An Introduction to Probabilistic Graphical Models

Michael I. Jordan
University of California, Berkeley

June 30, 2003

Chapter 20

Iterative scaling algorithms

In this chapter we turn our attention to algorithms for maximum likelihood estimation in general graphical models. We revisit the iterative proportional fitting (IPF) algorithm for estimating the clique potentials of undirected graphical models from Chapter 9, as well as the special case of decomposable models. With the junction tree algorithm in hand, we are in a position to understand these methods more deeply. Also, now that we understand that a clique potential can be viewed as a special case of collections of parameterized features, it is natural to ask if the IPF algorithm generalizes to general collections of features. We will see that the answer is yes, and will present the *generalized iterative scaling algorithm*, emphasizing the connections to IPF. Finally, we consider parameter estimation for general graphical family models with latent variables, relating the material in this chapter to the material in Chapter 11 on the EM algorithm.

We focus for the most part on discrete random variables, in order to provide a simple presentation that emphasizes the connections between the various methods that we present. In Section **??**, however, we discuss the case of general Gaussian models.

20.1 Complete observations

We begin by considering the case of completely observed data, returning to the material on decomposable models and general undirected models from Chapter 9.

Recall the necessary condition for obtaining maximum likelihood estimates—the expectations of the sufficient statistics ("features") must be equal to their empirical expectations. In the case of indicator features, this is equivalent to asking for the marginals of the model to be equal to the empirical marginals. The algorithms that we discuss in this section and in the following sections all aim to satisfy this condition.

20.1.1 Decomposable models

Decomposable graphical models can be treated either within the directed formalism or the undirected formalism; indeed, the decomposable models are precisely the intersection of directed model and undirected models. We begin by discussing the undirected case. Thus, our setting is the parameterization:

$$p(x) = \frac{1}{Z(\psi)} \prod_{C \in \mathcal{C}} \psi_C(x_C), \tag{20.1}$$

where $\mathcal{C}$ ranges over maximal cliques, and where the potential functions $\psi_C(x_C)$ are "fully parameterized"—that is, each configuration x_C is associated with an independently-varying parameter. In this setting—that of a decomposable graph, fully-parameterized potentials and maximal cliques—we are able to write down maximum likelihood estimates analytically.

Recall from our work in Chapter 17 that a graph is decomposable if and only if it is triangulated. Moreover, either of these conditions is equivalent to the existence of a junction tree on the maximal cliques of the graph. Let $\mathcal{T}$ be such a junction tree, and let $\mathcal{S}$ range over the separator sets in this junction tree. Define the following potentials for the junction tree:

$$\psi_C(x_C) = \tilde{p}(x_C) = \frac{1}{N} m(x_C) \tag{20.2}$$

$$\phi_S(x_S) = \tilde{p}(x_S) = \frac{1}{N} m(x_S). \tag{20.3}$$

That is, we let the potentials equal the empirical marginals.

With this choice of potentials, it is easy to see that the junction tree is globally consistent. In particular, consider passing a message from a clique V to a clique W via the separator S. We have:

$$\phi^*_S(x_S) = \sum_{V \setminus S} \psi_V(x_V) \tag{20.4}$$

$$= \sum_{V \setminus S} \tilde{p}(x_V) \tag{20.5}$$

$$= \tilde{p}(x_S), \tag{20.6}$$

which is the same as the initial value $\phi_S(x_S)$. Thus the update factor is one, and the message transmitted to W is vacuous. The junction tree is therefore globally consistent.

From our work on the junction tree algorithm, we know that if a junction tree is globally consistent, then the clique potentials must be proportional to the clique marginals. Thus, we have:

$$p(x_C) \propto \psi_C(x_C). \tag{20.7}$$

But we also have $\psi_C(x_C) = \tilde{p}(x_C)$, by definition, which implies that $\psi_C(x_C)$ is normalized and that the proportionality must in fact be an equality. We have:

$$p(x_C) = \tilde{p}(x_C), \tag{20.8}$$

for all C. But this is the necessary condition for the maximum likelihood estimate. Thus, it must be the case that the product:

$$\hat{p}_{ML}(x) = \frac{\prod_{C \in \mathcal{C}} \psi_C(x_C)}{\prod_{S \in \mathcal{S}} \psi_S(x_S)} = \frac{\prod_{C \in \mathcal{C}} \tilde{p}(x_C)}{\prod_{S \in \mathcal{S}} \tilde{p}(x_S)} \tag{20.9}$$

