# Chapter 7

# Graphical Models

Graphical models [Lau96, CGH97, Pea88, Jor98, Jen01] provide a pictorial representation of the way a joint probability distribution factorizes into a product of factors. They have been widely used in the area of computer vision, control theory, bioinformatics, communication, signal processing, sensor networks, neurovision, *etc.*

There is a relation between the conditional independence properties of a joint distribution and its factorization properties. Each of these properties can be represented graphically. This allows us to

1. organize complicated mathematics through graph based algorithms for calculation and computation and thus save complicated computations (in many cases, map the pictorial representation onto computations directly)

2. gain new insights into existing models; it is a standard practice to modify an existing model by addition/delection of nodes or links and adopt it to the problem at hand

3. motivate new models by simply ammending pictures

However, everything that you could do in graphical models using pictures could also be done without pictures, by grinding through all the mathematics while consistently using the innocuous-looking sum (7.1) and product (7.2) rules of probability

$$\Pr(X) = \sum_{y \in \mathcal{Y}} \Pr(X = x, Y = y) \tag{7.1}$$

$$\Pr(X = x, Y = y) = \Pr(X = x \mid Y = y) \Pr(Y = y) \tag{7.2}$$

where $X$ and $Y$ are discrete random variables, assuming values $x \in \mathcal{X} = \{x_1, x_2, \ldots, x_m\}$ and $y \in \mathcal{Y} = \{y_1, y_2, \ldots, y_n\}$ respectively. A combination of the two rules yields the Bayes theorem:

$$\begin{aligned} \Pr(X = x_i, Y = y_j) &= \frac{\Pr(X = x_i \mid Y = y_j)\Pr(Y = y_j)}{\Pr(X = x_i)} \\ &= \frac{\Pr(X = x_i \mid Y = y_j)\Pr(Y = y_j)}{\sum\limits_{y_j \in \mathcal{Y}} \Pr(X = x_i \mid Y = y_j)\Pr(Y = y_j)} \end{aligned} \tag{7.3}$$

The two main kinds of graphical models are *directed* and *undirected* models. The problems we will address in graphical models include

1. *Inference:* Broadly, there are two inference techniques for graphical models, *viz.*, exact and approximate inference. Exact inference is appropriate if the graphic is a tree, since it is a linear time algorithm. But for complex graphical models, exact inference may or may not be appropriate, since exact algorithms could be very slow. In such cases, approximate inference schemes are often resorted to. Markov chain monte carlo (which is exact if there were an infinite amount of computing resources and approximate otherwise) and variational inference (by approximating the analytical form for the posterior distribution) are two popular techniques. While variational techniques scale better, their other strengths and weaknesses are complementary to those of MCMC. An often adopted stepping stone for explaining variational inference is the expectation maximization algorithm (EM) and we will take the same route.

2. *Learning:*

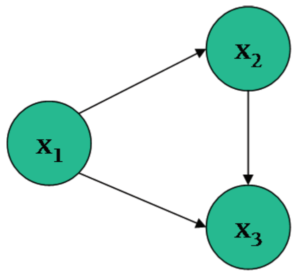
## 7.1 Semantics of Graphical Models

We will first discuss the semantics of graphical models, both directed and undirected. In the sections that follow, we will discuss the computational aspects of graphical models - in particular, inferencing and learning techniques.

### 7.1.1 Directed Graphical Models

We will start with the example of a directed graphical model. Consider an arbitrary joint distribution $\Pr(X_1 = x_1, X_2 = x_2, X_3 = x_3)$ over three discrete random variables $X_1, X_2$ and $X_3$ that assume values $x_1 \in \mathcal{X}_1$, $x_2 \in \mathcal{X}_2$ and $x_3 \in \mathcal{X}_3$ respectively. Denoting[1] $\Pr(X_i = x_i)$ by $p(x_i)$ and $\Pr(X_1 = x_1, X_2 = x_2, X_3 = x_3)$ by $p(x_1, x_2, x_3)$ and applying the product rule of probability successively, we obtain

[1]As a convention, we will use capital letters to denote random variables and lower case letters to denote their realizations.



$$p(x_1, x_2, x_3) = p(x_1)p(x_2, x_3 \mid x_1) = p(x_1)p(x_2 \mid x_1)p(x_3 \mid x_1, x_2) \qquad (7.4)$$

The successive application of the product rule is often termed as the *chain rule*. We should note that this rule applies even if $x_1$, $x_2$ and $x_3$ happen to be continuous random variables (in which case $p$ is a density function) or vectors of random variables. We should note that this factorization is quite non-symmetrical in the three random variables. This factorization can be represented in the form of the following directed graph: There is one node for each variable. We draw a directed edge between every conditioning variable (*i.e.* the corresponding node) to the conditioned variable. The way to go from a graph to the factorization of the joint distribution is to write down the product of the conditional distribution of every node (*i.e.*, the corresponding variable) conditioned on its parents within the graph. In the example above, $x_1$ had no parent, and therefore the term corresponds to its conditional $p(x_1)$ turned out to be its marginal distribution.

The factorization in the last example holds for any joint distribution over any three variables and the graph is therefore uninformative. In fact, any completely connected graph will be uninformative, as we will soon see. What interests us in graphical models is not the *presence* of edges but rather, the *absence* of edges. Since the graph in the previous example had no missing edges, it was uninteresting.

**Definition 43** *Let $\mathcal{R} = \{X_1, X_2, \ldots, X_n\}$ be a set of random variables, with each $X_i$ ($1 \le i \le n$) assuming values $x_i \in \mathcal{X}_i$. Let $\mathcal{X}_S = \{X_i \mid i \in S\}$ where $S \subseteq \{1, 2, \ldots, n\}$. Let $\mathcal{G} =< \mathcal{V}, \mathcal{E} >$ be a directed acyclic graph with vertices $\mathcal{V} = \{1, 2, \ldots, n\}$ and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ such that each edge $e = (i, j) \in \mathcal{E}$ is a directed edge. We will assume a one to one correspondence between the set of variables $\mathcal{R}$ and the vertex set $\mathcal{V}$; vertex $i$ will correspond to random variable $X_i$. Let $\pi_i$ be the set of vertices from which there is edge incident on vertex $i$. That is, $\pi_i = \{j \mid j \in \mathcal{V},\ (j, i) \in \mathcal{E}\}$. Then, the family $\mathcal{F}(\mathcal{G})$ of joint distributions associated with the DAG[2] $\mathcal{G}$ is specified by the factorization induced by $\mathcal{G}$ as follows:*

[2] As we saw earlier, the family of probability distributions specified by the related formalism of undirected graphical models is somewhat different.

$$\mathcal{F}(\mathcal{G}) = \left\{ p(\mathbf{x}) \middle| p(\mathbf{x}) = \prod_{i=1}^{n} p(x_i \mid \mathbf{x}_{\pi_i}),\ p(x_i \mid \mathbf{x}_{\pi_i}) \forall\ 1 \leq i \leq n \geq 0,\ \sum_{x_i \in \mathcal{X}_i} p(x_i \mid \mathbf{x}_{\pi_i}) = 1 \right\} \tag{7.5}$$

*where,* $\mathbf{x}$ *denotes the vector of values* $[x_1, x_2, \ldots, x_n]$ *and* $x_i$ *is the value assumed by random variable* $X_i$ *and* $\mathbf{x}_{\pi_i}$ *denotes the vector of values from* $\mathbf{x}$*, composed from positions in* $\pi_i$*.*

For notational convenience with directed acyclic graphs, it is a common practice to assume a topological ordering on the indices of vertices in $\mathcal{V}$ so that $\pi_i \subseteq \mu_{i-1} = \{1, 2, \ldots, i-1\}$. Note that, by the chain rule, the following factorization always holds:

$$p(\mathbf{x}) = \prod_{i=1}^{n} p(x_i \mid \mathbf{x}_{\mu_{i-1}}) \tag{7.6}$$

Making use of the sum rule, in conjunction with (7.5), for any $p \in \mathcal{F}(\mathcal{G})$, we have

$$\begin{aligned} p(\mathbf{x}_{\mu_i}) &= \sum_{x_{i+1} \in \mathcal{X}_{i+1}, \ldots, x_n \in \mathcal{X}_n} p(\mathbf{x}) \\ &= \sum_{x_{i+1} \in \mathcal{X}_{i+1}, \ldots, x_n \in \mathcal{X}_n} p(x_1) p(x_2 \mid \mathbf{x}_{\pi_2}) \ldots p(x_i \mid \mathbf{x}_{\pi_i}) \end{aligned} \tag{7.7}$$

Since the vertex indices are topologically ordered, it can be proved using the principle of induction (working backwards from $n$) on the basis of the sum rule in (7.7), that for any $p \in \mathcal{F}(\mathcal{G})$:

$$p(\mathbf{x}_{\mu_i}) = \prod_{j=1}^{i-1} p(x_i \mid \mathbf{x}_{\pi_i}) \tag{7.8}$$

Contrasting (7.6) against (7.5), we can think of the set of probability distribution $\mathcal{F}(\mathcal{G})$ as a sort of restricted class of distributions that arises from throwing away some of the dependencies. In particular, if $p \in \mathcal{F}(\mathcal{G})$ then

$$p(x_i \mid \mathbf{x}_{\mu_{i-1}}) = p(x_i \mid \mathbf{x}_{\pi_i})$$

that is, $X_i$ is independent of $\mathbf{X}_{\mu_{i-1}}$, given $\mathbf{X}_{\pi_i}$. The independence is denoted by: $X_i \perp \mathbf{X}_{\mu_{i-1}} \mid \mathbf{X}_{\pi_i}$. This leads us to another approach to defining the class of probability distributions based on a DAG $\mathcal{G}$.

**Definition 44** *Let $\mathcal{R} = \{X_1, X_2, \ldots, X_n\}$ be a set of random variables, with each $X_i$ ($1 \leq i \leq n$) assuming values $x_i \in \mathcal{X}_i$. Let $\mathbf{X}_\mathcal{S} = \{X_i \mid i \in \mathcal{S}\}$ where $\mathcal{S} \subseteq \{1, 2, \ldots, n\}$. Let $\mathcal{G} =< \mathcal{V}, \mathcal{E} >$ be a directed acyclic graph with vertices $\mathcal{V} = \{1, 2, \ldots, n\}$ and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ such that each edge $e = (i,j)$ is a directed edge. Let $\pi_i = \{j \mid j \in \mathcal{V},\ (j,i) \in \mathcal{E}\}$. Then, the family $\mathcal{C}(\mathcal{G})$ of joint distributions associated with the DAG $\mathcal{G}$ is specified by the conditional independence induced by $\mathcal{G}$ as follows:*

$$\mathcal{C}(\mathcal{G}) = \left\{ p(\mathbf{x}) \middle| X_i \perp \mathbf{X}_{\mu_{i-1}} \mid \mathbf{X}_{\pi_i}\ \forall\ 1 \leq i \leq n,\ \sum_{\mathbf{x}} p(\mathbf{x}) = 1 \right\} \tag{7.9}$$

We will next show that the class $\mathcal{F}(\mathcal{G})$ defined in terms of factorizations is equivalent to the class $\mathcal{C}(\mathcal{G})$ defined in terms of independences. This is called the Hammersley Clifford theorem.

**Theorem 87** *The sets $\mathcal{F}(\mathcal{G})$ and $\mathcal{C}(\mathcal{G})$ are equal. That is $p \in \mathcal{F}(\mathcal{G})$ iff $p \in \mathcal{C}(\mathcal{G})$*

*Proof:* $\Leftarrow$: We will first prove that $\mathcal{F}(\mathcal{G}) \subseteq \mathcal{C}(\mathcal{G})$. Let $p \in \mathcal{F}(\mathcal{G})$. We will prove that $p \in \mathcal{C}(\mathcal{G})$, that is, $p(x_i \mid \mathbf{x}_{\mu_{i-1}}, \mathbf{x}_{\pi_i}) = p(x_i \mid \mathbf{x}_{\pi_i})$. This trivially holds for $i = 1$, since $\mathbf{x}_{\pi_i} = \emptyset$. For $i = 2$:

$$p(x_1, x_2) = p(x_1)p(x_1 \mid x_2) = p(x_1)p(x_1 \mid \mathbf{x}_{\pi_2})$$

where, the first equality follows by chain rule, whereas the second equality follows by virtue of (7.8). Consequently,

$$p(x_1 \mid x_2) = p(x_1 \mid \mathbf{x}_{\pi_2})$$

Assume that $p(x_i \mid \mathbf{x}_{\mu_{i-1}}) = p(x_i \mid \mathbf{x}_{\pi_i})$ for $i \leq k$. For $i = k+1$, it follows from chain rule and from (7.8) that

$$p(\mathbf{x}_{\mu_{k+1}}) = \prod_{i=1}^{k+1} p(x_i \mid \mathbf{x}_{\mu_{i-1}}) = \prod_{i=1}^{k+1} p(x_i \mid \mathbf{x}_{\pi_i})$$

Making use of the induction assumption for $i \leq k$ in the equation above, we can derive that

$$p(x_k \mid \mathbf{x}_{\mu_{k-1}}) = p(x_k \mid \mathbf{x}_{\pi_k})$$

By induction on $i$, we obtain that $p(x_i \mid \mathbf{x}_{\mu_{i-1}}) = p(x_k \mid \mathbf{x}_{\pi_i})$ for all $i$. That is, $p \in \mathcal{C}(\mathcal{G})$. Since this holds for any $p \in \mathcal{F}(\mathcal{G})$, we must have that $\mathcal{F}(\mathcal{G}) \subseteq \mathcal{C}(\mathcal{G})$.

