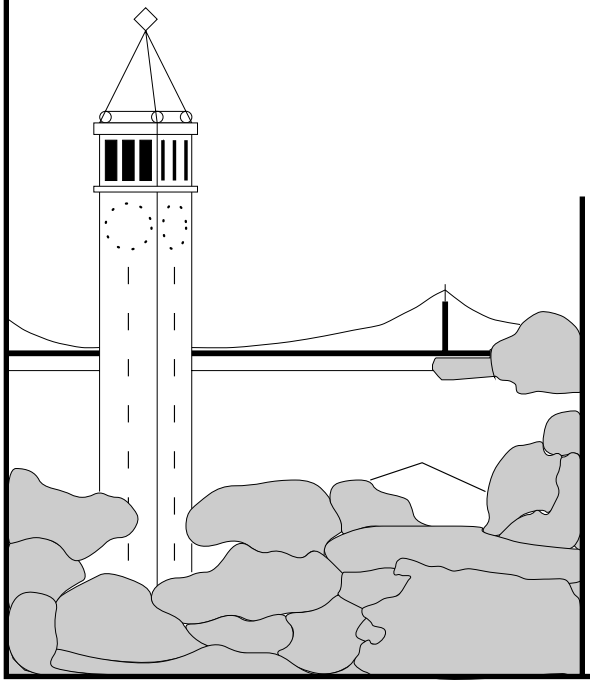
An Introduction to Variational Methods for Graphical Models

Michael I. Jordan
E25-229, MIT
Cambridge, MA 02139
[jordan@ai.mit.edu]

Zoubin Ghahramani
University of Toronto
Toronto, Ontario
[zoubin@cs.toronto.edu]

Tommi S. Jaakkola
University of California
Santa Cruz, CA
[tommi@cse.ucsc.edu]

Lawrence K. Saul
AT&T Labs – Research
Florham Park, NJ
[lsaul@research.att.com]



Report No. UCB/CSD-98-980

January 1998

Computer Science Division (EECS)
University of California
Berkeley, California 94720

An Introduction to Variational Methods for Graphical Models

Michael I. Jordan
E25-229, MIT
Cambridge, MA 02139
[jordan@ai.mit.edu]

Zoubin Ghahramani
University of Toronto
Toronto, Ontario
[zoubin@cs.toronto.edu]

Tommi S. Jaakkola
University of California
Santa Cruz, CA
[tommi@cse.ucsc.edu]

Lawrence K. Saul
AT&T Labs – Research
Florham Park, NJ
[lsaul@research.att.com]

January 1998

Abstract

This paper presents a tutorial introduction to the use of variational methods for inference and learning in graphical models. We present a number of examples of graphical models, including the QMR-DT database, the sigmoid belief network, the Boltzmann machine, and several variants of hidden Markov models, in which it is infeasible to run exact inference algorithms. We then introduce variational methods, showing how upper and lower bounds can be found for local probabilities, and discussing methods for extending these bounds to bounds on global probabilities of interest. Finally we return to the examples and demonstrate how variational algorithms can be formulated in each case.

To appear: M. I. Jordan, (Ed.), *Learning in Graphical Models*, Kluwer Academic Publishers.

1 Introduction

The problem of probabilistic inference in graphical models is the problem of computing a conditional probability distribution over the values of some of the nodes (the "hidden" or "unobserved" nodes), given the values of other nodes (the "evidence" or "observed" nodes). Thus, letting H represent the set of hidden nodes and letting E represent the set of evidence nodes, we wish to calculate $P(H|E)$:

$$P(H|E) = \frac{P(H,E)}{P(E)}. \tag{1}$$

General exact inference algorithms have been developed to perform this calculation (Cowell, this volume; Jensen, 1996); these algorithms take systematic advantage of the conditional independencies present in the joint distribution as inferred from the pattern of missing edges in the graph.

We often also wish to calculate marginal probabilities in graphical models, in particular the probability of the observed evidence, $P(E)$. Viewed as a function of the parameters of the graphical model, for fixed E, $P(E)$ is an important quantity known as the *likelihood*. As is suggested by Eq. (1), the evaluation of the likelihood is closely related to the calculation of $P(H|E)$. Indeed, although inference algorithms do not simply compute the numerator and denominator of Eq. (1) and divide, they in fact generally produce the likelihood as a by-product of the calculation of $P(H|E)$. Moreover, algorithms that maximize likelihood (and related quantities) generally make use of the calculation of $P(H|E)$ as a subroutine.

Although there are many cases in which the exact algorithms provide a satisfactory solution to inference and learning problems, there are other cases, several of which we discuss in this paper, in which the time or space complexity of the exact calculation is unacceptable and it is necessary to have recourse to approximation procedures. Within the context of the junction tree construction, for example, the time complexity is exponential in the size of the maximal clique in the junction tree. As we will see, there are natural architectural assumptions that necessarily lead to large cliques.

Even in cases in which the complexity of the exact algorithms is manageable, there can be reason to consider approximation procedures. Note in particular that the exact algorithms make no use of the numerical representation of the joint probability distribution associated with a graphical model; put another way, the algorithms have the same complexity regardless of the particular probability distribution under consideration within the family of distributions that is consistent with the conditional independencies implied by the graph. There may be situations in which nodes or clusters of nodes are "nearly" conditionally independent, situations in which node probabilities are well determined by a subset of the neighbors of the node, or situations in which small subsets of configurations of variables contain most of the probability mass. In such cases the exactitude achieved by an exact algorithm may not be worth the computational cost. A variety of approximation procedures have been developed that attempt to identify and exploit such situations. Examples include the pruning algorithms of Kjærulff (1994), the "bounded conditioning" method of Horvitz, Suermondt, and Cooper (1989), search-based methods (e.g., Henrion, 1991), and the "localized partial evaluation" method of Draper and Hanks (1994). A virtue of all of

these methods is that they are closely tied to the exact methods and thus are able to take full advantage of conditional independencies. This virtue can also be a vice, however, given the exponential growth in complexity of the exact algorithms.

A related approach to approximate inference has arisen in in applications of graphical model inference to error-control decoding (McEliece, MacKay, & Cheng, 1996). In particular, Kim and Pearl's algorithm for singly-connected graphical models (Pearl, 1988) has been used successfully as an iterative approximate method for inference in non-singly-connected graphs.

Another approach to the design of approximation algorithms involves making use of Monte Carlo methods. A variety of Monte Carlo algorithms have been developed (see MacKay, this volume, and Neal, 1993) and applied to the inference problem in graphical models (Dagum & Luby, 1993; Fung & Favero, 1994; Gilks, Thomas, & Spiegelhalter, 1994; Jensen, Kong, & Kjærulff, 1995; Pearl, 1988). Advantages of these algorithms include their simplicity of implementation and theoretical guarantees of convergence. The disadvantages of the Monte Carlo approach are that the algorithms can be slow to converge and it can be hard to diagnose their convergence.

In this chapter we discuss variational methods, which provide yet another approach to the design of approximate inference algorithms. Variational methodology yields deterministic approximation procedures that generally provide bounds on probabilities of interest. The basic intuition underlying variational methods is that complex graphs can be probabilistically simple; in particular, in graphs with dense connectivity there are averaging phenomena that can come into play, rendering nodes relatively insensitive to particular settings of values of their neighbors. Taking advantage of these averaging phenomena can lead to simple, accurate approximation procedures.

It is important to emphasize that the various approaches to inference that we have outlined are by no means mutually exclusive; indeed they exploit complementary features of the graphical model formalism. The best solution to any given problem may well involve an algorithm that combines aspects of the different methods. In this vein, we will present variational methods in a way that emphasizes their links to exact methods. Indeed, as we will see, exact methods often appear as subroutines within an overall variational approximation (cf. Jaakkola & Jordan, 1996; Saul & Jordan, 1996).

It should be acknowledged at the outset that there is as much "art" as there is "science" in our current understanding of how variational methods can be applied to probabilistic inference. Variational transformations form a large, open-ended class of approximations, and although there is a general mathematical picture of how these transformations can be exploited to yield bounds on probabilities in graphical models, there is not as yet a systematic algebra that allows particular variational transformations to be matched optimally to particular graphical models. We will provide illustrative examples of general families of graphical models to which variational methods have been applied successfully, and we will provide a general mathematical framework which encompasses all of these particular examples, but we are not as yet able to provide assurance that the framework will transfer easily to other examples.

We begin in Section 2 with a brief overview of exact inference in graphical models, bas-

ing the discussion on the junction tree algorithm. Section 3 presents several examples of graphical models, both to provide motivation for variational methodology and to provide examples that we return to and develop in detail as we proceed through the chapter. The core material on variational approximation is presented in Section 4. Sections 5 and 6 fill in some of the details, focusing on sequential methods and block methods, respectively. In these latter two sections, we also return to the examples and work out variational approximations in each case. Finally, Section 7 presents conclusions and directions for future research.

2 Exact inference

In this section we provide a brief overview of exact inference for graphical models, as represented by the junction tree algorithm (for relationships between the junction tree algorithm and other exact inference algorithms, see Shachter, Andersen, and Szolovits, 1994; see also Dechter, this volume, and Shenoy, 1992, for recent developments in exact inference). Our intention here is not to provide a complete description of the junction tree algorithm, but rather to introduce the "moralization" and "triangulation" steps of the algorithm. An understanding of these steps, which create data structures that determine the run time of the inference algorithm, will suffice for our purposes.[1] For a comprehensive introduction to the junction tree algorithm see Cowell (this volume) and Jensen (1996).

Graphical models come in two basic flavors—*directed* graphical models and *undirected* graphical models. A directed graphical model is specified numerically by associating local conditional probabilities with each of the nodes in an acyclic directed graph. These conditional probabilities specify the probability of node S_i given the values of its parents, i.e., $P(S_i|S_{\pi(i)})$, where $\pi(i)$ represents the set of indices of the parents of node S_i and $S_{\pi(i)}$ represents the corresponding set of parent nodes (see Fig. 1).[2] To obtain the joint probability distribution for all of the N nodes in the graph, i.e., $P(S) = P(S_1, S_2, \ldots, S_N)$, we take the product over the local node probabilities:

$$P(S) = \prod_{i=1}^{N} P(S_i|S_{\pi(i)}) \tag{2}$$

Inference involves the calculation of conditional probabilities under this joint distribution.

An undirected graphical model (also known as a "Markov random field") is specified numerically by associating "potentials" with the cliques of the graph.[3] A potential is a function on the set of configurations of a clique (that is, a setting of values for all of the nodes in the clique) that associates a positive real number with each configuration. Thus, for every subset of nodes C_i that forms a clique, we have an associated potential $\phi_i(C_i)$ (see Fig. 2). The joint probability distribution for all of the nodes in the graph is obtained by taking the product over the clique potentials:

$$P(S) = \frac{\prod_{i=1}^{M} \phi_i(C_i)}{Z}, \tag{3}$$

[1] Our presentation will take the point of view that moralization and triangulation, when combined with a local message-passing algorithm, are *sufficient* for exact inference. It is also possible to show that, under

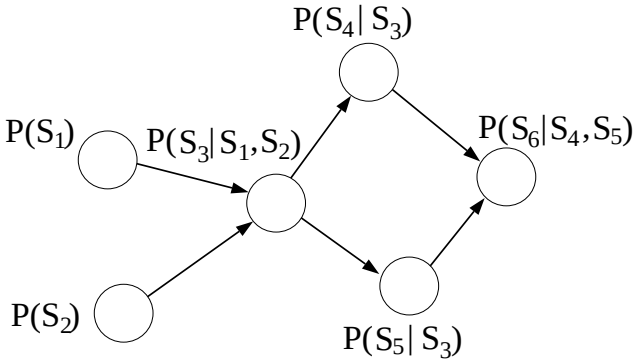


Figure 1: A directed graph is parameterized by associating a local conditional probability with each node. The joint probability is the product of the local probabilities.

where M is the total number of cliques and where the normalization factor Z is obtained by summing the numerator over all configurations:

$$Z = \sum_{\{S\}} \left\{ \prod_{i=1}^{M} \phi_i(C_i) \right\}. \tag{4}$$

In keeping with statistical mechanical terminology we will refer to this sum as a "partition function."

The junction tree algorithm compiles directed graphical models into undirected graphical models; subsequent inferential calculation is carried out in the undirected formalism. The step that converts the directed graph into an undirected graph is called "moralization." (If the initial graph is already undirected, then we simply skip the moralization step). To understand moralization, we note that in both the directed and the undirected cases, the joint probability distribution is obtained as a product of local functions. In the directed case, these functions are the node conditional probabilities $P(S_i|S_{\pi(i)})$. In fact, this probability nearly qualifies as a potential function; it is certainly a real-valued function on the configurations of the set of variables $\{S_i, S_{\pi(i)}\}$. The problem is that these variables do not always appear together within a clique. That is, the parents of a common child are not necessarily linked. To be able to utilize node conditional probabilities as potential functions, we "marry" the parents of all of the nodes with undirected edges. Moreover we drop the arrows on the other edges in the graph. The result is a "moral graph," which can be used to represent the probability distribution on the original directed graph within the undirected formalism.[4]

The second phase of the junction tree algorithm is somewhat more complex. This phase, known as "triangulation," takes a moral graph as input and produces as output

certain conditions, these steps are *necessary* for exact inference. See Jensen and Jensen (1994).

[2]Here and elsewhere we identify the ith node with the random variable S_i associated with the node.

[3]We define a clique to be a subset of nodes which are fully connected and maximal; i.e., no additional node can be added to the subset so that the subset remains fully connected.

[4]Note in particular that Fig. 2 is the moralization of Fig. 1.

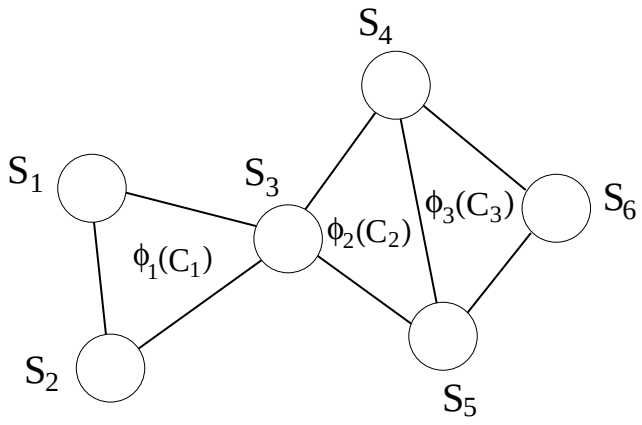


Figure 2: An undirected graph is parameterized by associating a potential with each clique in the graph. The cliques in this example are $C_1 = \{S_1, S_2, S_3\}$, $C_2 = \{S_3, S_4, S_5\}$, and $C_3 = \{S_4, S_5, S_6\}$. A potential assigns a positive real number to each configuration of the corresponding clique. The joint probability is the normalized product of the clique potentials.

an undirected graph in which additional edges have (possibly) been added. This latter graph has a special property that allows recursive calculation of probabilities to take place. In particular, in a triangulated graph, it is possible to build up a joint distribution by proceeding sequentially through the graph, conditioning blocks of interconnected nodes only on predecessor blocks in the sequence. The simplest graph in which this is *not* possible is the "4-cycle," the cycle of four nodes shown in Fig. 3(a). If we try to write the joint probability sequentially as, for example, $P(A)P(B|A)P(C|B)P(D|C)$, we see that we have a problem. In particular, A depends on D, and we are unable to write the joint probability as a sequence of conditionals.

A graph is *not triangulated* if there are 4-cycles which do not have a *chord*, where a chord

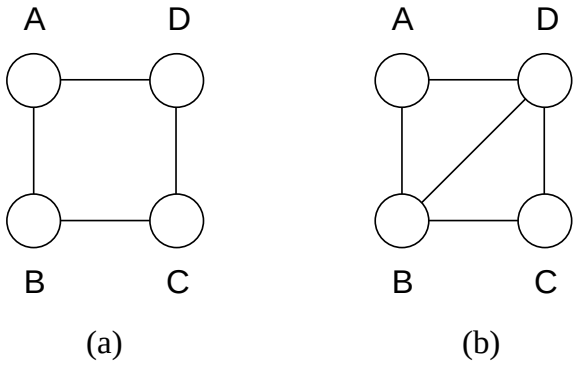


Figure 3: (a) The simplest non-triangulated graph. The graph has a 4-cycle without a chord. (b) Adding a chord between nodes B and D renders the graph triangulated.

is an edge between non-neighboring nodes. Thus the graph in Fig. 3(a) is not triangulated; it can be triangulated by adding a chord as in Fig. 3(b). In the latter graph we can write the joint probability sequentially as $P(A, B, C, D) = P(A)P(B, D|A)P(C|B, D)$.

More generally, once a graph has been triangulated it is possible to arrange the cliques of the graph into a data structure known as a *junction tree*. A junction tree has the *running intersection property*: If a node appears in any two cliques in the tree, it appears in all cliques that lie on the path between the two cliques. This property has the important consequence that a general algorithm for probabilistic inference can be based on achieving local consistency between cliques. (That is, the cliques assign the same marginal probability to the nodes that they have in common). In a junction tree, because of the running intersection property, local consistency implies global consistency.

