Expectation Propagation in Factor Graphs: A Tutorial

Charles Sutton

October 28, 2005

Abstract

Expectation propagation is an important variational inference algorithm for graphical models, especially if some of the variables are continuous. This tutorial presents two views EP: as repeatedly projecting into an approximating family, and as a message-passing algorithm. We present EP in terms of factor graphs, which simplifies some of the presentation and provides concreteness, while remaining completely general. We give special emphasis to explaining why belief propagation is a special case of EP, and how EP can be used to approximate a factor graph's partition function.

1 Introduction

Expectation propagation [4] is an interesting variational inference algorithm for graphical models, especially if some of the variables are continuous. It can be viewed as a a generalization of belief propagation, a well-known message-passing algorithm that is exact if the model is a tree.

In continuous distributions, however, belief propagation can be problematic even for trees, because messages can be complicated continuous functions that are impossible to represent efficiently. (For example, in the clutter problem, mentioned in Section **??**, an exact message is a mixture of 2^n Gaussians, where n is the number of data points. [*TODO:* This discussion doesn't exist yet.])

Expectation propagation extends BP in two different ways. First, instead of sending an entire message, EP sends only a small number of expected sufficient statistics of the message, which is especially useful for continuous exponential families. Second, EP allows for structured approximations, so that regions of the graph send messages to each other, rather than just nodes and edges.

2 Factor Graphs

Factor graphs [1] are a generalization of Bayesian networks and Markov random fields that are especially useful for describing and implementing inference algorithms. Often, inference algorithms don't care about the pecularities of local versus global normalization that separate directed from undirected models. Most inference algorithms exploit only the fact that you have a function over many variables that factorizes into a product of local functions over only a few variables. Factor graphs are so useful because they encapsulate this insight. In this section, we provide a brief overview and notation for factor graphs.

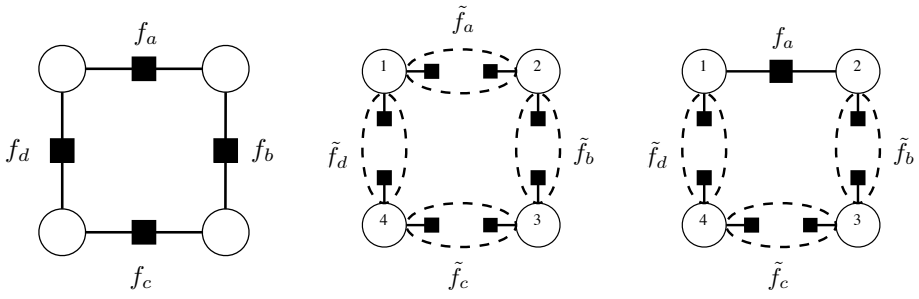

Figure 1: (a) An example factor graph with four variables and four pairwise factors. (b) A fully-factorized factor graph for approximating (a). This is the approximating factor graph used by BP. (c) A factor graph generated from (b) after one refinement step on factor f_a.

Factor graphs represent a multivariate distribution $p(\mathbf{x})$ by a product of local factors. A factor graph is a bipartite graph with two kinds of nodes: variable nodes, which correspond to components of $\mathbf{x}$, and factor nodes, which correspond to local functions over subsets of the variables. A variable node is connected to a factor node if that variable is a argument to the factor. The joint distribution corresponding to a factor graph $\mathcal{F}$ is defined by:

$$p(\mathbf{x}) = \frac{1}{Z} \prod_{\alpha} f_\alpha(\mathbf{x}_\alpha) \tag{1}$$

where α indexes the factor nodes in $\mathcal{F}$, $\mathbf{x}_\alpha$ are the components of $\mathbf{x}$ adjacent to α, and Z is called the *partition function* and is defined as

$$Z = \sum_{\mathbf{x}} \prod_{\alpha} f_\alpha(\mathbf{x}_\alpha). \tag{2}$$

If $\mathbf{x}$ is discrete, Z is a simple summation, but over an exponential number of assignments $\mathbf{x}$, so that computing it can be quite difficult. Indeed, computing Z is intractable in general. It should be easy to see how Markov random fields and Bayesian networks can be converted into factor graphs. For a Bayesian network, e.g., each local CPT is a factor, and $Z = 1$ if there is no evidence; otherwise Z is the probability of the evidence variables.

