Structured Variational Inference Procedures and their Realizations

Dan Geiger[*]
Computer Science Department
Technion
Haifa, 36000, Israel
dang@cs.technion.ac.il

Christopher Meek
Microsoft Research
Microsoft Cooperation
Redmond, WA 98052, USA
meek@microsoft.com

Abstract

We describe and prove the convergence of several algorithms for approximate structured variational inference. We discuss the computation cost of these algorithms and describe their relationship to the mean-field and generalized-mean-field variational approaches and other structured variational methods.

1 Introduction

Graphical models are an important class of probabilistic models. Their graphical structure, whether directed, undirected, or mixed, provides an appealing description of the qualitative properties of the model. Furthermore, the modularity of the defined probability distribution allows one to define general algorithms, called inference algorithms, for computing marginal and conditional probabilities and allows one to easily incorporate prior knowledge. Inference algorithms are also useful for parameter learning for graphical models with missing data because the E-step of the EM algorithm can be computed using inference.

Although the inference problem is tractable for graphical models with small treewidth, the general inference problem is NP-hard (Cooper, 1990; Dagum and Luby, 1993) . In fact, for many graphical models of interest the treewidth is too large to allow efficient inference and one must use approximate or heuristic inference methods. In this paper, we examine the family of approaches that optimize the KL divergence between a distribution Q and the target distribution P where Q is constrained to be from some family of distributions for which inference is tractable.

[*] This work was partially done while the author visited Microsoft Research.

One of the nice properties of this family of approaches is that they provide a bound on marginal probabilities that are useful in model evaluation and learning. In particular, let us assume that we are given an intractable joint distribution $P(X)$ over a set of discrete variables X and our goal is to compute the marginal probability $P(Y = y)$ where $Y \subseteq X$. We let $H = X \setminus Y$. The quantity of interest is bounded by $\log P(Y = y) \geq -D(Q(H) \,\|\, P(Y = y, H))$ where $D(\cdot \,\|\, \cdot)$ denotes the KL divergence between two probability distributions. The quantity $-D(Q \,\|\, P)$ is often called the free-energy and denoted by $F(Q; P)$ where Q and P are possibly un-normalized distributions. The bound can be shown by the following argument:

$$\begin{aligned}
-D(Q(H) \,\|\, P(Y = y, H)) &= -\sum_h Q(h) \log \frac{Q(h)}{P(y, h)} \\
&= \sum_h Q(h) \log P(y) - \sum_h Q(h) \log \frac{Q(h)}{P(h|y)} \\
&= \log P(y) - D(Q(H) \,\|\, P(H|Y = y)) \leq \log P(y).
\end{aligned}$$

The final inequality follows from the fact that $D(Q(H) \,\|\, P(H|Y = y)) \geq 0$ with equality holding only if $Q(H) = P(H|Y = y)$. It is important to note that if Q is tractable then $D(Q(H) \,\|\, P(Y = y, H))$ can be effectively computed. The goal of approaches in this family is to find the $Q(H)$ that minimizes $D(Q(H) \,\|\, P(Y = y, H))$ (or maximizes the free-energy). Approaches in this family include the mean field, generalized mean field, and structured mean field approaches to variational inference. These methods differ with respect to the family of approximating distributions that can be used with the structural mean field approach subsuming the remaining approaches as special cases.

In this paper, we develop a set of structural variational methods inspired by the sequence of papers Saul and Jordan (1996), Ghahramani and Jordan (1997), Wiegerinck (2000) and Bishop and Winn (2003). We make several contributions with respect to this earlier work. We provide a set of alternative structured variational methods and prove convergence of the alternatives with a novel simple proof technique. Our alternative algorithms differ in their computational profile with successive algorithms providing refined control over the computational cost of obtaining a variational approximation. We note that special cases of our final algorithm, called VIP$^\sharp$, were used in Jojic et al. (2004) for applying variational inference techniques to types of phylogenic models. For $N \times N$ grid-like models, algorithm VIP$^\sharp$ is $4N$ fold faster than algorithm VIP$^+$ and $12N$ folder faster than algorithm VIP, yielding a potential three orders of magnitude improvement in applications such as phylogeny and genetic linkage analyses.

2 Single Potential Update Algorithms

We denote distributions by $P(x)$ and $Q(x)$ and related un-normalized distributions by $\tilde{P}(x) \propto P(X)$ and $\tilde{Q}(x) \propto Q(x)$. Let X be a set of variables and x be an instantiation of these variables. Let $P(x) = \frac{1}{Z_P} \prod_i \Psi_i(d_i)$ where d_i is the projection of the instantiation x to the variables in $D_i \subseteq X$. The constant Z_P normalizes the product of potentials and the subsets $\{D_i\}_{i=1}^I$ are allowed to be overlapping. Note that we often suppress the arguments of a potential and of a distribution, using Φ_j instead of $\Phi_j(c_j)$ and P instead of $P(X)$.