$\Rightarrow$: Next we prove that $\mathcal{C}(\mathcal{G}) \subseteq \mathcal{F}(\mathcal{G})$. Let $p' \in \mathcal{C}(\mathcal{G})$ satisfy the conditional independence assertions. That is, for any $1 \leq i \leq n$, $p'(x_i \mid \mathbf{x}_{\mu_{i-1}}) = p'(x_i \mid \mathbf{x}_{\pi_i})$. Then by chain rule, we must have:

$$p'(\mathbf{x}_{\mu_n}) = \prod_{i=1}^{n} p'(x_i \mid \mathbf{x}_{\mu_{i-1}}) = \prod_{i=1}^{k+1} p'(x_i \mid \mathbf{x}_{\pi_i})$$

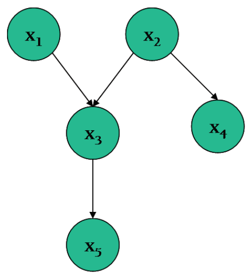


Figure 7.1: A directed graphical model.

which proves that $p' \in \mathcal{F}(\mathcal{G})$ and consequently that $\mathcal{C}(\mathcal{G}) \subseteq \mathcal{F}(\mathcal{G})$ $\Box$

As an example, we will discuss the directed graphical model, as shown in Figure 7.1. Based on theorem 87, the following conclusions can be drawn from the graphical representation of a family of distributions represented by Figure 7.1.

1. Given the value of $X_3$, the values of $X_1, X_2$ and $X_4$ will be completely uninformative about the value of $X_5$. That is, $(X_5 \perp \{X_1, X_2, X_4\} \mid \{X_3\})$. Similarly, given the value of $X_2$, the values of $X_1$ and $X_3$ will be completely uninformative about the value of $X_4$. That is, $(X_4 \perp \{X_1, X_3\} \mid \{X_2\})$.

2. Secondly, since $p \in \mathcal{F}(\mathcal{G})$, we have

$$p(x_1, x_2, x_3, x_4, x_5) = p(x_1)p(x_2)p(x_3 \mid x_1, x_2)p(x_4 \mid x_2)p(x_5|x_3)$$

What about other independence assertions? Is $X_5$ independent of $X_4$ given $X_2$? Is $X_3$ independent of $X_4$, given $X_2$? The answer to both these questions happens to be yes. And these could be derived using either of the equivalent definitions of graphical models. In fact, some such additional conditional independence assertions can always follow from the two equivalent definitions of graphical models. Before delving into these properties, we will define an important concept called d-separation, introduced by Pearl [Pea88].

**Definition 45** *A set of nodes $\mathcal{A}$ in a directed acyclic graph $\mathcal{G}$ is d-separated from a set of nodes $\mathcal{B}$ by a set of nodes $\mathcal{C}$, iff* **every** *undirected path from a vertex $A \in \mathcal{A}$ to a vertex $B \in \mathcal{B}$ is 'blocked'. An undirected path between $A$ and $B$ is blocked by a node $C$ either (i) if $C \in \mathcal{C}$ and both the edges (which might be the same) on the path through $C$ are directed away from $C$ ($C$ is then called a tail-to-tail node) or (ii) if $C \in \mathcal{C}$ and of the two edges (which might be the same) on the path through $C$, one is directed toward $C$ while the other is directed away from $C$ ($C$ is then called a head-to-tail node) or (iii) if $C \notin \mathcal{C}$ and both the edges (which might be the same) on the path through $C$ are directed toward $C$ ($C$ is then called a head-to-head node).*

In Figure 7.1, node $X_2$ blocks the only path between $X_3$ and $X_4$, node $X_3$ blocks the path between $X_2$ and $X_5$. Whereas, node $X_3$ does not block the path between $X_1$ and $X_2$. Consequently, $\{X_3\}$ and $\{X_4\}$ are d-separated by $\{X_2\}$, while $\{X_2\}$ and $\{X_5\}$ are d-separated by $\{X_3\}$. However, $\{X_1\}$ and $\{X_2\}$ are not d-separated by $\{X_3\}$, since $X_3$ is a head-to-head node. $X_3$ does not d-separate $X_1$ and $X_2$ even though it separates them. Thus, not every pair of (graph) separated nodes in the graph need be d-separated. We next define a family of probability distribution that have independences characterized by d-separation.

**Definition 46** *The set of probability distributions $\mathcal{D}(\mathcal{G})$ for a DAG $\mathcal{G}$ is defined as follows:*

$$\mathcal{D}(\mathcal{G}) = \{p(\mathbf{x})\,|\mathbf{X}_{\mathcal{A}} \perp \mathbf{X}_{\mathcal{B}} \mid \mathbf{X}_{\mathcal{C}},\ \textit{whenever } \mathcal{A} \textit{ and } \mathcal{B} \textit{ are } d-\textit{separated by } \mathcal{C}\} \tag{7.10}$$

It can be proved that the notion of conditional independence is equivalent to the notion of d-separation in DAGs. That is,

**Theorem 88** *For any directed acyclic graph $\mathcal{G}$, $\mathcal{D}(\mathcal{G}) = \mathcal{C}(\mathcal{G}) = \mathcal{F}(\mathcal{G})$.*

Thus, in Figure 7.1. $\{X_3\} \perp \{X_4\} \mid \{X_2\}$ and $\{X_2\} \perp \{X_5\} \mid \{X_3\}$. Whereas, $\{X_1\} \not\perp \{X_2\} \mid \{X_3\}$. We could think of $X_1$ and $X_2$ as completing explanations for $X_3$. Thus, given a value of $X_3$, any value of $X_1$ will, to some extent 'explain away' the value of $X_3$, thus withdrawing the independence of $X_2$. In terms of a real life example, if $X_1$, $X_2$ and $X_3$ are discrete random variables corresponding to 'the color of light', 'the surface color' and 'the image color' respectively, then, given the value of $X_3$ (image color), any value of $X_1$ (color of light) will explain away the color of the image, thus constraining the values that $X_3$ (surface color) might take. On the other hand, $\{X_1\} \perp \{X_2\} \mid \{\}$. What about the independence of $X_1$ and $X_2$ given $X_5$? The path $X_1, X_3, X_5, X_3, X_2$ involves a head-to-head node $X_5$ and therefore, $X_1 \not\perp X_2 \mid \{X_5\}$. The Bayes ball algorithm provides a convenient algorithmic way for deciding if $\mathbf{X}_{\mathcal{A}} \perp \mathbf{X}_{\mathcal{B}} \mid \mathbf{X}_{\mathcal{C}}$, by using the d-separation property.

## Bayesian Networks and Logic

The logical component of Bayesian networks essentially corresponds to a propositional logic program. This has already been observed by Haddawy [1994] and Langley [1995]. Langley, for instance, does not represent Bayesian networks graphically but rather uses the notation of propositional definite clause programs. Consider the following program. This program encodes the structure of the blood type Bayesian network in Figure 7.2. Observe that the random variables in this notation correspond to logical atoms. Furthermore, the direct influence relation in the Bayesian network corresponds to the immediate consequence operator.

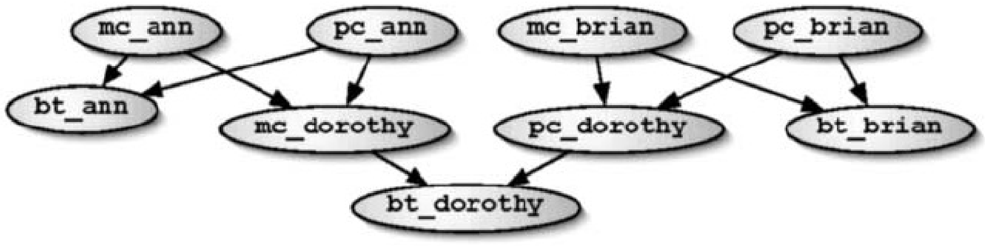


Figure 7.2: The graphical structure of a Bayesian network modeling the inheritance of blood types within a particular family.

$PC(ann).$
$MC(ann).$
$MC(dorothy) : -MC(ann), PC(ann).$
$BT(ann) : -MC(ann), PC(ann).$
$BT(dorothy) : -MC(dorothy), PC(dorothy).$

$PC(brian).$
$MC(brian).$
$PC(dorothy) : -MC(brian), PC(brian).$
$BT(brian) : -MC(brian), PC(brian).$

### 7.1.2 Undirected Graphical Models

We will move on to an undirected graphical models (also known as Markov Random fields) primer, while drawing parallels with the directed counterpart. An undirected graph $\mathcal{G}$ is a tuple $< \mathcal{V}, \mathcal{E} >$ where $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ and such that each edge $e = (i, j) \in \mathcal{E}$ is a directed edge. While the conditional independence property for directed graphical models was tricky (involving concepts such as d-separation, *etc.*), the conditional independence property for undirected models is easier to state. On the other hand, the factorization property for directed graphical models simply involved local conditional probabilities as factors. It is however not as simple with undirected graphical models. Taking the easier route, we will first define the conditional independence property for undirected graphical models. Toward that, we introduce the notion of graph separation.

**Definition 47** *Given an undirected graph $\mathcal{G} =< \mathcal{V}, \mathcal{E} >$, and $\mathcal{A}, \mathcal{B}, \mathcal{C} \subseteq \mathcal{V}$, we say that $\mathcal{C}$ separates $\mathcal{A}$ from $\mathcal{B}$ in $\mathcal{G}$ if every path from any node $A \in \mathcal{A}$ to any node $B \in \mathcal{B}$ passes through some node $C \in \mathcal{C}$. $C$ is also called a separator or a vertex cut set in $\mathcal{G}$.*

In Figure 7.3, the set $\mathcal{A} = \{X_1, X_7, X_8\}$ is separated from $\mathcal{B} = \{X_3, X_4, X_5\}$ by the (vertex cut) set $\mathcal{C} = \{X_2, X_6\}$. It is easy to see that separation is symmetric in $\mathcal{A}$ and $\mathcal{B}$. This simple notion of separation gives rise to a conditional indpendence assertion for undirected graphs. A random vector $\mathbf{X}_{\mathcal{C}}$ for $\mathcal{C} \subseteq \mathcal{V}$

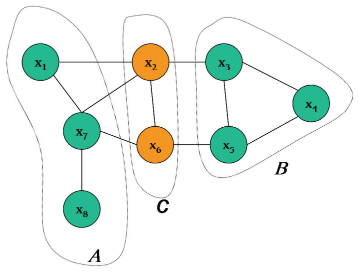


Figure 7.3: A directed graphical model.

is said to be *markov* with respect to a graph $\mathcal{G}$ if $\mathbf{X}_{\mathcal{A}} \perp \mathbf{X}_{\mathcal{B}} \mid \mathbf{X}_{\mathcal{C}}$ whenever $\mathcal{C}$ separates $\mathcal{A}$ from $\mathcal{B}$. That is, the random variables corresponding to the vertex cut set acts like a mediator between the values assumed by variables in $\mathcal{A}$ and $\mathcal{B}$ so that the variables in $\mathcal{A}$ and $\mathcal{B}$ are independent of each other, if we knew the values of variables in $\mathcal{C}$. It is straightforward to develop a 'reachability' algorithm (as with the bayes ball algorithm) for undirected graphs, to assess conditional independence assumptions. Based on the definition of markov random vector, we next define the family $\mathcal{M}(\mathcal{G})$ of distributions associated with an undirected graph $\mathcal{G}$.

**Definition 48** *Let $\mathcal{R} = \{X_1, X_2, \ldots, X_n\}$ be a set of random variables, with each $X_i$ ($1 \leq i \leq n$) assuming values $x_i \in \mathcal{X}_i$. Let $\mathbf{X}_{\mathcal{S}} = \{X_i \mid i \in \mathcal{S}\}$ where $\mathcal{S} \subseteq \{1, 2, \ldots, n\}$. Let $\mathcal{G} =< \mathcal{V}, \mathcal{E} >$ be an undirected graph with vertices $\mathcal{V} = \{1, 2, \ldots, n\}$ and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ such that each edge $e = (i, j) \in \mathcal{E}$ is an undirected edge. Then, the family $\mathcal{M}(\mathcal{G})$ of joint distributions associated with $\mathcal{G}$ is specified as follows:*

$$\mathcal{M}(\mathcal{G}) = \{p(\mathbf{x})\,|\mathbf{X}_{\mathcal{A}} \perp \mathbf{X}_{\mathcal{B}} \mid \mathbf{X}_{\mathcal{C}}\ \forall\ \mathcal{A}, \mathcal{B}, \mathcal{C} \subseteq \mathcal{V},\ \textit{whenever}\ \mathcal{C}\ \textit{seperates}\ \mathcal{A}\ \textit{from}\ \mathcal{B}\} \tag{7.11}$$

As in the case of directed models, there is another family of probability distributions that could be specified for a given undirected graph $\mathcal{G}$ based on a factorization assertion. The main difference is that, while the factorization for DAGs was obtained in terms of local conditional probabilities or local marginals, it turns out that this factorization is not possible for a general undirected graph (specifically when it has a cycle). Instead there is another notion of 'local' for undirected graphs – there should be no function involving any two variables $X_i$ and $X_j$ where $(i, j) \notin \mathcal{E}$ (otherwise, such a term will not break further, prohibiting assertions about conditional independences). Instead, we will have a function $\phi_{\mathcal{C}}(\mathbf{X}_{\mathcal{C}})$ for every clique $\mathcal{C} \subseteq \mathcal{V}$, since a clique is a subset of vertices that all 'talk to' one another. The most general version of factorization will be one for which there is a function corresponding to each *maximal clique*; all other factorizations involving factors over smaller cliques will be specialized versions. These functions will be referred to as *compatibility* or *potential* functions. A *potential* or *compatibility* function on a clique $\mathcal{C}$ is a non-negative real valued function $\phi_{\mathcal{C}}(\mathbf{x}_{\mathcal{C}})$ defined over all instantiations $\mathbf{x} \in \mathcal{X}_1 \times \mathcal{X}_2 \times \ldots \times \mathcal{X}_n$ of $\mathbf{X}$. Other than these restrictions, the potential function can be arbitrary.

