Introduction to Probabilistic Graphical Models

Christoph Lampert

IST Austria (Institute of Science and Technology Austria)

IST AUSTRIA
Institute of Science and Technology

Exponential Family Distribution

Reminder: Learning from observations

Given: a set of samples, $x^1, \dots, x^N$.

Goal: estimate $p(x)$, e.g. by maximum likelihood.

Without further assumption, maximum likelihood learning boils down to counting.

$$\hat{p}(x) = \frac{1}{N} \sum_{n=1}^{N} [\![x^n = x]\!]$$

What, if $\mathcal{X}$ is very large?

Reminder: Learning from observations

Given: a set of samples, $x^1, \ldots, x^N$.

Goal: estimate $p(x)$, e.g. by maximum likelihood.

Without further assumption, maximum likelihood learning boils down to counting.

$$\hat{p}(x) = \frac{1}{N}\sum_{n=1}^{N} [\![x^n = x]\!]$$

What, if $\mathcal{X}$ is very large?

- most $x \in \mathcal{X}$ we will never see, the others maybe once. We learn a mixture of δ peaks:

$$\hat{p}(x) = \frac{1}{N}\sum_{n=1}^{N} \delta_{x^n = x}$$

- simply assigning the others a fixed small probability (Laplace smoothing) sounds fishy

If $\mathcal{X}$ is very large, we want restrict ourselves to a suitable subset of distributions, such that the available data suffices to estimate a good model out of all. What's a suitable parameterization?

Principle of Parsimoney, aka Occam's razor

Principle of Parsimoney, aka Occam's razor

"Pluralitas non est ponenda sine neccesitate."

William of Ockham

Principle of Parsimoney, aka Occam's razor

"Pluralitas non est ponenda sine neccesitate."

William of Ockham

"We are to admit no more causes of natural things than such as are both true and sufficient to explain their appearances."

Isaac Newton

Principle of Parsimoney, aka Occam's razor

"Pluralitas non est ponenda sine neccesitate."

William of Ockham

"We are to admit no more causes of natural things than such as are both true and sufficient to explain their appearances."

Isaac Newton

"Make everything as simple as possible, but not simpler."

(paraphrasing) Albert Einstein

Principle of Parsimoney, aka Occam's razor

"Pluralitas non est ponenda sine neccesitate."

William of Ockham

"We are to admit no more causes of natural things than such as are both true and sufficient to explain their appearances."

Isaac Newton

"Make everything as simple as possible, but not simpler."

(paraphrasing) Albert Einstein

"Use the simplest explanation that explains all relevant facts."

what we'll use

- 1) Define what aspects we consider relevant facts about the data.
- 2) Pick the simplest distribution reflecting that.

- 1) Define what aspects we consider relevant facts about the data.
- 2) Pick the simplest distribution reflecting that.

Simplicity $\equiv$ Entropy

The *simplicity* of a distribution p is given by its entropy:

$$H(p) = -\sum_{z \in \mathcal{Z}} p(z) \log p(z)$$

A mixture of δ-peaks has low entropy, a uniform distribution has high entropy.

- 1) Define what aspects we consider relevant facts about the data.
- 2) Pick the simplest distribution reflecting that.

Simplicity $\equiv$ Entropy

The *simplicity* of a distribution p is given by its entropy:

$$H(p) = -\sum_{z \in \mathcal{Z}} p(z) \log p(z)$$

A mixture of δ-peaks has low entropy, a uniform distribution has high entropy.

Relevant Facts $\equiv$ Feature Functions

Let $\phi_i : \mathcal{Z} \to \mathbb{R}$ for $i = 1, \ldots, d$ denote a set of feature functions that express all properties we want to be able to model about our data.

For example:

- the grayvalue of a pixel,
- length of the contour of a shape,
- the time of day an image was taken,
- if a word starts with a capital letter.

Maximum Entropy Principle

Let $z^1, \dots, z^N$ be samples from a distribution $d(z)$. Let $\phi_1, \dots, \phi_D$ be feature functions, and denote by $\mu_i := \frac{1}{N}\sum_n \phi_i(z^n)$ their average over the sample set.
The maximum entropy distribution, p, is the solution to

$$\max_{p \text{ is a prob.distr.}} H(p) \quad \text{subject to} \quad \mathbb{E}_{z\sim p(z)}\{\phi_i(z)\} = \mu_i.$$

Maximum Entropy Principle

Let $z^1, \dots, z^N$ be samples from a distribution $d(z)$. Let $\phi_1, \dots, \phi_D$ be feature functions, and denote by $\mu_i := \frac{1}{N}\sum_n \phi_i(z^n)$ their average over the sample set.
The maximum entropy distribution, p, is the solution to

$$\underbrace{\max_{p \text{ is a prob.distr.}} H(p)}_{\text{be as simple as possible}} \quad \text{subject to} \quad \underbrace{\mathbb{E}_{z\sim p(z)}\{\phi_i(z)\} = \mu_i.}_{\text{be faithful to what we know}}$$

Maximum Entropy Principle

Let $z^1, \dots, z^N$ be samples from a distribution $d(z)$. Let $\phi_1, \dots, \phi_D$ be feature functions, and denote by $\mu_i := \frac{1}{N}\sum_n \phi_i(z^n)$ their average over the sample set.
The maximum entropy distribution, p, is the solution to

$$\underbrace{\max_{p \text{ is a prob.distr.}} H(p)}_{\text{be as simple as possible}} \quad \text{subject to} \quad \underbrace{\mathbb{E}_{z\sim p(z)}\{\phi_i(z)\} = \mu_i.}_{\text{be faithful to what we know}}$$

That sounds restrictive. What, if we want to preserve more than the mean, e.g. variance?

Maximum Entropy Principle

Let $z^1, \dots, z^N$ be samples from a distribution $d(z)$. Let $\phi_1, \dots, \phi_D$ be feature functions, and denote by $\mu_i := \frac{1}{N}\sum_n \phi_i(z^n)$ their average over the sample set.
The maximum entropy distribution, p, is the solution to

$$\underbrace{\max_{p \text{ is a prob.distr.}} H(p)}_{\text{be as simple as possible}} \quad \text{subject to} \quad \underbrace{\mathbb{E}_{z\sim p(z)}\{\phi_i(z)\} = \mu_i.}_{\text{be faithful to what we know}}$$

That sounds restrictive. What, if we want to preserve more than the mean, e.g. variance?