Our goal is to find a distribution Q that minimizes the KL distance between Q and P. We further constrain Q to be of the form $Q(x) = \frac{1}{Z_Q} \prod_j \Phi_j(c_j)$ where Z_Q is a normalizing constant and where $C_1, \ldots, C_J$ are possibly overlapping subsets of X, which we call clusters. Finding an optimum Q, however, can be difficult. We set a more modest goal of finding a distribution Q which is a stationary point for the KL distance between Q and P, that is, $\nabla_\Phi D(Q \,\|\, P) = 0$ where $\Phi = \{\Phi_j\}_j$.

An algorithm, called VIP (for Variational Inference Procedure), that finds such a distribution Q is given in Figure 1. The algorithm uses the following indicator functions: $g_{kj} = 0$ if $C_k \cap C_j = \emptyset$ and 1 otherwise, and $f_{ij} = 0$ if $D_i \cap C_j = \emptyset$, and 1 otherwise. VIP relies at each step on an (inference) algorithm to compute some conditional probabilities from an un-normalized distribution $\tilde{Q}$ represented by a set of potentials $\Phi_j(c_j)$, $j = 1, \ldots, J$. This is accomplished by using *bucket elimination algorithm* or the *sum-product algorithm* described in (Dechter, 1999; Kschischang et al., 2001) as follows. To compute $Q(a|b)$ the algorithm first computes $\tilde{Q}(a,b)$ and then $\tilde{Q}(b)$. The conditional distribution of interest is the ratio of these two quantities because the normalizing constant cancels. It is important to note that for $\tilde{Q}(x) = \prod_j \Phi_j(c_j)$ the computation of these conditionals is not affected by multiplying any Φ_j by a constant α.

Algorithm VIP generalizes the mean field (MF) algorithm and the generalized mean field (GMF) algorithm (Xing et al. 2003,2004). The mean field algorithm is the special case of VIP in which each C_j contains a single variable. Similarly, the generalized mean field algorithm is the special case in which the C_j are disjoint subsets of variables. Note that if C_j are disjoint clusters, then the formula for γ_j in VIP simplifies to the GMF equations as follows (first term drops out):

$$\gamma_j(c_j) \leftarrow \sum_{\{i: f_{ij}=1\}} \sum_{D_i \setminus C_j} Q(d_i|c_j) \log \psi_i(d_i). \tag{2}$$

The term $Q(d_i|c_j)$ can be made more explicit when C_j are disjoint clusters. In particular, we partition the set $D_i \setminus C_j$ into $D_i^k = (D_i \setminus C_j) \cap C_k$ for $k = 1, \ldots, J$ where $k \neq j$. Note that $D_i^k = D_i \cap C_k$. Using this notation we have $Q(d_i|c_j) = \prod_k Q(d_j^k)$ where $Q(d_j^k) = 1$ whenever $D_i^k = \emptyset$. This factorization further simplifies the formula for γ_j in VIP as follows:

$$\gamma_j(c_j) \leftarrow \sum_{\{i: f_{ij}=1\}} \sum_{D_i^1} Q(d_i^1) \ldots \sum_{D_i^J} Q(d_i^J) \log \psi_i(d_i) \tag{3}$$

We note that this simplification is achieved automatically by the usage of bucket elimination for computing γ_j. The iterated sums in Eq. 3 are in fact the buckets formed by bucket elimination when C_j are disjoint.

Wiegerinck (2000) presents a less refined version of the update equation (Equation 1) and proves convergence to a stationary point of the KL distance between Q and P among all distributions Q of the given form using this update equation. We provide an alternative novel proof of convergence for our refined version of Wiegerinck's algorithm in Section 4 as a corollary to Theorem 1.

Equation 1 of VIP requires the computation of the quantities $Q(c_k|c_j)$ and $Q(d_i|c_j)$, and this is done in VIP using the bucket elimination algorithm. However, because there could be many indices k such that $C_k \cap C_j$ is not empty, and many indices i such that $D_i \cap C_j$ is not empty, these function calls are repeatedly applied independent of each other. However, these computations share many sub-computations, and it is therefore reasonable to add a data structure to facilitate a more efficient implementation for these function calls. In particular, it is possible to save computations if the sets $C_1, \ldots, C_J$ form a junction tree.