We will restrict the local factors to have a log-linear form

$$f_\alpha(\mathbf{x}_\alpha) = \exp\{< \theta, \phi(\mathbf{x}_\alpha) >\} \tag{3}$$

so that the factor graph defines an exponential family with sufficient statistics. In most of this tutorial, we will assume that $\mathbf{x}$ is discrete. But our discussion generalizes to exponential families of continuous distributions, and indeed this generalization is one of the chief advantages of EP.

This tutorial also assumes familiarity with belief propagation in factor graphs, also known as the sum-product algorithm. For discussion of BP, and more details on factor graphs, see [1]. *TODO:* Provide some background on BP.

3 Two views of EP

EP follows the following general approach: Given a factor graph whose marginal are difficult to compute, approximate it by a simpler graph whose marginals are easy to compute. This is done by replacing each of the factors f_α by an approximate factor $\tilde{f}_\alpha$, so the the approximation has a whole can be described by a graph with factors $\{\tilde{f}_\alpha\}$. You specify the parametric form of each $\tilde{f}_\alpha$ when designing an EP algorithm. In so doing, you define a family of possible approximating distributions, and EP selects a particular distribution from this family to approximate a given intractable distribution.

Typically, each of the $\tilde{f}_\alpha$ factorize further. For example, Figure 1(a) shows a grid shaped factor graph with pairwise factors $f_{st}(x_s, x_t)$, and Figure 1(b) shows an approximation where each factor f_{st} is replaced by two single-variable factors, i.e.,

$$f_{st}(x_s, x_t) \approx \tilde{f}_{st}(x_s, x_t) \stackrel{\text{def}}{=} \tilde{f}_{st;1}(x_s)\tilde{f}_{st;2}(x_t) \tag{4}$$

In the next two sections, we describe two different views of EP, the first in which EP repeatedly projects complex distribution down to the approximating family, and the second view, in which EP is a message-passing algorithm that sends expected sufficient statistics between regions—thus explaining the name "Expectation Propagation". It is this second view that explains why BP is a special case of EP.

3.1 EP as Projection

Once the form of the approximating distribution is chosen, choosing the best approximating distribution is no small task. Naively, one could do this by independently choosing each approximate factor $\tilde{f}_\alpha$ to most closely match f_α, e.g., by squared loss. This approximation tends to be bad, however; after all, if it were any good, then we shouldn't have bothered with the intractable model in the first place.

We can do better by making each of the term approximations depend on each other, so that later term approximations can adjust for errors in earlier ones. Perhaps the simplest way to do this is to approximate each of the factors one at a time. That is, start with an empty factor graph for each factor f_α, incorporate it into the current approximating distribution q by multiplication. This yields some new distribution $q^{+\alpha}$ that is not in the approximating family, so project it into the set of all approximating distributions using KL divergence. This procedure is illustrated in Figure **??**. This idea has arisen in the control, statistics, and artificial intelligence literatures; in control, it is called *assumed-density filtering*. The algorithm is:

function ADF ()
Initialize. Set $q^0(\mathbf{x})$ to uniform.
For each factor f_α,
1. *Refinement*. Incorporate exact factor f_α into the approximation:
   $q^{+\alpha} \propto q^{i-1} f_\alpha$

2. *Projection*. Compute the new approximate distribution by projecting $q^{+\alpha}$ into the approximating family:

$$q^i = \arg\min_q \mathrm{KL}(q^{+\alpha} \| q). \tag{5}$$

The disadvantage of this algorithm is that the quality of the approximation depends on the order in which factors are introduced, and in practice different choices of order have very different accuracy. To fix this, we can loop through the factors repeatedly, removing the previous factor approximating, incorporating the exact factor, and projecting down to the approximating family. This yields expectation propagation. The algorithm is as follows. For completeness, we also include an approximation of the partition function. We will explain this in Section 4.

algorithm EP

1. *Initialize.* Set $q_0(\mathbf{x})$ to uniform.

2. Repeat until converged:

   (a) Choose a factor f_α to refine.