The probabilistic calculations that are performed on the junction tree involve marginalizing and rescaling the clique potentials so as to achieve local consistency between neighboring cliques. The time complexity of performing this calculation depends on the size of the cliques; in particular for discrete data the number of values required to represent the potential is exponential in the number of nodes in the clique. For efficient inference, it is therefore critical to obtain small cliques.

In the remainder of this paper, we will investigate specific graphical models and consider the computational costs of exact inference for these models. In all of these cases we will either be able to display the "obvious" triangulation, or we will be able to lower bound the size of cliques in a triangulated graph by considering the cliques in the moral graph. Thus we will not need to consider specific algorithms for triangulation (for discussion of triangulation algorithms, see, e.g., Kjærulff, 1990).

3 Examples

In this section we present examples of graphical models in which exact inference is generally infeasible. Our first example involves a diagnostic system in which a fixed graphical model is used to answer queries. The remaining examples involve estimation problems in which a graphical model is fit to data and subsequently used for prediction or diagnosis.

3.1 The QMR-DT database

The QMR-DT database is a large-scale probabilistic database that is intended to be used as a diagnostic aid in the domain of internal medicine.[5] We provide a brief overview of the QMR-DT database here; for further details see Shwe, et al. (1991).

The QMR-DT database is a bipartite graphical model in which the upper layer of nodes represent diseases and the lower layer of nodes represent symptoms (see Fig. 4). There are approximately 600 disease nodes and 4000 symptom nodes in the database.

The evidence is a set of observed symptoms; henceforth we refer to observed symptoms as "findings" and represent the vector of findings with the symbol f. The symbol d denotes the vector of diseases. All nodes are binary, thus the components f_i and d_i are binary

[5]The acronym "QMR-DT" refers to the "Decision Theoretic" version of the "Quick Medical Reference."

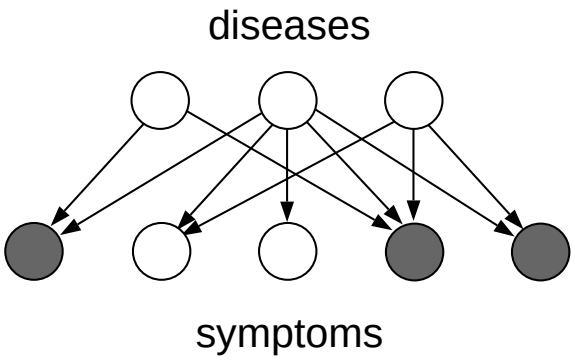


Figure 4: The structure of the QMR-DT graphical model. The shaded nodes represent evidence nodes and are referred to as "findings."

random variables. Making use of the conditional independencies implied by the bipartite form of the graph,[6] and marginalizing over the unobserved symptom nodes, we obtain the following joint probability over diseases and findings:

$$P(f,d) = P(f|d)P(d) \tag{5}$$

$$= \left[\prod_i P(f_i|d)\right]\left[\prod_j P(d_j)\right]. \tag{6}$$

The prior probabilities of the diseases, $P(d_j)$, were obtained by Shwe, et al. from archival data. The conditional probabilities of the findings given the diseases, $P(f_i|d)$, were obtained from expert assessments under a "noisy-OR" model. That is, the conditional probability that the ith symptom is absent, $P(f_i = 0|d)$, is expressed as follows:

$$P(f_i = 0|d) = (1 - q_{i0}) \prod_{j \in \pi(i)} (1 - q_{ij})^{d_j} \tag{7}$$

where the q_{ij} are parameters obtained from the expert assessments. Considering cases in which only one disease is present, that is, $\{d_j = 1\}$ and $\{d_k = 0; k \neq j\}$, we see that q_{ij} can be interpreted as the probability that the ith finding is present if only the jth disease is present. Considering the case in which all diseases are absent, we see that the q_{i0} parameter can be interpreted as the probability that the ith finding is present even though no disease is present.

We will find it useful to rewrite the noisy-OR model in an exponential form:

$$P(f_i = 0|d) = e^{-\sum_{j \in \pi(i)} \theta_{ij} d_j - \theta_{i0}} \tag{8}$$

where $\theta_{ij} \equiv -\ln(1 - q_{ij})$ are the transformed parameters. Note also that the probability of a positive finding is given as follows:

$$P(f_i = 1|d) = 1 - e^{-\sum_{j \in \pi(i)} \theta_{ij} d_j - \theta_{i0}} \tag{9}$$

[6] In particular, the pattern of missing edges in the graph implies that (a) the diseases are marginally independent, and (b) given the diseases, the symptoms are conditionally independent.

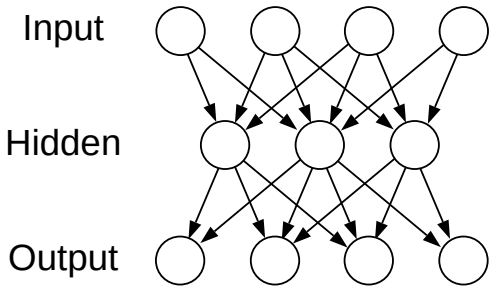


Figure 5: The layered graphical structure of a neural network. The input nodes and output nodes comprise the set of evidence nodes.

These forms express the noisy-OR model as a generalized linear model.

If we now form the joint probability distribution by taking products of the local probabilities $P(f_i|d)$ as in Eq. (6), we see that negative findings are benign with respect to the inference problem. In particular, a product of exponential factors that are linear in the diseases (cf. Eq. (8)) yields a joint probability that is also the exponential of an expression linear in the diseases. That is, each negative finding can be incorporated into the joint probability in a linear number of operations.

Products of the probabilities of positive findings, on the other hand, yield cross products terms that are problematic for exact inference. These cross product terms couple the diseases (they are responsible for the "explaining away" phenomena that arise for the noisy-OR model; see Pearl, 1988). Unfortunately, these coupling terms can lead to an exponential growth in inferential complexity. Considering a set of standard diagnostic cases (the "CPC cases"; see Shwe, et al. 1991), Jaakkola and Jordan (1997c) found that the median size of the maximal clique of the moralized QMR-DT graph is 151.5 nodes. Thus even without considering the triangulation step, we see that diagnostic calculation under the QMR-DT model is generally infeasible.[7]

3.2 Neural networks as graphical models

Neural networks are layered graphs endowed with a nonlinear "activation" function at each node (see Fig. 5). Let us consider activation functions that are bounded between zero and one, such as those obtained from the logistic function $f(z) = 1/(1+e^{-z})$. We can treat such a neural network as a graphical model by associating a binary variable S_i with each node and interpreting the activation of the node as the probability that the associated binary variable takes one of its two values. For example, using the logistic function, we write:

$$P(S_i = 1|S_{\pi(i)}) = \frac{1}{1 + e^{-\sum_{j \in \pi(i)} \theta_{ij} S_j - \theta_{i0}}} \tag{10}$$

where θ_{ij} are the parameters associated with edges between parent nodes j and node i, and θ_{i0} is the "bias" parameter associated with node i. This is the "sigmoid belief network"

[7] Jaakkola and Jordan (1997c) also calculated the median of the pairwise cutset size. This value was found to be 106.5, which also rules out exact cutset methods for inference for the QMR-DT.

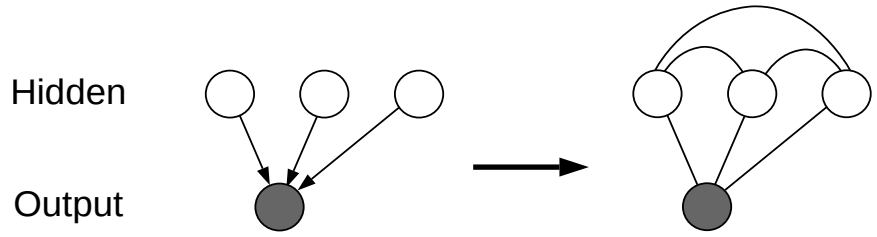


Figure 6: Moralization of a neural network. The output nodes are evidence nodes during training. This creates probabilistic dependencies between the hidden nodes which are captured by the edges added by the moralization.

introduced by Neal (1992). The advantages of treating a neural network in this manner include the ability to perform diagnostic calculations, to handle missing data, and to treat unsupervised learning on the same footing as supervised learning. Realizing these benefits, however, requires that the inference problem be solved in an efficient way.

In fact, it is easy to see that exact inference is infeasible in general layered neural network models. A node in a neural network generally has as parents all of the nodes in the preceding layer. Thus the moralized neural network graph has links between all of the nodes in this layer (see Fig. 6). That these links are necessary for exact inference in general is clear—in particular, during training of a neural network the output nodes are evidence nodes, thus the hidden units in the penultimate layer become probabilistically dependent, as do their ancestors in the preceding hidden layers.

Thus if there are N hidden units in a particular hidden layer, the time complexity of inference is at least $O(2^N)$, ignoring the additional growth in clique size due to triangulation. Given that neural networks with dozens or even hundreds of hidden units are commonplace, we see that training a neural network using exact inference is not generally feasible.

3.3 Boltzmann machines

A Boltzmann machine is an undirected graphical model with binary-valued nodes and a restricted set of potential functions (see Fig. 7). In particular, the clique potentials are formed by taking products of "Boltzmann factors"—exponentials of terms that are at most quadratic in the S_i (Hinton & Sejnowski, 1986). Thus each clique potential is a product of factors $\exp\{\theta_{ij} S_i S_j\}$ and factors $\exp\{\theta_{i0} S_i\}$, where $S_i \in \{0, 1\}$.[8]

A given pair of nodes S_i and S_j can appear in multiple, overlapping cliques. For each such pair we assume that the expression $\exp\{\theta_{ij} S_i S_j\}$ appears as a factor in one and only one clique potential. Similarly, the factors $\exp\{\theta_{i0} S_i\}$ are assumed to appear in one and only one clique potential. Taking the product over all such clique potentials (cf. Eq. (3)),

[8]It is also possible to consider more general Boltzmann machines with multivalued nodes, and potentials that are exponentials of arbitrary functions on the cliques. Such models are essentially equivalent to the general undirected graphical model of Eq. (3) (although the latter can represent zero probabilities while the former cannot).

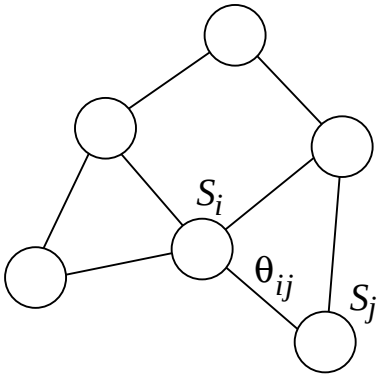


Figure 7: A Boltzmann machine. An edge between nodes S_i and S_j is associated with a factor $\exp(\theta_{ij} S_i S_j)$ that contributes multiplicatively to the potential of one of the cliques containing the edge. Each node also contributes a factor $\exp(\theta_{i0} S_i)$ to one and only one potential.

we have:

$$P(S) = \frac{e^{\sum_{i<j} \theta_{ij} S_i S_j + \sum_i \theta_{i0} S_i}}{Z}, \tag{11}$$

where we have set $\theta_{ij} = 0$ for nodes S_i and S_j that are not neighbors in the graph—this convention allows us to sum indiscriminately over all pairs S_i and S_j and still respect the clique boundaries. We refer to the negative of the exponent in Eq. (11) as the *energy*. With this definition the joint probability in Eq. (11) has the general form of a *Boltzmann distribution*.

Saul and Jordan (1994) pointed out that exact inference for certain special cases of Boltzmann machine—such as trees, chains, and pairs of coupled chains—is tractable and they proposed a *decimation* algorithm for this purpose. For more general Boltzmann machines, however, decimation is not immune to the exponential time complexity that plagues other exact methods. Indeed, despite the fact that the Boltzmann machine is a special class of undirected graphical model, it is a special class only by virtue of its parameterization, not by virtue of its conditional independence structure. Thus, exact algorithms such as decimation and the junction tree algorithm, which are based solely on the graphical structure of the Boltzmann machine, are no more efficient for Boltzmann machines than they are for general graphical models. In particular, when we triangulate generic Boltzmann machines, including the layered Boltzmann machines and grid-like Boltzmann machines, we obtain intractably large cliques.

Sampling algorithms have traditionally been used to attempt to cope with the intractability of the Boltzmann machine (Hinton & Sejnowski, 1986). The sampling algorithms are overly slow, however, and more recent work has considered the faster "mean field" approximation (Peterson & Anderson, 1987). We will describe the mean field approximation for Boltzmann machines later in the paper—it is a special form of the variational approximation approach that provides lower bounds on marginal probabilities. We will

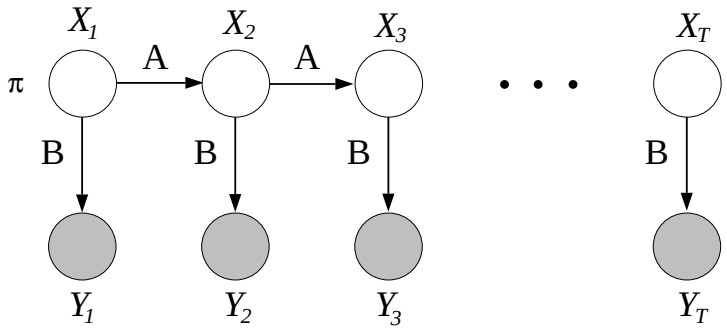


Figure 8: A HMM represented as a graphical model. The left-to-right spatial dimension represents time. The output nodes Y_i are evidence nodes during the training process and the state nodes X_i are hidden.

also discuss a more general variational algorithm that provides upper and lower bounds on probabilities (marginals and conditionals) for Boltzmann machines (Jaakkola & Jordan, 1997a).

3.4 Hidden Markov models

In this section, we briefly review hidden Markov models. The hidden Markov model (HMM) is an example of a graphical model in which exact inference is tractable; our purpose in discussing HMMs here is to lay the groundwork for the discussion of intractable variations on HMMs in the following sections. See Smyth, Heckerman, and Jordan (1997) for a fuller discussion of the HMM as a graphical model.

An HMM is a graphical model in the form of a chain (see Fig. 8). Consider a sequence of multinomial "state" nodes X_i and assume that the conditional probability of node X_i, given its immediate predecessor X_{i-1}, is independent of all other preceding variables. (The index i can be thought of as a time index). The chain is assumed to be homogeneous; that is, the matrix of transition probabilities, $A = P(X_i|X_{i-1})$, is invariant across time. We also require a probability distribution $\pi = P(X_1)$ for the initial state X_1.

The HMM model also involves a set of "output" nodes Y_i and an emission probability law $B = P(Y_i|X_i)$, again assumed time-invariant.

An HMM is trained by treating the output nodes as evidence nodes and the state nodes as hidden nodes. An expectation-maximization (EM) algorithm (Baum, et al., 1970; Dempster, Laird, & Rubin, 1977) is generally used to update the parameters A, B, π; this algorithm involves a simple iterative procedure having two alternating steps: (1) run an inference algorithm to calculate the conditional probabilities $P(X_i|\{Y_i\})$ and $P(X_i, X_{i-1}|\{Y_i\})$; (2) update the parameters via weighted maximum likelihood where the weights are given by the conditional probabilities calculated in step (1).

It is easy to see that exact inference is tractable for HMMs. The moralization and triangulation steps are vacuous for the HMM; thus the time complexity can be read off from Fig. 8 directly. We see that the maximal clique is of size N^2, where N is the dimensionality

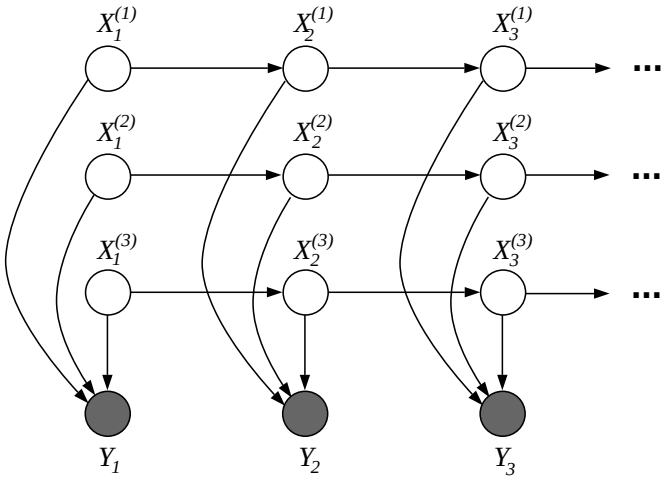


Figure 9: A factorial HMM with three chains. The transition matrices are $A^{(1)}$, $A^{(2)}$, and $A^{(3)}$ associated with the horizontal edges, and the output probabilities are determined by matrices $B^{(1)}$, $B^{(2)}$, and $B^{(3)}$ associated with the vertical edges.

of a state node. Inference therefore scales as $O(N^2T)$, where T is the length of the time series.

3.5 Factorial hidden Markov models

In many problem domains it is natural to make additional structural assumptions about the state space and the transition probabilities that are not available within the simple HMM framework. A number of structured variations on HMMs have been considered in recent years (see Smyth, et al., 1997); generically these variations can be viewed as "dynamic belief networks" (Dean & Kanazawa, 1989; Kanazawa, Koller, & Russell, 1995). Here we consider a particular simple variation on the HMM theme known as the "factorial hidden Markov model" (Ghahramani & Jordan, 1997; Williams & Hinton, 1991).