**Definition 49** *Let $\mathcal{R} = \{X_1, X_2, \ldots, X_n\}$ be a set of random variables, with each $X_i$ ($1 \leq i \leq n$) assuming values $x_i \in \mathcal{X}_i$. Let $\mathbf{X}_{\mathcal{S}} = \{X_i \mid i \in \mathcal{S}\}$ where $\mathcal{S} \subseteq \{1, 2, \ldots, n\}$. Let $\mathcal{G} =< \mathcal{V}, \mathcal{E} >$ be an undirected graph with vertices $\mathcal{V} = \{1, 2, \ldots, n\}$ and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ such that each edge $e = (i, j) \in \mathcal{E}$ is an undirected edge. Then, the family $\mathcal{M}(\mathcal{G})$ of joint distributions associated with $\mathcal{G}$ is specified as follows:*

$$\mathcal{F}(\mathcal{G}) = \left\{ p(\mathbf{x}) \middle| p(\mathbf{x}) = \frac{1}{Z} \prod_{\mathcal{C} \in \Pi} \phi_{\mathcal{C}}(\mathbf{x}_{\mathcal{C}}),\ \textit{such that } \phi_{\mathcal{C}}(\mathbf{x}_{\mathcal{C}}) \in \Re^+, \forall \mathcal{C} \in \Pi \right\} \tag{7.12}$$

*where, $\Pi$ is the set of all cliques in $\mathcal{G}$, $\mathbf{x}$ denotes the vector of values $[x_1, x_2, \ldots, x_n]$, $x_i \in \mathcal{X}_i$ is the value assumed by random variable $X_i$ and where each $\phi_{\mathcal{C}}$ is a potential function defined over the clique $\mathcal{C} \subseteq \mathcal{V}$. Without loss of generality, we can assume that $\Pi$ is the set of maximal cliques in $\mathcal{G}$. The normalization constant $Z$ is called the partition function and is given by*

$$Z = \sum_{\mathbf{x}_1 \in \mathcal{X}_1, \ldots, \mathbf{x}_n \in \mathcal{X}_n} \prod_{\mathcal{C} \in \Pi} \phi_{\mathcal{C}}(\mathbf{x}_{\mathcal{C}}) \tag{7.13}$$

The potential functions are typically represented as tables – each row listing a unique assignment of values to the random variables in the clique and the corresponding potential. Thus, the value $\phi_{\mathcal{C}}(\mathbf{x}_{\mathcal{C}})$ can be obtained by a simple table lookup.

The form of the potential function can be chosen based on the particular application at hand. For instance, the clique potential can be decomposed into a product of potentials defined over each edge of the graph. When the domain of each random variable is the same, the form of the potential can be chosen to either encourage or discourage similar configurations (such as similar disease infection for patients who are related) at adjacent nodes. The potential function is often interpreted as an energy function in the modeling of crystals, protein folding, *etc.*, where a minimum energy configuration is desirable. Frequently, the energy function is assumed to have the form $\phi_{\mathcal{C}}(\mathbf{X}_{\mathcal{C}}) = \exp(-\theta_{\mathcal{C}}(\mathbf{X}_{\mathcal{C}}))$, which leads to the factorization as an exponential form distribution $p(\mathbf{x}) = \exp(-\sum_{\mathcal{C} \in \Pi} \theta_{\mathcal{C}}(\mathbf{x}_{\mathcal{C}}) - \log Z)$. The quantities $\theta_{\mathcal{C}}(\mathbf{X}_{\mathcal{C}})$ are called sufficient statistics.

From our discussion on the equivalence of directed and undirected trees in terms of conditional independencies, we may be tempted to conclude that the factors for the undirected tree can be the local conditional probabilities. This is easily established if we prove that for strictly positive distributions, the definition of an undirected graphical model in terms of conditional independences is equivalent to the definition in terms of factorization, that is, $\mathcal{M}(\mathcal{G}) = \mathcal{F}(\mathcal{G})$.

**Theorem 89** *For strictly positive distributions, $\mathcal{M}(\mathcal{G}) = \mathcal{F}(\mathcal{G})$. That is, $\mathcal{M}(\mathcal{G}) \cap \mathcal{D}^+ = \mathcal{F}(\mathcal{G}) \cap \mathcal{D}^+$, where, $\mathcal{D}^+ = \{p(\mathbf{x})\,|p(\mathbf{x}) > 0, \forall \mathbf{x} \in \mathcal{X}_1 \times \mathcal{X}_2 \ldots \times \mathcal{X}_n\}$.*

*Proof:* The proof that $\mathcal{M}(\mathcal{G}) \subseteq \mathcal{F}(\mathcal{G})$ is a bit involved and requires Mobius inversion. On the other hand, that $\mathcal{F}(\mathcal{G}) \subseteq \mathcal{M}(\mathcal{G})$ can be shown as follows. For any given $\mathcal{A}, \mathcal{B} \subseteq \mathcal{V}$ that are separated by $\mathcal{C} \subseteq \mathcal{V}$, consider the partitions $\mathcal{P}_1$, $\mathcal{P}_2$ and $\mathcal{P}_3$ of $\Pi$:

$$\mathcal{P}_1 = \{\mathcal{K}\,|\mathcal{K} \in \Pi and \mathcal{K} \subseteq \mathcal{A} \cap \mathcal{C} \text{ and } \mathcal{K} \nsubseteq \mathcal{C}\}$$

$$\mathcal{P}_2 = \{\mathcal{K}\,|\mathcal{K} \in \Pi and \mathcal{K} \subseteq \mathcal{B} \cap \mathcal{C} \text{ and } \mathcal{K} \nsubseteq \mathcal{C}\}$$

$$\mathcal{P}_3 = \{\mathcal{K}\,|\mathcal{K} \in \Pi and \mathcal{K} \subseteq \mathcal{C}\}$$

Now, $p(\mathbf{x})$ can be factorized into factors involving cliques in $\mathcal{P}_1$, $\mathcal{P}_2$ and $\mathcal{P}_3$. Consequently,

$$\frac{p(\mathbf{x}_\mathcal{A}, \mathbf{x}_\mathcal{B}, \mathbf{x}_\mathcal{C})}{p(\mathbf{x}_\mathcal{B}, \mathbf{x}_\mathcal{C})} = \frac{\prod\limits_{\mathcal{K}\in\mathcal{P}_1} \phi_\mathcal{K}(\mathbf{x}_\mathcal{K}) \prod\limits_{\mathcal{K}\in\mathcal{P}_2} \phi_\mathcal{K}(\mathbf{x}_\mathcal{K}) \prod\limits_{\mathcal{K}\in\mathcal{P}_3} \phi_\mathcal{K}(\mathbf{x}_\mathcal{K})}{\sum\limits_{\mathbf{x}_\mathcal{A}} \prod\limits_{\mathcal{K}\in\mathcal{P}_1} \phi_\mathcal{K}(\mathbf{x}_\mathcal{K}) \prod\limits_{\mathcal{K}\in\mathcal{P}_2} \phi_\mathcal{K}(\mathbf{x}_\mathcal{K}) \prod\limits_{\mathcal{K}\in\mathcal{P}_2} \phi_\mathcal{K}(\mathbf{x}_\mathcal{K})} = \frac{\prod\limits_{\mathcal{K}\subseteq\mathcal{A}\cup\mathcal{C}} \phi_\mathcal{K}(\mathbf{x}_\mathcal{K})}{\sum\limits_{\mathbf{x}_\mathcal{A}} \prod\limits_{\mathcal{K}\subseteq\mathcal{A}\cup\mathcal{C}} \phi_\mathcal{K}(\mathbf{x}_\mathcal{K})} = p(\mathbf{x}_\mathcal{A} \mid \mathbf{x}_\mathcal{C})$$

$\square$

While conditional independence is useful for modeling purposes, factorization is more useful for computatinal purposes.

### 7.1.3 Comparison between directed and undirected graphical models

Is there any difference between the undirected and directed formalisms? Are they equally powerful? Or is more powerful than the other. It turns out that there are families of probability distributions which can be represented using undirected models, whereas they have no directed counterparts. Figure 7.4 shows one such example. Imagine that random variables $X_1$ and $X_3$ represent hubs in a computer network, while $X_2$ and $X_4$ represent computers in the network. Computers do not interact directly, but only through hubs. Similarly, hubs interact only through computers. This leads to two independences: (i) conditioned on $X_1$ and $X_3$ (the hubs), nodes $X_2$ and $X_4$ (the computers) become independent and (ii) conditioned on $X_2$ and $X_4$, nodes $X_1$ and $X_3$ become independent. However, with a directed acyclic graph on four nodes, we will always have some head-to-head node and therefore, it is impossible to simultanesouly satisfy both conditions (i) and (ii) using a DAG. Larger bipartitie graphs

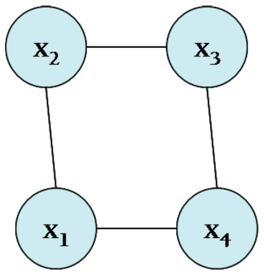


Figure 7.4: An undirected graphical model which has no equivalent directed model.

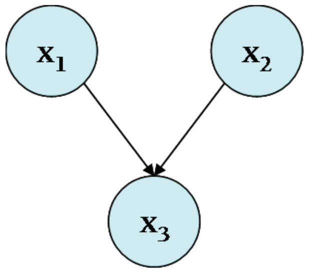


Figure 7.5: A directed graphical model which has no equivalent undirected model.

will have similar conditional independence assertions, which are inexpressible through DAGs.

Similarly, there are familiies of probability distibutions which can be represented using directed models, whereas they have no undirected counterparts. Figure 7.5 shows one such example and corresponds to the 'explaining away' phenomenon, which we discussed earlier in this chapter. The node $X_3$ is blocked if not observed (so that $\{X_1\} \perp \{X_2\} \mid \emptyset$), whereas it is unblocked if its value is known (so that $\{X_1\} \not\perp \{X_2\} \mid \{X_3\}$). Can you get this behaviour with an undirected graph? The answer is no. This is because, with an undirected graph, there is no way of getting dependence between $X_1$ and $X_2$ if they were apriori independent.

On the other hand, for graphical models such as markov chains, dropping the arrows on the edges preserves the independencies, yielding an equivalent undirected graphical model. Similarly, directed trees are fundamentally no different from undirected trees.

An important point to note is that it is the absence of edges that characterizes a graphical model. For any graphical model, it is possible that the compatibility functions (or local conditional probabilities) assume a very special form so that there are more (conditional) independences that hold than what is indicated by the graphical model (which means that some of the edges in the graph could be redundant).

## 7.2 Inference

In this section, we discuss the problem of determining the marginal distribution $p(\mathbf{x}_{\mathcal{A}})$, the conditional distribution $p(\mathbf{x}_{\mathcal{A}} \mid \mathbf{x}_{\mathcal{B}})$ and the partition function $Z$, given a graphical model $\mathcal{G} =< \mathcal{V}, \mathcal{E} >$ and for any $\mathcal{A}, \mathcal{B} \subseteq \mathcal{V}$. We will assume that the conditional probability tables (for directed models) or the potential function table (for undirected models) are already known. The sum and product rules of probability yield the following formulae[3] for the marginal, the conditional[4] and the partition function[5] respectively:

$$p(\mathbf{x}_{\mathcal{A}}) = \sum_{\mathbf{x}_{\mathcal{V}\setminus\mathcal{A}}} p(\mathbf{x})$$

$$p(\mathbf{x}_{\mathcal{A}} \mid \mathbf{x}_{\mathcal{O}}) = \frac{p(\mathbf{x}_{\mathcal{A}}, \mathbf{x}_{\mathcal{O}})}{p(\mathbf{x}_{\mathcal{O}})}$$

$$Z = \sum_{\mathbf{x}\in\mathcal{X}_1\times\mathcal{X}_2\times\ldots\mathcal{X}_n} \prod_{\mathcal{C}\in\Pi} \phi_{\mathcal{C}}(\mathbf{x}_{\mathcal{C}})$$

All these problems are somewhat similar, in that they involve summation over a very high dimensional space. Computing any of these quantities will involve number of computations that are atleast exponential in the size of $\mathcal{V}\setminus\mathcal{A}$. This is not feasible in many practical applications, where the number of nodes will run into the hundreds or the thousands. As we will see, there is ample redundancy in these computations. We will briefly discuss a simple and pedagogical algorithm called the *elimination algorithm* that provides the intuition as to how the structure of the graph could be exploited to answer some of the questions listed above. More clever algorithms such as the *sum-product algorithm* that captures redundancies more efficiently will be discussed subsequently.

