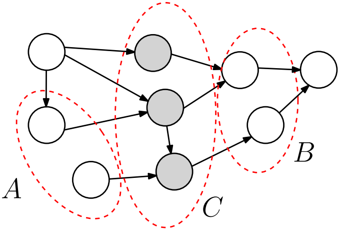
Directed, acyclic graph in which A, B, C are arbitrary, non-intersecting sets of nodes. Does $A \perp\!\!\!\perp B | C$ hold?
Note: C is observed if we condition on it (and the nodes in the GM are shaded)

▶▶ Consider all possible paths from any node in A to any node in B.

Any such **path is blocked** if it includes a node such that either

- 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

D-Separation (Directed Graphs)



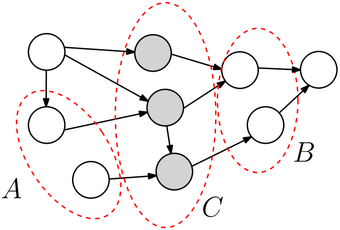
Directed, acyclic graph in which A, B, C are arbitrary, non-intersecting sets of nodes. Does $A \perp\!\!\!\perp B | C$ hold?
Note: C is observed if we condition on it (and the nodes in the GM are shaded)

▶▶ Consider all possible paths from any node in A to any node in B.

Any such **path is blocked** if it includes a node such that either

- 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
- Arrows meet head-to-head at the node and neither the node nor any of its descendants is in the set C

D-Separation (Directed Graphs)



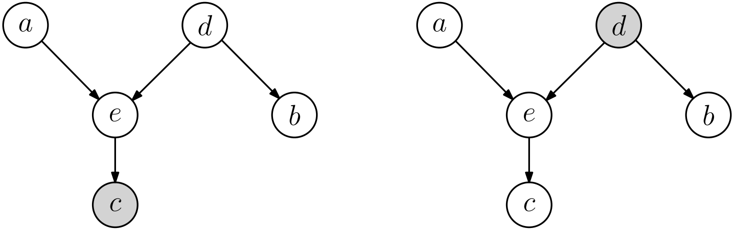
Directed, acyclic graph in which A, B, C are arbitrary, non-intersecting sets of nodes. Does $A \perp\!\!\!\perp B | C$ hold?
Note: C is observed if we condition on it (and the nodes in the GM are shaded)

▶▶ Consider all possible paths from any node in A to any node in B.
Any such **path is blocked** if it includes a node such that either

- 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
- Arrows meet head-to-head at the node and neither the node nor any of its descendants is in the set C

If all paths are blocked, then A is d-separated (conditionally indep.) from B by C, and the joint distribution satisfies $A \perp\!\!\!\perp B | C$.

Exam skill: Find conditional independencies

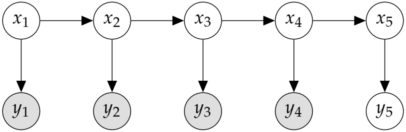


(a) $a \perp\!\!\!\perp b | c$?   (b) $a \perp\!\!\!\perp b | d$?

A path is **blocked** if it includes a node such that either

- 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 (observed) or
- The arrows meet head-to-head at the node, and neither the node nor any of its descendants is in the set C (observed)

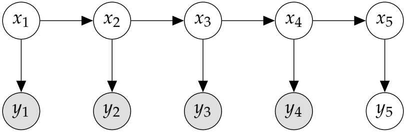
Markov Chains



$$p(\{x_t\}_{t=1}^5, \{y_t\}_{t=1}^5) = p(x_1)\left[\prod_{t=2}^{5} p(x_t|x_{t-1})\right]\left[\prod_{t=1}^{5} p(y_t|x_t)\right] \tag{5}$$

How do we compute the posterior on x_5 (so we can predict y_5)?

Markov Chains

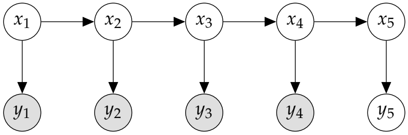


$$p(\{x_t\}_{t=1}^5, \{y_t\}_{t=1}^5) = p(x_1)\left[\prod_{t=2}^5 p(x_t|x_{t-1})\right]\left[\prod_{t=1}^5 p(y_t|x_t)\right] \tag{5}$$

How do we compute the posterior on x_5 (so we can predict y_5)?