Just define a suitable feature function: $\phi'(z) = \phi(z)^2$

Finding the Maximum Entropy Distribution

- Given: samples $z^1, \dots, z^N$ and feature functions $\phi_1, \dots, \phi_D$
- Define: $\mu_i := \frac{1}{N} \sum_n \phi_i(z^n)$
- Task: find maximum entropy distribution $p(z)$, i.e. solve

$$\max_p \quad -\sum_z p(z) \log p(z) \quad \text{subject to} \quad \mathbb{E}_{z \sim p(z)}\{\phi_i(z)\} = \mu_i \quad \text{for } i = 1, \dots, d.$$

Finding the Maximum Entropy Distribution

- Given: samples $z^1, \dots, z^N$ and feature functions $\phi_1, \dots, \phi_D$
- Define: $\mu_i := \frac{1}{N} \sum_n \phi_i(z^n)$
- Task: find maximum entropy distribution $p(z)$, i.e. solve

$$\max_p \quad -\sum_z p(z) \log p(z) \quad \text{subject to} \quad \mathbb{E}_{z \sim p(z)}\{\phi_i(z)\} = \mu_i \quad \text{for } i = 1, \dots, d.$$

Lagrangian:

$$\mathfrak{L}(p, \theta, \lambda) = -\sum_z p(z) \log p(z) - \sum_{i=1}^{d} \theta_i \big(\mathbb{E}_{z \sim p(z)}\{\phi_i(z)\} - \mu_i\big) - \lambda\big(\sum_z p(z) - 1\big)$$

Finding the Maximum Entropy Distribution

- Given: samples $z^1, \dots, z^N$ and feature functions $\phi_1, \dots, \phi_D$
- Define: $\mu_i := \frac{1}{N} \sum_n \phi_i(z^n)$
- Task: find maximum entropy distribution $p(z)$, i.e. solve

$$\max_p \quad -\sum_z p(z) \log p(z) \quad \text{subject to} \quad \mathbb{E}_{z \sim p(z)}\{\phi_i(z)\} = \mu_i \quad \text{for } i = 1, \dots, d.$$

Lagrangian:

$$\mathfrak{L}(p, \theta, \lambda) = -\sum_z p(z) \log p(z) - \sum_{i=1}^{d} \theta_i \big(\mathbb{E}_{z \sim p(z)}\{\phi_i(z)\} - \mu_i \big) - \lambda \big(\sum_z p(z) - 1 \big)$$

Solution (see blackboard):

$$p(z) = \frac{1}{Z} \exp\Big(\sum_{i=1}^{d} \theta_i \phi_i(z) \Big) \quad \text{with} \quad Z = \sum_{z \in \mathcal{Z}} \exp\Big(\sum_{i=1}^{d} \theta_i \phi_i(z) \Big)$$

for some values of $\theta_1, \dots, \theta_d$ (that depend on $\mu_1, \dots, \mu_d$, of course)

Exponential Family Distribution

For feature functions $\phi_1, \ldots, \phi_D$, the set of distributions

$$p(z;\theta) = \frac{1}{Z(\theta)} \exp\Big(\sum_{i=1}^{d} \theta_i \phi_i(z)\Big) \quad \text{with} \quad Z(\theta) = \sum_{z\in\mathcal{Z}} \exp\Big(\sum_{i=1}^{d} \theta_i \phi_i(z)\Big)$$

is called **exponential family distribution** with features $\phi_1, \ldots, \phi_d$.

Often, we use vector notation: $\phi(z) = (\phi_1(z), \ldots, \phi_D(z))$ and $\theta = (\theta_1, \ldots, \theta_D)$, such that

$$p(z;\theta) = \frac{1}{Z(\theta)} \exp\Big(\sum_{i=1}^{d} \theta^\top \phi(z)\Big) \quad \text{with} \quad Z(\theta) = \sum_{z\in\mathcal{Z}} \exp\Big(\sum_{i=1}^{d} \theta^\top \phi(z)\Big)$$

The exponential family distribution makes a natural parameterization for learning.
Given $z^1, \ldots, z^n$, the best $\theta_1, \ldots, \theta_D$ are unknown, but we know the functional form of $p(z)$.

Example: Exponential Family Distribution

Example:

- Let $\mathcal{Z} = \mathbb{R}$, $\phi_1(z) = z$, $\phi_2(z) = z^2$.
- The exponential family distribution is

$$p(z) = \frac{1}{Z(\theta_1, \theta_2)} \exp(\theta_1 z + \theta_2 z^2)$$

Example: Exponential Family Distribution

Example:

- Let $\mathcal{Z} = \mathbb{R}$, $\phi_1(z) = z$, $\phi_2(z) = z^2$.
- The exponential family distribution is

$$\begin{aligned} p(z) &= \frac{1}{Z(\theta_1, \theta_2)} \exp(\ \theta_1 z + \theta_2 z^2) \\ &= \frac{b^2 a}{Z(a, b)} \exp(\ a\big(z - b\big)^2\ ) \quad \text{for } a = \theta_2,\ b = -\frac{\theta_1}{\theta_2}. \end{aligned}$$

It's a Gaussian!

Example: Exponential Family Distribution

Example:

- Let $\mathcal{Z} = \mathbb{R}$, $\phi_1(z) = z$, $\phi_2(z) = z^2$.
- The exponential family distribution is

$$
\begin{aligned}
p(z) &= \frac{1}{Z(\theta_1, \theta_2)} \exp(\ \theta_1 z + \theta_2 z^2) \\
&= \frac{b^2 a}{Z(a, b)} \exp(\ a(z - b)^2\ ) \quad \text{for } a = \theta_2,\ b = -\frac{\theta_1}{\theta_2}.
\end{aligned}
$$

It's a Gaussian!

- Given examples $z^1, \dots, z^N$, we can compute a and b, and derive θ.

Example: Exponential Family Distribution

Example:

- Let $\mathcal{Z} = \{1, \ldots, K\}$, $\phi_k(z) = [\![z = k]\!]$, for $k = 1, \ldots, K$.
- The exponential family distribution is

$$p(z) = \frac{1}{Z} \exp\Big(\sum_i \theta_i \phi_i(z) \Big)$$

Example: Exponential Family Distribution

Example:

- Let $\mathcal{Z} = \{1, \ldots, K\}$, $\phi_k(z) = [\![z = k]\!]$, for $k = 1, \ldots, K$.
- The exponential family distribution is

$$p(z) = \frac{1}{Z} \exp\Big(\sum_i \theta_i \phi_i(z) \Big) = \begin{cases} \exp(\theta_1)/Z & \text{for } z = 1, \\ \exp(\theta_2)/Z & \text{for } z = 2, \\ \ldots & \\ \exp(\theta_K)/Z & \text{for } z = K. \end{cases}$$

with $Z = \exp(\theta_1) + \cdots + \exp(\theta_K)$.

