Exact Inference

Rico Krueger

Filipe Rodrigues

DTU Management Engineering
Department of Management Engineering

Outline

- Recap
- Analytical Derivation (Bayes' theorem)
- Variable Elimination
- Belief Propagation

Learning objectives

- Explain the concept of (exact) inference
- Perform analytical derivation inference with Gaussian distribution
- Perform variable elimination with discrete variables
- Perform belief propagation with discrete variables

Previously, in MBML...

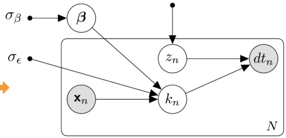
- Representation (weeks 2-4)
- Modelling toolbox (weeks 5-8, 12)
- Inference (weeks 9-11)

Previously, in MBML...

- Representation (weeks 2-4)
  - PGM basics
  - Conditional independence
  - Generative processes
  - Joint probability distribution
  - Priors and likelihood
  - Factorization

$$p(\boldsymbol{\beta}, \mathbf{z}, \mathbf{k}, \mathbf{dt}) = p(\boldsymbol{\beta}|\sigma_\beta) \prod_{n=1}^{N} p(k_n|\mathbf{x}_n, \boldsymbol{\beta}, \sigma_\epsilon)\, p(z_n|\pi)\, p(dt_n|z_n, k_n)$$

1. Draw a pair of parameters[1], $\boldsymbol{\beta} \sim \mathcal{N}(\mathbf{0}, I\sigma_\beta)$
2. For $n = 1..N$
   1. Draw one value for z_n, such that $z_n \sim Bern(\pi)$.
      - If $z_n = 1$, the bus has stopped ($z_n = 0$ otherwise)
      - Distributed as Bernoulli, with parameter π
   2. Draw one value for k_n, such that $k_n \sim \mathcal{N}(\mathbf{x}_n^T\boldsymbol{\beta}, \sigma_\epsilon)$
   3. If $z_n = 1$, $dt_n = k_n$,
      - otherwise $dt_n = 0$



Previously, in MBML...

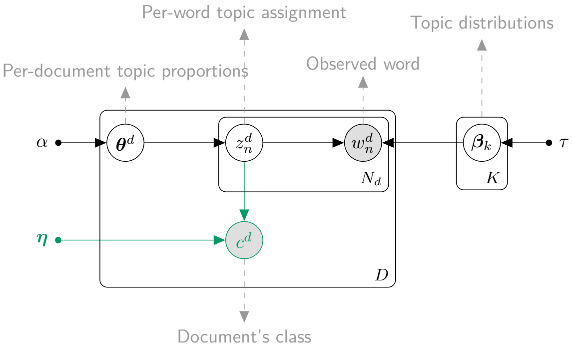
- Representation (weeks 2-4)
- Modelling toolbox (weeks 5-8, 12)
    - Mixture models
    - Different likelihoods (Gaussian, Poisson, etc.)
    - Link functions (log, softmax, Probit, etc.)
    - Non-linear relationships (e.g. using neural networks)
    - Discrete vs. continuous target variables
    - Heteroscedastic models
    - Hierarchical models
    - Temporal models (continuous and discrete)
    - Topic modelling (discrete data; e.g. text corpora)
    - Bayesian non-parametric models (week 13)
- Inference (weeks 9-11)

Previously, in MBML...

- Representation (weeks 2-4)
- Modelling toolbox (weeks 5-8, 12)
    - Bayesian Gaussian Mixture models
    - Bayesian Linear regression
    - Poisson regression
    - Heteroscedastic regression
    - Bayesian Logistic regression
    - Bayesian Probit regression
    - Hierarchical Logistic regression
    - Autoregressive models
    - Linear dynamical systems (e.g. Kalman filter)
    - Hidden Markov models
    - Latent Dirichlet allocation
    - Gaussian processes (week 13)
- Inference (weeks 9-11)

Previously, in MBML...

- Representation (weeks 2-4)
- Modelling toolbox (weeks 5-8, 12)
  - Mix & Match (e.g. LDA + (Logistic) Regression = Supervised LDA)

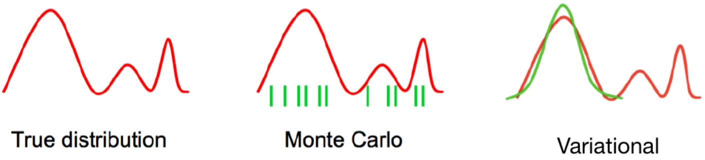


$$p(c^d|\bar{\mathbf{z}}, \boldsymbol{\eta}) = \frac{\exp(\boldsymbol{\eta}_c^T \bar{\mathbf{z}})}{\sum_{l=1}^{C} \exp(\boldsymbol{\eta}_l^T \bar{\mathbf{z}})}$$

- Inference (weeks 9-11)

Previously, in MBML...

