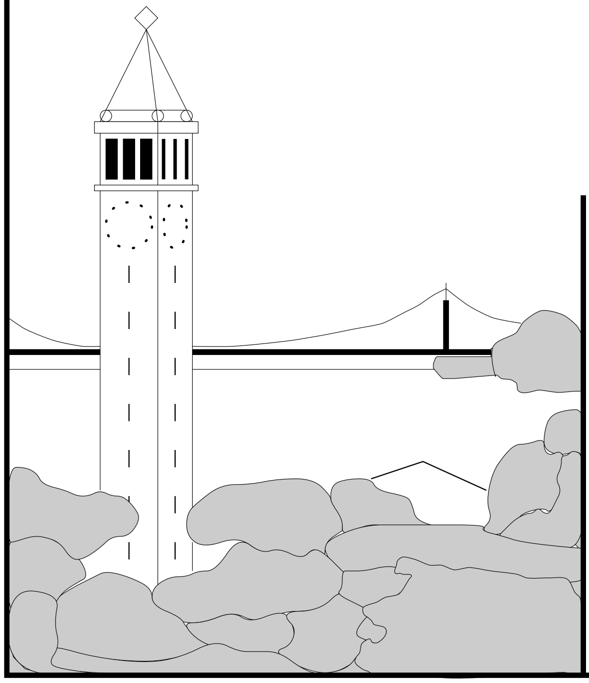
Inference and Learning in Hybrid Bayesian Networks

Kevin P. Murphy



Report No. UCB/CSD-98-990

January 1998

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

Abstract

We survey the literature on methods for inference and learning in Bayesian Networks composed of discrete and continuous nodes, in which the continuous nodes have a multivariate Gaussian distribution, whose mean and variance depends on the values of the discrete nodes. We also briefly consider hybrid Dynamic Bayesian Networks, an extension of switching Kalman filters. This report is meant to summarize what is known at a sufficient level of detail to enable someone to implement the algorithms, but without dwelling on formalities.[1]

1 Update

The algorithm described in this report, due to [Lau92] (see also [CDLS99, ch. 7]), has been implemented as part of my Bayes Net Toolbox[2]. However, it can be numerically unstable. A different algorithm, which fixes this problem, is described in [LJ99]. Other references to more recent material have also been added where appropriate.

2 Introduction

We discuss Bayesian networks (BNs [Jen96]) in which each node is either discrete or continuous, scalar or vector-valued, and in which the joint distribution over all the nodes is Conditional Gaussian (CG) [LW89, Lau92] i.e., for each instantiation $\mathbf{i}$ of the discrete nodes $\mathbf{Y}$, the distribution over the continuous nodes $\mathbf{X}$ has the form $f(\mathbf{x}|\mathbf{Y} = \mathbf{i}) = N(\mathbf{x}, \vec{\mu}(\mathbf{i}), \Sigma(\mathbf{i}))$, where $N()$ represents a multivariate Gaussian (MVG) or Normal density. (Note that discrete nodes cannot have continuous parents in this model.) This is the most general kind of BN for which exact inference algorithms are known. A related review article is [RG99].

We start by discussing how to represent the conditional probability distribution of each node, and the joint distribution this encodes. We then give an example of a hybrid BN before discussing inference and learning techniques.

3 Representing the local conditional probability distributions

3.1 Discrete nodes

The conditional distribution of a discrete node X_i given its parents $\text{Pa}(X_i)$ can be specified by means of a table (called a Conditional Probability Table or CPT), whose entries are $\theta_{ijk} = \Pr(X_i = k|\text{Pa}(X_i) = j)$. Here, j denotes the j'th possible value (instantiation) that $\text{Pa}(X_i)$ can have. Clearly we require $\sum_k \theta_{ijk} = 1$ for all i and j. Note that there are many ways of specifying the conditional distribution of a discrete node that require fewer parameters than a full table, e.g., noisy-OR [Pea88], causal interaction models [MH97], decision trees [BFGK96], default tables [FG96], etc.

[1]Updated 18 August 2000. This work was supported by grant number 442427-21957 awarded to Prof. Russell

[2]`www.cs.berkeley.edu/ ~murphyk/Bayes/bnt.html`

3.2 Continuous nodes

The conditional distribution of a continuous node X_i given its parents $\mathrm{Pa}(X_i)$ can be specified by a Gaussian function, whose mean is a linear function of the parents, and whose covariance is fixed. This is the standard linear regression model. We shall start by assuming that X_i and its parents are real valued scalars; then we shall consider the vector case, and finally the case where some of the parents are discrete.

3.2.1 Scalar case

Let node X_i have parents $X_{k_1}, \ldots, X_{k_p}$. Then its conditional distribution is

$$f(x_i|x_{k_1}, \ldots, x_{k_p}) = (2\pi\sigma^2)^{-\frac{1}{2}} \exp\left[-\frac{1}{2\sigma^2}(x_i - u_i)^2\right] \tag{1}$$

where $u_i = \mu_i + \sum_{k \in \mathrm{Pa}(X_i)} b_{ki}(x_k - \mu_k)$, and the b_{ki} are the "weights" or regression coefficients on the arcs coming into node i from its parents. Equivalently, we may write

$$X_i \;=\; \mu_i + \sum_{k \in \mathrm{Pa}(X_i)} b_{ki}(X_k - \mu_k) + \sigma_i W_i \tag{2}$$

where $W_i \sim N(0,1)$ is a white noise random variable.

Alternatively, we might consider the following model in which we don't subtract off the parents' means:

$$X_i \;=\; \sum_{k \in \mathrm{Pa}(X_i)} b_{ki}\, X_k + C_i \tag{3}$$

where $C_i \sim N(\mu_i, \sigma_i)$ is a colored noise term.

3.2.2 Vector case

We can imagine a simple extension to the above scheme in which each node can be a vector. In this case, the conditional distribution becomes

$$f(\mathbf{x}_i|\mathbf{x}_{k_1}, \ldots, \mathbf{x}_{k_p}) = N(\mathbf{x}_i, \mathbf{u}_i, \Sigma_i) = (2\pi)^{-\frac{n}{2}} |\Sigma_i|^{-\frac{1}{2}} \exp\left[-\tfrac{1}{2}(\mathbf{x}_i - \mathbf{u}_i)^T\, \Sigma_i^{-1}\, (\mathbf{x}_i - \mathbf{u}_i)\right] \tag{4}$$

where $\mathbf{u}_i = \vec{\mu}_i + \sum_k B_{ki}(\mathbf{x}_k - \vec{\mu}_k)$. We can view this as multivariate or generalized linear regression:

$$E\,\mathbf{X}_i = \mathbf{u}_i = \vec{\mu}_i + \sum_k B_{ki}(\mathbf{X}_k - \vec{\mu}_k)$$

or

$$E\,\mathbf{X}_i = \mathbf{u}_i = \vec{\mu}_i + \sum_k B_{ki}\mathbf{X}_k$$

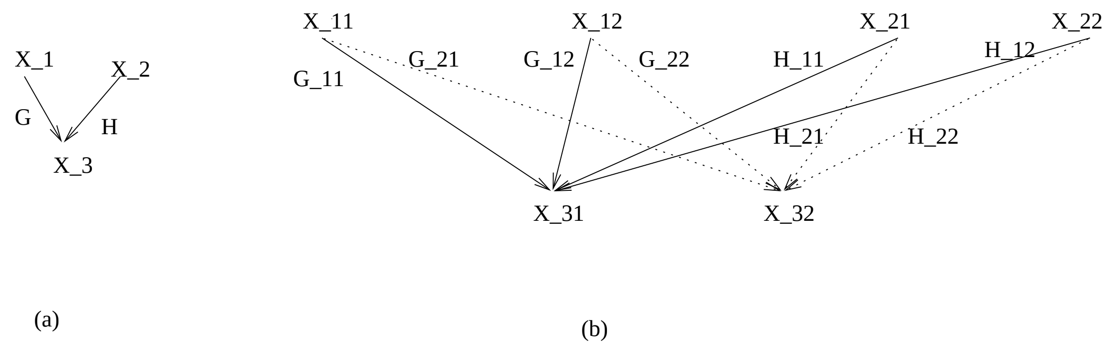


Figure 1: (a) A BN with vector-valued nodes. G and H are matrices. (b) The scalar BN corresponding to (a) in the case where $\mathbf{X}_i \in R^2$. The solid arcs to Z_{31} have coefficients which correspond to the first rows of G and H. That is, $\mathbf{b}_{X_{31}} = (G_{11}, G_{12}, H_{11}, H_{12})$. Similarly, the dotted arcs to X_{32} have coefficients which correspond to the second rows of G and H.