It's a Multinomial!

Example: Exponential Family Distribution

- Let $\mathcal{Z} = \{0,1\}^{N \times M}$ image grid,
- let $\phi_i(z) = z_i$, for each pixel i,
- let $\phi_0(z) = \sum_{(i,j)\in\mathcal{E}} [\![z_i \neq z_j]\!]$ (summing over all 4-neighbor pairs) $\rightarrow$ boundary length
- The exponential family distribution is

$$
\begin{aligned}
p(z) &= \frac{1}{Z(\theta)} \exp\Big(\sum_i \theta_i \phi_i(z) + \theta_0 \phi_0(z) \Big) \\
&= \frac{1}{Z(\theta)} \exp\Big(\sum_i \theta_i z_i + \theta_0 \sum_{i,j} [\![z_i \neq z_j]\!] \Big)
\end{aligned}
$$

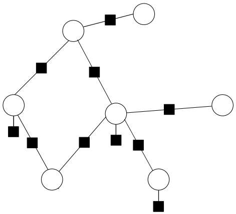
It's a Markov Random Field! with unary and pairwise factors.

Probabilistic Inference in Factor Graphs

Probabilistic Inference

We return to more general graphical models, given by a factor graph:

$$p(y_1, \ldots, y_n) = \frac{1}{Z} \prod_{F \in \mathcal{F}} \phi_F(y_F)$$
$$= \frac{1}{Z} e^{-E(y)} = \frac{1}{Z} e^{-\sum_{F \in \mathcal{F}} E_F(y_F)}$$



with $\qquad y_F = (y_{f_1}, \ldots, y_{f_{|F|}}) \quad \text{for} \quad F = (f_1, \ldots, f_{|F|})$

$$Z = \sum_{y_1, \ldots, y_n} \prod_{F \in \mathcal{F}} \phi_F(y_F)$$

Inference tasks:

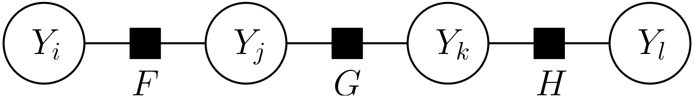
- compute $p(y_i)$ or $p(y_F)$ for some i or F
- compute $p(y_F | y_G)$ for some F, G
- compute Z

Probabilistic Inference – Overview

- Exact Inference
  - Belief Propagation on chains
  - Belief Propagation on trees
  - Junction tree algorithm
- Approximate Inference
  - Loopy Belief Propagation
  - Sampling / MCMC (next year)
  - Variational Inference / Mean Field (next year)

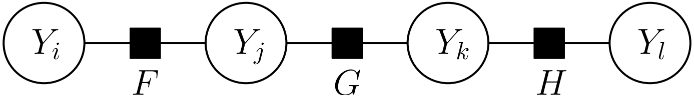
Probabilistic Inference – Belief Propagation

Assume $y = (y_i, y_j, y_k, y_l)$, $\mathcal{Y} = \mathcal{Y}_i \times \mathcal{Y}_j \times \mathcal{Y}_k \times \mathcal{Y}_l$ for finite $\mathcal{Y}_i, \mathcal{Y}_j, \mathcal{Y}_k, \mathcal{Y}_l$, and $p(y) \propto \phi(y)$ for $\phi(y) = \phi_F(y_i, y_j)\phi_G(y_j, y_k)\phi_H(y_k, y_l)$ compatible with the following factor graph:



Probabilistic Inference – Belief Propagation

Assume $y = (y_i, y_j, y_k, y_l)$, $\mathcal{Y} = \mathcal{Y}_i \times \mathcal{Y}_j \times \mathcal{Y}_k \times \mathcal{Y}_l$ for finite $\mathcal{Y}_i, \mathcal{Y}_j, \mathcal{Y}_k, \mathcal{Y}_l$, and $p(y) \propto \phi(y)$ for $\phi(y) = \phi_F(y_i, y_j)\phi_G(y_j, y_k)\phi_H(y_k, y_l)$ compatible with the following factor graph:

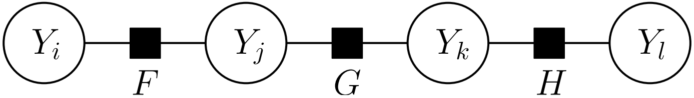


Task 1: for any $y \in \mathcal{Y}$, compute $p(y)$, using

$$p(y) = \frac{1}{Z}\phi(y)$$

Probabilistic Inference – Belief Propagation

Assume $y = (y_i, y_j, y_k, y_l)$, $\mathcal{Y} = \mathcal{Y}_i \times \mathcal{Y}_j \times \mathcal{Y}_k \times \mathcal{Y}_l$ for finite $\mathcal{Y}_i, \mathcal{Y}_j, \mathcal{Y}_k, \mathcal{Y}_l$, and $p(y) \propto \phi(y)$ for $\phi(y) = \phi_F(y_i, y_j)\phi_G(y_j, y_k)\phi_H(y_k, y_l)$ compatible with the following factor graph:



Task 1: for any $y \in \mathcal{Y}$, compute $p(y)$, using

$$p(y) = \frac{1}{Z}\phi(y)$$

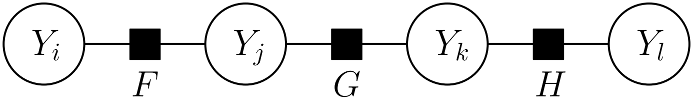
Problem: We don't know Z, and computing it using

$$Z = \sum_{y \in \mathcal{Y}} \phi(y)$$

looks expensive (the sum has $|\mathcal{Y}_i| \cdot |\mathcal{Y}_j| \cdot |\mathcal{Y}_k| \cdot |\mathcal{Y}_l|$ terms).

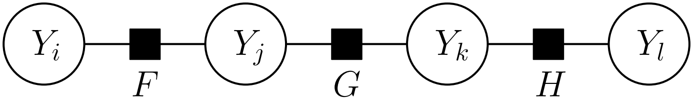
A lot research has been done on how to **efficiently compute (or approximate)** Z.