The graphical model for a factorial HMM (FHMM) is shown in Fig. 9. The system is composed of a set of M chains indexed by m. Let the state node for the mth chain at time i be represented by $X_i^{(m)}$ and let the transition matrix for the mth chain be represented by $A^{(m)}$. We can view the effective state space for the FHMM as the Cartesian product of the state spaces associated with the individual chains. The overall transition probability for the system by taking the product across the intra-chain transition probabilities:

$$P(X_i|X_{i-1}) = \prod_{m=1}^{M} A^{(m)}(X_i^{(m)}|X_{i-1}^{(m)}), \tag{12}$$

where the symbol X_i stands for the M-tuple $(X_i^{(1)}, X_i^{(2)}, \ldots, X_i^{(M)})$.

Ghahramani and Jordan utilized a linear-Gaussian distribution for the emission proba-

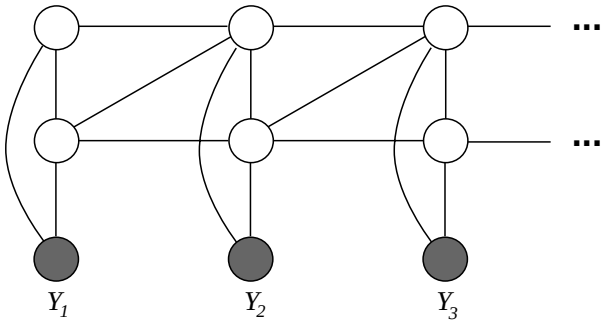


Figure 10: A triangulation of an FHMM with two component chains. The moralization step links states at a single time step. The triangulation step links states diagonally between neighboring time steps.

bilities of the FHMM. In particular, they assumed:

$$P(Y_i|X_i) = \mathcal{N}(\sum_m B^{(m)} X_i^{(m)}, \Sigma), \tag{13}$$

where the $B^{(m)}$ and Σ are matrices of parameters.

The FHMM is a natural model for systems in which the hidden state is realized via the joint configuration of an uncoupled set of dynamical systems. Moreover, an FHMM is able to represent a large effective state space with a much smaller number of parameters than a single unstructured Cartesian product HMM. For example, if we have 5 chains and in each chain the nodes have 10 states, the effective state space is of size 100,000, while the transition probabilities are represented compactly with only 500 parameters. A single unstructured HMM would require 10^{10} parameters for the transition matrix in this case.

The fact that the output is a function of the states of all of the chains implies that the states become stochastically coupled when the outputs are observed. Let us investigate the implications of this fact for the time complexity of exact inference in the FHMM. Fig. 10 shows a triangulation for the case of two chains (in fact this is an optimal triangulation). The cliques for the hidden states are of size N^3; thus the time complexity of exact inference is $O(N^3T)$, where N is the number of states in each chain (we assume that each chain has the same number of states for simplicity). Fig. 11 shows the case of a triangulation of three chains; here the triangulation (again optimal) creates cliques of size N^4. (Note in particular that the graph in Fig. 12, with cliques of size three, is *not* a triangulation; there are 4-cycles without a chord). In the general case, it is not difficult to see that cliques of size N^{M+1} are created, where M is the number of chains; thus the complexity of exact inference for the FHMM scales as $O(N^{M+1}T)$. For a single unstructured Cartesian product HMM having the same number of states as the FHMM—i.e., N^M states—the complexity scales as $O(N^{2M}T)$, thus exact inference for the FHMM is somewhat less costly, but the exponential growth in complexity in either case shows that exact inference is infeasible for general FHMMs.

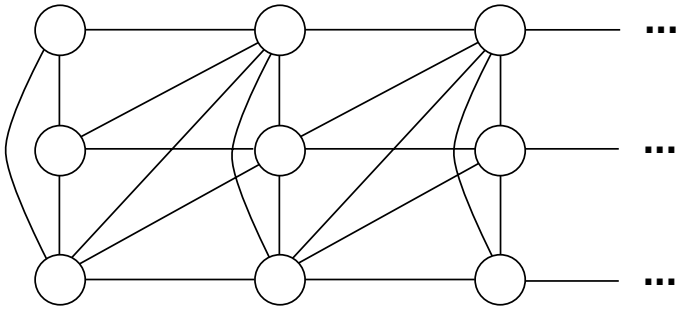


Figure 11: A triangulation of the state nodes of a three-chain FHMM with three component chains. (The observation nodes have been omitted in the interest of simplicity).

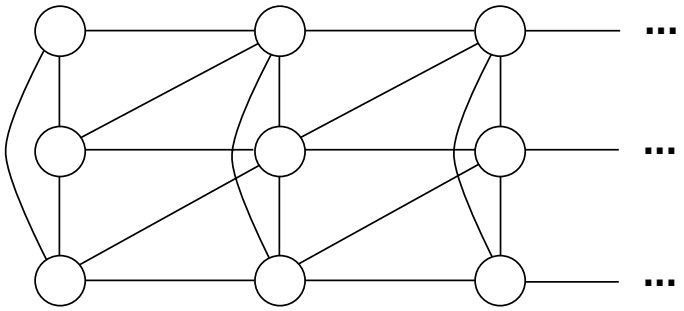


Figure 12: This graph is not a triangulation of a three-chain FHMM.

3.6 Higher–order Hidden Markov models

A related variation on HMMs considers a higher-order Markov model in which each state depends on the previous K states instead of the single previous state. In this case it is again readily shown that the time complexity is exponential in K. We will not discuss the higher–order HMM further in this chapter; for a variational algorithm for the higher–order HMM see Saul and Jordan (1996).

3.7 Hidden Markov decision trees

Finally, we consider a model in which a decision tree is endowed with Markovian dynamics (Jordan, et al., 1997). A decision tree can be viewed as a graphical model by modeling the decisions in the tree as multinomial random variables, one for each level of the decision tree. Referring to Fig. 13, and focusing on a particular time slice, the shaded node at the top of the diagram represents the input vector. The unshaded nodes below the input nodes are the decision nodes. Each of the decision nodes are conditioned on the input and on the entire sequence of preceding decisions (the vertical arrows in the diagram). In terms of a traditional decision tree diagram, this dependence provides an indication of the path followed by the data point as it drops through the decision tree. The node at the bottom of the diagram is the output variable.

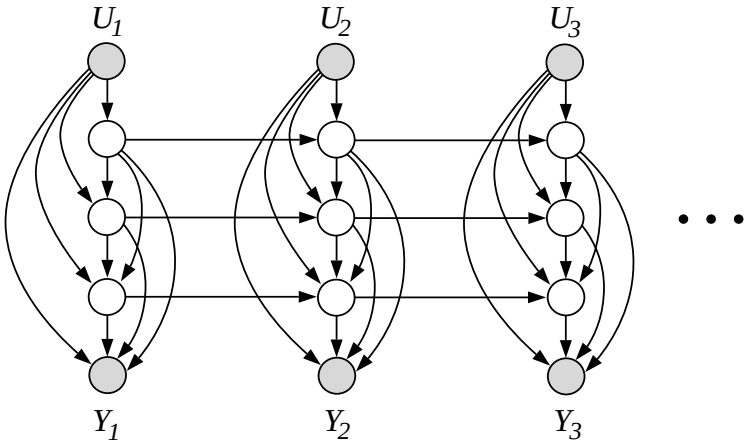


Figure 13: A hidden Markov decision tree. The shaded nodes $\{U_i\}$ and $\{Y_i\}$ represent a time series in which each element is an (input, output) pair. Linking the inputs and outputs are a sequence of decision nodes which correspond to branches in a decision tree. These decisions are linked horizontally to represent Markovian temporal dependence.

If we now make the decisions in the decision tree conditional not only on the current data point, but also on the decisions at the previous moment in time, we obtain a hidden Markov decision tree (HMDT). In Fig. 13, the horizontal edges represent this Markovian temporal dependence. Note in particular that the dependency is assumed to be level-specific—the probability of a decision depends only on the previous decision at the same level of the decision tree.

Given a sequence of input vectors U_i and a corresponding sequence of output vectors Y_i, the inference problem is to compute the conditional probability distribution over the hidden states. This problem is intractable for general HMDTs—as can be seen by noting that the HMDT includes the FHMM as a special case.

4 Basics of variational methodology

Variational methods are used as approximation methods in a wide variety of settings, include finite element analysis (Bathe, 1996), quantum mechanics (Sakurai, 1985), statistical mechanics (Parisi, 1988), and statistics (Rustagi, 1976). In each of these cases the application of variational methods converts a complex problem into a simpler problem, where the simpler problem is generally characterized by a decoupling of the degrees of freedom in the original problem. This decoupling is achieved via an expansion of the problem to include additional parameters, known as variational parameters, that must be fit to the problem at hand.

The terminology comes from the roots of the techniques in the calculus of variations. We will not start systematically from the calculus of variations; instead, we will jump off from an intermediate point that emphasizes the important role of convexity in variational ap-

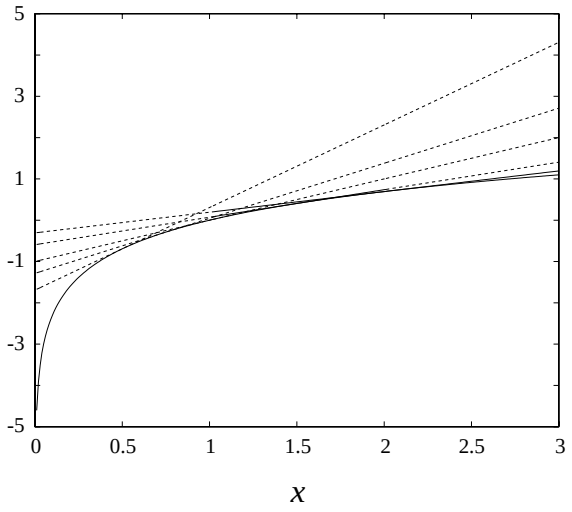


Figure 14: Variational transformation of the logarithm function. The linear functions $(\lambda x - \ln \lambda - 1)$ form a family of upper bounds for the logarithm, each of which is exact for a particular value of x.

proximation. This point of view turns out to be particularly well suited to the development of variational methods for graphical models.

4.1 Examples

Let us begin by considering a simple example. In particular, let us express the logarithm function variationally:

$$\ln(x) = \min_{\lambda} \{\lambda x - \ln \lambda - 1\}. \tag{14}$$

In this expression λ is the variational parameter, and we are required to perform the minimization for each value of x. The expression is readily verified by taking the derivative with respect to λ, solving and substituting. The situation is perhaps best appreciated geometrically, as we show in Fig. 14. Note that the expression in braces in Eq. (14) is linear in x with slope λ. Clearly, given the concavity of the logarithm, for each line having slope λ there is a value of the intercept such that the line touches the logarithm at a single point. Indeed, $-\ln \lambda - 1$ in Eq. (14) is precisely this intercept. Moreover, if we range across λ, the family of such lines forms an upper envelope of the logarithm function. That is, for any given x, we have:

$$\ln(x) \leq \lambda x - \ln \lambda - 1, \tag{15}$$

for all λ. Thus the variational transformation provides a family of upper bounds on the logarithm. The minimum over these bounds is the exact value of the logarithm.

The pragmatic justification for such a transformation is that we have converted a nonlinear function into a linear function. The cost is that we have obtained a free parameter

λ that must be set, once for each x. For any value of λ we obtain an upper bound on the logarithm; if we set λ well we can obtain a good bound. Indeed we can recover the exact value of logarithm for the optimal choice of λ.

Let us now consider a second example that is more directly relevant to graphical models. For binary-valued nodes it is common to represent the probability that the node takes one of its values via a monotonic nonlinearity that is a simple function—e.g., a linear function—of the values of the parents of the node. An example is the logistic regression model:

$$f(x) = \frac{1}{1 + e^{-x}}, \tag{16}$$

which we have seen previously in Eq. (10). Here x is the weighted sum of the values of the parents of a node.

The logistic function is neither convex nor concave, so a simple linear bound will not work. However, the logistic function is *log concave*. That is, the function

$$g(x) = -\ln(1 + e^{-x}) \tag{17}$$

is a concave function of x (as can readily be verified by calculating the second derivative). Thus we can bound the log logistic function with linear functions and thereby bound the logistic function by the exponential. In particular, we can write:

$$g(x) = \min_{\lambda} \{\lambda x - H(\lambda)\}, \tag{18}$$

where $H(\lambda)$ is the binary entropy function, $H(\lambda) = -\lambda \ln \lambda - (1 - \lambda) \ln(1 - \lambda)$. (We will explain how the binary entropy function arises below; for now it suffices to think of it simply as the appropriate intercept term for the log logistic function). We now take the exponential of both sides, noting that the minimum and the exponential function commute:

$$f(x) = \min_{\lambda} \left[e^{\lambda x - H(\lambda)} \right]. \tag{19}$$

This is a variational transformation for the logistic function; examples are plotted in Fig. 15. Finally, we note once again that for any value of λ we obtain an upper bound of the logistic function for all values of x:

$$f(x) \leq e^{\lambda x - H(\lambda)}. \tag{20}$$

Good choices for λ provide better bounds.

The advantages of the transformation in Eq. (20) are significant in the context of graphical models. In particular, to obtain the joint probability in a graphical model we are required to take a product over the local conditional probabilities (cf. Eq. (2)). For conditional probabilities represented with logistic regression, we obtain products of functions of the form $f(x) = 1/(1 + e^{-x})$. Such a product is not in a simple form. If instead we augment our network representation by including variational parameters—i.e. representing each logistic function variationally as in Eq. (20)—we see that a bound on the joint probability is obtained by taking products of exponentials. This is tractable computationally, particularly so given that the exponents are linear in x.

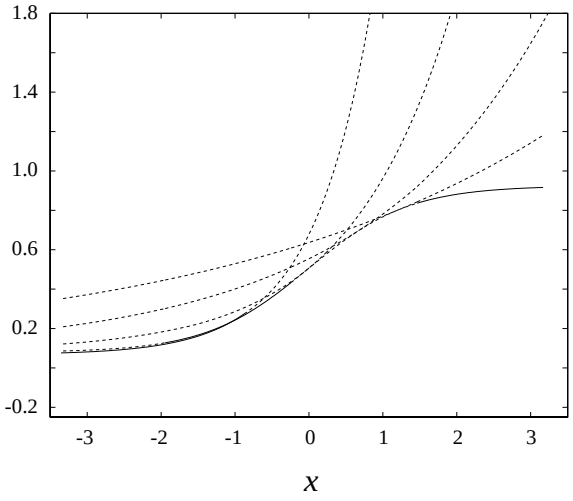


Figure 15: Variational transformation of the logistic function.

4.2 Convex duality

Can we find variational transformations more systematically? Indeed, many of the variational transformations that have been utilized in the literature on graphical models are examples of the general principle of *convex duality*. It is a general fact of convex analysis (Rockafellar, 1972) that a concave function $f(x)$ can be represented via a *conjugate* or *dual* function as follows:

$$f(x) = \min_{\lambda}\{\lambda^T x - f^*(\lambda)\}, \tag{21}$$

where we now allow x and λ to be vectors. The conjugate function $f^*(\lambda)$ can be obtained from the following dual expression:

$$f^*(\lambda) = \min_{x}\{\lambda^T x - f(x)\}. \tag{22}$$

This relationship is easily understood geometrically, as shown in Fig. 16. Here we plot $f(x)$ and the linear function λx for a particular value of λ. The short vertical segments represent values $\lambda x - f(x)$. It is clear from the figure that we need to shift the linear function λx vertically by an amount which is the minimum of the values $\lambda x - f(x)$ in order to obtain an upper bounding line with slope λ that touches $f(x)$ at a single point. This observation both justifies the form of the conjugate function, as a minimum over differences $\lambda x - f(x)$, and explains why the conjugate function appears as the intercept in Eq. (21).

It is an easy exercise to verify that the conjugate function for the logarithm is $f^*(\lambda) = \ln\lambda + 1$, and the conjugate function for the log logistic function is the binary entropy $H(\lambda)$.

Although we have focused on upper bounds in this section, the framework of convex duality applies equally well to lower bounds; in particular for *convex* $f(x)$ we have:

$$f(x) = \max_{\lambda}\{\lambda^T x - f^*(\lambda)\}, \tag{23}$$

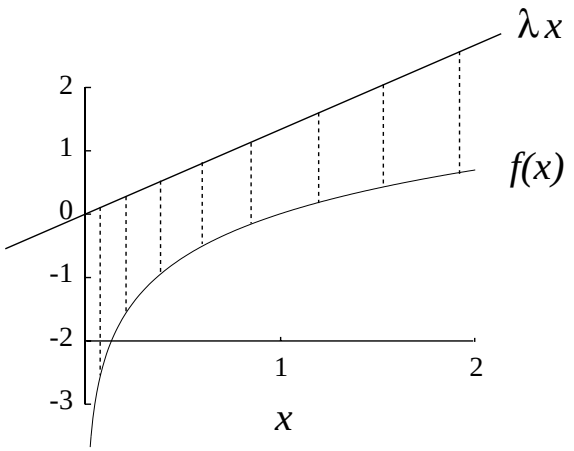


Figure 16: The conjugate function $f^*(\lambda)$ is obtained by minimizing across the deviations—represented as dashed lines—between λx and $f(x)$.

where

$$f^*(\lambda) = \max_x \{\lambda^T x - f(x)\} \tag{24}$$

is the conjugate function.

