1 Basic Notations

We will look at a graph $G = (V, E)$ as a basis for a probabilistic model. We define $n$ random variables: $x_1, ..., x_n$. Each random variable $x_i$ is represented as a node in $V$:

$$V = \{x_1, ..., x_n\}$$
$$E = \{(x_i, x_j) | x_i, x_j \in V\}$$

The parents of a node $x_i$ in the graph are the nodes that point into $x_i$, labeled $Par_{x_i}$:

$$Par_{x_i} = \{x_j | (x_j \rightarrow x_i) \in E\}$$

2 Graph Factorization

We’ll define: $p(x_1|Par_{x_1})$ as the conditional probability of $x_i$ given his parents. The edges of the graph $G$ define the dependencies. A probability model $P$ is consistent with a graph $G$ if:

$$p(x_1, ..., x_n) = \prod_i p(x_i|Par_{x_i})$$

Given a definition of a certain graph $G$ and a probability model, we would like to ask the following question: is the probability model consistent with this graph? A full graph is called a clique. In a directed graph, a clique is actually double-sided. Figure 1-(1) contains a full graph in the case of two nodes. In this graph:

$$P(x_1, x_2) = P(x_1|x_2)P(x_2|x_1)$$

This is a problematic example, because it usually does not take place in the real world. This graph does not really give us any information because it means that $x_1$ determines the value of $x_2$ and $x_2$ determines the value of $x_1$. However, what always takes place is the chain rule. For any probability model, we can always apply the chain rule in the following way:

$$p(x_1, ..., x_n) = p(x_1) \cdot p(x_2|x_1) \cdot \cdots p(x_n|x_1, ..., x_{n-1})$$

The graph that is consistent with the chain rule is a graph in which the parents of a node $x_i$ are all the nodes that precede it in the graph (meaning: nodes with lower indexes). This type of graph is called DAG: directed a-cyclic graph. Figure 1-(2) contains an example of such a graph with 4 nodes. Note that every edge we will add to this graph will create a cycle. Although this graph is consistent with every probability model - it does not contain “interesting” information about the model, since as we mentioned, the chain rule always applies. The interesting graphs are therefore more sparse. The more edges we will remove from the graph - the more clear the structure of the probability model is. In a graph that lacks any edge, all variables (nodes) are actually independent of each other.
Figure 1: (1) A clique in a directed graph with two nodes. (2) A DAG with 4 nodes which fulfills the chain rule. (3) A graph in which $x_1$ is the last node in the DAG and $z, u_1$ are as defined in the notes. (4) The case in which $x_1$ is not the last node in the DAG, $T$ is the group of nodes that come after $x_1$ in the tree.

**Theorem 2.1** If $G$ is an acyclic graph then for any choice of

\[ p(x_i | \text{Par}_{x_i}) \]

the following distribution function:

\[ p(x_1, ..., x_n) = \prod_i p(x_i | \text{Par}_{x_i}) \]

is a legal distribution.

In order to prove this theorem, we need to show that the sum of probabilities over all possible values of $x_i$ equals 1. In order to get some intuition, we will think of an example of a graph that contains two nodes:

\[ E = \{x_1 \rightarrow x_2\} \]

then:

\[ \sum_{x_1} \sum_{x_2} p(x_1)p(x_2|x_1) = \sum_{x_1} p(x_1) \cdot \sum_{x_2} p(x_2|x_1) = \sum_{x_1} p(x_1) \cdot 1 = 1 \]

The general case is proven in a similar way.

Given a certain distribution $q$: $q(x_i | \text{Par}_{x_i})$, define:

\[ p(x_1, ..., x_n) = \prod_i q(x_i | \text{Par}_{x_i}) \]

Our question is whether the distribution that was created really equals what we defined on the graph, meaning, does the following actually take place:

\[ p(x_i | \text{Par}_{x_i}) = \frac{p(x_i, \text{Par}_{x_i})}{p(\text{Par}_{x_i})} = q(x_i | \text{Par}_{x_i}) \]
We will use the following notation: \( u_i = Par_{x_i} \).