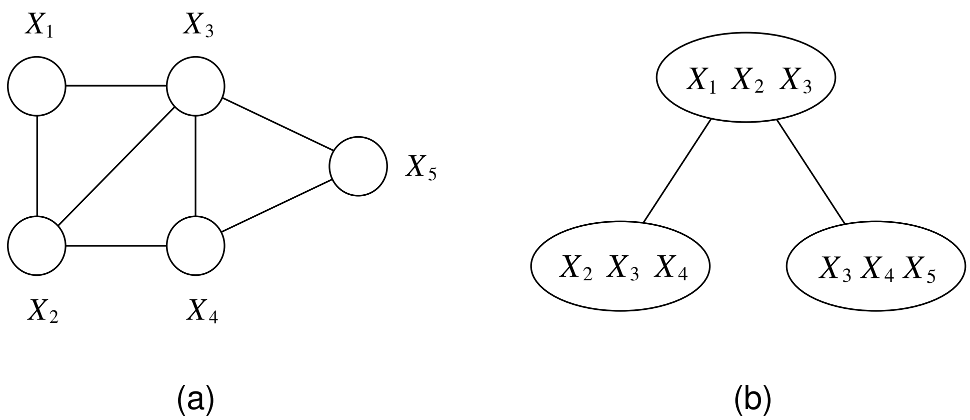

Figure 20.1: (a) A decomposable graph. (b) A junction tree for the graph in (a).

is a representation of the maximum likelihood estimate of the joint probability distribution.

We can readily turn this representation into an estimate of the parameters ψ. We simply need to divide the denominator factors in Eq. (20.9) into appropriate factors in the numerator so as to be able to write the equation as a simple product over clique potentials (i.e., rather than a ratio). For each $S \in \mathcal{S}$, there is a clique C such that $S \subseteq C$. We divide $\phi_S(x_S)$ into $\psi_C(x_C)$, redefining $\psi_C(x_C)$ and setting $\phi_S(x_S)$ equal to unity. These transformations leave Eq. (20.9) invariant, and thus we still have a representation of the maximum likelihood estimate of the joint probability distribution. Moreover, given our assumption of fully-parameterized clique potentials, dividing $\phi_S(x_S)$ into $\psi_C(x_C)$ leaves us in the same model family.

It therefore suffices to associate S with a neighboring clique C. One way to achieve this is to orient the junction tree, and associate each separator $S \in \mathcal{S}$ with the clique $C \in \mathcal{C}$ that is on the path to the root. Letting $\xi(C)$ denote the set of separators associated with clique C, we define:

$$\hat{\psi}_{C,ML}(x_C) = \frac{\tilde{p}(x_C)}{\prod_{S \in \xi(C)} \tilde{p}(x_S)} \tag{20.10}$$

as a maximum likelihood estimate of the parameters. Taking a product of such estimates over all C accounts for all of the factors in Eq. (20.9), and thus the normalization factor $Z(\hat{\psi}_{ML})$ must be one.

Consider the example shown in Figure 20.1(a). Orienting the junction tree as shown in Figure 20.1(b), we obtain:

$$\hat{\psi}_{123,ML}(x_1, x_2, x_3) = \tilde{p}(x_1, x_2, x_3) \tag{20.11}$$

$$\hat{\psi}_{234,ML}(x_2, x_3, x_4) = \frac{\tilde{p}(x_2, x_3, x_4)}{\tilde{p}(x_2, x_3)} \tag{20.12}$$

$$\hat{\psi}_{345,ML}(x_3, x_4, x_5) = \frac{\tilde{p}(x_3, x_4, x_5)}{\tilde{p}(x_3, x_4)}. \tag{20.13}$$

Note that the latter two potentials can be rewritten as conditionals under the empirical distribution.

If the original graph is a directed graph, then we can orient the junction tree so as to respect the ordering among the nodes in the original graph (see Exercise **??**). In this case, the parameter estimates are conditional probabilities, as they must be in the directed case.

20.1.2 Iterative proportional fitting

The iterative proportional fitting (IPF) algorithm is a general algorithm for finding maximum likelihood estimates in graphical models, decomposable or not (see Chapter 9). To apply IPF we still require the clique potentials to be fully parameterized, but we no longer require the parameterized cliques to be maximal cliques.

Recall from our previous discussion that IPF is a coordinate ascent algorithm. The parameters are the clique potentials, and a single "coordinate" ascent step adjusts all of the parameters in a single clique potential simultaneously. Such a step has the following form:

$$\psi_C^{(t+1)}(x_C) = \psi_C^{(t)}(x_C)\frac{\tilde{p}(x_C)}{p^{(t)}(x_C)}, \tag{20.14}$$

where the old potential is multiplied by an "update factor" consisting of the empirical marginal divided by the current model marginal. Recall that in Chapter 9 we also saw that IPF can be written in terms of the joint probability:

$$p^{(t+1)}(x_U) = p^{(t)}(x_U)\frac{\tilde{p}(x_C)}{p^{(t)}(x_C)}, \tag{20.15}$$

and this equation implies:

$$p^{(t+1)}(x_U) = p^{(t)}(x_{U\setminus C} \,|\, x_C)\tilde{p}(x_C). \tag{20.16}$$

Note that while these latter two versions of the IPF update provide insight into the nature of the algorithm, the implementation of the algorithm is based on Eq. (20.14).