For example, in Figure 1(a), we have

$$\mathbf{X}_3 = G\mathbf{X}_1 + H\mathbf{X}_2 + \mathbf{C}$$

where $\mathbf{C} \sim N(\vec{\mu}_i, \Sigma_i)$ and we write $G = B_{13}$ and $H = B_{23}$ to avoid a profusion of subscripts.

Note that we can expand each vector into its components, yielding the equivalent scalar network shown in Figure 1(b). However, the vector notation is more compact.

3.2.3 Discrete parent case

If node X has discrete parents Y and continuous parents Z, it has a different mean, covariance and weight matrix for every value of Y. That is,

$$f(\mathbf{x}|\mathbf{y} = i, \mathbf{z}) = N(\mathbf{x}, \vec{\mu}_i + B_i\mathbf{z}, \Sigma_i).$$

Note that in this case it does not make sense to subtract off the parents' means, since they may depend on *their* discrete parents.

3.2.4 Continuous parent/ discrete child case

If a discrete node has continuous parents (e.g., a threshold unit), we can use the probit or logistic distribution. Unfortunately, exact inference in this case is intractable (unless all the continuous nodes are observed). For one possible approximation, see [Mur99].

4 Characterizing the corresponding joint distribution

In the introduction we stated that, if $\mathbf{X}$ represents all the continuous nodes and $\mathbf{Y}$ represents all the discrete nodes, then the joint distribution on $\mathbf{X}$ given a specific $\mathbf{Y} = \mathbf{i}$ is a multivariate Gaussian with parameters,

$\vec{\mu}(\mathbf{i})$ and $\Sigma(\mathbf{i})$. We now show how to to compute these parameters as a function of the the local parameters of each node.

4.1 Scalar case

We start by considering the scalar case, as in [SK89]. First we compute Σ and then $\vec{\mu}$.

Construct a diagonal matrix containing the variances of each node, $D = \text{diag}(\sigma_i^2)$, and another containing the standard deviations, $S = \text{diag}(\sigma_i)$. Also, construct a matrix B in which the i'th column contains the weight vector for node i. We assume the nodes are numbered topologically, so B is upper triangular. Now rewrite Equation 2 in vector form as follows:

$$\mathbf{X} - \vec{\mu} = B^T(\mathbf{X} - \vec{\mu}) + S\mathbf{W}$$

where $\mathbf{W} = (W_1, \ldots, W_n)$ is a vector of all the noise terms. Let $\mathbf{E}$ be the innovations or residuals, i.e., the differences (due to noise) between the values of $\mathbf{X}$ actually realized and those predicted by the linear model:

$$\mathbf{E} \overset{\text{def}}{=} S\mathbf{W} = (I - B^T)(\mathbf{X} - \vec{\mu}).$$

Since B is strictly upper triangular, $(I - B^T)$ is invertible, so we may write

$$\mathbf{X} - \vec{\mu} = (I - B^T)^{-1}\mathbf{E} = U^T\mathbf{E} = U^T S^T \mathbf{W}$$

where we have defined $U^T \overset{\text{def}}{=} (I - B^T)^{-1}$, so $U = (I - B)^{-1}$. Finally, we have

$$\Sigma = \text{Var}[\mathbf{X}] = \text{Var}[\mathbf{X} - \vec{\mu}] = \text{Var}[U^T S^T \mathbf{W}] = U^T S^T \text{Var}[\mathbf{W}] S U = (U^T S^T)(SU) = U^T D U.$$

(This result also holds if we use Equation 3 instead.)

Now we compute the global mean, $\vec{\mu}$. If we use Equation 2 (i.e., subtract off the parents' means), $\vec{\mu}$ is just the local means stacked together. To see this, consider the following example, where X_1 is the parent of X_2.

$$E\left[X_2\right] = E\left[E\left[X_2|X_1\right]\right] = E\left[\mu_2 + b(X_1 - \mu_1)\right] = \mu_2$$

If instead we use Equation 3, we need to traverse the graph in topological order to compute $\vec{\mu}$. In our simple example we have

$$E\left[X_2\right] = E\left[E\left[X_2|X_1\right]\right] = E\left[\mu_2 + b_{12}X_1\right] = b_{12}\mu_1 + \mu_2$$

4.2 Vector case

In this case, D and S will be block diagonal, and B will be block upper triangular. For example, in Figure 1, we have

$$D = \begin{pmatrix} \Sigma_{X_1} & 0 & 0 \\ 0 & \Sigma_{X_2} & 0 \\ 0 & 0 & \Sigma_{X_3} \end{pmatrix} = \begin{pmatrix} \sigma_{X_{11}X_{11}} & \sigma_{X_{11}X_{12}} & 0 & 0 & 0 & 0 \\ \sigma_{X_{12}X_{11}} & \sigma_{X_{12}X_{12}} & 0 & 0 & 0 & 0 \\ 0 & 0 & \sigma_{X_{21}X_{21}} & \sigma_{X_{21}X_{22}} & 0 & 0 \\ 0 & 0 & \sigma_{X_{22}X_{21}} & \sigma_{X_{22}X_{22}} & 0 & 0 \\ 0 & 0 & 0 & 0 & \sigma_{X_{31}X_{31}} & \sigma_{X_{31}X_{32}} \\ 0 & 0 & 0 & 0 & \sigma_{X_{32}X_{31}} & \sigma_{X_{32}X_{32}} \end{pmatrix}$$

where we have ordered the nodes as $X_{11}, X_{12}, X_{21}, X_{22}, X_{31}, X_{32}$. The global matrix of weights is

$$B = \begin{pmatrix} 0 & \cdot & G^T \\ & 0 & H^T \\ & & 0 \end{pmatrix} = \begin{pmatrix} 0 & 0 & \cdot & \cdot & G_{11} & G_{21} \\ 0 & 0 & \cdot & \cdot & G_{12} & G_{22} \\ 0 & 0 & 0 & 0 & H_{11} & H_{21} \\ 0 & 0 & 0 & 0 & H_{12} & H_{22} \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

where $\cdot$ represents a value that happens to be 0 (because $\mathbf{X}$ does not connect to $\mathbf{Y}$), whereas 0 represents a value that must be 0 (because of the topological ordering).

4.3 Conditional independence properties of Gaussian graphical models

In this section we will show that

$$X_i \perp X_j | \text{ (the rest)} \iff K_{ij} = 0 \tag{5}$$

where $K = \Sigma^{-1}$ is the inverse covariance matrix (also called the precision matrix) of the joint distribution, and "the rest" means all the other nodes [Whi90, Edw95].

We can represent the joint distribution over all the nodes as

$$\phi(\mathbf{x}; p, \vec{\mu}, \Sigma) = p \times \exp -\tfrac{1}{2}(\mathbf{x} - \vec{\mu})^T \Sigma^{-1} (\mathbf{x} - \vec{\mu}) \tag{6}$$

where

$$p = (2\pi)^{-|\mathbf{x}|/2} |\Sigma|^{-\frac{1}{2}}$$

is a normalizing constant to ensure $\int_{\mathbf{x}} \phi(\mathbf{x}; p, \vec{\mu}, \Sigma) = 1$. p, $\vec{\mu}$ and Σ are called the moment characteristics of the distribution.