A set of clusters $C_1, \ldots, C_J$ forms a *junction tree* iff there exists a tree JT having one node, called C_j, for each subset of variables C_j, and for every two nodes C_i and C_j of JT, which are connected with a path in JT,

```
Algorithm VIP$(Q, P)$
Input: A set of potentials $\Psi_i(d_i)$ defining a probability distribution $P$ via $P(x) = \frac{1}{Z_P}\prod_i \Psi_i(d_i)$ and a set of clusters $C_j$, $j = 1, \ldots, J$, with initial non-negative potentials $\Phi_j(c_j)$.
Output: A revised set of potentials $\Phi_j(c_j)$ defining a probability distribution $Q$ via $Q(x) = \frac{1}{Z_Q}\prod_j \Phi_j(c_j)$ where $Z_Q$ is a normalizing constant, such that $Q$ is a stationary point of the KL distance $D(Q \parallel P)$.

Iterate over all clusters $C_j$ until convergence
    For every instantiation $c_j$ of cluster $C_j$ do:
        $\gamma_j(c_j) \leftarrow -\sum_{\{k:g_{kj}=1\}}\sum_{C_k \setminus C_j} Q(c_k|c_j)\log Q(c_k|c_j) + \sum_{\{i:f_{ij}=1\}}\sum_{D_i \setminus C_j} Q(d_i|c_j)\log \Psi_i(d_i)$    (1)
            using the sum-product algorithm on $\tilde{Q}(X) = \prod_i \Phi_i(C_i)$
            to compute the quantities $Q(c_k|c_j)$ and $Q(d_i|c_j)$.
    $\Phi_j(c_j) \leftarrow e^{\gamma_j(c_j)}$
```

Figure 1. The VIP algorithm

and for each node C_k on this path, $C_i \cap C_j \subseteq C_k$ holds. By a tree we mean an undirected graph, not necessarily connected, with no cycles. Note that this definition allows a junction tree to be a disconnected graph. When $C_1, \ldots, C_J$ form a junction tree, $Q(x)$ has the decomposable form $Q(x) = \prod_j \Phi_j(c_j)/\prod_e \Phi_e(s_e)$, where Φ_j are marginals on the subsets C_j of X, $j = 1, \ldots, J$, and where Φ_e are the marginals on intersections $S_e = C_i \cap C_j$, one for each two neighboring clusters in the junction tree (Jensen 1996).

The revised algorithm, which we call VIP$^+$, maintains a consistent junction tree JT for the distribution $Q(x)$. By consistency we mean that $\sum_{C_j \setminus C_k} \Phi_j = \sum_{C_k \setminus C_j} \Phi_k$ for every two clusters. In a consistent junction tree, each potential $\Phi_j(C_j)$ is proportional to $Q(C_j)$. There are two standard operations for junction trees: DistributeEvidence(Φ_j), and CollectEvidence(Φ_j) (Jensen 1996). Algorithm VIP$^+$ uses the former. The procedure DistributeEvidence(Φ_j) accepts as input a consistent junction tree and a new cluster marginal Φ_j for C_j, and outputs a consistent junction tree, having the same clusters, where Φ_j is the (possibly unnormalized) marginal probability of Q on C_j, and where the conditional probability $Q(X|C_j)$ remains unchanged. Algorithm VIP$^+$ is given in Figure 2. This algorithm is identical to Wiegerink's algorithm except that the normalizing constant is not computed in each iteration. The fact that algorithm VIP$^+$ converges to a distribution Q which is a stationary point of the KL distance $D(Q \parallel P)$ is proved in Section 4 in Theorem 1.

Next we compare the computational benefit of VIP$^+$ versus VIP. The algorithms differ in two ways. First, VIP$^+$ makes the junction tree consistent with respect to the updated cluster. Second, VIP$^+$ uses junction tree inference to compute the quantities $Q(c_k|c_j)$ and $Q(d_i|c_j)$ whereas VIP uses the sum-product algorithm.

Most of the computation in both algorithms is directed towards computing conditional probabilities $Q(c_k|c_j)$ and $Q(d_i|c_j)$. We distinguish among these conditional probabilities as follows.

Definition: A conditional probability $Q(A|c_j)$ is *subsumed* by Q if the set of target variables A is a subset of some cluster C_k in Q (i.e., $A \setminus C_j \subseteq C_k$).

In the non-subsumed case, the set of target variables spans multiple clusters (i.e., $A \setminus C_j \not\subseteq C_k$). Clearly, all probabilities of the form $Q(C_k|c_j)$ are subsumed by Q.

The cost of running both the junction tree algorithm and the sum-product algorithm to compute a subsumed conditional probability $Q(A|c_j)$ is exponential in the treewidth of the model Q. The cost of the junction tree algorithm is typically twice the cost of the sum-product algorithm but, as we see below, this extra factor can be useful in reducing overall costs. For non-subsumed conditionals, both algorithms can cost upto a multiplicative factor of the size of the non-subsumed set. In the case of using junction trees, one can use the *variable propagation* algorithm in Jensen (1996) for each non-subsumed conditional.

The next example highlights the computational difference between VIP and VIP$^+$.