Probabilistic Inference – Belief Propagation



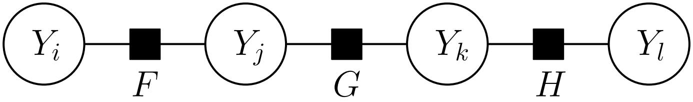
$$Z = \sum_{y \in \mathcal{Y}} \phi(y)$$

Probabilistic Inference – Belief Propagation



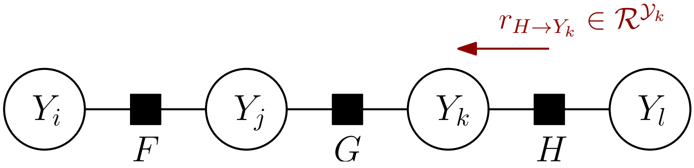
$$Z = \sum_{y \in \mathcal{Y}} \phi(y) = \sum_{y_i \in \mathcal{Y}_i} \sum_{y_j \in \mathcal{Y}_j} \sum_{y_k \in \mathcal{Y}_k} \sum_{y_l \in \mathcal{Y}_l} \phi_F(y_i, y_j) \phi_G(y_j, y_k) \phi_H(y_k, y_m)$$

Probabilistic Inference – Belief Propagation



$$
\begin{aligned}
Z = \sum_{y \in \mathcal{Y}} \phi(y) &= \sum_{y_i \in \mathcal{Y}_i} \sum_{y_j \in \mathcal{Y}_j} \sum_{y_k \in \mathcal{Y}_k} \sum_{y_l \in \mathcal{Y}_l} \phi_F(y_i, y_j)\phi_G(y_j, y_k)\phi_H(y_k, y_m) \\
&= \sum_{y_i} \sum_{y_j} \phi_F(y_i, y_j) \sum_{y_k} \phi_G(y_j, y_k) \sum_{y_l} \phi_H(y_k, y_m)
\end{aligned}
$$

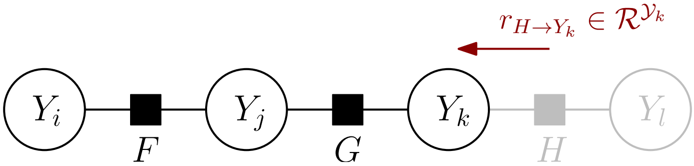
Probabilistic Inference – Belief Propagation



$$Z = \sum_{y\in\mathcal{Y}} \phi(y) = \sum_{y_i\in\mathcal{Y}_i} \sum_{y_j\in\mathcal{Y}_j} \sum_{y_k\in\mathcal{Y}_k} \sum_{y_l\in\mathcal{Y}_l} \phi_F(y_i, y_j)\phi_G(y_j, y_k)\phi_H(y_k, y_m)$$

$$= \sum_{y_i} \sum_{y_j} \phi_F(y_i, y_j) \sum_{y_k} \phi_G(y_j, y_k) \underbrace{\sum_{y_l} \phi_H(y_k, y_m)}_{r_{H\to Y_k}(y_k)}$$

Probabilistic Inference – Belief Propagation

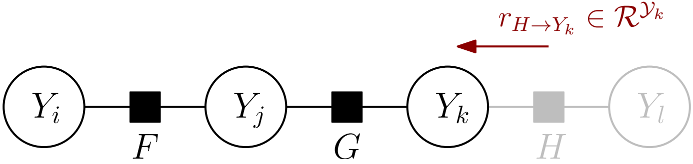


$$Z = \sum_{y\in\mathcal{Y}} \phi(y) = \sum_{y_i\in\mathcal{Y}_i} \sum_{y_j\in\mathcal{Y}_j} \sum_{y_k\in\mathcal{Y}_k} \sum_{y_l\in\mathcal{Y}_l} \phi_F(y_i, y_j)\phi_G(y_j, y_k)\phi_H(y_k, y_m)$$

$$= \sum_{y_i} \sum_{y_j} \phi_F(y_i, y_j) \sum_{y_k} \phi_G(y_j, y_k) \underbrace{\sum_{y_l} \phi_H(y_k, y_m)}_{r_{H\to Y_k}(y_k)}$$

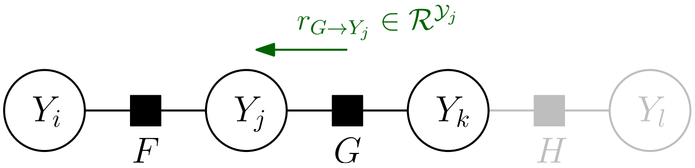
$$= \sum_{y_i} \sum_{y_j} \phi_F(y_i, y_j) \sum_{y_k} \phi_G(y_j, y_k) r_{H\to Y_k}(y_k)$$

Probabilistic Inference – Belief Propagation



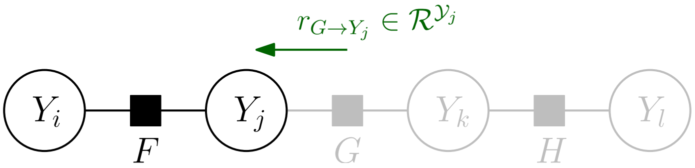
$$Z = \sum_{y_i} \sum_{y_j} \phi_F(y_i, y_j) \sum_{y_k} \phi_G(y_j, y_k) r_{H \to Y_k}(y_k)$$

Probabilistic Inference – Belief Propagation



$$Z = \sum_{y_i} \sum_{y_j} \phi_F(y_i, y_j) \underbrace{\sum_{y_k} \phi_G(y_j, y_k) r_{H\to Y_k}(y_k)}_{r_{G\to Y_j}(y_j)}$$

Probabilistic Inference – Belief Propagation

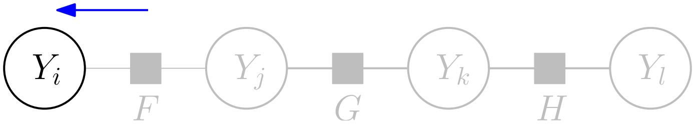


$$Z = \sum_{y_i} \sum_{y_j} \phi_F(y_i, y_j) \underbrace{\sum_{y_k} \phi_G(y_j, y_k) r_{H \to Y_k}(y_k)}_{r_{G \to Y_j}(y_j)}$$

$$= \sum_{y_i} \sum_{y_j} \phi_F(y_i, y_j) r_{G \to Y_j}(y_j)$$

Probabilistic Inference – Belief Propagation

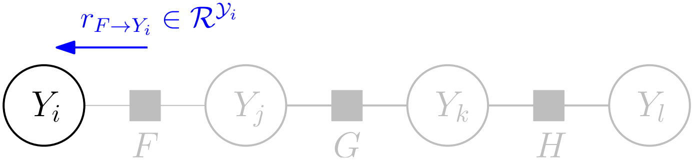
$r_{F \to Y_i} \in \mathcal{R}^{\mathcal{Y}_i}$



$$Z = \sum_{y_i} \sum_{y_j} \phi_F(y_i, y_j) \underbrace{\sum_{y_k} \phi_G(y_j, y_k) r_{H \to Y_k}(y_k)}_{r_{G \to Y_j}(y_j)}$$

$$= \sum_{y_i} \underbrace{\sum_{y_j} \phi_F(y_i, y_j) r_{G \to Y_j}(y_j)}_{r_{F \to Y_i}(y_i)}$$