Expanding out the quadratic form and collecting terms, we can rewrite this as follows.

$$\phi(\mathbf{x}) = \exp\left[g + \mathbf{x}^T \mathbf{h} - \tfrac{1}{2}\mathbf{x}^T K \mathbf{x}\right],$$

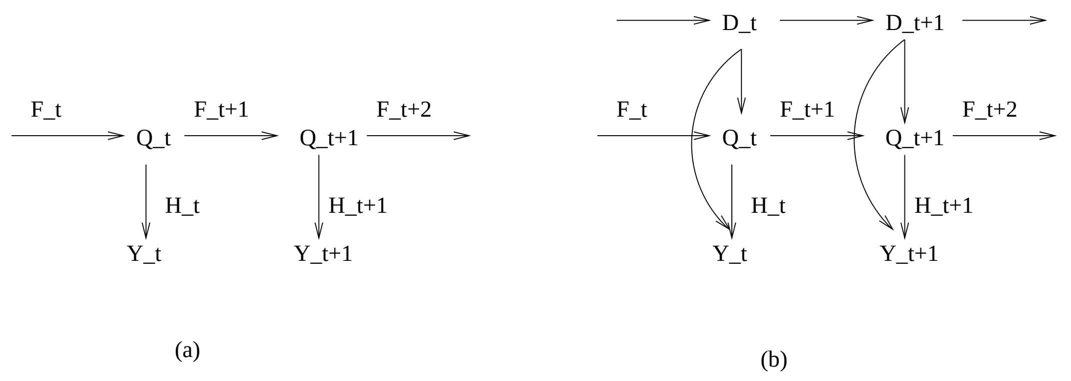


Figure 2: (a) The Kalman Filter represented as a Dynamic Bayesian Network (DBN). The hidden state variables are $\mathbf{Q}_t$ and the observation variables are $\mathbf{Y}_t$. The noise terms in the state evolution and sensor models are implicit in the fact that the distributions of $\mathbf{Q}_t|\mathbf{Q}_{t-1}$ and $\mathbf{Y}_t|\mathbf{Q}_t$ are Gaussian. That is, we do not have nodes for the noise variables. If the state variables are discrete, this model is called a Hidden Markov Model (HMM). (b) A discrete node D_t has been added to model a switching Kalman filter.

In exponential family terminology, g, $\mathbf{h}$ and K are called the canonical characteristics, and are related to the moment characteristics as follows:

$$
\begin{aligned}
K &= \Sigma^{-1} \\
\mathbf{h} &= \Sigma^{-1}\vec{\mu} \\
g &= \log p - \tfrac{1}{2}\vec{\mu}^T\Sigma^{-1}\vec{\mu}
\end{aligned}
$$

where $|\mathbf{x}| = n$. Finally, we can write the above equation in scalar form:

$$\phi(\mathbf{x}) = \exp\left(g + \sum_{i=1}^{n} h_i x_i - \tfrac{1}{2}\sum_i\sum_j K_{ij}x_i x_j\right)$$

Using Dawid's theorem, which states that $X \perp Y|Z$ if the joint density can be factored as

$$f_{X,Y,Z}(x,y,z) = g(x,z)h(y,z)$$

we prove Equation 5.

5 Example of hybrid DBNs: switching Kalman filters

A Dynamic Bayesian Network [DW91, Kja92] is a BN used to model a temporal stochastic process. It can be be created by specifying the network (structure and parameters) for two consecutive "time slices", and then "unrolling" it into a static network of the required size. For example, in Figure 2 we show how to represent a linear dynamical system subject to Gaussian noise.

$\mathbf{Q}_t$ represents the hidden state of the system at time t, which is assumed to evolve according to the following linear equation:

$$\mathbf{Q}_t = F_t \mathbf{Q}_{t-1} + G_t \mathbf{W}_t$$

where $\mathbf{W}_t \sim N(\mathbf{0}, I)$ is a white noise random vector whose distribution is stationary. Thus we set the parameters of $\mathbf{Q}_t$ to be $\vec{\mu} = \mathbf{0}$, $\Sigma = G_t G_t^T$ and $B = F_t$. $\mathbf{Y}_t$ represents the observation vector at time t, which is assumed to be a linear function of the hidden state:

$$\mathbf{Y}_t = H_t \mathbf{Q}_t + J_t \mathbf{V}_t$$

where $\mathbf{V}_t \sim N(\mathbf{0}, I)$ (and is uncorrelated with $\{\mathbf{W}_t\}$). So we set the parameters of $\mathbf{Y}_t$ to be $\vec{\mu} = \mathbf{0}$, $\Sigma = H_t H_t^T$, and $B = H_t$.

The task of computing the probability of the hidden state given all the past observations, $\Pr(\mathbf{Q}_t | \mathbf{y}_t, \ldots, \mathbf{y}_0)$, is called filtering, and the classical algorithm for it was invented by Kalman. The task of computing the probability of the hidden state given *all* the observations, $\Pr(\mathbf{Q}_t | \mathbf{y}_0, \ldots, \mathbf{y}_n)$, is called smoothing, and the classical algorithm for it was invented by Rauch. See [BSF88, BSL93] for details.

The Kalman filter was developed for tracking point-like objects, such as planes and missiles. It is reasonable to represent the state (e.g., position and velocity) of a missile with a single node, $\mathbf{Q}_t$. However, if we want to track more complicated objects, such as people, we would like to represent the complex internal spatial structure of the object with an entire network (e.g., with one node per limb). Since $\mathbf{Q}_t$ is a jointly Gaussian rv, it can be replaced by an entire subnetwork, which also encodes a jointly Gaussian rv. The resulting network is equivalent to the one in Figure 2(a), except that the various matrices are now sparse. However, we claim that it is easier to exploit the conditional independence assumptions (for learning and possibly for speeding up inference) if they are encoded graphically as a Bayes net, rather than encoded implicitely in a sparse matrix.

We can imagine that the dynamical system has different "modes", which we can represent by means of a discrete variable, as shown in Figure 2(b). For example, we might have one set of parameters for when a plane is taking off, another for when it is cruising, etc. This is sometimes called a jump-linear system, and the corresponding inference algorithm is the switching Kalman filter. The state evolution equation is

$$\mathbf{Q}_t = F[D_t] \mathbf{Q}_{t-1} + G[D_t] \mathbf{W}_t$$

and the sensor model equation is

$$\mathbf{Y}_t = H[D_t] \mathbf{Q}_t + J[D_t] \mathbf{V}_t$$

We briefly discuss the computational issues involved in performing inference in hybrid DBNs in Section 6.3.

6 Inference

We shall discuss how to perform inference in hybrid networks using the join tree algorithm [LS88, Jen96], which works on undirected Markov trees. (Similar results have been derived for directed trees [Pea88, PS91, AA96, DM95].) We start by reviewing the discrete case, and then show how to generalize this to handle Gaussian networks, and finally hybrid networks [LS88, LW89, Lau92, Ole93, Lau96].

