Systems Biology:
Inferring gene regulatory network using graphical models

Eric Xing
Lecture 25, April 19, 2006

Bayesian Network – CPDs

Local Probabilities: CPD - conditional probability distribution $P(X_i | \text{Pa}_i)$

- *Discrete variables*: Multinomial Distribution (can represent any kind of statistical dependency)
Bayesian Network – CPDs (cont.)

- **Continuous variables**: e.g. linear Gaussian
  \[
P(X | Y_1, \ldots, Y_k) \sim N(a_0 + \sum_{i=1}^{k} a_i Y_i, \sigma^2)
  \]

- **Learning Bayesian Network**

  - **The goal:**
    - Given set of independent samples (assignments of random variables), find the best (the most likely?) Bayesian Network (both DAG and CPDs)

  - \[(B,E,A,C,R) = (T,F,F,T,F)\]
  - \[(B,E,A,C,R) = (T,F,T,T,F)\]
  - \[
    \begin{array}{c|cc}
    E & B & P(A | E,B) \\
    \hline
    e & b & 0.9 \ 0.1 \\
    e & b & 0.2 \ 0.8 \\
    e & b & 0.9 \ 0.1 \\
    e & b & 0.01 \ 0.99 \\
    \end{array}
  \]
Learning Graphical Models

- Scenarios:
  - completely observed GMs
    - directed
    - undirected
  - partially observed GMs
    - directed
    - undirected (an open research topic)

- Estimation principles:
  - Maximal likelihood estimation (MLE)
  - Bayesian estimation

- We use **learning** as a name for the process of estimating the parameters, and in some cases, the topology of the network, from data.

The basic idea underlying MLE

- Likelihood:
  \[ L(\theta | X) = p(X | \theta) = p(X_1 | \theta_1)p(X_2 | \theta_2)p(X_3 | X_2, X_3, \theta_3) \]

- Log-Likelihood:
  \[ l(\theta | X) = \log p(X | \theta) = \log p(X_1 | \theta_1) + \log p(X_2 | \theta_2) + \log p(X_3 | X_2, X_3, \theta_3) \]

- Data log-likelihood
  \[ l(\theta | DATA) = \log \prod_{n} p(X_n | \theta) \]
  \[ = \sum_{n} \log p(X_1(n) | \theta_1) + \sum_{n} \log p(X_2(n) | \theta_2) + \sum_{n} \log p(X_3(n) | X_1(n), X_2(n), \theta_3) \]

- MLE
  \[ \{\theta_1, \theta_2, \theta_3\}_{MLE} = \arg \max \{l(\theta | DATA)\} \]
  \[ \theta_1^{*} = \arg \max_{\theta_1} \sum_{n} \log p(X_1(n) | \theta_1), \quad \theta_2^{*} = \arg \max_{\theta_2} \sum_{n} \log p(X_2(n) | \theta_2), \quad \theta_3^{*} = \arg \max_{\theta_3} \sum_{n} \log p(X_3(n) | X_1(n), X_2(n), \theta_3) \]
Learning Bayesian Network

- **Learning of best CPDs given DAG is easy**
  - collect statistics of values of each node given specific assignment to its parents

- **Learning of the graph topology (structure) is NP-hard**
  - heuristic search must be applied, generally leads to a **locally** optimal network

- **Overfitting**
  - It turns out, that richer structures give higher likelihood $P(D|G)$ to the data
    - (adding an edge is always preferable)
    - more parameters to fit => more freedom => always exist more "optimal" CPD(C)
  - **We prefer simpler** (more explanatory) networks
    - Practical scores **regularize** the likelihood improvement complex networks.

BN Learning Algorithms

- **Structural EM (Friedman 1998)**
  - The original algorithm

- **Sparse Candidate Algorithm (Friedman et al.)**
  - Discretizing array signals
  - Hill-climbing search using local operators: add/delete/swap of a single edge
  - Feature extraction: Markov relations, order relations
  - Re-assemble high-confidence sub-networks from features

- **Module network learning (Segal et al.)**
  - Heuristic search of structure in a "module graph"
  - Module assignment
  - Parameter sharing
  - Prior knowledge: possible regulators (TF genes)
Confidence Estimates

Bootstrap approach:

Estimate "Confidence level":

\[ C(f) = \frac{1}{m} \sum_{i=1}^{m} 1 \{ f \in G_i \} \]

Results from SCA + feature extraction (Friedman et al.)