We have focused on linear bounds in this section, but convex duality is not restricted to linear bounds. More general bounds can be obtained by transforming the argument of the function of interest rather than the value of the function (Jaakkola & Jordan, 1997a). For example, if $f(x)$ is concave in x^2 we can write:

$$f(x) = \min_\lambda \{\lambda x^2 - \bar{f}^*(\lambda)\}, \tag{25}$$

where $\bar{f}^*(\lambda)$ is the conjugate function of $\bar{f}(x) \equiv f(x^2)$. Thus the transformation yields a quadratic bound on $f(x)$. It is also worth noting that such transformations can be combined with the logarithmic transformation utilized earlier to obtain Gaussian representations for the upper bounds. This can be useful in obtaining variational approximations for posterior distributions (Jaakkola & Jordan, 1997b).

To summarize, the general methodology suggested by convex duality is the following. We wish to obtain upper or lower bounds on a function of interest. If the function is already convex or concave then we simply calculate the conjugate function. If the function is not convex or concave, then we look for an invertible transformation that renders the function convex or concave. We may also consider transformations of the argument of the function. We then calculate the conjugate function in the transformed space and transform back. For this approach to be useful we need to to find a transform, such as the logarithm, whose inverse has useful algebraic properties.

4.3 Approximations for joint probabilities and conditional probabilities

The discussion thus far has focused on approximations for the local probability distributions at the nodes of a graphical model. How do these approximations translate into approximations for the global probabilities of interest, in particular for the conditional distribution $P(H|E)$ that is our interest in the inference problem and the marginal probability $P(E)$ that is our interest in learning problems?

Let us focus on directed graphs for concreteness. Suppose that we have a lower bound and an upper bound for each of the local conditional probabilities $P(S_i|S_{\pi(i)})$. That is, assume that we have forms $P^U(S_i|S_{\pi(i)}, \lambda_i^U)$ and $P^L(S_i|S_{\pi(i)}, \lambda_i^L)$, providing upper and lower bounds, respectively, where λ_i^U and λ_i^L are (generally different) variational parameterizations appropriate for the upper and lower bounds. Consider first the upper bounds. Given that the product of upper bounds is an upper bound, we have:

$$\begin{aligned} P(S) &= \prod_i P(S_i|S_{\pi(i)}) \\ &\leq \prod_i P^U(S_i|S_{\pi(i)}, \lambda_i^U) \end{aligned} \tag{26}$$

for any settings of values of the variational parameters λ_i^U. Moreover, Eq. (26) must hold for any subset of S whenever some other subset is held fixed, thus upper bounds on marginal probabilities can be obtained by taking sums over the variational form on the right-hand side of the equation. For example, letting E and H be a disjoint partition of S, we have:

$$\begin{aligned} P(E) &= \sum_{\{H\}} P(H, E) \\ &\leq \sum_{\{H\}} \prod_i P^U(S_i|S_{\pi(i)}, \lambda_i^U), \end{aligned} \tag{27}$$

where, as we will see in the examples to discussed below, we choose the variational forms $P^U(S_i|S_{\pi(i)}, \lambda_i^U)$ so that the summation over H can be carried out efficiently (this is the key step in developing a variational method). In either Eq. (26) or Eq. (27), given that these upper bounds hold for any settings of values the variational parameters λ_i^U, they hold in particular for optimizing settings of the parameters. That is, we can treat the right-hand side of Eq. (26) or the right-hand side Eq. (27) as a function to be minimized with respect to λ_i^U. In the latter case, this optimization process will induce interdependencies between the parameters λ_i^U. These interdependencies are desirable; indeed they are critical for obtaining a good variational bound on the marginal probability of interest. In particular, the best global bounds are obtained when the probabilistic dependencies in the distribution are reflected in dependencies in the approximation.

To clarify the nature of variational bounds, note that there is an important distinction to be made between joint probabilities (Eq. (26)) and marginal probabilities (Eq. (27)). In Eq. (26), if we allow the variational parameters to be set optimally for each value of the argument S, then it is possible (in principle) to find optimizing settings of the variational parameters that recover the exact value of the joint probability. (Here we assume that the

local probabilities $P(S_i|S_{\pi(i)})$ can be represented exactly via a variational transformation, as in the examples discussed in Section 4.1). In Eq. (27), on the other hand, we are *not* generally able to recover exact values of the marginal by optimizing over variational parameters that depend only on the argument E. Consider, for example, the case of a node $S_i \in E$ that has parents in H. As we range across $\{H\}$ there will be summands on the right-hand side of Eq. (27) that will involve evaluating the local probability $P(S_i|S_{\pi(i)})$ for different values of the parents $S_{\pi(i)}$. If the variational parameter λ_i^U depends only on E, we cannot in general expect to obtain an exact representation for $P(S_i|S_{\pi(i)})$ in each summand. Thus, some of the summands in Eq. (27) are necessarily bounds and not exact values.

This observation provides a bit of insight into reasons why a variational bound might be expected to be tight in some circumstances and loose in others. In particular, if $P(S_i|S_{\pi(i)})$ is nearly constant as we range across $S_{\pi(i)}$, or if we are operating at a point where the variational representation is fairly insensitive to the setting of λ_i^U (for example the right-hand side of the logarithm in Fig. 14), then the bounds may be expected to be tight. On the other hand, if these conditions are not present one might expect that the bound would be loose. However the situation is complicated by the interdependencies between the λ_i^U that are induced during the optimization process. We will return to these issues in the discussion.

Although we have discussed upper bounds, similar comments apply to lower bounds, and to marginal probabilities obtained from lower bounds on the joint distribution.

The conditional distribution $P(H|E)$, on the other hand, is the ratio of two marginal distributions; i.e., $P(H|E) = P(H,E)/P(E)$.[9] To obtain upper and lower bounds on the conditional distribution, we must have upper and lower bounds on both the numerator and the denominator. Generally speaking, however, if we can obtain upper and lower bounds on the denominator, then our labor is essentially finished, because the numerator involves fewer sums. Indeed, in the case in which $S = H \cup E$, the numerator involves no sums and is simply a function evaluation.

Finally, it is worth noting that variational methods can also be of interest simply as tractable approximations rather than as methods that provide strict bounds (much as sampling methods are used). One way to do this is to obtain a variational approximation that is a bound for a *marginal* probability, and to substitute the variational parameters thus obtained into the *conditional* probability distribution. Thus, for example, we might obtain a lower bound on the likelihood $P(E)$ by fitting variational parameters. We can substitute these parameters into the parameterized variational form for $P(H,E)$ and then utilize this variational form to calculate an approximation to $P(H|E)$.

In the following sections we will illustrate the general variational framework as it has been applied in a number of worked-out examples. All of these examples involve architectures of practical interest and provide concrete examples of variational methodology. To a certain degree the examples also serve as case histories that can be generalized to related architectures. It is important to emphasize, however, that it is not necessarily straightforward to develop a variational approximation for a new architecture. The ease and the utility

[9] Note that we treat $P(H,E)$ in general as a marginal probability; that is, we do not necessarily assume that H and E jointly exhaust the set of nodes S.

of applying the methods outlined in this section depend on architectural details, including the choice of node probability functions, the graph topology and the particular parameter regime in which the model is operated. In particular, certain choices of node conditional probability functions lend themselves more readily than others to variational transformations that have useful algebraic properties. Also, certain architectures simplify more readily under variational transformation than others; in particular, the marginal bounds in Eq. (27) are simple functions in some cases and complex in others. These issues are currently not well understood and the development of effective variational approximations can in some cases require substantial creativity.

4.4 Sequential and block methods

Let us now consider in somewhat more detail how variational methods can be applied to probabilistic inference problems. The basic idea is that suggested above—we wish to simplify the joint probability distribution by transforming the local probability functions. By an appropriate choice of variational transformation, we can simplify the form of the joint probability distribution and thereby simplify the inference problem. We can transform some or all of the nodes. The cost of performing such transformations is that we obtain bounds or approximations to the probabilities rather than exact results.

The option of transforming only some of the nodes is important; it implies a role for the exact methods as subroutines within a variational approximation. In particular, partial transformations of the graph may leave some of the original graphical structure intact and/or introduce new graphical structure to which exact methods can be fruitfully applied. In general, we wish to use variational approximations in a limited way, transforming the graph into a simplified graph to which exact methods can be applied. This will in general yield tighter bounds than an algorithm that transforms the entire graph without regard for computationally tractable substructure.

The majority of variational algorithms proposed in the literature to date can be divided into two main classes: *sequential* and *block*. In the sequential approach, nodes are transformed in an order that is determined during the inference process. This approach has the advantage of flexibility and generality, allowing the particular pattern of evidence to determine the best choices of nodes to transform. In some cases, however, particularly when there are obvious substructures in a graph which are amenable to exact methods, it can be advantageous to designate in advance the nodes to be transformed. We will see that this block approach is particularly natural in the setting of parameter estimation.

5 The sequential approach

The sequential approach introduces variational transformations for the nodes in a particular order. The goal is to transform the network until the resulting transformed network is amenable to exact methods. As we will see in the examples below, certain variational transformations can be understood graphically as a sparsification in which edges are removed from the graph. A series of edge removals eventually renders the graph sufficiently sparse

that an exact method becomes applicable. Alternatively, we can variationally transform all of the nodes of the graph and then reinstate the exact node probabilities sequentially while making sure that the resulting graph stays computationally tractable. The first example in the following section illustrates the latter approach and the second example illustrates the former approach.

Many of the exact methods provide tests that bound their run time. For example, one can run a greedy triangulation algorithm to upper bound the run time of the junction tree inference algorithm. If this estimated run time is sufficiently small, in terms of the overall time allotted to the inference procedure, the system can stop introducing variational transformations and run the exact procedure.

Ideally the choice of the order in which to transform nodes would be made optimally, that is, an ordering of the nodes would be chosen so that the resulting graph would be as simple as possible at each step (in particular, such that the maximal clique of the resulting triangulated graph would be as small as possible). Thus is a difficult problem, particularly given that a single ordering is unlikely to produce the simplest graph at each step; that is, different partial orders must be considered. In the literature to date heuristic procedures have been used to choose node orderings.

The sequential approach is perhaps best presented in the context of a specific example. In the following section we return to the QMR-DT network and show how a sequential variational approach can be used for inference in this network.

5.1 The QMR-DT network

Jaakkola and Jordan (1997c) present an application of sequential variational methods to the QMR-DT network. As we have seen, the QMR-DT network is a bipartite graph in which the conditional probabilities for the findings are based on the noisy-OR model (Eq. (8) for the negative findings and Eq. (9) for the positive findings). Note that symptom nodes that are not findings—i.e., symptoms that are not observed—can simply be marginalized out of the joint distribution by omission and therefore they have no impact on inference. Moreover, as we have discussed, the negative findings present no difficulties for inference—given the exponential form of the probability in Eq. (8), the effects of negative findings on the disease probabilities can be handled in linear time. Let us therefore assume that the updates associated with the negative findings have already been made and focus on the problem of performing inference when there are positive findings.

Repeating Eq. (9) for convenience, we have the following representation for the probability of a positive finding:

$$P(f_i = 1|d) = 1 - e^{-\sum_{j\in\pi(i)} \theta_{ij} d_j - \theta_{i0}} \tag{28}$$

The function $1 - e^{-x}$ is log concave; thus, as in the case of the logistic function, we are able to express the variational upper bound in terms of the exponential of a linear function. In particular:

$$1 - e^{-x} \leq e^{\lambda x - f^*(\lambda)}, \tag{29}$$

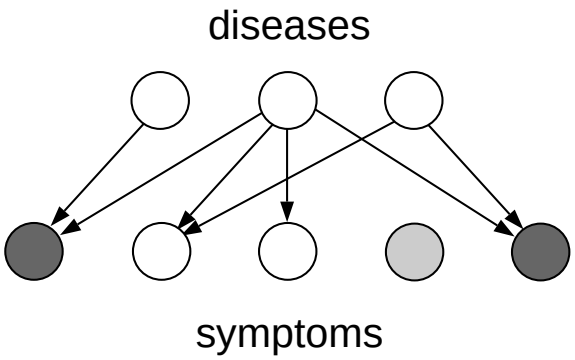


Figure 17: The QMR-DT graph after the lightly shaded finding has been subjected to a variational transformation. The effect is equivalent to delinking the node from the graph.

where the conjugate function is as follows:

$$f^*(\lambda) = -\lambda \ln \lambda + (\lambda + 1) \ln(\lambda + 1). \tag{30}$$

Plugging the argument of Eq. (28) into Eq. (29), and noting that we need a different variational parameter λ_i for each transformed node, we obtain:

$$P(f_i = 1|d) \leq e^{\lambda_i \left(\sum_{j \in \pi(i)} \theta_{ij} d_j + \theta_{i0} \right) - f^*(\lambda_i)} \tag{31}$$

$$= e^{\lambda_i \theta_{i0} - f^*(\lambda_i)} \prod_{j \in \pi(i)} \left[e^{\lambda_i \theta_{ij}} \right]^{d_j}. \tag{32}$$

The final equation displays the effect of the variational transformation. The exponential factor outside of the product is simply a constant. The product is taken over all nodes in the parent set for node i, but unlike the case in which the graph is moralized for exact computation, the contributions associated with the d_j nodes are uncoupled. That is, each factor $\exp(\lambda_i \theta_{ij})$ is simply a constant that is multiplied into the probability that was previously associated with node d_j (for $d_j = 1$). There is no coupling of d_j and d_k nodes as there would be if we had taken products of the untransformed noisy-OR. The graphical effect of the variational transformation is shown in Fig. 17; we see that the variational transformation essentially delinks the ith finding from the graph. In our particular example, the graph is now rendered singly connected and an exact inference algorithm can be invoked. (Recall that marginalizing over the *unobserved* symptoms simply removes them from the graph).

The sequential methodology utilized by Jaakkola and Jordan for inference in the QMR-DT network actually proceeds in the opposite direction. They first transform all of the nodes in the graph. They then make use of a simple heuristic to choose the ordering of nodes to reinstate, basing the choice on the effect of reinstating each node individually starting from the completely transformed state. (Despite the suboptimality of this heuristic, they found that it yielded an approximation that was orders of magnitude more accurate than that of an algorithm that used a random ordering). The algorithm then proceeds as follows:

(1) Pick a node to reinstate, and consider the effect of reintroducing the links associated with the node into the current graph. (2) If the resulting graph is still amenable to exact methods, reinstate the node and iterate. Otherwise stop and run an exact method. Finally, (3) we must also choose the parameters λ_i so as to make the approximation as tight as possible. It is not difficult to verify that products of the expression in Eq. (32) yield an overall bound that is a convex function of the λ_i parameters (Jaakkola & Jordan, 1997c). Thus standard optimization algorithms can be used to find good choices for the λ_i.

Jaakkola and Jordan (1997c) presented results for approximate inference on the "CPC cases" that were mentioned earlier. These are difficult cases which have up to 100 positive findings. Their study was restricted to upper bounds because it was found that the simple lower bounds that they tried were not sufficiently tight. They used the upper bounds to determine variational parameters that were subsequently used to form an approximation to the conditional posterior probability. They found that the variational approach yielded reasonably accurate approximations to the conditional posterior probabilities for the CPC cases, and did so within less than a minute of computer time.

5.2 The Boltzmann machine

Let us now consider a rather different example. As we have discussed, the Boltzmann machine is a special subset of the class of undirected graphical models in which the potential functions are composed of products of quadratic and linear "Boltzmann factors." Jaakkola and Jordan (1997a) introduced a sequential variational algorithm for approximate inference in the Boltzmann machine. Their method, which we discuss in this section, yields both upper and lower bounds on marginal and conditional probabilities of interest.

Recall the form of the joint probability distribution for the Boltzmann machine:

$$P(S) = \frac{e^{\sum_{i<j} \theta_{ij} S_i S_j + \sum_i \theta_{i0} S_i}}{Z}. \tag{33}$$

To obtain marginal probabilities such as $P(E)$ under this joint distribution, we must calculate sums over exponentials of quadratic energy functions. Moreover, to obtain conditional probabilities such as $P(H|E) = P(H,E)/P(E)$, we take ratios of such sums, where the numerator requires fewer sums than the denominator. The most general such sum is the partition function itself, which is a sum over *all* configurations $\{S\}$. Let us therefore focus on upper and lower bounds for the partition function as the general case; this allows us to calculate bounds on any other marginals or conditionals of interest.

Our approach is to perform the sums one sum at a time, introducing variational transformations to ensure that the resulting expression stays computationally tractable. In fact, at every step of the process that we describe, the transformed potentials involve no more than quadratic Boltzmann factors. (Exact methods can be viewed as creating increasingly higher-order terms when the marginalizing sums are performed). Thus the transformed Boltzmann machine remains a Boltzmann machine.