Example 1 *The target distribution P is a square grid of pairwise potentials (see Figure 3a) and the approximating family is defined by the set of columns in the*

```
Algorithm VIP$^+(Q,P)$

Input: A set of potentials $\Psi_i(d_i)$ defining a probability distribution $P$ via $P(x) = \frac{1}{Z_P}\prod_i \Psi_i(d_i)$ and a set of clusters $C_j$, $j = 1,\ldots,J$, with initial potentials $\Phi_j(c_j)$ that form a consistent junction tree $JT$.

Output: A revised set of potentials $\Phi_j(c_j)$ defining a probability distribution $Q$ via $Q(x) = \prod_j \Phi_j(c_j)/\prod_e \Phi_e(s_e)$, such that $Q$ is a stationary point of the KL distance $D(Q \,\|\, P)$.

Note: The potentials $\Phi_j$ are consistent un-normalized marginals encoding $\tilde{Q} \propto Q$. This fact is an invariant of the loop due to initialization and Step 2.

Iterate over all clusters $C_j$ until convergence

Step 1.    For every instantiation $c_j$ of cluster $C_j$ do:

        $\gamma_j(c_j) \leftarrow -\sum_{\{k:g_{kj}=1\}} \sum_{C_k \setminus C_j} Q(c_k|c_j)\log Q(c_k|c_j) + \sum_{\{i:f_{ij}=1\}} \sum_{D_i \setminus C_j} Q(d_i|c_j)\log \Psi_i(d_i)$    (4)

            where the quantities $Q(c_k|c_j)$ and $Q(d_i|c_j)$ are computed via the junction tree
            algorithm operating on $JT$.

        $\Phi_j(c_j) \leftarrow e^{\gamma_j(c_j)}$

Step 2. Make $JT$ consistent with respect to $\Phi_j$:    DistributeEvidence($\Phi_j$)
```

Figure 2. The VIP$^+$ algorithm

grid and denoted by Q_F (see Figure 3b) in which the clusters C_i ($i = 1,\ldots,30$) correspond to edges.

Note that all conditionals required by the algorithms when optimizing Q_F are subsumed. In this example, for each c_j not on the boundary, there are six conditional probabilities that need to be computed. By using the junction tree algorithm all of these conditional probabilities can be computed with one call to DistributeEvidence whereas, when using the sum-product algorithm, each of these is computed separately. This yields a 3-fold speed up for VIP$^+$ with respect to VIP. For those c_j on a boundary, the speedup is a factor less than 3. As the size of the grid grows, a smaller fraction of the edges are on the boundary, and, thus, the speedup approaches a 3-fold speedup. For small grids, VIP$^+$ can be slower than VIP.

3 Multiple Potential Update Algorithm

In this section, we develop an algorithm to update multiple potentials at once to reduce the computational cost of optimizing the Q distribution. Algorithms VIP and VIP$^+$ do not assume any structure for Φ_j, namely, these algorithms hold tables Φ_j with an explicit entry for every instantiation of C_j. Since the computations $Q(c_k|c_j)$ and $Q(d_i|c_j)$ grow exponentially in the size of D_i and C_k, these algorithms become infeasible for large cliques or clusters. However, when structure is added to Φ_j, these algorithms can be modified to be more efficient by simultaneously updating this structure. In particular, one can use structure of the form,

$$\Phi_j(c_j) = \prod_{l=1}^{n_j} \Phi_{jl}(c_{jl}),$$

where the sets C_{jl} are possibly overlapping subsets of C_j, and c_{jl} is the projection of the instantiation c_j on the variables in C_{jl}. The potentials Φ_{jl} are assumed to be full tables and to form a junction tree JT$_j$.

Central to our development is a compatibility condition which allows us to simultaneously update the potentials.

Definition: A distribution Q with clusters C_j and subsets C_{jl} is *compatible* with a distribution P with sets D_i if for every D_i and C_j the set of indices $B_{ij} = \{l : D_i \cap C_j \subseteq C_{jl}\}$ is non-empty.

Our refined algorithm, VIP$^\sharp$, given in Figure 4, uses an indicator function $f_{ij}(l)$ which equals 0 when $D_i \cap C_j = \emptyset$, and when $D_i \cap C_j \neq \emptyset$, it equals 1 for a single fixed index $l \in B_{ij}$ and 0 for all other indices in B_{ij}. Our algorithm is closely related to the algorithm in Bishop and Winn (2003). As in Bishop and Winn (2003), we assume that the clusters of the approximating distribution are independent, that is, $Q(C_k|c_j) = Q(C_k)$. Our algorithm also generalizes the algorithm employed in (Jojic et al. 2004), which concentrates on specific models for phylogenetic analysis. We prove convergence of VIP$^\sharp$ in Section 4.