- We note that $x_4 \perp\!\!\!\perp \{x_t\}_{t=1}^2, \{y_t\}_{t=1}^3 | x_3$ (blocked path, tail-to-tail)
  $p(x_4|\{y_t\}_{t=1}^4) = \int p(x_4|x_3, y_4)p(x_3|\{y_t\}_{t=1}^3)\mathrm{d}x_3$

Markov Chains

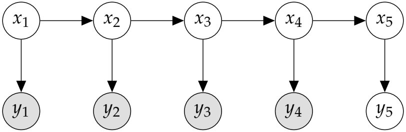


$$p(\{x_t\}_{t=1}^5, \{y_t\}_{t=1}^5) = p(x_1)\left[\prod_{t=2}^5 p(x_t|x_{t-1})\right]\left[\prod_{t=1}^5 p(y_t|x_t)\right] \tag{5}$$

How do we compute the posterior on x_5 (so we can predict y_5)?

- We note that $x_4 \perp\!\!\!\perp \{x_t\}_{t=1}^2, \{y_t\}_{t=1}^3 | x_3$ (blocked path, tail-to-tail)
  $p(x_4|\{y_t\}_{t=1}^4) = \int p(x_4|x_3, y_4)p(x_3|\{y_t\}_{t=1}^3)\mathrm{d}x_3$
- This can be applied recursively!

Markov Chains



$$p(\{x_t\}_{t=1}^5, \{y_t\}_{t=1}^5) = p(x_1)\left[\prod_{t=2}^5 p(x_t|x_{t-1})\right]\left[\prod_{t=1}^5 p(y_t|x_t)\right] \tag{5}$$

How do we compute the posterior on x_5 (so we can predict y_5)?

- We note that $x_4 \perp\!\!\!\perp \{x_t\}_{t=1}^2, \{y_t\}_{t=1}^3 | x_3$ (blocked path, tail-to-tail)
  $p(x_4|\{y_t\}_{t=1}^4) = \int p(x_4|x_3, y_4)p(x_3|\{y_t\}_{t=1}^3)\mathrm{d}x_3$
- This can be applied recursively!
- Finding the posterior can be done in linear time!

Recommended Reading

Bishop: Pattern Recognition and Machine Learning, Chapter 8
Directed graphical models

References I

Not examinable from here

Factor Graphs

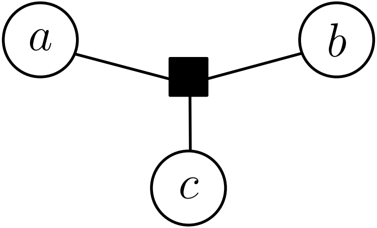
A different graphical representation

Good references:

Kschischang et al.: Factor Graphs and the Sum-Product Algorithm. IEEE Transactions on Information Theory (2001)

Loeliger: An Introduction to Factor Graphs. IEEE Signal Processing Magazine, (2004)

Factor Graphs



- (Un)directed graphical models express a global function of several variables as a product of factors over subsets of those variables
- Factor graphs make this decomposition explicit by introducing additional nodes for the factors themselves

Factorizing the Joint

The joint distribution is a product of factors:

$$p(\boldsymbol{x}) = \prod_s f_s(\boldsymbol{x}_s)$$

- $\boldsymbol{x} = (x_1, \dots, x_n)$
- $\boldsymbol{x}_s$: Subset of variables
- f_s: Factor; non-negative function of the variables $\boldsymbol{x}_s$

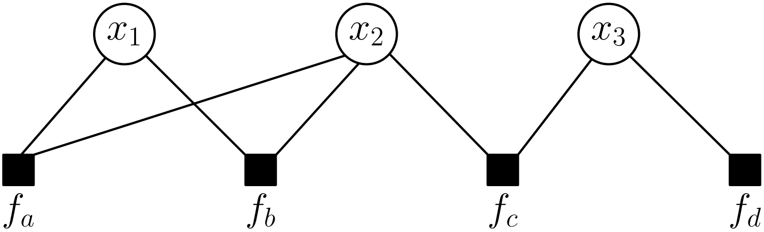
Factorizing the Joint

The joint distribution is a product of factors:

$$p(\boldsymbol{x}) = \prod_s f_s(\boldsymbol{x}_s)$$

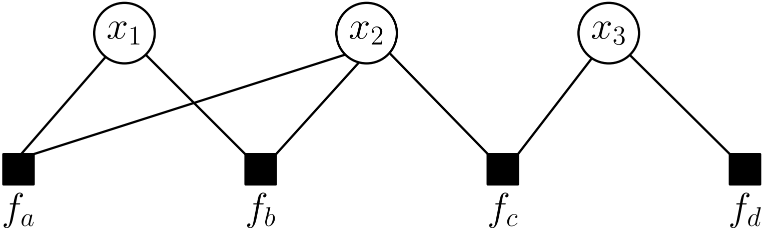
- $\boldsymbol{x} = (x_1, \dots, x_n)$
- $\boldsymbol{x}_s$: Subset of variables
- f_s: Factor; non-negative function of the variables $\boldsymbol{x}_s$
- Building a factor graph as a bipartite graph:
  - Nodes for all random variables (same as in (un)directed graphical models)
  - Additional nodes for factors (black squares) in the joint distribution
- Undirected links connecting each factor node to all of the variable nodes the factor depends on

