Journal of Machine Learning Research 5 (2004) ?-? Submitted 2/04; Published ?/04

Variational Message Passing

John Winn JWINN@MICROSOFT.COM
and
Christopher M. Bishop CMBISHOP@MICROSOFT.COM
Microsoft Research Cambridge
Roger Needham Building
7 J. J. Thomson Avenue
Cambridge CB3 0FB, U.K

Editor: ??

Abstract

This paper presents Variational Message Passing (VMP), a general purpose algorithm for applying variational inference to a Bayesian Network. Like belief propagation, Variational Message Passing proceeds by passing messages between nodes in the graph and updating posterior beliefs using local operations at each node. Each such update increases a lower bound on the log evidence (unless already at a local maximum). In contrast to belief propagation, VMP can be applied to a very general class of conjugate-exponential models because it uses a factorised variational approximation. Furthermore, by introducing additional variational parameters, VMP can be applied to models containing non-conjugate distributions. The VMP framework also allows the lower bound to be evaluated, and this can be used both for model comparison and for detection of convergence. Variational Message Passing has been implemented in the form of a general purpose inference engine called VIBES ('Variational Inference for BayEsian networkS') which allows models to be specified graphically and then solved variationally without recourse to coding.

Keywords: Bayesian Networks, Variational methods

1. Introduction

Variational inference methods (Neal and Hinton, 1998, Jordan et al., 1998) have been used successfully for a wide range of models, and new applications are constantly being explored. In each previous application, the equations for optimising the variational approximation have been worked out by hand, a process which is both time consuming and error prone.

For several other inference methods, general purpose algorithms have been developed which can be applied to large classes of probabilistic models. For example, belief propagation can be applied to any acyclic discrete network (Pearl, 1986) or mixed-Gaussian network (Lauritzen, 1992), and the algorithm used in Thomas et al. (1992) can perform Gibbs sampling in almost any Bayesian network. Each of these algorithms relies on being able to decompose the required computation into calculations that are local to each node in the graph and which require only messages passed along the edges connected to that node.

In this paper, the Variational Message Passing algorithm is developed, which optimises a variational approximation using a set of local computations for each node together with a mechanism for passing messages between them. Hence, VMP allows for variational inference

to be applied automatically to a very large class of Bayesian networks. In VMP, the messages are exponential family distributions, summarised by either their natural parameter vector (for child-to-parent messages) or a vector of moments (for parent-to-child messages). These messages are defined so that the optimal variational distribution for a node can be found by summing the messages from its children together with a function of the messages from its parents, where this function depends on the conditional distribution for the node.

The paper is organised as follows. Section 2 gives a brief review of variational inference methods. Section 3 contains the derivation of the Variational Message Passing algorithm, along with an example of its use. In Section 4 the class of models which can be handled by the algorithm is defined, while Section 5 describes an implementation of the algorithm, called VIBES (Variational Inference in BayEsian networkS). Some extensions to the algorithm are given in Section 6, and Section 7 concludes with an overall discussion and suggestions for future research directions.

2. Variational Inference

In this section, variational inference will be reviewed briefly with particular focus on the case where the variational distribution has a factorised form. The random variables in the model will be denoted by $\mathbf{X} = (\mathbf{V}, \mathbf{H})$ where $\mathbf{V}$ are the visible (observed) variables and $\mathbf{H}$ are the hidden (latent) variables. We assume that the model has the form of a Bayesian network and so the joint distribution $P(\mathbf{X})$ can be expressed in terms of the conditional distributions at each node i,

$$P(\mathbf{X}) = \prod_i P(X_i \,|\, \mathrm{pa}_i) \tag{1}$$

where pa_i denotes the set of variables corresponding to the parents of node i and X_i denotes the variable or group of variables associated with node i.

Ideally, we would like to perform exact inference within this model to find posterior marginal distributions over individual latent variables. Unfortunately, exact inference algorithms, such as the junction tree algorithm (Cowell et al., 1999), are typically only applied to discrete or linear-Gaussian models and are computationally intractable for all but the simplest models. Instead, we must turn to approximate inference methods and, in particular, the deterministic approximation method of variational inference.

The goal in variational inference is to find a tractable variational distribution $Q(\mathbf{H})$ that closely approximates the true posterior distribution $P(\mathbf{H} \,|\, \mathbf{V})$. To do this we note the following decomposition of the log marginal probability of the observed data, which holds for any choice of distribution $Q(\mathbf{H})$

$$\ln P(\mathbf{V}) = \mathcal{L}(Q) + \mathrm{KL}(Q \,||\, P) \tag{2}$$

where

$$\begin{aligned} \mathcal{L}(Q) &= \sum_{\mathbf{H}} Q(\mathbf{H}) \ln \frac{P(\mathbf{H}, \mathbf{V})}{Q(\mathbf{H})} \\ \mathrm{KL}(Q \,||\, P) &= -\sum_{\mathbf{H}} Q(\mathbf{H}) \ln \frac{P(\mathbf{H} \,|\, \mathbf{V})}{Q(\mathbf{H})} \end{aligned} \tag{3}$$

and the sums are replaced by integrals in the case of continuous variables. Here $\mathrm{KL}(Q \,||\, P)$ is the Kullback-Leibler divergence between the true posterior $P(\mathbf{H} \,|\, \mathbf{V})$ and the variational

approximation $Q(\mathbf{H})$. Since this satisfies $\mathrm{KL}(Q \,||\, P) \geqslant 0$ it follows from (2) that the quantity $\mathcal{L}(Q)$ forms a lower bound on $\ln P(\mathbf{V})$.

We now choose some family of distributions to represent $Q(\mathbf{H})$ and then seek a member of that family that maximises the lower bound $\mathcal{L}(Q)$. If we allow $Q(\mathbf{H})$ to have complete flexibility then we see that the maximum of the lower bound occurs when the Kullback-Leibler divergence is zero. In this case, the variational posterior distribution equals the true posterior and $\mathcal{L}(Q) = \ln P(\mathbf{V})$. However, working with the true posterior distribution is computationally intractable (otherwise we wouldn't be resorting to variational methods). We must therefore consider a more restricted family of Q distributions which has the property that the lower bound (3) can be evaluated and optimised efficiently and yet which is still sufficiently flexible as to give a good approximation to the true posterior distribution.

2.1 Factorised Variational Distributions

We wish to choose a variational distribution $Q(\mathbf{H})$ with a simpler structure than that of the model, so as to make calculation of the lower bound $\mathcal{L}(Q)$ tractable. One way to simplify the dependency structure is by choosing a variational distribution where disjoint groups of variables are independent. This is equivalent to choosing Q to have a factorised form

$$Q(\mathbf{H}) = \prod_i Q_i(\mathbf{H}_i), \tag{4}$$

where $\{\mathbf{H}_i\}$ are the disjoint groups of variables. This approximation has been successfully used in many applications of variational methods (Attias, 2000, Ghahramani and Beal, 2001, Bishop, 1999). Substituting (4) into (3) gives

$$\mathcal{L}(Q) = \sum_{\mathbf{H}} \prod_i Q_i(\mathbf{H}_i) \ln P(\mathbf{H}, \mathbf{V}) - \sum_i \sum_{\mathbf{H}_i} Q_i(\mathbf{H}_i) \ln Q_i(\mathbf{H}_i).$$

We now separate out all terms in one factor Q_j,

$$\begin{aligned} \mathcal{L}(Q) &= \sum_{\mathbf{H}_j} Q_j(\mathbf{H}_j) \langle \ln P(\mathbf{H}, \mathbf{V}) \rangle_{\sim Q_j(\mathbf{H}_j)} + \mathbb{H}(Q_j) + \sum_{i \neq j} \mathbb{H}(Q_i) \\ &= -\mathrm{KL}(Q_j \,||\, Q_j^\star) + \text{terms not in } Q_j \end{aligned} \tag{5}$$

where $\mathbb{H}$ denotes entropy and we have introduced a new distribution $Q_j^\star$, defined by

$$\ln Q_j^\star(\mathbf{H}_j) = \langle \ln P(\mathbf{H}, \mathbf{V}) \rangle_{\sim Q(\mathbf{H}_j)} + \text{const.} \tag{6}$$

and $\langle \cdot \rangle_{\sim Q(\mathbf{H}_j)}$ denotes an expectation with respect to all factors except $Q_j(\mathbf{H}_j)$. The bound is maximised with respect to Q_j when the KL divergence in (5) is zero, which occurs when $Q_j = Q_j^\star$. Therefore, we can maximise the bound by setting Q_j equal to $Q_j^\star$. Taking exponentials of both sides we obtain

$$Q_j^\star(\mathbf{H}_j) = \frac{1}{Z} \exp \langle \ln P(\mathbf{H}, \mathbf{V}) \rangle_{\sim Q(\mathbf{H}_j)}, \tag{7}$$

where Z is the normalisation factor needed to make $Q_j^\star$ a valid probability distribution. Note that the equations for all of the factors are coupled since the solution for each $Q_j(\mathbf{H}_j)$ depends on expectations with respect to the other factors $Q_{i \neq j}$. The variational optimisation proceeds by initialising each of the $Q_j(\mathbf{H}_j)$ and then cycling through each factor in turn replacing the current distribution with a revised estimate given by (7).

3. Variational Message Passing

In this section, the Variational Message Passing algorithm will be derived and shown to optimise a factorised variational distribution using a message passing procedure on a graphical model. For the initial derivation, it will be assumed that the variational distribution is factorised with respect to each hidden variable H_i and so can be written

$$Q(\mathbf{H}) = \prod_i Q_i(H_i).$$

From (6), the optimised form of the jth factor is given by

$$\ln Q_j^\star(H_j) = \langle \ln P(\mathbf{H}, \mathbf{V}) \rangle_{\sim Q(H_j)} + \text{const.}$$

We now substitute in the form of the joint probability distribution of a Bayesian network, as given in (1),

$$\ln Q_j^\star(H_j) = \left\langle \sum\nolimits_i \ln P(X_i \,|\, \mathrm{pa}_i) \right\rangle_{\sim Q(H_j)} + \text{const.}$$

Any terms in the sum over i that do not depend on H_j will be constant under the expectation and can be subsumed into the constant term. This leaves only the conditional $P(H_j \,|\, \mathrm{pa}_j)$ together with the conditionals for all the children of H_j, as these have H_j in their parent set,

$$\ln Q_j^\star(H_j) = \langle \ln P(H_j \,|\, \mathrm{pa}_j) \rangle_{\sim Q(H_j)} + \sum\nolimits_{k \in \mathrm{ch}_j} \langle \ln P(X_k \,|\, \mathrm{pa}_k) \rangle_{\sim Q(H_j)} + \text{const.} \qquad (8)$$

where ch_j are the children of node j in the graph. Thus, the expectations required to evaluate $Q_j^\star$ involve only those variables lying in the Markov blanket of H_j, consisting of its parents, children and co-parents[1] $\mathrm{cp}_k^{(j)}$. This is illustrated in the form of a directed graphical model in Figure 1. Note that we use the notation X_k to denote both a random variable and the corresponding node in the graph. The optimisation of Q_j can therefore be expressed as a local computation at the node H_j. This computation involves the sum of a term involving the parent nodes, along with one term from each of the child nodes. These terms can be thought of as 'messages' from the corresponding nodes. Hence, we can decompose the overall optimisation into a set of local computations that depend only on messages from neighbouring (i.e. parent and child) nodes in the graph.

3.1 Conjugate-Exponential Models

The exact form of the messages in (8) will depend on the functional form of the conditional distributions in the model. It has been noted (Attias, 2000, Ghahramani and Beal, 2001) that important simplifications to the variational update equations occur when the distributions of variables, conditioned on their parents, are drawn from the exponential family and are conjugate[2] with respect to the distributions over these parent variables. A model where both of these constraints hold is known as a *conjugate-exponential* model.

1. The co-parents of a node X are all the nodes with at least one child which is also a child of X (excluding X itself).
2. A parent distribution $P(X \,|\, Y)$ is said to be *conjugate* to a child distribution $P(W \,|\, X)$ if $P(X \,|\, Y)$ has the same functional form, with respect to X, as $P(W \,|\, X)$.

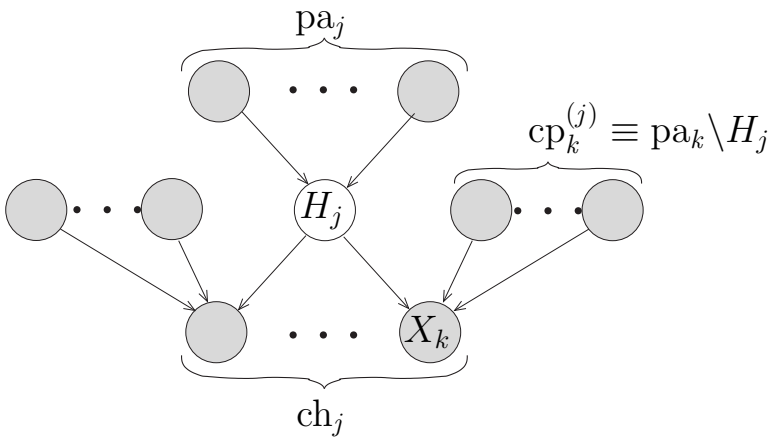

Figure 1: A key observation is that the variational update equation for a node H_j depends only on expectations over variables in the Markov blanket of that node (shown shaded), defined as the set of parents, children and co-parents of that node.

