Exact Inference on Conditional Linear Γ-Gaussian Bayesian Networks

Ivar Simonsson & Petter Mostad

September 9, 2016

CHALMERS | UNIVERSITY OF GOTHENBURG

Context:

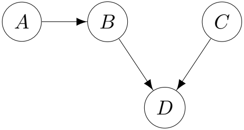
- We are given a BN over a set of random variables $\mathcal{X} = \{X_1, \ldots, X_n\}$, i.e. we know the structure of the BN and we know the CPDs, $\pi(X_i|\text{Pa}_{X_i})$.
- We want to compute $\pi(X_i|X_j = x_j)$ *exactly*.

Context:

- We are given a BN over a set of random variables $\mathcal{X} = \{X_1, \ldots, X_n\}$, i.e. we know the structure of the BN and we know the CPDs, $\pi(X_i|\text{Pa}_{X_i})$.
- We want to compute $\pi(X_i|X_j = x_j)$ *exactly*.
- Assumption: the BN is small enough for this to be feasible.

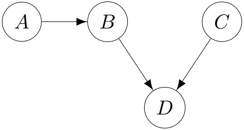
- Unfortunately, the class of networks on which exact inference is possible is quite limited.
- We work on ideas combining conjugacy and numerical methods to extend this class.
- As part of this work, we introduce Γ-Gaussian BNs.

The variable elimination algorithm



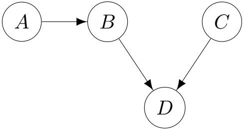
$\pi(D) =$

The variable elimination algorithm



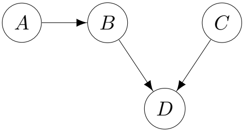
$$\pi(D) = \sum_{A,B,C} \pi(A)\pi(B|A)\pi(C)\pi(D|B,C)$$

The variable elimination algorithm



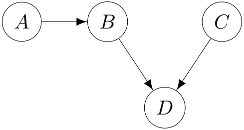
$$\pi(D) = \sum_A \pi(A) \sum_B \pi(B|A) \sum_C \pi(C)\pi(D|B,C)$$

The variable elimination algorithm



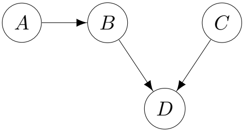
$$\pi(D) = \sum_A \phi_1(A) \sum_B \phi_2(A,B) \sum_C \phi_3(C)\phi_4(B,C,D)$$

The variable elimination algorithm



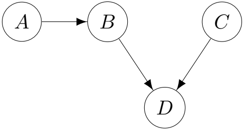
$$\pi(D) = \sum_A \phi_1(A) \sum_B \phi_2(A,B) \sum_C \underbrace{\phi_3(C)\phi_4(B,C,D)}_{\psi(B,C,D)}$$

The variable elimination algorithm



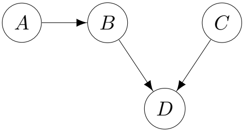
$$\pi(D) = \sum_{A} \phi_1(A) \sum_{B} \phi_2(A, B) \sum_{C} \psi(B, C, D)$$

The variable elimination algorithm



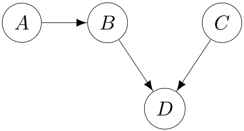
$$\pi(D) = \sum_A \phi_1(A) \sum_B \phi_2(A, B) \underbrace{\sum_C \psi(B, C, D)}_{\tau(B,D)}$$

The variable elimination algorithm



$$\pi(D) = \sum_A \phi_1(A) \sum_B \phi_2(A,B)\color{red}{\tau(A,B)}$$

The variable elimination algorithm



$$\pi(D) = \sum_A \phi_1(A) \sum_B \phi_2(A, B) \tau(A, B)$$

Three operations: reduction, multiplication, marginalization

- In principle, the algorithm works for any Bayesian network.
- We need a good way to represent the factors
  $\Rightarrow$ closed under the local operations.
- If all variables are finite, not hard to implement.
- If some variables are continuous, a bit harder.

- In a *Gaussian* Bayesian network (GBN)

$$X_i|\text{Pa}_{X_i} \sim \mathcal{N}(\mu_i, \sigma_i^2)$$

  where
  - $\mu_i = \alpha_i + \sum_{k=1}^{m_i} \beta_{i,k} Y_{i,k}$
  - $\{Y_{i,1}, \ldots, Y_{i,m_i}\} = \text{Pa}_{X_i}$ is the set of parents of X_i
  - $\sigma_i^2 > 0$ is fixed
- Factor parameterization:

$$\phi(\mathbf{X}) = \exp\left(-\frac{1}{2}\mathbf{X}^T K \mathbf{X} + \mathbf{h}^T \mathbf{X} + g\right)$$

  where $K \in \mathbb{R}^{n \times n}$ is symmetric, $\mathbf{h} \in \mathbb{R}^n$ and $g \in \mathbb{R}$. These are called *canonical forms*.

- Family of canonical forms is closed under the local operations.
- Explanation: conjugacies

$$
\begin{aligned}
&\text{If} & X_1, \ldots, X_n &\sim \mathcal{N}(\mu, \sigma^2) \\
&\text{and} & \mu &\sim \mathcal{N}(\mu_0, \sigma_0^2) \\
&\text{then} & \mu | X_1, \ldots, X_n &\sim \mathcal{N}(\mu', \sigma'^2)
\end{aligned}
$$

- Can we use other conjugacies?

