

# ECE 590.17: Lecture 6 – Learning Factor Graph Parameters

Graphical Models and Inference for Machine Learning

Duke University, Spring 2026

**History:** Written by Henry Pfister (2026)

**Last Modified:** 02/20/2026

## Outline of lecture:

6.1	Outline	1
6.1.1	Notation and setup	2
6.2	Warm-up: MLE in Bayesian networks	3
6.3	Factor graphs: learning factor node functions	4
6.3.1	Energy-factor notation and exponential-family parameterization	4
6.3.2	Log-likelihood for fully observed data	6
6.3.3	Learning on trees: exact BP in the inner loop	7
6.3.4	Learning on loopy graphs: approximate inference in the inner loop	7
6.4	Missing data / latent variables: EM with BP	8
6.4.1	EM lower bound and the Q-function	8
6.4.2	EM gradient form and “soft” moment matching	10
6.4.3	EM + BP: algorithm template	10
6.5	Example: EM for a discrete mixture model	11
6.6	Example: EM on a hidden Markov model (HMM)	12
6.7	Tree factor graphs revisited: learning by counting	12
6.7.1	Example: learning on a single 3-cycle using (loopy) BP	14
6.8	Summary: what to remember	15
6.9	Exercises	15

## 6.1 Outline

In this lecture we focus on *parameter learning* for graphical models when the *graphical structure is known*. That is, we assume we are given a factor graph (or a Bayesian network DAG, or a Markov random field graph), and we want to learn the *local factor functions* from data.

**What students should already know.** We assume you are familiar with: (i) the definition of factor graphs and how a global function factors as a product of local functions; (ii) the definition of a probability distribution from an unnormalized factorization  $\mu(x) = \frac{1}{Z} \prod_a f_a(x_{\partial a})$ ; (iii) belief propagation (BP) on trees for computing marginals and partition functions; (iv) basic probability and the phrase “maximum likelihood,” but not necessarily how to compute maximum likelihood estimates in practice.

**What we will do.** We will treat three increasingly challenging settings:

1. **Directed models (Bayesian networks), fully observed data.** We will derive the maximum likelihood estimator (MLE) for discrete conditional probability tables (CPTs). This case is conceptually important and computationally easy.

2. **Undirected / factor graph models, fully observed data.** We will see that MLE becomes an optimization problem involving the partition function. The gradient takes a simple “data moment minus model moment” form, but computing the model moment requires inference. On trees, BP gives exact inference, so learning is efficient. On loopy graphs, we use approximations (loopy BP, pseudo-likelihood, sampling, etc.).
3. **Missing data / latent variables: EM using BP.** When some variables are hidden, the log-likelihood contains a log-sum over hidden configurations. Expectation–Maximization (EM) converts this into alternating steps: the E-step runs inference (BP) to compute expected sufficient statistics, and the M-step updates parameters (often in closed form for discrete factors).

**One unifying theme.** Learning in graphical models often reduces to repeated inference. So: *inference is the inner loop, learning is the outer loop.*

### 6.1.1 Notation and setup

Let  $X = (X_1, \dots, X_n)$  be discrete random variables with finite alphabets  $\mathcal{X}_i$  and product alphabet  $\mathcal{X} = \mathcal{X}_1 \times \dots \times \mathcal{X}_n$ . The goal is to learn a model from an observed i.i.d. dataset

$$\mathcal{D} = \{x^{(1)}, \dots, x^{(N)}\}, \quad x^{(\ell)} \in \mathcal{X}.$$

For any subvector  $X_S$ , we write the empirical probability under  $\mathcal{D}$  of  $x_S \in \mathcal{X}_S := \prod_{i \in S} \mathcal{X}_i$  by

$$\hat{p}_{X_S}^{\mathcal{D}}(x_S) := \frac{1}{N} \sum_{\ell=1}^N \mathbb{I}\{x_S^{(\ell)} = x_S\}.$$

Thus, we can write the empirical average of an arbitrary function  $g : \mathcal{X} \rightarrow \mathbb{R}$  as

$$\hat{\mathbb{E}}_{\mathcal{D}}[g(X)] := \frac{1}{N} \sum_{\ell=1}^N g(x^{(\ell)}) = \sum_{x \in \mathcal{X}} \hat{p}_X^{\mathcal{D}}(x) g(x).$$

**Definition 1** (Maximum likelihood estimation). Given a parametric family  $\{p_X^\theta(x)\}_{\theta \in \Theta}$  of distributions on  $\mathcal{X}$ , the *log-likelihood of the dataset  $\mathcal{D}$*  is given by

$$\mathcal{L}(\theta) = \ln \prod_{\ell=1}^N p_X^\theta(x^{(\ell)}) = \sum_{\ell=1}^N \ln p_X^\theta(x^{(\ell)}).$$

A *maximum likelihood estimate* of  $\theta$  satisfies

$$\hat{\theta}_{\text{ML}} \in \arg \max_{\theta} \mathcal{L}(\theta).$$

**Why the logarithm?** The logarithm turns products into sums and connects to cross-entropy / KL divergence: maximizing likelihood is equivalent to minimizing  $D(\hat{p}_X^{\mathcal{D}} \| p_X^\theta)$ :

$$\begin{aligned} \frac{1}{N} \mathcal{L}(\theta) &= \frac{1}{N} \sum_{\ell=1}^N \ln p_X^\theta(x^{(\ell)}) \\ &= \sum_{x \in \mathcal{X}} \hat{p}_X^{\mathcal{D}}(x) \ln p_X^\theta(x) \\ &= - \sum_{x \in \mathcal{X}} \hat{p}_X^{\mathcal{D}}(x) \ln \frac{1}{\hat{p}_X^{\mathcal{D}}(x)} - \sum_{x \in \mathcal{X}} \hat{p}_X^{\mathcal{D}}(x) \ln \frac{\hat{p}_X^{\mathcal{D}}(x)}{p_X^\theta(x)} \\ &= -H(\hat{p}_X^{\mathcal{D}}) - D(\hat{p}_X^{\mathcal{D}} \| p_X^\theta). \end{aligned}$$

## 6.2 Warm-up: MLE in Bayesian networks

It can be easier to start with directed models, because the normalization is local.

**Bayesian network factorization.** Let  $G$  be a DAG on nodes  $[n]$ . Write  $\text{pa}(i)$  for the parent set of node  $i$ . A Bayesian network specifies the joint distribution as

$$p_X(x) = \prod_{i=1}^n p_{X_i|X_{\text{pa}(i)}}(x_i | x_{\text{pa}(i)}).$$

For discrete variables, each conditional can be defined by a table of probabilities. We parameterize the local conditional  $p_{\theta_i}$  by non-negative real numbers  $\{\theta_i(x_i | x_{\text{pa}(i)})\}$  for  $x_i \in \mathcal{X}_i$  and  $x_{\text{pa}(i)} \in \mathcal{X}_{\text{pa}(i)}$ . These unnormalized numbers represent the conditional distribution over  $x_i$  given  $x_{\text{pa}(i)}$  via

$$p_{X_i|X_{\text{pa}(i)}}^{\theta_i}(x_i | x_{\text{pa}(i)}) = \frac{\theta_i(x_i | x_{\text{pa}(i)})}{\sum_{x' \in \mathcal{X}_i} \theta_i(x' | x_{\text{pa}(i)})}.$$