A conditional distribution is in the exponential family if it can be written in the form

$$P(\mathbf{X} \,|\, \mathbf{Y}) = \exp[\boldsymbol{\phi}(\mathbf{Y})^{\mathrm{T}} \mathbf{u}(\mathbf{X}) + f(\mathbf{X}) + g(\mathbf{Y})] \tag{9}$$

where $\boldsymbol{\phi}(\mathbf{Y})$ is called the *natural parameter* vector and $\mathbf{u}(\mathbf{X})$ is called the *natural statistic* vector. The term $g(\mathbf{Y})$ acts as a normalisation function that ensures the distribution integrates to unity for any given setting of the parameters $\mathbf{Y}$. The advantages of exponential family distributions are that their logarithms are tractable to compute and their state can be summarised completely by the natural parameter vector. The use of conjugate distributions means that the posterior for each factor has the same form as the prior and so learning changes only the values of the parameters, rather than the functional form of the distribution.

If we know the natural parameter vector $\boldsymbol{\phi}(\mathbf{Y})$ for an exponential family distribution, then we can find the expectation of the natural statistic vector with respect to the distribution. Rewriting (9) and defining $\widetilde{g}$ as a reparameterisation of g in terms of $\boldsymbol{\phi}$ gives,

$$P(\mathbf{X} \,|\, \boldsymbol{\phi}) = \exp[\boldsymbol{\phi}^{\mathrm{T}} \mathbf{u}(\mathbf{X}) + f(\mathbf{X}) + \widetilde{g}(\boldsymbol{\phi})].$$

We integrate with respect to $\mathbf{X}$,

$$\int_{\mathbf{X}} \exp[\boldsymbol{\phi}^{\mathrm{T}} \mathbf{u}(\mathbf{X}) + f(\mathbf{X}) + \widetilde{g}(\boldsymbol{\phi})] \, d\mathbf{X} = \int_{\mathbf{X}} P(\mathbf{X} \,|\, \boldsymbol{\phi}) \, d\mathbf{X} \;=\; 1$$

and then differentiate with respect to $\boldsymbol{\phi}$

$$\begin{aligned} \int_{\mathbf{X}} \frac{d}{d\boldsymbol{\phi}} \exp[\boldsymbol{\phi}^{\mathrm{T}} \mathbf{u}(\mathbf{X}) + f(\mathbf{X}) + \widetilde{g}(\boldsymbol{\phi})] \, d\mathbf{X} &= \frac{d}{d\boldsymbol{\phi}}(1) \;=\; 0 \\ \int_{\mathbf{X}} P(\mathbf{X} \,|\, \boldsymbol{\phi}) \left[\mathbf{u}(\mathbf{X}) + \frac{d\widetilde{g}(\boldsymbol{\phi})}{d\boldsymbol{\phi}} \right] d\mathbf{X} &= 0 \end{aligned}$$

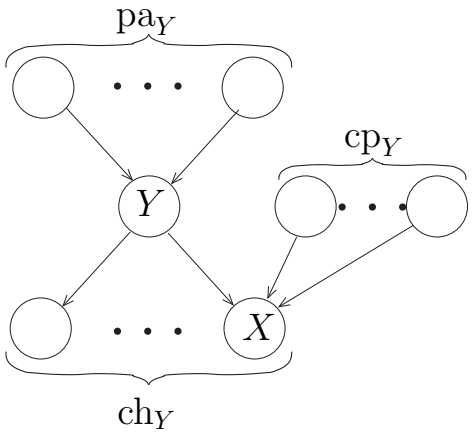

Figure 2: Part of a graphical model showing a node Y, the parents and children of Y, and the co-parents of Y with respect to a child node X.

and so the expectation of the natural statistic vector is given by

$$\langle \mathbf{u}(\mathbf{X}) \rangle_{P(\mathbf{X} \,|\, \boldsymbol{\phi})} \;=\; -\frac{d\widetilde{g}(\boldsymbol{\phi})}{d\boldsymbol{\phi}}. \tag{10}$$

3.2 Optimisation of Q in Conjugate-Exponential Models

We will now demonstrate how the optimisation of the variational distribution can be carried out, given that the model is conjugate-exponential. We consider the general case of optimising a factor $Q(Y)$ corresponding to a node Y, whose children include X, as illustrated in Figure 2. From (9), the log conditional probability of the variable Y given its parents can be written

$$\ln P(Y \,|\, \mathrm{pa}_Y) = \boldsymbol{\phi}_Y(\mathrm{pa}_Y)^{\mathrm{T}} \mathbf{u}_Y(Y) + f_Y(Y) + g_Y(\mathrm{pa}_Y). \tag{11}$$

The subscript Y on each of the functions $\boldsymbol{\phi}_Y, \mathbf{u}_Y, f_Y, g_Y$ is required as these functions differ for different members of the exponential family and so need to be defined separately for each node.

Consider a node $X \in \mathrm{ch}_Y$ which is a child of Y. The conditional probability of X given its parents will also be in the exponential family and so can be written in the form

$$\ln P(X \,|\, Y, \mathrm{cp}_Y) = \boldsymbol{\phi}_X(Y, \mathrm{cp}_Y)^{\mathrm{T}} \mathbf{u}_X(X) + f_X(X) + g_X(Y, \mathrm{cp}_Y) \tag{12}$$

where cp_Y are the co-parents of Y with respect to X, in other words, the set of parents of X excluding Y itself. The quantity $P(Y \,|\, \mathrm{pa}_Y)$ in(11) can be thought of as a prior over Y, and $P(X \,|\, Y, \mathrm{cp}_Y)$ as a (contribution to) the likelihood of Y.

For example, if X is Gaussian distributed with mean Y and precision β, it follows that the co-parent set cp_Y contains only β, and the log conditional for X is

$$\ln P(X \,|\, Y, \beta) = \begin{bmatrix} \beta Y \\ -\beta/2 \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} X \\ X^2 \end{bmatrix} + \tfrac{1}{2}(\ln \beta - \beta Y^2 - \ln 2\pi). \tag{13}$$

Conjugacy requires that the conditionals of (11) and (12) have the same functional form with respect to Y, and so the latter can be rewritten in terms of $\mathbf{u}_Y(Y)$ by defining functions $\boldsymbol{\phi}_{XY}$ and λ as follows

$$\ln P(X \,|\, Y, \mathrm{cp}_Y) = \boldsymbol{\phi}_{XY}(X, \mathrm{cp}_Y)^{\mathrm{T}} \mathbf{u}_Y(Y) + \lambda(X, \mathrm{cp}_Y). \tag{14}$$

It may appear from this expression that the function $\boldsymbol{\phi}_{XY}$ depends on the form of the parent conditional $P(Y \,|\, \mathrm{pa}_Y)$ and so cannot be determined locally at X. This is not the case, because the conjugacy constraint dictates $\mathbf{u}_Y(Y)$ for any parent Y of X, implying that $\boldsymbol{\phi}_{XY}$ can be found directly from the form of the conditional $P(X \,|\, \mathrm{pa}_k)$.

Continuing the above example, we can find $\boldsymbol{\phi}_{XY}$ by rewriting the log conditional in terms of Y to give

$$\ln P(X \,|\, Y, \beta) = \begin{bmatrix} \beta X \\ -\beta/2 \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} Y \\ Y^2 \end{bmatrix} + \tfrac{1}{2}(\ln \beta - \beta X^2 - \ln 2\pi),$$

which lets us define $\boldsymbol{\phi}_{XY}$ and dictate what $\mathbf{u}_Y(Y)$ must be to enforce conjugacy,

$$\boldsymbol{\phi}_{XY}(X, \beta) \stackrel{\mathrm{def}}{=} \begin{bmatrix} \beta X \\ -\beta/2 \end{bmatrix}, \qquad \mathbf{u}_Y(Y) = \begin{bmatrix} Y \\ Y^2 \end{bmatrix}. \tag{15}$$

In order to compute the variational update for Y, we need to be able to find the expectations of the log conditionals (11) and (12) with respect to all factors but $Q_Y(Y)$. The expectation of the natural statistic vector $\mathbf{u}$ under any exponential family distribution can be found from the natural parameter vector of that distribution using (10). The consequence of this is that, for any variable A, we can find $\langle \mathbf{u}_A(A) \rangle_Q$. In the case where A is observed, the expectation is irrelevant and we can simply calculate $\mathbf{u}_A(A)$ directly.

From (12) and (14), it can be seen that $\ln P(X \,|\, Y, \mathrm{cp}_Y)$ is linear in $\mathbf{u}_X(X)$ and $\mathbf{u}_Y(Y)$ respectively. Conjugacy also dictates that this log conditional will be linear in $\mathbf{u}_Z(Z)$ for each co-parent $Z \in \mathrm{cp}_Y$. Hence, $\ln P(X \,|\, Y, \mathrm{cp}_Y)$ must be a multi-linear[3] function of the natural statistic functions $\mathbf{u}$ of X and its parents. This result is general in that *any* log conditional $\ln P(A \,|\, \mathrm{pa}_A)$ in a conjugate-exponential model must be a multi-linear function of the natural statistic functions of A and its parents.

For example, the log conditional $\ln P(X \,|\, Y, \beta)$ in (13) is multi-linear in each of the vectors,

$$\mathbf{u}_X(X) = \begin{bmatrix} X \\ X^2 \end{bmatrix}, \quad \mathbf{u}_Y(Y) = \begin{bmatrix} Y \\ Y^2 \end{bmatrix}, \quad \mathbf{u}_\beta(\beta) = \begin{bmatrix} \beta \\ \ln \beta \end{bmatrix}.$$

Returning to the variational update equation (8) for a node Y, it follows that all the expectations on the right hand side can be calculated in terms of the $\langle \mathbf{u} \rangle$ for each node in the Markov blanket of Y. Substituting for these expectations, we get

$$\begin{aligned} \ln Q_Y^*(Y) \;\; = \;\; & \big\langle \boldsymbol{\phi}_Y(\mathrm{pa}_Y)^{\mathrm{T}} \mathbf{u}_Y(Y) + f_Y(Y) + g_Y(\mathrm{pa}_Y) \big\rangle_{\sim Q(Y)} \\ & + \sum_{k \in \mathrm{ch}_Y} \big\langle \boldsymbol{\phi}_{XY}(X, \mathrm{cp}_Y)^{\mathrm{T}} \mathbf{u}_Y(Y) + \lambda(X, \mathrm{cp}_Y) \big\rangle_{\sim Q(Y)} + \mathrm{const.} \end{aligned}$$

3. A function f is a multi-linear function of variables $a, b \ldots$ if it varies linearly with respect to each variable, for example, $f(a, b) = ab + 3b$ is multi-linear in a and b.

which can be rearranged to give

$$\begin{aligned}\ln Q_Y^*(Y) &= \left[\langle \boldsymbol{\phi}_Y(\mathrm{pa}_Y)\rangle_{\sim Q(Y)} + \sum_{k\in \mathrm{ch}_Y} \langle \boldsymbol{\phi}_{XY}(X, \mathrm{cp}_Y)\rangle_{\sim Q(Y)}\right]^{\mathrm{T}} \mathbf{u}_Y(Y) \\ &\quad + f_Y(Y) + \text{const.}\end{aligned}$$

It follows that Q_Y^* is an exponential family distribution of the same form as Q_Y but with an updated natural parameter vector $\boldsymbol{\phi}_Y^*$ such that

$$\boldsymbol{\phi}_Y^* = \langle \boldsymbol{\phi}_Y(\mathrm{pa}_Y)\rangle + \sum_{k\in \mathrm{ch}_Y} \langle \boldsymbol{\phi}_{XY}(X, \mathrm{cp}_Y)\rangle \tag{16}$$

where all expectations are with respect to Q. As explained above, the expectations of $\boldsymbol{\phi}_Y$ and $\boldsymbol{\phi}_{XY}$ are both multi-linear functions of the expectations of the natural statistic vectors corresponding to their dependent variables. It is therefore possible to reparameterise these functions in terms of these expectations

$$\begin{aligned}\widetilde{\boldsymbol{\phi}}_Y\left(\{\langle \mathbf{u}_i\rangle\}_{i\in \mathrm{pa}_Y}\right) &= \langle \boldsymbol{\phi}_Y(\mathrm{pa}_Y)\rangle \\ \widetilde{\boldsymbol{\phi}}_{XY}\left(\langle \mathbf{u}_X\rangle, \{\langle \mathbf{u}_j\rangle\}_{j\in \mathrm{cp}_Y}\right) &= \langle \boldsymbol{\phi}_{XY}(X, \mathrm{cp}_Y)\rangle\,.\end{aligned}$$

In (15), we defined $\boldsymbol{\phi}_{XY}(X,\beta) = \begin{bmatrix} \beta X \\ -\beta/2 \end{bmatrix}$. We now reparameterise it as

$$\widetilde{\boldsymbol{\phi}}_{XY}\left(\langle \mathbf{u}_X\rangle, \langle \mathbf{u}_\beta\rangle\right) \stackrel{\mathrm{def}}{=} \begin{bmatrix} \langle \mathbf{u}_\beta\rangle_0 \langle \mathbf{u}_X\rangle_0 \\ -\frac{1}{2}\langle \mathbf{u}_\beta\rangle_0 \end{bmatrix}$$

where $\langle \mathbf{u}_X\rangle_0$ and $\langle \mathbf{u}_\beta\rangle_0$ are the first elements of the vectors $\langle \mathbf{u}_X\rangle$ and $\langle \mathbf{u}_\beta\rangle$ respectively (and so are equal to $\langle X\rangle$ and $\langle \beta\rangle$). As required, we have reparameterised $\boldsymbol{\phi}_{XY}$ into a function $\widetilde{\boldsymbol{\phi}}_{XY}$ which is a multi-linear function of natural statistic vectors.