Example



$$p(\boldsymbol{x}) = f_a(x_1, x_2) f_b(x_1, x_2) f_c(x_2, x_3) f_d(x_3)$$

Example



$$p(\boldsymbol{x}) = f_a(x_1, x_2) f_b(x_1, x_2) f_c(x_2, x_3) f_d(x_3)$$

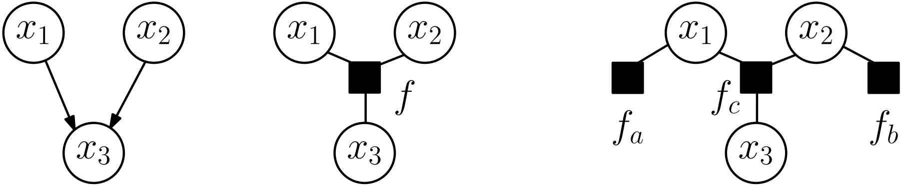
▶▶ Efficient inference algorithms for factor graphs (e.g., sum-product algorithm)

Directed Graphical Model → Factor Graph

1. Take variable nodes from Bayesian network
2. Create additional factor nodes corresponding to the conditional distributions
3. Add appropriate links

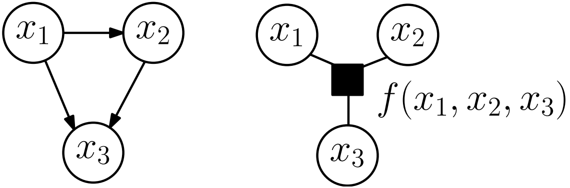
Not unique

Example: Directed Graph → Factor Graph



- Directed graph with factorization $p(x_1)p(x_2)p(x_3|x_1, x_2)$
- Factor graph with factor $f(x_1, x_2, x_3) = p(x_1)p(x_2)p(x_3|x_1, x_2)$
- Factor graph with factors $f_a = p(x_1),\ f_b = p(x_2),\ f_c = p(x_3|x_1, x_2)$

Removing Cycles



$$p(x_3|x_2, x_1)p(x_2|x_1)p(x_1) = f_a(x_1, x_2, x_3)f_b(x_1, x_2)f_c(x_2) = f(x_1, x_2, x_3) \tag{6}$$

- Local cycles in an (un)directed graph (due to links connecting parents of a node) can be removed on conversion to a factor graph

Exact Inference in Factor Graphs

Sum-Product Algorithm for Factor Graphs

- Factor graphs give a uniform treatment to message passing, which is used for inference in graphs
- Inference: Find (marginal) posterior distributions

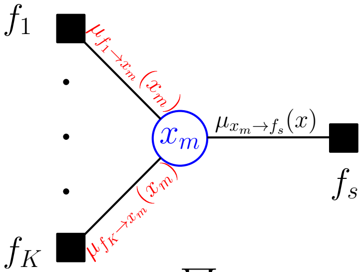
Sum-Product Algorithm for Factor Graphs

- Factor graphs give a uniform treatment to message passing, which is used for inference in graphs
- Inference: Find (marginal) posterior distributions
- Idea: Local message passing between nodes and factors
- Two different types of messages:
    - Messages $\mu_{x\to f}(x)$ from variable nodes to factors
    - Messages $\mu_{f\to x}(x)$ from factors to variable nodes

Sum-Product Algorithm for Factor Graphs

- Factor graphs give a uniform treatment to message passing, which is used for inference in graphs
- Inference: Find (marginal) posterior distributions
- Idea: Local message passing between nodes and factors
- Two different types of messages:
  - Messages $\mu_{x \to f}(x)$ from variable nodes to factors
  - Messages $\mu_{f \to x}(x)$ from factors to variable nodes
- Repeated sending of these messages through the graph converges
- Factors transform messages into evidence for the receiving node