**Log-likelihood decomposes into local terms.** Using this, the normalized log-likelihood of the dataset can be written as

$$\begin{aligned} \frac{1}{N} \mathcal{L}(\theta) &= \frac{1}{N} \ln \prod_{\ell=1}^N \prod_{i=1}^n \frac{\theta_i(x_i^{(\ell)} | x_{\text{pa}(i)}^{(\ell)})}{\sum_{x' \in \mathcal{X}_i} \theta_i(x' | x_{\text{pa}(i)}^{(\ell)})} \\ &= \frac{1}{N} \sum_{\ell=1}^N \sum_{i=1}^n \ln \frac{\theta_i(x_i^{(\ell)} | x_{\text{pa}(i)}^{(\ell)})}{\sum_{x' \in \mathcal{X}_i} \theta_i(x' | x_{\text{pa}(i)}^{(\ell)})} \\ &= \widehat{\mathbb{E}}_{\mathcal{D}} \left[ \sum_{i=1}^n \ln \frac{\theta_i(x_i^{(\ell)} | x_{\text{pa}(i)}^{(\ell)})}{\sum_{x' \in \mathcal{X}_i} \theta_i(x' | x_{\text{pa}(i)}^{(\ell)})} \right] \\ &= \sum_{i=1}^n \widehat{\mathbb{E}}_{\mathcal{D}} \left[ \ln \frac{\theta_i(x_i^{(\ell)} | x_{\text{pa}(i)}^{(\ell)})}{\sum_{x' \in \mathcal{X}_i} \theta_i(x' | x_{\text{pa}(i)}^{(\ell)})} \right] \\ &= \sum_{i=1}^n \widehat{\mathbb{E}}_{\mathcal{D}} \left[ \ln \frac{\theta_i(x_i^{(\ell)} | x_{\text{pa}(i)}^{(\ell)})}{\sum_{x' \in \mathcal{X}_i} \theta_i(x' | x_{\text{pa}(i)}^{(\ell)})} \cdot \frac{\widehat{p}_{X_i|X_{\text{pa}(i)}}^{\mathcal{D}}(x_i | x_{\text{pa}(i)}^{(\ell)})}{\widehat{p}_{X_i|X_{\text{pa}(i)}}^{\mathcal{D}}(x_i | x_{\text{pa}(i)}^{(\ell)})} \right] \\ &= - \sum_{i=1}^n \widehat{\mathbb{E}}_{\mathcal{D}} \left[ \ln \frac{\widehat{p}_{X_i|X_{\text{pa}(i)}}^{\mathcal{D}}(x_i^{(\ell)} | x_{\text{pa}(i)}^{(\ell)})}{\theta_i(x_i^{(\ell)} | x_{\text{pa}(i)}^{(\ell)})} \right] - \sum_{i=1}^n H(\widehat{p}_{X_i|X_{\text{pa}(i)}}^{\mathcal{D}}) - \sum_{i=1}^n \widehat{\mathbb{E}}_{\mathcal{D}} \left[ \ln \sum_{x' \in \mathcal{X}_i} \theta_i(x' | x_{\text{pa}(i)}^{(\ell)}) \right] \\ &= - \sum_{i=1}^n D(\widehat{p}_{X_i|X_{\text{pa}(i)}}^{\mathcal{D}} \| p_{X_i|X_{\text{pa}(i)}}^{\theta_i}) - \sum_{i=1}^n H(\widehat{p}_{X_i|X_{\text{pa}(i)}}^{\mathcal{D}}) - \sum_{i=1}^n \widehat{\mathbb{E}}_{\mathcal{D}} \left[ \ln \sum_{x' \in \mathcal{X}_i} \theta_i(x' | x_{\text{pa}(i)}^{(\ell)}) \right]. \end{aligned}$$

The sum of KL divergences in the first term is independent of the  $\theta_i$  normalizations and minimized if and only if  $\widehat{p}_{X_i|X_{\text{pa}(i)}}^{\mathcal{D}} = p_{X_i|X_{\text{pa}(i)}}^{\theta_i}$  for all  $i \in [n]$ . The second term is independent of  $\theta$ . The last term is zero if we choose all the  $\theta_i$  parameters to define normalized distributions. This implies the following simple rule for MLE of Bayesian networks when the graph is known.

**Closed-form MLE for conditional probability tables.** Fix a node  $i$  and a parent configuration  $u \in \mathcal{X}_{\text{pa}(i)}$ . Define parent-child configuration counts

$$N_i(x_i, x_{\text{pa}(i)}) := \sum_{\ell=1}^N \mathbb{I}\{x_i^{(\ell)} = x_i, x_{\text{pa}(i)}^{(\ell)} = x_{\text{pa}(i)}\},$$

This implies that

$$\hat{\theta}_i^{\text{ML}}(x_i | u) = \frac{N_i(x_i, u)}{\sum_{x \in \mathcal{X}_i} N_i(x, u)}.$$

So the MLE rule of thumb is: *normalize the empirical conditional frequencies.*

*Remark 2* (Why this is easy). Bayesian networks have built-in local normalization: each factor is a conditional distribution, so there is no global partition function. As a result, the MLE is just counting and normalizing.

**Smoothing and zero counts.** If  $N_i(u) = 0$  for some parent configuration  $u$ , the MLE is undefined. Even when  $N_i(u)$  is small, the MLE can overfit. A common fix is Laplace/Dirichlet smoothing:

$$\hat{\theta}_i^{\text{Dir}(\alpha)}(x_i | u) = \frac{N_i(x_i, u) + \alpha}{\sum_{x \in \mathcal{X}_i} (N_i(x, u) + \alpha)},$$

which corresponds to a symmetric Dirichlet prior with pseudo-count  $\alpha$ . We will not emphasize Bayesian learning today, but it is worth remembering this practical issue.

## 6.3 Factor graphs: learning factor node functions

We now switch to the factor-graph viewpoint. Let  $G = (V, F, E)$  be a factor graph with variable nodes  $i \in V = [n]$  and factor nodes  $a \in F$ . In contrast to earlier descriptions, this note does not require each variable to use the same alphabet. For each factor node  $a$ , let  $\partial a \subseteq V$  denote its neighboring variable indices, and write

$$x_{\partial a} := (x_i)_{i \in \partial a} \in \mathcal{X}_{\partial a} := \prod_{i \in \partial a} \mathcal{X}_i.$$

### 6.3.1 Energy-factor notation and exponential-family parameterization

We will write the probability of a configuration  $x \in \mathcal{X}$  under the assumed model as a product of *factor functions*

$$p_X^\theta(x) = \frac{1}{Z(\theta)} \prod_{a \in F} f_a(x_{\partial a}; \theta_a), \quad Z(\theta) := \sum_{x \in \mathcal{X}} \prod_{a \in F} f_a(x_{\partial a}; \theta_a).$$

The quantity  $Z(\theta)$  is the *partition function* and is the source of most computational difficulty in undirected models.

*Remark 3.* Directed models such as Bayesian networks can also be written in factor-graph form. In that case, learning is simplified by the fact that  $Z(\theta) = 1$  identically.