- Representation (weeks 2-4)
- Modelling toolbox (weeks 5-8, 12)
- Inference (weeks 9-11)
  - Exact inference
    - Analytical (Bayes' rule)
    - Variable elimination
    - Belief propagation
  - Approximate inference
    - Stochastic methods - Markov chain Monte Carlo (MCMC)
    - Deterministic methods - Variational inference (VI)



Exact inference

- Computation of the exact posterior probability distribution over the variables of interest
    - Best possible solution, given data and specification
- Analytical derivations:
    - High computational efficiency
    - However, quite often, not possible at all
      (when it is not possible, we say that it is *intractable*)
- Algorithmic methods
    - Variable elimination
    - Message passing

Analytical derivation

- The Gaussian form has very nice properties[1]
- A multivariate Gaussian with d dimensions is:

$$\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{D/2}|\boldsymbol{\Sigma}|^{1/2}} e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})}$$

- The inverse of the covariance matrix is $\boldsymbol{\Lambda} = \boldsymbol{\Sigma}^{-1}$ is called *precision matrix*
- If you have a marginal Gaussian distribution for $\mathbf{x}$ and a conditional Gaussian distribution of $\mathbf{y}$ given $\mathbf{x}$ in the form

$$p(\mathbf{x}) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Lambda}^{-1})$$
$$p(\mathbf{y}|\mathbf{x}) = \mathcal{N}(\mathbf{y}|\mathbf{A}\mathbf{x} + \mathbf{b}, \mathbf{L}^{-1})$$

- Then the marginal distribution for $\mathbf{y}$ and a conditional Gaussian distribution of $\mathbf{x}$ given $\mathbf{y}$ have the form

$$p(\mathbf{y}) = \mathcal{N}(\mathbf{y}|\mathbf{A}\boldsymbol{\mu} + \mathbf{b}, \mathbf{L}^{-1} + \mathbf{A}\boldsymbol{\Lambda}^{-1}\mathbf{A}^{\mathsf{T}})$$
$$p(\mathbf{x}|\mathbf{y}) = \mathcal{N}(\mathbf{x}|\boldsymbol{\Gamma}[\mathbf{A}^{\mathsf{T}}\mathbf{L}(\mathbf{y} - \mathbf{b})] + \boldsymbol{\Gamma}\boldsymbol{\mu}, \boldsymbol{\Gamma})$$

- where $\boldsymbol{\Gamma} = (\boldsymbol{\Lambda} + \mathbf{A}^{\mathsf{T}}\mathbf{L}\mathbf{A})^{-1}$

[1] Check appendix of Bishop's book for many more useful properties!

Analytical derivation

- If we have a joint Gaussian distribution $N(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$, and we define the following partitions:

$$\mathbf{x} = \begin{pmatrix} \mathbf{x}_a \\ \mathbf{x}_b \end{pmatrix}, \boldsymbol{\mu} = \begin{pmatrix} \boldsymbol{\mu}_a \\ \boldsymbol{\mu}_b \end{pmatrix}, \boldsymbol{\Sigma} = \begin{pmatrix} \boldsymbol{\Sigma}_{aa} \boldsymbol{\Sigma}_{ab} \\ \boldsymbol{\Sigma}_{ba} \boldsymbol{\Sigma}_{bb} \end{pmatrix}$$

- Then the conditional distribution $p(\mathbf{x}_a|\mathbf{x}_b)$ is given by

$$p(\mathbf{x}_a|\mathbf{x}_b) = \mathcal{N}(\bar{\boldsymbol{\mu}}, \bar{\boldsymbol{\Sigma}})$$

  with

$$\bar{\boldsymbol{\mu}} = \boldsymbol{\mu}_a + \boldsymbol{\Sigma}_{ab}\boldsymbol{\Sigma}_{bb}^{-1}(\mathbf{x}_b - \boldsymbol{\mu}_b)$$
$$\bar{\boldsymbol{\Sigma}} = \boldsymbol{\Sigma}_{aa} - \boldsymbol{\Sigma}_{ab}\boldsymbol{\Sigma}_{bb}^{-1}\boldsymbol{\Sigma}_{ba}$$

- And the marginal distribution $p(\mathbf{x}_a)$ is given by

$$p(\mathbf{x}_a) = \mathcal{N}(\mathbf{x}_a|\boldsymbol{\mu}_a, \boldsymbol{\Sigma}_{aa})$$

Analytical derivation

- If we have two (univariate) Gaussian distributions (for the **same** variable x):

$$p(x) = \mathcal{N}(x|\mu_a, \sigma_a^2), \qquad p(x) = \mathcal{N}(x|\mu_b, \sigma_b^2)$$

- Their product is:

$$p(x) = \mathcal{N}(x|\mu_{ab}, \sigma_{ab}^2)$$

- where

$$\mu_{ab} = \frac{\mu_a \sigma_b^2 + \mu_b \sigma_a^2}{\sigma_a^2 + \sigma_b^2}, \qquad \sigma_{ab}^2 = \frac{\sigma_a^2 \sigma_b^2}{\sigma_a^2 + \sigma_b^2}$$

- This is very useful to directly combine models into a single prediction (e.g. ensemble models)!

Analytical derivation

- Example of Bayesian linear regression
- The joint probability of our model is given by:

$$p(\mathbf{y}, \boldsymbol{\beta} | \mathbf{X}, \lambda, \sigma) = p(\boldsymbol{\beta} | \mathbf{0}, \lambda \mathbf{I}) \prod_{n=1}^{N} p(y_n | \boldsymbol{\beta}^T \mathbf{x}_n, \sigma^2)$$

- Since both our prior $p(\boldsymbol{\beta}|\lambda)$ and likelihood $p(y_n|\mathbf{x}_n, \boldsymbol{\beta}, \sigma)$ are Gaussian, we apply Bayes' theorem to compute posterior over $\boldsymbol{\beta}$:

$$\underbrace{p(\boldsymbol{\beta} | \mathbf{y}, \mathbf{X}, \lambda, \sigma)}_{\text{posterior}} \propto \underbrace{p(\boldsymbol{\beta} | \mathbf{0}, \lambda \mathbf{I})}_{\text{prior}} \underbrace{\prod_{n=1}^{N} p(y_n | \boldsymbol{\beta}^T \mathbf{x}_n, \sigma^2)}_{\text{likelihood}}$$

$$= \mathcal{N}(\boldsymbol{\beta} | \mathbf{0}, \lambda \mathbf{I}) \prod_{n=1}^{N} \mathcal{N}(y_n | \boldsymbol{\beta}^T \mathbf{x}_n, \sigma^2)$$

$$= \mathcal{N}(\boldsymbol{\beta} | \boldsymbol{\mu} = ?, \boldsymbol{\Sigma} = ?)$$

- **TO DO! INDIVIDUALLY OR IN PAIRS:** Find $\boldsymbol{\mu} = ?$, $\boldsymbol{\Sigma} = ?$ (Hint: look at slide 11...)