Let us first consider lower bounds. We write the partition function as follows:

$$\sum_{\{S\}} e^{\sum_{j<k} \theta_{jk} S_j S_k + \sum_j \theta_{j0} S_j} = \sum_{\{S \setminus S_i\}} \sum_{S_i \in \{0,1\}} e^{\sum_{j<k} \theta_{jk} S_j S_k + \sum_j \theta_{j0} S_j}, \tag{34}$$

and attempt to find a tractable lower bound on the inner summand over S_i on the right-hand side. It is not difficult to show that this expression is log convex. Thus we bound its logarithm variationally:

$$\begin{aligned}
&\ln\left(\sum_{S_i\in\{0,1\}} e^{\sum_{j<k}\theta_{jk}S_jS_k+\sum_j\theta_{j0}S_j}\right) \\
&= \sum_{\{j<k\}\neq i}\theta_{jk}S_jS_k+\sum_{j\neq i}\theta_{j0}S_j+\ln\left(\sum_{S_i\in\{0,1\}} e^{\sum_{j\neq i}\theta_{ij}S_iS_j+\theta_{i0}S_i}\right) \\
&= \sum_{\{j<k\}\neq i}\theta_{jk}S_jS_k+\sum_{j\neq i}\theta_{j0}S_j+\ln\left(1+e^{\sum_{j\neq i}\theta_{ij}S_j+\theta_{i0}}\right) \qquad (35)\\
&\geq \sum_{\{j<k\}\neq i}\theta_{jk}S_jS_k+\sum_{j\neq i}\theta_{j0}S_j+\lambda_i^L\left(\sum_{j\neq i}\theta_{ij}S_j+\theta_{i0}\right)+H(\lambda_i^L), \qquad (36)
\end{aligned}$$

where the sum in the first term on the right-hand side is a sum over all pairs $j<k$ such that neither j nor k is equal to i, where $H(\cdot)$ is as before the binary entropy function, and where λ_i^L is the variational parameter associated with node S_i. In the first line we have simply pulled outside of the sum all of those terms not involving S_i, and in the second line we have performed the sum over the two values of S_i. Finally, to lower bound the expression in Eq. (35) we need only lower bound the term $\ln(1+e^x)$ on the right-hand side. But we have already found variational bounds for a related expression in treating the logistic function; recall Eq. (18). The upper bound in that case translates into the lower bound in the current case:

$$\ln(1+e^{-x})\geq\lambda x+H(\lambda). \qquad (37)$$

This is the bound that we have utilized in Eq. (36).

Let us consider the graphical consequences of the bound in Eq. (36) (see Fig. 18). Note that for all nodes in the graph other than node S_i and its neighbors, the Boltzmann factors are unaltered (see the first two terms in the bound). Thus the graph is unaltered for such nodes. From the term in parentheses we see that the neighbors of node S_i have been endowed with new linear terms; importantly, however, these nodes have not become linked (as they would have become if we had done the exact marginalization). Neighbors that were linked previously remain linked with the same θ_{jk} parameter. Node S_i is absent from the transformed partition function and thus absent from the graph, but it has left its trace via the new linear Boltzmann factors associated with its neighbors. We can summarize the effects of the transformation by noting that the transformed graph is a new Boltzmann machine with one fewer node and the following parameters:

$$\begin{aligned}
\tilde{\theta}_{jk} &= \theta_{jk} & j,k &\neq i\\
\tilde{\theta}_{j0} &= \theta_{j0}+\lambda_i^L\theta_{ij} & j &\neq i\ .
\end{aligned}$$

Note finally that we also have a constant term $\lambda_i^L\theta_{i0}+H(\lambda_i^L)$ to keep track of. This term

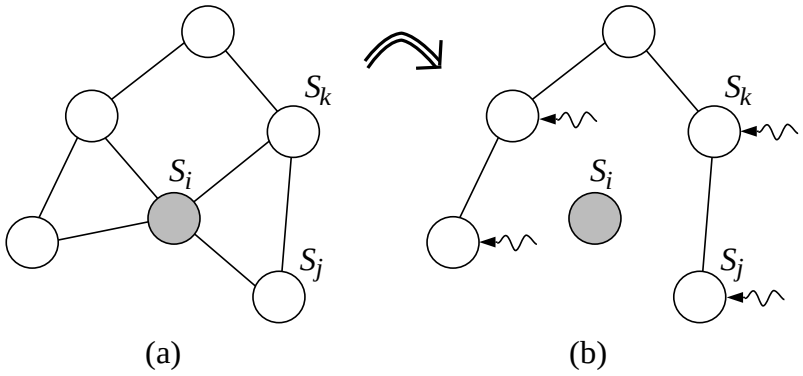


Figure 18: The transformation of the Boltzmann machine under the approximate marginalization over node S_i for the case of lower bounds. (a) The Boltzmann machine before the transformation. (b) The Boltzmann machine after the transformation, where S_i has become delinked. All of the pairwise parameters, θ_{jk}, for j and k not equal to i, have remained unaltered. As suggested by the wavy lines, the linear coefficients have changed for those nodes that were neighbors of S_i.

will have an interesting interpretation when we return to the Boltzmann machine later in the context of block methods.

Upper bounds are obtained in a similar way. We again break the partition function into a sum over a particular node S_i and a sum over the configurations of the remaining nodes $S \backslash S_i$. Moreover, the first three lines of the ensuing derivation leading to Eq. (35) are identical. To complete the derivation we now find an upper bound on $\ln(1+e^x)$. Jaakkola and Jordan (1997a) proposed using quadratic bounds for this purpose. In particular, they noted that:

$$\ln(1+e^x) = \ln(e^{x/2}+e^{-x/2}) + x/2 \tag{38}$$

and that $\ln(e^{x/2}+e^{-x/2})$ is a concave function of x^2 (as can be verified by taking the second derivative with respect to x^2). This implies that $\ln(1+e^x)$ must have a quadratic upper bound of the following form:

$$\ln(1+e^x) \leq \lambda x^2 + x/2 - \bar{g}^*(\lambda). \tag{39}$$

where $\bar{g}^*(\lambda)$ is an appropriately defined conjugate function. Using these upper bounds in Eq. (35) we obtain:

$$\begin{aligned} \ln\left(\sum_{S_i\in\{0,1\}} e^{\sum_{j<k}\theta_{jk}S_jS_k+\sum_j\theta_{j0}S_j}\right) &\leq \sum_{\{j<k\}\neq i}\theta_{jk}S_jS_k + \sum_{j\neq i}\theta_{j0}S_j \\ &+ \lambda_i^U\left(\sum_{j\neq i}\theta_{ij}S_j+\theta_{i0}\right)^2 + \frac{1}{2}\left(\sum_{j\neq i}\theta_{ij}S_j+\theta_{i0}\right) - \bar{g}^*(\lambda_i^U), \end{aligned} \tag{40}$$

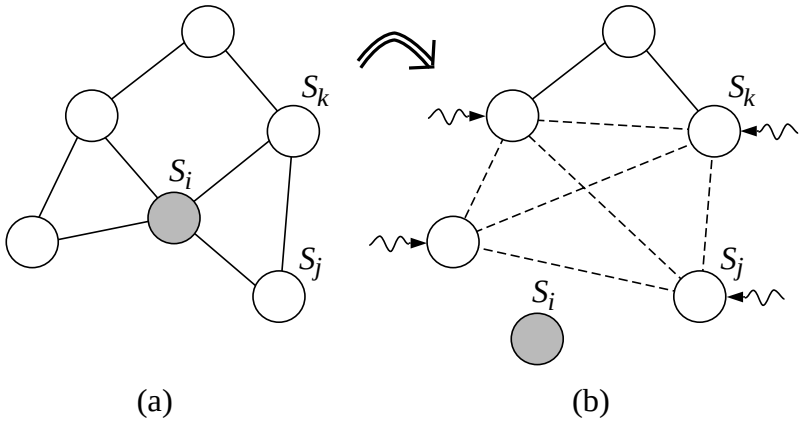


Figure 19: The transformation of the Boltzmann machine under the approximate marginalization over node S_i for the case of upper bounds. (a) The Boltzmann machine before the transformation. (b) The Boltzmann machine after the transformation, where S_i has become delinked. As the dashed edges suggest, all of the neighbors of S_i have become linked and those that were formerly linked have new parameter values. As suggested by the wavy lines, the neighbors of S_i also have new linear coefficients. All other edges and parameters are unaltered.

where λ_i^U is the variational parameter associated with node S_i.

The graphical consequences of this transformation are somewhat different than those of the lower bounds (see Fig. 19). Considering the first two terms in the bound, we see that it is still the case that the graph is unaltered for all nodes in the graph other than node S_i and its neighbors, and moreover neighbors of S_i that were previously linked remain linked. The quadratic term, however, gives rise to new links between the previously unlinked neighbors of node S_i and alters the parameters between previously linked neighbors. Each of these nodes also acquires a new linear term. Expanding Eq. (40) and collecting terms, we see that the approximate marginalization has yielded a Boltzmann machine with the following parameters:

$$\begin{aligned} \tilde{\theta}_{jk} &= \theta_{jk} + 2\lambda_i^U \theta_{ji}\theta_{ik} \qquad j,k \neq i \\ \tilde{\theta}_{j0} &= \theta_{j0} + \theta_{ij}/2 + 2\lambda_i^U \theta_{i0}\theta_{ij} + \lambda_i^U \theta_{ij}^2 \qquad j \neq i \ . \end{aligned}$$

Finally, the constant term is given by $\theta_{i0}/2 + \lambda_i^U \theta_{i0}^2 - \bar{g}^*(\lambda_i^U)$.

The graphical consequences of the lower and upper bound transformations also have computational consequences. In particular, given that the lower bound transformation introduces no additional links when nodes are delinked, it is somewhat more natural to combine these transformations with exact methods. In particular, the algorithm simply delinks nodes until a tractable structure (such as a tree) is revealed; at this point an exact algorithm is called as a subroutine. The upper bound transformation, on the other hand, by introducing links between the neighbors of a delinked node, does not reveal tractable structure as readily. This seeming disadvantage is mitigated by the fact that the upper

bound is a tighter bound (Jaakkola & Jordan, 1997a).

6 The block approach

An alternative approach to variational inference is to designate in advance a set of nodes that are to be transformed. We can in principle view this "block approach" as an off-line application of the sequential approach. In the case of lower bounds, however, there are advantages to be gained by developing a methodology that is specific to block transformation. In this section, we show that a natural global measure of approximation accuracy can be obtained for lower bounds via a block version of the variational formalism. The method meshes readily with exact methods in cases in which tractable substructure can be identified in the graph. This approach was first presented by Saul and Jordan (1996), as a refined version of mean field theory for Markov random fields, and has been developed further in a number of recent studies (e.g., Ghahramani & Jordan, 1997; Ghahramani & Hinton, 1996; Jordan, et al., 1997).

In the block approach, we begin by identifying a substructure in the graph of interest that we know is amenable to exact inference methods (or, more generally, to efficient approximate inference methods). For example, we might pick out a tree or a set of chains in the original graph. We wish to use this simplified structure to approximate the probability distribution on the original graph. To do so, we consider a family of probability distributions that are obtained from the simplified graph via the introduction of variational parameters. We choose a particular approximating distribution from the simplifying family by making a particular choice for the variational parameters. As in the sequential approach a new choice of variational parameters must be made each time new evidence is available.

More formally, let $P(S)$ represent the joint distribution on the graphical model of interest, where as before S represents all of the nodes of the graph and H and E are disjoint subsets of S representing the hidden nodes and the evidence nodes, respectively. We wish to approximate the conditional probability $P(H|E)$. We introduce an approximating family of conditional probability distributions, $Q(H|E,\lambda)$, where λ are variational parameters. The graph representing Q is not generally the same as the graph representing P; generally it is a sub-graph. From the family of approximating distributions Q, we choose a particular distribution by minimizing the Kullback-Leibler (KL) divergence, $D(Q\|P)$, with respect to the variational parameters:

$$\lambda^* = \mathrm{argmin}_\lambda \; D(Q(H|E,\lambda) \parallel P(H|E)), \tag{41}$$

where for any probability distributions $Q(S)$ and $P(S)$ the KL divergence is defined as follows:

$$D(Q\|P) = \sum_{\{S\}} Q(S) \ln \frac{Q(S)}{P(S)}. \tag{42}$$

The minimizing values of the variational parameters, λ^*, define a particular distribution, $Q(H|E,\lambda^*)$, that we treat as the best approximation of $P(H|E)$ in the family $Q(H|E,\lambda)$.

One simple justification for using the KL divergence as a measure of approximation accuracy is that it yields the best *lower bound* on the probability of the evidence $P(E)$ (i.e., the likelihood) in the family of approximations $Q(H|E,\lambda)$. Indeed, we bound the logarithm of $P(E)$ using Jensen's inequality as follows:

$$\begin{aligned} \ln P(E) &= \ln \sum_{\{H\}} P(H,E) \\ &= \ln \sum_{\{H\}} Q(H|E) \cdot \frac{P(H,E)}{Q(H|E)} \\ &\geq \sum_{\{H\}} Q(H|E) \ln \left[\frac{P(H,E)}{Q(H|E)} \right]. \end{aligned} \tag{43}$$

The difference between the left and right hand sides of this equation is easily seen to be the KL divergence $D(Q\|P)$. Thus, by the positivity of the KL divergence (Cover & Thomas, 1991), the right-hand side of Eq. (43) is a lower bound on $P(E)$. Moreover, by choosing λ according to Eq. (41), we obtain the tightest lower bound.

6.1 Convex duality and the KL divergence

We can also justify the choice of KL divergence by making an appeal to convex duality theory, thereby linking the block approach with the sequential approach (Jaakkola, 1997). Consider, for simplicity, the case of discrete-valued nodes H. The distribution $Q(H|E,\lambda)$ can be viewed as a vector of real numbers, one for each configuration of the variables H. Treat this vector as the vector-valued variational parameter "λ" in Eq. (23). Moreover, the log probability $\ln P(H,E)$ can also be viewed as a vector of real numbers, defined on the set of configurations of H. Treat this vector as the variable "x" in Eq. (23). Finally, define $f(x)$ to be $\ln P(E)$. It can be verified that the following expression for $\ln P(E)$:

$$\ln P(E) = \ln \left(\sum_{\{H\}} e^{\ln P(H,E)} \right) \tag{44}$$

is indeed convex in the values $\ln P(H,E)$. Moreover, by direct substitution in Eq. (23):

$$f^*(Q) = \min \left\{ \sum_{\{H\}} Q(H|E,\lambda) \ln P(H,E) - \ln P(E) \right\} \tag{45}$$

and minimizing with respect to $\ln P(H,E)$, the conjugate function $f^*(Q)$ is seen to be the negative entropy function $\sum_{\{H\}} Q(H|E) \ln Q(H|E)$. Thus, using Eq. (23), we can lower bound the log likelihood as follows:

$$\ln P(E) \geq \sum_{\{H\}} Q(H|E) \ln P(H,E) - Q(H|E) \ln Q(H|E) \tag{46}$$

This is identical to Eq. (43). Moreover, we see that we could in principle recover the exact log likelihood if Q were allowed to range over all probability distributions $Q(H|E)$. By ranging over a parameterized family $Q(H|E,\lambda)$, we obtain the tightest lower bound that is available within the family.

6.2 Parameter estimation via variational methods

Neal and Hinton (this volume) have pointed out that the lower bound in Eq. (46) has a useful role to play in the context of maximum likelihood parameter estimation. In particular, they make a link between this lower bound and parameter estimation via the EM algorithm.

Let us augment our notation to include parameters θ in the specification of the joint probability distribution $P(S|\theta)$. As before, we designate a subset of the nodes E as the observed evidence. The marginal probability $P(E|\theta)$, thought of as a function of θ, is known as the *likelihood*. The EM algorithm is a method for maximum likelihood parameter estimation that hillclimbs in the log likelihood. It does so by making use of the convexity relationship between $\ln P(H, E|\theta)$ and $\ln P(E|\theta)$ described in the previous section.

In Section 6 we showed that the function

$$\mathcal{L}(Q, \theta) = \sum_{\{H\}} Q(H|E) \ln P(H, E|\theta) - Q(H|E) \ln Q(H|E) \tag{47}$$

is a lower bound on the log likelihood for any probability distribution $Q(H|E)$. Moreover, we showed that the difference between $\ln P(E|\theta)$ and the bound $\mathcal{L}(Q, \theta)$ is the KL divergence between $Q(H|E)$ and $P(H|E)$. Suppose now that we allow $Q(H|E)$ to range over all possible probability distributions on H and minimize the KL divergence. It is a standard result (cf. Cover & Thomas, 1991) that the KL divergence is minimized by choosing $Q(H|E) = P(H|E, \theta)$, and that the minimal value is zero. This is verified by substituting $P(H|E, \theta)$ into the right-hand side of Eq. (47) and recovering $\ln P(E|\theta)$.

This suggests the following algorithm. Starting from an initial parameter vector $\theta^{(0)}$, we iterate the following two steps, known as the "E (expectation) step" and the "M (maximization) step." First, we maximize the bound $\mathcal{L}(Q, \theta)$ with respect to probability distributions Q. Second, we fix Q and maximize the bound $\mathcal{L}(Q, \theta)$ with respect to the parameters θ. More formally, we have:

$$\text{(E step)}: \quad Q^{(k+1)} = \text{argmax}_Q \ \mathcal{L}(Q, \theta^{(k)}) \tag{48}$$

$$\text{(M step)}: \quad \theta^{(k+1)} = \text{argmax}_\theta \ \mathcal{L}(Q^{(k+1)}, \theta) \tag{49}$$

which is coordinate ascent in $\mathcal{L}(Q, \theta)$.