**Local energies.** We define the local energy associated with factor  $a$  by

$$E_a(x_{\partial a}; \theta_a) := -\ln f_a(x_{\partial a}; \theta_a).$$

Using this, the global energy is

$$E_\theta(x) := \sum_{a \in F} E_a(x_{\partial a}; \theta_a),$$

and the model takes the standard Gibbs / energy-based form

$$p_X^\theta(x) = \frac{1}{Z(\theta)} \exp(-E_\theta(x)), \quad Z(\theta) = \sum_{x \in \mathcal{X}} \exp(-E_\theta(x)).$$

**Exponential-family factors.** An exponential family of distributions is defined by a set of parameters and sufficient statistics for those parameters. In this lecture, we only deal with the simple case of unrestricted discrete distributions where the parameters are the negative log probabilities of each outcome and a sufficient statistic for a dataset is its histogram.

**Definition 4.** A finite set  $\mathcal{S}$  set with  $m = |\mathcal{S}|$  is ordered if there exists a bijection  $\sigma: \mathcal{S} \rightarrow [m]$  that maps  $s \in \mathcal{S}$  to its rank in the set. With this, one can define the *indicator vector* (or one-hot encoding) of  $s \in \mathcal{S}$  via  $T: \mathcal{S} \rightarrow \mathbb{R}^m$  where  $[T(s)]_{\sigma(s)} = 1$  and  $[T(s)]_j = 0$  for all  $j \neq \sigma(s)$ . We can apply this definition to any finite set by choosing a fixed (but arbitrary) ordering for that set.

For the case of a single factor  $f(x) = e^{-E(x;\theta)}$  with  $x \in \mathcal{X}$  and parameter  $\theta \in \mathbb{R}^{|\mathcal{X}|}$ , we can define the sufficient statistic  $T: \mathcal{X} \rightarrow \mathbb{R}^{|\mathcal{X}|}$  to be the one-hot encoding of  $x \in \mathcal{X}$ . In particular, if  $\mathcal{X} = \{00, 01, 10, 11\}$ , then  $T(00) = (1, 0, 0, 0)$ ,  $T(01) = (0, 1, 0, 0)$ , etc. Then, we can write  $E(x; \theta) = -\theta^\top T(x)$  and  $f(x; \theta) = \exp(-E(x; \theta)) = \exp(\theta^\top T(x))$ .

We assume each factor lives in the canonical exponential family described above. For each  $a \in F$ , fix a sufficient-statistic function  $T_a: \mathcal{X}_{\partial a} \rightarrow \mathbb{R}^{|\mathcal{X}_{\partial a}|}$  that maps each configuration to a vector. Then, we define parameters  $\theta_a \in \mathbb{R}^{|\mathcal{X}_{\partial a}|}$  and

$$f_a(x_{\partial a}; \theta_a) = \exp(\theta_a^\top T_a(x_{\partial a})).$$

With this convention, one gets the energies

$$E_a(x_{\partial a}; \theta_a) = -\theta_a^\top T_a(x_{\partial a}).$$

It is sometimes preferred to use energies that look like positive “costs”. To do this, one can simply flip the sign by redefining  $T_a$  with a sign flip.

*Remark 5* (Indicator statistics as table parameters). If  $x_{\partial a}$  is discrete and you want a fully general factor, then take the indicator statistics: for each  $u \in \mathcal{X}_{\partial a}$  define the indicator vector  $[T_a(x_{\partial a})]_u = \mathbb{I}\{x_{\partial a} = u\}$ . Then  $f_a(x_{\partial a}) = \exp([\theta_a]_{x_{\partial a}})$  is an arbitrary positive table. There is also one extra degree of freedom because adding a constant to all  $[\theta_a]_u$  multiplies  $f_a$  by a constant and is absorbed into  $Z(\theta)$ . A common fix is  $\sum_u [\theta_a]_u = 0$  or pinning one entry (e.g.  $[\theta_a]_{u_0} = 0$ ).

Using this setup, we can write the model as

$$p_X^\theta(x) = \frac{1}{Z(\theta)} \exp\left(\sum_{b \in F} \theta_b^\top T_b(x_{\partial b})\right), \quad Z(\theta) = \sum_{x \in \mathcal{X}} \exp\left(\sum_{b \in F} \theta_b^\top T_b(x_{\partial b})\right). \quad (6.1)$$

**Lemma 6** (Gradient of the log-partition function). *For every  $a \in F$ , it follows from (6.1) that*

$$\nabla_{\theta_a} \ln Z(\theta) = \mathbb{E}_{p_X^\theta}[T_a(X_{\partial a})].$$

*Proof.* Since  $\mathcal{X}$  is finite, we may differentiate under the summation sign:

$$\begin{aligned}
 \nabla_{\theta_a} \ln Z(\theta) &= \frac{1}{Z(\theta)} \nabla_{\theta_a} Z(\theta) \\
 &= \frac{1}{Z(\theta)} \sum_{x \in \mathcal{X}} \nabla_{\theta_a} \exp\left(\sum_{b \in F} \theta_b^\top T_b(x_{\partial b})\right) \\
 &= \frac{1}{Z(\theta)} \sum_{x \in \mathcal{X}} \exp\left(\sum_{b \in F} \theta_b^\top T_b(x_{\partial b})\right) T_a(x_{\partial a}) \\
 &= \sum_{x \in \mathcal{X}} p_X^\theta(x) T_a(x_{\partial a}) \\
 &= \mathbb{E}_{p_X^\theta}[T_a(X_{\partial a})]. \quad \square
 \end{aligned}$$

### 6.3.2 Log-likelihood for fully observed data

Given an i.i.d. dataset  $\mathcal{D} = \{x^{(1)}, \dots, x^{(N)}\}$  with full observations, the log-likelihood is

$$\begin{aligned}
 \mathcal{L}(\theta) &:= \ln p_X^\theta(\mathcal{D}) = \sum_{\ell=1}^N \ln p_X^\theta(x^{(\ell)}) \\
 &= \sum_{\ell=1}^N \left[ \sum_{b \in F} \ln f_b(x_{\partial b}^{(\ell)}; \theta_b) - \ln Z(\theta) \right] \\
 &= \sum_{\ell=1}^N \left[ \sum_{b \in F} \theta_b^\top T_b(x_{\partial b}^{(\ell)}) - \ln Z(\theta) \right].
 \end{aligned}$$

Equivalently, dividing by  $N$  and using empirical averages,

$$\frac{1}{N} \mathcal{L}(\theta) = \sum_{b \in F} \theta_b^\top \widehat{\mathbb{E}}_{\mathcal{D}}[T_b(X_{\partial b})] - \ln Z(\theta).$$

The first term is easy to compute from the data. In particular, since  $T$  returns a one-hot vector, the expectation term equals the empirical distribution vector

$$\widehat{\mathbb{E}}_{\mathcal{D}}[T_b(X_{\partial b})] = (p_{X_{\partial b}}^{\mathcal{D}}(x_{\partial b}))_{x_{\partial b} \in \mathcal{X}_{\partial b}}.$$

The second term is independent of the data and depends on  $\theta$  through the partition function.

**Gradient: data moments minus model moments** Differentiate with respect to  $\theta_a$ :

$$\nabla_{\theta_a} \left( \frac{1}{N} \mathcal{L}(\theta) \right) = \widehat{\mathbb{E}}_{\mathcal{D}}[T_a(X_{\partial a})] - \nabla_{\theta_a} \ln Z(\theta).$$

From Lemma 6, we get the standard derivative identity for exponential-family energy models:

$$\nabla_{\theta_a} \ln Z(\theta) = \mathbb{E}_{p_X^\theta}[T_a(X_{\partial a})].$$

Therefore,

$$\nabla_{\theta_a} \left( \frac{1}{N} \mathcal{L}(\theta) \right) = \widehat{\mathbb{E}}_{\mathcal{D}}[T_a(X_{\partial a})] - \mathbb{E}_{p_X^\theta}[T_a(X_{\partial a})].$$

This is a standard learning approach:

update parameters to match model sufficient statistics to empirical sufficient statistics.