3.3 Definition of the Variational Message Passing algorithm

We have now reached the point where we can specify exactly what form the messages between nodes must take and so define the Variational Message Passing algorithm. The message from a parent node Y to a child node X is just the expectation under Q of the natural statistic vector

$$\mathbf{m}_{Y\to X} = \langle \mathbf{u}_Y\rangle. \tag{17}$$

The message from a child node X to a parent node Y is

$$\mathbf{m}_{X\to Y} = \widetilde{\boldsymbol{\phi}}_{XY}\left(\langle \mathbf{u}_X\rangle, \{\mathbf{m}_{i\to X}\}_{i\in \mathrm{cp}_Y}\right) \tag{18}$$

which relies on X having received messages previously from all the co-parents. If any node A is observed then the messages are as defined above but with $\langle \mathbf{u}_A\rangle$ replaced by $\mathbf{u}_A$.

For example, if X is Gaussian distributed with conditional $P(X \mid Y, \beta)$, the messages to its parents Y and β are

$$\mathbf{m}_{X \to Y} = \begin{bmatrix} \langle \beta \rangle \langle X \rangle \\ -\langle \beta \rangle / 2 \end{bmatrix}, \qquad \mathbf{m}_{X \to \beta} = \begin{bmatrix} -\frac{1}{2}\left(\langle X^2 \rangle - 2 \langle X \rangle \langle Y \rangle + \langle Y^2 \rangle \right) \\ \frac{1}{2} \end{bmatrix}$$

and the message from X to any child node is $\begin{bmatrix} \langle X \rangle \\ \langle X^2 \rangle \end{bmatrix}$.

When a node Y has received messages from all parents and children, we can finds its updated posterior distribution Q_Y^* by finding its updated natural parameter vector $\boldsymbol{\phi}_Y^*$. This vector $\boldsymbol{\phi}_Y^*$ is computed from all the messages received at a node using

$$\boldsymbol{\phi}_Y^* = \widetilde{\boldsymbol{\phi}}_Y \left(\{ \mathbf{m}_{i \to Y} \}_{i \in \mathrm{pa}_Y} \right) + \sum_{j \in \mathrm{ch}_Y} \mathbf{m}_{j \to Y}, \tag{19}$$

which follows from (16). The new expectation of the natural statistic vector $\langle \mathbf{u}_Y \rangle_{Q_Y^*}$ can then be found, as it is a deterministic function of $\boldsymbol{\phi}_Y^*$.

The Variational Message Passing algorithm uses these messages to optimise the variational distribution iteratively, as described in Algorithm 1 below. This algorithm requires that the lower bound $\mathcal{L}(Q)$ be evaluated, which will be discussed in Section 3.5.

Algorithm 1 The Variational Message Passing algorithm

1. Initialise each factor distribution Q_j by initialising the corresponding moment vector $\langle \mathbf{u}_j(X_j) \rangle$.
2. For each node X_j in turn,
   - Retrieve messages from all parent and child nodes, as defined in (17) and (18). This will require child nodes to retrieve messages from the co-parents of X_j.
   - Compute updated natural parameter vector $\boldsymbol{\phi}_j^*$ using (19).
   - Compute updated moment vector $\langle \mathbf{u}_j(X_j) \rangle$ given the new setting of the parameter vector.
3. Calculate the new value of the lower bound $\mathcal{L}(Q)$ (if required).
4. If the increase in the bound is negligible or a specified number of iterations has been reached, stop. Otherwise repeat from step 2.

3.4 Example: the Univariate Gaussian Model

To illustrate how Variational Message Passing works, let us apply it to a model which represents a set of observed one-dimensional data $\{x_n\}_{n=1}^N$ with a univariate Gaussian distribution of mean μ and precision γ,

$$P(\mathbf{x} \mid \mathcal{H}) = \prod_{n=1}^{N} \mathcal{N}(x_n \mid \mu, \gamma^{-1}).$$

The conditional distribution of each data point x_n is a univariate Gaussian, which is in the exponential family and so its logarithm can be expressed in standard form as

$$\ln P(x_n \,|\, \mu, \gamma^{-1}) = \begin{bmatrix} \gamma\mu \\ -\gamma/2 \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} x_n \\ x_n^2 \end{bmatrix} + \frac{1}{2}(\ln\gamma - \gamma\mu^2 - \ln 2\pi)$$

and so $\mathbf{u}_x(x_n) = [x_n, x_n^2]^{\mathrm{T}}$. This conditional can also be written so as to separate out the dependencies on μ and γ

$$\begin{aligned} \ln P(x_n \,|\, \mu, \gamma^{-1}) &= \begin{bmatrix} \gamma x_n \\ -\gamma/2 \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} \mu \\ \mu^2 \end{bmatrix} + \frac{1}{2}(\ln\gamma - \gamma x_n^2 - \ln 2\pi) \qquad (20) \\ &= \begin{bmatrix} -\frac{1}{2}(x_n - \mu)^2 \\ \frac{1}{2} \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} \gamma \\ \ln\gamma \end{bmatrix} - \ln 2\pi \qquad (21) \end{aligned}$$

which shows that, for conjugacy, $\mathbf{u}_\mu(\mu)$ must be $[\mu, \mu^2]^{\mathrm{T}}$ and $\mathbf{u}_\gamma(\gamma)$ must be $[\gamma, \ln\gamma]^{\mathrm{T}}$ or linear transforms of these[4]. Therefore, μ must have a Gaussian prior and γ a Gamma prior since these are the exponential family distributions having these natural statistic vectors. We introduce the parameters m, β, a and b, so that

$$\begin{aligned} \ln P(\mu \,|\, m, \beta) &= \begin{bmatrix} \beta m \\ -\beta/2 \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} \mu \\ \mu^2 \end{bmatrix} + \frac{1}{2}(\ln\beta - \beta m^2 - \ln 2\pi) \\ \ln P(\gamma \,|\, a, b) &= \begin{bmatrix} -b \\ a - 1 \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} \gamma \\ \ln\gamma \end{bmatrix} + a\ln b - \ln\Gamma(a). \end{aligned}$$

3.4.1 Variational Message Passing in the Univariate Gaussian Model

We can now apply Variational Message Passing to infer the distributions over μ and γ variationally. The variational distribution is fully factorised and takes the form

$$Q(\mu, \gamma) = Q_\mu(\mu) Q_\gamma(\gamma).$$

We start by initialising $Q_\mu(\mu)$ and $Q_\gamma(\gamma)$ and find initial values of $\langle \mathbf{u}_\mu(\mu) \rangle$ and $\langle \mathbf{u}_\gamma(\gamma) \rangle$. Let us choose to update $Q_\mu(\mu)$ first, in which case Variational Message Passing will proceed as follows (illustrated in Figure 3a-d).

(a) As we wish to update $Q_\mu(\mu)$, we must first ensure that messages have been sent to the children of μ by any co-parents. Thus, messages $\mathbf{m}_{\gamma \to x_n}$ are sent from γ to each of the observed nodes x_n. These messages are the same, and are just equal to $\langle \mathbf{u}_\gamma(\gamma) \rangle = [\langle\gamma\rangle, \langle\ln\gamma\rangle]^{\mathrm{T}}$, where the expectation are with respect to the initial setting of Q_γ.

(b) Each x_n node has now received messages from all co-parents of μ and so can send a message to μ which is the expectation of the natural parameter vector in (20),

$$\mathbf{m}_{x_n \to \mu} = \begin{bmatrix} \langle\gamma\rangle x_n \\ -\langle\gamma\rangle/2 \end{bmatrix}.$$

4. To prevent the need for linear transformation of messages, a normalised form of natural statistic vectors will always be used, for example $[\mu, \mu^2]^{\mathrm{T}}$ or $[\gamma, \ln\gamma]^{\mathrm{T}}$.

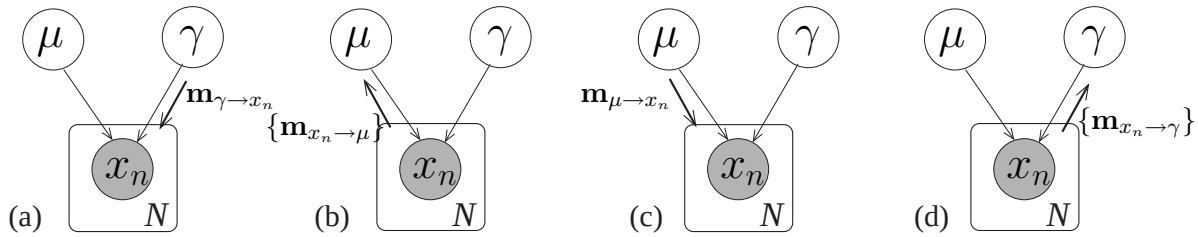

Figure 3: (a)-(d) Message passing procedure for variational inference in a univariate Gaussian model. The box around the x_i node denotes a *plate*, which indicates that the contained node and its connected edges are duplicated N times. The braces around the messages leaving the plate indicate that a set of N distinct messages are being sent.

(c) Node μ has now received its full complement of incoming messages and can update its natural parameter vector,

$$\boldsymbol{\phi}_\mu^* = \begin{bmatrix} \beta m \\ -\beta/2 \end{bmatrix} + \sum_{n=1}^{N} \mathbf{m}_{x_n \to \mu}$$

The new expectation $\langle \mathbf{u}_\mu(\mu) \rangle$ can then be computed under the updated distribution Q_μ^* and sent to each x_n as the message $\mathbf{m}_{\mu \to x_n} = [\langle \mu \rangle, \langle \mu^2 \rangle]^{\mathrm{T}}$.

(d) Finally, each x_n node sends a message back to γ which is

$$\mathbf{m}_{x_n \to \gamma} = \begin{bmatrix} -\frac{1}{2}(x_n^2 - 2x_n \langle \mu \rangle + \langle \mu^2 \rangle) \\ \frac{1}{2} \end{bmatrix}$$

and γ can update its variational posterior

$$\boldsymbol{\phi}_\gamma^* = \begin{bmatrix} -b \\ a - 1 \end{bmatrix} + \sum_{n=1}^{N} \mathbf{m}_{x_n \to \gamma}.$$

As the expectation of $\mathbf{u}_\gamma(\gamma)$ has changed, we can now go back to step (a) and send an updated message to each x_n node and so on. Hence, in Variational Message Passing, the message passing procedure is repeated again and again until convergence (unlike in belief propagation on a junction tree where the exact posterior is available after a message passing is performed once). Each round of message passing is equivalent to one iteration of the update equations in standard variational inference.

3.5 Calculation of the Lower Bound $\mathcal{L}(Q)$

The Variational Message Passing algorithm makes use of the lower bound $\mathcal{L}(Q)$ as a diagnostic of convergence. Evaluating the lower bound is also useful for performing model selection, or model averaging, because it provides an estimate of the log evidence for the model.

The lower bound can also play a useful role in helping to check the correctness both of the analytical derivation of the update equations and of their software implementation, simply by evaluating the bound after updating each factor in the variational posterior distribution and checking that the value of the bound does not decrease. This can be taken a stage further (Bishop and Svensén, 2003) by using numerical differentiation applied to the lower bound. After each update, the gradient of the bound is evaluated in the subspace corresponding to the parameters of the updated factor, to check that it is zero (within numerical tolerances). This requires that the differentiation take account of any constraints on the parameters (for instance that they be positive or that they sum to one). These checks, of course, provide necessary but not sufficient conditions for correctness. Also, they add computational cost so would typically only be employed whilst debugging the implementation.

In previous applications of variational inference, however, the evaluation of the lower bound has typically been done using separate code from that used to implement the update equations. Although the correctness tests discussed above also provide a check on the mutual consistency of the two bodies of code, it would clearly be more elegant if their evaluation could be unified.