We have two goals in this section. The first is to provide a geometrical interpretation of the algorithm, showing that it can be understood in terms of I-projections. The second is to delve more deeply into the implementation of the algorithm, in particular discussing the interaction between IPF and the junction tree algorithm.

IPF as a sequence of I-projections

IPF is an algorithm for finding maximum likelihood estimates. Our duality results, however, also allow us to view IPF as an algorithm for finding maximum entropy distributions. In this section we exploit this duality to uncover some of the geometry underlying the IPF algorithm.

The overall geometry is that shown earlier in Figure 19.2. IPF converges to the solution p_M which is at the intersection of the set $\mathcal{E}$ and the set $\mathcal{M}$. We can view the procedure as finding the member of the exponential family which has the correct marginals, or as finding the member of the set of distributions with the correct marginals that has maximum entropy. In the latter case, we minimize $D(p \parallel h)$ for $p \in \mathcal{M}$. That is, we find the I-projection of h on $\mathcal{M}$.

But we can also say more. In particular we can characterize each step of the IPF algorithm as an I-projection. To see this, consider the following maximum entropy problem:

$$\min \quad D(p(x) \parallel p^{(t)}(x)) \tag{20.17}$$

$$\text{subject to} \quad p(x_C) = \tilde{p}(x_C), \tag{20.18}$$

where C is a particular fixed clique. (Thus, Eq. (20.18) is a set of constraints for all configurations x_C in the clique C). From the dual point of view, we must find maximum likelihood estimates in a particular exponential family, namely one that has a single fully-parameterized clique C. Moreover, the base density (which is denoted $h(x)$ in the general case) is $p^{(t)}(x)$. We can thus write this exponential family as follows:

$$p(x) = \frac{1}{Z} p^{(t)}(x) \exp\{\phi_C(x_C)\}, \tag{20.19}$$

where $\phi_C(x_C)$ is a clique potential for the clique C. This problem is readily solved analytically. Summing both sides of Eq. (20.19) over $U \backslash C$, and using $p(x_C) = \tilde{p}(x_C)$, we obtain:

$$\exp\{\phi(x_C)\} = \frac{\tilde{p}(x_C)}{p^{(t)}(x_C)} \tag{20.20}$$

and $Z = 1$. Substituting these results back into Eq. (20.19), we obtain the IPF update.

Thus, the distribution $p^{(t+1)}(x)$ in Eq. (20.15) solves our maximum likelihood problem and therefore must also be the solution to the maximum entropy problem in Eq. (20.18). An IPF step is an I-projection of $p^{(t)}(x)$ on the set defined by a single marginal constraint.

Let $\mathcal{M}_\mathcal{C}$ denote the set corresponding to the marginal constraint $p(x_C) = \tilde{p}(x_C)$. The overall solution p_M is a distribution in the set of distributions which satisfy all of these marginal constraints. We can write this set as an intersection:

$$\mathcal{M} = \cap_{C \in \mathcal{C}} \mathcal{M}_C. \tag{20.21}$$

The resulting geometry is shown in Figure 20.2. The basic setting is the exponential family $\mathcal{E}$; all IPF iterates lie in this set. Each of the sets $\mathcal{M}_\mathcal{C}$ has an intersection with this family—these are the lines shown in the figure (the sets $\mathcal{M}_\mathcal{C}$ should be envisaged as planes extending out of the paper in a third dimension). The intersection of the planes $\mathcal{M}_\mathcal{C}$ is a line $\mathcal{M}$, again extending out of the paper, and the intersection of $\mathcal{E}$ with this line is the point p_M.

As shown in the figure, at each step IPF projects onto one of the subsets $\mathcal{M}_\mathcal{C}$. Each of these projections is an I-projection. This sequence of projections converges to p_M in the limit. Moreover, each step of the algorithm draws nearer to the empirical distribution $\tilde{p}$, a point which lies in the set $\mathcal{M}$ but not (in general) in the set $\mathcal{E}$.

Thus we obtain a geometric picture of IPF that is very reminiscent of the picture of the LMS algorithm as a sequence of projections in Chapter 6—recall Figure 6.2. The projections in the case of LMS were Euclidean projections, and the orthogonality that served as the basis of much of our analysis was Euclidean orthogonality. Here the projections are I-projections, and the projections are "orthogonal" in the sense of Eq. (19.32). But the overall picture and much of the analysis is the same.