The initially learned network of ~800 genes

The “mating response” substructure
A Module Network

Nature Genetics 34, 166 - 176 (2003)

Gaussian Graphical Models

- Why?

Sometimes an UNDIRECTED association graph makes more sense and/or is more informative

- gene expressions may be influenced by unobserved factor that are post-transcriptionally regulated

- The unavailability of the state of B results in a constrain over A and C
Recap of Basic Prob. Concepts

- Joint probability dist. on multiple variables:

\[ P(X_1, X_2, X_3, X_4, X_5, X_6) = P(X_1)P(X_2 | X_1)P(X_3 | X_2)P(X_4 | X_1)P(X_5 | X_4)P(X_6 | X_2, X_5) \]

- If \(X_i\)'s are independent: \(P(X_i) = P(X_i)\)

\[ P(X_1, X_2, X_3, X_4, X_5, X_6) = P(X_1)P(X_2 | X_1)P(X_3 | X_2)P(X_4 | X_1)P(X_5 | X_4)P(X_6 | X_2, X_5) = \prod_i P(X_i) \]

- If \(X_i\)'s are conditionally independent (as described by a GM), the joint can be factored to simpler products, e.g.,
Probabilistic Inference

- We now have compact representations of probability distributions: **Graphical Models**

- A GM $\mathcal{M}$ describes a unique probability distribution $P$

- How do we answer queries about $P$?

- We use **inference** as a name for the process of computing answers to such queries

---

Query 1: Likelihood

- Most of the queries one may ask involve evidence
  - Evidence $e$ is an assignment of values to a set $E$ variables in the domain
  - Without loss of generality $E = \{X_{k+1}, \ldots, X_n\}$

- Simplest query: compute probability of evidence

  $$P(e) = \sum_{x_k} \cdots \sum_{x_1} P(x_1, \ldots, x_k, e)$$

  - this is often referred to as computing the **likelihood** of $e$
Often we are interested in the conditional probability distribution of a variable given the evidence. The conditional probability of a variable $X$ given the evidence $e$ is defined as:

$$P(X | e) = \frac{P(X,e)}{P(e)} = \frac{\sum_X P(X = x, e)}{P(e)}$$

- This is the a posteriori belief in $X$, given evidence $e$.

- We usually query a subset $Y$ of all domain variables $X = \{Y, Z\}$ and "don't care" about the remaining, $Z$:

$$P(Y | e) = \sum_z P(Y, Z = z | e)$$

- The process of summing out the "don't care" variables $Z$ is called marginalization, and the resulting $P(Y|e)$ is called a marginal probability.

### Applications of a posteriori Belief

- **Prediction**: what is the probability of an outcome given the starting condition?

- **Diagnosis**: what is the probability of disease/fault given symptoms?

- **Learning** under partial observation:
  - Fill in the unobserved values under an "EM" setting (more later).

- The directionality of information flow between variables is not restricted by the directionality of the edges in a GM:
  - Probabilistic inference can combine evidence from all parts of the network.
Query 3: Most Probable Assignment

- In this query we want to find the most probable joint assignment (MPA) for some variables of interest.

- Such reasoning is usually performed under some given evidence $e$, and ignoring (the values of) other variables $z$:

$$\text{MPA}(Y \mid e) = \arg \max_y P(y \mid e) = \arg \max_y \sum_z P(y \mid z, e)$$

- this is the maximum a posteriori configuration of $y$.

Applications of MPA

- Classification
  - find most likely label, given the evidence

- Explanation
  - what is the most likely scenario, given the evidence

Cautionary note:

- The MPA of a variable depends on its "context"---the set of variables been jointly queried.

- Example:
  - MPA of $X$?
  - MPA of $(X, Y)$?