A problem fundamentally different from the three listed above is that of *maximum aposteriori optimization* - determining the mode of a conditional distribution.

$$\widehat{\mathbf{x}}_{\mathcal{A}} = \underset{\mathbf{x}_{\mathcal{A}}}{\operatorname{argmax}}\ p(\mathbf{x}_{\mathcal{A}} \mid \mathbf{x}_{\mathcal{O}})$$

For discrete problems, this is an integer linear programming problem. For general graphs, you cannot do much better than a brute force, whereas, for special graphs (such as trees), this mode can be computed efficiently.

[3]For continuous valued random variables, you can expect the summation to be replaced by integration in each formula.

[4]The problem of computing conditionals is not fundamentally different from the problem of computing the marginals, since every conditional is simply the ratio of two marginals.

[5]The partition function needs to be computed for parameter estimation.

### 7.2.1 Elimination Algorithm

The elimination algorithm provides a systematic framework for optimizing computations by rearranging sums and products, an instance of which we saw in the proof of theorem 89. Consider the simple directed graph in Figure 7.1. Let us say each random variable takes values from the set $\{v_1, v_2, \ldots, v_k\}$. Brute force computation of $p(x_1 \mid x_5) = \frac{p(x_1,x_5)}{p(x_5)}$ will involve $k \times k^3 = k^4$ computations for the numerator and $k \times k^4 = k^5$ computations for the denominator. To take into account conditioning, we introduce a new potential function $\delta(x_i, x_i')$ which is defined as follows:

$$\delta(x_i, x_i') = \begin{cases} 1 & \text{if } x_i = x_i' \\ 0 & \text{otherwise} \end{cases}$$

and simply write the conditional as

$$p(x_1 \mid x_5) = \frac{p(x_1, x_5)}{p(x_5)} = \sum_{x_2,x_3,x_4,x_5'} p(x_5'|x_3)\delta(x_5, x_5')p(x_3|x_1, x_2)p(x_4|x_2)p(x_2)$$

That is, whenever a variable is observed, you imagine that you have imposed the indicator function for that observation into the joint distribution. Given this simplicification, we will focus on efficiently computing $p(x_1)$, assuming that the $\delta$ function will be slapped onto the corresponding factor while computing conditionals.

Using the structure of the graphical model (implicit in the topological ordering over the indices) , we can rearrange the sums and products for $p(x_1|x_5)$

$$p(x_1 \mid x_5) = \left(\sum_{x_2} p(x_2)\left(\sum_{x_3} p(x_3|x_1, x_2)\left(\sum_{x_4} p(x_4|x_2)\right)\left(\sum_{x_5'} p(x_5'|x_3)\delta(x_5, x_5')\right)\right)\right) \tag{7.14}$$

where brackets have been placed at appropriate places to denote domains of summation.

Analysing this computational structure inside-out,

1. We find two innermost factors to be common across all the summations, *viz.*,

$$m_{x_4}(x_2) = \sum_{x_4} p(x_4|x_2)$$

$$m_{x_5}(x_3) = \sum_{x_5'} p(x_5'|x_3)\delta(x_5, x_5')$$

where $m_{x_4}$ is a *message* function of $x_2$ and is obtained using $k \times k = k^2$ computations and $m_{x_5}$ is a *message* function of $x_3$ and similarly obtained using $k^2$ computations.

2. Further, we can decipher from (7.14), the following message function $m_{x_3}$ of $x_1$ and $x_2$ which can be computed using $k^3$ operations:
$$m_{x_3}(x_1, x_2) = \sum_{x_3} p(x_3|x_1, x_2) m_{x_4}(x_2) m_{x'_5}(x_3)$$
3. Putting all the messages together, we get the message function $m_{x_2}$ of $x_1$, computable with $k^2$ operations.
$$m_{x_2}(x_1) = \sum_{x_2} m_{x_3}(x_1, x_2)$$

Thus, the summation over the factorization can be expressed as a flow of message from one node to another; when the message from a node passes on to another, the former gets stripped or *eliminated*. In the step (1), nodes $x_4$ and $x_5$ got stripped. In step (2), node $x_3$ was stripped and finally, in step (3), $x_2$ got stripped. This yields an elimination ordering[6] $[x_4, x_5, x_3, x_2]$. The order of computation is thus brought down from $O(k^5)$ to $O(max(k^2, k^3)) = O(k^3)$ computations. While some researchers could argue that this may not be a substantial decrease, for larger graphs the gain in speed using this procedure is always substantial.

More formally, consider a root to leaf ordering $\mathcal{I}$ of the nodes, where $r$ is the root node (equivalently, an ordering that corresponds to leaf stripping). Figure 7.1 shows a numbering of the nodes corresponding to such an ordering. We will define as the current active set $\mathcal{A}(k)$, a set of indices of general potential functions. At each step of the algorithm the potential functions are of three different types: (i) some of the local conditional probabilities $p(x_i|\mathbf{x}_{\pi_i})$, (ii) some of the indicators $\delta(x_j, x'_j)$ of the observed nodes and (iii) some messages (*c.f.* page 400) generated so far. More formally, the active set of potentials is given by $\{\Psi_\alpha(\mathbf{x}_\alpha)\}_{\alpha \in \mathcal{A}^{(k)}}$, with $\alpha$ being a generic index that ranges over sets of nodes. $\mathcal{A}^{(0)}$ is initialized as the set of all cliques that are associated with potential functions in the graphical model. For example, in Figure 7.1, $\mathcal{A}^{(0)} = \{\{1\}, \{2\}, \{3, 2, 1\}, \{2, 4\}, \{3, 5\}\}$. Then, $\mathcal{A}^{(k)}$ can be computed using the algorithm presented in Figure 7.6.

When the active set construction is complete, the desired conditional/marginal probability can be obtained as
$$p(x_r \mid \mathbf{x}_o) = \frac{\Psi_{\{r\}} x_r}{\sum\limits_{x_r} \Psi_{\{r\}} x_r}$$

A flip side of the elimination algorithm is that it requires a 'good' elimination order to be first determined. The number of elimination orderings is obviously a large value of $(n-1)!$, where $n$ is the number of nodes. Finding a good elimination ordering is an NP hard problem and heuristics have been the only recourse. We will not discuss the elimination algorithm any further, but instead jump to the more efficient sum-product algorithm.

[6]Since either $x_4$ and $x_5$ may get stripped first, $[x_5, x_4, x_3, x_2]$ is also a valid elimination ordering.

```
1. Construct an elimination ordering of the nodes so that the target node
at which condition/marginal is desired, is last in the ordering.
2. Initialize $\mathcal{A}^{(0)}$ to the set of all cliques on which potentials are defined.
Set $k = 0$.
for Each $i \in \mathcal{I}$ do
    4. Compute $\Psi_{\beta_i}(\mathbf{x}_{\beta_i}) = \prod_{\{\alpha \in \mathcal{A}^{(k)} | i \in \alpha\}} \Psi_\alpha(\mathbf{x}_\alpha)$ where, $\beta_i = \{i\} \cup$
    $\left\{j \mid \exists \alpha \in \mathcal{A}^{(k)}, \{i,j\} \subset \alpha\right\}$. That is, $\Psi_{\beta_i}(\mathbf{x}_{\beta_i})$ is a product of potentials
    that have shared factors with $\{i\}$.
    5. Message Computation: Compute the message communicated to
    $x_i$ by stripping out $x_i$ through summing over $x_i$. $M_i(\mathbf{x}_{\gamma_i}) = \Psi_{\gamma_i}(\mathbf{x}_{\gamma_i}) =$
    $\sum_{x_i} \Psi_{\beta_i}(\mathbf{x}_{\beta_i})$, where, $\gamma_i = \beta_i \setminus \{i\}$, that is, $\gamma_i$ is the resudial left after strip-
    ping out $\{i\}$ from $\beta_i$ and the message $M_i$ depends only on this residual.
    The computational complexity is determined by the size of the residuals.
    6. Stripping out factors: Remove all $\alpha$ such that $i \in \alpha$ and add $\gamma_i$ to
    the current active set to obtain $\mathcal{A}^{(k+1)}$.
        $\mathcal{A}^{(k+1)} = \mathcal{A}^{(k)} \setminus \{\alpha \in \mathcal{A}^{(k)} \mid i \in \alpha\} \cup \{\gamma_i\}$
end for
```

Figure 7.6: Procedure for constructing the active set, and the message at the desired target node $t$.

### 7.2.2 Sum-product Algorithm

The sum-product algorithm builds on the idea of 'messages' as motivated by the elimination algorithm. It involves local computations at nodes, to generate 'messages' which are related by nodes along their edges to their neighbors. This formalism enables simultaneous computation of marginals, conditionals as well as modes for several variable sets. This algorithm generalizes special algorithms such as viterbi, the forward-backward algorithm, kalman filtering, gaussian elimination as well as the fast fourier transform.

We will initially restrict our attention to undirected trees and will later generalize the algorithm. Figure 7.7 shows an example tree structured graphical model. Since the cliques consist of edges and individual nodes, the potential functions are basically either node potentials $\phi_p(x_p)$ or edge potentials $\phi_{p,q}(x_p, x_q)$. The joint distribution for the tree is

$$p(\mathbf{x}) = \frac{1}{Z} \prod_{p \in \mathcal{V}} \phi_p(x_p) \prod_{(p,q) \in \mathcal{E}} \phi_{p,q}(x_p, x_q) \tag{7.15}$$

Note that, all discussions that follow, hold equally well for directed trees, with the special parametrization $\phi_{p,q}(x_p, x_q) = p(x_p|x_q)$ if $x_p \in \pi_{x_q}$ and $\phi_p(x_p) =$

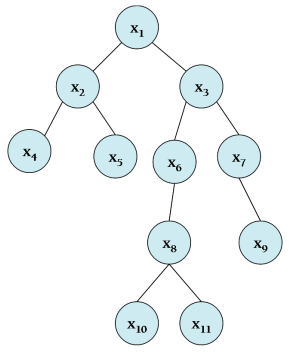


Figure 7.7: A tree graphical model.

$p(x_p)$.

We will deal with conditioning in a manner similar to the way we dealt with conditioning in the case of directed graphical models. For every observed variable $X_o = x_o$, we impose the indicator function $\delta(x_o, x'_o)$ onto $\phi_o(x'_o)$. Then, for a set of observed variables $\mathbf{X}_{\mathcal{O}}$, the conditional then takes the form:

$$p(\mathbf{x} \mid \mathbf{x}_{\mathcal{O}}) = \frac{1}{Z_{\mathcal{O}}} \prod_{p \in \mathcal{V} \setminus \mathcal{O}} \phi_p(x_p) \prod_{o \in \mathcal{O}} \phi_o(x'_o)\delta(x_o, x'_o) \prod_{(p,q) \in \mathcal{E}} \phi_{p,q}(x_p, x_q) \tag{7.16}$$

Thus, modifying the compatibility functions appropriately reduces the conditional problem to an instance of the base problem.

The crux of the sum-product algorithm is the following observation on the application of the leaf stripping and message construction procedure in Figure 7.6 to a tree-structured graphical model.

**Theorem 90** *When a node $i$ is eliminated, $\gamma_i = \{p\}$, where $p$ is the unique parent of node $i$. This means, we can write the message as $M_{i \to p}$.*

*Proof Sketch:* This can be proved by induction on the number of steps in the leaf-stripping (elimination) order. You will need to show that at every step, $\beta_i = \{i, p\}$, where $p$ is the unique parent of $i$. Consequently, $\gamma_i = \{p\}$ and we can derive the (recursive) expression for $M_{i \to p}(x_p)$ as

$$M_{i \to p}(x_p) = \Psi_{x_p}(x_p) = \underbrace{\sum_{x_i} \phi_i(x_i)\phi_{i,p}(x_i, x_p)}_{SUM} \underbrace{\prod_{q \in \mathcal{N}(i) \setminus p} M_{q \to i}(x_i)}_{PRODUCT} \tag{7.17}$$

Note that the proof of this statement will again involve induction. □

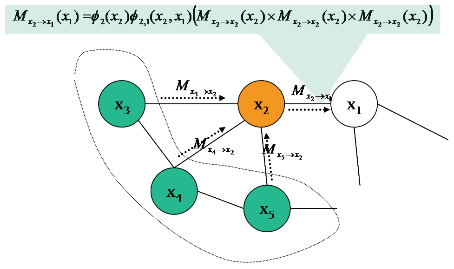

$M_{x_2\to x_1}(x_1) = \phi_2(x_2)\phi_{2,1}(x_2,x_1)\left(M_{x_2\to x_2}(x_2)\times M_{x_2\to x_2}(x_2)\times M_{x_2\to x_2}(x_2)\right)$
$M_{x_3\to x_2}$
$M_{x_2\to x_1}$
$M_{x_4\to x_2}$
$M_{x_5\to x_2}$
$x_3$
$x_2$
$x_1$
$x_4$
$x_5$

Figure 7.8: An illustration of (7.17) on a part of a tree structured (undirected) graphical model.

Figure 7.8 illustrates an application of (7.17) on a part of a tree structured graphical model.