This is achieved naturally in the Variational Message Passing framework by providing a way to calculate the bound automatically, as will now be described. To recap, the lower bound on the log evidence is defined to be

$$\mathcal{L}(Q) = \langle \ln P(\mathbf{H}, \mathbf{V}) \rangle - \langle Q(\mathbf{H}) \rangle \,,$$

where the expectations are with respect to Q. In a Bayesian network, with a factorised Q distribution, the bound becomes

$$\begin{aligned} \mathcal{L}(Q) &= \sum_i \langle \ln P(X_i \,|\, \mathrm{pa}_i) \rangle - \sum_{i \in \mathbf{H}} \langle \ln Q_i(H_i) \rangle \\ &\stackrel{\text{def}}{=} \sum_i \mathcal{L}_i \end{aligned}$$

where it has been decomposed into contributions from the individual nodes $\{\mathcal{L}_i\}$. For a particular latent variable node H_j, the contribution is

$$\mathcal{L}_j = \langle \ln P(H_j \,|\, \mathrm{pa}_j) \rangle - \langle \ln Q_j(H_j) \rangle \,.$$

Given that the model is conjugate-exponential, we can substitute in the standard form for the exponential family

$$\begin{aligned} \mathcal{L}_j &= \langle \boldsymbol{\phi}_j(\mathrm{pa}_j)^{\mathrm{T}} \rangle \langle \mathbf{u}_j(H_j) \rangle + \langle f_j(H_j) \rangle + \langle g_j(\mathrm{pa}_j) \rangle \\ &\quad - \left[\boldsymbol{\phi}_j^{*\mathrm{T}} \langle \mathbf{u}_j(H_j) \rangle + \langle f_j(H_j) \rangle + \widetilde{g}_j(\boldsymbol{\phi}_j^*) \right], \end{aligned}$$

where the function $\widetilde{g}_j$ is a reparameterisation of g_j so as to make it a function of the natural parameter vector rather than the parent variables. This expression simplifies to

$$\mathcal{L}_j = (\langle \boldsymbol{\phi}_j(\mathrm{pa}_j) \rangle - \boldsymbol{\phi}_j^*)^{\mathrm{T}} \langle \mathbf{u}_j(H_j) \rangle + \langle g_j(\mathrm{pa}_j) \rangle - \widetilde{g}_j(\boldsymbol{\phi}_j^*). \tag{22}$$

Three of these terms are already calculated during the Variational Message Passing algorithm: $\langle \boldsymbol{\phi}_j(\mathrm{pa}_j) \rangle$ and $\boldsymbol{\phi}_j^*$ when finding the posterior distribution over H_j in (19), and

$\langle \mathbf{u}_j(H_j) \rangle$ when calculating outgoing messages from H_j. Thus, considerable saving in computation are made compared to when the bound is calculated separately.

Each observed variable V_k also makes a contribution to the bound

$$\begin{aligned} \mathcal{L}_k &= \langle \ln P(V_k \,|\, \mathrm{pa}_k) \rangle \\ &= \langle \boldsymbol{\phi}_j(\mathrm{pa}_j) \rangle^{\mathrm{T}} \mathbf{u}_k(V_k) + f_k(V_k) + \widetilde{g}_k \left(\langle \boldsymbol{\phi}_j(\mathrm{pa}_j) \rangle \right). \end{aligned}$$

Again, computation can be saved by computing $\mathbf{u}_k(V_k)$ during the initialisation of the message passing algorithm.

Example 1 ***Calculation of the Bound for the Univariate Gaussian Model***
In the univariate Gaussian model, the bound contribution from each observed node x_n is

$$\mathcal{L}_{x_n} = \begin{bmatrix} \langle \gamma \rangle \langle \mu \rangle \\ -\langle \gamma \rangle / 2 \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} x_n \\ x_n^2 \end{bmatrix} + \frac{1}{2} \left(\langle \ln \gamma \rangle - \langle \gamma \rangle \langle \mu^2 \rangle - \ln 2\pi \right)$$

and the contributions from the parameter nodes μ and γ are

$$\begin{aligned} \mathcal{L}_\mu &= \begin{bmatrix} \beta' m' - \beta m \\ -\beta'/2 + \beta/2 \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} \langle \mu \rangle \\ \langle \mu^2 \rangle \end{bmatrix} + \frac{1}{2} \left(\ln \beta - \beta m^2 - \ln \beta' + \beta' m'^2 \right) \\ \mathcal{L}_\gamma &= \begin{bmatrix} -b' + b \\ a - a' \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} \langle \gamma \rangle \\ \langle \ln \gamma \rangle \end{bmatrix} + a \ln b - \ln \Gamma(a) - a' \ln b' + \ln \Gamma(a'). \end{aligned}$$

The bound for this univariate Gaussian model is given by the sum of the contributions from the μ and γ nodes and all x_n nodes.

4. Allowable Models

The Variational Message Passing algorithm can be applied to a wide class of models, which will be characterised in this section.

4.1 Conjugacy Constraints

The main constraint on the model is that each parent–child edge must satisfy the constraint of conjugacy. Conjugacy allows a Gaussian variable to have a Gaussian parent for its mean and we can extend this hierarchy to any number of levels. Each Gaussian node has a Gamma parent as the distribution over its precision. Furthermore, each Gamma distributed variable can have a Gamma distributed scale parameter b, and again this hierarchy can be extended to multiple levels.

A discrete variable can have multiple discrete parents with a Dirichlet prior over the entries in the conditional probability table. This allows for an arbitrary graph of discrete variables. A variable with an Exponential or Poisson distribution can have a Gamma prior over its scale or mean respectively, although, as these distributions do not lead to hierarchies, they may be of limited interest.

These constraints are listed in Table 1. This table can be encoded in implementations of the Variational Message Passing algorithm and used during initialisation to check the conjugacy of the supplied model.

Distribution	1st parent	Conjugate dist.	2nd parent	Conjugate dist.
Gaussian	mean μ	Gaussian	precision γ	Gamma
Gamma	shape a	None	scale b	Gamma
Discrete	probabilities $\mathbf{p}$	Dirichlet	parents $\{x_i\}$	Discrete
Dirichlet	pseudo-counts $\mathbf{a}$	None		
Exponential	scale a	Gamma		
Poisson	mean λ	Gamma		

Table 1: Distributions for each parameter of a number of exponential family distributions if the model is to satisfy conjugacy constraints. Conjugacy also holds if the distributions are replaced by their multivariate counterparts e.g. the distribution conjugate to the precision matrix of a multivariate Gaussian is a Wishart distribution. Where "None" is specified, no standard distribution satisfies conjugacy.

4.1.1 Truncated Distributions

The conjugacy constraint does not put any restrictions on the $f_X(X)$ term in the exponential family distribution. If we choose f_X to be a step function

$$f_X(X) = \begin{cases} 0 & : \quad X \geq 0 \\ -\infty & : \quad X < 0 \end{cases}$$

then we end up with a rectified distribution, so that $P(X \,|\, \boldsymbol{\theta}) = 0$ for $X < 0$. The choice of such a truncated distribution will change the form of messages to parent nodes (as the g_X normalisation function will also be different) but will not change the form of messages that are passed to child nodes. However, truncation will affect how the moments of the distribution are calculated from the updated parameters, which will lead to different values of child messages. For example, the moments of a rectified Gaussian distribution are expressed in terms of the standard 'erf' function. Similarly, we can consider doubly truncated distributions which are non-zero only over some finite interval, as long as the calculation of the moments and parent messages remains tractable. One potential problem with the use of a truncated distribution is that no standard distributions may exist which are conjugate for each distribution parameter.

4.2 Deterministic Functions

We can considerably enlarge the class of tractable models if variables are allowed to be defined as deterministic functions of the states of their parent variables. This is achieved by adding deterministic nodes into the graph, as have been used to similar effect in the BUGS software (see Section 5).

Consider a deterministic node X which has stochastic parents $\mathbf{Y} = \{Y_1, \ldots, Y_M\}$ and which has a stochastic child node Z. The state of X is given by a deterministic function f of the state of its parents, so that $X = f(\mathbf{Y})$. If X were stochastic, the conjugacy constraint with Z would require that $P(X \,|\, \mathbf{Y})$ must have the same functional form, with respect to X, as $P(Z \,|\, X)$. This in turn would dictate the form of the natural statistic vector $\mathbf{u}_X$ of X, whose expectation $\langle \mathbf{u}_X(X) \rangle_Q$ would be the message from X to Z.

Returning to the case where X is deterministic, it is still necessary to provide a message to Z of the form $\langle \mathbf{u}_X(X) \rangle_Q$ where the function $\mathbf{u}_X$ is dictated by the conjugacy constraint. This message can be evaluated only if it can be expressed as a function of the messages from the parent variables, which are the expectations of their natural statistics functions $\{\langle \mathbf{u}_{Y_i}(Y_i) \rangle_Q\}$. In other words, there must exist a vector function $\boldsymbol{\psi}_X$ such that

$$\langle \mathbf{u}_X(f(\mathbf{Y})) \rangle_Q = \boldsymbol{\psi}_X(\langle \mathbf{u}_{Y_1}(Y_1) \rangle_Q, \ldots, \langle \mathbf{u}_{Y_M}(Y_M) \rangle_Q).$$

As was discussed in Section 3.2, this constrains $\mathbf{u}_X(f(\mathbf{Y}))$ to be a multi-linear function of the set of functions $\{\mathbf{u}_{Y_i}(Y_i)\}$.

A deterministic node can be viewed as a having a conditional distribution which is a delta function, so that $P(X \,|\, \mathbf{Y}) = \delta(X - f(\mathbf{Y}))$. If X is discrete, this is the distribution that assigns probability one to the state $X = f(\mathbf{Y})$ and zero to all other states. If X is continuous, this is the distribution with the property that $\int g(X)\ \delta(X - f(\mathbf{Y}))\, dX = g(f(\mathbf{Y}))$. The contribution to the lower bound from a deterministic node is zero.

Example 2 ***Using a Deterministic Function as the Mean of a Gaussian***
Consider a model where a deterministic node X is to be used as the mean of a child Gaussian distribution $\mathcal{N}(Z \,|\, X, \beta^{-1})$ and where X equals a function f of Gaussian-distributed variables $Y_1, \ldots, Y_M$. The natural statistic vectors of X (as dictated by conjugacy with Z) and those of $Y_1, \ldots, Y_M$ are

$$\mathbf{u}_X(X) = \begin{bmatrix} X \\ X^2 \end{bmatrix}, \quad \mathbf{u}_{Y_i}(Y_i) = \begin{bmatrix} Y_i \\ Y_i^2 \end{bmatrix} \text{ for } i = 1 \ldots M$$

The constraint on f is that $\mathbf{u}_X(f)$ must be multi-linear in $\{\mathbf{u}_{Y_i}(Y_i)\}$ and so both f and f^2 must be multi-linear in $\{Y_i\}$ and $\{Y_i^2\}$. Hence, f can be any multi-linear function of $Y_1, \ldots, Y_M$. In other words, the mean of a Gaussian can be the sum of products of other Gaussian-distributed variables.

Example 3 ***Using a Deterministic Function as the Precision of a Gaussian***
As another example, consider a model where X is to be used as the precision of a child Gaussian distribution $\mathcal{N}(Z \,|\, \mu, X^{-1})$ and where X is a function f of Gamma-distributed variables $Y_1, \ldots, Y_M$. The natural statistic vectors of X and $Y_1, \ldots, Y_M$ are

$$\mathbf{u}_X(X) = \begin{bmatrix} X \\ \ln X \end{bmatrix}, \quad \mathbf{u}_{Y_i}(Y_i) = \begin{bmatrix} Y_i \\ \ln Y_i \end{bmatrix} \text{ for } i = 1 \ldots M.$$

and so both f and $\ln f$ must be multi-linear in $\{Y_i\}$ and $\{\ln Y_i\}$. This restricts f to be proportional to a product of the variables $Y_1, \ldots, Y_M$ as the logarithm of a product can be found in terms of the logarithms of terms in that product. Hence $f = c \prod_i Y_i$ where c is a constant. A function containing a summation, such as $f = \sum_i Y_i$, would not be valid as the logarithm of the sum cannot be expressed as a multi-linear function of Y_i and $\ln Y_i$.

4.2.1 Validating Chains of Deterministic Functions

The validity of a deterministic function for a node X is dependent on the form of the stochastic nodes it is connected to, as these dictate the functions $\mathbf{u}_X$ and $\{\mathbf{u}_{Y_i}(Y_i)\}$. For

example, if the function was a summation $f = \sum_i Y_i$, it would be valid for the first of the above examples but not for the second. In addition, it is possible for deterministic functions to be chained together to form more complicated expressions. For example, the expression $X = Y_1 + Y_2 Y_3$ can be achieved by having a deterministic product node A with parents Y_2 and Y_3 and a deterministic sum node X with parents Y_1 and A. In this case, the form of the function $\mathbf{u}_A$ is not determined directly by its immediate neighbours, but instead is constrained by the requirement of consistency for the connected deterministic subgraph.

In a software implementation of Variational Message Passing, the validity of a particular deterministic structure can most easily be checked by requiring that the function $\mathbf{u}_{X_i}$ be specified explicitly for each deterministic node X_i, thereby allowing the existing mechanism for checking conjugacy to be applied uniformly across both stochastic and deterministic nodes.