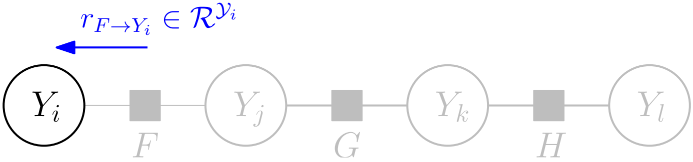
Probabilistic Inference – Belief Propagation



$$Z = \sum_{y_i} \sum_{y_j} \phi_F(y_i, y_j) \underbrace{\sum_{y_k} \phi_G(y_j, y_k) r_{H\to Y_k}(y_k)}_{r_{G\to Y_j}(y_j)}$$

$$= \sum_{y_i} \underbrace{\sum_{y_j} \phi_F(y_i, y_j) r_{G\to Y_j}(y_j)}_{r_{F\to Y_i}(y_i)} = \sum_{y_i} r_{F\to Y_i}(y_i)$$

Probabilistic Inference – Belief Propagation

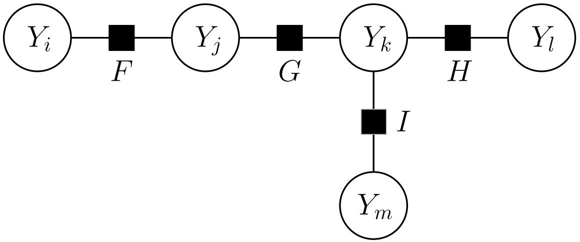


$$Z = \sum_{y_i} \sum_{y_j} \phi_F(y_i, y_j) \underbrace{\sum_{y_k} \phi_G(y_j, y_k) r_{H\to Y_k}(y_k)}_{r_{G\to Y_j}(y_j)}$$

$$= \sum_{y_i} \underbrace{\sum_{y_j} \phi_F(y_i, y_j) r_{G\to Y_j}(y_j)}_{r_{F\to Y_i}(y_i)} = \sum_{y_i} r_{F\to Y_i}(y_i)$$

Total effort for n variables and L states per variable: $O(nL^2)$ instead of $O(L^n)$

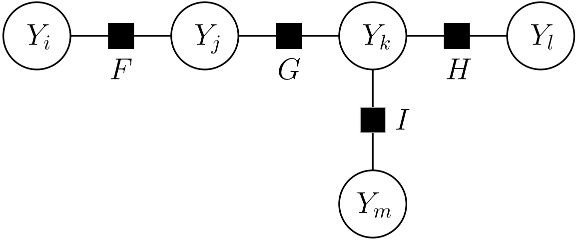
Example: Inference on Trees



1) pick a root (here: i)
2) sort sums such that parents nodes are left of their children

$$Z = \sum_{y \in \mathcal{Y}} \phi(y)$$

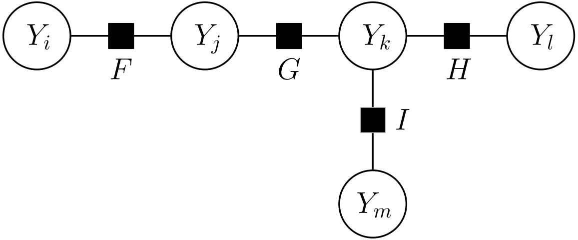
Example: Inference on Trees



1) pick a root (here: i)
2) sort sums such that parents nodes are left of their children

$$Z = \sum_{y\in\mathcal{Y}} \phi(y) = \sum_{y_i\in\mathcal{Y}_i} \sum_{y_j\in\mathcal{Y}_j} \sum_{y_k\in\mathcal{Y}_k} \sum_{y_l\in\mathcal{Y}_l} \sum_{y_m\in\mathcal{Y}_m} \phi_F(y_i, y_j)\phi_G(y_j, y_k)\phi_H(y_k, y_l)\phi_I(y_k, y_m)$$

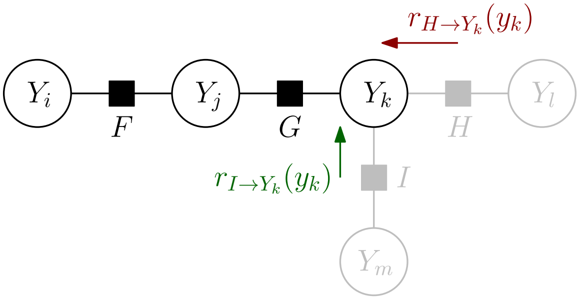
Example: Inference on Trees



1) pick a root (here: i)
2) sort sums such that parents nodes are left of their children

$$
\begin{aligned}
Z &= \sum_{y \in \mathcal{Y}} \phi(y) = \sum_{y_i \in \mathcal{Y}_i} \sum_{y_j \in \mathcal{Y}_j} \sum_{y_k \in \mathcal{Y}_k} \sum_{y_l \in \mathcal{Y}_l} \sum_{y_m \in \mathcal{Y}_m} \phi_F(y_i, y_j)\phi_G(y_j, y_k)\phi_H(y_k, y_l)\phi_I(y_k, y_m) \\
&= \sum_{y_i \in \mathcal{Y}_i} \sum_{y_j \in \mathcal{Y}_j} \phi_F(y_i, y_j) \sum_{y_k \in \mathcal{Y}_k} \phi_G(y_j, y_k) \Big[\sum_{y_l \in \mathcal{Y}_l} \phi_H(y_k, y_l) \Big] \Big[\sum_{y_m \in \mathcal{Y}_m} \phi_I(y_k, y_m) \Big]
\end{aligned}
$$