   (b) *Refinement.* Remove the previous approximation $\tilde{f}_\alpha$ from $\tilde{\mathcal{F}}$, and replace it with the corresponding exact factor f_α. Graphically, this operation is depicted in Figure 1(b-c). Mathematically, we write this in two steps:

$$q^{\backslash\alpha}(\mathbf{x}) \propto \frac{q^{i-1}(\mathbf{x})}{\tilde{f}_\alpha(\mathbf{x}_\alpha)} \tag{6}$$

$$q^{+\alpha}(\mathbf{x}) \propto f_\alpha(\mathbf{x}_\alpha) q^{\backslash\alpha}(\mathbf{x}) \tag{7}$$

   The distribution $q^{\backslash\alpha}$ is sometimes called the *cavity distribution*. The distribution $q^{+\alpha}$ does not have a standard name, so we will refer to it as the *refined distribution*.

   (c) *Projection*. Compute the new approximate distribution by projecting $q^{+\alpha}$ into the approximating family:

$$q^i = \arg\min_q \mathrm{KL}(q^{+\alpha} \| q). \tag{8}$$

   Also, estimate a scale factor as:

$$C_\alpha = \frac{\int_{\mathbf{x}} f_\alpha(\mathbf{x}) \prod_{\beta\backslash\alpha} \tilde{f}_\alpha(\mathbf{x})}{\int_{\mathbf{x}} \tilde{f}_\alpha(\mathbf{x}) \prod_{\beta\backslash\alpha} \tilde{f}_\beta(\mathbf{x})}. \tag{9}$$

   If desired, approximate the partition function as:

$$Z \approx \sum_{\mathbf{x}} \prod_{\alpha} C_\alpha \tilde{f}_\alpha(\mathbf{x}_\alpha) \tag{10}$$

If the approximating factors are restricted to an exponential family, then the projection step is moment matching, i.e., at the minimizer q^i, we have

$$\mathbf{E}_{q^i}[\phi_{\alpha;k}(\mathbf{x})] = \mathbf{E}_{q^{+\alpha}}[(\phi_{\alpha;k}(\mathbf{x})], \tag{11}$$

for all sufficient statistics $\phi_{\alpha;k}$. This fact can be demonstrated by setting to 0 the gradient of the KL divergence in Equation 8.

The projection step can be confusing, because it might not be clear whether to understand the projection globally or locally. We have written the projection globally, meaning that during the refinement of $\tilde{f}_\alpha$, all factors in the approximation can potentially be modified. In the literature, the projection step is often written as:

$$\tilde{f}_\alpha^{\text{new}}(\mathbf{x}_\alpha) = \arg\min_{\tilde{f}_\alpha} \mathrm{KL}(q^{\backslash\alpha} f_\alpha \| q^{\backslash\alpha} \tilde{f}_\alpha), \tag{12}$$

meaning that the factor $\tilde{f}_\alpha$ that is currently being refined is the only approximating factor that can be modified during the projection step.

This confusion arises because the approximating model typically has special structure such that the single-factor optimizer of Equation 12 is also a optimizer of the global optimization problem in Equation 8. For example, this happens when all the approximating terms $\tilde{f}_\alpha$ belong to the same exponential family, that is, all are defined over the entire space $\mathbf{x}$, with the same sufficient statistics. For example, this situation can arise when approximating distributions of a single continuous variable, such as a multimodal continuous distribution approximated by a single Gaussian.

In general, by our choice of approximating factors, we can usually ensure that they all belong to the same exponential family. For example, suppose we are using the fully-factorized approximation of Figure 1(b). Then, instead of approximating $f_\alpha(x_1, x_2)$ by two univariate factors, we approximate it by a factor $\tilde{f}_\alpha(\mathbf{x})$ defined over the whole space but constrained to a fully-factorized:

$$\tilde{f}_\alpha(\mathbf{x}) \stackrel{\text{def}}{=} \prod_{s \in V} \tilde{f}_{\alpha;s}(x_s). \tag{13}$$

This constraint is readily enforced in the exponential family representation by choosing sufficient statistics that depend only on single variables.