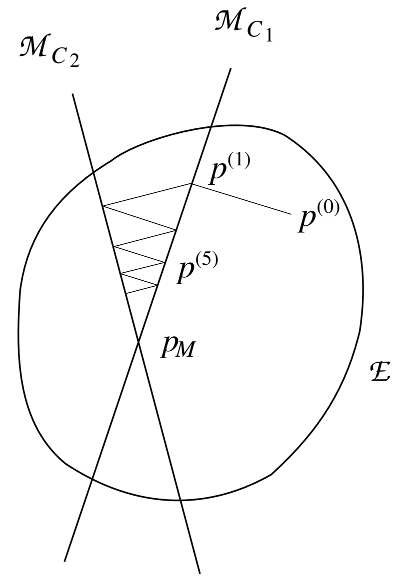

Figure 20.2: The geometry underlying the IPF algorithm. Each step of IPF is an I-projection onto a manifold which has the correct marginals for one of the cliques. All iterates lie in the exponential family $\mathcal{E}$, and the algorithm converges to the single point that is at the intersection of $\mathcal{E}$ and all of the manifolds $\mathcal{M}_{\mathcal{C}}$.

Efficient IPF

[section not yet written].

20.1.3 Iterative scaling

We now turn to the general case in which the clique potentials are parameterized by arbitrary collections of features. As in Section 19.1, we drop the association of features to cliques in our notation, and simply consider a general exponential family model:

$$p(x \mid \theta) = \frac{1}{Z(\theta)} \exp\left\{\sum_i \theta_i f_i(x)\right\} \tag{20.22}$$

with features $f_i(x)$, where $Z(\theta)$ is the normalization factor:

$$Z(\theta) = \sum_x \exp\left\{\sum_i \theta_i f_i(x)\right\}. \tag{20.23}$$

The goal is to estimate the parameters θ_i from data $\mathcal{D}$.

We will work with the scaled likelihood function:

$$\tilde{l}(\theta \mid \mathcal{D}) = \sum_x \tilde{p}(x) \log p(x), \tag{20.24}$$

where $\tilde{l}(\theta \mid \mathcal{D}) = l(\theta \mid \mathcal{D})/N$ and $\tilde{p}(x)$ is the empirical distribution.

The *generalized iterative scaling (GIS)* algorithm is an iterative algorithm for finding maximum likelihood parameter estimates. GIS applies to arbitrary collections of features, subject to two constraints: the features must be nonnegative and they must sum to one. That is,

$$f_i(x) \geq 0, \qquad \sum_i f_i(x) = 1, \tag{20.25}$$

for each x. We will see later in this section how to maneuver around these constraints, but let us impose them for now.

The GIS algorithm forms a sequence of (unnormalized) distributions $p^{(t)}(x)$. The algorithm, which we derive later in this section, takes the following general form:

$$p^{(t+1)}(x) = p^{(t)}(x) \prod_i \left(\frac{\sum_x \tilde{p}(x) f_i(x)}{\sum_x p^{(t)}(x) f_i(x)} \right)^{f_i(x)}. \tag{20.26}$$

As in case of the IPF algorithm, GIS involves a ratio of expectations, in which the numerators are expectations with respect to the empirical distribution and the denominators are expectations with respect to the current model. These ratios are "update factors" that multiply the current distribution. Note an important difference between GIS and IPF, however: GIS is a fully "parallel" algorithm—the product in the algorithm is a product over all features.

Let us consider the relationship between IPF and GIS is somewhat more detail. Recall that the IPF algorithm is parallel at the level of a single clique, where we parameterize the clique potential $\psi_C(x_C)$ using a set of indicator features $f_i(x_C)$. For any given x, only one of these features are equal to one and the others are equal to zero. Thus we meet the constraints imposed in Eq. (20.25) and can apply GIS to this set of features. Moreover, the expectation of an indicator, $\sum_x p(x) f_i(x_C)$, is equal to the marginal probability, $p(x_C)$. From this it can easily be seen that Eq. (20.26) reduces to the IPF update Eq. (20.14). Thus, each step of IPF—which is parallel across the features corresponding to a single clique—is a GIS update. On the other hand, a sequence of IPF iterations corresponds to a sequence of GIS iterations with a changing set of features at each iteration.

Is it possible to obtain a fully parallel version of IPF, in which all of the features throughout the graph are updated in parallel? A problem arises here, which is that the sum across features, $\sum_i f_i(x)$, is no longer one in such a setting. (It is equal to the number of cliques, and hence is a constant, but not equal to one). As we mentioned above, however, it will be possible to maneuver around the sum-to-one constraint, and obtain a fully parallel IPF.