Theorem 90 provides a very useful observation; the message is not a function of the vector of all nodes eliminated so far (which could have assumed $k^m$ possible values for $m$ eliminated nodes having $k$ possible values each), but is instead is a function only of the child from which the message is being passed. This allows us to move from the elimination algorithm to the sum-product algorithm. In particular, the parts underlined as 'SUM' and 'PRODUCT' in (7.17) form the basis of the sum-product algorithm. In the sum-product algorithm, at every time step, the computation in (7.17) is performed at every node; each node does local computations and passes 'updated' messages to each of its neighbors. In this way, the computations are not tied to any particular elimination ordering.

**Theorem 91** *Consider an undirected tree structured graphical model $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with factorization given by (7.15). Let each random variable $X_i$, $i \in \mathcal{V}$ assume $k$ possible values from $\mathcal{X} = \{\alpha_1, \alpha_2, \dots, \alpha_k\}$. For each edge $(u, v)$, define non-negative messages along both directions: $M_{v\to u}(\alpha_i)$ along $v \to u$ and $M_{u\to v}(\alpha_i)$ along $u \to v$ for each $\alpha_i \in \mathcal{X}$. If $r$ is the iteration number, then the update rule*

$$M^{(r+1)}_{u\to v}(x_v) = \frac{1}{Z^{(r)}_{u\to v}} \underbrace{\sum_{x_u} \phi_u(x_u)\phi_{u,v}(x_u, x_v) \prod_{q\in\mathcal{N}(u)\setminus v} M^{(r)}_{q\to u}(x_u)}_{\widetilde{M^{(r+1)}_{u\to v}}(x_v)}$$

*can be executed parallely at every node $u$ along every edge $u, v$ originating at $u$ and will converge, resulting in a fix-point for each $M^{(r^*)}_{u\to v}(x_v)$ for some $r^*$. That is, there exists an $r^*$ such that $M^{(r^*+1)}_{u\to v}(x_v) = M^{(r^*)}_{u\to v}(x_v)$ for all $(u, v) \in \mathcal{E}$. At convergence,*

$$p(x_v) = \phi(x_v) \prod_{u\in\mathcal{N}(v)} M_{u\to v}(x_v) \tag{7.18}$$

*Note that $Z^{(r)}_{u\to v}$ is the normalization factor (required for ensuring numerical stability across iterations but not otherwise) and can be computed at each iteration as*

$$Z^{(r)}_{u\to v} = \sum_{x_v} \widetilde{M^{(r+1)}_{u\to v}}(x_v)$$

Theorem 91 does away with the need for any elimination ordering on the nodes and lets computations at different nodes happen in parallel. It leads to the sum-product algorithm[7] for trees, which is summarized in Figure 7.9. The so-called flooding[8] schedule does a 'for' loop at each iteration of the algorithm, for each node in $\mathcal{V}$. By Theorem 91, the procedure will converge. In fact, it can be proved that the algorithm will converge after at most $\kappa$ iterations, where $\kappa$ is the diameter (length of the longest path) of $\mathcal{G}$. The intuition is that message passing needs the message from every node to reach every other node and this will take $\kappa$ iterations for that to happen in the sum-product algorithm.

```
Initialize $M^{(0)}_{u\to v}(x_v)$ for each $(u,v) \in \mathcal{E}$ to some strictly positive random values.
Set $r = 0$.
repeat
  for Each $u \in \mathcal{V}$ do
    for Each $v \in \mathcal{N}(u)$ do
      $M^{(r+1)}_{u\to v}(x_v) = \frac{1}{Z^{(r)}_{u\to v}} \sum_{x_u} \phi_u(x_u)\phi_{u,v}(x_u,x_v) \prod_{q\in\mathcal{N}(u)\setminus v} M^{(r)}_{q\to u}(x_u)$.
    end for
  end for
  Set $r = r + 1$.
until $M^{(r)}_{u\to v}(x_v) = M^{(r-1)}_{u\to v}(x_v)$ for each $(u,v) \in \mathcal{E}$ and each $x_v \in \mathcal{X}$
Set $r^* = r - 1$.
```

Figure 7.9: The sum-product algorithm for a tree, using the flooding schedule. It converges for $r^* \leq \kappa$, $\kappa$ being the tree diameter.

There have been modern, interesting uses of message passing techniques on graphs with cycles, such as in the field of sensor networks, where locally parallelizable algorithms such as message passing are highly desirable. While there is nothing that stops us in principle from applying the algorithm in Figure 7.9 to general graphs having cycles, the theory does not guarantee anything at all - neither in terms of convergence nor in terms of the number of iterations. However, in solving partial differential equations, the message passing algorithm is often used. The algorithm is widely used on certain interesting cyclic graphs in the

[7]SIMPLE EXERCISE : Implement the algorithm using Matlab.

[8]The flooding schedule somewhat mimics the flow of water or some liquid through a network; water flows in to a node along an edge and then spreads through the other edges incident on the node.

field of communications. For special problems such as solving linear systems, this method converges[9] on graphs with cycles as well.

However, a schedule such as in Figure 7.9 will typically incur a heavy communication cost, owing to message transmissions. While the flooding schedule is conceptually easy in terms of parallelization, an alternative schedule called the serial schedule is often preferred when parallelization is not of paramount importance. The serial schedule minimizes the number of messages passed. In this schedule, a node $u$ transmits message to node $v$ only when it has received messages from all other nodes $q \in \mathcal{N}(u) \setminus v$. This algorithm will pass a message only once along every direction of each edge (although during different steps). Thus, the scheduling begins with each leaf passing a message to its immediate neighbor. An overhead involved in the serial schedule is that every node needs to keep track of the edges along which it has received messages thus far. For chip level design, the highly parallelizable flooding schedule is always preferred over the serial schedule.

A point to note is that while the algorithm in Figure 7.9 is guaranteed to converge within $\kappa$ steps, in practice, you might want to run the algorithm for fewer steps, until the messages reasonably converge. This strategy is especially adopted in the belief propagation algorithm, which consists of the following steps at each node of a general graphical model, until some convergence criterion is met:

1. form product of incoming messages and local evidence
2. marginalize to give outgoing message
3. propagate one message in each direction across every link

The belief propagation algorithm will be discussed later.

### 7.2.3 Max Product Algorithm

The max product algorithm solves the problem of determining the mode or peak of a conditional distribution, specified by a graphical model $\mathcal{G}$, first addressed on page 399.

$$\widehat{\mathbf{x}}_{\mathcal{A}} = \underset{\mathbf{x}_{\mathcal{A}}}{\operatorname{argmax}}\; p(\mathbf{x}_{\mathcal{A}} \mid \mathbf{x}_{\mathcal{O}})$$

where $\mathbf{X}_{\mathcal{O}}$ is the set of observed variables, having observed values $\mathbf{x}_{\mathcal{O}}$, and $\mathbf{X}_{\mathcal{A}}$ are the query variables.

Before looking at the procedure for finding the model of a distribution, we will take a peek at an algorithm for determining the maximum value of $p(\mathbf{x})$, assuming it to be a distribution over an undirected tree $\mathcal{G}$. This algorithm is closely related to the sum product algorithm and can easily be obtained from the sum-product algorithm by replacing the 'SUM' with a 'MAX' in (7.17). This is because, maximums can be pushed inside products and computations can be

[9]For solving linear systems, the message passing algorithm is more efficient than Jacobi, though less efficient than conjugate gradient.

structured in a similar manner as in the sum product algorithm. This places max-product in the league of message passing algorithms.

The sum-product and max-product algorithms are formally very similar. Both methods are based on the distributive law[10]:

- For sum-product: $ab + ac = a(b + c)$.
- For max-product (whenever $a \geq 0$): $\max\,\{ab, ac\} = a \times \max\,\{b, c\}$.

```
Initialize $M^{(0)}_{u\to v}(x_v)$ for each $(u,v) \in \mathcal{E}$ to some strictly positive random values.
Set $r = 0$.
**repeat**
  **for** Each $u \in \mathcal{V}$ **do**
    **for** Each $v \in \mathcal{N}(u)$ **do**
      $M^{(r+1)}_{u\to v}(x_v) = \frac{1}{Z^{(r)}_{u\to v}} \max_{x_u} \phi_u(x_u)\phi_{u,v}(x_u, x_v) \prod_{q\in\mathcal{N}(u)\setminus v} M^{(r)}_{q\to u}(x_u)$.
    **end for**
  **end for**
  Set $r = r + 1$.
**until** $M^{(r)}_{u\to v}(x_v) = M^{(r-1)}_{u\to v}(x_v)$ for each $(u,v) \in \mathcal{E}$ and each $x_v \in \mathcal{X}$
Set $r^* = r - 1$.
```

Figure 7.10: The max-product algorithm for a tree, using the flooding schedule. It converges for $r^* \leq \kappa$, $\kappa$ being the tree diameter.

The max-product and sum-product algorithms can be implemented in a common framework, with a simple flag to toggle between the two. The convergence of the max-product algorithm in Figure 7.10 is very similar to the proof of the convergence of the sum-product algorithm in Figure 7.9 (the proof is rooted in Theorem 91). After convergence of the max-product algorithm in Figure 7.10, the maxium value of $\Pr(\mathbf{X} = \mathbf{x})$ can be retrieved as

$$\max_{\mathbf{x}}\ \Pr(\mathbf{X} = \mathbf{x}) = \max_{x_r}\ \phi(x_r) \prod_{u\in\mathcal{N}(r)} M_{u\to r}(x_r) \tag{7.19}$$

where, we assume that $x_r$ is the root of the tree. However, we need to go beyond the maximum value of a distribution; we need to find the point at which the maximum is attained. To traceback this, and in general to compute $\underset{\mathbf{x}_{\mathcal{A}}}{\text{argmax}}\ p(\mathbf{x}_{\mathcal{A}} \mid \mathbf{x}_{\mathcal{O}})$ we will introduce some additional machinery.

As before (*c.f.* page 403), the conditioning can be obtained as in (7.16).

[10]See commutative semirings for the general algebriac framework. Also refer to work on generalized distributive laws.

**Definition 50** *We define the singleton marginal of a distribution $p(\mathbf{x})$ as*

$$\mu_s(x_s) = \frac{1}{\alpha} \max_{\mathbf{x} \setminus \{x_s\}} p(\mathbf{x}) \tag{7.20}$$

*and the pairwise marginal as*

$$\nu_s(x_s, x_t) = \frac{1}{\alpha} \max_{\mathbf{x} \setminus \{x_s, x_t\}} p(\mathbf{x}) \tag{7.21}$$

*where*

$$\alpha = \max_{\mathbf{x}} p(\mathbf{x})$$

The max marginals are analogs of marginilization, but with the summation replaced by the max operator over all variables, except $x_s$ in the former or $(x_s, x_t)$ in the latter. While $\mu_s(x_s)$ gives a vector of max-marginals for the variable $X_s$, $\nu_s(x_s, x_t)$ corresponds to a matrix of values, for the pair of variables $(X_s, X_t)$. You can easily convince yourself that the maximum value of any max-marginal is 1 and it will be attained for atleast one value $x_s$ for each variable $X_s$.

How can the marginals be tracedback efficiently? And how can they be useful? The marginals encode preferences in the form of sufficient statistics. For example:

$$\mu_1(x_1) = \frac{1}{\alpha} \max_{x_2, x_3, \ldots, x_n} p(\mathbf{x})$$

In fact, we can look at the local maximal configurations, when they are unique and traceback the (unique) global maximal configuration. This is stated in the following powerful theorem.

**Theorem 92** *If* $\underset{x_s}{\operatorname{argmax}}\ \mu_s(x_s) = \{x_s^*\}\ \forall s \in \mathcal{V}$ *(that is, there is a unique value of variable $X_s$ that maximizes $\mu_s(x_s)$ for every $s \in \mathcal{V}$), then* $\mathbf{x}^* = \{x_1^*, x_2^*, \ldots, x_n^*\} = \underset{\mathbf{x}}{\operatorname{argmax}}\ p(\mathbf{x})$ *is the unique MAP configuration.*

*Proof Sketch:* The theorem can be equivalently stated as follows: if $\mu_i(x_i^*) > \mu_i(x_i)$ for all $x_i \neq x_i^*$, and for all $i \in \mathcal{V}$, then $p(\mathbf{x}^*) \geq p(\mathbf{x})$ for all $\mathbf{x} \neq \mathbf{x}^*$. This can be proved by contradiction as follows. Suppose $\mathbf{x}' \in \underset{\mathbf{x}}{\operatorname{argmax}}\ p(\mathbf{x})$. Then, for any $s \in \mathcal{V}$,

$$\mu_s(x_s') = \max_{\mathbf{x}} p(\mathbf{x}) = \max_{x_s} \max_{\mathbf{x} \setminus x_s} p(\mathbf{x}) > \mu_s(x_s)\ \forall\ x_s$$

But by definition

$$\max_{x_s} \max_{\mathbf{x} \setminus x_s} p(\mathbf{x}) = \max_{x_s} \mu_s(x_s) = \{x_s^*\}$$

Thus, $x_s' = x_s^*$. Since $s \in \mathcal{V}$ was arbitrary, we must have $\mathbf{x}' = \mathbf{x}^*$. $\square$

The singleton max marginal $\mu_s(x_s)$ can be directly obtained as an outcome of the max-product algorithm:

$$\mu_s(x_s) \propto \phi_s(x_s) \prod_{u \in \mathcal{N}(s)} M_{u \to s}(x_s) \tag{7.22}$$

What if $\{x_s^1, x_s^2\} \subseteq \mu_s(x_s)$? That is, if $\mu_s(x_s)$ violates the assumption in theorem 92? Then we have to start worrying about what is happening on the edges, through the medium of the max marginal $\nu_{s,t}(x_s, x_t)$. All we do is randomly sample from the set of configurations that have maximum probability, without caring which one we really get. We first randomly sample from $\underset{x_r}{\text{argmax}}\ \mu_r(x_r)$ at the root $r$ of the tree and then keep randomly sample for a configuration for a child $s$, given its parent $t$ (*i.e.* for an edge) $\underset{x_s}{\text{argmax}}\ \nu_{st}(x_s, x_t)$ which respects the pairwise coupling. The following theorem states the general traceback mechanism.