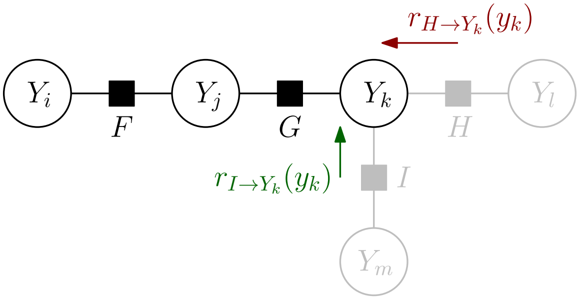
Example: Inference on Trees



1) pick a root (here: i)
2) sort sums such that parents nodes are left of their children

$$
\begin{aligned}
Z &= \sum_{y\in\mathcal{Y}} \phi(y) = \sum_{y_i\in\mathcal{Y}_i}\sum_{y_j\in\mathcal{Y}_j}\sum_{y_k\in\mathcal{Y}_k}\sum_{y_l\in\mathcal{Y}_l}\sum_{y_m\in\mathcal{Y}_m} \phi_F(y_i,y_j)\phi_G(y_j,y_k)\phi_H(y_k,y_l)\phi_I(y_k,y_m) \\
&= \sum_{y_i\in\mathcal{Y}_i}\sum_{y_j\in\mathcal{Y}_j} \phi_F(y_i,y_j)\sum_{y_k\in\mathcal{Y}_k} \phi_G(y_j,y_k) \underbrace{\left[\sum_{y_l\in\mathcal{Y}_l}\phi_H(y_k,y_l)\right]}_{r_{H\to Y_k}(y_k)} \underbrace{\left[\sum_{y_m\in\mathcal{Y}_m}\phi_I(y_k,y_m)\right]}_{r_{I\to Y_k}(y_k)}
\end{aligned}
$$

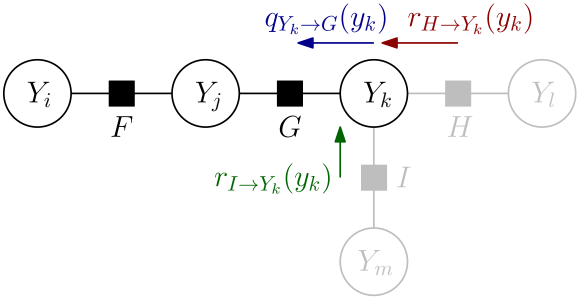
Example: Inference on Trees



1) pick a root (here: i)
2) sort sums such that parents nodes are left of their children

$$Z = \sum_{y_i \in \mathcal{Y}_i} \sum_{y_j \in \mathcal{Y}_j} \phi_F(y_i, y_j) \sum_{y_k \in \mathcal{Y}_k} \phi_G(y_j, y_k) \; r_{H \to Y_k}(y_k) r_{I \to Y_k}(y_k)$$

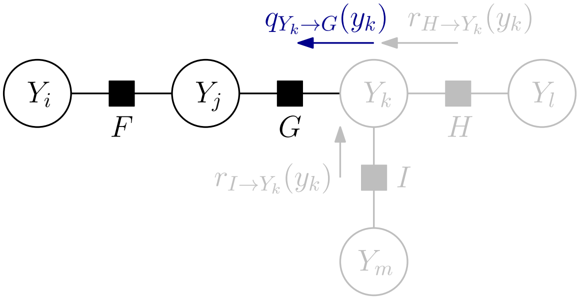
Example: Inference on Trees



1) pick a root (here: i)
2) sort sums such that parents nodes are left of their children

$$Z = \sum_{y_i \in \mathcal{Y}_i} \sum_{y_j \in \mathcal{Y}_j} \phi_F(y_i, y_j) \sum_{y_k \in \mathcal{Y}_k} \phi_G(y_j, y_k) \underbrace{r_{H \to Y_k}(y_k) r_{I \to Y_k}(y_k)}_{q_{Y_k \to G}(y_k)}$$

Example: Inference on Trees

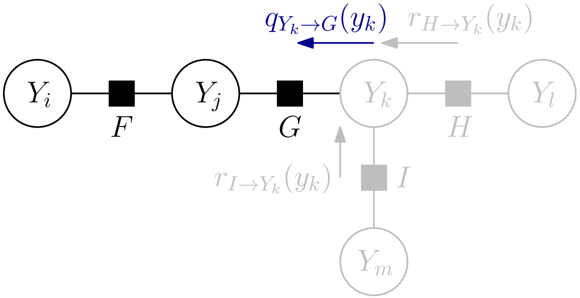


1) pick a root (here: i)
2) sort sums such that parents nodes are left of their children

$$Z = \sum_{y_i \in \mathcal{Y}_i} \sum_{y_j \in \mathcal{Y}_j} \phi_F(y_i, y_j) \sum_{y_k \in \mathcal{Y}_k} \phi_G(y_j, y_k) \underbrace{r_{H \to Y_k}(y_k) r_{I \to Y_k}(y_k)}_{q_{Y_k \to G}(y_k)}$$

$$Z = \sum_{y_i \in \mathcal{Y}_i} \sum_{y_j \in \mathcal{Y}_j} \phi_F(y_i, y_j) \sum_{y_k \in \mathcal{Y}_k} \phi_G(y_j, y_k) q_{Y_k \to G}(y_k)$$

Example: Inference on Trees



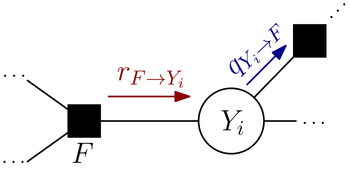
1) pick a root (here: i)
2) sort sums such that parents nodes are left of their children

$$Z = \sum_{y_i \in \mathcal{Y}_i} \sum_{y_j \in \mathcal{Y}_j} \phi_F(y_i, y_j) \sum_{y_k \in \mathcal{Y}_k} \phi_G(y_j, y_k) \underbrace{r_{H \to Y_k}(y_k) r_{I \to Y_k}(y_k)}_{q_{Y_k \to G}(y_k)}$$

$$\begin{aligned} Z &= \sum_{y_i \in \mathcal{Y}_i} \sum_{y_j \in \mathcal{Y}_j} \phi_F(y_i, y_j) \sum_{y_k \in \mathcal{Y}_k} \phi_G(y_j, y_k) q_{Y_k \to G}(y_k) \\ &= \ldots \end{aligned}$$

Factor Graph Sum-Product Algorithm

"Message": pair of vectors at each factor graph edge $(i, F) \in \mathcal{E}$



1) $r_{F \to Y_i} \in \mathbb{R}^{\mathcal{Y}_i}$: factor-to-variable message

$$r_{F \to Y_i}(y_i) = \sum_{y_F \in \mathcal{Y}_F} \phi_F(y_F) \prod_{j:(j,F) \in \mathcal{E} \setminus \{i\}} q_{Y_j \to F}$$

