We now consider the gradient of $\ell(\boldsymbol{\theta} : \mathcal{D})$ with respect to $\boldsymbol{\theta}$. Since the term $\boldsymbol{H}_Q(\mathcal{H})$ does not depend on $\boldsymbol{\theta}$, we get that

$$\nabla_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta} : \mathcal{D}) = \nabla_{\boldsymbol{\theta}} \boldsymbol{E}_Q[\ell(\boldsymbol{\theta} : \langle \mathcal{D}, \mathcal{H} \rangle)] + \nabla_{\boldsymbol{\theta}} \boldsymbol{D}(Q(\mathcal{H}) \| P(\mathcal{H} \mid \mathcal{D}, \boldsymbol{\theta})).$$

This observation is true for any choice of Q. Now suppose we are in an EM iteration. In this case, we set $Q = P(\mathcal{H} \mid \mathcal{D}, \boldsymbol{\theta}^t)$ and evaluate the gradient at $\boldsymbol{\theta}^t$.

A somewhat simplified proof runs as follows. Because $\boldsymbol{\theta} = \boldsymbol{\theta}^t$ is a minimum of the KL-divergence term, we know that $\nabla_{\boldsymbol{\theta}} \boldsymbol{D}(Q(\mathcal{H}) \| P(\mathcal{H} \mid \mathcal{D}, \boldsymbol{\theta}^t))$ is 0. This implies that

$$\nabla_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}^t : \mathcal{D}) = \nabla_{\boldsymbol{\theta}} \boldsymbol{E}_Q \left[\ell(\boldsymbol{\theta}^t : \langle \mathcal{D}, \mathcal{H} \rangle) \right].$$

Or, in other words, $\nabla_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}^t : \mathcal{D}) = 0$ if and only if $\nabla_{\boldsymbol{\theta}} \boldsymbol{E}_Q \left[\ell(\boldsymbol{\theta}^t : \langle \mathcal{D}, \mathcal{H} \rangle) \right] = 0$.

Recall that $\boldsymbol{\theta}^{t+1} = \arg\max_{\boldsymbol{\theta}} \boldsymbol{E}_Q \left[\ell(\boldsymbol{\theta}^t : \langle \mathcal{D}, \mathcal{H} \rangle) \right]$. Hence the gradient of the expected likelihood at $\boldsymbol{\theta}^{t+1}$ is 0. Thus, we conclude that $\boldsymbol{\theta}^{t+1} = \boldsymbol{\theta}^t$ only if $\nabla_{\boldsymbol{\theta}} \boldsymbol{E}_Q \left[\ell(\boldsymbol{\theta}^t : \langle \mathcal{D}, \mathcal{H} \rangle) \right] = 0$. And so, at this point, $\nabla_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}^t : \mathcal{D}) = 0$. This implies that this set of parameters is a stationary point of the log-likelihood function.

The actual argument has to be somewhat more careful. Recall that the parameters must lie within some allowable set. For example, the parameters of a discrete random variable must sum up to one. Thus, we are searching within a constrained space of parameters. When we have constraints, we often do not have zero gradient. Instead, we get to a stationary point when the gradient is orthogonal to the constraints (that is, local changes within the allowed space do not improve the likelihood). The arguments we have stated apply equally well when we replace statements about equality to 0 with orthogonality to the constraints on the parameter space. ■

19.2.2.6 Hard-Assignment EM

In section 19.2.2.4, we briefly mentioned the idea of using a hard assignment to the hidden variables, in the context of applying EM to Bayesian clustering. We now generalize this simple idea to the case of arbitrary Bayesian networks.

hard-assignment EM

This algorithm, called *hard-assignment EM*, also iterates over two steps: one in which it completes the data given the current parameters $\boldsymbol{\theta}^t$, and the other in which it uses the completion to estimate new parameters $\boldsymbol{\theta}^{t+1}$. However, rather than using a soft completion of the data, as in standard EM, it selects for each data instance $\boldsymbol{o}[m]$ the single assignment $\boldsymbol{h}[m]$ that maximizes $P(\boldsymbol{h} \mid \boldsymbol{o}[m], \boldsymbol{\theta}^t)$.