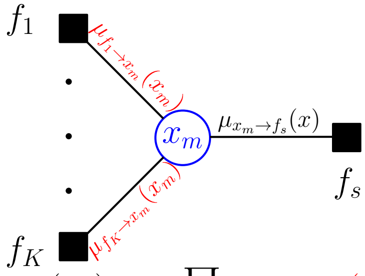
Variable-to-Factor Message



$$\mu_{x_m \to f_s}(x_m) = \prod_{l \in \mathrm{ne}(x_m) \backslash f_s} \mu_{f_l \to x_m}(x_m)$$

- Take the product of all incoming messages along all other links

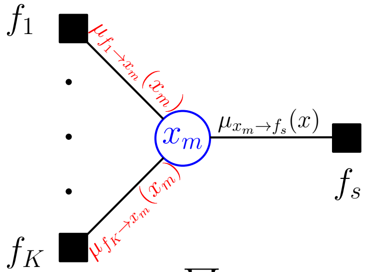
Variable-to-Factor Message



$$\mu_{x_m \to f_s}(x_m) = \prod_{l \in \mathrm{ne}(x_m) \setminus f_s} \mu_{f_l \to x_m}(x_m)$$

- Take the product of all incoming messages along all other links
- A variable node can send a message to a factor node once it has received messages from all other neighboring factors

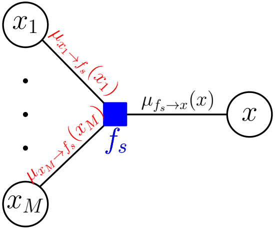
Variable-to-Factor Message



$$\mu_{x_m \to f_s}(x_m) = \prod_{l \in \text{ne}(x_m) \setminus f_s} \mu_{f_l \to x_m}(x_m)$$

- Take the product of all incoming messages along all other links
- A variable node can send a message to a factor node once it has received messages from all other neighboring factors
- The message that a node sends to a factor is made up of the messages that it receives from all other factors.

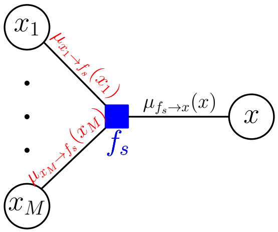
Factor-to-Variable Message



$$\mu_{f_s \to x}(x) = \prod_{m \in \mathrm{ne}(f_s) \setminus x} \mu_{x_m \to f_s}(x_m)$$

- Take the product of the incoming messages along all other links coming into the factor node

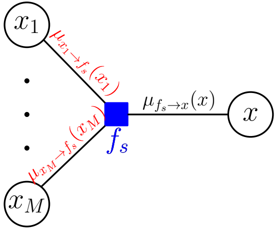
Factor-to-Variable Message



$$\mu_{f_s \to x}(x) = \quad f_s(x, x_1, \dots, x_M) \prod_{m \in \mathrm{ne}(f_s) \setminus x} \mu_{x_m \to f_s}(x_m)$$

- Take the product of the incoming messages along all other links coming into the factor node
- Multiply by the factor associated with that node

Factor-to-Variable Message



$$\mu_{f_s \to x}(x) = \sum_{x_1} \cdots \sum_{x_M} f_s(x, x_1, \ldots, x_M) \prod_{m \in \mathrm{ne}(f_s) \setminus x} \mu_{x_m \to f_s}(x_m)$$

- Take the product of the incoming messages along all other links coming into the factor node
- Multiply by the factor associated with that node
- Marginalize over all variables associated with the incoming messages

Initialization

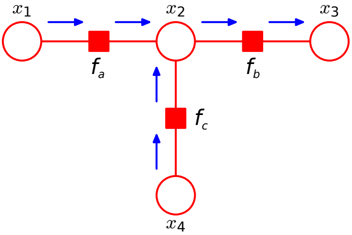
- If the leaf node is a variable node, initialize the corresponding messages to 1:

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

- If the leaf node is a factor node, the message should be

$$\mu_{f \to x}(x) = f(x)$$

Example (1)