Analytical derivation

$$\underbrace{p(\boldsymbol{\beta}|\mathbf{y}, \mathbf{X}, \lambda, \sigma)}_{\text{posterior}} \propto \underbrace{p(\boldsymbol{\beta}|\mathbf{0}, \lambda\mathbf{I})}_{\text{prior}} \underbrace{\prod_{n=1}^{N} p(y_n|\boldsymbol{\beta}^T\mathbf{x}_n, \sigma^2)}_{\text{likelihood}}$$

$$= \mathcal{N}(\boldsymbol{\beta}|\mathbf{0}, \lambda\mathbf{I}) \prod_{n=1}^{N} \mathcal{N}(y_n|\boldsymbol{\beta}^T\mathbf{x}_n, \sigma^2)$$

- Notice that, because our observations y_n are i.i.d., we can re-write:

$$= \mathcal{N}(\boldsymbol{\beta}|\mathbf{0}, \lambda\mathbf{I})\, \mathcal{N}(\mathbf{y}|\mathbf{X}\boldsymbol{\beta}, \sigma^2\mathbf{I})$$
$$= \mathcal{N}(\boldsymbol{\beta}|\boldsymbol{\mu} = ?, \boldsymbol{\Sigma} = ?)$$

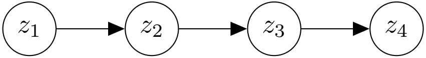
- where $\mathbf{y} = \{y_1, \ldots, y_N\}$ and $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$
- Thus, we can apply the properties in slide 11, yielding:

$$\boldsymbol{\mu} = \boldsymbol{\Sigma}\,(\sigma^{-2}\mathbf{X}^T\mathbf{y})$$
$$\boldsymbol{\Sigma} = (\lambda^{-1}\mathbf{I} + \sigma^{-2}\mathbf{X}^T\mathbf{X})^{-1}$$

Variable Elimination

simple PGM

- A chain graph



$$p(z_1, z_2, z_3, z_4) = p(z_4|z_3)\, p(z_3|z_2)\, p(z_2|z_1)\, p(z_1)$$

- Notice that: $z_4 \perp\!\!\!\perp \{z_1, z_2\}|z_3$
- We want to make inference on z_4

$$p(z_4) = \sum_{z_1, z_2, z_3} p(z_1, z_2, z_3, z_4)$$

Variable Elimination

simple PGM

$$z_1 \longrightarrow z_2 \longrightarrow z_3 \longrightarrow z_4$$

- A trivial solution is to go over all possible combinations of values!

$$p(z_4) = \sum_{z_1} \sum_{z_2} \sum_{z_3} p(z_1, z_2, z_3, z_4)$$

- Generally, if each of the m variables has k possible values, we'd have a complexity of $O(k^{m-1})$
- This quickly becomes intractable (just remember our *trivial* example with a mixture model) $\rightarrow$ NP-hard problem

Variable Elimination

simple PGM

- We can take advantage of the PGM structure

$$
\begin{aligned}
p(z_4) &= \sum_{z_1,z_2,z_3} p(z_1, z_2, z_3, z_4) \\
&= \sum_{z_1,z_2,z_3} p(z_4|z_3)\, p(z_3|z_2)\, p(z_2|z_1)\, p(z_1) \\
&= \sum_{z_3} p(z_4|z_3) \sum_{z_2} p(z_3|z_2) \sum_{z_1} p(z_2|z_1)\, p(z_1)
\end{aligned}
$$

- In a chain graph, the complexity reduces to $O(mk^2)$

Variable Elimination

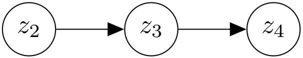
simple PGM

$$z_1 \to z_2 \to z_3 \to z_4$$

$$p(z_4) = \sum_{z_3} p(z_4|z_3) \sum_{z_2} p(z_3|z_2) \sum_{z_1} p(z_2|z_1)\, p(z_1)$$

Variable Elimination

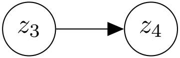
simple PGM



$$p(z_4) = \sum_{z_3} p(z_4|z_3) \sum_{z_2} p(z_3|z_2) \sum_{z_1} p(z_2|z_1)\, p(z_1)$$
$$= \sum_{z_3} p(z_4|z_3) \sum_{z_2} p(z_3|z_2)\, f_a(z_2)$$

- Notice that $f_a(z_2)$ is a function of z_2 (also called a *factor*). For example, a CPT with the probabilities of the k values of z_2
- We just "got rid" of z_1 by marginalizing over its values
- Same as in last lecture, when we marginalized over z for implementing LDA in STAN...