- What about modeling the precision?
- Recall the normal-gamma distribution: $Z \sim \Gamma(\alpha_0, \beta_0)$ and $\theta | Z \sim \mathcal{N}(\theta_0, (\lambda_0 Z)^{-1})$, then $(\theta, Z) \sim \mathcal{NG}(\theta_0, \lambda_0, \alpha_0, \beta_0)$.

- What about modeling the precision?
- Recall the normal-gamma distribution: $Z \sim \Gamma(\alpha_0, \beta_0)$ and $\theta | Z \sim \mathcal{N}(\theta_0, (\lambda_0 Z)^{-1})$, then $(\theta, Z) \sim \mathcal{NG}(\theta_0, \lambda_0, \alpha_0, \beta_0)$.

$$
\begin{aligned}
&\text{If} && X_1, \ldots, X_n \sim \mathcal{N}(\theta, Z^{-1}) \\
&\text{and} && (\theta, Z) \sim \mathcal{NG}(\theta_0, \lambda_0, \alpha_0, \beta_0) \\
&\text{then} && (\theta, Z) | X_1, \ldots, X_n \sim \mathcal{NG}(\theta', \lambda', \alpha', \beta')
\end{aligned}
$$

- Introducing a gamma distributed node in this way leads to the following parameterization

$$\phi(\mathbf{X}, Z) = \exp\left(Z\left(-\frac{1}{2}\mathbf{X}^T K \mathbf{X} + \mathbf{h}^T \mathbf{X} + g\right) + a \log(Z) + b\right)$$

  which we call Γ-canonical forms.
- Unfortunately we run into problems when naively trying to implement this.

- Problem 1: multiplying a Γ-canonical form with a canonical form results in something we don't recognize.
- Current solution: $Z \in \text{Pa}_{X_i}$, for all $X_i \in \mathcal{X}$.

- Problem 1: multiplying a Γ-canonical form with a canonical form results in something we don't recognize.
- Current solution: $Z \in \text{Pa}_{X_i}$, for all $X_i \in \mathcal{X}$.

- Γ-*Gaussian* BN over $\mathcal{X} \cup \{Z\}$
  - $Z \sim \Gamma(\alpha_0, \beta_0)$
  - $Z \in \text{Pa}_{X_i}$ for all $X_i \in \mathcal{X}$
  - $X_i | \text{Pa}_{X_i} \sim \mathcal{N}(\mu_i, \sigma_i^2 / Z)$
  - $\mu_i = \alpha_i + \sum_{k=1}^{m_i} \beta_{i,k} Y_{i,k}$
  - $\{Y_{i,1}, \ldots, Y_{i,m_i}\} = \text{Pa}_{X_i} \setminus \{Z\}$
  - $\sigma_i^2 > 0$ is fixed

- Problem 2: marginalizing a Γ-canonical form over the gamma variable results in a generalized Student-*t* distribution.

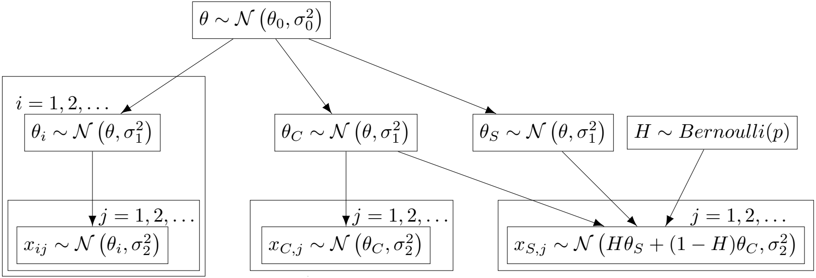
$$\int \phi(\mathbf{X}, Z) dZ \propto \left(1 + \frac{(\mathbf{X} - \delta)^T \Sigma^{-1} (\mathbf{X} - \delta)}{\nu}\right)^{-\frac{\nu+m}{2}}$$

- Bad news: this can not be written in either of the forms we have used so far.
- Good news: we have an analytical expression for it.
- Solution: insist that the gamma variable will be the last variable we eliminate.

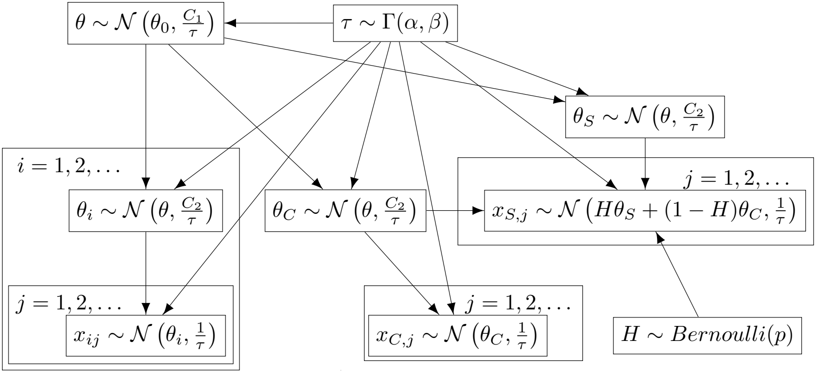
What about networks with a combination of discrete and continuous variables?

- Discrete nodes can be added to Γ-GBNs in the exact same way as for GBNs.
- Restriction: no finite variables with continuous parents.
- Conditional linear GBNs $\Rightarrow$ canonical tables.
- Conditional linear Γ-GBNs $\Rightarrow$ Γ-canonical tables.

Model with a Gaussian BN



Model with a Γ-Gaussian BN



Future work:

- Further extensions in similar ways
- Using more numerical integration
- Take a look at the elimination ordering

Thank you!