From PRML (Bishop, 2006)

$$\mu_{x_1 \to f_a}(x_1) = 1$$

$$\mu_{f_a \to x_2}(x_2) = \sum_{x_1} f_a(x_1, x_2) \cdot 1$$

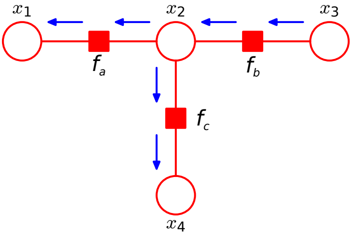
$$\mu_{x_4 \to f_c}(x_4) = 1$$

$$\mu_{f_c \to x_2}(x_2) = \sum_{x_4} f_c(x_2, x_4) \cdot 1$$

$$\mu_{x_2 \to f_b}(x_2) = \mu_{f_a \to x_2}(x_2)\mu_{f_c \to x_2}(x_2)$$

$$\mu_{f_b \to x_3}(x_3) = \sum_{x_2} f_b(x_2, x_3)\mu_{x_2 \to f_b}(x_2)$$

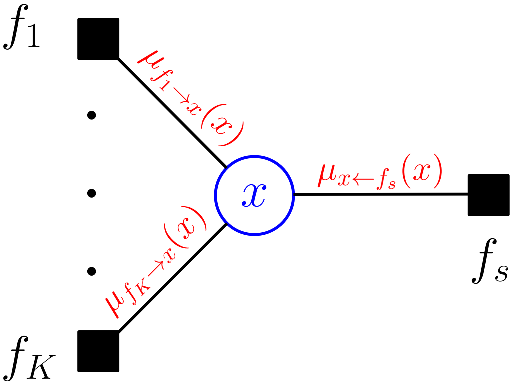
Example (2)



From PRML (Bishop, 2006)

$$
\begin{aligned}
\mu_{x_3 \to f_b}(x_3) &= 1 \\
\mu_{f_b \to x_2}(x_2) &= \sum_{x_3} f_b(x_2, x_3) \cdot 1 \\
\mu_{x_2 \to f_a}(x_2) &= \mu_{f_b \to x_2}(x_2) \mu_{f_c \to x_2}(x_2) \\
\mu_{f_a \to x_1}(x_1) &= \sum_{x_2} f_a(x_1, x_2) \mu_{x_2 \to f_a}(x_2) \\
\mu_{x_2 \to f_c}(x_2) &= \mu_{f_a \to x_2}(x_2) \mu_{f_b \to x_2}(x_2) \\
\mu_{f_c \to x_4}(x_4) &= \sum_{x_2} f_c(x_2, x_4) \mu_{x_2 \to f_c}(x_2)
\end{aligned}
$$

Marginals



For a single variable node the marginal is given as the product of all incoming messages:

$$p(x) = \prod_{f_i \in \mathrm{ne}(x)} \mu_{f_i \to x}(x)$$

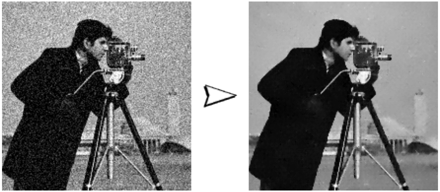
Observed Variables ⏩ Posterior

- Thus far, we have focused on the case where all variables are unobserved.
- Posterior is always conditioned on observations
- Partition $\boldsymbol{x} = \boldsymbol{h} \cup \boldsymbol{v}$, $\boldsymbol{h}$: hidden variables, $\boldsymbol{v}$: visible variables with observations $\hat{\boldsymbol{v}}$
- $p(\boldsymbol{v} = \hat{\boldsymbol{v}}) = \prod_i I(v_i = \hat{v}_i)$
- $p(\boldsymbol{x})p(\boldsymbol{v} = \hat{\boldsymbol{v}}) = p(\boldsymbol{h}, \boldsymbol{v} = \hat{\boldsymbol{v}}) \propto p(\boldsymbol{h}|\boldsymbol{v} = \hat{\boldsymbol{v}})$
- Marginal posteriors $p(h_i|\boldsymbol{v} = \hat{\boldsymbol{v}})$ can be obtained via sum-product algorithm and some local computations
  ⏩ (Koller & Friedman, 2009)

Exact Inference in (Un)Directed Graphical Models

- Loops are possible ⏩ **Junction Tree Algorithm** (Lauritzen & Spiegelhalter, 1988)
- Alternative: **Loopy Belief Propagation** (Frey & MacKay 1998)

Applications of Inference in Graphical Models



- Ranking: TrueSkill (Herbrich et al., 2007)
- Computer vision: de-noising, segmentation, semantic labeling, ...
  (e.g., Sucar & Gillies, 1994; Shotton et al., 2006; Szeliski et al., 2008)
- Coding theory: Low-density parity-check codes, turbo codes, ...
  (e.g., McEliece et al., 1998)
- Linear algebra: Solve linear equation systems (Shental et al., 2008)
- Signal processing: Iterative state estimation (e.g., Bickson et al., 2007; Deisenroth & Mohamed, 2012)