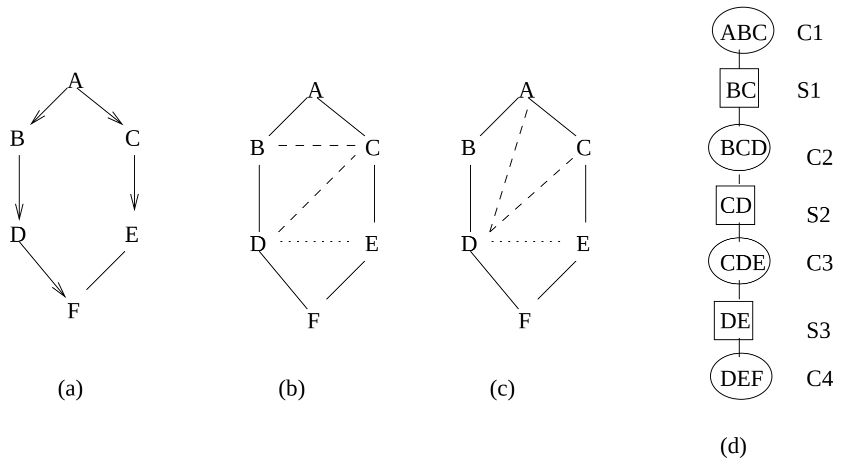


Figure 3: (a) The original DAG. (b) and (c) show two different moralized, triangulated graphs. Dotted arcs denote arcs introduced during moralization. Dashed arcs denote arcs introduced during triangulation. (d) The join tree produced from (b). Squares denote separators, ellipses denote cliques. The different triangulations correspond to the elimination orders f, e, d, c, b, a and f, e, c, b, d, a respectively. For example, in the first ordering, when we eliminate e, we ensure that all its neighbors (c, d, f) which are lower than it in the ordering (c, d) are mutually connected by adding the $c - d$ edge. Similarly, eliminating d will connect b and c.

6.1 Pure discrete case

We will associate a "potential" function with each clique, which is the joint probability of its variables and the evidence. (In the discrete case, potential functions can be represented as multidimensional tables; we discuss the continuous case later.) If we also associate a potential with each separator (a separator is the intersection of the two cliques on the ends of the arc to which the separator is attached), we can write the complete joint probability distribution as

$$\Pr_U = \frac{\prod_{V \in \mathcal{C}} \Pr_V}{\prod_{S \in \mathcal{S}} \Pr_S}$$

where $\mathcal{C}$ is the set of cliques, $\mathcal{S}$ is the set of separators, and $\Pr_U$ is the joint on the whole "universe", $U = (X_1, \ldots, X_n)$. For example, referring to Figure 3(d), and assuming we have no evidence, we have

$$\frac{\Pr(A,B,C)\Pr(B,C,D)\Pr(C,D,E)\Pr(D,E,F)}{\Pr(B,C)\Pr(C,D)\Pr(D,E)} = \Pr(A|B,C)\Pr(B|C,D)\Pr(C|D,E)\Pr(D,E,F)$$

This follows from the "separation implies independence" property of undirected graphical models, e.g., since D and E separate C from F in the moralized, triangulated graph, $\Pr(C|D,E)\Pr(D,E,F) = \Pr(C,D,E,F)$; continuing in this way we have $\Pr(B|C,D)\Pr(C,D,E,F) = \Pr(B,C,D,E,F)$ and finally $\Pr(A|B,C)\Pr(B,C,D,E,F) = \Pr(A,B,C,D,E,F)$.

Now suppose some evidence arrives on node D e.g., we observe its value to be d. We need to update all the potentials to reflect this fact. For each variable X, we find a clique C which contains X and its parents; call this clique the representative for X (we say that X is assigned to C). For example, the representative of D must be $C2$. We update $\Pr_{C2}$ by assigning zero probability to all combinations which are inconsistent with

$D = d$. (In the discrete case, we just set the table entries to 0; we discuss the continuous case later.) This gives us $\Pr(B, C, D, \mathbf{e}) = \Pr^*(B, C, D)$, where $\mathbf{e}$ is the evidence (namely the event $D = d$). We now need to propogate this change to all the other potentials. The idea is that each clique sends a "message" to its neighbors, which they "absorb" (i.e., they update their potential to reflect the new piece of information in a way which we shall explain shortly). When a clique node has received messages from all its neighbors bar one, it may send send a message to that one. In this way, every clique eventually gets updated, and global consistency is restored.

A centralized version of the message passing protocol as as follows: pick a root or pivot node R, thereby inducing directionality on the tree. In the first pass, all nodes send messages to R after receiving from their children (i.e., in postorder); in the second pass, the root sends messages down to the leaves (i.e., in preorder). The first pass is sometimes called the "collect evidence" phase, and the second pass is called the "distribute evidence" phase. If we let $\mathbf{e}_i^-$ denote all the evidence in the subtree rooted at C_i, and $\mathbf{e}_i^+$ denote all the rest of the evidence ("above" C_i), then after the first pass each clique potential contains $\Pr(C_i, \mathbf{e}_i^-)$, and after the second pass, each clique potential contains $\Pr(C_i, \mathbf{e}_i^-, \mathbf{e}_i^+) = \Pr(C_i, \mathbf{e})$. Hence after two passes we can recover the posterior marginal of a family by finding any clique that contains it, and marginalizing out all the other variables in that clique. (To compute the marginal on a set of variables which is not contained within a clique, see [Xu95].) If the tree has a caterpillar-like chain structure (e.g., Figure 2), this algorithm becomes identical to the forwards-backwards algorithm for HMMs [SHJ96].

The only things that remain to be specified are how we initialize and update the clique potentials. We initialize the potential for clique C by multiplying together all the CPTs for all the variables which are assigned to C. For example, if we assign A, B and C to $C1$, we get $\Pr(C1) = \Pr(A)\Pr(B|A)\Pr(C|A) = \Pr(A, B, C)$. Similarly, if we assign D to $C2$, we get $\Pr(C2) = \Pr(D|B)$. Separators are initialized to 1. We then do one forward pass and one backward pass, and the result will be that each clique potential contains the joint probability over its member variables.

The absorption/update process is best illustrated by example. Referring to Figure 3, suppose we observe that $D = d$ and compute $\Pr(B, C, D, \mathbf{e})$. To update the potential on $C3$, $\Pr(C, D, E)$, we write $\Pr(C, D, E, \mathbf{e}) = \Pr(E|C, D)\Pr(C, D, \mathbf{e})$, which follows since the conditional probability $\Pr(E|C, D)$ is a fixed constant independent of the evidence $\mathbf{e}$. $\Pr(C, D, \mathbf{e})$ is the potential on the separator $S2$ and can be computed by marginalization: $\Pr(C, D, \mathbf{e}) = \sum_B \Pr(B, C, D, \mathbf{e})$. The conditional probability can be computed as $\Pr(E|C, D) = \frac{\Pr(E,C,D)}{\Pr(C,D)}$. In summary, if W and V are neighbors with separator S, and W absorbs from V, we must perform the following steps:

- Calculate $\Pr_S^* = \sum_{V \setminus S} \Pr_V$.
- Give S the new potential $\Pr_S^*$.
- Give W the new potential $\Pr_W^* = \Pr_W \frac{\Pr_S^*}{\Pr_S}$.

We require that if $\Pr_S(\mathbf{x}) = 0$ for some value $\mathbf{x}$, then $\Pr_W(\mathbf{x}) = 0$ also, so we can set $\Pr_W^*(\mathbf{x}) = 0/0 = 0$. (A tree which satisfies this requirement is called supportive.)

6.2 Pure Gaussian case

In the discrete case, the potential over a clique can be represented as a table. In the Gaussian case, the potential can be represented as a Gaussian function in either moment or canonical form. It turns out that some operations are easier to express in terms of canonical characteristics and others are easier to express in terms of moment characteristics.

6.2.1 Initialization

In the Lauritzen and Spiegelhalter algorithm, each clique potential is initialized to be the product of the conditional distributions of all the nodes that have been assigned to that clique. (Each node is assigned to exactly one clique, which must contain its family.) After one forwards and one backwards pass over the tree, each clique potential will be the joint distribution over all its member variables. (Let us call these "virgin potentials".) We are then ready to incorporate evidence.

Unfortunately, we may not be able to represent the initial potential (before the initial forwards and backwards pass over the tree) in moment form. The reason is that the mean may depend on the values of some variables which have not been assigned to the clique, e.g., if only one node has been assigned to a clique, the initial potential will be of the form $f(X|Y)$; here, the mean depends on Y.