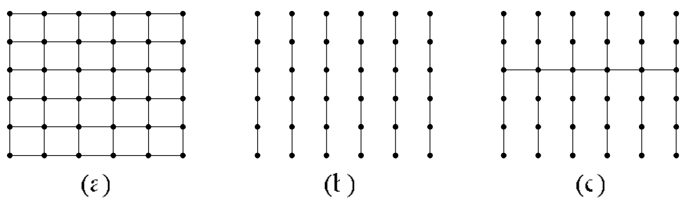

Figure 3. (a) Grid-like P distribution (b) factored structured distribution Q_F (c) connected distribution Q_C.

Example 2 *The target distribution P is a square grid of pairwise potentials (see Figure 3a) and the approximating family is a defined by the set of columns in the grid and denoted by Q_F (see Figure 3b) where C_i ($i = 1, \ldots, 6$) are columns of the grid.*

The approximating family with clusters defined by columns in this example satisfy the compatibility condition and the independence condition required by our algorithm.

Example 3 *The target distribution P is a square grid of pairwise potentials (see Figure 3a) and the approximating family is a defined by the set of columns in the grid and denoted by Q_C (see Figure 3c) where C_1 is the connected row of the grid and C_i ($i = 2, \ldots, 7$) are columns of the grid.*

The approximating family defined in Example 3 satisfies the compatibility condition but not the independence condition required by our algorithm. Note that, while the approximating family in Example 3 cannot be optimized using our refined algorithm below, it can be optimized using either VIP or VIP$^+$ in which, for instance, the C_i each contain a single edge.

We use Examples 1 and 2 to compare the benefits of VIP$^\sharp$ as compared to VIP$^+$. To analyze the difference between VIP$^\sharp$ and VIP$^+$ we need to analyze the number of times that one needs to call DistributeEvidence while computing conditional and marginal probabilities.

We begin by noting that the update Equation 5 in VIP$^\sharp$ takes advantage of the strong independence assumption to factor $Q(D_i|c_j)$, yielding a set of marginal probabilities that do not depend on c_j. Furthermore, the assumption of compatibility between P and Q implies that all the conditionals are subsumed. The factorization and compatibility conditions imply that each of these marginal probabilities can be obtained by lookup from the appropriate junction tree without calling DistributeEvidence. Therefore, we need no calls to DistributeEvidence in Step 1 and only one call to an inference algorithm to calibrate the junction tree associated with the potential being updated (Step 2).

In VIP$^+$, for each cluster C_j (edge) not in the boundary of the grid, there are twenty four conditionals that we need to compute, six for each of the four possible values for the cluster C_j. Every group of six conditionals can be updated with a single call to DistributeEvidence for a given c_j which gives four calls to the DistributeEvidence per cluster (edge). Again, as the size of the $N \times N$ grid grows, a smaller fraction of the edges are on the boundary, and, thus, the speedup approaches a 4N-fold speedup for VIP$^\sharp$ as compared to VIP$^+$, and 12N-fold as compared to VIP.

4 Proof of Convergence

In order to prove convergence of our algorithms, namely, that they converge to a stationary point of the KL distance between Q and P among all distributions Q of the given form, we examine properties of the KL distance between two distributions Q and P. Our proof technique is novel in that it uses properties of the KL distance rather than being based on Lagrangians (e.g., Wiegerinck 2000). The following lemmas furnish the needed properties of KL via basic algebra.

Lemma 1 *Let $P(x) = \frac{1}{Z_P}\prod_i \Psi_i(c_i)$ and $Q(x) = \frac{1}{Z_Q}\prod_j \Phi_j(d_j)$. Then,*

$$D(Q \,\|\, P) = \sum_{C_j} Q(c_j) \log \frac{Q(c_j)}{\Gamma_j(c_j)} + \log(Z_P) \tag{6}$$

where $\Gamma_j(c_j) = e^{\gamma_j(c_j)}$ and where

$$\gamma_j(c_j) = -\sum_k \sum_{C_k \setminus C_j} Q(c_k|c_j) \log Q(c_k|c_j) + \sum_i \sum_{D_i \setminus C_j} Q(d_i|c_j) \log \Psi_i(d_i).$$