**Proposition 7** (Moment matching at the MLE (interior optimum)). *If  $\hat{\theta}$  maximizes  $\mathcal{L}(\theta)$  in the interior of the parameter space, then for every factor  $a$ ,*

$$\mathbb{E}_{p_X^{\hat{\theta}}}[T_a(X_{\partial a})] = \widehat{\mathbb{E}}_{\mathcal{D}}[T_a(X_{\partial a})].$$

**Inference is the bottleneck.** The empirical averages are trivial, but the model expectations  $\mathbb{E}_{p_X^{\theta}}[T_a(X_{\partial a})]$  require computing the marginal distribution of  $X_{\partial a}$  under  $p_X^{\theta}$ . This is an inference problem. Thus, in undirected / factor-graph models, *learning is an outer loop wrapped around repeated inference.*

### 6.3.3 Learning on trees: exact BP in the inner loop

If the factor graph is a tree, sum-product BP computes exact factor marginals  $p_{X_{\partial a}}^{\theta}(x_{\partial a})$  and thus

$$\mathbb{E}_{p_X^{\theta}}[T_a(X_{\partial a})] = \sum_{x_{\partial a}} p_{X_{\partial a}}^{\theta}(x_{\partial a}) T_a(x_{\partial a})$$

exactly and efficiently.

A simple gradient-ascent scheme is therefore:

$$\theta_a^{(t+1)} = \theta_a^{(t)} + \eta_t \left( \widehat{\mathbb{E}}_{\mathcal{D}}[T_a(X_{\partial a})] - \mathbb{E}_{p_X^{\theta^{(t)}}}[T_a(X_{\partial a})] \right),$$

where the step size  $\eta_t > 0$  can always be chosen small enough that  $\mathcal{L}(\theta^{(t+1)}) > \mathcal{L}(\theta^{(t)})$ . Each iteration uses BP to compute the current model marginals.

*Remark 8.* Two implementation details worth keeping in mind are: (i) on trees, BP recovers  $\ln Z(\theta)$  (along with marginals), and (ii) some model classes with local normalization constraints admit closed-form M-step style updates instead of generic gradient steps.

### 6.3.4 Learning on loopy graphs: approximate inference in the inner loop

When the factor graph has cycles, BP is no longer guaranteed to be exact or even to converge. Nevertheless the same gradient identity holds. Typical approximations include:

1. **Loopy BP gradients:** run loopy BP to get approximate factor beliefs  $\mu_a(x_{\partial a})$ , and plug them into  $\mathbb{E}_{p_X^{\theta}}[T_a] \approx \sum_{x_{\partial a}} \mu_a(x_{\partial a}) T_a(x_{\partial a})$ .
2. **Sampling-based estimates:** use MCMC (e.g. Gibbs sampling) to approximate  $\mathbb{E}_{p_X^{\theta}}[T_a]$  by Monte Carlo.
3. **Pseudo-likelihood:** replace the likelihood by a product of local conditional likelihoods to avoid  $Z(\theta)$ .
4. **Contrastive divergence / persistent chains.** Approximate the model expectation by running a small number of MCMC steps from data. This is historically popular for energy-based models.

In all cases, the message is the same: *learning is easy once inference is easy.*

## 6.4 Missing data / latent variables: EM with BP

We now move to the setting where some variables are unobserved. Let the vector  $X = (Y, H)$  where  $Y$  is a vector of observed variables and  $H$  is a vector of hidden (latent) variables. Each sample provides only  $y^{(\ell)}$ . We wish to maximize the observed-data log-likelihood

$$\mathcal{L}_{\text{obs}}(\theta) := \sum_{\ell=1}^N \ln p_Y^\theta(y^{(\ell)}), \quad p_Y^\theta(y) = \sum_h p_X^\theta(y, h).$$

The difficulty is the log of the sum over hidden states.

### 6.4.1 EM lower bound and the Q-function

EM is an alternating maximization procedure. For each data point  $y^{(\ell)}$ , introduce an auxiliary distribution  $q^{(\ell)}(h)$  over hidden states. Using Jensen's inequality, we find that

$$\ln \sum_h p_X^\theta(y^{(\ell)}, h) = \ln \sum_h q^{(\ell)}(h) \frac{p_X^\theta(y^{(\ell)}, h)}{q^{(\ell)}(h)} \geq \sum_h q^{(\ell)}(h) \ln \frac{p_X^\theta(y^{(\ell)}, h)}{q^{(\ell)}(h)},$$

with equality if and only if  $\frac{p_X^\theta(y^{(\ell)}, h)}{q^{(\ell)}(h)}$  is constant for all  $h$  where  $q^{(\ell)}(h) > 0$ , which is equivalent to

$$q^{(\ell)}(h) = p_{H|Y}^\theta(h | y^{(\ell)}).$$

Summing over  $\ell$ , we find that

$$\begin{aligned} \mathcal{L}_{\text{obs}}(\theta) &\geq \sum_{\ell=1}^N \sum_h q^{(\ell)}(h) \ln p_X^\theta(y^{(\ell)}, h) - \sum_{\ell=1}^N \sum_h q^{(\ell)}(h) \ln q^{(\ell)}(h) \\ &= \sum_{\ell=1}^N \mathbb{E}_{q^{(\ell)}} [\ln p_X^\theta(Y = y^{(\ell)}, H)] + \sum_{\ell=1}^N H(q^{(\ell)}) \\ &=: \mathcal{F}(\theta, \{q^{(\ell)}\}), \end{aligned}$$

where  $H(q) = -\sum_h q(h) \ln q(h)$  and  $\mathcal{F}$  is the *EM functional*.

*Remark 9* (EM algorithm). We can increase the likelihood by alternating:

- E-step: choose  $q^{(\ell)}$  to maximize  $\mathcal{F}$  for fixed  $\theta$ ;
- M-step: choose  $\theta$  to maximize  $\mathcal{F}$  for fixed  $\{q^{(\ell)}\}$ .

Each step is guaranteed to maintain or increase the observed-data log-likelihood.