Hence we represent the initial potentials using canonical characteristics, and only convert to moment form where necessary (as descibed in Section 6.3).

For a vector node, the conditional distribution has the form

$$\begin{aligned} f(\mathbf{x}|\mathbf{z}) &= c\exp\left[-\tfrac{1}{2}\left((\mathbf{x}-\vec{\mu}-B^T\mathbf{z})^T\Sigma^{-1}(\mathbf{x}-\vec{\mu}-B^T\mathbf{z})\right)\right] \\ &= \exp\left[-\tfrac{1}{2}\begin{pmatrix}\mathbf{x} & \mathbf{z}\end{pmatrix}\begin{pmatrix}\Sigma^{-1} & -\Sigma^{-1}B^T \\ -B\Sigma^{-1^T} & B\Sigma^{-1}B^T\end{pmatrix}\begin{pmatrix}\mathbf{x}\\ \mathbf{z}\end{pmatrix} + \begin{pmatrix}\mathbf{x} & \mathbf{z}\end{pmatrix}\begin{pmatrix}\Sigma^{-1}\vec{\mu}\\ -B\Sigma^{-1}\vec{\mu}\end{pmatrix} - \tfrac{1}{2}\vec{\mu}^T\Sigma^{-1}\vec{\mu} + \log c\right] \end{aligned}$$

where $c = (2\pi)^{-n/2}|\Sigma|^{-\frac{1}{2}}$. Hence we set the canonical characteristics to

$$\begin{aligned} g &= -\tfrac{1}{2}\vec{\mu}^T\Sigma^{-1}\vec{\mu} - \frac{n}{2}\log(2\pi) - \tfrac{1}{2}\log|\Sigma| \\ \mathbf{h} &= \begin{pmatrix}\Sigma^{-1}\vec{\mu}\\ -B\Sigma^{-1}\vec{\mu}\end{pmatrix} \\ K &= \begin{pmatrix}\Sigma^{-1} & -\Sigma^{-1}B^T \\ -B\Sigma^{-T} & B\Sigma^{-1}B^T\end{pmatrix} \end{aligned}$$

This generalizes the result in [Lau92] to the vector case. In the scalar case, $\Sigma^{-1} = 1/\sigma$, $\vec{\mu} = \mu$, $B = \mathbf{b}$ and $n = 1$, so the above becomes

$$\begin{aligned} g &= \frac{-\mu^2}{2\sigma} - \tfrac{1}{2}\log(2\pi\sigma) \\ \mathbf{h} &= \frac{\mu}{\sigma}\begin{pmatrix}1\\ -\mathbf{b}\end{pmatrix} \\ K &= \frac{1}{\sigma}\begin{pmatrix}1 & -\mathbf{b}^T \\ -\mathbf{b} & \mathbf{b}\mathbf{b}^T\end{pmatrix}. \end{aligned}$$

Once we have the canonical characteristics, we can compute the initial potentials for each clique by multiplying together the potentials associated with each variable which is assigned to this clique. Unfortunately, we cannot convert these canonical characteristics to moment characteristics because K is not of full rank, and hence is not invertible. (This is easy to see in the scalar case, since K contains an outer product and hence is of rank 1.)

6.2.2 Entering evidence

If we observe that a continuous variable $\mathbf{Y}$ takes on a specific value $\mathbf{y}$, we must modify the potentials of all the cliques/separators that contain $\mathbf{Y}$, since their dimensionality will be reduced. Let the clique contain $\mathbf{X}$ and $\mathbf{Y}$. The new potential is

$$\begin{aligned}
\phi^*(\mathbf{x}) &= \exp[g + (\,\mathbf{x}^T \quad \mathbf{y}^T\,) \begin{pmatrix} \mathbf{h}_X \\ \mathbf{h}_Y \end{pmatrix} - \tfrac{1}{2} (\,\mathbf{x}^T \quad \mathbf{y}^T\,) \begin{pmatrix} K_{XX} & K_{XY} \\ K_{YX} & K_{YY} \end{pmatrix} \begin{pmatrix} \mathbf{x} \\ \mathbf{y} \end{pmatrix} \\
&= \exp[(g + \mathbf{h}_Y^T \mathbf{y} - \tfrac{1}{2} \mathbf{y}^T K_{YY} \mathbf{y}) + \mathbf{x}^T (\mathbf{h}_X - K_{XY} \mathbf{y}) - \tfrac{1}{2} \mathbf{x}^T K_{XX} \mathbf{x}]
\end{aligned}$$

This generalizes the equation in [Lau92] to the vector case.

We can compute the analogous result for moment characteristics as follows. We will start by just considering the quadratic form

$$Q = (\,\mathbf{x}' - \vec{\mu}_x' \quad \mathbf{y}' - \vec{\mu}_y'\,) \begin{pmatrix} K_{XX} & K_{XY} \\ K_{YX} & K_{YY} \end{pmatrix} \begin{pmatrix} \mathbf{x} - \vec{\mu}_x \\ \mathbf{y} - \vec{\mu}_y \end{pmatrix}$$

Expanding out,

$$\begin{aligned}
Q &= (\mathbf{x} - \vec{\mu}_x)' K_{XX} (\mathbf{x} - \vec{\mu}_x) + 2(\mathbf{x} - \vec{\mu}_x)' K_{XY} (\mathbf{y} - \vec{\mu}_y) + (\mathbf{y} - \vec{\mu}_y)' K_{YY} (\mathbf{y} - \vec{\mu}_y) \\
&= \mathbf{x}' K_{XX} \mathbf{x} - 2\mathbf{x}' K_{XX} \vec{\mu}_x + \vec{\mu}_x' K_{XX} \vec{\mu}_x + 2\mathbf{x}' K_{XY} (\mathbf{y} - \vec{\mu}_y) - 2\vec{\mu}_x' K_{XY} (\mathbf{y} - \vec{\mu}_y) + (\mathbf{y} - \vec{\mu}_y)' K_{YY} (\mathbf{y} - \vec{\mu}_y) \\
&= \mathbf{x}' K_{XX} \mathbf{x} - 2\mathbf{x}' \left(K_{XX} \vec{\mu}_x - K_{XY} (\mathbf{y} - \vec{\mu}_y) \right) + \left(\vec{\mu}_x' K_{XX} \vec{\mu}_x - 2\vec{\mu}_x' K_{XY} (\mathbf{y} - \vec{\mu}_y) + (\mathbf{y} - \vec{\mu}_y)' K_{YY} (\mathbf{y} - \vec{\mu}_y) \right) \\
&\overset{\text{def}}{=} \mathbf{x}' A \mathbf{x} - 2\mathbf{x}' \mathbf{b} + c
\end{aligned}$$

Now we use the following rule, called completing the square:

$$\mathbf{x}^T A \mathbf{x} - 2\mathbf{x}^T \mathbf{b} + c = (\mathbf{x} - A^{-1}\mathbf{b})^T A (\mathbf{x} - A^{-1}\mathbf{b}) + c - \mathbf{b}^T A^{-1} \mathbf{b} \tag{7}$$

to yield $\phi^*(\mathbf{x}) = p^* \times Q(\mathbf{x}; \vec{\mu}, \Sigma)$ where

$$\begin{aligned}
\Sigma &= A^{-1} \\
\vec{\mu} &= A^{-1}\mathbf{b} \\
\log p^* &= \log p - \tfrac{1}{2} \left(c - \mathbf{b}' A^{-1} \mathbf{b} \right)
\end{aligned}$$

6.2.3 Multiplication and division

In the discrete case, we use multiplication and division to update potentials when new evidence arrives: $\Pr_W^* = \Pr_W \frac{\Pr_S^*}{\Pr_S}$, where S is a separator and W is a clique. Notice that $\frac{\Pr_W}{\Pr_S} = \Pr(W|S)$, so we are really computing a conditonal distribution "on the fly" and multiplying in new information.