Although hard-assignment EM is similar in outline to EM, there are important differences. In fact, hard-assignment EM can be described as optimizing a different objective function, one that involves both the learned parameters and the learned assignment to the hidden variables. This objective is to maximize the likelihood of the complete data $\langle \mathcal{D}, \mathcal{H} \rangle$, given the parameters:

$$\max_{\boldsymbol{\theta}, \mathcal{H}} \ell(\boldsymbol{\theta} : \mathcal{H}, \mathcal{D}).$$

See exercise 19.14. Compare this objective to the EM objective, which attempts to maximize $\ell(\boldsymbol{\theta} : \mathcal{D})$, averaging over all possible completions of the data.

Does this observation provide us insight on these two learning procedures? The intuition is that these two objectives are similar if $P(\mathcal{H} \mid \mathcal{D}, \boldsymbol{\theta})$ assigns most of the probability mass to

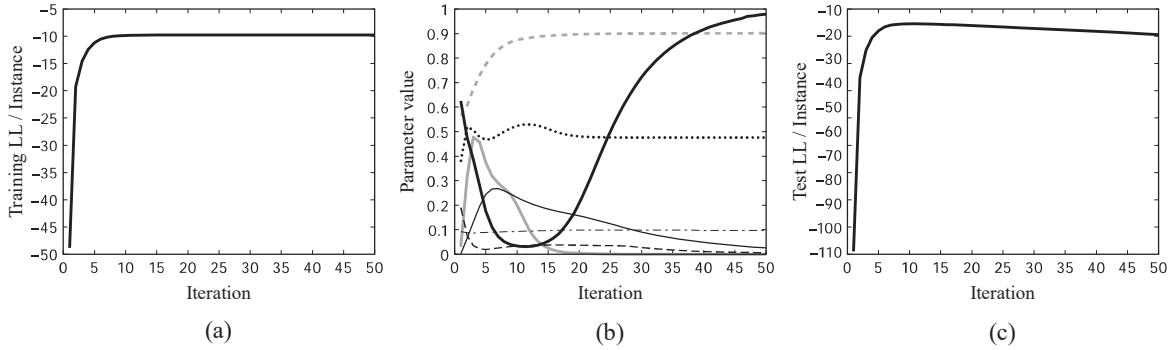

Figure 19.B.1 — Convergence of EM run on the ICU Alarm network. (a) Training log-likelihood. (b) Progress of several sample parameters. (c) Test data log-likelihood.

one completion of the data. In such a case, EM will effectively perform hard assignment during the E-step. However, if $P(\mathcal{H} \mid \mathcal{D}, \boldsymbol{\theta})$ is diffuse, the two algorithms will lead to very different solutions. In clustering, the hard-assignment version tends to increase the contrast between different classes, since assignments have to choose between them. In contrast, EM can learn classes that are overlapping, by having many instances contributing to two or more classes.

Another difference between the two EM variants is in the way they progress during the learning. Note that for a given data set, at the end of an iteration, the hard-assignment EM can be in one of a finite number of parameter values. Namely, there is only one parameter assignment for each possible assignment to $\mathcal{H}$. Thus, hard-assignment EM traverses a path in the combinatorial space of assignments to $\mathcal{H}$. The soft-assignment EM, on the other hand, traverses the continuous space of parameter assignments. The intuition is that hard-assignment EM converges faster, since it makes discrete steps. In contrast, soft-assignment EM can converge very slowly to a local maximum, since close to the maximum, each iteration makes only small changes to the parameters. The flip side of this argument is that soft-assignment EM can traverse paths that are infeasible to the hard-assignment EM. For example, if two clusters need to shift their means in a coordinated fashion, soft-assignment EM can progressively change their means. On the other hand, hard-assignment EM needs to make a "jump," since it cannot simultaneously reassign multiple instances and change the class means.