**Theorem 93** *Given a set of singleton max-marginals, $\{\mu_s \mid s \in \mathcal{V}\}$ and a set of pairwise marginals: $\{\nu_{st} \mid (s,t) \in \mathcal{E}\}$ on a tree $\mathcal{G} =< \mathcal{V}, \mathcal{E} >$, $\mathbf{x}^* = (x_1^*, \dots, x_n^*)$, constructed using the following procedure is a maximal configuration, that is $\mathbf{x}^* \in \underset{\mathbf{x}}{\text{argmax}}\ p(\mathbf{x})$.*

1. *Let $r$ be the root. Let $x_r^* \in \underset{x_r}{\text{argmax}}\ \mu_r(x_r)$.*
2. *In root to leaf order, choose $x_s^* \in \underset{x_s}{\text{argmax}}\ \nu_{st}(x_s, x_t^*)$*

*Proof:* We will first prove that $\mathbf{x}^*$ is optimal for the root term. For any arbitrary $\mathbf{x}$, by step 1,

$$\mu_r(x_r) \leq \mu_r(x_r^*) \tag{7.23}$$

If $t \to s$ is an edge, then we have by definition of the singleton max-marginal

$$\underset{x_s}{\text{argmax}}\ \nu_{st}(x_s, x_t) = \mu_t(x_t)$$

Thus,

$$\frac{\nu_{st}(x_s, x_t)}{\mu_t(x_t)} \leq 1$$

Since $x_s^* \in \underset{x_s}{\text{argmax}}\ \nu_{st}(x_s, x_t^*)$ by step 2, we must have $\nu_{st}(x_s^*, x_t^*) = \mu_t(x_t^*)$ and therefore, the following upperbound

$$\frac{\nu_{st}(x_s, x_t)}{\mu_t(x_t)} \leq \frac{\nu_{st}(x_s^*, x_t^*)}{\mu_t(x_t^*)} \tag{7.24}$$

for all edges $t \rightarrow s$. With repeated application of step 2, we can stitch together the local optimality conditions (7.23) and (7.24) to get

$$\mu_r(x_r) \prod_{t \rightarrow s} \frac{\nu_{st}(x_s, x_t)}{\mu_t(x_t)} \leq \mu_r(x_r^*) \prod_{t \rightarrow s} \frac{\nu_{st}(x_s^*, x_t^*)}{\mu_t(x_t^*)} \tag{7.25}$$

Just as the singleton max marginals (7.22) can be expressed in terms of the messages from the max product algorithm, the edge max-marginals can also be expressed similarly, by restricting attention to the pair of variables $(x_s, X_t)$ instead of the world of singletons $X_s$, and by avoiding accounting for the message $M_{s \rightarrow t}$ or $M_{t \rightarrow s}$ between the pair:

$$\nu_{st}(x_s, x_t) \propto \phi_s(x_s)\phi_t(x_t)\phi_{st}(x_s, x_t) \prod_{u \in \mathcal{N}(s) \setminus t} M_{u \rightarrow s}(x_s) \prod_{u \in \mathcal{N}(t) \setminus s} M_{u \rightarrow t}(x_t) \tag{7.26}$$

Combining (7.22) with (7.26), we get

$$\frac{\nu_{st}(x_s, x_t)}{\mu_s(x_s)\mu_t(x_t)} \propto \frac{\phi_{st}(x_s, x_t)}{M_{t \rightarrow s}(x_s) M_{s \rightarrow t}(x_t)} \tag{7.27}$$

Applying (7.27) and (7.25) in the factorization for $p(\mathbf{x})$, we get[11], we obtain

$$p(\mathbf{x}) \leq p(\mathbf{x}^*)$$

Since $\mathbf{x}$ was arbitrary, we must have $\mathbf{x}^* \in \underset{\mathbf{x}}{\text{argmax}}\ p(\mathbf{x})$. $\square$

An outcome of the proof above is that for trees, the factorization can be written in terms of max-marginals instead of the potential functions:

$$p(\mathbf{x}) \propto \mu_r(x_r) \prod_{t \rightarrow s} \frac{\nu_t s(x_s, x_t)}{\mu_t(x_t)}$$

The above form is a directed form of factorization and does not hold for general graphs that may contain cycles. For general graphs, it can be further proved that the following factorization holds

$$p(\mathbf{x}) \propto \prod_{s \in \mathcal{E}} \mu_s(x_s) \prod_{(s,t) \in \mathcal{E}} \frac{\nu_{ts}(x_s, x_t)}{\mu_s(x_s)\mu_t(x_t)}$$

[11]EXERCISE: Prove.

### 7.2.4 Junction Tree Algorithm

In many applications, the graphs are not trees. We have not discussed any pricipled techniques for finding marginals and modes for graphs that are not trees, though we have discussed them for trees. The elimination algorithm discussed earlier is applicable for some general graphs, but the question of what elimination should be chosen, needs to be addressed. The junction tree algorithm is very much related to the elimination algorithm, but is a more principled approach for inferencing in directed acyclic graphs. Like the sum-product algorithm, the junction tree algorithm can 'recycle' computations. This is unlike the general elimination algorithm, whose computation was focused completely on a single node. The work on junction trees is primarily attributed to Lauritzen and Spielgelhalter (1998). The correspondence between the graph-theoretic aspect of locality and the algorithmic aspect of computational complexity is made explicit in the junction tree framework.

For a graph with nodes, it is an intuitive idea to consider clustering completely connected nodes, form a tree connecting these clusters and finally perform message passing the tree. This idea can be formalized using the concept of a *clique tree*.

**Definition 51** *Given a graph $\mathcal{G} =< \mathcal{V}, \mathcal{E} >$ with a set $\Pi \subseteq 2^{\mathcal{V}}$ of maximal cliques, a clique tree $\mathcal{T}_{\mathcal{G}}$ is a tree, whose vertices correspond the maximal cliques $\Pi$ of $\mathcal{G}$ and such that there is an edge between two nodes in the tree only if*[12] *there is an edge in $\mathcal{G}$ between two nodes across the corresponding maximal cliques.*

Let us take some examples:

- For the acylcic graph $\mathcal{G}$ with $\mathcal{V} = \{1,2,3,4,5,6\}$, $\mathcal{E} = \{(1,2),(1,3),(2,4),(2,5),(3,6)\}$, the (not so interesting) clique tree $\mathcal{T}_{\mathcal{G}} =< \Pi, \mathcal{E}_\Pi >$ would be $\Pi = \{[1,2],[1,3],[2,4],[2,5],[3,6]\}$ and $\mathcal{E}_\Pi = \{([1,2],[2,4]),([1,2],[2,5]),([1,2],[1,3]),([1,3],[3,6])\}$.

- For the cyclic graph $\mathcal{G}$ with $\mathcal{V} = \{1,2,3,4\}$, $\mathcal{E} = \{(1,2),(1,3),(2,4),(3,4),(2,3)\}$, the clique tree $\mathcal{T}_{\mathcal{G}} =< \Pi, \mathcal{E}_\Pi >$ would have $\Pi = \{[1,2,3],[2,3,4]\}$ and $\mathcal{E}_\Pi = \{([1,2,3],[2,3,4])\}$. We will adopt the practice of labeling the edge connecting nodes corresponding to two maximal cliques $\mathcal{C}_1$ and $\mathcal{C}_2$, with their intersection $\mathcal{C}_1 \cap \mathcal{C}_2$, which will be called the *separator set*. In the second example here, the separator set for vertices $[1,2,3]$ and $[2,3,4]$ is $[2,3]$.

- For the slightly different cyclic graph $\mathcal{G}$ with $\mathcal{V} = \{1,2,3,4\}$, $\mathcal{E} = \{(1,2),(1,3),(2,4),(3,4)\}$, there are many possible clique trees. One possible clique tree $\mathcal{T}_{\mathcal{G}} =< \Pi, \mathcal{E}_\Pi >$ would have $\Pi = \{[1,2],[1,3],[2,4],[3,4]\}$ and $\mathcal{E}_\Pi = \{([1,2],[1,3]),([1,2],[2,4]),([1,3],[3,4])\}$. It is a bit worrisome here that $[2,4]$ and $[3,4]$ are not connected, since a myopic or 'local' sum-product algorithm, running on the clique tree might make a wrong inference that $[2,4]$ and $[3,4]$ do not share anything in common. In this example, for instance, the message coming from $[2,4]$,

[12]Since we are interested in a clique 'tree', it may be required to drop certain edges in the derived graph. This can be seen through the third example.

through $[1,2]$ to $[3,4]$ has marginalized over $X_4$. But this is incorrect, since $[3,4]$ include the variable $X_4$.

With the last example in mind, we will refine the notion of a clique tree to a *junction tree*, in order to ensure that local computations are guaranteed to produce globally consistent answers; **that different copies of the same random variable have ways of communicating with each other**.

**Definition 52** *A junction tree for a graph $\mathcal{G} =< \mathcal{V}, \mathcal{E} >$ having a set $\Pi \subseteq 2^{\mathcal{V}}$ of maximal cliques, is a particular type of clique tree $\mathcal{T}_{\mathcal{G}}$ such that for any two $\mathcal{C}_1, \mathcal{C}_2 \in \Pi$, $\mathcal{C}_1 \cap \mathcal{C}_2$ is a subset of every separator set on the unique path from $\mathcal{C}_1$ to $\mathcal{C}_2$ in $\mathcal{T}_{\mathcal{G}}$. This property of junction trees is called the running intersection property.*

Based on this definition, the clique trees for the first two examples are junction trees, whereas that for the third is not a junction tree. In fact, there are no junction tree for the third example. Let us consider another example.

- Consider the cyclic graph $\mathcal{G}$ with $\mathcal{V} = \{1,2,3,4,5\}$, $\mathcal{E} = \{(1,2),(1,3),(2,3),(2,4),(3,4),(3,5)\}$. There are many possible clique trees for $\mathcal{G}$. One possible clique tree $\mathcal{T}_{\mathcal{G}} =< \Pi, \mathcal{E}_{\Pi} >$ has $\Pi = \{[1,2,3],[2,3,4],[3,4,5]\}$ and $\mathcal{E}_{\Pi} = \{([1,2,3],[2,3,4]),([2,3,4],[3,4,5])\}$ and this happens to also be a junction tree. Another clique tree with same vertex set $\Pi$ and $\mathcal{E}'_{\Pi} = \{([1,2,3],[3,4,5]),([2,3,4],[3,4,5])\}$ is not a junction tree, since node 2 which is in the intersection of $[1,2,3]$ and $[2,3,4]$ is not on every separator on the path between these two nodes. This illustrates that how you generate your clique tree matters; some clique trees may happen to be junction trees, while some may not.

- For the cyclic graph $\mathcal{G}$ with $\mathcal{V} = \{1,2,3,4\}$, $\mathcal{E} = \{(1,2),(1,3),(2,4),(3,4)\}$, considered in the third example above, there are no junction trees possible.

The global picture on a graph is specified by the factorization property:

$$p(\mathbf{x}) \propto \prod_{\mathcal{C} \in \Pi} \phi_{\mathcal{C}}(\mathbf{x}_{\mathcal{C}}) \tag{7.28}$$

On the other hand, the local message passing algorithm should honor constraints set by the separator set; that the configuration of variables in the separator set are the same in both its neighbors. More precisely, if we define the set $\widetilde{\mathbf{x}}_{\mathcal{C}} = \{\widetilde{x}_{i,\mathcal{C}} | i \in \mathcal{C}\}$ for each $\mathcal{C} \in \Pi$, then, for each separator set $\mathcal{C}_1 \cap \mathcal{C}_2$, the separator sets define the constraints in the form

$$\prod_{i \in \mathcal{C}_1 \mathcal{C}_2} \delta\left(\widetilde{x}_{i,\mathcal{C}_1} = \widetilde{x}_{i,\mathcal{C}_2}\right) \tag{7.29}$$

The factorization that message passing on $\mathcal{T}_{\mathcal{G}} =< \Pi, \mathcal{C}' >$ should see with the set of additional constraints will involve multiple copies of each random variable, but will be tied together through the $\delta$ constraints in (7.29):

$$\widetilde{p}(\widetilde{x}_{i,\mathcal{C}},\ \forall i \in \mathcal{C},\ \forall \mathcal{C} \in \Pi) \propto \prod_{\mathcal{C}\in\Pi} \phi_{\mathcal{C}}(\widetilde{\mathbf{x}}_{\mathcal{C}}) \prod_{(\mathcal{C}_1,\mathcal{C}_2)\in\mathcal{E}_\Pi} \prod_{i\in\mathcal{C}_1\cap\mathcal{C}_2} \delta\left(\widetilde{x}_{i,\mathcal{C}_1} = \widetilde{x}_{i,\mathcal{C}_2}\right) \quad (7.30)$$

The (localized) sum-product algorithm will work precisely on the junction tree $\mathcal{T}_{\mathcal{G}}$ with factorization as specified in (7.30). The factorization in (7.30) is in fact equivalent to the factorization in (7.28). In (7.30), the multiple copies of $x_i$'s cancel out aginst the constraints $\delta$ constraint. The is called the junction tree property and is formally stated in the next proposition.