We can define multiplication and division in the Gaussian case in terms of canonical characteristics, as follows. To multiply $\phi_1(x_1, \ldots, x_k; g_1, \mathbf{h}_1, K_1)$ by $\phi_2(x_{k+1}, \ldots, x_n; g_2, \mathbf{h}_2, K_2)$, we extend them both to the same domain $x_1, \ldots, x_n$ by adding zeros to the appropriate dimensions, and compute

$$(g_1, \mathbf{h}_1, K_1) * (g_2, \mathbf{h}_2, K_2) = (g_1 + g_2, \mathbf{h}_1 + \mathbf{h}_2, K_1 + K_2)$$

The support of the new function is the intersection of the previous supports. Division is similar, except that we define $(\phi_1/\phi_2)(\mathbf{x}) = 0$ if $\phi_1(\mathbf{x}) = 0$.

6.2.4 Marginalization

Let ϕ_W be a potential over a set W of variables. We can compute the potential over a subset $V \subset W$ of variables by marginalizing, denoted $\phi_V = \sum_{W \backslash V} \phi_W$. Let

$$\mathbf{y} = \begin{pmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \end{pmatrix}, \qquad \mathbf{h} = \begin{pmatrix} \mathbf{h}_1 \\ \mathbf{h}_2 \end{pmatrix}, \qquad K = \begin{pmatrix} K_{11} & K_{12} \\ K_{21} & K_{22} \end{pmatrix}$$

with $\mathbf{y}_1$ having dimension p and $\mathbf{y}_2$ having dimension q. It can be shown (by completing the square and using nice properties of multidimensional Gaussians) that

$$\int \phi[\begin{pmatrix} \mathbf{y}_1^T & \mathbf{y}_2^T \end{pmatrix}^T] d\mathbf{y}_1 \quad = \quad \phi[\mathbf{y}_2; \hat{g}, \hat{\mathbf{h}}, \hat{K}]$$

where

$$\begin{aligned} \hat{g} &= g + \tfrac{1}{2}\left(p\log(2\pi) - \log|K_{11}| + \mathbf{h}_1^T K_{11}^{-1}\mathbf{h}_1\right) = g + \tfrac{1}{2}\left(p\log(2\pi) + \log|K_{11}^{-1}| + \mathbf{h}_1^T K_{11}^{-1}\mathbf{h}_1\right) \\ \hat{\mathbf{h}} &= \mathbf{h}_2 - K_{21}K_{11}^{-1}\mathbf{h}_1 \\ \hat{K} &= K_{22} - K_{21}K_{11}^{-1}K_{12} \end{aligned}$$

In the moment case, things are much simpler. We simply extract out the components of $\vec{\mu}$ and Σ which relate to $\mathbf{y}_2$, and change the constant so that it normalizes the new distribution.

6.3 Hybrid case

The only change in the hybrid case is that the potential functions will now be over both continuous and discrete nodes. Essentially we have one set of canonical or moment characteristics for each value of the discrete nodes. All the operations go through as before, except for marginalization. If we marginalize out over some continuous nodes, we can proceed as in Section 6.2.4, once for each value of the discrete nodes. If we marginalize out over some discrete nodes d, but the mean/variance do not depend on $\mathbf{j}$, we just sum the appropriate constants (g or p) for each value of d: this is called strong marginalization. However, if the mean and variance depend on $\mathbf{j}$, we will get a mixture of Gaussians:

$$\sum_{\mathbf{j}} \phi(\mathbf{x}, \mathbf{j}, \mathbf{i}) = \sum_{\mathbf{j}} p \times Q(\mathbf{x}; \vec{\mu}(\mathbf{j}, \mathbf{i}), \Sigma(\mathbf{j}, \mathbf{i}))$$

This cannot be simplified any further, and must be kept as a list of terms. We would therefore like to arrange things so that we integrate out all continuous nodes before the discrete nodes on which they depend, e.g., we write $\sum_i \int_{\mathbf{x}} f(\mathbf{x}, \vec{\mu}(i), \Sigma(i))$ rather than $\int_{\mathbf{x}} \sum_i f(\mathbf{x}, \vec{\mu}(i), \Sigma(i))$. This can be achieved by ensuring that all the continuous nodes are eliminated before their discrete ancestors. Such a node elimination ordering is called a strong triangulation, c.f.[JJD94].

Unfortunately, in the case of hybrid DBNs, the need to eliminate all the continuos nodes before their discrete ancestors clashes with our desire to eliminate all the nodes in slice t before we eliminate any in slice $t+1$. If we don't do strong triangulation, the number of mixture components becomes exponential in the length of the sequence. The standard approach (see e.g., [TSM85, BSL93, Kim94, WH97]) is to "collapse" the mixture into k components. If $k = 1$, this corresponds to computing the "weak" moments:

$$\begin{aligned}
\hat{p}(\mathbf{i}) &= \sum_{\mathbf{j}} p(\mathbf{i},\mathbf{j}) \\
\hat{\vec{\mu}}(\mathbf{i}) &= \sum_{\mathbf{j}} \vec{\mu}(\mathbf{i},\mathbf{j}) p(\mathbf{i},\mathbf{j})/\hat{p}(\mathbf{i}) \\
\hat{\Sigma}(\mathbf{i}) &= \sum_{\mathbf{j}} \Sigma(\mathbf{i},\mathbf{j}) p(\mathbf{i},\mathbf{j})/\hat{p}(\mathbf{i}) + \sum_{\mathbf{j}} \left(\vec{\mu}(\mathbf{i},\mathbf{j}) - \hat{\vec{\mu}}(\mathbf{i})\right) \left((\vec{\mu}(\mathbf{i},\mathbf{j}) - \hat{\vec{\mu}}(\mathbf{i})\right)^T p(\mathbf{i},\mathbf{j})/\hat{p}(\mathbf{i})
\end{aligned}$$

These will give the "correct" mean and variance:

$$\begin{aligned}
\Pr(\mathbf{I} = \mathbf{i}) &= \hat{p}(\mathbf{i}) \\
E\left[\mathbf{Y}|\mathbf{I} = \mathbf{i}\right] &= E_{\Pr(\mathbf{J}=\mathbf{j}|\mathbf{I}=\mathbf{i})}\left[E\left[\mathbf{Y}|\mathbf{I},\mathbf{J}\right]|\mathbf{i} = \mathbf{i}\right] \\
&= \sum_{\mathbf{j}} \vec{\mu}(\mathbf{i},\mathbf{j}) \Pr(\mathbf{J} = \mathbf{j}|\mathbf{I} = \mathbf{i}) \\
\mathrm{Var}[\mathbf{Y}|\mathbf{I} = \mathbf{i}] &= E\left[\mathrm{Var}[\mathbf{Y}|\mathbf{I},\mathbf{J}]|\mathbf{i} = \mathbf{i}\right] + \mathrm{Var}\left[E\left[\mathbf{Y}|\mathbf{I},\mathbf{J}\right]|\mathbf{i} = \mathbf{i}\right] \\
&= E\,\Sigma(\mathbf{i},\mathbf{j}) + E\left[(\vec{\mu}(\mathbf{i},\mathbf{j}) - E\,\vec{\mu}(\mathbf{i},\mathbf{j}))\,(\vec{\mu}(\mathbf{i},\mathbf{j}) - E\,\vec{\mu}(\mathbf{i},\mathbf{j}))^T\right]
\end{aligned}$$

Lauritzen [Lau96] shows that this is the best approximation (in the KL sense) if $k = 1$.

7 Learning

In this section, we discuss how to find the Maximum Likelihood Estimates (MLEs) of the parameters associated with each node. We assume that we have a set of N training examples, where each example assigns a value to every node in the network (this is called the fully observable case). In Section 7.2, we address the issue of what to do when the values of some variables are unknown.