**E-step: posterior inference with evidence.** In a factor graph, conditioning on  $Y = y^{(\ell)}$  is implemented by clamping the observed variable nodes (or equivalently adding unary “evidence factors” that zero out inconsistent states). For each  $\ell$ , we can rewrite the bound term as

$$\sum_h q^{(\ell)}(h) \ln \frac{p_X^\theta(y^{(\ell)}, h)}{q^{(\ell)}(h)} = \ln p_Y^\theta(y^{(\ell)}) - D(q^{(\ell)} \parallel p_{H|Y}^\theta(\cdot | y^{(\ell)})).$$

Therefore,

$$\mathcal{F}(\theta, \{q^{(\ell)}\}) = \mathcal{L}_{\text{obs}}(\theta) - \sum_{\ell=1}^N D(q^{(\ell)} \parallel p_{H|Y}^\theta(\cdot | y^{(\ell)})),$$

so for fixed  $\theta$ , the maximizing choice for  $q^{(\ell)}$  is the posterior:

$$q^{(\ell),\text{opt}}(h) = p_{H|Y}^\theta(h | y^{(\ell)}).$$

For factor graphs, we can use BP to compute the posterior of the clamped distribution

$$p_{X_{\partial a}|Y}^\theta(x_{\partial a} | y^{(\ell)}).$$

**M-step: maximize expected complete-data log-likelihood.** Using the exponential-family energy form, we get

$$\ln p_X^\theta(y^{(\ell)}, h) = \sum_{a \in F} \theta_a^\top T_a((y^{(\ell)}, h)_{\partial a}) - \ln Z(\theta).$$

So the M-step objective becomes

$$\mathcal{F}(\theta, \{q^{(\ell)}\}) = \sum_{\ell=1}^N \sum_h q^{(\ell)}(h) \left( \sum_{a \in F} \theta_a^\top T_a((y^{(\ell)}, h)_{\partial a}) - \ln Z(\theta) \right) + \sum_{\ell=1}^N H(q^{(\ell)}).$$

Thus, compared to the fully observed case, the empirical features are replaced by *expected* features under the posterior and we have

$$\frac{1}{N} \mathcal{F}(\theta, \{q^{(\ell)}\}) = \sum_{a \in F} \theta_a^\top \hat{T}_{a,\text{soft}} - \ln Z(\theta) + \frac{1}{N} \sum_{\ell=1}^N H(q^{(\ell)}),$$

where the *soft* (posterior-averaged) statistics are given by

$$\hat{T}_{a,\text{soft}} := \frac{1}{N} \sum_{\ell=1}^N \sum_{x_{\partial a}} p_{X_{\partial a}|Y}^\theta(x_{\partial a} | y^{(\ell)}) T_a(x_{\partial a}).$$

Therefore the M-step has the same structure as the fully observed case, but with empirical statistics replaced by soft statistics. Since the entropy term does not depend on  $\theta$ , differentiating gives

$$\nabla_{\theta_a} \left( \frac{1}{N} \mathcal{F}(\theta, \{q^{(\ell)}\}) \right) = \hat{T}_{a,\text{soft}} - \mathbb{E}_{p_X^\theta}[T_a(X_{\partial a})].$$

This now equals the “(soft) data moments minus model moments.”

**Two inference problems.** Notice what EM requires:

- In the E-step: inference in the *clamped* model  $p_{X|Y}^{\theta^{\text{old}}}(\cdot | y^{(\ell)})$  to compute soft statistics.
- In the M-step (for undirected models): inference in the *unclamped* model  $p_X^\theta$  to compute  $\mathbb{E}_{p_X^\theta}[T_a]$  (unless you adopt an M-step approximation or a locally normalized parameterization).

On trees both are exact via BP; on loopy graphs both are typically approximate (loopy BP, sampling, etc.).

### 6.4.2 EM gradient form and “soft” moment matching

Define the concatenated sufficient statistic

$$T(x) := (T_a(x_{\partial a}))_{a \in F},$$

and the corresponding *soft empirical sufficient-statistic average*

$$\hat{T}_{\text{soft}} := \frac{1}{N} \sum_{\ell=1}^N \mathbb{E}_{q^{(\ell)}}[T(Y = y^{(\ell)}, H)].$$

Then

$$\nabla_{\theta} \left( \frac{1}{N} \mathcal{F}(\theta, \{q^{(\ell)}\}) \right) = \hat{T}_{\text{soft}} - \mathbb{E}_{p_X^{\theta}}[T(X)].$$

So it is the same “data minus model” structure as before, except that “data” is replaced by posterior expectations from the E-step.

**Where BP appears.** In many discrete models,  $\hat{T}_{\text{soft}}$  can be computed from local marginals  $p_{X_{\partial a}|Y}^{\theta^{\text{old}}}(x_{\partial a} | y^{(\ell)})$ , which are exactly what BP estimates (or computes exactly on trees).

### 6.4.3 EM + BP: algorithm template

When the factor graph is a tree, an EM iteration can be written as:

1. **E-step (BP with evidence).** For each  $\ell = 1, \dots, N$ : clamp  $Y = y^{(\ell)}$  and run BP to obtain exact marginals  $p_{X_{\partial a}|Y}^{\theta}(x_{\partial a} | y^{(\ell)})$  for each factor  $a$ . Accumulate soft sufficient statistics  $\hat{T}_{a,\text{soft}}$ .
2. **M-step (optimize  $Q$ ).** In the standard EM algorithm, the M-step solves for a  $\theta$  with zero derivative in order to find a maximum over  $\theta$  for the current  $q$ . Instead, modern approaches often update  $\theta$  by (e.g.) gradient ascent:

$$\theta_a \leftarrow \theta_a + \eta \left( \hat{T}_{a,\text{soft}} - \mathbb{E}_{p_X^{\theta}}[T_a(X_{\partial a})] \right),$$

where  $\mathbb{E}_{p_X^{\theta}}[T_a]$  is computed by a BP run on the unclamped tree at the current  $\theta$ . (Depending on the model class, some M-steps admit closed forms; for general undirected exponential families, iterative optimization is typical.)

**Rule of thumb: “compute soft counts then normalize”** In many familiar latent-variable models (mixtures, HMMs, etc.), the M-step reduces to:

*estimate posterior (soft) counts and use them like hard counts to estimate fully observed MLE.*

On chains (HMMs), BP is forward–backward; on trees, BP is sum-product; on loopy graphs, EM typically becomes “approximate EM” because the E-step posteriors are approximate.

## 6.5 Example: EM for a discrete mixture model

Consider a two-variable Bayesian network with a latent class variable  $H \in [K]$  and an observed variable  $Y \in \mathcal{Y}$ . The model is

$$p_{Y,H}^\theta(y, h) = p_H^\theta(h) p_{Y|H}^\theta(y | h),$$

where the parameters satisfy

$$p_H^\theta(h) \geq 0, \quad \sum_{h=1}^K p_H^\theta(h) = 1, \quad p_{Y|H}^\theta(y | h) \geq 0, \quad \sum_{y \in \mathcal{Y}} p_{Y|H}^\theta(y | h) = 1 \quad \forall h.$$

Given observed data  $\mathcal{D} = \{y^{(1)}, \dots, y^{(N)}\}$  (classes unobserved), the observed-data log-likelihood is

$$\mathcal{L}_{\text{obs}}(\theta) = \sum_{\ell=1}^N \ln p_Y^\theta(y^{(\ell)}) = \sum_{\ell=1}^N \ln \left( \sum_{h=1}^K p_H^\theta(h) p_{Y|H}^\theta(y^{(\ell)} | h) \right).$$

**E-step (latent-state posteriors).** For current parameters  $\theta$ ,  $\mathcal{F}$  is maximized by computing the posterior distribution over  $H$  for each data point:

$$q^{(\ell)}(h) := p_{H|Y}^\theta(h | y^{(\ell)}) = \frac{p_H^\theta(h) p_{Y|H}^\theta(y^{(\ell)} | h)}{\sum_{h'=1}^K p_H^\theta(h') p_{Y|H}^\theta(y^{(\ell)} | h')}.$$