This can be related to the traditional presentation of the EM algorithm (Dempster, Laird, & Rubin, 1977) by noting that for fixed Q, the right-hand side of Eq. (47) is a function of θ only through the $\ln P(H, E|\theta)$ term. Thus maximizing $\mathcal{L}(Q, \theta)$ with respect to θ in the M step is equivalent to maximizing the following function:

$$\sum_{\{H\}} P(H|E, \theta^{(k)}) \ln P(H, E|\theta). \tag{50}$$

Maximization of this function, known as the "complete log likelihood" in the EM literature, defines the M step in the traditional presentation of EM.

Let us now return to the situation in which we are unable to compute the full conditional distribution $P(H|E, \theta)$. In such cases variational methodology suggests that we consider a family of approximating distributions. Although we are no longer able to perform a

true EM iteration given that we cannot avail ourselves of $P(H|E,\theta)$, we can still perform coordinate ascent in the lower bound $\mathcal{L}(Q,\theta)$. Indeed, the variational strategy of minimizing the KL divergence with respect to the variational parameters that define the approximating family is exactly a restricted form of coordinate ascent in the first argument of $\mathcal{L}(Q,\theta)$. We then follow this step by an "M step" that increases the lower bound with respect to the parameters θ.

This point of view, which can be viewed as a computationally tractable approximation to the EM algorithm, has been exploited in a number of recent architectures, including the sigmoid belief network, factorial hidden Markov model and hidden Markov decision tree architectures that we discuss in the following sections, as well as the "Helmholtz machine" of Dayan, et al. (1995) and Hinton, et al. (1995).

6.3 Examples

We now return to the problem of picking a tractable variational parameterization for a given graphical model. We wish to pick a simplified graph which is both rich enough to provide distributions that are close to the true distribution, and simple enough so that an exact algorithm can be utilized efficiently for calculations under the approximate distribution. Similar considerations hold for the variational parameterization: the variational parameterization must be representationally rich so that good approximations are available and yet simple enough so that a procedure that minimizes the KL divergence has some hope of finding good parameters and not getting stuck in a local minimum. It is not necessarily possible to realize all of these desiderata simultaneously; however, in a number of cases it has been found that relatively simple variational approximations can yield reasonably accurate solutions. In this section we discuss several such examples.

6.3.1 Mean field Boltzmann machine

In Section 5.2 we discussed a sequential variational algorithm that yielded upper and lower bounds for the Boltzmann machine. We now revisit the Boltzmann machine within the context of the block approach and discuss lower bounds. We also relate the two approaches.

Recall that the joint probability for the Boltzmann machine can be written as follows:

$$P(S|\theta) = \frac{e^{\sum_{i<j}\theta_{ij}S_iS_j + \sum_i \theta_{i0}S_i}}{Z}, \tag{51}$$

where $\theta_{ij} = 0$ for nodes S_i and S_j that are not neighbors in the graph. Consider now the representation of the conditional distribution $P(H|E,\theta)$ in a Boltzmann machine. For nodes $S_i \in E$ and $S_j \in E$, the contribution $\theta_{ij}S_iS_j$ reduces to a constant, which vanishes when we normalize. If $S_i \in H$ and $S_j \in E$, the quadratic contribution becomes a linear contribution that we associate with node S_i. Finally, linear terms associated with nodes $S_i \in E$ also become constants and vanish. In summary, we can express the conditional distribution $P(H|E,\theta)$ as follows:

$$P(H|E,\theta) = \frac{e^{\sum_{i<j}\theta_{ij}S_iS_j + \sum_i \theta^c_{i0}S_i}}{Z_c}, \tag{52}$$

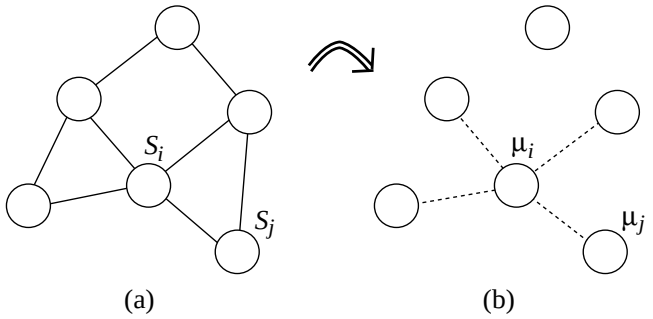


Figure 20: (a) A node S_i in a Boltzmann machine with its Markov blanket. (b) The approximating mean field distribution Q is based on a graph with no edges. The mean field equations yield a deterministic relationship, represented in the figure with the dotted lines, between the variational parameters μ_i and μ_j for nodes j in the Markov blanket of node i.

where the sums are restricted to range over nodes in H and the updated parameters θ^c_{i0} include contributions associated with the evidence nodes:

$$\theta^c_{i0} = \theta_{i0} + \sum_{j \in E} \theta_{ij} S_j. \tag{53}$$

The updated partition function Z_c is given as follows:

$$Z_c = \sum_{\{H\}} \left\{ e^{\sum_{i<j} \theta_{ij} S_i S_j + \sum_i \theta^c_{i0} S_i} \right\}. \tag{54}$$

In sum, we have a Boltzmann machine on the subset H.

The "mean field" approximation (Peterson & Anderson, 1987) for Boltzmann machines is a particular form of variational approximation in which a completely factorized distribution is used to approximate $P(H|E,\theta)$. That is, we consider the simplest possible approximating distribution; one that is obtained by dropping *all* of the edges in the Boltzmann graph (see Fig. 20). For this choice of $Q(H|E,\mu)$, (where we now use μ to represent the variational parameters), we have little choice as to the variational parameterization—to represent as large an approximating family as possible we endow each degree of freedom S_i with its own variational parameter μ_i. Thus Q can be written as follows:

$$Q(H|E,\mu) = \prod_{i \in H} \mu_i^{S_i} (1-\mu_i)^{1-S_i}, \tag{55}$$

where the product is taken over the hidden nodes H.

Forming the KL divergence between the fully factorized Q distribution and the P distribution in Eq. (52), we obtain:

$$\begin{aligned} D(Q\|P) = &\sum_i \left[\mu_i \ln \mu_i + (1-\mu_i)\ln(1-\mu_i)\right] \\ &- \sum_{i<j} \theta_{ij}\mu_i\mu_j - \sum_i \theta^c_{i0}\mu_i + \ln Z_c, \end{aligned} \tag{56}$$

where the sums range across nodes in H. In deriving this result we have used the fact that, under the Q distribution, S_i and S_j are independent random variables with mean values μ_i and μ_j.

We now take derivatives of the KL divergence with respect to μ_i—noting that Z_c is independent of μ_i—and set the derivative to zero to obtain the following equations:

$$\mu_i = \sigma\left(\sum_j \theta_{ij}\mu_j + \theta_{i0}\right), \tag{57}$$

where $\sigma(z) = 1/(1+e^{-z})$ is the logistic function and we define θ_{ij} equal to θ_{ji} for $j < i$. Eq. (57) defines a set of coupled equations known as the "mean field equations." These equations are solved iteratively for a fixed point solution. Note that each variational parameter μ_i updates its value based on a sum across the variational parameters in its Markov blanket (cf. Fig. 20b). This can be viewed as a variational form of a local message passing algorithm.

The mean field approximation for Boltzmann machines can provide a reasonably good approximation to conditional distributions in dense Boltzmann machines, and is the basis of a useful approach to combinatorial optimization known as "deterministic annealing." There are also cases, however, in which it is known to break down. These cases include sparse Boltzmann machines and Boltzmann machines with "frustrated" interactions; these are networks whose potential functions embody constraints between neighboring nodes that cannot be simultaneously satisfied (see also Galland, 1993). In the case of sparse networks, exact algorithms can provide help; indeed, this observation led to the use of exact algorithms as subroutines within the "structured mean field" approach pursued by Saul and Jordan (1996).

Let us now consider the parameter estimation problem for Boltzmann machines. Writing out the lower bound in Eq. (47) for this case, we have:

$$\begin{aligned}\ln P(E|\theta) \geq & \sum_{i<j}\theta_{ij}\mu_i\mu_j + \sum_i \theta^c_{i0}\mu_i - \ln Z \\ & - \sum_i \left[\mu_i \ln \mu_i + (1-\mu_i)\ln(1-\mu_i)\right]\end{aligned} \tag{58}$$

Taking the derivative with respect to θ_{ij} yields a gradient which has a simple "Hebbian" term $\mu_i\mu_j$ as well as a contribution from the derivative of $\ln Z$ with respect to θ_{ij}. It is not hard to show that this derivative is $\langle S_i S_j\rangle$; where the brackets signify an average with respect to the unconditional distribution $P(S|\theta)$. Thus we have the following gradient algorithm for performing an approximate M step:

$$\Delta\theta_{ij} \propto (\mu_i\mu_j - \langle S_i S_j\rangle). \tag{59}$$

Unfortunately, however, given our assumption that calculations under the Boltzmann distribution are intractable for the graph under consideration, it is intractable to compute the unconditional average. We can once again appeal to mean field theory and compute an approximation to $\langle S_i S_j\rangle$, where we now use a factorized distribution on all of the nodes;

however, the M step is now a difference of gradients of two different bounds and is therefore no longer guaranteed to increase $\mathcal{L}$. There is a more serious problem, moreover, which is particularly salient in unsupervised learning problems. If the data set of interest is a heterogeneous collection of sub-populations, such as in unsupervised classification problems, the unconditional distribution will generally be required to have multiple modes. Unfortunately the factorized mean field approximation is unimodal and is a poor approximation for a multi-modal distribution. One approach to this problem is to utilize multi-modal Q distributions within the mean-field framework; for example, Jaakkola and Jordan (this volume) discuss the use of mixture models as approximating distributions.

These issues find a more satisfactory treatment in the context of directed graphs, as we see in the following section. In particular, the gradient for a directed graph (cf. Eq. (68)) does not require averages under the unconditional distribution.

Finally, let us consider the relationship between the mean field approximation and the lower bounds that we obtained via a sequential algorithm in Section 5.2. In fact, if we run the latter algorithm until all nodes are eliminated from the graph, we obtain a bound that is identical to the mean field bound (Jaakkola, 1997). To see this, note that for a Boltzmann machine in which all of the nodes have been eliminated there are no quadratic and linear terms; only the constant terms remain. Recall from Section 5.2 that the constant that arises when node i is removed is $\mu_i^L \hat{\theta}_{i0} + H(\mu_i^L)$, where $\hat{\theta}_{i0}$ refers to the value of θ_{i0} after it has been updated to absorb the linear terms from previously eliminated nodes $j < i$. (Recall that the latter update is given by $\tilde{\theta}_{i0} = \theta_{i0} + \mu_i^L \theta_{ij}$ for the removal of a particular node j). Collecting together such updates for $j < i$, and summing across all nodes i, we find that the resulting constant term is given as follows:

$$\sum_i \left\{ \hat{\theta}_{i0} \mu_i + H(\mu_i) \right\} = \sum_{i<j} \theta_{ij} \mu_i \mu_j + \sum_i \theta_{i0}^c \mu_i - \sum_i \left[\mu_i \ln \mu_i + (1 - \mu_i) \ln(1 - \mu_i) \right] \tag{60}$$

This differs from the lower bound in Eq. (58) only by the term $\ln Z$, which disappears when we maximize with respect to μ_i.

6.3.2 Neural networks

As discussed in Section 3, the "sigmoid belief network" is essentially a (directed) neural network with graphical model semantics. We utilize the logistic function as the node probability function:

$$P(S_i = 1 | S_{\pi(i)}) = \frac{1}{1 + e^{-\sum_{j \in \pi(i)} \theta_{ij} S_j - \theta_{i0}}}, \tag{61}$$

where we assume that $\theta_{ij} = 0$ unless j is a parent of i. (In particular, $\theta_{ij} \neq 0 \Rightarrow \theta_{ji} = 0$). Noting that the probabilities for both the $S_i = 0$ case and the $S_i = 1$ case can be written

in a single expression as follows:

$$P(S_i|S_{\pi(i)}) = \frac{e^{\left(\sum_{j\in\pi(i)}\theta_{ij}S_j+\theta_{i0}\right)S_i}}{1+e^{\sum_{j\in\pi(i)}\theta_{ij}S_j+\theta_{i0}}}, \tag{62}$$

we obtain the following representation for the joint distribution:

$$P(S|\theta) = \prod_i \left[\frac{e^{\left(\sum_{j\in\pi(i)}\theta_{ij}S_j+\theta_{i0}\right)S_i}}{1+e^{\sum_{j\in\pi(i)}\theta_{ij}S_j+\theta_{i0}}}.\right], \tag{63}$$

We wish to calculate conditional probabilities under this joint distribution.

As we have seen (cf. Fig. 6), inference for general sigmoid belief networks is intractable, and thus it is sensible to consider variational approximations. Saul, Jaakkola, and Jordan (1996) and Saul and Jordan (this volume) have explored the viability of the simple completely factorized distribution. Thus once again we set:

$$Q(H|E,\mu) = \prod_{i\in H}\mu_i^{S_i}(1-\mu_i)^{1-S_i}, \tag{64}$$

and attempt to find the best such approximation by varying the parameters μ_i.

The computation of the KL divergence $D(Q\|P)$ proceeds much as it does in the case of the mean field Boltzmann machine. The entropy term $(Q\ln Q)$ is the same as before. The energy term $(Q\ln P)$ is found by taking the logarithm of Eq. (63) and averaging with respect to Q. Putting these results together, we obtain:

$$\begin{aligned}\ln P(E|\theta) \geq & \sum_{i<j}\theta_{ij}\mu_i\mu_j + \sum_i \theta_{i0}^c\mu_i \\ & - \sum_i \left\langle \ln\left[1+e^{\sum_{j\in\pi(i)}\theta_{ij}S_j+\theta_{i0}}\right]\right\rangle \\ & - \sum_i \left[\mu_i\ln\mu_i + (1-\mu_i)\ln(1-\mu_i)\right]\end{aligned} \tag{65}$$

where $\langle\cdot\rangle$ denotes an average with respect to the Q distribution. Note that, despite the fact that Q is factorized, we are unable to calculate the average of $\ln[1+e^{z_i}]$, where z_i denotes $\sum_{j\in\pi(i)}\theta_{ij}S_j+\theta_{i0}$. This is an important term which arises directly from the directed nature of the sigmoid belief network (it arises from the denominator of the sigmoid, a factor which is necessary to define the sigmoid as a local conditional probability). To deal with this term, Saul et al. (1996) introduced additional variational parameters ξ_i. These parameters can be viewed as providing a tight form of Jensen's inequality. Note in particular that we require an upper bound on $\langle\ln[1+e^{z_i}]\rangle$ (given that this term appears with a negative sign in Eq. (65)). Jensen's inequality provides such a bound, however Saul et al. found that this bound was not sufficiently tight and introduced a tighter bound due to Seung (1995). In

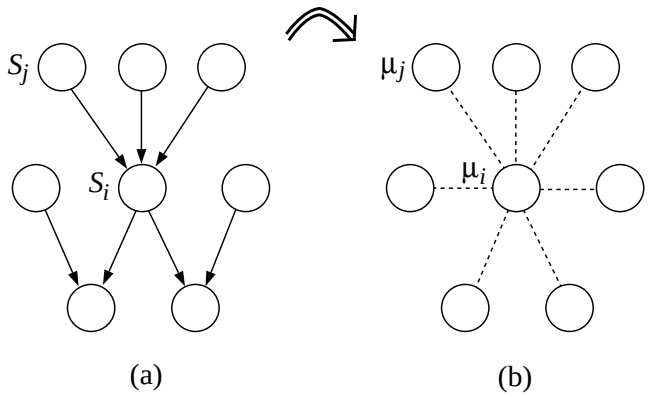


Figure 21: (a) A node S_i in a sigmoid belief network machine with its Markov blanket. (b) The mean field equations yield a deterministic relationship, represented in the figure with the dotted lines, between the variational parameters μ_i and μ_j for nodes j in the Markov blanket of node i.

particular:

$$\begin{aligned} \langle \ln[1+e^{z_i}] \rangle &= \left\langle \ln[e^{\xi_i z_i} e^{-\xi_i z_i}(1+e^{z_i})] \right\rangle \\ &= \xi_i \langle z_i \rangle + \left\langle \ln[e^{-\xi_i z_i} + e^{(1-\xi_i) z_i}] \right\rangle \\ &\leq \xi_i \langle z_i \rangle + \ln \left\langle e^{-\xi_i z_i} + e^{(1-\xi_i) z_i} \right\rangle, \end{aligned} \tag{66}$$

which reduces to standard Jensen for $\xi_i = 0$. The final result can be utilized directly in Eq. (65) to provide a tractable lower bound on the log likelihood.