We return to the GIS algorithm. Although we have written the algorithm in terms of (unnormalized) joint probabilities, it is also possible to view the algorithm in terms of updates to individual parameters. Let the estimate of the parameter θ_i at iteration t be denoted $\theta_i^{(t)}$. Consider the following update:

$$\theta_i^{(t+1)} = \theta_i^{(t)} + \log \left(\frac{\sum_x \tilde{p}(x) f_i(x)}{\sum_x p^{(t)}(x) f_i(x)} \right). \tag{20.27}$$

Multiplying both sides by $f_i(x)$, and taking the exponential, we obtain:

$$\exp \left\{ (\theta_i^{(t+1)} - \theta_i^{(t)}) f_i(x) \right\} = \left(\frac{\sum_x \tilde{p}(x) f_i(x)}{\sum_x p^{(t)}(x) f_i(x)} \right)^{f_i(x)}. \tag{20.28}$$

If we multiply an exponential family distribution by the left-hand side of this equation, we change $\theta_i^{(t+1)}$ to $\theta_i^{(t)}$. Thus the right-hand side is the "update factor" for parameter θ_i. Multiplying by the update factors for all of the parameters, we recover the GIS algorithm.

What about normalization? In general the sequence of GIS iterates $p^{(t)}(x)$ are *not* normalized. This might seem problematic, in that we need to take expectations in the denominator of Eq. (20.26). But if we explicitly normalize both occurrences of $p^{(t)}(x)$ on the right-hand side of Eq. (20.26), it is easily verified that the normalization factors cancel (we carry out the calculation below). Thus we obtain the same result whether or not we use normalized distributions. Critical to this argument is the fact that $\sum_i f_i(x) = 1$.

In fact, even though the sequence of iterates are not normalized, the limiting distribution $p^{\infty}(x)$ *is* normalized. We prove this fact below.

Let us turn to a derivation of the GIS algorithm. Our derivation is in the spirit of our derivation of the EM algorithm in Chapter 11. In particular we make use of convexity to obtain an *auxiliary function*—a lower bound for the log likelihood. This bound is tight at the current values of the parameters, and thus by increasing the lower bound, we increase the log likelihood.

We begin by writing out the log likelihood:

$$\begin{aligned}
\tilde{l}(\theta \,|\, \mathcal{D}) &= \sum_x \tilde{p}(x) \log p(x) &(20.29)\\
&= \sum_x \tilde{p}(x) \sum_i \theta_i f_i(x) - \log Z(\theta) &(20.30)\\
&= \sum_i \theta_i \sum_x \tilde{p}(x) f_i(x) - \log Z(\theta). &(20.31)
\end{aligned}$$

As in the EM derivation, we must bound the log of a sum, but here the logarithm appears with a negative sign. To obtain a lower bound, we upper bound the logarithm using the elementary convexity result (see Figure 20.3), which yields:

$$\log Z(\theta) \leq \mu Z(\theta) - \log \mu - 1. \tag{20.32}$$

This bound holds for all μ, and in particular it holds for $\mu = Z^{-1}(\theta^{(t)})$. Thus we have:

$$\tilde{l}(\theta \,|\, \mathcal{D}) \geq \sum_x \tilde{p}(x) \sum_i \theta_i f_i(x) - \frac{Z(\theta)}{Z(\theta^{(t)})} - \log Z(\theta^{(t)}) + 1. \tag{20.33}$$

Note that the inequality is an equality at $Z(\theta^{(t)})$, which is verified by simple substitution. Moreover, given the smoothness of the logarithm we expect the bound to be reasonably tight in the neighborhood of $Z(\theta^{(t)})$.

Define $\Delta\theta_i^{(t)} \triangleq \theta_i - \theta_i^{(t)}$. Continuing the derivation, we have:

$$\begin{aligned}
\tilde{l}(\theta \,|\, \mathcal{D}) &= \sum_x \tilde{p}(x) \sum_i \theta_i f_i(x) - \frac{1}{Z(\theta^{(t)})} \sum_x \exp\left(\sum_i \theta_i f_i(x)\right) - \log Z(\theta^{(t)}) + 1\\
&= \sum_i \theta_i \sum_x \tilde{p}(x) f_i(x) - \frac{1}{Z(\theta^{(t)})} \sum_x \exp\left(\sum_i \theta_i^{(t)} f_i(x)\right) \exp\left(\sum_i \Delta\theta_i^{(t)} f_i(x)\right) - \log Z(\theta^{(t)}) + 1
\end{aligned}$$