4.2.2 Deterministic Node Messages

To examine message passing for deterministic nodes, we must consider the general case where the deterministic node X has multiple children $\{Z_j\}$. The message from the node X to any child Z_j is simply

$$\begin{aligned} \mathbf{m}_{X \to Z_j} &= \langle \mathbf{u}_X(f(\mathbf{Y})) \rangle_Q \\ &= \boldsymbol{\psi}_X(\mathbf{m}_{Y_1 \to X}, \ldots, \mathbf{m}_{Y_M \to X}). \end{aligned}$$

For a particular parent Y_k, the function $\mathbf{u}_X(f(\mathbf{Y}))$ is linear with respect to $\mathbf{u}_{Y_k}(Y_k)$ and so it can be written as

$$\mathbf{u}_X(f(\mathbf{Y})) = \boldsymbol{\Psi}_{X,Y_k}(\{\mathbf{u}_{Y_i}(Y_i)\}_{i \neq k}) \,.\, \mathbf{u}_{Y_k}(Y_k) + \boldsymbol{\lambda}(\{\mathbf{u}_{Y_i}(Y_i)\}_{i \neq k})$$

where $\boldsymbol{\Psi}_{X,Y_k}$ is a matrix function of the natural statistics vectors of the co-parents of Y_k. The message from a deterministic node to a parent Y_k is then

$$\mathbf{m}_{X \to Y_k} = \left[\sum_j \mathbf{m}_{Z_j \to X} \right] \boldsymbol{\Psi}_{X,Y_k}(\{\mathbf{m}_{Y_i \to X}\}_{i \neq k})$$

which relies on having received messages from all the child nodes and from all the co-parents. The sum of child messages can be computed and stored locally at the node and used to evaluate all child-to-parent messages. In this sense, it can be viewed as the natural parameter vector of a distribution which acts as a kind of pseudo-posterior over the value of X.

4.3 Mixture Distributions

So far, only distributions from the exponential family have been considered. Often it is desirable to use richer distributions that better capture the structure of the system that generated the data. Mixture distributions, such as mixtures of Gaussians, provide one common way of creating richer probability densities. A mixture distribution over a variable X is a weighted sum of a number of component distributions

$$P(X \,|\, \{\pi_k\}, \{\boldsymbol{\theta}_k\}) = \sum_{k=1}^{K} \pi_k P_k(X \,|\, \boldsymbol{\theta}_k)$$

where each P_k is a component distribution with parameters $\boldsymbol{\theta}_k$ and a corresponding mixing coefficient π_k indicating the weight of the distribution in the weighted sum. The K mixing coefficients must be non-negative and sum to one.

A mixture distribution is not in the exponential family and therefore cannot be used directly as a conditional distribution within a conjugate-exponential model. Instead, we can introduce an additional discrete latent variable λ which indicates from which component distribution each data point was drawn, and write the distribution as

$$P(X \mid \lambda, \{\boldsymbol{\theta}_k\}) = \prod_{k=1}^{K} P_k(X \mid \boldsymbol{\theta}_k)^{\delta_{\lambda k}}.$$

Conditioned on this new variable, the distribution is now in the exponential family provided that all of the component distributions are also in the exponential family. In this case, the log conditional probability of X given all the parents (including λ) can be written as

$$\ln P(X \mid \lambda, \{\boldsymbol{\theta}_k\}) = \sum_k \delta(\lambda, k) \left[\boldsymbol{\phi}_k(\boldsymbol{\theta}_k)^{\mathrm{T}} \mathbf{u}_k(X) + f_k(X) + g_k(\boldsymbol{\theta}_k)\right].$$

If X has a child Z, then conjugacy will require that all the component distributions have the same natural statistic vector, which we can then call $\mathbf{u}_X$ so: $\mathbf{u}_1(X) = \mathbf{u}_2(X) = \ldots = \mathbf{u}_K(X) \stackrel{\text{def}}{=} \mathbf{u}_X(X)$. In addition, we may choose to specify, as part of the model, that all these distributions have exactly the same form (that is, $f_1 = f_2 = \ldots = f_K \stackrel{\text{def}}{=} f_X$), although this is not required by conjugacy. In this case, where all the distributions are the same, the log conditional becomes

$$\begin{aligned}
\ln P(X \mid \lambda, \{\boldsymbol{\theta}_k\}) &= \left[\sum_k \delta(\lambda, k) \boldsymbol{\phi}_k(\boldsymbol{\theta}_k)\right]^{\mathrm{T}} \mathbf{u}_X(X) + f_X(X) \\
&\quad + \sum_k \delta(\lambda, k) g_k(\boldsymbol{\theta}_k) \\
&= \boldsymbol{\phi}_X(\lambda, \{\boldsymbol{\theta}_k\})^T \mathbf{u}_X(X) + f_X(X) + \widetilde{g}_X(\boldsymbol{\phi}_X(\lambda, \{\boldsymbol{\theta}_k\}))
\end{aligned}$$

where we have defined $\boldsymbol{\phi}_X = \sum_k \delta(\lambda, k) \boldsymbol{\phi}_k(\boldsymbol{\theta}_k)$ to be the natural parameter vector of this mixture distribution and the function $\widetilde{g}_X$ is a reparameterisation of g_X to make it a function of $\boldsymbol{\phi}_X$ (as in Section 3.5). The conditional is therefore in the same exponential family form as each of the components.

We can now apply Variational Message Passing. The message from the node X to any child is $\langle \mathbf{u}_X(X) \rangle$ as calculated from the mixture parameter vector $\boldsymbol{\phi}_X(\lambda, \{\boldsymbol{\theta}_k\})$. Similarly, the message from X to a parent $\boldsymbol{\theta}_k$ is the message that would be sent by the corresponding component if it were not in a mixture, scaled by the variational posterior over the indicator variable $Q(\lambda = k)$. Finally, the message from X to λ is the vector of size K whose kth element is $\langle \ln P_k(X \mid \theta_k) \rangle$.

4.4 Multivariate Distributions

Until now, only scalar variables have been considered. It is also possible to handle vector variables in this framework (or to handle scalar variables which have been grouped into a

vector to capture posterior dependencies between the variables). In each case, a multivariate conditional distribution is defined in the overall joint distribution P and the corresponding factor in the variational posterior Q will also be multivariate, rather than factorised with respect to the elements of the vector. To understand how multivariate distributions are handled, consider the d-dimensional Gaussian distribution with mean $\boldsymbol{\mu}$ and precision matrix[5] $\boldsymbol{\Lambda}$:

$$P(\mathbf{x} \,|\, \boldsymbol{\mu}, \boldsymbol{\Lambda}^{-1}) = \sqrt{\frac{|\boldsymbol{\Lambda}|}{(2\pi)^d}} \exp\left(-\tfrac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\mathrm{T}} \boldsymbol{\Lambda}\,(\mathbf{x}-\boldsymbol{\mu})\right).$$

This distribution can be written in exponential family form

$$\ln \mathcal{N}(\mathbf{x} \,|\, \boldsymbol{\mu}, \boldsymbol{\Lambda}^{-1}) = \begin{bmatrix} \boldsymbol{\Lambda}\boldsymbol{\mu} \\ -\frac{1}{2}\mathrm{vec}(\boldsymbol{\Lambda}) \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} \mathbf{x} \\ \mathrm{vec}(\mathbf{x}\mathbf{x}^{\mathrm{T}}) \end{bmatrix} + \tfrac{1}{2}(\ln|\boldsymbol{\Lambda}| - \boldsymbol{\mu}^{\mathrm{T}}\boldsymbol{\Lambda}\boldsymbol{\mu} - d\ln 2\pi)$$

where $\mathrm{vec}(\cdot)$ is a function that re-arranges the elements of a matrix into a column vector in some consistent fashion, such as by concatenating the columns of the matrix. The natural statistic function for a multivariate distribution therefore depends on both the type of the distribution and its dimensionality d. As a result, the conjugacy constraint between a parent node and a child node will also constrain the dimensionality of the corresponding vector-valued variables to be the same. Multivariate conditional distributions can therefore be handled by VMP like any other exponential family distribution, which extends the class of allowed distributions to include multivariate Gaussian and Wishart distributions.

A group of scalar variables can act as a single parent of a vector-valued node. This is achieved using a deterministic *concatenation* function which simply concatenates a number of scalar values into a vector. In order for this to be a valid function, the scalar distributions must still be conjugate to the multivariate distribution. For example, a set of d univariate Gaussian distributed variables can be concatenated to act as the mean of a d-dimensional multivariate Gaussian distribution.

4.4.1 Normal-Gamma Distribution

The mean μ and precision γ parameters of a Gaussian distribution can be grouped together into a single bivariate variable $\mathbf{c} = \{\mu, \gamma\}$. The conjugate distribution for this variable is the Normal-Gamma distribution, which is written

$$\ln P(\mathbf{c} \,|\, m, \lambda, a, b) = \begin{bmatrix} m\lambda \\ -\frac{1}{2}\lambda \\ -b - \frac{1}{2}\lambda m^2 \\ a - \frac{1}{2} \end{bmatrix} \begin{bmatrix} \mu\gamma \\ \mu^2\gamma \\ \gamma \\ \ln\gamma \end{bmatrix} + \tfrac{1}{2}(\ln\lambda - \ln 2\pi) + a\ln b - \ln\Gamma(a).$$

This distribution therefore lies in the exponential family and can be used within VMP instead of separate Gaussian and Gamma distributions. In general, grouping these variables together will improve the approximation and so increase the lower bound. The multivariate form of this distribution, the Normal-Wishart distribution, is handled as described above.

5. The precision matrix of a multivariate Gaussian is the inverse of its covariance matrix.

4.5 Summary of Allowable Models

In summary, the Variational Message Passing algorithm can handle probabilistic models with the following very general architecture: arbitrary directed acyclic subgraphs of multinomial discrete variables (each having Dirichlet priors) together with arbitrary subgraphs of univariate and multivariate linear Gaussian nodes (having Gamma and Wishart priors), with arbitrary mixture nodes providing connections from the discrete to the continuous subgraphs. In addition, deterministic nodes can be included to allow parameters of child distributions to be deterministic functions of parent variables. Finally, any of the continuous distributions can be singly or doubly truncated to restrict the range of allowable values, provided that the appropriate moments under the truncated distribution can be calculated along with any necessary parent messages.

This architecture includes as special cases models such as Hidden Markov Models, Kalman filters, factor analysers, principal component analysers and independent component analysers, as well as mixtures and hierarchical mixtures of these.

5. VIBES: An Implementation of Variational Message Passing

The Variational Message Passing algorithm has been implemented in a software package called VIBES (Variational Inference in BayEsian networkS), first described by Bishop et al. (2002). Inspired by WinBUGS (a graphical user interface for BUGS by Lunn et al., 2000), VIBES allows for models to be specified graphically, simply by constructing the Bayesian network for the model. This involves drawing the graph for the network (using operations similar to those in a drawing package) and then assigning properties to each node such as its name, the functional form of the conditional distribution, its dimensionality and its parents. As an example, Figure 4 shows the Bayesian network for the univariate Gaussian model along with a screenshot of the same model in VIBES. Models can also be specified in a text file, which contains XML according to a pre-defined model definition schema. VIBES is written in Java and so can be used on Windows, Linux or any operating system with a Java 1.3 virtual machine.

As in WinBUGS, the convention of making deterministic nodes explicit in the graphical representation has been adopted, as this greatly simplifies the specification and interpretation of the model. VIBES also uses the plate notation of a box surrounding one or more nodes to denote that those nodes are replicated some number of times, specified by the parameter in the bottom right hand corner of the box.

Once the model is specified, data can be attached from a separate data file which contains observed values for some of the nodes, along with sizes for some or all of the plates. The model can then be *initialised* which involves: (i) checking that the model is valid by ensuring that conjugacy and dimensionality constraints are satisfied and that all parameters are specified; (ii) checking that the observed data is of the correct dimensionality; (iii) allocating memory for all moments and messages; (iv) initialisation of the individual distributions Q_i.