Saul and Jordan (this volume) show that in the limiting case of networks in which each hidden node has a large number of parents, so that a central limit theorem can be invoked, the parameter ξ_i has a probabilistic interpretation as the approximate expectation of $\sigma(z_i)$, where $\sigma(\cdot)$ is again the logistic function.

For fixed values of the parameters ξ_i, by differentiating the KL divergence with respect to the variational parameters μ_i, we obtain the following consistency equations:

$$\mu_i = \sigma \left(\sum_j \theta_{ij} \mu_j + \theta_{i0} + \sum_j \theta_{ji} (\mu_j - \xi_j) + K_{ij} \right) \tag{67}$$

where K_{ij} is an expression that depends on node i, its child j, and the other parents (the "co-parents") of node j. Given that the first term is a sum over contributions from the parents of node i, and the second term is a sum over contributions from the children of node i, we see that the consistency equation for a given node again involves contributions from the Markov blanket of the node (see Fig. 21). Thus, as in the case of the Boltzmann machine, we find that the variational parameters are linked via their Markov blankets and the consistency equation (Eq. (67)) can be interpreted as a local message-passing algorithm.

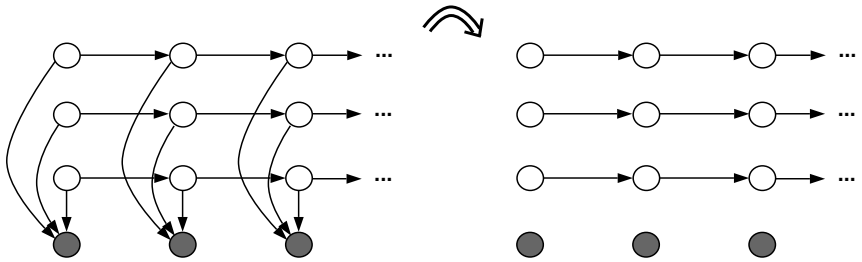


Figure 22: (a) The FHMM. (b) A variational approximation for the FHMM can be obtained by picking out a tractable substructure in the FHMM graph. Parameterizing this graph leads to a family of tractable approximating distributions.

Saul, Jaakkola, and Jordan (1996) and Saul and Jordan (this volume) also show how to update the variational parameters ξ_i. The two papers utilize these parameters in slightly different ways and obtain different update equations. Yet another variational approximation for the sigmoid belief network, including both upper and lower bounds, is presented in Jaakkola and Jordan (1996).

Finally, we can compute the gradient with respect to the parameters θ_{ij} for fixed variational parameters μ and ξ. The result obtained by Saul and Jordan (this volume) takes the following form:

$$\Delta\theta_{ij} \propto (\mu_i - \xi_i)\mu_j - \theta_{ij}\xi_i(1-\xi_i)\mu_i(1-\mu_i). \tag{68}$$

Note that there is no need to calculate variational parameters under the unconditional distribution, $P(S|\theta)$, as in the case of the Boltzmann machine (a fact first noted by Neal, 1992). Note also the interesting appearance of a regularization term—the second term in the equation is a "weight decay" term that is maximal for non-extreme values of the variational parameters (both of these parameters are bounded between zero and one).

Saul, et al. (1996) tested the sigmoid belief network on a handwritten digit recognition problem, obtaining results that were competitive with other supervised learning systems. An important advantage of the graphical model approach is its ability to deal with missing data. Indeed, Saul and Jordan (this volume) report that the degradation in performance with missing pixels in the digits is slight. For further comparative empirical work on sigmoid belief networks and related architectures, including comparisons with Gibbs sampling, see Frey, Hinton, and Dayan (1996).

6.3.3 Factorial hidden Markov models

The factorial hidden Markov model (FHMM) is a multiple chain structure (see Fig. 22(a)). Using the notation developed earlier (see Section 3.5), the joint probability distribution for the FHMM is given by:

$$P(\{X_t^{(m)}\},\{Y_t\}|\theta) = \prod_{m=1}^{M}\left[\pi^{(m)}(X_1^{(m)})\prod_{t=2}^{T}A^{(m)}(X_t^{(m)}|X_{t-1}^{(m)})\right]\prod_{t=1}^{T}P(Y_t|\{X_t^{(m)}\}_{m=1}^{M}) \tag{69}$$

Computation under this probability distribution is generally infeasible, because, as we saw earlier, the clique size becomes unmanageably large when the FHMM chain structure is moralized and triangulated. Thus it is necessary to consider approximations.

For the FHMM there is a natural substructure on which to base a variational algorithm. In particular, the chains that compose the FHMM are individually tractable. Therefore, rather than removing all of the edges, as in the naive mean field approximation discussed in the previous two sections, it would seem more reasonable to remove only as many edges as are necessary to decouple the chains. In particular, we remove the edges that link the state nodes to the output nodes (see Fig. 22(b)). Without these edges the moralization process no longer links the state nodes and no longer creates large cliques. In fact, the moralization process on the delinked graph in Fig. 22(b) is vacuous, as is the triangulation. Thus the cliques on the delinked graph are of size N^2, where N is the number of states for a single chain. Inference in the approximate graph runs in time $O(MTN^2)$, where M is the number of chains and T is the length of the time series.

Let us now consider how to express a variational approximation using the delinked graph of Fig. 22(b) as an approximation. The idea is to introduce one free parameter into the approximating probability distribution, Q, for each edge that we have dropped. These free parameters, which we denote as $\lambda_t^{(m)}$, essentially serve as surrogates for the effect of the observation at time t on state component m. When we optimize the divergence $D(Q\|P)$ with respect to these parameters they become interdependent; this (deterministic) interdependence can be viewed as an approximation to the probabilistic dependence that is captured in an exact algorithm via the moralization process.

Referring to Fig. 22(b), we write the approximating Q distribution in the following factorized form:

$$Q(\{X_t^{(m)}\}|\{Y_t\},\theta,\lambda) = \prod_{m=1}^{M} \tilde{\pi}^{(m)}(X_1^{(m)}) \prod_{t=2}^{T} \tilde{A}^{(m)}(X_t^{(m)}|X_{t-1}^{(m)}), \tag{70}$$

where λ is the vector of variational parameters $\lambda_t^{(m)}$. We define the transition matrix $\tilde{A}(m)$ to be the product of the exact transition matrix $A(m)$ and the variational parameter $\lambda_t^{(m)}$:

$$\tilde{A}^{(m)}(X_t^{(m)}|X_{t-1}^{(m)}) = A^{(m)}(X_t^{(m)}|X_{t-1}^{(m)})\lambda_t^{(m)}, \tag{71}$$

and similarly for the initial state probabilities $\tilde{\pi}^{(m)}$:

$$\tilde{\pi}^{(m)}(X_1^{(m)}) = \pi^{(m)}(X_1^{(m)})\lambda_1^{(m)}. \tag{72}$$

This family of distributions respects the conditional independence statements of the approximate graph in Fig. 22, and provides additional degrees of freedom via the variational parameters.

Ghahramani and Jordan (1997) present the equations that result from minimizing the KL divergence between the approximating probability distribution (Eq. (70)) and the true probability distribution (Eq. (69)). The result can be summarized as follows. As in the other architectures that we have discussed, the equation for a variational parameter ($\lambda_t^{(m)}$)

is a function of terms that are in the Markov blanket of the corresponding delinked node (i.e., Y_t). In particular, the update for $\lambda_t^{(m)}$ depends on the parameters $\lambda_t^{(n)}$, for $n \neq m$, thus linking the variational parameters at time t. Moreover, the update for $\lambda_t^{(m)}$ depends on the expected value of the states $X_t^{(m)}$, where the expectation is taken under the distribution Q. Given that the chains are decoupled under Q, expectations are found by running one of the exact algorithms (for example, the forward-backward algorithm for HMMs), separately for each chain. These expectations of course depend on the current values of the parameters $\lambda_t^{(m)}$ (cf. Eq. (70)), and it is this dependence that effectively couples the chains.

To summarize, fitting the variational parameters for a FHMM is an iterative, two-phase procedure. In the first phase, an exact algorithm is run as a subroutine to calculate expectations for the hidden states. This is done independently for each of the M chains, making reference to the current values of the parameters $\lambda_t^{(m)}$. In the second phase, the parameters $\lambda_t^{(m)}$ are updated based on the expectations computed in the first phase. The procedure then returns to the first phase and iterates.

Ghahramani and Jordan (1997) reported results on fitting an FHMM to the Bach chorale data set (Merz & Murphy, 1996). They showed that significantly larger effective state spaces could be fit with the FHMM than with an unstructured HMM, and that performance in terms of probability of the test set was an order of magnitude larger for the FHMM. Moreover, evidence of overfitting was seen for the HMM for 35 states or more; no evidence of overfitting for the FHMM was seen for up to 1000 states.

6.3.4 Hidden Markov decision trees

As a final example we return to the hidden Markov decision tree (HMDT) described in the introduction and briefly discuss variational approximation for this architecture. As we have discussed, a HMDT is essentially a Markov time series model, where the probability model at each time step is a (probabilistic) decision tree with hidden decision nodes. The Markovian dependence is obtained via separate transition matrices at the different levels of the decision tree, giving the model a factorized structure.

The variational approach to fitting a HMDT is closely related to that of fitting a FHMM; however, there are additional choices as to the variational approximation. In particular, we have two substructures worth considering in the HMDT: (1) Dropping the vertical edges, we recover a decoupled set of chains. As in the FHMM, these chains can each be handled by the forward-backward algorithm. (2) Dropping the horizontal edges, we recover a decoupled set of decision trees. We can calculate probabilities in these trees using the posterior propagation algorithm described in Jordan (1994).

The first approach, which we refer to as the "forest of chains approximation," is shown in Fig. 23. As in the FHMM, we write a variational approximation for the forest of chains approximation by respecting the conditional independencies in the approximating graph and incorporating variational parameters to obtain extra degrees of freedom (see Jordan, et al., 1997, for the details).

We can also consider a "forest of trees approximation" in which the horizontal links are eliminated (see Fig. 24). Given that the decision tree is a fully connected graph, this is

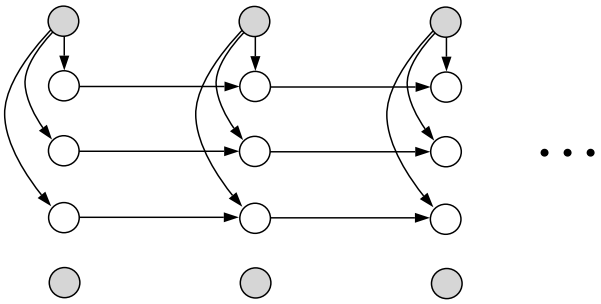


Figure 23: The "forest of chains approximation" for the HMDT. Parameterizing this graph leads to an approximating family of Q distributions.

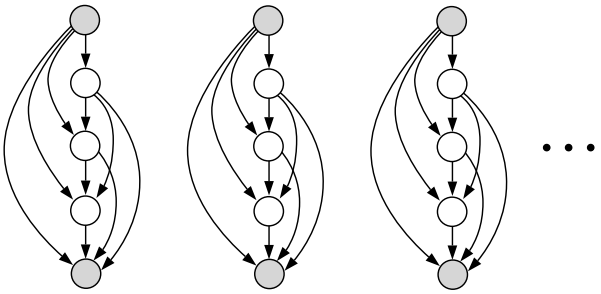


Figure 24: The "forest of trees approximation" for the HMDT. Parameterizing this graph leads to an approximating family of Q distributions.

essentially a naive mean field approximation on a hypergraph.

Finally, it is also possible to develop a variational algorithm for the HMDT that is analogous to the Viterbi algorithm for HMMs. In particular, we utilize an approximation Q that assigns probability one to a single path in the state space. The KL divergence for this Q distribution is particularly easy to evaluate, given that the entropy contribution to the KL divergence (i.e., the $Q \ln Q$ term) is zero. Moreover, the evaluation of the energy (i.e., the $Q \ln P$ term) reduces to substituting the states along the chosen path into the P distribution.

The resulting algorithm involves a subroutine in which a standard Viterbi algorithm is run on a single chain, with the other chains held fixed. This subroutine is run on each chain in turn.

Jordan, et al. (1997) found that performance of the HMDT on the Bach chorales was essentially the same as that of the FHMM. The advantage of the HMDT was its greater interpretability; most of the runs resulted in a coarse-to-fine ordering of the temporal scales of the Markov processes from the top to the bottom of the tree.

7 Discussion

We have described a variety of applications of variational methods to problems of inference and learning in graphical models. We hope to have convinced the reader that variational methods can provide a powerful and elegant tool for graphical models, and that the algorithms that result are simple and intuitively appealing. It is important to emphasize, however, that research on variational methods for graphical models is of quite recent origin, and there are many open problems and unresolved issues. In this section we discuss a number of these issues. We also broaden the scope of the presentation and discuss a number of related strands of research.

7.1 Related research

The methods that we have discussed all involve deterministic, iterative approximation algorithms. It is of interest to discuss related approximation schemes that are either nondeterministic or non-iterative.

7.1.1 Recognition models and the Helmholtz machine

All of the algorithms that we have presented have at their core a nonlinear optimization problem. In particular, after having introduced the variational parameters, whether sequentially or as a block, we are left with a bound such as that in Eq. (27) that must be optimized. Optimization of this bound is generally achieved via a fixed-point iteration or a gradient-based algorithm. This iterative optimization process induces interdependencies between the variational parameters which give us a "best" approximation to the marginal or conditional probability of interest.

Consider in particular a problem in which a directed graphical model is used for unsupervised learning. A common approach in unsupervised learning is to consider graphical models that are oriented in the "generative" direction; that is, they point from hidden variables to observables. In this case the "predictive" calculation of $P(E|H)$ is elementary. The calculation of $P(H|E)$, on the other hand, is a "diagnostic" calculation that proceeds backwards in the graph. Diagnostic calculations are generally non-trivial and require the full power of an inference algorithm.

An alternative approach to solving iteratively for an approximation to the diagnostic calculation is to learn both a generative model and a "recognition" model that approximates the diagnostic distribution $P(H|E)$. Thus we associate different parameters with the generative model and the recognition model and rely on the parameter estimation process to bring these parameterizations into register. This is the basic idea behind the "Helmholtz machine" (Dayan, et al., 1995; Hinton, et al., 1995).

The key advantage of the recognition-model approach is that the calculation of $P(H|E)$ is reduced to an elementary feedforward calculation that can be performed quickly.

There are some disadvantages to the approach as well. In particular, the lack of an iterative algorithm makes the Helmholtz machine unable to deal naturally with missing data, and with phenomena such as "explaining-away," in which the couplings between

hidden variables change as a function of the conditioning variables. Moreover, although in some cases there is a clear natural parameterization for the recognition model that is induced from the generative model (in particular for linear models such as factor analysis), in general it is difficult to insure that the models are matched appropriately.[10] Some of these problems might be addressed by combining the recognition-model approach with the iterative variational approach; essentially treating the recognition-model as a "cache" for storing good initializations for the variational parameters.

7.1.2 Sampling methods

In this section we make a few remarks on the relationships between variational methods and stochastic methods, in particular the Gibbs sampler. In the setting of graphical models, both classes of methods rely on extensive message-passing. In Gibbs sampling, the message-passing is particularly simple: each node learns the current instantiation of its Markov blanket. With enough samples the node can estimate the distribution over its Markov blanket and (roughly speaking) determine its own statistics. The advantage of this scheme is that in the limit of very many samples, it is guaranteed to converge to the correct statistics. The disadvantage is that very many samples may be required.

The message-passing in variational methods is quite different. Its purpose is to couple the variational parameters of one node to those of its Markov blanket. The messages do not come in the form of samples, but rather in the form of approximate statistics (as summarized by the variational parameters). For example, in a network of binary nodes, while the Gibbs sampler is circulating messages of binary vectors that correspond to the *instantiations* of Markov blankets, the variational methods are circulating real-valued numbers that correspond to the *statistics* of Markov blankets. This may be one reason why variational methods often converge faster than Gibbs sampling. Of course, the disadvantage of these schemes is that they do not necessarily converge to the correct statistics. On the other hand, they can provide bounds on marginal probabilities that are quite difficult to estimate by sampling. Indeed, sampling-based methods—while well-suited to estimating the statistics of individual hidden nodes—are ill-equipped to compute marginal probabilities such as $P(E) = \sum_H P(H, E)$.

An interesting direction for future research is to consider combinations of sampling methods and variational methods. Some initial work in this direction has been done by Hinton, Sallans, and Ghahramani (this volume), who discuss brief Gibbs sampling from the point of view of variational approximation.

7.1.3 Bayesian methods

Variational inference can be applied to the general problem of Bayesian parameter estimation. Indeed we can quite generally treat parameters as additional nodes in a graphical

[10] The particular recognition model utilized in the Helmholtz machine is a layered graph, which makes weak conditional independence assumptions and thus makes it possible, in principle, to capture fairly general dependencies.

model (cf. Heckerman, this volume) and thereby treat Bayesian inference on the same footing as generic probabilistic inference in a graphical model. This probabilistic inference problem is often intractable, and variational approximations can be useful.