**M-step (maximize expected complete log-likelihood).** For fixed  $q^{(\ell)}$ , the  $\theta$ -dependent part of  $\mathcal{F}$  is

$$\sum_{\ell=1}^N \sum_{h=1}^K q^{(\ell)}(h) \left( \ln p_H^\theta(h) + \ln p_{Y|H}^\theta(y^{(\ell)} | h) \right),$$

subject to normalization constraints on  $p_H^\theta$  and each conditional table  $p_{Y|H}^\theta(\cdot | h)$ . Define the soft averages

$$\widehat{p}_H^{\text{soft}}(h) := \frac{1}{N} \sum_{\ell=1}^N q^{(\ell)}(h), \quad \widehat{p}_{Y|H}^{\text{soft}}(y | h) := \frac{\sum_{\ell=1}^N q^{(\ell)}(h) \mathbb{I}\{y^{(\ell)} = y\}}{\sum_{\ell=1}^N q^{(\ell)}(h)}.$$

Then the M-step objective can be rewritten as

$$\begin{aligned} & \sum_{\ell=1}^N \sum_{h=1}^K q^{(\ell)}(h) \left( \ln p_H^\theta(h) + \ln p_{Y|H}^\theta(y^{(\ell)} | h) \right) \\ &= -N D\left(\widehat{p}_H^{\text{soft}} \parallel p_H^\theta\right) - \sum_{h=1}^K N_h^{\text{soft}} D\left(\widehat{p}_{Y|H}^{\text{soft}}(\cdot | h) \parallel p_{Y|H}^\theta(\cdot | h)\right) + \text{const}, \end{aligned}$$

where  $N_h^{\text{soft}} := \sum_{\ell=1}^N q^{(\ell)}(h)$  and the constant does not depend on  $\theta$ . Hence, optimization is equivalent to matching the soft empirical marginals/conditionals (i.e., soft sufficient-statistic moment matching). As a sanity check, one can verify that Lagrangian optimization give the same closed-form updates.

### EM algorithm (mixture model).

1. Initialize  $p_H^\theta$  and  $p_{Y|H}^\theta$  to have full support.
2. E-step: compute  $q^{(\ell)}(h)$  for all  $\ell, h$ .
3. M-step: update  $p_H^\theta(h)$  and  $p_{Y|H}^\theta(y|h)$  using the formulas above.
4. Repeat E/M steps until the increase in  $\mathcal{L}_{\text{obs}}(\theta)$  is below a tolerance.

This is exactly the recipe above: E-step computes posterior expectations (soft counts), and M-step normalizes those soft counts to update local CPT parameters.

## 6.6 Example: EM on a hidden Markov model (HMM)

Although HMMs are usually introduced as directed models, they have a clean factor graph view and provide an excellent showcase for BP/EM. Let  $S = (S_1, \dots, S_T) \in \mathcal{S}^T$  be hidden states and  $(Y_1, \dots, Y_T) \in \mathcal{Y}^T$  be observed symbols. The joint distribution factors as

$$p_{Y,S}^\theta(y, s) = p_{S_1}^\theta(s_1) \prod_{t=2}^T p_{S_t|S_{t-1}}^\theta(s_t | s_{t-1}) \prod_{t=1}^T p_{Y_t|S_t}^\theta(y_t | s_t).$$

This is a chain factor graph. BP on a chain is exact and known as the forward-backward algorithm.

**E-step.** Compute posteriors:

$$\gamma_t(i) := p_{S_t|Y}^{\theta^{\text{old}}}(i|y), \quad \xi_t(i, j) := p_{S_{t-1}, S_t|Y}^{\theta^{\text{old}}}(i, j|y).$$

These are obtained from BP messages.

**M-step.** Update transition probabilities (via  $\theta$ ) by normalized expected counts:

$$p_{S_t|S_{t-1}}^\theta(j|i) = \frac{\sum_{t=2}^T \xi_t(i, j)}{\sum_{t=2}^T \sum_{j'} \xi_t(i, j')}.$$

Update emission probabilities (via  $\theta$ ) similarly:

$$p_{Y_t|S_t}^\theta(y|i) = \frac{\sum_{t=1}^T \gamma_t(i) \mathbb{I}\{y_t = y\}}{\sum_{t=1}^T \gamma_t(i)}.$$

This is a concrete template: compute local marginals by BP, then normalize. If the HMM is known to be time-invariant, then the E-step counts can be pooled across time steps to get better estimates.

## 6.7 Tree factor graphs revisited: learning by counting

The discussion above emphasized that, for general factor graphs, MLE requires repeated inference because of the global partition function  $Z(\theta)$ . On trees, there are many sets of parameters that give the same observation probabilities. However, there are important subsets where a tree-structured factor graph acts just like a Bayesian network: *when the factors are chosen to be locally normalized conditional tables.*

**Rooting a tree factor graph and reading conditionals.** Let  $G = (V, F, E)$  be a factor graph whose underlying bipartite graph is a tree. Choose a root variable node  $r \in V$ . Direct every edge away from  $r$ , so that each factor node  $a \in F$  has a unique parent variable  $p(a) \in \partial a$  (the neighbor of  $a$  on the path to the root), and the remaining neighbors  $\text{ch}(a) := \partial a \setminus \{p(a)\}$  are the children variables of  $a$ . Now restrict the model class by requiring each factor to be *locally normalized* as a conditional:

$$\sum_{x_{\text{ch}(a)} \in \mathcal{X}_{\text{ch}(a)}} f_a(x_{p(a)}, x_{\text{ch}(a)}) = 1 \quad \text{for every fixed } x_{p(a)} \in \mathcal{X}_{p(a)}. \quad (6.2)$$

Under (6.2), we may interpret

$$f_a(x_{p(a)}, x_{\text{ch}(a)}) = p_\theta(x_{\text{ch}(a)} \mid x_{p(a)}),$$

i.e., factor  $a$  is a conditional probability table that generates its child variables given its parent variable. If we also specify a root marginal  $p_\theta(x_r)$ , then the full joint distribution is locally normalized and can be written in a directed, BN-like form:

$$p_\theta(x) = p_\theta(x_r) \prod_{a \in F} p_\theta(x_{\text{ch}(a)} \mid x_{p(a)}). \quad (6.3)$$

In particular, the global partition function is identically  $Z(\theta) = 1$  for this parameterization, because the model is constructed as a product of normalized conditionals.

**MLE becomes counting + normalization.** For fully observed i.i.d. data  $\mathcal{D} = \{x^{(1)}, \dots, x^{(N)}\}$ , define for each factor  $a$  the parent/children configuration counts

$$N_a(x_{p(a)}, x_{\text{ch}(a)}) := \sum_{\ell=1}^N \mathbb{I}\left\{x_{p(a)}^{(\ell)} = x_{p(a)}, x_{\text{ch}(a)}^{(\ell)} = x_{\text{ch}(a)}\right\}.$$

Then the maximum likelihood estimate for the conditional table at  $a$  is exactly the BN rule:

$$\hat{p}^{\text{ML}}(x_{\text{ch}(a)} \mid x_{p(a)}) = \frac{N_a(x_{p(a)}, x_{\text{ch}(a)})}{\sum_{x'_{\text{ch}(a)} \in \mathcal{X}_{\text{ch}(a)}} N_a(x_{p(a)}, x'_{\text{ch}(a)})} \quad (\text{whenever the denominator is nonzero}). \quad (6.4)$$

Similarly, the root marginal is estimated by its empirical frequency  $\hat{p}^{\text{ML}}(x_r) = \hat{p}_{X_r}^{\mathcal{D}}(x_r)$ . Thus, on trees, if we *choose* a locally normalized parameterization, learning can be done without any inference: one simply computes local histograms and normalizes.