Following a successful initialisation, inference can begin immediately. As inference proceeds, the current state of the distribution Q_i for any node can be inspected using a range of diagnostics including tables of values and Hinton diagrams. If desired, the lower bound $\mathcal{L}(Q)$ can be monitored (at the expense of slightly increased computation), in which case the

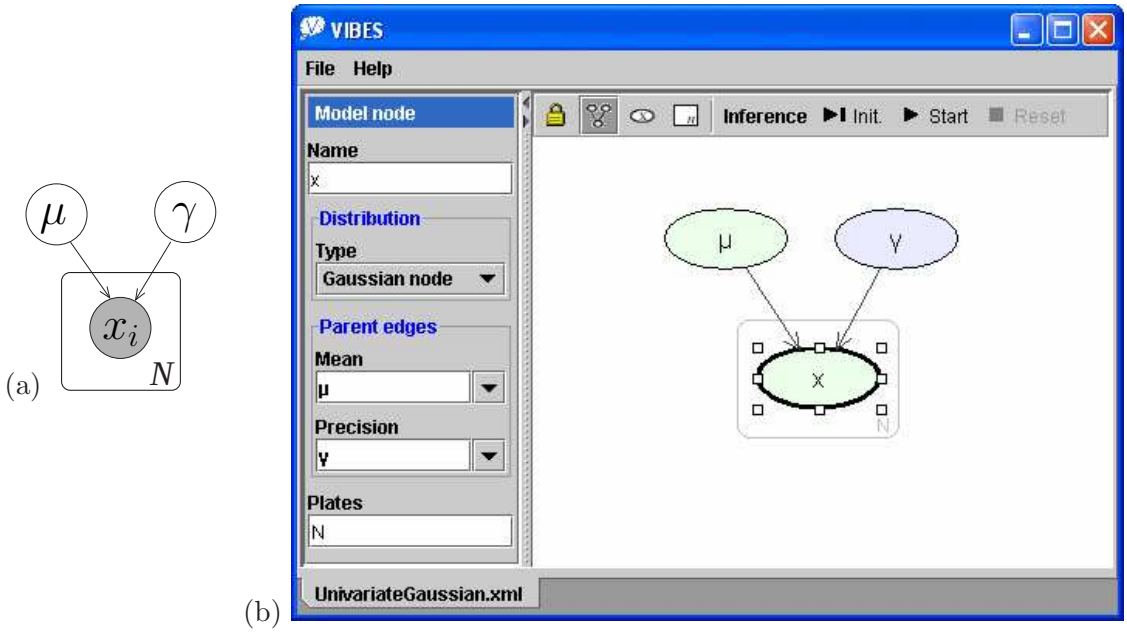

Figure 4: (a) Bayesian network for the univariate Gaussian model. (b) Screenshot of VIBES showing how the same model appears as it is being edited. The node x is selected and the panel to the left shows that it has a Gaussian conditional distribution with mean μ and precision γ. The plate surrounding x shows that it is duplicated N times and the heavy border indicates that it is observed (according to the currently attached data file).

optimisation can be set to terminate automatically when the change in the bound during one iteration drops below a small value. Alternatively, the optimisation can be stopped after a fixed number of iterations.

The VIBES software can be downloaded from `http://vibes.sourceforge.net`. This software was written by one of the authors (John Winn) whilst a Ph.D. student at the University of Cambridge and is free and open source. Appendix A contains a tutorial for applying VIBES to an example problem involving a Gaussian Mixture model. The VIBES web site also contains an online version of this tutorial.

6. Extensions to Variational Message Passing

In this section, three extensions to the Variational Message Passing algorithm will be described. These extensions are intended to illustrate how the algorithm can be modified to perform alternative inference calculations and to show how the conjugate-exponential constraint can be overcome in certain circumstances.

6.1 Further Variational Approximations: The Logistic Sigmoid Function

As it stands, the VMP algorithm requires that the model be conjugate-exponential. However, it is possible to sidestep the conjugacy requirement by introducing additional variational parameters and approximating non-conjugate conditional distributions by valid conjugate ones. We will now illustrate how this can be achieved using the example of a conditional distribution over a binary variable $x \in 0, 1$ of the form

$$\begin{aligned} P(x \,|\, a) &= \sigma(a)^x [1 - \sigma(a)]^{1-x} \\ &= e^{ax} \sigma(-a) \end{aligned}$$

where

$$\sigma(a) = \frac{1}{1 + \exp(-a)}$$

is the logistic sigmoid function.

We take the approach of Jaakkola and Jordan (1996) and use a variational bound for the logistic sigmoid function defined as

$$\sigma(a) \geqslant F(a, \xi) \stackrel{\text{def}}{=} \sigma(\xi) \exp[(a - \xi)/2 + \lambda(\xi)(a^2 - \xi^2)]$$

where $\lambda(\xi) = [1/2 - g(\xi)]/2\xi$ and ξ is a variational parameter. For any given value of a we can make this bound exact by setting $\xi^2 = a^2$. The bound is illustrated in Figure 5 in which the solid curve shows the logistic sigmoid function $\sigma(a)$ and the dashed curve shows the lower bound $F(a, \xi)$ for $\xi = 2$.

We use this result to define a new lower bound $\widetilde{\mathcal{L}} \leqslant \mathcal{L}$ by replacing each expectation of the form $\langle \ln[e^{ax} \sigma(-a)] \rangle$ with its lower bound $\langle \ln[e^{ax} F(-a, \xi)] \rangle$. The effect of this transformation is to replace the logistic sigmoid function with an exponential, therefore restoring conjugacy to the model. Optimisation of each ξ parameter is achieved by maximising this new bound $\widetilde{\mathcal{L}}$, leading to the re-estimation equation

$$\xi^2 = \left\langle a^2 \right\rangle_Q .$$

It is important to note that, as the quantity $\widetilde{\mathcal{L}}$ involves expectations of $\ln F(-a, \xi)$, it is no longer guaranteed to be exact for any value of ξ.

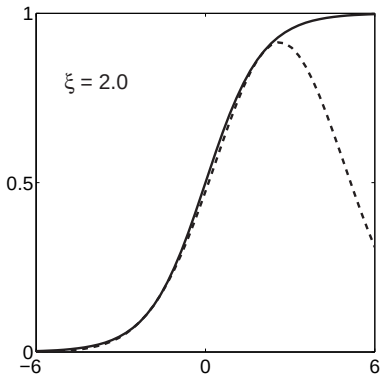

Figure 5: The logistic sigmoid function $\sigma(a)$ and variational bound $F(a, \xi)$.

It follows from (8) that the factor in Q corresponding to $P(x \mid a)$ is updated using

$$\begin{aligned}\ln Q_x^\star(x) &= \langle \ln(e^{ax}F(-a,\xi))\rangle_{\sim Q_x(x)} + \sum_{k\in ch_x} \langle \ln P(X_k|\mathrm{pa}_k)\rangle_{\sim Q_x(x)} + \text{const.} \\ &= \langle ax\rangle_{\sim Q_x(x)} + \sum_{k\in ch_x} \langle b_k x\rangle_{\sim Q_x(x)} + \text{const.} \\ &= a^\star x + \text{const.}\end{aligned}$$

where $a^\star = \langle a\rangle + \sum_k \langle b_k\rangle$ and the $\{b_k\}$ arise from the child terms which must be in the form $(b_k x + \text{const.})$ due to conjugacy. Therefore, the variational posterior $Q_x(x)$ takes the form

$$Q_x(x) = \sigma(a^\star)^x[1-\sigma(a^\star)]^{1-x}.$$

6.1.1 Using the Logistic Approximation within VMP

We will now explain how this additional variational approximation can be used within the VMP framework. The lower bound $\widetilde{\mathcal{L}}$ contains terms like $\langle \ln(e^{ax}F(-a,\xi))\rangle$ which need to be evaluated and so we must be able to evaluate $[\langle a\rangle \ \langle a^2\rangle]^{\mathrm{T}}$. The conjugacy constraint on a is therefore that its distribution must have a natural statistic vector $\mathbf{u}_a(a) = [a\ a^2]$. Hence it could, for example, be Gaussian.

For consistency with general discrete distributions, we write the bound on the log conditional $\ln P(x \mid a)$ as

$$\begin{aligned}\ln P(x \mid a) &\geqslant \begin{bmatrix} 0 \\ a \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} \delta(x-0) \\ \delta(x-1) \end{bmatrix} + (-a-\xi)/2 + \lambda(\xi)(a^2-\xi^2) + \ln\sigma(\xi) \\ &= \begin{bmatrix} \delta(x-1) - \frac{1}{2} \\ \lambda(\xi) \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} a \\ a^2 \end{bmatrix} - \xi/2 - \lambda(\xi)\xi^2 + \ln\sigma(\xi).\end{aligned}$$

The message from node x to node a is therefore

$$\mathbf{m}_{x\to a} = \begin{bmatrix} \langle\delta(x-1)\rangle - \frac{1}{2} \\ \lambda(\xi) \end{bmatrix}$$

and all other messages are as in standard VMP. The update of variational factors can then be carried out as normal except that each ξ parameter must also be re-estimated during optimisation. This can be carried out, for example, just before sending a message from x to a. The only remaining modification is to the calculation of the lower bound in (22), where the term $\langle g_j(\mathrm{pa}_j)\rangle$ is replaced by the expectation of its bound,

$$\langle g_j(\mathrm{pa}_j)\rangle \geqslant (-\langle a\rangle - \xi)/2 + \lambda(\xi)(\langle a^2\rangle - \xi^2) + \ln\sigma(\xi).$$

This extension to VMP enables discrete nodes to have continuous parents, further enlarging the class of allowable models. In general, the introduction of additional variational parameters enormously extends the class of models to which VMP can be applied, as the constraint that the model distributions must be conjugate no longer applies.

6.2 Finding a Maximum A Posteriori Solution

The advantage of using a variational distribution is that it provides a posterior distribution over latent variables. It is, however, also possible to use VMP to find a Maximum A Posteriori (MAP) solution, in which values of each latent variable are found that maximise the posterior probability. Consider choosing a variational distribution which is a delta function

$$Q^{\mathrm{MAP}}(\mathbf{H}) = \delta(\mathbf{H} - \mathbf{H}^\star)$$

where $H^\star$ is the MAP solution. From (3), the lower bound is

$$\begin{aligned} \mathcal{L}(\mathcal{Q}) &= \langle \ln P(\mathbf{H}, \mathbf{V}) \rangle - \langle \ln Q(\mathbf{H}) \rangle \\ &= \ln P(\mathbf{H}^\star, \mathbf{V}) \end{aligned}$$

and so maximising the bound is equivalent to finding the MAP solution. The variational distribution can be written in factorised form as

$$Q^{\mathrm{MAP}}(\mathbf{H}) = \prod_j Q_j(H_j).$$

with $Q_j(H_j) = \delta(H_j - H_j^\star)$. The KL divergence between the approximating distribution and the true posterior is minimised if $\mathrm{KL}(Q_j \,||\, Q_j^\star)$ is minimised, where $Q_j^\star$ is the standard variational solution given by (6). Normally, Q_j is unconstrained so we can simply set it to $Q_j^\star$. However, in this case, Q_j is a delta function and so we have to find the value of $H_j^\star$ that minimises $\mathrm{KL}(\delta(H_j - H_j^\star) \,||\, Q_j^\star)$. Unsurprisingly, this is simply the value of $H_j^\star$ that maximises $Q_j^\star(H_j^\star)$.

In the message passing framework, a MAP solution can be obtained for a particular latent variable H_j directly from the updated natural statistic vector $\phi_j^\star$ using

$$(\phi_j^\star)^{\mathrm{T}} \frac{d\mathbf{u}_j(H_j)}{dH_j} = 0.$$

For example, if $Q_j^\star$ is Gaussian with mean μ then $H_j^\star = \mu$ or if $Q_j^\star$ is Gamma with parameters a, b, then $H_j^\star = (a-1)/b$.

Given that the variational posterior is now a delta function, the expectation of any function $\langle f(H_j) \rangle$ under the variational posterior is just $f(H_j^\star)$. Therefore, in any outgoing messages, $\langle \mathbf{u}_j(H_j) \rangle$ is replaced by $\mathbf{u}_j(H_j^\star)$. Since all surrounding nodes can process these messages as normal, a MAP solution may be obtained for any chosen subset of variables (such as particular hyper-parameters), whilst a full posterior distribution is retained for all other variables.

6.3 Learning Non-conjugate Priors by Sampling

For some exponential family distribution parameters, there is no standard probability distribution which can act as a conjugate prior. For example, there is no standard distribution which can act as a conjugate prior for the shape parameter a of the Gamma distribution. This implies that we cannot learn a posterior distribution over a Gamma shape parameter within the basic VMP framework. As discussed above, we can sometimes introduce

conjugate approximations by adding variational parameters, but this may not always be possible.

The purpose of the conjugacy constraint is two-fold. First, it means that the posterior distribution of each variable, conditioned on its neighbours, has the same form as the prior distribution. Hence, the updated variational distribution factor for that variable has the same form and inference involves just updating the parameters of that distribution. Second, conjugacy results in variational distributions being in standard exponential family form allowing their moments to be calculated analytically.

If we ignore the conjugacy constraint, we get non-standard posterior distributions and we must resort to using sampling or other methods to determine the moments of these distributions. The disadvantages of using sampling include computational expense, inability to calculate an analytical lower bound and the fact that inference is no longer deterministic for a given initialisation and ordering. The use of sampling methods will now be illustrated by an example showing how to sample from the posterior over the shape parameter of a Gamma distribution.

Example 4 ***Learning a Gamma Shape Parameter***
Let us assume that there is a latent variable a which is to be used as the shape parameter of K gamma distributed variables $\{x_1 \dots x_K\}$. We choose a to have a non-conjugate *prior of an inverse-Gamma distribution:*

$$P(a \,|\, \alpha, \beta) \propto a^{-\alpha-1} \exp\left(\frac{-\beta}{a}\right).$$

The form of the gamma distribution means that messages sent to the node a are with respect to a natural statistic vector

$$\mathbf{u}_a = \begin{bmatrix} a \\ \ln \Gamma(a) \end{bmatrix}$$

which means that the updated factor distribution $Q_a^\star$ has the form

$$\ln Q_a^\star(a) = \left[\sum_{i=1}^{K} \mathbf{m}_{x_i \to a}\right]^{\mathrm{T}} \begin{bmatrix} a \\ \ln \Gamma(a) \end{bmatrix} + (-\alpha - 1) \ln a - \frac{\beta}{a} + \text{const.}$$

This density is not of standard form, but it can be shown that $Q^\star(\ln a)$ is log-concave, so we can generate independent samples from the distribution for $\ln a$ using Adaptive Rejection Sampling from Gilks and Wild (1992). These samples are then transformed to get samples of a from $Q_a^\star(a)$, which is used to estimate the expectation $\langle \mathbf{u}_a(a) \rangle$. This expectation is then sent as the outgoing message to each of the child nodes.