If we assume the parameters of node X_i, Θ_i, are independent of those of all other nodes, we can maximize the Θ_i's separately. Further, the only terms in the joint distribution that depend on Θ_i involve X_i and its parents, so we just need to compute the sufficient statistics for each family.

Discrete, linear Gaussian and mixtures of linear Gaussian distributions are all in the exponential family [DeG70, Bun94, Lau96]; hence the size of the sufficient statistics we need to keep is equal to the size of the parameter vector (and independent of N).

7.1 Fully observable case

7.1.1 Discrete case

If X_i is a discrete variable, the parameter vector is $\Theta_i = (\theta_{ijk}) = (\Pr(X_i = k|\mathrm{Pa}(X_i) = j))$, which is just a table of numbers. The sufficient statistics are N_{ijk}, the number of times the event $(X_i = k, \mathrm{Pa}(X_i) = j)$

occurs in the training set. Since

$$\theta_{ijk} = \frac{\Pr(X_i = k, \mathrm{Pa}(X_i) = j)}{\Pr(\mathrm{Pa}(X_i) = j)} \approx \frac{\frac{1}{N} N_{ijk}}{\frac{1}{N} N_{ij}}$$

where $N_{ij} = \sum_k N_{ijk}$, the MLE is $\widehat{\theta_{ijk}} = N_{ijk}/N_{ij}$.

7.1.2 Gaussian case

The approach we will adopt is to model the joint distribution over a node and its parents (forming family X) as a MVG, compute its sufficient statistics and then find its MLE parameters. We discuss the simplest case below; for more general results, see [Mur98].

The sufficient statistics for an MVG after seeing N examples are $\mathbf{s}_N \stackrel{\text{def}}{=} \sum_{l=1}^N \mathbf{x}_l$ and $Q_N \stackrel{\text{def}}{=} \sum_{l=1}^N \mathbf{x}_l \mathbf{x}_l^T$, since

$$\widehat{\vec{\mu}_N} = \frac{1}{N}\mathbf{s}_N = \frac{1}{N}\sum_{l=1}^N \mathbf{x}_l \tag{8}$$

and

$$\begin{aligned}
\widehat{\Sigma_N} &= \frac{1}{N}\sum_{l=1}^N (\mathbf{x}_l - \widehat{\vec{\mu}_N})(\mathbf{x}_l - \widehat{\vec{\mu}_N})^T \\
&= \frac{1}{N}\left[\left(\sum_{l=1}^N \mathbf{x}_l \mathbf{x}_l^T\right) - \left(\sum_{l=1}^N \mathbf{x}_l\right)\widehat{\vec{\mu}_N}^T - \widehat{\vec{\mu}_N}\left(\sum_{l=1}^N \mathbf{x}_l^T\right) + N\widehat{\vec{\mu}_N}\widehat{\vec{\mu}_N}^T\right] \\
&= \frac{1}{N}Q_N - \widehat{\vec{\mu}_N}\widehat{\vec{\mu}_N}^T .
\end{aligned}$$

It is simple to update the sufficient statistics when we see the next example, $\mathbf{x}_{N+1}$.

To compute the parameters of a node given the sufficient statistics of its family, we use linear regression as follows. Let $\mathbf{X}_1$ represent the child and $\mathbf{X}_2$ the parents, i.e.,

$$\mathbf{X} = \begin{pmatrix} \mathbf{X}_1 \\ \mathbf{X}_2 \end{pmatrix}, \quad \mu_{\mathbf{X}} = \begin{pmatrix} \mu_{\mathbf{X}_1} \\ \mu_{\mathbf{X}_2} \end{pmatrix}, \quad \Sigma_{\mathbf{X}} = \begin{pmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{pmatrix}$$

Then the conditional density of $\mathbf{X}_1$ given $\mathbf{X}_2$ is a MVG with

$$\mu_{\mathbf{X}_1|\mathbf{X}_2} = E[\mathbf{X}_1|\mathbf{X}_2 = \mathbf{x}_2] = \mu_{\mathbf{X}_1} + \Sigma_{12}\Sigma_{22}^{-1}(\mathbf{x}_2 - \mu_{\mathbf{x}_2}) \tag{9}$$

and

$$\Sigma_{\mathbf{X}_1|\mathbf{X}_2} = \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}. \tag{10}$$

Hence the local parameters for the node are given by

$$\begin{aligned} B &= \Sigma_{YZ}\Sigma_{ZZ}^{-1} \\ \vec{\mu} &= \vec{\mu}_Y - B\vec{\mu}_Z \\ \Sigma &= \Sigma_{YY} - B\Sigma_{ZY} \end{aligned} \tag{11}$$

B can then be broken up into its individual blocks, one for each parent.

7.1.3 Hybrid case

The exact posterior distribution of a hybrid potential will be a mixture of Gaussians. It can be approximated by a single Gaussian by performing weak marginalization. In general, this can be an arbitrarily bad approximation, since we may be replacing a multimodal distribution with a unimodal one. However, let us suppose that the error introduced by this step is at most ϵ. Then the results in [BK98b, BK98a] show that for a hybrid DBN, the total error will be a function of ϵ and γ, the mixing rate of the Markov chain, but independent of t. An alternative approach to learning hybrid DBNs, taken in [GH96], is to maximize an exact lower bound on the likelihood, produced by considering a tractable approximation to the original structure.

7.2 Partially observable case

If we do not observe the value of every node in each training case, there is no longer a closed form expression for the MLE. In this section, we investigate two methods for learning under these circumstances. Both methods make many passes over the training data, and update the parameters at the end of each pass, until they reach a local maximum in likelihood space; hence they are batch methods. However, it is easy to convert them to incremental (online) versions, which update the parameters after seeing a subset of the training set (see e.g., [NH98] for incremental EM and [BC94] for incremental gradient descent).

7.2.1 EM

The basic idea of the Expectation Maximization (EM) algorithm is to "fill in" the missing values with their expected values (expectation w.r.t. the current set of parameters), and to use these Expected Sufficient Statistics (ESS) when computing the MLE. The parameters are then set to their MLE values, and the process repeats until the likelihood stops increasing (it can be proved that EM will converge to a local maximum).

In the discrete case, the ESS are

$$E\left[N_{ijk}\right] = \sum_l \Pr(X_i = k, \mathrm{Pa}(X_i) = j|\mathbf{e}_l) = \sum_l \frac{\Pr(X_i = k, \mathrm{Pa}(X_i) = j, \mathbf{e}_l)}{\Pr(\mathbf{e}_l)}.$$

In the Gaussian case, the ESS are

$$\mathbf{s}_N = \sum_l E[\mathbf{X}_l|\mathbf{e}_l] \text{ and } Q_N = \sum_l E[\mathbf{X}_l\mathbf{X}_l'|\mathbf{e}_l]$$

since

$$\mathrm{Var}[X] = E[XX'] - E[X]E[X]'$$

For the hybrid case, we just compute bot kinds of ESS for each discrete parent value.

We now present the EM algorithm in detail.

1. Choose (random) starting values for the parameters $B, \vec{\mu}, \Sigma$ for each node. A broad covariance is a good idea, so that samples far from the mean are not assigned unduly low likelihood.
2. Repeat
   - (a) Reset the ESS for each node.
   - (b) Reset the log-likelihood: $L = 0$.
   - (c) For each training case $\mathbf{e}$
     - i. Update the log-likelihood: $L += \log \Pr(\mathbf{e})$.
     - ii. Compute the posterior marginal over each family given the evidence.
     - iii. Update the ESS for each family.
   - (d) Compute the MLE of the parameters for each family given the ESS.
3. Until L converges.

Steps 2(c)i and 2(c)ii can be computed using the inference algorithms we discussed earlier.