We will look at a case in which \( x_1 \) is the last one in the DAG and define the group \( z \) as follows:

\[
z = \{ x_j | j \neq i, x_j \notin u_i \}
\]

Figure 1-(3) schematically describes the structure of \( z, u_1 \) and \( x_1 \).

\[
p(x_1, u_1) = \sum_z p(z, x_1, u_1) = (\ast) \sum_z [...] \cdot q(x_1|u_1) = q(x_1|u_1) \cdot \sum_z [...] \tag{1}
\]

(* Note that [...] is the product of factors that do not depend on \( x \)).

\[
p(u_1) = \sum_{x_1} \sum_z p(x_1, u_1, z) = \sum_z \sum_{x_1} [...] \cdot q(x_1|u_1) = \sum_z [...] \sum_{x_1} q(x_1|u_1) = \sum_z [...] \cdot 1 = \sum_z [...] \tag{2}
\]

Using Eq. 1 and Eq. 2 we can now write the following:

\[
p(x_1|u_1) = \frac{p(x_1, u_1)}{p(u_1)} = \frac{q(x_1|u_1) \cdot \sum_z [...]}{\sum_z [...]} = q(x_1|u_1) \tag{3}
\]

And what about the case in which \( x_1 \) is not the last node in the DAG? Denote \( T \) as the group of nodes that come after \( x_1 \) in the tree. Denote \( t_1 \) as the last leaf in the tree, as shown in figure 1-(4), then:

\[
p(x_1|u_1) = \sum_z \sum_T p(x_1, u_1, z, T) = \sum_z \sum_T \prod_i q(x_i|u_i) = \tag{4}
\]

\[
\sum_z \sum_T \prod_{i \neq t_1} [q(x_i|u_i)] \cdot q(t_1|u_{t_1}) = \sum_z \sum_{t_2, \ldots, t_k \neq t_1} \prod_{t_i} q(x_i|u_i) \cdot \sum_{t_1} q(t_1|u_{t_1}) \tag{5}
\]

\[
= \sum_z \sum_{t_2, \ldots, t_k \neq t_1} \prod_{t_i} q(x_i|u_i) \cdot \sum_{t_1} 1 = \sum_z \sum_{t_2, \ldots, t_k \neq t_1} \prod_{t_i} q(x_i|u_i) \tag{6}
\]

In a similar way, we can get rid of \( t_2, \ldots, t_k \) and then get a proof from part A Eq. 3.

3 Properties of independence according to the graph

\( x \) is independent of \( y \):

\[
x \perp y \iff p(x|y) = p(x) \iff p(x, y) = p(x) \cdot (y)
\]

\( x \) is independent of \( y \) given \( z \):

\[
(x \perp y) | z \iff p(x|y, z) = p(x|z) \iff p(x, y|z) = p(x|z) \cdot (y|z)
\]

Here are some examples of directed graphs and the independencies that can be derived from them:

\[
\begin{align*}
A & \quad B & \Rightarrow P(A, B) = P(A) \cdot P(B) \iff A \perp B \\
A & \rightarrow B & \Rightarrow P(A, B) = P(A) \cdot P(B|A) \\
A & \quad B & \quad C & \Rightarrow P(A, B, C) = P(A) \cdot P(B) \cdot P(C) \iff A \perp B, C \quad B \perp A, C \quad C \perp A, B
\end{align*}
\]
A → B
\[ \Rightarrow P(A, B, C) = P(A) \cdot P(B|A) \cdot P(C|A, B) \] (this is the chain rule)

A → B ⇒ P(A, B, C) = P(A) \cdot P(B|A) \cdot P(C|B) ⇔ (A ⊥ C)|B

because in this case:
\[ P(C|A, B) = \frac{P(A, B, C)}{P(A, B)} = \frac{P(A) \cdot P(B|A) \cdot P(C|B)}{P(A) \cdot P(B|A)} = P(C|B) \]

A ← B → C ⇒ P(A, B, C) = P(A) \cdot P(A|B) \cdot P(C|B) ⇔ (A ⊥ C)|B

A → B ← C ⇒ P(A, B, C) = P(A) \cdot P(C) \cdot P(B|A, C) ⇔ (A ⊥ C) but not (A ⊥ C)|B

The reason for (A ⊥ C) when B is not observed: If B is unknown, then we can sum over all of B’s possible values:
\[ P(A, C) = \sum_B P(A, B, C) = \sum_B P(A) \cdot P(C) \cdot P(B|A, C) = P(A) \cdot P(C) \cdot \sum_B P(B|A, C) = P(A) \cdot P(C) \]

This example is not intuitive: How is it possible for two variables to be independent, but when we are given more information (observing B) they become dependent? An example for this, is a child B that carries a gene for a certain disease. If we learn that his mother A doesn’t have that gene, then we conclude that his father C must be carrying the defective gene.

### 3.1 Markov Blanket

Let us look at a vertex \( x_i \) in a DAG. Every other vertex in the graph belongs to one of these groups:

- \( \text{Par}_i \) - The parents of \( x_i \). That is, there is an edge from them into \( x_i \).
- \( \text{Descendants}_i \) - There is an edge from \( x_i \) into these vertexes or into their ancestors.
- \( \text{ND}_i \) - Non descendants of \( x_i \), all other vertexes in the graph: ancestors (but not parents), siblings, cousins etc.