Each factor distribution is normally updated during every iteration and so, in this case, a number of independent samples from $Q_a^\star$ would have to be drawn during every iteration. If this proved too computationally expensive, then the distribution need only be updated intermittently.

It is worth noting that, as in this example, BUGS also uses Adaptive Rejection Sampling for sampling when the posterior distribution is log-concave but non-conjugate, whilst also providing techniques for sampling when the posterior is not log-concave. This suggests

that non-conjugate parts of a general graphical model could be handled within a BUGS-style framework whilst Variational Message Passing is used for the rest of the model. The resulting hybrid variational/sampling framework would, to a certain extent, capture the advantages of both techniques.

7. Discussion

The Variational Message Passing algorithm allows approximate inference using a factorised variational distribution in any conjugate-exponential model, and in a range of non-conjugate models. As a demonstration of its utility, this algorithm has already been used to solve problems in the domain of machine vision and bioinformatics (see Winn, 2003, Bishop and Winn, 2000). In general, Variational Message Passing dramatically simplifies the construction and testing of new variational models and readily allows a range of alternative models to be tested on a given problem.

The general form of VMP also allows the inclusion of arbitrary nodes in the graphical model provided that each node is able to receive and generate appropriate messages in the required form, whether or not the model remains conjugate-exponential. The extensions to VMP concerning the logistic function and sampling illustrate this flexibility.

One limitation of the current algorithm is that it uses a variational distribution which is factorised across nodes, giving an approximate posterior which is separable with respect to individual (scalar or vector) variables. In general, an improved approximation will be achieved if a posterior distribution is used which retains some dependency structure. Whilst Wiegerinck (2000) has presented a general framework for such structured variational inference, he does not provide a general-purpose algorithm for applying this framework. Winn (2003) and Bishop and Winn (2003) have therefore proposed an extended version of Variational Message Passing which allows for structured variational distributions. VIBES has been extended to implement a limited version of this algorithm that can only be applied to a constrained set of models. However, a complete implementation and evaluation of this extended algorithm has yet to be undertaken.

The VIBES software is free and open source and can be downloaded from the VIBES web site at `http://vibes.sourceforge.net`. The web site also contains a tutorial that provides an introduction to using VIBES.

Acknowledgments

The authors would like to thank David Spiegelhalter for his help with the VIBES project. We would also like to thank Zoubin Ghahramani, David MacKay, Matthew Beal and Michael Jordan for many helpful discussions about variational inference.

This work was carried out whilst John Winn was a Ph.D. student at the University of Cambridge, funded by a Microsoft Research studentship.

Appendix A: VIBES Tutorial

In this appendix, we demonstrate the application of VIBES to an example problem involving a Gaussian Mixture model. We then demonstrate the flexibility of VIBES by changing the model to fit the data better, using the lower bound as an estimate of the log evidence for each model. An online version of this tutorial is available at `http://vibes.sourceforge.net/tutorial`.

The data used in this tutorial is two-dimensional and consists of nine clusters in a three-by-three grid, as illustrated in Figure 6.

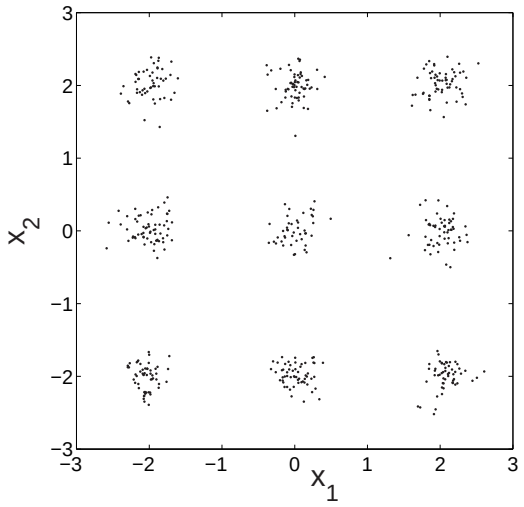

Figure 6: The two-dimensional data set used in the tutorial, which consists of nine clusters in a three-by-three grid.

Loading Matlab Data Into VIBES

The first step is to load the data set into VIBES. This is achieved by creating a node with the name x which corresponds to a matrix x in a Matlab `.mat` file. As the data matrix is two dimensional, the node is placed inside two plates N and d and the data filename (in this case `MixGaussianData2D.mat`) is entered. Selecting `File`→`Load data` loads the data into the node and also sets the size of the N and d plates to 500 and 2 respectively. The node is marked as observed (shown with a bold edge) and the observed data can be inspected by double-clicking the node with the mouse. At this point, the display is as shown in Figure 7.

Creating and Learning a Gaussian Model

The node x has been marked as Gaussian by default and so the model is invalid as neither the mean nor the precision of the Gaussian have been set (attempting to initialise the model by pressing the `Init.` button will give an error message to this effect). We can specify latent variables for these parameters by creating a node μ for the mean parameter and a node γ

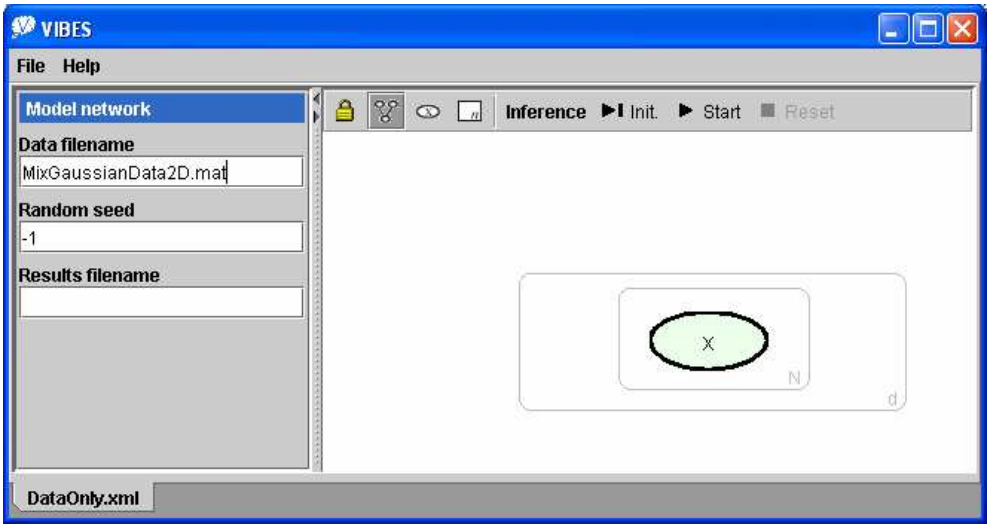

Figure 7: A VIBES model with a single observed node x which has attached data.

for the precision parameter. These nodes are created within the d plate to give a model which is separable over each data dimension. These are then set as the `Mean` and `Precision` properties of x, as shown in Figure 8.

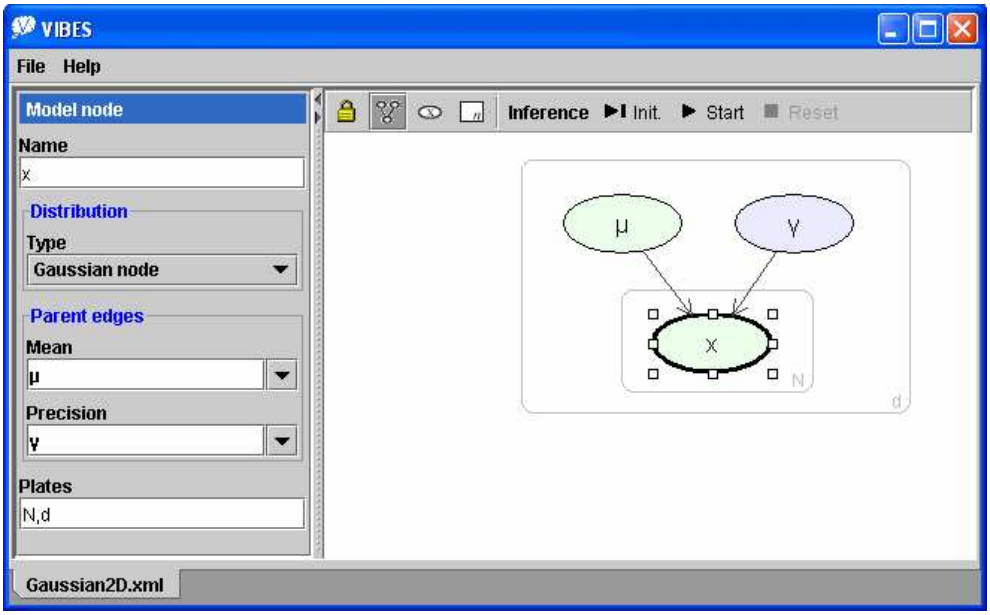

Figure 8: A two-dimensional Gaussian model, showing that the variables μ and γ are being used as the mean and precision parameters of the conditional distribution over x.

The model is still invalid as the parameters of μ and γ are unspecified. In this case, rather than create further latent variables, these parameters will be set to fixed values

to give appropriate priors (for example setting μ to have mean = 0 and precision = 0.3 and γ to have $a = 10$ and $b = 1$). The network now corresponds to a two-dimensional Gaussian model and variational inference can be performed automatically by pressing the `Start` button (which also performs initialisation). For this data set, inference converges after four iterations and gives a bound of -1984 nats. At this point, the expected values of each latent variable under the fully-factorised Q distribution can be displayed or graphed by double-clicking on the corresponding node.

Extending the Gaussian model to a Gaussian mixture model

Our aim is to create a Gaussian mixture model and so we must extend our simple Gaussian model to be a mixture with K Gaussian components. As there will now be K sets of the latent variables μ and γ, these are placed in a new plate, called K, whose size is set to 20. We modify the conditional distribution for the x node to be a mixture of dimension K, with each component being Gaussian. The display is then as shown in Figure 9.

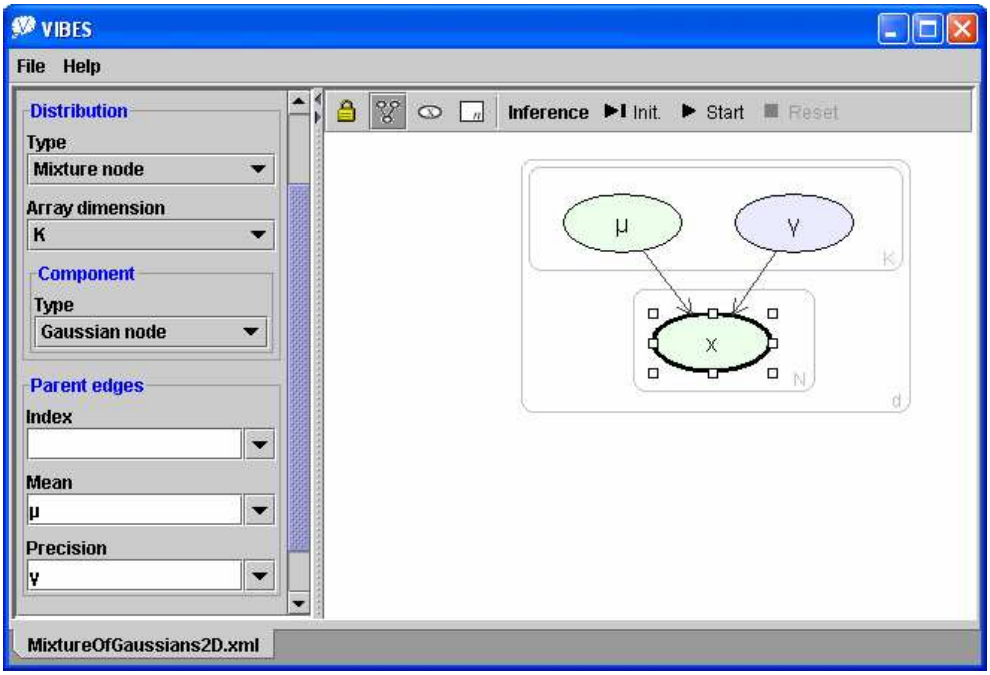

Figure 9: An incomplete model which shows that x is now a mixture of K Gaussians. There are now K sets of parameters and so μ and γ have been placed in a plate K. The model is incomplete as the `Index` parent of x has not been specified.

The model is currently incomplete as making x a mixture requires a new discrete `Index` parent to indicate which component distribution each data point was drawn from. We must therefore create a new node λ, sitting in the N plate, to represent this new discrete latent variable. We also create a node π with a Dirichlet distribution which provides a prior over λ. The completed mixture model is shown in Figure 10.