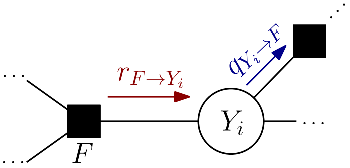
2) $q_{Y_i \to F} \in \mathbb{R}^{\mathcal{Y}_i}$: variable-to-factor message

$$q_{Y_i \to F}(y_i) = \prod_{G:(i,G) \in \mathcal{E} \setminus \{F\}} r_{G \to Y_i}$$

- Algorithm updates messages from root to leafs

Factor Graph Sum-Product Algorithm

"Message": pair of vectors at each factor graph edge $(i, F) \in \mathcal{E}$



1) $r_{F \to Y_i} \in \mathbb{R}^{\mathcal{Y}_i}$: factor-to-variable message

$$r_{F \to Y_i}(y_i) = \sum_{y_F \in \mathcal{Y}_F} \phi_F(y_F) \prod_{j:(j,F) \in \mathcal{E} \setminus \{i\}} q_{Y_j \to F}$$

2) $q_{Y_i \to F} \in \mathbb{R}^{\mathcal{Y}_i}$: variable-to-factor message

$$q_{Y_i \to F}(y_i) = \prod_{G:(i,G) \in \mathcal{E} \setminus \{F\}} r_{G \to Y_i}$$

- Algorithm updates messages from root to leafs

(Sum-Product) Belief Propagation

Factor Graph Sum-Product Algorithm

- After termination: Z, $p(y_i)$ and $p(y_F)$ can be obtained from the messages

$$Z = \sum_{y_{\text{root}}} \prod_{F:(\text{root},F)\in\mathcal{E}} r_{F\rightarrow Y_{\text{root}}}(y_{\text{root}})$$

$$p(Y_i = y_i) \propto \prod_{F:(i,F)\in\mathcal{E}} r_{F\rightarrow Y_i}(y_i)$$

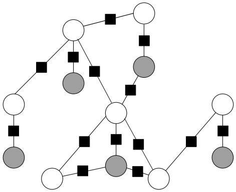
$$p(Y_F = y_F) \propto e^{-E_F(y_F)} \prod_{i:(i,F)\in\mathcal{E}} q_{Y_i\rightarrow F}(y_i)$$

Normalization constants by explicit summation over $y_i \in \mathcal{Y}_i$ or $y_F \in \mathcal{Y}_F$.

Probabilistic Inference

What, if distribution is conditioned on data $x = (x_1, \dots, x_m)$?

$$
\begin{aligned}
p(y_1, \dots, y_n | x_1, \dots, x_m) &= \frac{1}{Z(x)} \prod_{F \in \mathcal{F}} \phi_F(y_F, x_F) \\
Z(x) &= \sum_{y_1, \dots, y_n} \prod_{F \in \mathcal{F}} \phi_F(y_F, x_F)
\end{aligned}
$$



Inference tasks:

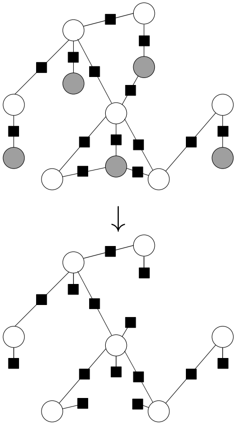
- compute $Z(x)$ or $p(y_i|x)$ or $p(y_F|x)$ for some i or F

Probabilistic Inference

What, if distribution is conditioned on data $x = (x_1, \ldots, x_m)$?

$$p(y_1, \ldots, y_n | x_1, \ldots, x_m) = \frac{1}{Z(x)} \prod_{F \in \mathcal{F}} \phi_F(y_F, x_F)$$

$$Z(x) = \sum_{y_1, \ldots, y_n} \prod_{F \in \mathcal{F}} \phi_F(y_F, x_F)$$



Inference tasks:

- compute $Z(x)$ or $p(y_i|x)$ or $p(y_F|x)$ for some i or F

Reduce to unconditioned case:

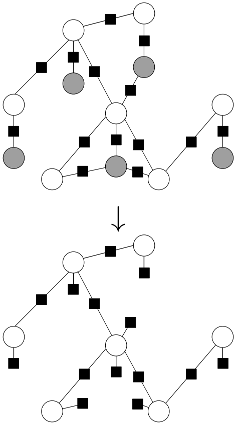
- define new factor graph: $\tilde{F}(y_F) \leftarrow F(y_F, x_F)$, $\tilde{Z} \leftarrow Z(x)$, ...

Probabilistic Inference

What, if distribution is conditioned on data $x = (x_1, \ldots, x_m)$?

$$p(y_1, \ldots, y_n | x_1, \ldots, x_m) = \frac{1}{Z(x)} \prod_{F \in \mathcal{F}} \phi_F(y_F, x_F)$$

$$Z(x) = \sum_{y_1, \ldots, y_n} \prod_{F \in \mathcal{F}} \phi_F(y_F, x_F)$$



Inference tasks:

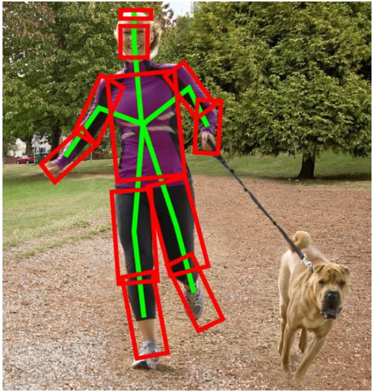
- compute $Z(x)$ or $p(y_i|x)$ or $p(y_F|x)$ for some i or F

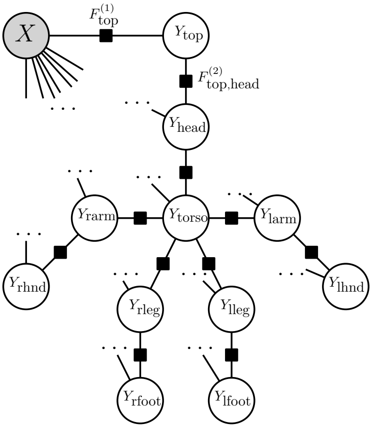
Reduce to unconditioned case:

- define new factor graph: $\tilde{F}(y_F) \leftarrow F(y_F, x_F)$, $\tilde{Z} \leftarrow Z(x)$, ...

All computation is performed on the new graph. Only its topology (cyclic or not) matters.