**Theorem 94** *For a graph $\mathcal{G}$ having distribution $p(\mathbf{x})$ as in (7.28) and for its junction tree $\mathcal{T}_{\mathcal{G}}$ having distribution $\widetilde{p}$ specified by (7.30), the $\widetilde{p}$ distribution satisfies the following property:*

$$\widetilde{p}(\widetilde{x}_{i,\mathcal{C}},\ \forall i \in \mathcal{C},\ \forall \mathcal{C} \in \Pi) = \begin{cases} p(\mathbf{x}) & \text{if } \{x_{i,\mathcal{C}_1} = x_{i,\mathcal{C}_2} \mid \forall \mathcal{C}_1, \mathcal{C}_2 \in \Pi,\ \forall\ i \in \mathcal{C}_1 \cap \mathcal{C}_2\} \\ 0 & \text{otherwise} \end{cases}$$

*That is, the new distribution $\widetilde{p}$ is faithful to the original distribution. The transitivity due to the running intersection property of junction trees is exactly what you need for this desirable property to hold.*

The proof of this theorem is trivial, given the junction tree assumption. As an exercise, you may verify the truth of this statement for all the junction tree examples considered so far. The message passing formalisms in Figures 7.9 and 7.10, when applied to the junction tree, will land up not accepting contributions from inconsistent configurations, owing to the $\delta$ constraint and will therefore discover the true marginal/mode.

**Theorem 95** *Suppose that $\mathcal{G}$ has a junction tree $\mathcal{T}_{\mathcal{G}} =< \Pi, \mathcal{E}_\Pi >$. Then running the sum-product or max-product algorithm on the distribution $\widetilde{p}$ defined in (7.28) will output the correct marginals or modes respectively, for $p$ defined for $\mathcal{G}$, in (7.30). The $\phi_{\mathcal{C}}(\widetilde{\mathbf{x}}_{\mathcal{C}})$ can be thought of as node potentials for $\mathcal{T}_{\mathcal{G}}$, while $\delta\left(\widetilde{x}_{i,\mathcal{C}_1} = \widetilde{x}_{i,\mathcal{C}_2}\right)$ are the edge potentials.*

Theorem 95 presentes a transformation of the original problem to a problem on a right kind of tree on which the running intersection property holds so that marginals and modes are preserved. The proof of this theorem is also simple. In practice, the message passing algorithm need not create multiple copies of the shared variables; the sharing can be imposed implicitly.

### 7.2.5 Junction Tree Propagation

The *junction tree propagation algorithm* is the sum-product algorithm applied to the junction tree, with factorization specified by (7.30). It is due to Shafer and Shenoy [SS90]. Consider the message update rule from the algorithm in Figure 7.9.

$$M_{u\to v}^{(r+1)}(x_v) = \frac{1}{Z_{u\to v}^{(r)}} \sum_{x_u} \phi_u(x_u)\phi_{u,v}(x_u, x_v) \prod_{q\in\mathcal{N}(u)\backslash v} M_{q\to u}^{(r)}(x_u)$$

If neighbors $u$ and $v$ are replaced by neighboring cliques $\mathcal{C}_1$ and $\mathcal{C}_2$ respectively, the equation becomes

$$M_{\mathcal{C}_1\to\mathcal{C}_2}^{(r+1)}(\mathbf{x}_{\mathcal{C}_2}) = \frac{1}{Z_{\mathcal{C}_1\to\mathcal{C}_2}^{(r)}} \sum_{\mathbf{x}'_{\mathcal{C}_1}} \phi_{\mathcal{C}_1}(\mathbf{x}'_{\mathcal{C}_1}) \underbrace{\prod_{i\in\mathcal{C}_1\cap\mathcal{C}_2} \delta\left(x'_{i,\mathcal{C}_1} = x_{i,\mathcal{C}_2}\right)}_{based\ on\ separator\ set\ \mathcal{C}_1\cap\mathcal{C}_2} \prod_{\mathcal{C}_3\in\mathcal{N}(\mathcal{C}_1)\backslash\mathcal{C}_2} M_{\mathcal{C}_3\to\mathcal{C}_1}^{(r)}(\mathbf{x}'_{\mathcal{C}_1})$$

The constraint based on the separator set ensures that configurations that are not consistent do not contribute to the outermost summation $\sum\limits_{\mathbf{x}'_{\mathcal{C}_1}}$. The expression for the message can therefore be equivalently written as

$$M_{\mathcal{C}_1\to\mathcal{C}_2}^{(r+1)}(\mathbf{x}_{\mathcal{C}_2}) = \frac{1}{Z_{\mathcal{C}_1\to\mathcal{C}_2}^{(r)}} \sum_{\mathbf{x}'_{\mathcal{C}_1\backslash\mathcal{C}_2}} \phi_{\mathcal{C}_1}(\mathbf{x}'_{\mathcal{C}_1\backslash\mathcal{C}_2}, \mathbf{x}_{\mathcal{C}_2}) \prod_{\mathcal{C}_3\in\mathcal{N}(\mathcal{C}_1)\backslash\mathcal{C}_2} M_{\mathcal{C}_3\to\mathcal{C}_1}^{(r)}(\mathbf{x}'_{\mathcal{C}_1}) \tag{7.31}$$

Note that in (7.31), the constraints based on separator sets are implicitly captured in the summation $\sum\limits_{\mathbf{x}'_{\mathcal{C}_1\backslash\mathcal{C}_2}}$ over only a partial set of variables from $\mathbf{x}'_{\mathcal{C}_1}$. Further, the message $M_{\mathcal{C}_1\to\mathcal{C}_2}^{(r+1)}(\mathbf{x}_{\mathcal{C}_2})$ is not a function of the complete vector $\mathbf{x}_{\mathcal{C}_2}$ but is only a function of $\mathbf{x}_{\mathcal{C}_2\cap\mathcal{C}_1}$. Rewriting (7.31) to reflect this finding, we have

$$M_{\mathcal{C}_1\to\mathcal{C}_2}^{(r+1)}(\mathbf{x}_{\mathcal{C}_2\cap\mathcal{C}_1}) = \frac{1}{Z_{\mathcal{C}_1\to\mathcal{C}_2}^{(r)}} \sum_{\mathbf{x}'_{\mathcal{C}_1\backslash\mathcal{C}_2}} \phi_{\mathcal{C}_1}(\mathbf{x}'_{\mathcal{C}_1\backslash\mathcal{C}_2}, \mathbf{x}_{\mathcal{C}_2\cap\mathcal{C}_1}) \prod_{\mathcal{C}_3\in\mathcal{N}(\mathcal{C}_1)\backslash\mathcal{C}_2} M_{\mathcal{C}_3\to\mathcal{C}_1}^{(r)}(\mathbf{x}'_{\mathcal{C}_1\cap\mathcal{C}_3}) \tag{7.32}$$

This finding is important, since it helps reduce the computational complexity of the algorithm. You need to send messages whose sizes do not depend on the size of the cliques themselves but only on the size of the separator sets. Thus, if each variable was multinomial with $k$ possible values, then the message size would be $k^{|\mathcal{C}_1\cap\mathcal{C}_2|}$ instead of $k^{|\mathcal{C}_2|}$. Thus, the complexity of junction tree propagation is exponential in the size of the separator sets. Typically however, the size of seperator sets are not much smaller than the cliques themselves.

The junction tree propagation will converge, by the convergence property of the sum-product algorithm. After convergence, the marginals can be obtained as

$$p(\mathbf{x}_{\mathcal{C}}) = \phi_{\mathcal{C}}(\mathbf{x}_{\mathcal{C}}) \prod_{\mathcal{D} \in \mathcal{N}(\mathcal{C})} M_{\mathcal{D} \to \mathcal{C}}(\mathbf{x}_{\mathcal{C} \cap \mathcal{D}}) \tag{7.33}$$

### 7.2.6 Constructing Junction Trees

How do we obtain a junction tree for a graph. And what classes of graphs have junction trees? We already saw an example of a graph that did not have any junction tree; $\mathcal{G} =< \mathcal{V}, \mathcal{E} >$, $\mathcal{V} = \{1,2,3,4\}$, $\mathcal{E} = \{(1,2),(1,3),(2,4),(3,4)\}$. We also saw an example for which a particular clique tree was not a junction tree, though it had another clique tree that was a junction tree: $\mathcal{G}'$ with $\Pi = \{1,2,3,4,5\}$, $\mathcal{E}_\Pi = \{(1,2),(1,3),(2,3),(2,4),(3,4),(3,5)\}$. The junction tree $< \mathcal{V}_t', \mathcal{E}_t >$ has $\mathcal{V}_t' = \{[1,2,3],[2,3,4],[3,4,5]\}$ and $\mathcal{E}_t' = \{([1,2,3],[2,3,4]),([2,3,4],[3,4,5])\}$, which corresponds to the elimination ordering $[1,3,2,4,5]$. While the clique tree $\mathcal{V}_t'' = \mathcal{V}_t'$ and $\mathcal{E}_t'' = \{([1,2,3],[3,4,5]),([2,3,4],[3,4,5])\}$ is not a junction tree and corresponds to the elimination ordering $[1,2,4,5,3]$. We can see that the construction of the junction tree really depends on the choice of a 'nice' elimination ordering. One difference between $\mathcal{G}$ and $\mathcal{G}'$ is that, while in the former, you cannot eliminate any node without adding additional edges, in the latter, you have an elimination ordering $[1,3,2,4,5]$ that does not need to add extra edges. For $\mathcal{G}'$, the elimination ordering $[1,2,3,4,5]$ will not yield a junction tree.

This leads to the definition of a triangulated graph, one of the key properties of any graph which can be transformed into a junction tree. In fact, a requirement will be that an elimination algorithm is 'good' for junction tree construction only if it leads to a triangulated graph.

**Definition 53** *A cycle is chordless if no two non-adjacent vertices on the cycle are joined by an edge. A graph is triangulated it is has no chordless cycles.*

Thus, the graph $\mathcal{G}$ with $\mathcal{V} = \{1,2,3,4\}$, $\mathcal{E} = \{(1,2),(1,3),(2,4),(3,4)\}$ is not triangulated, whereas, the graph $\mathcal{G}'$ with $\Pi' = \{1,2,3,4,5\}$, $\mathcal{E}_\Pi' = \{(1,2),(1,3),(2,3),(2,4),(3,4),(3,5)\}$ is triagulated. In fact, every triangulated graph has at least one junction tree. Another equivalent characterization of a triangulated graph is as a *decomposable graph.*

**Definition 54** *A graph $\mathcal{G} =< \mathcal{V}, \mathcal{E} >$ is decomposable either if it is complete or if $\mathcal{V}$ can be recursively divided into three disjoint sets $\mathcal{A}$, $\mathcal{B}$ and $\mathcal{S}$ such that*

1. *$\mathcal{S}$ separates $\mathcal{A}$ and $\mathcal{B}$ and*
2. *$\mathcal{S}$ is fully connected (i.e., a clique).*
3. *$\mathcal{A} \cup \mathcal{S}$ and $\mathcal{B} \cup \mathcal{S}$ are also decomposable.*

Following are examples of decomposable graphs:

- $\mathcal{V} = \{1, 2, 3\}$, $\mathcal{E} = \{(1, 2), (2, 3)\}$.
- $\mathcal{V} = \{1, 2, 3, 4\}$, $\mathcal{E} = \{(1, 2), (2, 3), (3, 4), (4, 1), (2, 4)\}$. Application of elimination procedure on this graph, say starting with 3 should lead in 2 and 4 being connected together, which are already connected in this graph. This shows the connection between decomposable graphs and elimination.

However, for $\mathcal{V} = \{1, 2, 3, 4\}$ and $\mathcal{E} = \{(1, 2), (2, 3), (3, 4), (4, 1)\}$, $\mathcal{G}$ is not decomposable.

Recall from Section 7.2.1, the elimination algorithm, which eliminated one node at a time, while connecting its immediate neighbors. Thus, for any undirected graph, by the very construction of the elimination algorithm, it is obvious that the reconstituted graph, output by the elimination algorithm is always triangulated. This can be proved by induction on the number of vertices in the graph[13]. The statement is trivial for the base case of a one node graph.

The following theorem is a *fundamental characterization* of graphs that have junction trees.

**Theorem 96** *The following are equivalent ways of characterizing a graph $\mathcal{G}$:*

1. *$\mathcal{G}$ is decomposable.*
   - *(This captures the 'divide-and-conquer' nature of message passing algorithms. In fact, the message passing algorithm exploited*

   *a divide and conquer strategy for computation on trees.)*
2. *$\mathcal{G}$ is triangulated.*
   - *(Elimination can result in a triangulated graph.)*
3. *$\mathcal{G}$ has a junction tree.*
   - *(If the graph is triangulated, it must have at least one junction tree. And junction tree is a good canonical data structure for conducting computations on general graphs.)*

Some practical impacts of this theorem are listed itemized in brackets by the side of each of the equivalent characterizations of $\mathcal{G}$. The equivalence of the first and second statements in the theorem can be proved very simply by induction on the number of nodes in the graph.