Proof: Recall that

$$D(Q \,\|\, P) = \sum_x Q(x) \log \frac{Q(x)}{P(x)} = -\left[H(Q) + E_Q[\log P(x)]\right] \tag{7}$$

```
ALGORITHM VIP$^\sharp$ (Q,P)
Input: A set of disjoint clusters $C_j$, $j = 1, \dots, J$ and a nested structure $C_{jl}$ ($l = 1 \dots, n_j$) where $Q(c) \propto \prod_{j,l} \Phi_{jl}(c_{jl})$. A set of potentials $\Psi_i(d_i)$ defining a probability distribution $P$ via $P(x) = \frac{1}{Z_P} \prod_i \Psi_i(d_i)$ such that the potentials are compatible with $Q$. A set of junction trees ($\mathrm{JT}_j$) for each cluster $C_j$ and a set of initial potentials $\Phi_j(c_j) = \prod_{l=1}^{n_j} \Phi_{jl}(c_{jl})$.
Output: A revised set of potentials $\Phi_{jl}(c_j)$ defining a probability distribution $Q$ via $Q(x) = \prod_j Q_j(c_j)$ where $Q(c_j) \propto \prod_{l=1}^{n_j} \Phi_{jl}(c_{jl})$ such that $Q$ is a stationary point of $D(Q \,\|\, P)$.

Iterate over all clusters $C_j$ until convergence

Step 1. Compute messages for $l = 1, \dots, n_j$:
        For every instantiation $c_{jl}$ of $C_{jl}$ do:

            $\gamma_{jl}(c_{jl}) \leftarrow \sum_{\{i : f_{ij}(l)=l\}} \sum_{D_i^1} Q(d_i^1) \dots \sum_{D_i^J} Q(d_i^J) \log \Psi_i(d_i)$        (5)

            where the quantities $Q(d_i^k)$ can be obtained by lookup in $\mathrm{JT}_k$ for $\Phi_k$.

            $\Phi_{jl}(c_{jl}) \leftarrow e^{\gamma_{jl}(c_{jl})}$

    Note: $\Phi_{jl}(c_{jl})$, $l = 1, \dots, n_j$, implicitly encode the potential $\Phi_j(c_j)$, which is not being held explicitly anymore as in VIP, via $\Phi_j(c_j) = \prod_{l=1}^{n_j} \Phi_{jl}(c_{jl})$. Recall that $D_i^k = D_i \cap C_k$.

Step 2. Make $JT_j$ consistent with respect to $\Phi_j$:    DISTRIBUTEEVIDENCE($\Phi_j$)
```

Figure 4. The VIP$^\sharp$ algorithm

where $H(Q)$ denotes the entropy of $Q(x)$ and E_Q denotes expectation with respect to Q. The entropy term can be written as

$$H(Q) = -\sum_{C_j} Q(c_j) \log Q(c_j) - \sum_{C_j} Q(c_j) \sum_{X \setminus C_j} Q(x|c_j) \log Q(x|c_j)$$

where the first term is the entropy of $Q(C_j)$ and the second term is the conditional entropy of $Q(X|C_j)$. This well known form of $H(Q)$ is derived by splitting summation over X into summation over C_j and over $X \setminus C_j$, and using the fact that $\sum_{X \setminus C_j} Q(x|c_j) = 1$. By splitting the sum over $X \setminus C_j$, this entropy term is further rewritten as

$$H(Q) = -\sum_{C_j} Q(c_j) \log Q(c_j) - \sum_{C_j} Q(c_j) \sum_k \sum_{C_k \setminus C_j} Q(c_k|c_j) \log Q(c_k|c_j).$$

The second term of Eq. 7 is similarly written as

$$\begin{aligned} E_Q[\log P(x)] &= \\ &= \sum_i \sum_{C_j} Q(c_j) \sum_{X \setminus C_j} Q(x|c_j) \log \Psi_i(d_i) - \log(Z_P) \\ &= \sum_{C_j} Q(c_j) \sum_i \sum_{D_i \setminus C_j} Q(d_i|c_j) \log \Psi_i(d_i) - \log(Z_P) \end{aligned}$$

Hence Eq. 7 is rewritten as

$$D(Q \,\|\, P) = \sum_{C_j} Q(c_j) \log Q(c_j) - \sum_{C_j} Q(c_j) \Bigg[-\sum_k \sum_{C_k \setminus C_j} Q(c_k|c_j) \log Q(c_k|c_j) + \sum_i \sum_{D_i \setminus C_j} Q(d_i|c_j) \log \Psi_i(d_i) \Bigg] + \log(Z_P)$$

Denoting the bracketed term by $\gamma_j(c_j)$, and letting $\Gamma_j(c_j) = e^{\gamma_j(c_j)}$, we get

$$D(Q \,\|\, P) = \sum_{C_j} Q(c_j) \log \frac{Q(c_j)}{\Gamma_j(c_j)} + \log(Z_P). \qquad \diamond$$

Note that $\Gamma_j(c_j)$ in Eq. 6 does not depend on $Q(c_j)$ and is a function of $Q(x)$ only through the conditional distribution of $X \setminus C_j$ given C_j (via $Q(c_k|c_j)$). Eq. 6 states that the KL distance between $Q(x)$ and $P(x)$ is equal, up to an additive constant, to the KL distance between $Q(c_j)$ and an un-normalized potential $\Gamma_j(c_j)$. This interesting result generalizes a similar equation for a special case derived in (Jojic et al, 2004).