<table>
<thead>
<tr>
<th>$x$</th>
<th>$y$</th>
<th>$P(x,y)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0.35</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0.05</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0.3</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0.3</td>
</tr>
</tbody>
</table>
Complexity of Inference

Thm:
Computing $P(X = x | e)$ in a GM is NP-hard

- Hardness does not mean we cannot solve inference
  - It implies that we cannot find a general procedure that works efficiently for arbitrary GMs
  - For particular families of GMs, we can have provably efficient procedures

Approaches to inference

- Exact inference algorithms
  - The elimination algorithm ✓
  - The junction tree algorithms ✓ (but will not cover in detail here)

- Approximate inference techniques
  - Stochastic simulation / sampling methods ✓
  - Markov chain Monte Carlo methods ✓
  - Variational algorithms (later lectures)
Marginalization and Elimination

A signal transduction pathway:

What is the likelihood that protein E is active?

- Query: \( P(e) \)

\[ P(e) = \sum \sum \sum \sum P(a, b, c, d, e) \]

- By chain decomposition, we get

\[ = \sum \sum \sum \sum P(a)P(b \mid a)P(c \mid b)P(d \mid c)P(e \mid d) \]

Elimination on Chains

- Rearranging terms ...

\[ P(e) = \sum \sum \sum \sum P(a)P(b \mid a)P(c \mid b)P(d \mid c)P(e \mid d) \]

\[ = \sum \sum \sum P(c \mid b)P(d \mid c)P(e \mid d) \sum P(a)P(b \mid a) \]
Elimination on Chains

Now we can perform innermost summation

\[ P(e) = \sum_{d} \sum_{c} \sum_{b} \sum_{a} P(c|b) P(d|c) P(e|d) \sum_{a} P(a) P(b|a) \]

\[ = \sum_{d} \sum_{c} \sum_{b} P(c|b) P(d|c) P(e|d) p(b) \]

This summation "eliminates" one variable from our summation argument at a "local cost".

Elimination in Chains

Rearranging and then summing again, we get

\[ P(e) = \sum_{d} \sum_{c} \sum_{b} P(c|b) P(d|c) P(e|d) p(b) \]

\[ = \sum_{d} \sum_{c} P(d|c) P(e|d) \sum_{b} P(c|b) p(b) \]

\[ = \sum_{d} \sum_{c} P(d|c) P(e|d) p(c) \]
Elimination in Chains

- Eliminate nodes one by one all the way to the end, we get

\[ P(e) = \sum_d P(e \mid d) p(d) \]

Complexity:
- Each step costs \( O(|\text{Val}(X_1)| \cdot |\text{Val}(X_{n-1})|) \) operations: \( O(kn^2) \)
- Compare to naïve evaluation that sums over joint values of \( n-1 \) variables \( O(n^k) \)

Inference on General GM via Variable Elimination

General idea:
- Write query in the form

\[ P(X_1, e) = \sum_{x_n} \sum_{x_3} \sum_{x_2} \prod_{i} P(x_i \mid pa_i) \]

- this suggests an "elimination order" of latent variables to be marginalized

- Iteratively
  - Move all irrelevant terms outside of innermost sum
  - Perform innermost sum, getting a new term
  - Insert the new term into the product

- wrap-up

\[ P(X_1 \mid e) = \frac{P(X_1, e)}{P(e)} \]
A more complex network

A food web

What is the probability that hawks are leaving given that the grass condition is poor?

Example: Variable Elimination

- Query: $P(A | h)$
  - Need to eliminate: $B, C, D, E, F, G, H$
- Initial factors:
  $$P(a)P(b)P(c | b)P(d | a)P(e | c, d)P(f | a)P(g | e)P(h | e, f)$$
- Choose an elimination order: $H, G, F, E, D, C, B$
- Step 1:
  - Conditioning (fix the evidence node (i.e., $h$) to its observed value (i.e., $\tilde{h}$)):
    $$m_h(e, f) = p(h = \tilde{h} | e, f)$$
  - This step is isomorphic to a marginalization step:
    $$m_h(e, f) = \sum_h p(h | e, f) \delta(h = \tilde{h})$$
Example: Variable Elimination

- Query: \(P(B \mid h)\)
  - Need to eliminate: \(B,C,D,E,F,G\)

- Initial factors:
  \[
P(a)P(b)P(c \mid b)P(d \mid a)P(e \mid c,d)P(f \mid a)P(g \mid e)P(h \mid e,f)
  \]
  \[
  \Rightarrow P(a)P(b)P(c \mid b)P(d \mid a)P(e \mid c,d)P(f \mid a)P(g \mid e)m_h(e,f)
  \]