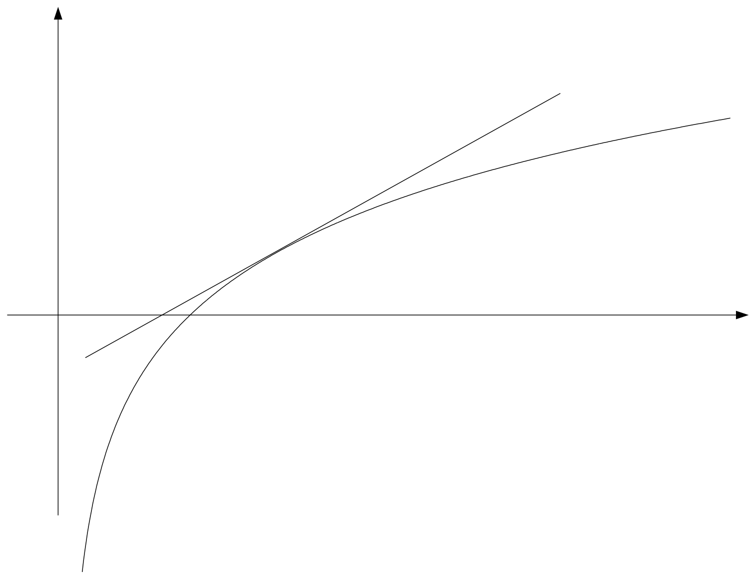

Figure 20.3: The convexity of the logarithm implies that we can upper bound the logarithm with a line: $\log x \leq \mu x - \log \mu - 1$. Moreover, the upper bound is tight at the point x when the slope μ matches the derivative of the logarithm at that point: $\mu = 1/x$.

$$= \sum_i \theta_i \sum_x \tilde{p}(x) f_i(x) - \sum_x p(x \,|\, \theta^{(t)}) \exp\left(\sum_i \Delta\theta_i^{(t)} f_i(x)\right) - \log Z(\theta^{(t)}) + 1. \tag{20.34}$$

We have a lower bound that has begun to take on the form of the difference between expectations. The exponential couples the parameters, however, and taking the derivative of this lower bound would not give the simple product form of the GIS update. To decouple the parameters, we need to invoke convexity a second time. In particular, we use Jensen's inequality:

$$\exp\left(\sum_i \pi_i x_i\right) \leq \sum_i \pi_i \exp\left(x_i\right), \tag{20.35}$$

for $\sum_i \pi = 1$. The $f_i(x)$ are positive and sum to one, and therefore can play the role of the π_i. We obtain:

$$\begin{aligned} \tilde{l}(\theta \,|\, \mathcal{D}) &\geq \sum_i \theta_i \sum_x \tilde{p}(x) f_i(x) - \sum_x p(x \,|\, \theta^{(t)}) \sum_i f_i(x) \exp(\Delta\theta_i^{(t)}) - \log Z(\theta^{(t)}) + 1 \\ &\triangleq \Lambda(\theta), \end{aligned} \tag{20.36}$$

a lower bound in which the parameters are decoupled. Note, moreover, that the bound continues to be exact at $\theta^{(t)}$.

Taking the derivative with respect to θ_i, we obtain:

$$\frac{\partial \Lambda}{\partial \theta_i} = \sum_x \tilde{p}(x) f_i(x) - e^{\Delta\theta_i^{(t)}} \sum_x p(x \,|\, \theta^{(t)}) f_i(x) \tag{20.37}$$

and setting to zero:

$$e^{\Delta\theta_i^{(t)}} = \frac{\sum_x \tilde{p}(x) f_i(x)}{\sum_x p(x \,|\, \theta^{(t)}) f_i(x)} = \frac{\sum_x \tilde{p}(x) f_i(x)}{\sum_x p^{(t)}(x) f_i(x)} Z(\theta^{(t)}) \tag{20.38}$$

where the second equation defines $p^{(t)}(x)$ as the unnormalized version of $p(x \,|\, \theta^{(t)})$.

To update the parameters from $\theta^{(t)}$ to $\theta^{(t+1)}$, we simply multiply $p(x \,|\, \theta^{(t)})$ by $e^{\Delta\theta_i^{(t)} f_i(x)}$ for all i. We obtain:

$$\begin{aligned}
p^{(t+1)}(x) &= \frac{p^{(t)}(x)}{Z(\theta^{(t)})} \prod_i \left(\frac{\sum_x \tilde{p}(x) f_i(x)}{\sum_x p^{(t)}(x) f_i(x)} Z(\theta^{(t)}) \right)^{f_i(x)} && (20.39)\\
&= \frac{p^{(t)}(x)}{Z(\theta^{(t)})} \prod_i \left(\frac{\sum_x \tilde{p}(x) f_i(x)}{\sum_x p^{(t)}(x) f_i(x)} \right)^{f_i(x)} \prod_i \left(Z(\theta^{(t)}) \right)^{f_i(x)} && (20.40)\\
&= p^{(t)}(x) \prod_i \left(\frac{\sum_x \tilde{p}(x) f_i(x)}{\sum_x p^{(t)}(x) f_i(x)} \right)^{f_i(x)}, && (20.41)
\end{aligned}$$

which is the GIS algorithm.