3.2 EP as Message-Passing

To illustrate how EP can be viewed as a message-passing algorithm, we begin with an example. Suppose we are approximating the graph of 1(a) using a fully factorized approximation, as in Figure 1(b). All variables are discrete. However, we don't want to just use the four factors in Figure 1(b) directly, because then we'd have to assign those four univariate approximate factors to the four pairwise exact factors. Trying to approximate a pairwise factor by a univariate factor would be bad.

Instead, we use the approximating factor graph of Figure 1(b), in which each pairwise factor is approximated as a product of two univariate factors:

$$\tilde{f}_\alpha(x_s, x_t) \stackrel{\text{def}}{=} m_{st}(x_t) m_{ts}(x_s), \tag{14}$$

for all exact factors α over pairs (s,t). We name the univariate approximating factors $m_{st}(x_t)$ to suggest that they correspond to messages in a BP algorithm, which we will see.

Let's look at the EP algorithm for this case:

1. *Initialize.* Set $q_0(\mathbf{x})$ to uniform, i.e., set all $m_{st}(x_t)$ to 1.

2. Repeat until converged:

   (a) Choose a factor $\tilde{f}_\alpha(x_s, x_t) = m_{st}(x_t)m_{ts}(x_s)$ to refine.

   (b) *Refinement.* Replace the previous approximation $\tilde{f}_\alpha$ with the exact factor f_α, yielding the cavity distribution

$$q^{\setminus\alpha}(\mathbf{x}) \propto \frac{q^{i-1}}{\tilde{f}_\alpha} = \prod_{(u,v)\setminus(s,t)} m_{uv}(x_v)m_{vu}(x_u) \tag{15}$$

   and the refined distribution

$$q^{+\alpha}(\mathbf{x}) = f_\alpha(x_s, x_t) \prod_{(u,v)\setminus(s,t)} m_{uv}(x_v)m_{vu}(x_u), \tag{16}$$

   which is illustrated in Figure 1(c).

   (c) *Projection*. Compute the new approximate distribution by projecting $q^{+\alpha}$ into the approximating family:

$$q^i = \arg\min_q \mathrm{KL}(q^{+\alpha} \| q). \tag{17}$$

The main idea is that the projection step can be implemented by two BP message passes, even though we wrote the projection as minimizing KL-divergence between two global distributions. This is demonstrated in the following lemma.

Lemma 1. *The KL divergence KL*$(q^{+\alpha}\|q)$ *is minimized at:*

$$m_{st}^{new}(x_t) = \sum_{x_s} f_\alpha(x_s, x_t) \prod_{u \in N(s)\setminus t} m_{us}(x_s) \tag{18}$$

$$m_{ts}^{new}(x_s) = \sum_{x_t} f_\alpha(x_s, x_t) \prod_{u \in N(t)\setminus s} m_{ut}(x_t) \tag{19}$$

$$m_{uv}^{new}(x_v) = m_{uv}(x_v) \qquad \text{for all other } (u,v) \tag{20}$$

Proof. It is sufficient to ensure that the moment-matching constraints of Equation 11 are satisfied. We have assumed that each message $m_{uv}(x_v)$ are discrete multinomials, so their sufficient statistics are indicator functions of the form $\mathbf{1}_{X_v=x_v}$. This means that the moment-matching constraints amount to matching the marginal distributions of each variable:

$$q^{+\alpha}(x_u) = q(x_u) \qquad \text{for all } u, x_u. \tag{21}$$

For variables u that are not connected to the exact factor f_α, it is clear that leaving the messages unchanged preserves the marginal distribution, because the approximating factor graph is disconnected.

So all that remains is to show that $q^{+\alpha}(x_u) = q(x_u)$. But the message

$$m_{st}^{\text{new}}(x_t) = \sum_{x_s} f_\alpha(x_s, x_t) \prod_{u \in N(s) \setminus t} m_{us}(x_s) \tag{22}$$

is exactly the marginal distribution of X_t with respect to $q^{+\alpha}$ (as if calculated by direct summation) because it sums over all variables and factors of $q^{+\alpha}$ except for those that are disconnected from X_t in the factor graph. A similar argument holds for X_s, completing the proof. □

To summarize, we have seen why BP can be readily represented as an EP algorithm. Although our example used only pairwise factors, it is easy to generalize to factors of larger cardinality.