Example: Pictorial Structures





- **Tree-structured model** for articulated pose
  (Felzenszwalb and Huttenlocher, 2000), (Fischler and Elschlager, 1973)
- Belief propagation is the state-of-the-art for prediction and inference

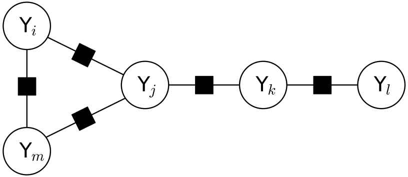
Example: Pictorial Structures

- Marginal probabilities $p(y_i|x)$ give us
  - potential positions
  - uncertainty

  of the body parts.

Belief Propagation in Cyclic Graphs?

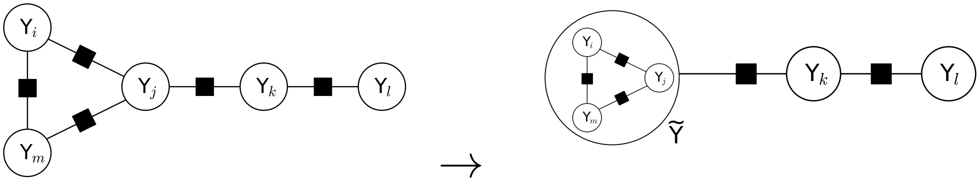
Belief propagation does not work for graph with cycles.



Belief Propagation in Cyclic Graphs?

Belief propagation does not work for graph with cycles.

We can construct equivalent chain/tree models:



$\tilde{Y} = (Y_i, Y_j, Y_m)$ with state space $\widetilde{\mathcal{Y}} = \mathcal{Y}_i \times \mathcal{Y}_j \times \mathcal{Y}_m$

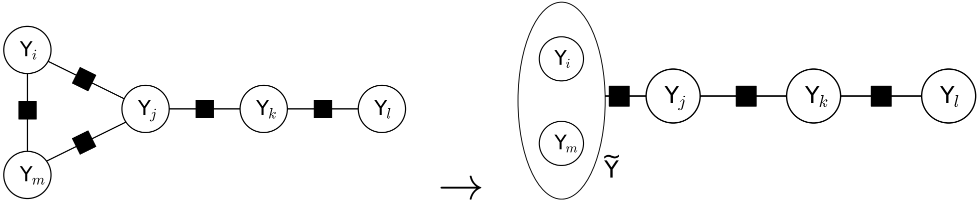
General procedure: **junction tree algorithm**

Problem: exponentially growing state space $\rightarrow$ BP often gets inefficient

Belief Propagation in Cyclic Graphs?

Belief propagation does not work for graph with cycles.

We can construct equivalent chain/tree models:



$\tilde{Y} = (Y_i, Y_m)$ with state space $\tilde{\mathcal{Y}} = \mathcal{Y}_i \times \mathcal{Y}_m$

General procedure: **junction tree algorithm**

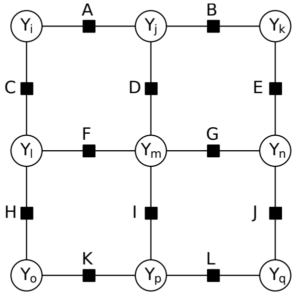
Problem: exponentially growing state space → BP often gets inefficient

Belief Propagation in Cyclic Graphs

Can we do belief propagation even for graphs with cycles? Messages can still be computed:

1) factor-to-variable message $r_{F \to Y_i}(y_i) = \sum_{y_F} \phi_F(y_F) \prod_{j:(j,F) \in \mathcal{E} \setminus \{i\}} q_{Y_j \to F}$

2) variable-to-factor message $q_{Y_i \to F}(y_i) = \prod_{G:(i,G) \in \mathcal{E} \setminus \{F\}} r_{G \to Y_i}$



Belief Propagation in Cyclic Graphs

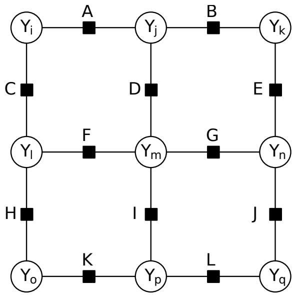
Can we do belief propagation even for graphs with cycles? Messages can still be computed:

1) factor-to-variable message $r_{F \to Y_i}(y_i) = \sum_{y_F} \phi_F(y_F) \prod_{j:(j,F) \in \mathcal{E} \setminus \{i\}} q_{Y_j \to F}$

2) variable-to-factor message $q_{Y_i \to F}(y_i) = \prod_{G:(i,G) \in \mathcal{E} \setminus \{F\}} r_{G \to Y_i}$

Problem: no *leaf–to–root* order

$\rightarrow$ where to start? when to terminate?



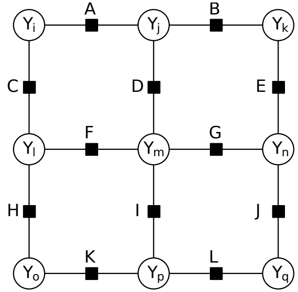
Belief Propagation in Cyclic Graphs

Can we do belief propagation even for graphs with cycles? Messages can still be computed:

1) factor-to-variable message $r_{F\to Y_i}(y_i) = \sum_{y_F} \phi_F(y_F) \prod_{j:(j,F)\in\mathcal{E}\setminus\{i\}} q_{Y_j\to F}$

2) variable-to-factor message $q_{Y_i\to F}(y_i) = \prod_{G:(i,G)\in\mathcal{E}\setminus\{F\}} r_{G\to Y_i}$

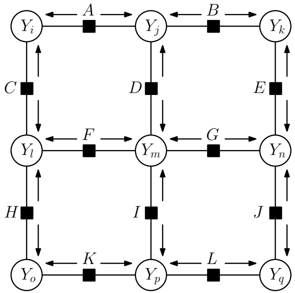
Problem: no *leaf–to–root* order
$\to$ where to start? when to terminate?

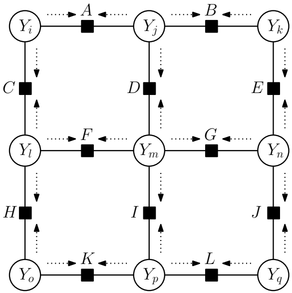


Loopy Belief Propagation (LBP)

- initialize all messages as constant 1
- pass messages using rules of BP until a stop criterion

Belief Propagation in Cyclic Graphs





Problems:

- loopy BP might not converge (*e.g.* messages can oscillate)
- even if it does, the computed probabilities are only approximate.

Several improved schemes exist, some even convergent (but approximate)

(Exact) inference in general graphs is **#P-hard**.