*Remark 10* (Tree factor graphs vs. tree Bayesian networks). Equation (6.3) shows that a tree factor graph with locally normalized factors can be viewed as a Bayesian network whose “conditional probability tables” live on factor nodes rather than variable nodes. This is not a contradiction with the earlier message “learning is outer-loop optimization while inference is inner-loop”. That approach is required if the factor graph has cycles. For trees, we can simplify things by restricting the parameterization to arbitrary positive tables  $f_a$  satisfying (6.2). Then, the likelihood involves  $\ln Z(\theta)$  and the MLE conditions couple factors through model expectations.

*Remark 11* (Why this breaks on graphs with cycles). The counting rule (6.4) relies on the existence of a globally consistent directed factorization with purely local normalization constraints. When the factor graph contains a cycle, there is generally no way to assign each factor a unique parent variable so that (i) every factor is normalized as a conditional given its parent and (ii) the product of these conditionals defines the desired undirected model without additional global correction. Intuitively,

a cycle creates “feedback”: changing one factor changes the global normalizer, and hence changes the distribution everywhere. Formally, for undirected loopy models the MLE gradient has the form

$$\nabla_{\theta_a} \ell(\theta) = \widehat{\mathbb{E}}_{\mathcal{D}}[T_a(X_{\partial a})] - \mathbb{E}_{p^\theta}[T_a(X_{\partial a})],$$

so the stationarity conditions couple all parameters through the model expectations. Therefore, even with fully observed data, learning on cycles requires inference (exact or approximate) to estimate  $\mathbb{E}_{p^\theta}[T_a]$ ; “count and normalize” is not valid in general.

### 6.7.1 Example: learning on a single 3-cycle using (loopy) BP

We now illustrate the standard “outer loop learning / inner loop inference” template on the smallest loopy factor graph. Let  $X_1, X_2, X_3 \in \{0, 1\}$  be binary variables connected in a single cycle with three pairwise factors:

$$p^\theta(x_1, x_2, x_3) = \frac{1}{Z(\theta)} \psi_{12}(x_1, x_2) \psi_{23}(x_2, x_3) \psi_{31}(x_3, x_1), \quad Z(\theta) = \sum_{x_1, x_2, x_3} \prod_{(i,j)} \psi_{ij}(x_i, x_j). \quad (6.5)$$

Use an Ising-style log-linear parameterization

$$\psi_{ij}(x_i, x_j) = \exp\left(J_{ij} s(x_i) s(x_j)\right), \quad s(x) := 2x - 1 \in \{\pm 1\}, \quad (6.6)$$

with parameters  $\theta = (J_{12}, J_{23}, J_{31})$ .

**Log-likelihood and gradients.** Given fully observed i.i.d. samples  $\mathcal{D} = \{x^{(1)}, \dots, x^{(N)}\}$ , the normalized log-likelihood is

$$\ell(\theta) = \frac{1}{N} \mathcal{L}(\theta) = \sum_{(i,j) \in \{(1,2), (2,3), (3,1)\}} J_{ij} \widehat{\mathbb{E}}_{\mathcal{D}}[s(X_i) s(X_j)] - \ln Z(\theta). \quad (6.7)$$

Differentiating gives the familiar “data moment minus model moment” form:

$$\frac{\partial \ell}{\partial J_{ij}} = \widehat{\mathbb{E}}_{\mathcal{D}}[s(X_i) s(X_j)] - \mathbb{E}_{p^\theta}[s(X_i) s(X_j)]. \quad (6.8)$$

Thus, MLE aims to match the three empirical pairwise correlations by adjusting  $(J_{12}, J_{23}, J_{31})$ . The difficulty is computing  $\mathbb{E}_{p^\theta}[s(X_i) s(X_j)]$ , which is an inference problem on the loopy graph.

**Approximating model moments by loopy BP beliefs.** Run loopy sum-product BP on the 3-cycle to obtain approximate pairwise beliefs  $b_{ij}(x_i, x_j)$ . Then approximate

$$\mathbb{E}_{p^\theta}[s(X_i) s(X_j)] \approx \sum_{x_i, x_j \in \{0,1\}} b_{ij}(x_i, x_j) s(x_i) s(x_j). \quad (6.9)$$

Plugging (6.9) into (6.8) yields an approximate gradient ascent update:

$$J_{ij} \leftarrow J_{ij} + \eta \left( \widehat{\mathbb{E}}_{\mathcal{D}}[s(X_i) s(X_j)] - \sum_{x_i, x_j} b_{ij}(x_i, x_j) s(x_i) s(x_j) \right), \quad (6.10)$$

where  $\eta > 0$  is a step size (often with damping / line search in practice).

**Loopy BP updates specialized to the 3-cycle.** Because every factor is pairwise and every variable has degree 2, the BP messages have a particularly simple form. Let  $m_{i \rightarrow (ij)}(x_i)$  be the variable-to-factor message and  $m_{(ij) \rightarrow j}(x_j)$  the factor-to-variable message. If  $k$  denotes the other neighbor of  $i$  on the cycle (so  $\{(ij), (ik)\}$  are the two adjacent factors), then

$$m_{i \rightarrow (ij)}(x_i) \propto m_{(ik) \rightarrow i}(x_i), \quad (6.11)$$

$$m_{(ij) \rightarrow j}(x_j) \propto \sum_{x_i \in \{0,1\}} \psi_{ij}(x_i, x_j) m_{i \rightarrow (ij)}(x_i). \quad (6.12)$$

After iterating (6.11)–(6.12) (normalizing messages at each step), form approximate singleton and pairwise beliefs:

$$b_i(x_i) \propto m_{(ij) \rightarrow i}(x_i) m_{(ik) \rightarrow i}(x_i), \quad (6.13)$$

$$b_{ij}(x_i, x_j) \propto \psi_{ij}(x_i, x_j) m_{i \rightarrow (ij)}(x_i) m_{j \rightarrow (ij)}(x_j). \quad (6.14)$$

Then compute the approximate correlations via (6.9) and update parameters via (6.10).

**What one learning iteration looks like (sanity check).** Suppose the empirical correlations (in  $\{\pm 1\}$  coding) are

$$\widehat{\mathbb{E}}_{\mathcal{D}}[s(X_1)s(X_2)] = 0.60, \quad \widehat{\mathbb{E}}_{\mathcal{D}}[s(X_2)s(X_3)] = 0.20, \quad \widehat{\mathbb{E}}_{\mathcal{D}}[s(X_3)s(X_1)] = 0.40.$$

Initialize  $J_{12} = J_{23} = J_{31} = 0$ , so  $\psi_{ij} \equiv 1$  and the BP beliefs are uniform:  $b_{ij}(x_i, x_j) = 1/4$ , hence the model correlations are initially 0. Then the first gradient step is

$$J_{12} \leftarrow \eta(0.60), \quad J_{23} \leftarrow \eta(0.20), \quad J_{31} \leftarrow \eta(0.40).$$

## 6.8 Summary: what to remember

1. **Bayesian networks (known DAG, fully observed):** MLE is counting + normalization (closed form).
2. **Undirected/factor graph models (fully observed):**

$$\nabla_{\theta} \left( \frac{1}{N} \ln p_X^{\theta}(\mathcal{D}) \right) = \widehat{\mathbb{E}}_{\mathcal{D}}[T(X)] - \mathbb{E}_{p_X^{\theta}}[T(X)],$$

where  $T(x)$  is a concatenation of  $T_a(x_{\partial a})$  over all factors  $a$ . Learning is outer-loop optimization while inference (and computing marginals) is inner-loop computation.