The next lemma provides a variant of a well known property of KL. Recall that for every two probability distributions $Q(x)$ and $P(x)$, the KL distance

$D(Q(x) \,\|\, P(x)) \geq 0$ and equality holds if and only if $Q(x) = P(x)$ (Cover and Thomas 1991; Theorem 2.6.3). A similar result holds also for un-normalized probability distributions.

Lemma 2 *Let $\tilde{Q}(x)$ and $\tilde{P}(x)$ be non-negative functions such that $\sum_x \tilde{P}(x) = Z_P > 0$, and let*

$$\hat{Q}(x) = \min_{\{\tilde{Q} | \sum_x \tilde{Q}(x) = Z_Q\}} D(\tilde{Q}(x) \,\|\, \tilde{P}(x))$$

where Z_Q is a positive constant. Then $\hat{Q}(x) = \frac{Z_Q}{Z_P} P(x)$.

Proof. We observe that

$$D(\tilde{Q}(x) \,\|\, \tilde{P}(x)) = Z_Q \cdot D(\tfrac{\tilde{Q}(x)}{Z_Q} \,\|\, \tfrac{\tilde{P}(x)}{Z_P}) + \log \frac{Z_Q}{Z_P}$$

which implies, using the cited result about normalized distributions, that the minimum is obtained when $\frac{\tilde{Q}(x)}{Z_Q} = \frac{\tilde{P}(x)}{Z_P}$, yielding the desired claim. ◇

Theorem 1 (Convergence of VIP$^+$) *Algorithm VIP$^+$ converges to a stationary point of the KL distance between Q and P among all distributions Q of the form $Q(x) = \frac{1}{Z_Q} \prod_j \Phi_j(c_j)$.*

Proof. We need to show that at the start of each iteration of VIP$^+$ the function Q defined by the revised potentials $\Phi_j(c_j)$ is closer to P in KL distance than Q at the start of the previous iteration. We rewrite the KL distance $D(Q \,\|\, P)$ using Eq. 6, as justified by Lemma 1. Using the given form of Q, we have

$$Q(c_j) = \frac{1}{Z_Q} \left[\sum_{X \setminus C_j} \prod_{k \neq j} \Phi_k(c_k) \right] \Phi_j(c_j). \tag{8}$$

We denote the bracketed coefficient of $\Phi_j(c_j)$ by B and note that it is constant in the sense that it does not depend on the quantity Φ_j being optimized. We now use Eq. 8 to rewrite Eq. 6 as

$$D(Q \,\|\, P) = \frac{B}{Z_Q} \left[\sum_{C_j} \Phi_j(c_j) \log \frac{\Phi_j(c_j)}{\Gamma_j(c_j)} \right] + \log \frac{B Z_P}{Z_Q}. \tag{9}$$

Recall that $\Gamma_j(c_j) = e^{\gamma_j(c_j)}$ does not depend on $\Phi_j(c_j)$ since it only depends on the conditional probability $Q(X|c_j)$. Hence, Lemma 2 states that the (global) minimum wrt Φ_j is achieved when $\Phi_j(c_j)$ is set to be proportional to $\Gamma_j(c_j)$. It is possible to set $\Phi_j(c_j)$ to be proportional to $\Gamma_j(c_j)$, as done in Step 1 of VIP$^+$, because $\Phi_j(c_j)$ is a full potential. The proportionality constant does not matter because if Φ_j is multiplied by α, and the arbitrary constraining constant Z_Q is also multiplied by α, these influences cancel in Eq. 9. For simplicity, in the algorithm, we use $\alpha = 1$ and therefore $\Phi_j(c_j) \leftarrow e^{\gamma_j(c_j)}$. Algorithm VIP$^+$ computes $\Phi_j(c_j)$ according to this formula and hence decreases the KL distance in each iteration by improving $\Phi_j(c_j)$ while holding all other cluster potentials fixed. Since the KL distance is lower bounded by zero, VIP$^+$ converges to a stationary point.

It remains to show that at the start of each iteration, the quantities $Q(c_k|c_j)$ and $Q(d_i|c_j)$ can be computed correctly from the junction tree for the un-normalized distribution $\tilde{Q}$ of the current normalized distribution Q. At each iteration, Q is computed up to some implicit normalizing factor, say α, so that $\tilde{Q}(x) = \alpha Q(x)$. The procedure DISTRIBUTEEVIDENCE(Φ_j) is based on the following update scheme. Starting with Φ_j, every neighboring cluster node C_k in the junction tree, representing the cluster potential $\Phi_k(c_k)$, is updated via

$$\Phi_k^{new}(c_k) \leftarrow \Phi_k(c_k) \frac{\Phi_k^{new}(s_{jk})}{\Phi_k(s_{jk})}$$

where s_{jk} is the instantiation for the separator $S_{jk} = C_j \cap C_k$ consistent with the instantiation c_k, and then the cluster neighbors of the neighboring clusters are updated similarly, until all clusters have been updated (Jensen, 1996). In each step, any normalizing constant implicitly appears 4 times in this update equation, and it cancels out regardless of its value. Hence, the quantities $Q(c_k|c_j)$ and $Q(d_i|c_j)$ are updated correctly from the un-normalized distribution $\tilde{Q}$. ◇