- Step 2: Eliminate \(G\)
  - compute \(m_y(e) = \sum p(g \mid e) = 1\)
  \[
  \Rightarrow P(a)P(b)P(c \mid b)P(d \mid a)P(e \mid c,d)P(f \mid a)m_y(e)\]
  \[
  = P(a)P(b)P(c \mid b)P(d \mid a)P(e \mid c,d)P(f \mid a)m_y(e,f)
  \]

Example: Variable Elimination

- Query: \(P(B \mid h)\)
  - Need to eliminate: \(B,C,D,E,F\)

- Initial factors:
  \[
P(a)P(b)P(c \mid b)P(d \mid a)P(e \mid c,d)P(f \mid a)P(g \mid e)P(h \mid e,f)
  \]
  \[
  \Rightarrow P(a)P(b)P(c \mid b)P(d \mid a)P(e \mid c,d)P(f \mid a)P(g \mid e)m_h(e,f)
  \]

- Step 3: Eliminate \(F\)
  - compute \(m_f(e,a) = \sum p(f \mid a)m_y(e,f)\)
  \[
  \Rightarrow P(a)P(b)P(c \mid b)P(d \mid a)P(e \mid c,d)m_f(a,e)
  \]
Example: Variable Elimination

- **Query:** $P(B \mid h)$
  - Need to eliminate: $B, C, D, E$

- **Initial factors:**
  $$
P(a)p(b)p(c \mid b)p(d \mid a)p(e \mid c, d)p(f \mid a)p(g \mid e)p(h \mid e, f)$$
  $$\Rightarrow P(a)p(b)p(c \mid b)p(d \mid a)p(e \mid c, d)p(f \mid a)p(g \mid e)m_b(e, f)$$
  $$\Rightarrow P(a)p(b)p(c \mid b)p(d \mid a)p(e \mid c, d)p(f \mid a)m_e(e, f)$$
  $$\Rightarrow P(a)p(b)p(c \mid b)p(d \mid a)p(e \mid c, d)m_{_b}(a, e)$$

- **Step 4: Eliminate $E$**
  - Compute
  $$m_e(a, c, d) = \sum_ep(e \mid c, d)m_e(a, e)$$
  $$\Rightarrow P(a)p(b)p(c \mid b)p(d \mid a)m_{_b}(a, c, d)$$

Example: Variable Elimination

- **Query:** $P(B \mid h)$
  - Need to eliminate: $B, C, D$

- **Initial factors:**
  $$
P(a)p(b)p(c \mid b)p(d \mid a)p(e \mid c, d)p(f \mid a)p(g \mid e)p(h \mid e, f)$$
  $$\Rightarrow P(a)p(b)p(c \mid b)p(d \mid a)p(e \mid c, d)p(f \mid a)p(g \mid e)m_b(e, f)$$
  $$\Rightarrow P(a)p(b)p(c \mid b)p(d \mid a)p(e \mid c, d)p(f \mid a)m_e(e, f)$$
  $$\Rightarrow P(a)p(b)p(c \mid b)p(d \mid a)p(e \mid c, d)m_{_b}(a, e)$$
  $$\Rightarrow P(a)p(b)p(c \mid b)p(d \mid a)m_{_b}(a, c, d)$$

- **Step 5: Eliminate $D$**
  - Compute
  $$m_d(a, c) = \sum_dm_d(d \mid a)m_{_d}(a, c, d)$$
  $$\Rightarrow P(a)p(b)p(c \mid d)m_{_d}(a, c)$$
Example: Variable Elimination

- Query: \( P(B \mid h) \)
  - Need to eliminate: \( B, C \)

- Initial factors:
  \[
  P(a)P(b)P(c \mid d)P(d \mid a)P(e \mid c, d)P(f \mid a)P(g \mid e)P(h \mid e, f)
  \]
  \[
  \Rightarrow P(a)P(b)P(c \mid d)P(d \mid a)P(e \mid c, d)P(f \mid a)P(g \mid e)m_h(a, e, f)
  \]
  \[
  \Rightarrow P(a)P(b)P(c \mid d)P(d \mid a)P(e \mid c, d)m_h(a, e)
  \]
  \[
  \Rightarrow P(a)P(b)P(c \mid d)P(d \mid a)m_h(a, c, d)
  \]
  \[
  \Rightarrow P(a)P(b)P(c \mid d)m_h(a, c)
  \]