20.1.4 IPF for Gaussian models

[Section not yet written. An alternation between setting cliques of Σ to the empirical covariances and setting entries of Σ^{-1} to zero.]

20.2 Latent variables

In this final section, we turn to the problem of parameter estimation when there are latent variables. Solving such problems involves making use of the EM algorithm, with the junction tree algorithm (or some other inference algorithm) providing the E step, and the generalized iterative scaling algorithm (or some other estimation algorithm) providing the M step. In this section we bring all of these ideas together.

We begin by discussing the EM algorithm in the context of general exponential family models with latent variables, showing how the the M step can be implemented via the GIS algorithm. We then turn to the special case of directed graphical models, parameterized with generalized linear models at each node, where the IRLS algorithm provides the preferred method for implementing the M step. Finally we discuss some of the geometry associated with the EM algorithm.

20.2.1 Exponential family models

Let X_E represent the totality of observed variables in the problem and let X_H represent the unobserved or latent variables. Our goal is to estimate the parameters θ_i in a general exponential family model that includes both X_E and X_H:

$$p(x_E, x_H \,|\, \theta) = \frac{1}{Z(\theta)} \exp\left\{ \sum_i \theta_i f_i(x_E, x_H) \right\}. \tag{20.42}$$

This is the *complete data model.* We assume that only X_E is observed and that X_H is latent. Thus we wish to estimate the parameters by maximizing the *incomplete log likelihood*:

$$\tilde{l}(\theta \,|\, \mathcal{D}) = \sum_{x_E} \tilde{p}(x_E) \log p(x_E \,|\, \theta), \tag{20.43}$$

where $\mathcal{D}$ is the observed data x_E, and where $p(x_E \,|\, \theta)$ is the marginal probability obtained from by summing the complete data probability model over x_H.

The features $f_i(x_E, x_H)$ depend in general on subsets of the variables x_E and x_H. Although we have not specified these subsets in our notation, it is of course critically important to identify them and to exploit them in implementing an inference algorithm, a point that we return to in our discussion of the E step of EM.

Let us briefly review the EM framework from Chapter 11. Recall that we define the following KL divergence:

$$D(q \parallel \theta) \triangleq D\left(\tilde{p}(x_E) q(x_H \,|\, x_E) \parallel p(x_E, x_H \,|\, \theta)\right), \tag{20.44}$$

where $q(x_H \,|\, x_E)$ is an arbitrary conditional probability distribution. The EM algorithm is an alternating minimization algorithm composed of two steps:

$$(\textbf{E step}) \qquad q^{(t+1)} = \arg\min_q \; D(q \parallel \theta^{(t)}) \tag{20.45}$$

$$(\textbf{M step}) \qquad \theta^{(t+1)} = \arg\min_\theta \; D(q^{(t+1)} \parallel \theta). \tag{20.46}$$

In the remainder of this section we discuss how to implement the EM algorithm using the general tools for inference and estimation that we now have at our disposal.

The E step of the EM algorithm involves computing the optimal averaging distribution $q(x_H \,|\, x_E)$. As we saw in Chapter 11, this problem can be solved—at an abstract level—once and for all. The conditional probability that minimizes the KL divergence is simply:

$$q^{(t+1)}(x_H \,|\, x_E) = p(x_H \,|\, x_E, \theta^{(t)}), \tag{20.47}$$

the conditional under the current complete data model. Note, however, we do not need to obtain this conditional in any explicit sense; rather, we need to be able to use this conditional to compute averages. To understand more clearly what is involved in implementing the E step, it is first useful to understand what averages we need to compute, and for this we turn to the M step.

Only one of the terms in the KL divergence involves the parameters θ, and thus the M step can be written equivalently as follows:

$$\begin{aligned}
\theta^{(t+1)} &= \arg\max_{\theta}\ D(q^{(t+1)} \parallel \theta) && (20.48)\\
&= \arg\max_{\theta}\ \sum_{x_E}\sum_{x_H} \tilde{p}(x_E)p(x_H \mid x_E, \theta^{(t)}) \log p(x_E, x_H \mid \theta) && (20.49)\\
&= \arg\max_{\theta}\ \sum_{x_E}\sum_{x_H} \tilde{p}(x_E)p(x_H \mid x_E, \theta^{(t)}) \sum_i \theta_i f_i(x_E, x_H) - \log Z(\theta) && (20.50)\\
&= \arg\max_{\theta}\ \sum_i \theta_i \sum_{x_E}\sum_{x_H} \tilde{p}(x_E)p(x_H \mid x_E, \theta^{(t)}) f_i(x_E, x_H) - \log Z(\theta). && (20.51)
\end{aligned}$$