A variational method known as "ensemble learning" was originally introduced as a way of fitting an "ensemble" of neural networks to data, where each setting of the parameters can be thought of as a different member of the ensemble (Hinton & van Camp, 1993). Let $Q(\theta|E)$ represent a variational approximation to the posterior distribution $P(\theta|E)$. The ensemble is fit by minimizing the appropriate KL divergence:

$$KL(Q\|P) = \int Q(\theta|E) \ln \frac{Q(\theta|E)}{P(\theta|E)} d\theta. \tag{73}$$

Following the same line of argument as in Section 6, we know that this minimization must be equivalent to the maximization of a lower bound. In particular, copying the argument from Section 6, we find that minimizing the KL divergence yields the best lower bound on the following quantity:

$$\ln P(E) = \ln \int P(E|\theta)P(\theta)d\theta, \tag{74}$$

which is the logarithm of the *marginal likelihood*; a key quantity in Bayesian model selection and model averaging.

More recently, the ensemble learning approach has been applied to mixture of experts architectures (Waterhouse, et al, 1996) and hidden Markov models (MacKay, 1997a). One interesting aspect of these applications is that they do not assume any particular parametric family for Q, just that Q factorizes in a specific way. The variational minimization itself determines the best family given this factorization and the prior on θ. In related work, MacKay (1997b) has described a connection between variational inference and Type II maximum likelihood inference.

Jaakkola and Jordan (1997b) have also developed variational methods for Bayesian inference, using a variational approach to find an analytically tractable approximation for logistic regression with a Gaussian prior on the parameters.

7.1.4 Perspective and prospectives

Perhaps the key issue that faces developers of variational methods is the issue of approximation accuracy. At the current state of development of variational methods for graphical models, we have little theoretical insight into conditions under which variational methods can be expected to be accurate and conditions under which they might be expected to be inaccurate. Moreover, there is little understanding of how to match variational transformations to architectures.

One can develop an intuition for when variational methods work by examining their properties in certain well-studied cases. For mean field methods, a good starting point is to understand the examples in the statistical mechanics literature where this approximation gives not only good, but indeed exact, results. These are densely connected graphs with uniformly weak (but non-negative) couplings between neighboring nodes (Parisi, 1988). The

mean field equations for these networks have a unique solution that determines the statistics of individual nodes in the limit of very large graphs.

In more general graphical models, of course, the conditions for a mean field approximation may not be so favorable. Typically, this can be diagnosed by the presence of multiple solutions to the mean field equations. Roughly speaking, one can interpret each solution as corresponding to a mode of the posterior distribution; thus, multiple solutions indicate a multimodal posterior distribution. The simplest mean field approximations, in particular those that utilize a completely factorized approximating distribution, are poorly designed for such situations. However, they can succeed rather well in applications where the joint distribution $P(H, E)$ is multimodal, but the posterior distribution $P(H|E)$ is not. It is worth emphasizing this distinction between joint and posterior distributions. This is what allows simple variational methods—which make rather strong assumptions of conditional independence—to be used in the learning of non-trivial graphical models.

A second key issue has to do with broadening the scope of variational methods. In this paper we have presented a restricted set of variational techniques, those based on convexity transformations. For these techniques to be applicable the appropriate convexity properties need to be identified. While it is relatively easy to characterize small classes of models where these properties lead to simple approximation algorithms, such as the case in which the local conditional probabilities are log-concave generalized linear models, it is not generally easy to develop variational algorithms for other kinds of graphical models. A broader characterization of variational approximations is needed and a more systematic algebra is needed to match the approximations to models.

Other open problems include: (1) the problem of combining variational methods with sampling methods and with search based methods, (2) the problem of making more optimal choices of node ordering in the case of sequential methods, (3) the development of upper bounds within the block framework, (4) the combination of multiple variational approximations for the same model, and (5) the development of variational methods for architectures that combine continuous and discrete random variables.

Similar open problems exist for sampling methods and for methods based on incomplete or pruned versions of exact methods. The difficulty in providing solid theoretical foundations in all of these cases lies in the fact that accuracy is contingent to a large degree on the actual conditional probability values of the underlying probability model rather than on the discrete properties of the graph.

8 Acknowledgments

We wish to thank Brendan Frey, David Heckerman, Uffe Kjærulff, and (as always) Peter Dayan for helpful comments on the manuscript.

References

[1] Bathe, K. J. (1996). *Finite Element Procedures.* Englewood Cliffs, NJ: Prentice-Hall.

[2] Baum, L.E., Petrie, T., Soules, G., & Weiss, N. (1970). A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains. *The Annals of Mathematical Statistics, 41*, 164–171.

[3] Cover, T., & Thomas, J. (1991). *Elements of Information Theory*. New York: John Wiley.

[4] Cowell, R. (in press). Introduction to inference for Bayesian networks. In M. I. Jordan (Ed.), *Learning in Graphical Models*. Norwell, MA: Kluwer Academic Publishers.

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

[6] Dayan, P., Hinton, G. E., Neal, R., & Zemel, R. S. (1995). The Helmholtz Machine. *Neural Computation*, *7*, 889–904.

[7] Dean, T., & Kanazawa, K. (1989). A model for reasoning about causality and persistence. *Computational Intelligence*, *5*, 142–150.

[8] Dechter, R. (in press). Bucket elimination: A unifying framework for probabilistic inference. In M. I. Jordan (Ed.), *Learning in Graphical Models*. Norwell, MA: Kluwer Academic Publishers.

[9] Dempster, A.P., Laird, N.M., & Rubin, D.B. (1977). Maximum-likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, B39*, 1-38.

[10] Draper, D. L., & Hanks, S. (1994). Localized partial evaluation of belief networks. *Uncertainty and Artificial Intelligence: Proceedings of the Tenth Conference*. San Mateo, CA: Morgan Kaufmann.

[11] Frey, B. Hinton, G. E., Dayan, P. (1996). Does the wake-sleep algorithm learn good density estimators? In D. S. Touretzky, M. C. Mozer, & M. E. Hasselmo (Eds.), *Advances in Neural Information Processing Systems 8*. Cambridge, MA: MIT Press.

[12] Fung, R. & Favero, B. D. (1994). Backward simulation in Bayesian networks. *Uncertainty and Artificial Intelligence: Proceedings of the Tenth Conference*. San Mateo, CA: Morgan Kaufmann.

[13] Galland, C. (1993). The limitations of deterministic Boltzmann machine learning. *Network*, *4*, 355–379.

[14] Ghahramani, Z., & Hinton, G. E. (1996). Switching state-space models. University of Toronto Technical Report CRG-TR-96-3, Department of Computer Science.

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

[16] Gilks, W., Thomas, A., & Spiegelhalter, D. (1994). A language and a program for complex Bayesian modelling. *The Statistician*, *43*, 169–178.

[17] Heckerman, D. (in press). A tutorial on learning with Bayesian networks. In M. I. Jordan (Ed.), *Learning in Graphical Models*. Norwell, MA: Kluwer Academic Publishers.

[18] Henrion, M. (1991). Search-based methods to bound diagnostic probabilities in very large belief nets. *Uncertainty and Artificial Intelligence: Proceedings of the Seventh Conference*. San Mateo, CA: Morgan Kaufmann.

[19] Hinton, G. E., & Sejnowski, T. (1986). Learning and relearning in Boltzmann machines. In D. E. Rumelhart & J. L. McClelland, (Eds.), *Parallel distributed processing*: *Volume 1*, Cambridge, MA: MIT Press.

[20] Hinton, G.E. & van Camp, D. (1993). Keeping neural networks simple by minimizing the description length of the weights. In *Proceedings of the 6th Annual Workshop on Computational Learning Theory*, pp 5-13. New York, NY: ACM Press.

[21] Hinton, G. E., Dayan, P., Frey, B., and Neal, R. M. (1995). The wake-sleep algorithm for unsupervised neural networks. *Science*, 268:1158–1161.

[22] Hinton, G. E., Sallans, B., & Ghahramani, Z. (in press). A hierarchical community of experts. In M. I. Jordan (Ed.), *Learning in Graphical Models*. Norwell, MA: Kluwer Academic Publishers.

[23] Horvitz, E. J., Suermondt, H. J., & Cooper, G.F. (1989). Bounded conditioning: Flexible inference for decisions under scarce resources. *Conference on Uncertainty in Artificial Intelligence: Proceedings of the Fifth Conference*. Mountain View, CA: Association for UAI.

[24] Jaakkola, T. S., & Jordan, M. I. (1996). Computing upper and lower bounds on likelihoods in intractable networks. *Uncertainty and Artificial Intelligence: Proceedings of the Twelth Conference*. San Mateo, CA: Morgan Kaufmann.

[25] Jaakkola, T. S. (1997). *Variational methods for inference and estimation in graphical models*. Unpublished doctoral dissertation, Massachusetts Institute of Technology.

[26] Jaakkola, T. S., & Jordan, M. I. (1997a). Recursive algorithms for approximating probabilities in graphical models. In M. C. Mozer, M. I. Jordan, & T. Petsche (Eds.), *Advances in Neural Information Processing Systems 9*. Cambridge, MA: MIT Press.

[27] Jaakkola, T. S., & Jordan, M. I. (1997b). Bayesian logistic regression: a variational approach. In D. Madigan & P. Smyth (Eds.), Proceedings of the 1997 Conference on Artificial Intelligence and Statistics, Ft. Lauderdale, FL.

[28] Jaakkola, T. S., & Jordan. M. I. (1997c). Variational methods and the QMR-DT database. Submitted to: *Journal of Artificial Intelligence Research*.

[29] Jaakkola, T. S., & Jordan. M. I. (in press). Improving the mean field approximation via the use of mixture distributions. In M. I. Jordan (Ed.), *Learning in Graphical Models*. Norwell, MA: Kluwer Academic Publishers.

[30] Jensen, C. S., Kong, A., & Kjærulff, U. (1995). Blocking-Gibbs sampling in very large probabilistic expert systems. *International Journal of Human-Computer Studies*, *42*, 647–666.

[31] Jensen, F. V., & Jensen, F. (1994). Optimal junction trees. *Uncertainty and Artificial Intelligence: Proceedings of the Tenth Conference*. San Mateo, CA: Morgan Kaufmann.

[32] Jensen, F. V. (1996). *An Introduction to Bayesian Networks*. London: UCL Press.

[33] Jordan, M. I. (1994). A statistical approach to decision tree modeling. In M. Warmuth (Ed.), *Proceedings of the Seventh Annual ACM Conference on Computational Learning Theory*. New York: ACM Press.

[34] Jordan, M. I., Ghahramani, Z., & Saul, L. K. (1997). Hidden Markov decision trees. In M. C. Mozer, M. I. Jordan, & T. Petsche (Eds.), *Advances in Neural Information Processing Systems 9*. Cambridge, MA: MIT Press.

[35] Kanazawa, K., Koller, D., & Russell, S. (1995). Stochastic simulation algorithms for dynamic probabilistic networks. *Uncertainty and Artificial Intelligence: Proceedings of the Eleventh Conference*. San Mateo, CA: Morgan Kaufmann.

[36] Kjærulff, U. (1990). Triangulation of graphs—algorithms giving small total state space. Research Report R-90-09, Department of Mathematics and Computer Science, Aalborg University, Denmark.

[37] Kjærulff, U. (1994). Reduction of computational complexity in Bayesian networks through removal of weak dependences. *Uncertainty and Artificial Intelligence: Proceedings of the Tenth Conference*. San Mateo, CA: Morgan Kaufmann.

[38] MacKay, D.J.C. (1997a). Ensemble learning for hidden Markov models. Unpublished manuscript. Department of Physics, University of Cambridge.

[39] MacKay, D.J.C. (1997b). Comparison of approximate methods for handling hyperparameters. Submitted to *Neural Computation*.

[40] MacKay, D.J.C. (1997b). Introduction to Monte Carlo methods. In M. I. Jordan (Ed.), *Learning in Graphical Models*. Norwell, MA: Kluwer Academic Publishers.

[41] McEliece, R.J., MacKay, D.J.C., & Cheng, J.-F. (1996) Turbo decoding as an instance of Pearl's "belief propagation algorithm." Submitted to: *IEEE Journal on Selected Areas in Communication*.

[42] Merz, C. J., & Murphy, P. M. (1996). *UCI repository of machine learning databases*. [`http:/www.ics.uci/~mlearn/MLRepository.html`]. Irvine, CA: University of California, Department of Information and Computer Science.

[43] Neal, R. (1992). Connectionist learning of belief networks, *Artificial Intelligence*, *56*, 71-113.

[44] Neal, R. (1993). Probabilistic inference using Markov chain Monte Carlo methods. University of Toronto Technical Report CRG-TR-93-1, Department of Computer Science.

[45] Neal, R., & Hinton, G. E. (in press). A view of the EM algorithm that justifies incremental, sparse, and other variants. In M. I. Jordan (Ed.), *Learning in Graphical Models*. Norwell, MA: Kluwer Academic Publishers.

[46] Parisi, G. (1988). *Statistical Field Theory*. Redwood City, CA: Addison-Wesley.

[47] Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, San Mateo, CA: Morgan Kaufmannn.

[48] Peterson, C., & Anderson, J. R. (1987). A mean field theory learning algorithm for neural networks. *Complex Systems*, *1*, 995–1019.

[49] Rockafellar, R. (1972). *Convex Analysis*. Princeton University Press.

[50] Rustagi, J. (1976). *Variational Methods in Statistics*. New York: Academic Press.

[51] Sakurai, J. (1985). *Modern Quantum Mechanics*. Redwood City, CA: Addison-Wesley.

[52] Saul, L. K., & Jordan, M. I. (1994). Learning in Boltzmann trees. *Neural Computation*, *6*, 1173–1183.

[53] Saul, L. K., Jaakkola, T. S., & Jordan, M. I. (1996). Mean field theory for sigmoid belief networks. *Journal of Artificial Intelligence Research*, *4*, 61–76.

[54] Saul, L. K., & Jordan, M. I. (1996). Exploiting tractable substructures in intractable networks. In D. S. Touretzky, M. C. Mozer, & M. E. Hasselmo (Eds.), *Advances in Neural Information Processing Systems 8*. Cambridge, MA: MIT Press.

[55] Saul, L. K., & Jordan, M. I. (in press). A mean field learning algorithm for unsupervised neural networks. In M. I. Jordan (Ed.), *Learning in Graphical Models*. Norwell, MA: Kluwer Academic Publishers.

[56] Seung, S. (1995). Annealed theories of learning. In J.-H Oh, C. Kwon, and S. Cho, (Eds.), *Neural Networks: The Statistical Mechanics Perspectives*. Singapore: World Scientific.

[57] Shachter, R. D., Andersen, S. K., & Szolovits, P. (1994). Global conditioning for probabilistic inference in belief networks. *Uncertainty and Artificial Intelligence: Proceedings of the Tenth Conference*. San Mateo, CA: Morgan Kaufmann.

[58] Shenoy, P. P. (1992). Valuation-based systems for Bayesian decision analysis. *Operations Research, 40*, 463–484.

[59] Shwe, M. A., Middleton, B., Heckerman, D. E., Henrion, M., Horvitz, E. J., Lehmann, H. P., & Cooper, G. F. (1991). Probabilistic diagnosis using a reformulation of the INTERNIST-1/QMR knowledge base. *Meth. Inform. Med.*, *30*, 241-255.

[60] Smyth, P., Heckerman, D., & Jordan, M. I. (1997). Probabilistic independence networks for hidden Markov probability models. *Neural Computation*, *9*, 227–270.

[61] Waterhouse, S., MacKay, D.J.C. & Robinson, T. (1996). Bayesian methods for mixtures of experts. In D. S. Touretzky, M. C. Mozer, & M. E. Hasselmo (Eds.), *Advances in Neural Information Processing Systems 8*. Cambridge, MA: MIT Press.

[62] Williams, C. K. I., & Hinton, G. E. (1991). Mean field networks that learn to discriminate temporally distorted strings. In Touretzky, D. S., Elman, J., Sejnowski, T., & Hinton, G. E., (Eds.), *Proceedings of the 1990 Connectionist Models Summer School*. San Mateo, CA: Morgan Kaufmann.

9 Appendix

In this section, we calculate the conjugate functions for the logarithm function and the log logistic function.

For $f(x) = \ln x$, we have:

$$f^*(\lambda) = \min_x \{\lambda x - \ln x\}. \tag{75}$$

Taking the derivative with respect to x and setting to zero yields $x = \lambda^{-1}$. Substituting back in Eq. (75) yields:

$$f^*(\lambda) = \ln \lambda + 1, \tag{76}$$

which justifies the representation of the logarithm given in Eq. (14).

For the log logistic function $g(x) = -\ln(1 + e^{-x})$, we have:

$$g^*(\lambda) = \min_x \{\lambda x + \ln(1 + e^{-x})\}. \tag{77}$$

Taking the derivative with respect to x and setting to zero yields:

$$\lambda = \frac{e^{-x}}{1 + e^{-x}}, \tag{78}$$

from which we obtain:

$$x = \ln \frac{1 - \lambda}{\lambda} \tag{79}$$

and

$$\ln(1 + e^{-x}) = \frac{1}{1 - \lambda}. \tag{80}$$

Plugging these expressions back into Eq. (77) yields:

$$f^*(\lambda) = -\lambda \ln \lambda - (1 - \lambda) \ln(1 - \lambda), \tag{81}$$

which is the binary entropy function $H(\lambda)$. This justifies the representation of the logistic function given in Eq. (19).