7.2.2 Gradient descent

It is possible to compute an expression for the gradient of the log-likelihood [XJ96, BKRK97] and hence to use gradient-based learning methods. However, we must maintain constraints on the parameters. In particular, for continuous nodes, Σ must remain symmetric and positive definite, and for discrete nodes, θ_{ijk} must lie inside the unit cube and on the surface $\sum_k \theta_{ijk} = 1$. The best way to maintain the constraints is to reparamaterize the problem, solve the problem in the unconstrained space, and convert back. For the CPT entries, we can learn the parameters of a softmax function [BC94]. For Σ, we can learn the parameters of $\Sigma^{\frac{1}{2}}$, which is unconstrained, and at the end set $\Sigma = \Sigma^{\frac{1}{2}}\Sigma^{\frac{1}{2}}$.

EM, while technically a first order method, often does better than nominally faster gradient-based methods, such as conjugate gradient or quasi-Newton [XJ96]. This is primarily because EM avoids a line-search at each iteration, which is expensive since it requires computing the log-likelihood at many points along the line, each such evaluation requiring a call to the inference engine. We have yet to see how techniques such as Levenberg-Marquardt (which approximate the Hessian yet don't require a line search) perform.

7.3 Using priors to compute the MAP estimate

To avoid overfitting when we have too little training data, we can use priors, and compute the MAP estimates instead of the ML estimates. A suitable prior for discrete nodes is the Dirichlet prior, which has a simple intuitive interpretation in terms of pseudo counts: we just imagine that we have seen a certain number N'_{ijk} of cases of the event $(X_i = k, \mathrm{Pa}(X_i) = k)$, and add these to our real counts. For the vector Gaussian case, things are a little more complicated. It is simpler to associate a prior with the MVG distribution $N(\vec{\mu}_F, \Sigma_F)$ on the family F, rather than with the parameters $(\vec{\mu}_X, \Sigma_X, B_X)$ of the node itself. A suitable prior is the Normal-Wishart [GH94, DeG70]. This can be important since it takes a lot of data to ensure $\widehat{\Sigma}$ is positive definite.

References

[AA96] S. Alag and A. Agogino. Inference using message propogation and topology transformation in vector Gaussian continuous networks. In *UAI*, 1996.

[BC94] P. Baldi and Y. Chauvin. A smooth learning algorithm for hidden Markov models. *Neural Computation*, 6:305–316, 1994.

[BFGK96] C. Boutilier, N. Friedman, M. Goldszmidt, and D. Koller. Context-specific independence in bayesian networks. In *UAI*, 1996.

[BK98a] X. Boyen and D. Koller. Approximate learning of dynamic models. In *NIPS-11*, 1998.

[BK98b] X. Boyen and D. Koller. Tractable inference for complex stochastic processes. In *UAI*, 1998.

[BKRK97] J. Binder, D. Koller, S. J. Russell, and K. Kanazawa. Adaptive probabilistic networks with hidden variables. *Machine Learning*, 29:213–244, 1997.

[BSF88] Y. Bar-Shalom and T. Fortmann. *Tracking and data association.* Academic Press, 1988.

[BSL93] Y. Bar-Shalom and X. Li. *Estimation and Tracking: Principles, Techniques and Software.* Artech House, 1993.

[Bun94] W. L. Buntine. Operations for learning with graphical models. *J. of AI Research*, pages 159–225, 1994.

[CDLS99] R. G. Cowell, A. P. Dawid, S. L. Lauritzen, and D. J. Spiegelhalter. *Probabilistic Networks and Expert Systems.* Springer, 1999.

[DeG70] M. DeGroot. *Optimal Statistical Decisions.* McGraw-Hill, 1970.

[DM95] E. Driver and D. Morrel. Implementation of continuous Bayesian networks usings sums of weighted Gaussians. In *UAI*, pages 134–140, 1995.

[DW91] Thomas L. Dean and Michael P. Wellman. *Planning and Control.* Morgan Kaufmann, 1991.

[Edw95] D. Edwards. *Introduction to graphical modelling.* Springer, 1995.

[FG96] N. Friedman and M. Goldszmidt. Learning Bayesian networks with local structure. In *UAI*, 1996.

[GH94] D. Geiger and D. Heckerman. Learning Gaussian networks. In *UAI*, volume 10, pages 235–243, 1994.

[GH96] Z. Ghahramani and G. Hinton. Switching state-space models. Technical Report CRG-TR-96-3, Dept. Comp. Sci., Univ. Toronto, 1996.

[Jen96] F. V. Jensen. *An Introduction to Bayesian Networks.* UCL Press, London, England, 1996.

[JJD94] F. V. Jensen, F. Jensen, and S. L. Dittmer. From influence diagrams to junction trees. In *UAI*, 1994.

[Kim94] C-J. Kim. Dynamic linear models with Markov-switching. *J. of Econometrics*, 60:1–22, 1994.

[Kja92] U. Kjaerulff. A computational scheme for reasoning in dynamic probabilistic networks. In *UAI-8*, 1992.

[Lau92] S. L. Lauritzen. Propagation of probabilities, means and variances in mixed graphical association models. *JASA*, 87(420):1098–1108, December 1992.

[Lau96] S. Lauritzen. *Graphical Models.* OUP, 1996.

[LJ99] S. Lauritzen and F. Jensen. Stable local compuation with conditional Gaussian distributions. Technical Report R-99-2014, Dept. Math. Sciences, Allborg Uni., 1999.

[LS88] S. L. Lauritzen and D. J. Spiegelhalter. Local computations with probabilities on graphical structures and their applicatins to expert systems. *J. R. Stat. Soc. B*, B(50):127–224, 1988.

[LW89] S. L. Lauritzen and N. Wermuth. Graphical models for associations between variables, some of which are qualitative and some quantitative. *Annals of Statistics*, 17:31–57, 1989.

[MH97] C. Meek and D. Heckerman. Structure and parameter learning for causal independence and causal interaction models. In *UAI*, pages 366–375, 1997.

[Mur98] K. P. Murphy. Fitting a conditional gaussian distribution. Technical report, U.C. Berkeley, Dept. Comp. Sci, 1998.

[Mur99] K. P. Murphy. A variational approximation for Bayesian networks with discrete and continuous latent variables. In *UAI*, 1999.

[NH98] R. M. Neal and G. E. Hinton. A new view of the EM algorithm that justifies incremental and other variants. In M. Jordan, editor, *Learning in Graphical Models.* MIT Press, 1998.

[Ole93] K. G. Olesen. Causal probabilistic networks with both discrete and continuous variables. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 3(15), 1993.

[Pea88] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann, 1988.

[PS91] M. Peot and R. Shachter. Fusion and propogation with multiple observations in belief networks. *Artificial Intelligence*, 48:299–318, 1991.

[RG99] S. Roweis and Z. Ghahramani. A unifying review of linear gaussian models. *Neural Computation*, 11(2), 1999.

[SHJ96] P. Smyth, D. Heckerman, and M. I. Jordan. Probabilistic independence networks for hidden Markov probability models. Technical Report MSR-TR-96-03, Microsoft Research, 1996.

[SK89] R. Shachter and C. R. Kenley. Gaussian influence diagrams. *Managment Science*, 35(5):527–550, 1989.

[TSM85] D. M. Titterington, A. F. M. Smith, and U. E. Makov. *Statistical analysis of finite mixture distributions.* Wiley, 1985.

[WH97] Mike West and Jeff Harrison. *Bayesian forecasting and dynamic models.* Springer, 1997.

[Whi90] J. Whittaker. *Graphical Models in Applied Multivariate Statistics.* Wiley, 1990.

[XJ96] L. Xu and M. I. Jordan. On convergence properties of the EM algorithm for Gaussian mixtures. *Neural Computation*, 8:129–151, 1996.

[Xu95] H. Xu. Computing marginals for arbitrary subsets from marginal representations in Markov trees. *AI*, 74(1):177–189, 1995.