- Step 6: Eliminate \( C \)
  - Compute \( m_C(a, b) = \sum_c P(c \mid b)m_f(a, c) \)
    \[
    \Rightarrow P(a)P(b)m_C(a, b)
    \]

Example: Variable Elimination

- Query: \( P(B \mid h) \)
  - Need to eliminate: \( B \)

- Initial factors:
  \[
  P(a)P(b)P(c \mid d)P(d \mid a)P(e \mid c, d)P(f \mid a)P(g \mid e)P(h \mid e, f)
  \]
  \[
  \Rightarrow P(a)P(b)P(c \mid d)P(d \mid a)P(e \mid c, d)P(f \mid a)P(g \mid e)m_h(a, e, f)
  \]
  \[
  \Rightarrow P(a)P(b)P(c \mid d)P(d \mid a)P(e \mid c, d)m_h(a, e)
  \]
  \[
  \Rightarrow P(a)P(b)P(c \mid d)m_h(a, c, d)
  \]
  \[
  \Rightarrow P(a)P(b)P(c \mid d)m_h(a, c)
  \]

- Step 7: Eliminate \( B \)
  - Compute \( m_b(a) = \sum_b P(b)m_C(a, b) \)
    \[
    \Rightarrow P(a)m_b(a)
    \]
Example: Variable Elimination

- Query: \( P(B \mid h) \)
  - Need to eliminate: \{ \}

- Initial factors:
  \[
P(a)p(b)p(c \mid d)p(d \mid a)p(e \mid c, d)p(f \mid a)p(g \mid e)p(h \mid e, f)
  \[
  \Rightarrow p(a)p(b)p(c \mid d)p(d \mid a)p(e \mid c, d)p(f \mid a)p(g \mid e)m_b(e, f)
  \[
  \Rightarrow p(a)p(b)p(c \mid d)p(d \mid a)p(e \mid c, d)p(f \mid a)m_e(a, e)
  \[
  \Rightarrow p(a)p(b)p(c \mid d)m_y(a, c)
  \[
  \Rightarrow p(a)p(b)m_y(a, b)
  \[
  \Rightarrow p(a)m_b(a)
  \]

- Step 8: Wrap-up
  \[
p(a, \tilde{h}) = p(a)m_b(a), \quad p(\tilde{h}) = \sum_a p(a)m_b(a)
  \]
  \[
  \Rightarrow P(a \mid \tilde{h}) = \frac{p(a)m_b(a)}{\sum_a p(a)m_b(a)}
  \]

Complexity of variable elimination

- Suppose in one elimination step we compute
  \[
m_x(y_1, \ldots, y_k) = \sum_x m'_x(x, y_1, \ldots, y_k)
  \]
  \[
m'_x(x, y_1, \ldots, y_k) = \prod_{i=1}^k m_i(x, y_i)
  \]
  This requires
  \[
  k \cdot |\text{Val}(X)| \cdot \prod_i |\text{Val}(Y_i)| \quad \text{multiplications}
  \]
  - For each value for \(x, y_1, \ldots, y_k\) we do \(k\) multiplications

  \[
  |\text{Val}(X)| \cdot \prod_i |\text{Val}(Y_i)| \quad \text{additions}
  \]
  - For each value of \(y_1, \ldots, y_k\), we do \(|\text{Val}(X)|\) additions

  Complexity is exponential in number of variables in the intermediate factor
### Understanding Variable Elimination

- A graph elimination algorithm

- Intermediate terms correspond to the *cliques* resulted from elimination
  - "good" elimination orderings lead to *small cliques* and hence reduce complexity (what will happen if we eliminate "e" first in the above graph?)
  - finding the optimum ordering is NP-hard, but for many graph optimum or near-optimum can often be heuristically found

- Applies to undirected GMs

### From Elimination to Message Passing

- Our algorithm so far answers only one query (e.g., on one node), do we need to do a complete elimination for every such query?