In the Graph, there is a set of conditional independencies of the form: \( x_i \perp \text{ND}_i | \text{Par}_i \) for every \( i \).

![Graph division into categories](image)

Figure 2: division of a graph into categories: \( \text{Par}_i \) (parents of \( x_i \)), \( \text{ND}_i \) (non descendants of \( x_i \)) and \( \text{Descendants}_i \).

Let us denote:

\[ \text{Markov}(G) = \{ x_i \perp \text{ND}_i | \text{Par}_i : i = 1..n \} \]

The name 'Markov' is fitting: As in Markov Chains, if you know what was your situation in the last step, what happened before that is irrelevant.

All the independencies we have found in the examples belong to the Markov Blanket.
**Theorem 3.1** \( P \) is consistent with \( G \) (meaning: \( P(x_1...x_n) = \prod_i P(x_i|\text{Par}_i) \iff P \) satisfies Markov(\( G \))

The ‘⇒’ direction is simple. For the ‘⇐’ direction we need this simple lemma:

**Lemma 3.2** (no proof) \( X \perp Y_1, Y_2 | Z \Rightarrow X \perp Y_1 | Z \)

Now, assuming that the order of the indexes is the topological order in the graph:

\[
P(x_1...x_n) = \prod_i P(x_i|x_1...x_{i-1}) = \prod_i P(x_i|\text{Par}_i, z_i)
\]

When \( z_i = \{x_1...x_{i-1}\} - \text{Par}_i \).

Because of the topological order, \( z_i \) does not contain any descendants of \( x_1 \), so \( z_i \subseteq ND_i \)

\[
= \prod_i P(x_i|\text{Par}_i)
\]

### 3.2 More independencies derived from Markov(G)

![Figure 3: It is obvious that nodes E and F are supposed to be independent, but this independency is not in Markov(G)](image)

If \( A \) is observed, Markov(G) tells us that \( B \) and \( C \) are independent. What about \( E \) and \( F \)? It is obvious that they are supposed to be independent, but this independency is not included in Markov(G). There is an algorithm that can tell us if two variables, \( X \) and \( Y \), are dependent by looking at the graph. The intuition is simple: if there is a path between \( X \) and \( Y \), then there is some information that \( X \) can tell us about \( Y \) (and vice versa) so the variables are dependent.

Paths in the graph:

1. \( ...A \rightarrow B \rightarrow C... \)
2. \( ...A \leftarrow B \leftarrow C... \)
3. \( ...A \leftarrow B \rightarrow C... \)
4. \( ...A \rightarrow B \leftarrow C... \)

In the first three cases, if \( B \) is known than this path is ‘blocked’ - It doesn’t allow information from one variable to pass to another. If \( B \) is not observed, this path is active and can pass on information from one variable to another. In the fourth case, the path is blocked unless \( B \) is known - knowing about \( B \) (or knowing something about \( B \)’s descendants) makes his parents \( A \) and \( C \) dependent. This is called a V-structure.

Two variables are independent if all of the paths between them are blocked. This is called ‘direct separation’ or d-separation. Two groups of variables \( V \) and \( U \) are d-seperated given the values of group \( W \) iff \( V \) and \( U \) are independent given \( W \). Meaning, if there is a d-seperation between two sets of variables, then they will be independent in every distribution that is consistent with the graph. On the other hand, if there is no d-seperation, then there exists at least one distribution that is consistent with the graph where the variables will be dependent.
4 Example: Clustering of Gene Expression

There are $n$ genes that were tested in $m$ experiments. Let $C_1...C_n \in \{1...k\}$ be a random variable that tells us to which cluster the genes $x_1...x_n$ belong (heat shock, cell cycle) and $D[1]...D[m] \in \{1...l\}$ tells us to which cluster each of the $m$ experiment belongs to (stress, tissue).

The distribution according to this graph: $P(x_1[1]...x_n[m]) = \prod_{i,j} P(x_i[j]|C_i, D[j])$

If the $x_i[j]$'s are known, then the $C_1...C_n$ and $D[1]...D[m]$ are dependent, because there is an active path between them (see below). But, if we know the C’s, then $D[1]...D[m]$ are independent (because all the paths are blocked between $D_i$ and $D_j$). This allows us to do set the values of the D’s and then use Gibbs Sampling to guess the values of the C’s. After selecting the values of the C’s, we can guess the values of the D’s and so on.

Figure 4: The bold arrows show an active path between two C’s