3. **Missing data / latent variables: EM.** E-step runs inference to compute posterior expectations of sufficient statistics. M-step updates parameters using these “soft” statistics (sometimes closed form, sometimes requiring optimization). BP is the natural inference engine in factor graphs (exact on trees, approximate on loopy graphs).

## 6.9 Exercises

**Exercise 1** (EM soft statistics). *Derive the expression for  $\widehat{T}_{a, \text{soft}}$  in the E-step and explain why it can be computed from the factor marginals  $p_{X_{\partial a}|Y}^{\theta^{\text{old}}}(x_{\partial a} | y^{(\ell)})$  returned by BP on a tree.*

**Exercise 2** (Gauge freedom for table factors). Let  $\mathcal{X}_{\partial a}$  be finite and define  $f_a(u) = \exp(\theta_{a,u})$ . Show that adding a constant  $c$  to all  $\theta_{a,u}$  multiplies  $f_a$  by  $e^c$  and does not change  $p_X^\theta$  after renormalization. Give one gauge constraint that removes this redundancy.

**Exercise 3** (MLE for a single CPT via direct optimization). Let  $X \in \{1, \dots, r\}$  and  $U \in \{1, \dots, s\}$ , and suppose the conditional distribution is a CPT  $p_\theta(x | u) = \theta_{x|u}$  with constraints  $\sum_{x=1}^r \theta_{x|u} = 1$  for each  $u$ . Given an i.i.d. dataset  $\mathcal{D} = \{(x^{(\ell)}, u^{(\ell)})\}_{\ell=1}^N$ , define counts  $N(x, u) = \sum_{\ell=1}^N \mathbb{I}\{x^{(\ell)} = x, u^{(\ell)} = u\}$  and  $N(u) = \sum_x N(x, u)$ .

1. Write the (normalized) log-likelihood  $\frac{1}{N}\mathcal{L}(\theta)$  in terms of  $N(x, u)$  and  $\theta_{x|u}$ .
2. Using Lagrange multipliers (one constraint per  $u$ ), derive the MLE for  $\theta_{x|u}$ .
3. Explain what happens if  $N(u) = 0$  and give a simple practical fix.

**Exercise 4** (Convexity/concavity of the undirected log-likelihood). Consider the exponential-family factor graph model

$$p^\theta(x) = \frac{1}{Z(\theta)} \exp\left(\sum_{a \in F} \theta_a^\top T_a(x_{\partial a})\right), \quad \theta = (\theta_a)_{a \in F}.$$

Let  $\frac{1}{N}\mathcal{L}(\theta)$  be the normalized log-likelihood for fully observed data.

1. Show that  $\nabla_\theta \ln Z(\theta) = \mathbb{E}_{p^\theta}[T(X)]$ , where  $T(X)$  is the concatenation of all  $T_a(X_{\partial a})$ .
2. Show that the Hessian satisfies  $\nabla_\theta^2 \ln Z(\theta) = \text{Cov}_{p^\theta}(T(X))$ .
3. Conclude that  $\ln Z(\theta)$  is convex and that  $\ell(\theta)$  is concave.

**Exercise 5** (One-step contrastive divergence update (CD-1) for a log-linear model). Consider a discrete energy-based model

$$p^\theta(x) = \frac{1}{Z(\theta)} \exp(\theta^\top T(x)).$$

Let  $\widehat{\mathbb{E}}_{\mathcal{D}}[T(X)]$  be the empirical feature average. In CD-1, one approximates the model expectation  $\mathbb{E}_{p^\theta}[T(X)]$  by sampling a data point  $X^0 \sim \widehat{p}_X^{\mathcal{D}}$ , performing a Gibbs step to get  $X^1$ , and using  $T(X^1)$ .

1. Write the exact gradient of the normalized log-likelihood  $\frac{1}{N}\mathcal{L}(\theta)$ .
2. Write the CD-1 approximate gradient  $\widehat{g}_{\text{CD1}}(\theta)$  in terms of  $T(X^0)$  and  $T(X^1)$ .
3. Give the resulting CD-1 update rule with step size  $\eta$ .

**Exercise 6** (EM monotonicity via a KL decomposition). Recall the EM functional

$$\mathcal{F}(\theta, \{q^{(\ell)}\}) = \sum_{\ell=1}^N \mathbb{E}_{q^{(\ell)}}[\ln p^\theta(y^{(\ell)}, H)] + \sum_{\ell=1}^N H(q^{(\ell)}).$$

1. Show that for each  $\ell$ ,

$$\ln p^\theta(y^{(\ell)}) = \mathcal{F}_\ell(\theta, q^{(\ell)}) + D(q^{(\ell)}(\cdot) \| p^\theta(H | y^{(\ell)})),$$

where  $\mathcal{F}_\ell$  is the  $\ell$ th summand of  $\mathcal{F}$ .

2. Use this identity to argue that choosing  $q^{(\ell)} = p^{\theta^{\text{old}}}(H | y^{(\ell)})$  maximizes  $\mathcal{F}$  over  $q$  for fixed  $\theta^{\text{old}}$ .

3. Conclude (informally) why the E-step followed by an exact M-step does not decrease the observed-data log-likelihood.

**Exercise 7** (HMM forward-backward messages and  $\gamma, \xi$ ). Consider an HMM with hidden states  $S_t \in \{1, \dots, K\}$  and observations  $Y_t \in \mathcal{Y}$  with parameters: initial  $\pi(i) = p(S_1 = i)$ , transition  $A_{ij} = p(S_t = j \mid S_{t-1} = i)$ , and emission  $B_j(y) = p(Y_t = y \mid S_t = j)$ . Given a single observation sequence  $y_{1:T}$ :

1. Define forward messages  $\alpha_t(j)$  and give their recursion (up to normalization).
2. Define backward messages  $\beta_t(i)$  and give their recursion (up to normalization).
3. Express  $\gamma_t(j) = p(S_t = j \mid y_{1:T})$  in terms of  $\alpha_t, \beta_t$ .
4. Express  $\xi_t(i, j) = p(S_{t-1} = i, S_t = j \mid y_{1:T})$  in terms of  $\alpha_{t-1}, A, B, \beta_t$ .

**Exercise 8** (EM M-step for an HMM with multiple sequences). Suppose we observe  $N$  independent sequences  $\{y_{1:T_\ell}^{(\ell)}\}_{\ell=1}^N$  from the same time-homogeneous HMM parameters  $(\pi, A, B)$ . Let  $\gamma_t^{(\ell)}(j)$  and  $\xi_t^{(\ell)}(i, j)$  be the posteriors computed in the E-step for sequence  $\ell$ .

1. Derive the M-step update for the initial distribution  $\pi(j)$ .
2. Derive the M-step update for the transition probabilities  $A_{ij}$ .
3. Assume emissions are categorical over  $\mathcal{Y}$  (finite alphabet). Derive the M-step update for  $B_j(y)$ .