TODO: As another example, let's take the same graphical structure in Figure 1, but assume the exact factors are all Gaussian.

TODO: This message-passing view can be applied to any EP algorithm, not just special cases like BP [7].

TODO: As a more complex example, let us consider tree-structured EP [3].

4 Approximating the Partition Function

Often, we want an approximation not only of the marginals, but also the partition function. For example, in a directed model, the partition function corresponds to the probability of the observed evidence. In this section, we discuss three different ways to do this, of increasing sophistication.

To get intuition, it helps to think of a univariate continuous distribution $p(x) \propto \prod_i f_i(x)$. The partition function is the area underneath the function

$$I(x) = \prod_i f_i(x). \tag{23}$$

Since EP provides an approximation to each term f_i, it seems reasonable to approximate the area $\int_{\mathbf{x}} I(x)$ as

$$Z \approx \int_{\mathbf{x}} \tilde{I}(x) = \int_{\mathbf{x}} \prod_\alpha \tilde{f}_\alpha(x). \tag{24}$$

The problem with this first approximation is that in EP, we selecting the $\tilde{f}_i$ to match the refined distribution once normalized, not to match its scale (and hence its integral) before normalization. To fix this, we estimate a scaling constant C_α whenever we refine a factor f_α. We choose C so that the area under the refined distribution equals the area under new approximation, i.e., so that

$$\int_{\mathbf{x}} f_\alpha(\mathbf{x}) \prod_{\beta \setminus \alpha} \tilde{f}_\alpha(\mathbf{x}) = C_\alpha \int_{\mathbf{x}} \tilde{f}_\alpha(\mathbf{x}) \prod_{\beta \setminus \alpha} \tilde{f}_\beta(\mathbf{x}). \tag{25}$$

All of these integrals are over the approximating distributions, so they should be readily computable. Essentially, this amounts to including the function $\phi(\mathbf{x}) = 1$ as a sufficient statistic in all approximating factors, and running EP as usual.

A third approximation arises from the EP free energy. EP can be viewed as optimizing a particular cost function on probability distributions, called a *free energy*. There is an exact version of the free energy whose optimum is $-\log Z$; optimizing this is clearly intractable, because computing Z is. The EP free energy is an approximation to the exact free energy, so when EP converges, we can take the value of the EP free energy as an approximation to $-\log Z$. This is especially useful when the model parameters are themselves being estimated outside of EP, for just as the partial derivatives of $\log Z$ yield expected sufficient statistics, the partial derivatives of the free energy yield expected sufficient statistics, where the expectation is taken with respect to the approximate marginal distribution. We will not go into this in detail (at least in this version of the tutorial).

TODO: Compare these on the clutter problem.

Related Work

EP was originally introduced by Minka [4, 2]. A detailed derivation of EP for a Gaussian mixture problem is given in an unpublished note by Murphy [5]. Another tutorial on EP focusing on the exponential family perspective appears in Seeger's thesis [6].

References

[1] F. R. Kschischang, B. J. Frey, and H. A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 47(2):498–519, 2001.

[2] Thomas Minka. Expectation propagation for approximate bayesian inference. In *17th Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 362–369, 2001.

[3] Thomas Minka and Yuan Qi. Tree-structured approximations by expectation propagation. In Sebastian Thrun, Lawrence Saul, and Bernhard Schölkopf, editors, *Advances in Neural Information Processing Systems 16*. MIT Press, Cambridge, MA, 2004.

[4] Tom Minka. *A family of algorithms for approximate Bayesian inference*. PhD thesis, MIT, 2001.

[5] Kevin Murphy. From belief propagation to expectation propagation. Available at `http://www.cs.ubc.ca/~murphyk/Papers/EP.ps.gz`, September 2001.

[6] Matthias Seeger. *Bayesian Gaussian Process Models: PAC-Bayesian Generalisation Error Bounds and Sparse Approximations*. PhD thesis, University of Edinburgh, 2003.

[7] Max Welling, Tom Minka, and Yee Whye Teh. Structured region graphs: Morphing EP into GBP. In *21st Conference on Uncertainty in Artificial Intelligence (UAI)*, 2005.