Variable Elimination
simple PGM



$$
\begin{aligned}
p(z_4) &= \sum_{z_3} p(z_4|z_3) \sum_{z_2} p(z_3|z_2) \sum_{z_1} p(z_2|z_1)\, p(z_1) \\
&= \sum_{z_3} p(z_4|z_3) \sum_{z_2} p(z_3|z_2)\, f_a(z_2) \\
&= \sum_{z_3} p(z_4|z_3)\, f_b(z_3)
\end{aligned}
$$

Variable Elimination

simple PGM

$$
\begin{aligned}
p(z_4) &= \sum_{z_3} p(z_4|z_3) \sum_{z_2} p(z_3|z_2) \sum_{z_1} p(z_2|z_1)\, p(z_1) \\
&= \sum_{z_3} p(z_4|z_3) \sum_{z_2} p(z_3|z_2)\, f_a(z_2) \\
&= \sum_{z_3} p(z_4|z_3)\, f_b(z_3) \\
&= f_c(z_4)
\end{aligned}
$$

Variable Elimination (VE)

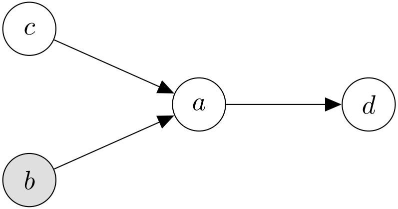
- Time complexity is exponential in size of largest factor
  - Each f_i is a factor
  - Size of factor is number of variables it depends on
- Order is vital (and sometimes a complicated problem)!
- Observed variables

$$p(z|x=k) = \frac{p(z,x=k)}{p(x=k)}$$

  - Perform VE on $p(z,x=k)$ (i.e. we fix x=k) and then on $p(x=k) = \sum_z p(z,x=k)$
- **Only acyclic graphs!**

Playtime!

- Apply the Variable Elimination algorithm to the following graph
- We want to infer $p(d|b=1)$



p(a|b,c)

	$a=0$	$a=1$
$b=0, c=0$	0.7	0.3
$b=0, c=1$	0.3	0.7
$b=1, c=0$	0.5	0.5
$b=1, c=1$	0.1	0.9

p(c)

$c=0$	$c=1$
0.7	0.3

p(b)

$b=0$	$b=1$
0.4	0.6

p(d|a)

	$d=0$	$d=1$
$a=0$	0.6	0.4
$a=1$	0.2	0.8

Polytrees

- A graph is a polytree if (and only if) there is at most one simple path between any two nodes, v_i and v_k

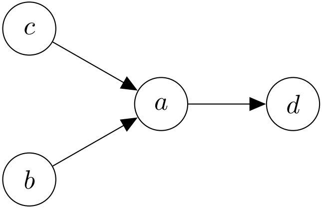


Figure: Polytree

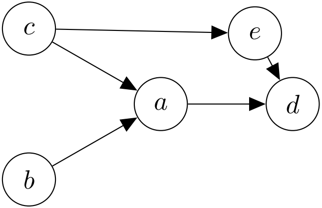


Figure: Not polytree

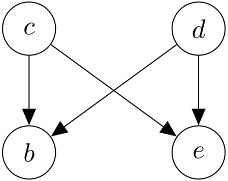


Figure: ?

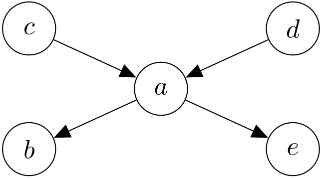


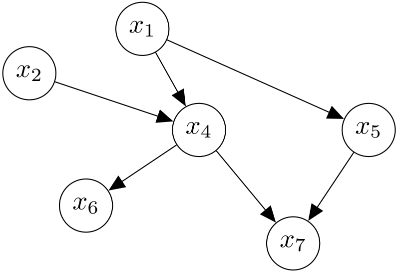
Figure: ?

Belief propagation

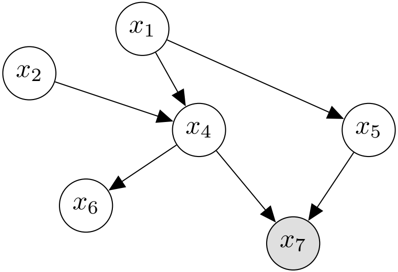
Pearl's algorithm

- We want to solve $p(\mathbf{z}|\mathbf{x})$, i.e. the conditional probability of all variables $\mathbf{z} = \{z_1, z_2, ..., z_V\}$, given evidence $\mathbf{x} = \{x_1, x_2, ..., x_E\}$
- Graph structure allows for incremental inference steps
    - Like in Variable Elimination...
- Some nodes are clearly specified from beginning
    - Evidence $x_i = k$
    - Priors
- The other nodes can build on them
- Information flows as **messages** between nodes