- Elimination = message passing on a **clique tree**

\[
m_{y}(a,c,d') = \sum_{e} p(e | c,d') m_y(e)m_y(a,e)
\]

- Messages can be reused
From Elimination to Message Passing

- Our algorithm so far answers only one query (e.g., on one node), do we need to do a complete elimination for every such query?
- Elimination $\equiv$ message passing on a clique tree
  - Another query ...
- Messages $m_f$ and $m_h$ are reused, others need to be recomputed

A Sketch of the Junction Tree Algorithm

- The algorithm
  - Construction of junction trees --- a special clique tree
  - Propagation of probabilities --- a message-passing protocol
- Results in marginal probabilities of all cliques --- solves all queries in a single run
- A generic exact inference algorithm for any GM
- Complexity: exponential in the size of the maximal clique --- a good elimination order often leads to small maximal clique, and hence a good (i.e., thin) JT
- Many well-known algorithms are special cases of JT
  - Forward-backward, Kalman filter, Peeling, Sum-Product ...
**Approaches to inference**

- **Exact inference algorithms**
  - The elimination algorithm
  - The junction tree algorithms (but will not cover in detail here)

- **Approximate inference techniques**
  - Stochastic simulation / sampling methods
  - Markov chain Monte Carlo methods
  - Variational algorithms (later lectures)

**Monte Carlo methods**

- Draw random samples from the desired distribution
- Yield a stochastic representation of a complex distribution
  - marginals and other expectations can be approximated using sample-based averages
    \[ E[f(x)] = \frac{1}{N} \sum_{i=1}^{N} f(x^{(i)}) \]
- Asymptotically exact and easy to apply to arbitrary models

**Challenges:**
- how to draw samples from a given dist. (not all distributions can be trivially sampled)?
- how to make better use of the samples (not all sample are useful, or equally useful, see an example later)?
- how to know we've sampled enough?
Example: naive sampling

- **Sampling:** Construct samples according to probabilities given in a BN.

  ![BN Diagram](image)

Alarm example: (Choose the right sampling sequence)

1) Sampling: $P(B) = <0.001, 0.999>$ suppose it is false, $B_0$. Same for $E_0$. $P(A|B_0, E_0) = <0.001, 0.999>$ suppose it is false...

2) Frequency counting: In the samples right, $P(J|A_0) = P(J, A_0)/P(A_0) = <1/9, 8/9>$.

Example: naive sampling

- **Sampling:** Construct samples according to probabilities given in a BN.

Alarm example: (Choose the right sampling sequence)

3) what if we want to compute $P(J|A_1)$? we have only one sample ...

   $P(J|A_1) = P(J, A_1)/P(A_1) = <0, 1>$.

4) what if we want to compute $P(J|B_1)$? No such sample available!

   $P(J|A_1) = P(J, B_1)/P(B_1)$ can not be defined.

For a model with hundreds or more variables, rare events will be very hard to garner enough samples even after a long time or sampling ...
Monte Carlo methods (cond.)

- **Direct Sampling**
  - We have seen it.
  - Very difficult to populate a high-dimensional state space

- **Rejection Sampling**
  - Create samples like direct sampling, only count samples which is consistent with given evidences.

- ....

- **Markov chain Monte Carlo (MCMC)**

Markov chain Monte Carlo

- Samples are obtained from a Markov chain (of sequentially evolving distributions) whose stationary distribution is the desired $p(x)$

- Gibbs sampling
  - we have variable set to $X = \{x_1, x_2, x_3, \ldots, x_N\}$
  - at each step one of the variables $X_i$ is selected (at random or according to some fixed sequences)
  - the conditional distribution $p(X_i | X_{-i})$ is computed
  - a value $x_i$ is sampled from this distribution
  - the sample $x_i$ replaces the previous of $X_i$ in $X$. 
MCMC

- Markov-Blanket
  - A variable is independent from others, given its parents, children and children’s parents. d-separation.
  
  \[ p(X|X_{-i}) = p(X|\text{MB}(X)) \]

- Gibbs sampling
  - Create a random sample. Every step, choose one variable and sample it by \( P(X|\text{MB}(X)) \) based on previous sample.

\[ \text{MB(A)} = \{B, E, J, M\} \]
\[ \text{MB(E)} = \{A, B\} \]

MCMC

- To calculate \( P(J|B1,M1) \)
  - Choose \((B1,E0,A1,M1,J1)\) as a start
  - Evidences are \(B1, M1\), variables are \(A, E, J\).
  - Choose next variable as \(A\)
  - Sample \(A\) by \( P(A|\text{MB}(A)) = P(A|B1, E0, M1, J1) \) suppose to be false.
  - \((B1, E0, A0, M1, J1)\)
  - Choose next random variable as \(E\), sample \(E \sim P(E|B1,A0)\)
  - ...
Complexity for Approximate Inference

- Inference problem is NP-hard.
- Approximate Inference will not reach the exact probability distribution in finite time, but only close to the value.
- Often much faster than exact inference when BN is big and complex enough. In MCMC, only consider \( P(X|MB(X)) \) but not the whole network.

