Pearl's algorithm for vector Gaussian Bayes Nets

Kevin Murphy

14 March 2002

1 Introduction

In [Pea88], Pearl gives the equations for belief propagation in a directed graphical model in which all nodes are scalar Gaussians. The generalization to vector-value nodes can be found in [Ala96]. We state, without proof, the results of [Ala96], with a few modifications. In particular, we use the information (canonical) form to represent λ messages, so that we don't try to invert matrices that might be uninvertible. This problem arises because λ messages, just like β in the forwards-backwards algorithm, represent conditional likelihoods, not probability distributions π messages, by contrast, represent probability distributions, and can be represented in moment form (using a mean and covariance matrix).

Another issue that arises is that the covariance matrix for a λ message from a perfectly observed node is 0, i.e., the precision is infinite. This cannot be represented in information (canonical) form, because $z = \Sigma^{-1}x$ does not exist. Hence we must represent this special case in moment form. However, it is straightforward to manipulate this "delta function", as we show below.

2 Computing bel

Alag Eqn 2.38, Pearl Eqn 7.22.

$$\text{bel}_X(\Sigma, \overline{x}) \stackrel{\text{def}}{=} \text{compute-bel}(\pi_X(\Sigma_\pi, \overline{x}_\pi), \lambda_X(\Sigma_\lambda^{-1}, \overline{z}_\lambda))$$

where $\overline{z}_\lambda = \Sigma_\lambda^{-1}\overline{x}_\lambda$. If π_X has infinite variance, we set $\text{bel}_X = (\Sigma_\lambda, \overline{x}_\lambda)$. If λ_X has infinite variance, we set $\text{bel}_X = (\Sigma_\pi, \overline{x}_\pi)$. If λ_X is a delta function, we set $\text{bel}_X = (0, x^*)$, where x^* is the observed value of X. Otherwise we proceed as follows.

$$\Sigma = \left(\Sigma_\pi^{-1} + \Sigma_\lambda^{-1}\right)^{-1}$$

$$\overline{x} = \Sigma(\Sigma_\pi^{-1}\overline{x}_\pi + \Sigma_\lambda^{-1}\overline{x}_\lambda)$$

3 Computing π

Alag eqn 2-36, Pearl eqn 7.21.

Let $P(X|U_1, \ldots, U_n) = \mathcal{N}(X; \mu + \sum_{i=1}^n B_i U_i, Q)$. Call all the parameters θ_X. Let the π message sent from U_i to X be $\pi_i(\Sigma_i, \overline{u}_i^+)$.

$$\pi_X(\Sigma_\pi, \overline{x}_\pi) \stackrel{\text{def}}{=} \text{compute-pi}(\{\pi_i\}, \theta_X)$$

$$\Sigma_\pi = \sum_{i=1}^n B_i \Sigma_i B_i^T + Q$$

$$\overline{x}_\pi = \mu + \sum_{i=1}^n B_i \overline{u}_i^+$$

4 Computing λ

We define the following subroutine which multiplies together a set of λ messages, excluding any in the set E (Alag eqn 2-28):

$$\lambda(\Sigma^{-1}, \overline{z}) \stackrel{\text{def}}{=} \text{prod-lamdba-msgs}(\{\lambda_j\}, E)$$

$$\Sigma^{-1} = \sum_{j \notin E} \Sigma_j^{-1}$$

$$\overline{z} = \sum_{j \notin E} z_j$$

If the λ message from self is a delta function, we simply pass it on, representing complete certainty (all the λ messages from the other children are ignored).

Since $\lambda_X(x) = \prod_{j \in \text{ch}(X)} \lambda_j(x)$, we have (Pearl eqn 7.20; different from Alag eqn 2-37):

$$\lambda_X(\Sigma_\lambda^{-1}, \overline{z}_\lambda) = \text{prod-lamdba-msgs}(\{\lambda_j\}, \emptyset)$$

5 Computing the π messages X sends to its children

Since $\pi_j(x) = \text{bel}(X|e_j^- = \emptyset)$, we have

$$\pi_j(\Sigma, \overline{x}^+) = \text{compute-bel}(\pi_X, \text{prod-lamdba-msgs}(\{\lambda_j\}, j))$$

6 Computing the λ messages X sends to its parents

First we present the result in the non-information form (Alag eqn 2-46).

$$\lambda_i(\Sigma_i, \overline{x}_i) \stackrel{\text{def}}{=} \text{compute-lambda-msg}(\lambda_X, \{\pi_k\}, \theta_X)$$

Covariance:

$$\Sigma_i = (B_i^T C B_i)^{-1}$$

where

$$C = \left(\Sigma_\lambda + Q + \sum_{k \neq i} B_k \Sigma_k B_k^T \right)^{-1}$$

Mean:

$$\overline{x}_i = \Sigma_i B_i^T C (\overline{x}_\lambda - u)$$

where

$$u = \mu + \sum_{k \neq i} B_k \overline{u}_k^+$$

In the special case of scalar nodes, we can use a simpler form of 2-24 to rewrite the mean as follows, which corresponds to Pearl Eqn 7.24.

$$\overline{x}_i = B_i^{-1} (\overline{x}_\lambda - u)$$

If λ_X is a delta function, we can use the above form by simply setting $\Sigma_\lambda = 0$ and $\overline{x}_\lambda = x^*$, the observed value.

Unfortunately, the above assumes that Σ_λ and $\overline{x}_\lambda$ can be computed. We can lift this assumption by using the matrix inversion lemma (Alag Eqn 2-2), which, in its simplest form, is

$$(P_1^{-1} + P_2^{-1})^{-1} = P_1 - P_1 (P_1 + P_2)^{-1} P_1$$

Using this, we can rewrite C as follows:

$$C = \Sigma_\lambda^{-1} - \Sigma_\lambda^{-1} A \Sigma_\lambda^{-1}$$

where

$$A = \left(\Sigma_\lambda^{-1} + (Q + \sum_{k \neq i} B_k \Sigma_k B_k^T)^{-1} \right)^{-1}$$

Hence

$$\Sigma_i^{-1} = B_i^T C B_i$$

and

$$\begin{aligned}
\overline{z}_i &= \Sigma_i^{-1}\overline{x}_i \\
&= B_i^T C(\overline{x}_\lambda - u) \\
&= B_i^T \left(\Sigma_\lambda^{-1}\overline{x}_\lambda - \Sigma_\lambda^{-1}A\Sigma_\lambda^{-1}\overline{x}_\lambda - \Sigma_\lambda^{-1}u + \Sigma_\lambda^{-1}A\Sigma_\lambda^{-1}u\right) \\
&= B_i^T \left((I - \Sigma_\lambda^{-1}A)\overline{z}_\lambda - (I - \Sigma_\lambda^{-1}A)\Sigma_\lambda^{-1}u\right)
\end{aligned}$$

References

[Ala96] S. Alag. *A Bayesian Decision-Theoretic Framework for Real-Time Monitoring and Diagnosis of Complex Systems: Theory and Application.* PhD thesis, U.C. Berkeley, Dept. Mech. Eng., 1996.

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