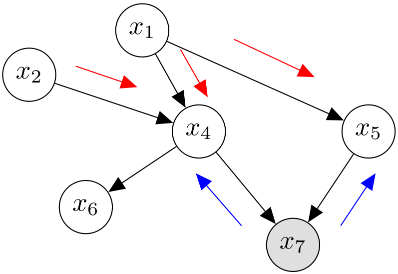
Belief propagation intuition



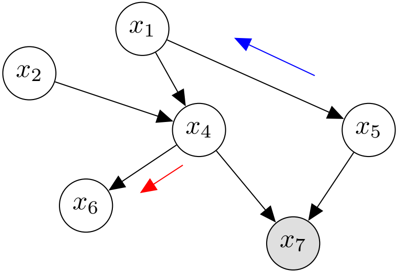
Belief propagation intuition



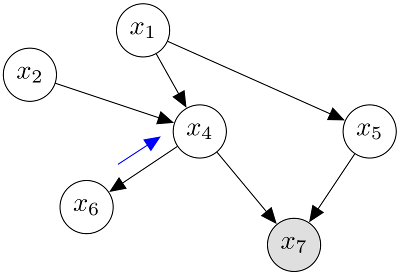
Belief propagation intuition



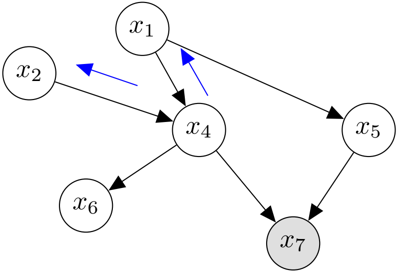
Belief propagation intuition



Belief propagation intuition

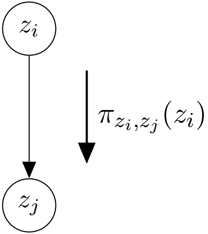


Belief propagation intuition



Belief propagation

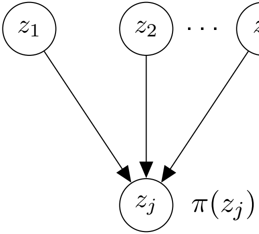
Pearl's algorithm



- π **messages** - from parent to child

Belief propagation

Pearl's algorithm

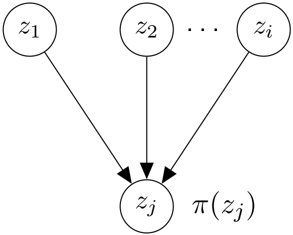


- π **messages** - from parent to children
- Represent the current belief about the parent - **causal evidence**
- After receiving all parents' messages, one can know more of the child

$$\pi(z_j) = \sum_{z_i \in \text{parent}(z_j)} p(z_j | z_1, z_2, ...) \prod_{z_i \in \text{parent}(z_j)} \pi_{z_i, z_j}(z_i)$$

Belief propagation

Pearl's algorithm



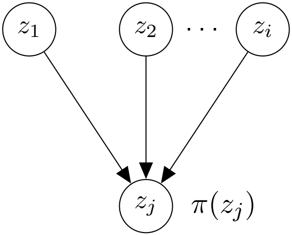
- π **messages** - from parent to children
- Represent the current belief about the parent - **causal evidence**
- After receiving all parents' messages, one can know more of the child

  factorization: $p(z_j, z_1, z_2, ..., z_j) =$
  $p(z_j|z_1, z_2, ..., z_j)p(z_1), p(z_2), ..., p(z_j)$

  $$p(z_j) = \sum_{z_1, z_2, \ldots, z_j} p(z_j|z_1, z_2, ..., z_j)p(z_1), p(z_2), ..., p(z_j)$$

Belief propagation

Pearl's algorithm

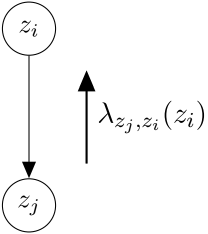


- π **messages** - from parent to children
- Represent the current belief about the parent - **causal evidence**
- After receiving all parents' messages, one can know more of the child

$$\pi(z_j) = \sum_{z_i \in \text{parent}(z_j)} p(z_j|z_1, z_2, \ldots) \prod_{z_i \in \text{parent}(z_j)} \pi_{z_i, z_j}(z_i)$$

Belief propagation

Pearl's algorithm



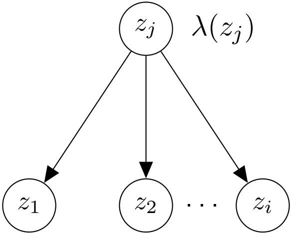
- **$\boldsymbol{\lambda}$ messages** - from child to parent

$$p(z_j|z_i)$$

- Represents the belief about the parent's value - **diagnostic evidence**

Belief propagation

Pearl's algorithm



- **$\boldsymbol{\lambda}$ messages** - from child to parent

$$p(z_i|z_j)$$

- Represents the belief about the parent's value - **diagnostic evidence**
- After receiving all children's messages, one can know more of the parent

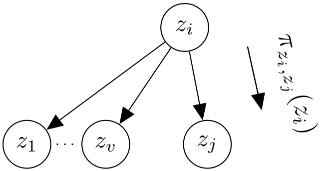
$$\lambda(z_j) = \prod_{z_i \in \text{child}(z_j)} \lambda_{z_i, z_j}(z_j)$$

Belief propagation

Pearl's algorithm

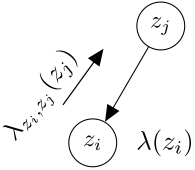
- But we still need the formulas for the messages!

$$\pi_{z_i,z_j}(z_i) = \pi(z_i) \prod_{v \neq j} \lambda_{z_v,z_i}(z_i)$$



- λ message when we have $p(z_i|z_j)$ (only one parent, z_j)

$$\lambda_{z_i,z_j}(z_j) = \sum_{z_i} \lambda(z_i)\, p(z_i|z_j)$$

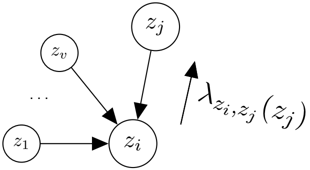


Belief propagation

Pearl's algorithm

- Generic formula for λ messages:

$$\lambda_{z_i,z_j}(z_j) = \sum_{z_i} \lambda(z_i) \sum_{z_v \in \mathbf{z} \setminus z_j} p(z_i | z_1, z_2, ...z_j) \prod_{v \neq j} \pi_{z_v,z_i}(z_v)$$