Box 19.B — Case Study: EM in Practice. *The EM algorithm is guaranteed to monotonically improve the training log-likelihood at each iteration. However, there are no guarantees as to the speed of convergence or the quality of the local maxima attained. To gain a better perspective of how the algorithm behaves in practice, we consider here the application of the method to the* ICU-Alarm *network discussed in earlier learning chapters.*

We start by considering the progress of the training data likelihood during the algorithm's iterations. In this example, $1,000$ *samples were generated from the* ICU-Alarm *network. For each instance, we then independently and randomly hid 50 percent of the variables. As can be seen in figure 19.B.1a, much of the improvement over the performance of the random starting point is in the*

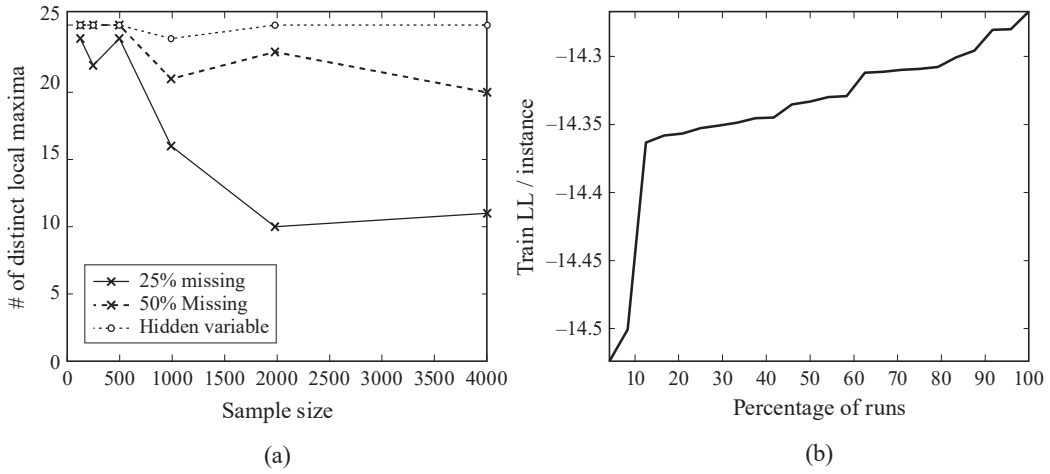

Figure 19.B.2 — Local maxima in likelihood surface. (a) Number of unique local maxima (in 25 runs) for different sample sizes and missing value configurations. (b) Distribution of training likelihood of local maxima attained for 25 random starting points with 1,000 samples and one hidden variable.

first few iterations. However, examining the convergence of different parameters in (b), we see that some parameters change significantly after the fifth iteration, even though changes to the likelihood are relatively small. In practice, any nontrivial model will display a wide range of sensitivity to the network parameters. Given more training data, the sensitivity will, typically, overall decrease. Owing to these changes in parameters, the training likelihood continues to improve after the initial iterations, but very slowly. This behavior of fast initial improvement, followed by slow convergence, is typical of EM.

We next consider the behavior of the learned model on unseen test data. As we can see in (c), early in the process, test-data improvement correlates with training-data performance. However, after the 10th iterations, training performance continues to improve, but test performance decreases. This phenomenon is an instance of overfitting *to the training data. With more data or fewer unobserved values, this phenomenon will be less pronounced. With less data or hidden variables, on the other hand, explicit techniques for coping with the problem may be needed (see box 19.C).*

overfitting

A second key issue any type of optimization of the likelihood in the case of missing data is that of local maxima. To study this phenomenon, we consider the number of local maxima for 25 random starting points under different settings. As the sample size (x-axis) grows, the number of local maxima diminishes. In addition, the number of local maxima when more values are missing (dashed line) is consistently greater than the number of local maxima in the setting where more data is available (solid line). Importantly, in the case where just a single variable is hidden, the number of local maxima is large, and remains large even when the amount of training data is quite large. To see that this is not just an artifact of possible permutations of the values of the hidden variable, and to demonstrate the importance of achieving a superior local maxima, in (b) we show the training set log-likelihood of the 25 different local maxima attained. The difference between the