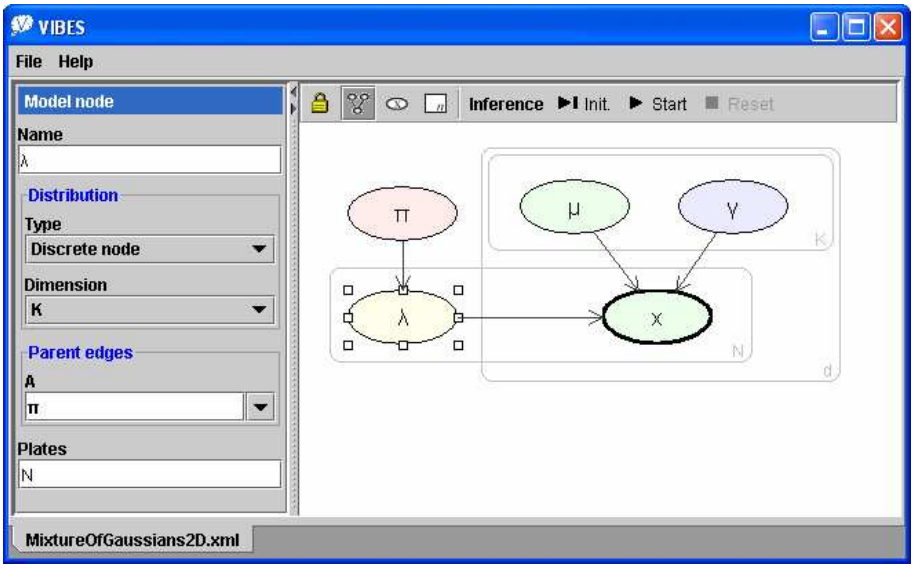

Figure 10: The completed Gaussian mixture model showing the discrete indicator node λ.

Inference using the Gaussian mixture model

With the model complete, inference can once again proceed automatically by pressing the `Start` button. A Hinton diagram of the expected value of π can be displayed by double-clicking on the π node, giving the result shown in Figure 11. As can be seen, nine of the twenty components have been retained.

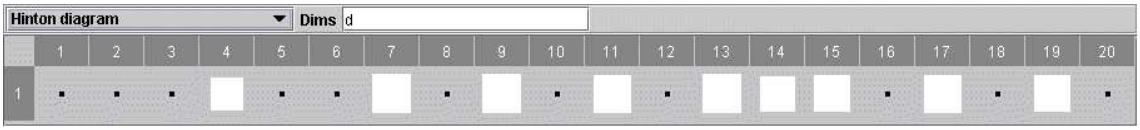

Figure 11: A Hinton diagram showing the expected value of π for each mixture component. The learned mixture consists of only nine components.

The means of the retained components can be inspected by double-clicking on the μ node, giving the Hinton diagram of Figure 12. These learned means correspond to the centres of each of the data clusters.

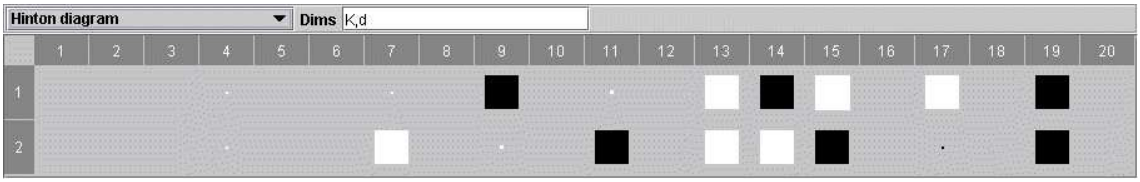

Figure 12: A Hinton diagram whose columns give the expected two-dimensional value of the mean μ for each mixture component. The mean of each of the eleven unused components is just the expected value under the prior which is $(0, 0)$. Column 4 corresponds to a retained component whose mean is roughly $(0, 0)$.

A graph of the evolution of the bound can be displayed by clicking on the bound value and is shown in Figure 13. The converged lower bound of this new model is -1019 nats, which is significantly higher than that of the single Gaussian model, showing that there is much greater evidence for this model. This is unsurprising since a mixture of 20 Gaussians has significantly more parameters than a single Gaussian and hence can give a much closer fit to the data. Note, however, that the model automatically chooses only to exploit 9 of these components, with the remainder being suppressed (by virtue of their mixing coefficients going to zero). This provides an elegant example of automatic model complexity selection within a Bayesian setting.

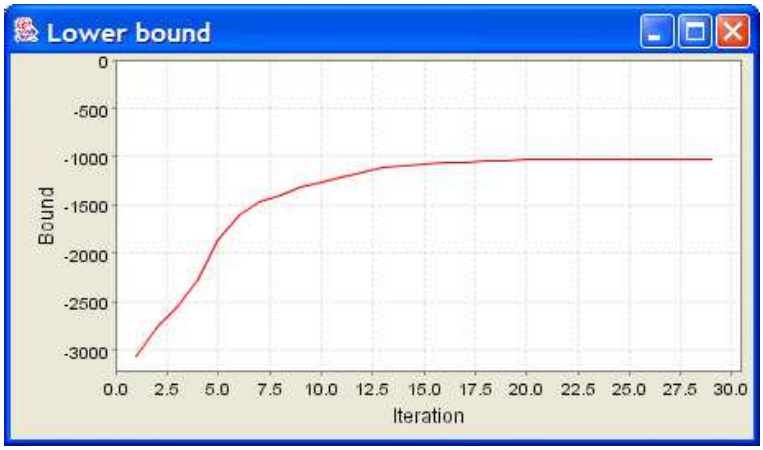

Figure 13: A graph of the evolution of the lower bound during inference.

Modifying the mixture model

The rapidity with which models can be constructed using VIBES allows new models to be quickly developed and compared. For example, we can take our existing mixture of Gaussians model and modify it to try and find a more probable model.

First, we may hypothesise that each of the clusters has similar size and so they may be modelled by a mixture of Gaussian components having a common variance in each dimension. Graphically, this corresponds to shrinking the K plate so that it no longer contains the γ node, as shown in Figure 14a. The converged lower bound for this new model is -937 nats showing that this modified model is better at explaining this data set than the standard mixture of Gaussians model. Note that the increase in model probability does not arise from an improved fit to the data, since this model and the previous one both contain 20 Gaussian components and in both cases 9 of these components contribute to the data fit. Rather, the constrained model having a single variance parameter can achieve almost as good a data fit as the unconstrained model yet with far fewer parameters. Since a Bayesian approach automatically penalises complexity, the simpler (constrained) model has the higher probability as indicated by the higher value for the variational lower bound.

We may further hypothesise that the data set is separable with respect to its two dimensions (i.e. the two dimensions are independent). Graphically this consists of moving all nodes inside the d plate (so we effectively have two copies of a one-dimensional mixture of

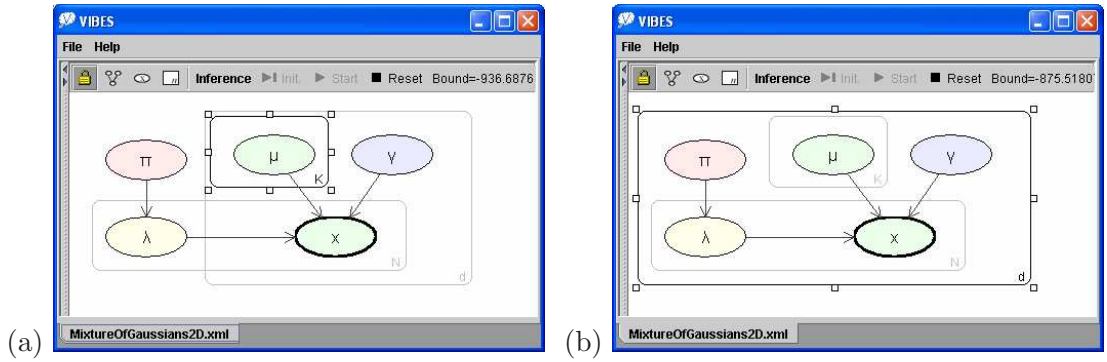

Figure 14: (a) Mixture of Gaussians model with shared precision parameter γ (the γ node is no longer inside the K plate). (b) Model with independent data dimensions, each a univariate Gaussian mixture with common variance.

Gaussians model with common variance). A VIBES screenshot of this further modification is shown in Figure 14b. Performing variational inference on this separable model leads to each one-dimensional mixture having three retained mixture components and gives an improved bound of -876 nats.

We will consider one final model. In this model both the π and the γ nodes are common to both data dimensions, as shown in Figure 15. This change corresponds to the assumption that the mixture coefficients are the same for each of the two mixtures and that the component variances are the same for all components in both mixtures. Inference leads to a final improved bound of -856 nats. Whilst this tutorial has been on a toy data set, the principles of model construction, modification and comparison can be applied just as readily to real data sets.

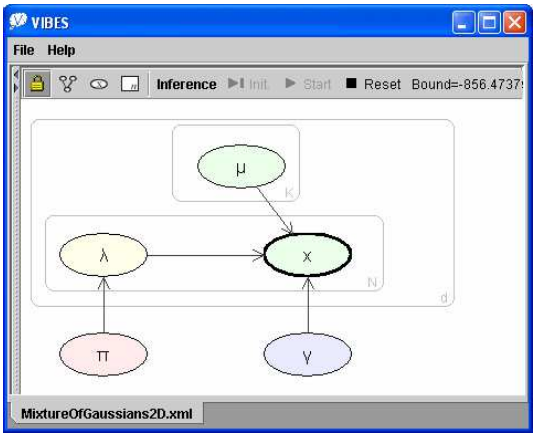

Figure 15: Further modified mixture model where the π and γ nodes are now common to all data dimensions.

References

H. Attias. A variational Bayesian framework for graphical models. In S. Solla, T. K. Leen, and K-L Muller, editors, *Advances in Neural Information Processing Systems*, volume 12, pages 209–215, Cambridge MA, 2000. MIT Press.

C. M. Bishop. Variational principal components. In *Proceedings Ninth International Conference on Artificial Neural Networks, ICANN'99*, volume 1, pages 509–514. IEE, 1999.

C. M. Bishop and M. Svensén. Bayesian Hierarchical Mixtures of Experts. In U. Kjaerulff and C. Meek, editors, *Proceedings Nineteenth Conference on Uncertainty in Artificial Intelligence*, pages 57–64. Morgan Kaufmann, 2003.

C. M. Bishop and J. M. Winn. Non-linear Bayesian image modelling. In *Proceedings Sixth European Conference on Computer Vision*, volume 1, pages 3–17. Springer-Verlag, 2000.

C. M. Bishop and J. M. Winn. Structured variational distributions in VIBES. In *Proceedings Artificial Intelligence and Statistics*, Key West, Florida, 2003. Society for Artificial Intelligence and Statistics.

C. M. Bishop, J. M. Winn, and D. Spiegelhalter. VIBES: A variational inference engine for Bayesian networks. In *Advances in Neural Information Processing Systems*, volume 15, 2002.

R. G. Cowell, A. P. Dawid, S. L. Lauritzen, and D. J. Spiegelhalter. *Probabilistic Networks and Expert Systems*. Statistics for Engineering and Information Science. Springer-Verlag, 1999.

Z. Ghahramani and M. J. Beal. Propagation algorithms for variational Bayesian learning. In T. K. Leen, T. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems*, volume 13, Cambridge MA, 2001. MIT Press.

W. R. Gilks and P. Wild. Adaptive rejection sampling for Gibbs sampling. *Applied Statistics*, 41(2):337–348, 1992.

T. Jaakkola and M. Jordan. A variational approach to Bayesian logistic regression problems and their extensions. In *In Proceedings of the 6th international workshop on artificial intelligence and statistics.*, 1996.

M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul. An introduction to variational methods for graphical models. In M. I. Jordan, editor, *Learning in Graphical Models*, pages 105–162. Kluwer, 1998.

Steffen L. Lauritzen. Propagation of probabilities, means, and variances in mixed graphical association models. *Journal of the American Statistical Association*, 87(420):1098–1108, 1992.

D. J. Lunn, A. Thomas, N. G. Best, and D. J. Spiegelhalter. WinBUGS – a Bayesian modelling framework: concepts, structure and extensibility. *Statistics and Computing*, 10:321–333, 2000. http://www.mrc-bsu.cam.ac.uk/bugs/.

R. M. Neal and G. E. Hinton. A new view of the EM algorithm that justifies incremental and other variants. In M. I. Jordan, editor, *Learning in Graphical Models*, pages 355–368. Kluwer, 1998.

J. Pearl. Fusion, propagation and structuring in belief networks. *Artificial Intelligence*, 29: 241–288, 1986.

A. Thomas, D. J. Spiegelhalter, and W. R. Gilks. BUGS: A program to perform Bayesian inference using Gibbs sampling. In J. M. Bernardo, J. O. Berger, A. P. Dawid, and A. F. M. Smith, editors, *Bayesian Statistics*, Oxford: Clarendon Press, 1992.

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

J. M. Winn. *Variational Message Passing and its Applications*. PhD thesis, University of Cambridge, October 2003.