Pearl's algorithm - Initialization step

- For all $x_i \in \mathbf{x}$ (evidence nodes):
  - $\lambda(x_i) = 1$, wherever $x_i = e_i$, 0 otherwise (i.e. change the CPT)
  - $\pi(x_i) = 1$, wherever $x_i = e_i$, 0 otherwise (i.e. change the CPT)
- For all nodes, z_i, without parents:
  - $\pi(z_i) = p(z_i)$, the prior
- For all nodes, z_i, without children:
  - $\lambda(z_i) = 1$

Pearl's algorithm

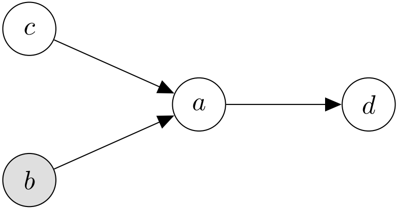
- Iterate until no change occurs

  1. For each node z_i, if it has received all π_{*,z_i} messages, calculate $\pi(z_i)$
  2. For each node z_i, if it has received all λ_{*,z_i} messages, calculate $\lambda(z_i)$
  3. For each node z_i, if $\pi(z_i)$ is calculated, and all λ_{*,z_i} messages have been received (except from z_j), calculate $\pi_{z_i,z_j}(z_i)$ and send the message to z_j
  4. For each node z_i, if $\lambda(z_i)$ is calculated, and all π_{*,z_i} messages have been received (except from z_j), calculate $\lambda_{z_i,z_j}(z_j)$ and send the message to z_j

- Compute $\mathrm{Bel}(z_i) \propto \lambda(z_i)\,\pi(z_i)$ and normalize, for all desired nodes

Example

- Let's apply Belief Propagation to the following graph
- We want to infer $p(d|b=1)$



p(a|b,c)

	$a=0$	$a=1$
$b=0, c=0$	0.7	0.3
$b=0, c=1$	0.3	0.7
$b=1, c=0$	0.5	0.5
$b=1, c=1$	0.1	0.9

p(c)

$c=0$	$c=1$
0.7	0.3

p(b)

$b=0$	$b=1$
0.4	0.6

p(d|a)

	$d=0$	$d=1$
$a=0$	0.6	0.4
$a=1$	0.2	0.8

Example

- From the initialization, we have

$\pi(c) = p(c)$

$c=0$	$c=1$
0.7	0.3

$\pi(b)$

$b=0$	$b=1$
0	1

$\lambda(b)$

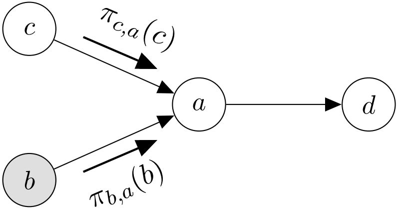
$b=0$	$b=1$
0	1

$\lambda(d)$

$b=0$	$b=1$
1	1

Example

- We start with the evidence and prior



$$\pi_{c,a}(c) = \pi(c) \prod_{x \neq a} \lambda_{x,c}(c) = \pi(c)$$

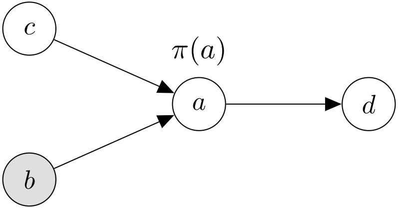
$\pi_{c,a}(c)$	
$c = 0$	$c = 1$
0.7	0.3

$$\pi_{b,a}(b) = \pi(b)$$

$\pi_{b,a}(b)$	
$b = 0$	$b = 1$
0	1

Example

- Now, $\pi(a)$



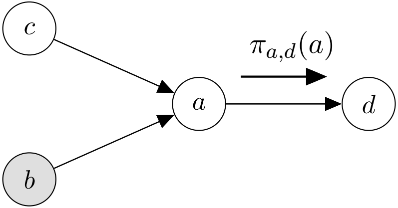
$$\pi(a) = \sum_{b,c} p(a|b,c)\,\pi_{b,a}(b)\,\pi_{c,a}(c)$$

For a=1:

$$\begin{aligned}
\pi(a=1) &= \textstyle\sum_{b=\{0,1\}} \sum_{c=\{0,1\}} p(a=1|b,c)\,\pi_{b,a}(b)\,\pi_{c,a}(c) \\
&= \textstyle\sum_{c=\{0,1\}} p(a=1|b=1,c)\,\pi_{c,a}(c) \\
&= p(a=1|b=1,c=0)\pi_{c,a}(c=0) + p(a=1|b=1,c=1)\,\pi_{c,a}(c=1) \\
&= 0.5\times 0.7 + 0.9\times 0.3 = 0.62,\ \text{so } \pi(a=1) = 0.62 \text{ and } \pi(a=0) = 0.38
\end{aligned}$$

Example

- Finally, we reach d



Since a has no other children, we have $\pi_{a,d}(a) = \pi(a)$

$\pi_{a,d}(a)$

$a=0$	$a=1$
0.38	0.62

Example

$$\pi(d) = \sum_{a=\{0,1\}} p(d|a)\, \pi_{a,d}(a)$$

$p(d|a)$

	$d=0$	$d=1$
$a=0$	0.6	0.4
$a=1$	0.2	0.8

$\pi_{a,d}(a)$

$a=0$	$a=1$
0.38	0.62

- Trivially becomes:

$\pi(d)$

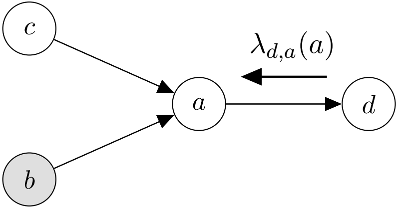
$d=0$	$d=1$
0.352	0.648

- Bel$(d) = \pi(d)\lambda(d)$
- In this case, the solution is trivially:

$$p(d|b=1) = \pi(d)$$

- But why stop here? We can propagate back, and also get p(a|b=1) and p(c|b=1)

Example



$$\lambda_{d,a}(a) = \sum_{d=\{0,1\}} \lambda(d)p(d|a)$$

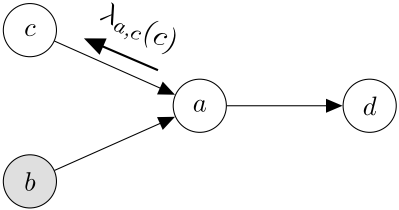
- Since $\lambda(d) = 1$, we have:

$$\lambda_{d,a}(a) = \sum_{d=\{0,1\}} p(d|a) = 1$$

- Notice that we have a unnormalized table!

$\lambda_{d,a}(a)$	
$a=0$	$a=1$
1	1

Example



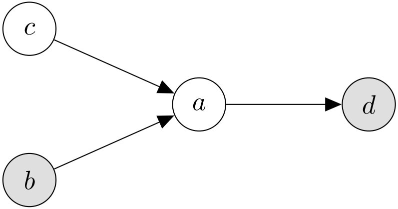
- Since $\lambda(a) = 1$, we have:

$$\lambda(c) = \lambda_{a,c}(c) = \sum_{a=\{0,1\}} \lambda(a)\, p(a|b=1,c) = \sum_{a=\{0,1\}} p(a|b=1,c) = 1$$

- Therefore:
  - $\text{Bel}(c) = \pi(c)\,\lambda(c) = \pi(c) = p(c)$
  - $\text{Bel}(a) = \pi(a)\,\lambda(a) = \pi(a) = \sum_c p(a|b=1,c)\,\pi_{c,a}(c)$, as calculated before
- Makes sense, right (notice the independence of the graph!)?

Example

- But, what if there's a new evidence, $d = 0$?



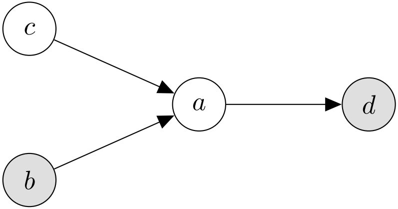
- We only need to update d and propagate back!
- A new initialization gives:

$\lambda(d) = \pi(d)$	
$d = 0$	$d = 1$
1	0

$$\lambda_{d,a}(a) = \sum_{d=\{0,1\}} \lambda(d) p(d|a) = p(d=0|a)$$

Example

- But, what if there's a new evidence, $d = 0$?



- Tables we need:

$p(d|a)$

	$d = 0$	$d = 1$
$a = 0$	0.6	0.4
$a = 1$	0.2	0.8

$\lambda(a)$

$a = 0$	$a = 1$
0.6	0.2

$$\lambda_{d,a}(a) = \sum_{d=\{0,1\}} \lambda(d) p(d|a) = p(d=0|a)$$

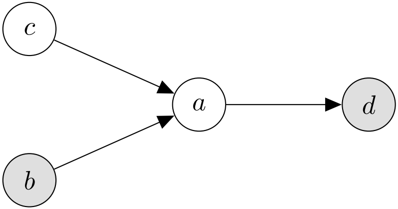
$$\lambda(a) = \lambda_{d,a}(a)$$

$$\lambda_{a,c}(c) = \sum_{a,b=\{0,1\}} \lambda(a) p(a|b,c) \pi_{b,a}(b) =$$

$$= \sum_{a=\{0,1\}} \lambda(a) p(a|b=1,c)$$

Example

- But, what if there's a new evidence, $d = 0$?



- Tables we need:

p(a|b,c)

	$a = 0$	$a = 1$
$b = 0, c = 0$	0.7	0.3
$b = 0, c = 1$	0.3	0.7
$b = 1, c = 0$	0.5	0.5
$b = 1, c = 1$	0.1	0.9

$\lambda(a)$

$a = 0$	$a = 1$
0.6	0.2

$$\lambda_{d,a}(a) = \sum_{d=\{0,1\}} \lambda(d) p(d|a) = p(d=0|a)$$

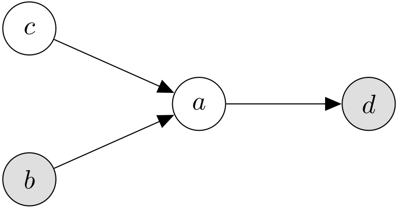
$$\lambda(a) = \lambda_{d,a}(a)$$

$$\lambda_{a,c}(c) = \sum_{a,b=\{0,1\}} \lambda(a) p(a|b,c) \pi_{b,a}(b) =$$

$$= \sum_{a=\{0,1\}} \lambda(a) p(a|b=1,c)$$

$$= 0.6 \times p(a=0|b=1,c) + 0.2 \times p(a=1|b=1,c)$$

Example



- Tables we need:

p(a|b,c)

	$a=0$	$a=1$
$b=0, c=0$	0.7	0.3
$b=0, c=1$	0.3	0.7
$b=1, c=0$	0.5	0.5
$b=1, c=1$	0.1	0.9