Covariance Selection

- Multivariate Gaussian over all continuous expressions
  \[
  p([x_1, \ldots, x_n]) = \frac{1}{(2\pi)^{\frac{n}{2}} |\Sigma|^\frac{1}{2}} \exp\left\{ -\frac{1}{2} (\bar{x} - \mu)^T \Sigma^{-1} (\bar{x} - \mu) \right\}
  \]

- The precision matrix \( K=\Sigma^{-1} \) reveals the topology of the (undirected) network
  \[
  E(x_i | x_j) = \sum_j (K_{ij} / K_{ii}) x_j
  \]
  - Edge \( \sim |K_{ij}| > 0 \)

- Learning Algorithm: Covariance selection
  - Want a sparse matrix
    - Regression for each node with degree constraint (Dobra et al.)
    - Regression for each node with hierarchical Bayesian prior (Li, et al)
A comparison of BN and GGM:

Table 1: modules with multiple regulators and more than 5 regulated genes.

<table>
<thead>
<tr>
<th>Module</th>
<th>Protein-DNA interaction</th>
<th>Article reference</th>
<th>Regulation pattern</th>
<th>Expression pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARRB1</td>
<td>Protein-DNA interaction</td>
<td>[12]</td>
<td>Enhancer and Enhancer</td>
<td>Enhancer and Enhancer</td>
</tr>
<tr>
<td>DUX5L1</td>
<td>Protein-DNA interaction</td>
<td>[14]</td>
<td>Enhancer and Enhancer</td>
<td>Enhancer and Enhancer</td>
</tr>
<tr>
<td>PRKABC1</td>
<td>Protein-DNA interaction</td>
<td>[15]</td>
<td>Enhancer and Enhancer</td>
<td>Enhancer and Enhancer</td>
</tr>
<tr>
<td>GXRAB</td>
<td>Protein-DNA interaction</td>
<td>[16]</td>
<td>Enhancer and Enhancer</td>
<td>Enhancer and Enhancer</td>
</tr>
<tr>
<td>GADD45A</td>
<td>Protein-DNA interaction</td>
<td>[17]</td>
<td>Enhancer and Enhancer</td>
<td>Enhancer and Enhancer</td>
</tr>
<tr>
<td>C6orf20</td>
<td>Protein-DNA interaction</td>
<td>[18]</td>
<td>Enhancer and Enhancer</td>
<td>Enhancer and Enhancer</td>
</tr>
<tr>
<td>ING1</td>
<td>Protein-DNA interaction</td>
<td>[19]</td>
<td>Enhancer and Enhancer</td>
<td>Enhancer and Enhancer</td>
</tr>
<tr>
<td>KLF5</td>
<td>Protein-DNA interaction</td>
<td>[20]</td>
<td>Enhancer and Enhancer</td>
<td>Enhancer and Enhancer</td>
</tr>
</tbody>
</table>

2: Protein-DNA Interaction Network

- Expression networks are not necessarily causal
- BNs are indefinable only up to Markov equivalence:
  - can give the same optimal score, but not further distinguishable under a likelihood score unless further experiment from perturbation is performed
- GGM have yields functional modules, but no causal semantics
- TF-motif interactions provide direct evidence of casual, regulatory dependencies among genes
  - stronger evidence than expression correlations
  - indicating presence of binding sites on target gene -- more easily verifiable
  - disadvantage: often very noisy, only applies to cell-cultures, restricted to known TFs...
ChIP-chip analysis

Advantages:
- Identifies “all” the sites where a TF binds “in vivo” under the experimental condition.

Limitations:
- Expense: Only 1 TF per experiment
- Feasibility: need an antibody for the TF
- Prior knowledge: need to know what TF to test.