We see that we obtain an expression that is formally identical to the log likelihood that we obtained earlier for the complete data setting (cf. Eq. (20.31)). Whereas the earlier log likelihood involved the average of the features with respect to the empirical distribution $\tilde{p}(x)$, the current log likelihood involves the average of the features with respect to the joint distribution $\tilde{p}(x_E)p(x_H \mid x_E, \theta^{(t)})$. In the earlier case the features were denoted $f_i(x)$, and depended only on the observed variables x, whereas in the current setting the features are denoted $f_i(x_E, x_H)$, and depend on both the observed variables x_E and the latent variables x_H. In either case, however, we compute averages of the feature values; that is, we compute *expected sufficient statistics*. Once these averages are computed, we are left with the same function of the parameters θ_i to minimize.

This line of argument shows that the algorithms that we have developed in this chapter for complete data apply immediately to the general latent variable problem as the M step in the EM algorithm. In particular, we can use the GIS algorithm as an M step simply by replacing the expectations $\sum_x \tilde{p}(x) f_i(x)$ in the numerator of the GIS update in Eq. (20.26) by the expectations $\sum_{x_E}\sum_{x_H} \tilde{p}(x_E)p(x_H \mid x_E, \theta^{(t)}) f_i(x_E, x_H)$. That is, each GIS iteration takes the form:

$$p^{(t+1)}(x_E, x_H) = p^{(t)}(x_E, x_H) \prod_i \left(\frac{\sum_{x_E} \tilde{p}(x_E) p(x_H \mid x_E, \theta^{(t)}) f_i(x_E, x_H)}{\sum_{x_E} p^{(t)}(x_E, x_H) f_i(x_E, x_H)} \right)^{f_i(x_E, x_H)}. \tag{20.52}$$

Note, moreover, that each step of GIS increases the expected complete log likelihood. Overall we obtain an alternating minimization algorithm composed of an outer loop (the alternation between E and M steps) and an inner loop (the GIS iterations that implement the M step). Both the E step and the inner loop M steps decrease the objective function $D(q \parallel \theta)$ at each and every step.

If the features are indicator features of fully-parameterized clique potentials, then we can use the IPF algorithm to implement the inner loop. Each IPF step again decreases the objective function. Moreover, in this case we can view the overall algorithm as coordinate descent in the joint set of variables $q(x_H|x_E)$ and θ_i.

Let us now return to the E step. We have seen that the job of the E step is to compute the expected sufficient statistics. Thus we do not need $p(x_H \mid x_E, \theta^{(t)})$ explicitly, but we do need to be able to compute the expectations $\sum_{x_H} p(x_H \mid x_E, \theta^{(t)}) f_i(x_E, x_H)$.

There are two ways to implement the E step, corresponding to the two ways of constructing a graphical model to perform inference for an exponential family model (see Section ??). On the one hand, we can construct a graphical model by linking all nodes that appear together as the arguments for any feature $f_i(x_E, x_H)$. We then construct a junction tree based on this graphical model. A given feature $f_i(x_E, x_H)$ will then have its support contained entirely in one of the cliques of the junction tree. To compute the expectation of the feature it suffices to multiply the initial clique potential by $f_i(x_E, x_H)$ and to run CollectEvidence with that clique as root. If we wish to compute the expectations of all features in parallel, we run a full pass of CollectEvidence and DistributeEvidence, abbreviating the latter algorithm to pass messages only into branches of the junction tree that contain features to be evaluated.

20.2.2 Geometry of the EM algorithm

[Section not yet written. We define two manifolds:

$$\mathcal{V} = \{p(x_E, x_H) \ : \ p(x_E) = \tilde{p}(x_E)\}. \tag{20.53}$$

and

$$\mathcal{E} = \left\{ p(x_E, x_H) \ : \ p(x_E, x_H \mid \theta) = \frac{1}{Z(\theta)} h(x_E, x_H) \exp \left\{ \sum_i \theta_i f_i(x_E, x_H) \right\} \right\}. \tag{20.54}$$

which are nonintersecting, except in the uninteresting case that $p(x_E, x_H \mid \theta)$ is such that the variables X form a clique (in which case there's no need for the latent variables). EM alternates between these manifolds. Moreover, we can display an IPF inner loop as a sequence of I-projections on manifolds that intersect $\mathcal{E}$ and are successively redefined by the E step.]

20.3 Historical remarks and bibliography