$\lambda(a)$

$a=0$	$a=1$
0.6	0.2

$$\lambda_{a,c}(c)=\sum_{a,b=\{0,1\}} \lambda(a)p(a|b,c)\pi_{b,a}(b)=$$
$$=\sum_{a=\{0,1\}} \lambda(a)p(a|b=1,c)$$
$$=0.6\times p(a=0|b=1,c)+0.2\times p(a=1|b=1,c)$$

For $c=0$

$$\lambda_{a,c}(c=0)=0.6\times p(a=0|b=1,c=0)+0.2\times p(a=1|b=1,c=0)$$
$$=0.6\times 0.5+0.2\times 0.5=0.4$$

For $c=1$

$$\lambda_{a,c}(c=1)=0.6\times 0.1+0.2\times 0.9=0.24$$

Example

- So,
  - $\mathsf{Bel}(a) = \alpha\,\lambda(a)\,\pi(a)$, with α the normalizing factor

$\pi(a)$

$a=0$	$a=1$
0.38	0.62

$\lambda(a)$

$a=0$	$a=1$
0.6	0.2

- $\alpha = (0.38 * 0.6 + 0.62 * 0.2)^{-1} = 2.84$

$p(a|b=1, d=0) = \mathsf{Bel}(a)$

$a=0$	$a=1$
0.65	0.35

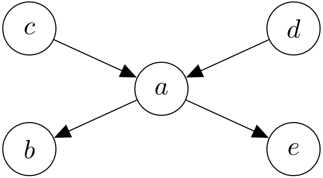
$p(c|b=1, d=0) = \mathsf{Bel}(c)$

$c=0$	$c=1$
0.8	0.2

Belief Propagation

- A *forward-backward* pass of the BP algorithm gives us the exact inference for every variable
- Updating can be done incrementally
- Plates can be treated by expansion (given that it is still a polytree)
- We used discrete variables, but the reasoning is the same for continuous
  - Summations become integrals
  - Need to derive a new function for each step (easy in some cases, particularly with exponential family)
- All of the above is valid for polytrees

Playtime!



p(c)

$c = 0$	$c = 1$
0.7	0.3

p(d)

$d = 0$	$d = 1$
0.8	0.2

p(a|c,d)

	$a = 0$	$a = 1$
$c = 0, d = 0$	0.5	0.5
$c = 0, d = 1$	0.9	0.1
$c = 1, d = 0$	0.1	0.9
$c = 1, d = 1$	0	1

p(b|a)

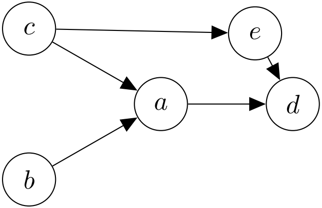
	$b = 0$	$b = 1$
$a = 0$	0.3	0.7
$a = 1$	0	1

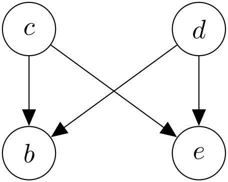
p(e|a)

	$e = 0$	$e = 1$
$a = 0$	0.2	0.8
$a = 1$	0.7	0.3

- Consider the graph and CPTs above
- Perform forward-backward Belief Propagation, assuming the evidence that $d = 1, e = 1$

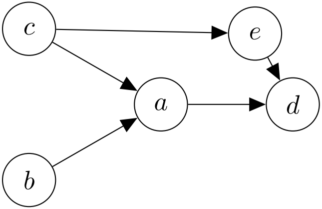
The problem of non-polytrees

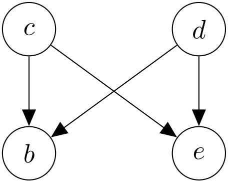




- Creates an (infinite) cycle
    - Loopy belief propagation
    - Clique Trees

The problem of non-polytrees





- Creates an (infinite) cycle
  - **Loopy belief propagation**
  - Clique Trees

Loopy Belief Propagation

- Apply BP on the original graph in the following way:
    1. Initialize all messages to 1
    2. Run BP algorithm as before (start with evidence/priors)
    3. Each node sends messages in parallel (i.e. when it sends to one chlid/parent, it sends to all children/parents)
    4. Loop until convergence
- Works very well when there are some loops, but not a fully connected graph
- With one complete loop, the MAP should be correct
- If $p(\mathbf{z})$ is jointly Gaussian, Loopy Belief Propagation will converge to the correct marginals

Some final notes

- There are actually many tools that do BP
  - Just check: https://www.cs.ubc.ca/∼ murphyk/Software/bnsoft.html
- BP is often applied on the tractable sub-parts of your model, for example combining belief from different sub-models:
  - A Random Forest, RF provides $p_{\mathsf{RF}}(y)$, and a Logistic Regression model, LR, provides $p_{\mathsf{LR}}(y)$:
  $$\mathsf{Bel}(y) = \alpha\, p_{\mathsf{RF}}(y)\, p_{\mathsf{LR}}(y)$$
  - Also called *Bayesian Model Averaging*
  - If you have gentle analytical forms for the sub-models (e.g. normal distribution for outputs), the combination is straightforward
  - You can even treat sub-models as "priors" that provide to your random variables, and combine many in a complex model!

References

- Kautz and Lowd, CSE 573: Introduction to Artificial Intelligence. https://courses.cs.washington.edu/courses/cse573/05au/10_26_pearl.ppt
- Koller and Friedman, 2009) Koller, D., and Friedman, N. Probabilistic graphical models: principles and techniques. MIT press. (2009) (Chapters 9 and 10).
- Christopher, M. Bishop. Pattern Recognition and Machine Learning. Springer-Verlag New York, 2016.