Corollary 1 (Convergence of VIP) *Algorithm VIP converges to a stationary point for the KL distance between Q and P among all distributions Q of the form $Q(x) = \frac{1}{Z_Q} \prod_j \Phi_j(c_j)$.*

Proof. VIP convergence follows from Theorem 1 because the update equations for the two algorithms, Equations 1 and 4, are identical; the two algorithms only differ in the method by which conditional probabilities are computed. ◇

Theorem 2 (Convergence of VIP$^\sharp$) *Algorithm VIP$^\sharp$ converges to a stationary point of the KL distance between Q and P among all distributions Q of the form $Q(x) = \frac{1}{Z_Q} \prod_{jl} \Phi_{jl}(c_{jl})$ where Q and P are compatible.*

Proof: We analyze Equation 4 in light of the assumptions made in VIP$^\sharp$. The first term in Equation 4 is constant with respect to c_j and, thus, does not effect the update and can be dropped. Next, the fact that the clusters C_j of Q are independent means that

$Q(D_i) = \prod_k Q(D_i^k)$ where $D_i^k = D_i \cap C_k$. This factorization implies that $Q(D_i|c_j) = \prod_{k \neq j} Q(D_i^k)$ which in turn implies that $F_i(c) = \sum_{D_i \setminus C_j} Q(d_i|c_j) \log \Psi_i(d_i)$ equals $\sum_{D_i^1} Q(d_i^1) \ldots \sum_{D_i^J} Q(d_i^J) \log \Psi_i(d_i)$. By the assumption of compatibility, each $F_i(c)$ is a function of c_{jl} (i.e., after summing out all variables in $D_i \setminus C_j$) and, thus, can be put into the potential $\Phi_{jl}(c_{jl})$. Every element in the second sum of Equation 4 is put into some potential and $\Gamma_j(c_j)$ from Theorem 1 is equal to $\prod_l e^{\gamma_{jl}(c_{jl})}$. Therefore, by updating, for all l, $\Phi_{jl}(c_{jl}) \propto \Gamma_{jl}(c_{jl})$ is equivalent to updating Φ_j in Theorem 1 and, thus, the algorithm converges.$\diamond$

Acknowledgments

We thank Nir Friedman, David Heckerman, and Nebojsa Jojic for various discussions on the subject of variational techniques.

References

Bishop, C. & Winn, J. (2003). Structured variational distributions in VIBES. In *Artificial Intelligence and Statistics*. Society for Artificial Intelligence and Statistics.

Cooper, G. (1990). Probabilistic inference using belief networks is NP-hard. *Artificial Intelligence*, *42*, 393–405.

Cover, T. M. & Thomas, J. A. (1991). *Elements of Information Theory*. Wiley.

Dagum, P. & Luby, M. (1993). Approximating probabilistic inference in Bayesian belief networks is NP-hard. *Artificial Intelligence*, *60*(1), 141–153.

Dechter, R. (1999). Bucket elimination: A unifying framework for reasoning. *Artificial Intelligence*, *113*(1-2), 41–85.

Ghahramani, Z. & Jordan, M. I. (1997). Factorial hidden Markov models. *Machine Learning*, *29*, 245–273.

Jensen, F. V. (1996). *An Introduction to Bayesian Networks*. UCL Press.

Jojic, V., Jojic, N., Meek, C., Geiger, D.,Siepel, A., Haussler, D., & Heckerman, D. (2004). Efficient approximations for learning phylogenetic HMM models from data. *Bioinformatics*, *20*, 161–168.

Kschischang, F. R., Frey, B. J., & Loeliger, H.-A. (2001). Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, *47*(2).

Saul, L. & Jordan, M. I. (1996). Exploiting tractable substructures in intractable networks. In *Advances in Neural Information Processing Systems (NIPS)*. MIT Press.

Wiegerinck, W. (2000). Variational approximations between mean field theory and the junction tree algorithm. In *Uncertainty in Artificial Intelligence*. Morgan Kaufmann.

Xing, E. P., Jordan, M. I., & Russell, S. (2003). A generalized mean field algorithm for variational inference in exponential families. In *Uncertainty in Artificial Intelligence*. Morgan Kaufmann.

Xing, E. P., Jordan, M. I., & Russell, S. (2004). Graph partition strategies for generalized mean field inference. In *Uncertainty in Artificial Intelligence*. Morgan Kaufmann.