The first step in the junction tree algorithm is triangulating a graph. This might mean adding extra edges or increasing clique size. But this cannot be harmful[14], since the potential function can be defined over a larger clique

[13]Prove: EXERCISE.

[14]What characterizes a graphical models is not the presence of edges, but the absence of edges. As an the extreme example, a completel graph potentially subsumes every graphical model.

as the product of potential functions over its sub-parts. Given a triangulated graph, we know that it must have a junction tree by virtue of theorem 96. How can a junction tree be constructed from a triangulated graph? The first step would be to isolate all its cliques. Going back to the second example on page 412, the triangulated graph $\mathcal{G}$ with $\mathcal{V} = \{1,2,3,4,5\}$, $\mathcal{E} = \{(1,2),(1,3),(2,3),(2,4),(3,4),(3,5)\}$ has three maximal cliques: $\Pi = \{[1,2,3],[2,3,4],[3,4,5]\}$. There are different ways to connect up the cliques to form a tree. One possible clique tree $\mathcal{T}_{\mathcal{G}} =< \Pi, \mathcal{E}_\Pi >$ has $\mathcal{E}_\Pi = \{([1,2,3],[2,3,4]),([2,3,4],[3,4,5])\}$ and this happens to also be a junction tree. Another clique tree has $\mathcal{E}'_\Pi = \{([1,2,3],[3,4,5]),([2,3,4],[3,4,5])\}$ is not a junction tree, since node 2 which is in the intersection of $[1,2,3]$ and $[2,3,4]$ is not on every separator on the path between these two nodes.

While you can discover a junction tree by exhaustive search, that is an infeasible idea.However, the search for a junction tree can be performed efficiently by making use of the following theorem.

**Theorem 97** *Let $\mathcal{G} =< \mathcal{V}, \mathcal{E} >$ be a triangulated graph with $\mathcal{V} = \{X_1, X_2, \ldots, X_n\}$ and with $\Pi$ being the set of maximal cliques. Let $\mathcal{T}_{\mathcal{G}}$ be a spanning tree for the clique graph of $\mathcal{G}$, having $\Pi = \{\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_m\}$ as the set of vertices and $\mathcal{E}_\Pi = \{e_1, e_2, \ldots, e_{m-1}\}$ as the set of edges ($|\Pi| = m$ and $|\mathcal{E}_\Pi| = m-1$). Let $\mathcal{S}(e)$ be the separator associated*[15] *with any edge $e \in \mathcal{E}_\Pi$. We will define the weight*[16] *of the spanning tree as*

$$w(\mathcal{T}_{\mathcal{G}}) \in \sum_{i=1}^{m-1} |\mathcal{S}(e_i)| = \sum_{i=1}^{m-1} \sum_{j=1}^{n} [X_j \in \mathcal{S}(e_i)] \tag{7.34}$$

*where [condition] is the indicator function that assumes value 1 if and only if condition is satisfied and is 0 otherwise. Then, if*

$$\widehat{\mathcal{T}}_{\mathcal{G}} = \operatorname*{argmax}_{\mathcal{T}_{\mathcal{G}}} w(\mathcal{T}_{\mathcal{G}})$$

*$\widehat{\mathcal{T}}_{\mathcal{G}}$ must be a junction tree.*

*Proof:* First we note that for any variable $X_j$, the number of separator sets in $\mathcal{T}_{\mathcal{G}}$ in which $X_j$ appears is upper bounded by the number of cliques in which $X_j$ appears.

$$\sum_{i=1}^{m-1} [X_j \in \mathcal{S}(e_i)] \leq \sum_{i=1}^{m} [X_j \in \mathcal{C}_i] \tag{7.35}$$

[15]Since any edge $e$ in the cliqe graph is of the form $(\mathcal{C}_1, \mathcal{C}_2)$, where $\mathcal{C}_1, \mathcal{C}_2 \in \Pi$, the separator associated with $e$ can be viewed as a function $\mathcal{S}(e) = \mathcal{C}_1 \cap \mathcal{C}_2$.

[16]For the example above with $\mathcal{E}_\Pi = \{([1,2,3],[2,3,4]),([2,3,4],[3,4,5])\}$, the weight is $2 + 2 = 4$, and this also happens to be a junction tree. For $\mathcal{E}'_\Pi = \{([1,2,3],[3,4,5]),([2,3,4],[3,4,5])\}$, the weight is $1 + 2 = 3$ and this is not a junction tree.

Equality will hold if and only if the running intersection property holds for $X_j$ in $\mathcal{T}_{\mathcal{G}}$. By interchanging the order of summations in (7.34) and applying the inequality in (7.35), it follows that

$$w(\mathcal{T}_{\mathcal{G}}) = \sum_{i=1}^{m-1} \sum_{j=1}^{n} [X_j \in \mathcal{S}(e_i)] \leq \sum_{j=1}^{n} \left( \sum_{i=1}^{m} [X_j \in \mathcal{C}_i] - 1 \right)$$

Interchanging the summations in the rightmost term yields an upper-bound on $w(\mathcal{T}_{\mathcal{G}})$, which can be attained if and only if $\mathcal{T}_{\mathcal{G}}$ is a junction tree (that is, if and only if equality holds in (7.35) for all $1 \leq j \leq n$)

$$w(\mathcal{T}_{\mathcal{G}}) \leq \sum_{i=1}^{m} |\mathcal{C}_i| - n$$

We know from lemma 96 that if $\mathcal{G}$ is triangulated, it must have a junction tree. Given that $\mathcal{G}$ is triangulated, $\widehat{\mathcal{T}_{\mathcal{G}}} = \underset{\mathcal{T}_{\mathcal{G}}}{\text{argmax}}\ w(\mathcal{T}_{\mathcal{G}})$ must be a junction tree and will satisfy $w(\widehat{\mathcal{T}_{\mathcal{G}}}) = \sum_{i=1}^{m} |\mathcal{C}_i| - n$. $\square$

The maximum weight spanning tree problem in (7.34) can be solved exactly by executing the following step $m-1$ times, after intializing $\mathcal{E}_{\Pi}^{all}$ to all possible 'legal' edges between nodes in $\Pi$ and $\mathcal{E}_{\Pi} = \{\}$

1. For $i = 2$ to $m-1$, if

$$\widehat{e} = \underset{e \in acyclic(\mathcal{E}_{\Pi}, \mathcal{E}_{\Pi}^{all})}{\text{argmax}} |\mathcal{S}(e)|$$

   then set $\mathcal{E}_{\Pi} = \mathcal{E}_{\Pi} \cup \{e\}$ and $\mathcal{E}_{\Pi}^{all} = \mathcal{E}_{\Pi}^{all} \setminus \{e\}$.

Here, $acyclic(\mathcal{E}_{\Pi}, \mathcal{E}_{\Pi}^{all}) = \left\{ e \in \mathcal{E}_{\Pi}^{all} \,|\mathcal{E}_{\Pi} \cup \{e\}\ \textit{has no cycles} \right\}$. This can be efficiently implemented using Kruksal and Prim's algorithm. The only additional requirement is that this problem requires specialized data structure to quickly check if $e \in acyclic(\mathcal{E}_{\Pi}, \mathcal{E}_{\Pi}^{all})$, that is, if addition of $e$ to the current set of edges would induce any cycle. This discussion is also relevant for learning tree structured graphical models such that the structure maximizes some objective function on the data.

In Figure 7.11, we present the overall junction tree algorithm. Typically, junction tree propagation is the most expensive step (and is the main 'online' step) and has complexity $O\left(m|\mathcal{C}_{max}|^k\right)$, where $k$ is the maximum number of states for any random variable and $\mathcal{C}_{max}$ is the largest clique. The treewidth $\tau$ of a graph is defined as $\tau = \mathcal{C}_{\max} - 1$ in the optimal triangulation. Thus, the junction tree propagation algorithm scales exponentially in the treewidth $\tau$. There are many elimination orderings/triangulations. The best triangulation is the one that leads to smallest value of $\mathcal{C}_{max}$. The problem of finding the best elimination ordering or of finding the best junction tree is NP-hard. In

**Input**: A graph $\mathcal{G} = < \mathcal{V}, \mathcal{E} >$.
**Pre-processing**:

1. **Moralization**: Convert a directed graph to an undirected graph by connecting parents.
2. Introduce delta functions for observed variables.

**Triangulation**: Triangulate the graph.
**Junction tree construction**: Construct a junction tree from the triangulated graph.
**Junction tree propagation**: Using sum-product or max-product, propagate messages on the junction tree.

Figure 7.11: The high-level view of the Junction tree algorithm.

practice, there are many heuristics that work well. Though NP-hard in a worst case sense, this problem is much easier in the average case.

Many problems do not have bounded treewidth. The junction tree algorithm is limited in its application to families of graphical models that have bounded treewidth. Many common graphical models, such as grid structured graphical model that is commonplace in image processing have very high treewidth. The treewidth of an $n \times n$ grid (*i.e.*, $n^2$ nodes) scales as $O(n)$. Thus, junction tree becomes infeasible for grids as large as $500 \times 500$, though it is applicable in theory.

### 7.2.7 Approximate Inference using Sampling

While the generic junction tree method is principled, it is limited to graphs with bounded treewidth. There are several approximate inference methods that could be considered as alternatives, in practice. One class of approximate inference techniques is the class of sampling methods.

#### Monte Carlo Methods

The general umbrella problem underlying Monte Carlo sampling methods is

$$E[f] = \int p(\mathbf{x})f(\mathbf{x})d\mathbf{x} \tag{7.36}$$

where $f(\mathbf{x})$ is some function defined on some possibly high dimensional space in $\Re^n$ and $p$ is a distribution defined on the same space. For $f(\mathbf{x}) = \mathbf{x}$, $E[f]$ turns out to be the mean. For $f(\mathbf{x}) = \delta(\mathbf{x} = \mathbf{x}')$, $E[f]$ becomes the marginal probability $p(\mathbf{X} = \mathbf{x}')$ or more general, for $f(\mathbf{x}) = \delta(\mathbf{x} \leq \mathbf{x}')$, $E[f]$ becomes the tail probability $p(\mathbf{x} \geq \mathbf{x}')$. The goal of sampling methods, such as Monte

Carlo methods is to approximate such integrals over such possibly high dimensional space using sampling techniques. If we could collect iid samples[17] $\overline{\mathbf{x}} = (\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(m)})$ from $p(.)$, then

$$\widehat{f} = \frac{1}{m}\sum_{i=1}^{m} f(\mathbf{x}^{(i)})$$

is a Monte Carlo estimate of $E[f]$. Since $\overline{\mathbf{x}}$ is a collection of random samples from $p(.)$, $\widehat{f}$ is also a random variable. We could ask some questions about such an estimator:

- Is $\widehat{f}$ unbiased? That is, on an average, will the estimator produce the right quantity? The estimator is unbiased if

  $$E[f] = E_{\overline{\mathbf{x}}}[\widehat{f}]$$

  Using the linearity property of expectation,

  $$E_{\overline{\mathbf{x}}}[\widehat{f}] = \frac{1}{m}\sum_{i=1}^{m} E\left[f(\mathbf{x}^{(i)})\right] = E[f]$$

  that is, the estimator $\widehat{f}$ is indeed unbiased and gives the right answer on an average.

- The unbiased requirement on the part of the estimator only requires the sample mean to match the expected mean, on an average. It may also be desirable that the variance be stable. A related expecation is that as the number $m$ of samples increases, the estimate should get closer to the actual answer. In fact, this is true for the estimator just discussed. It can be shown that if $f$ has finite variance,

  $$var(\widehat{f}) = \frac{1}{m^2}\sum_{i=1}^{m} var\left(f(\mathbf{x}^{(i)})\right) = \sum_{i=1}^{m} \frac{var\left(f(\mathbf{x}^{(i)})\right)}{m} \qquad (7.37)$$

  or equivalently, the spread for $var(\widehat{f})$ is $\frac{1}{\sqrt{m}}$ times the spread for $\widehat{f}$. From this, we can infer that as $m \rightarrow \infty$, $var(\widehat{f}) \rightarrow \infty$.

The discussion thus far was centered around the assumption that we can draw samples from $p(.)$. This area of sampling has warranted seperate attention for research. Even generation of pseudo random numbers is not straightforward[18]. As another example, it is not straightforward to sample efficiently from

[17]The difference between $\overline{\mathbf{x}}$ here and in the case of maximum likelihood estimation is that in the latter case, $\overline{\mathbf{x}}$ is data provided, whereas here, we consider $\overline{\mathbf{x}}$ sampled from the model itself. However, the analytical form is similar.

[18]If a distribution function can be 'inverted' a common strategy is to pass a unform distribution through it to generate samples.

a typically graphical model-like distribution (especially if it is multi-model) such as

$$p(x) = \frac{1}{Z} \underbrace{\exp\left\{ax^4 + bx^3 + cx^2 + dx + e\right\}}_{\wp(x)}$$

where, the $\wp(x)$ part is easy to compute, whereas $\frac{1}{Z}$ is hard to compute.

We will not look at a series of attempts at sampling from a distribution $p(.)$.

### Adopting Numeric Methods

One natural way out is to adopt standard numerical methods, such as the standard numerical recipe for evaluating an integral from first principles - creating discrete intervals and then letting the intervals shrink.

1. Discretize the space of $x$ (such as the real line) into $k$ discrete points, $x_1, x_2, \ldots, x_k$
2. Compute an approximation to $z$ as $\widehat{z} = \sum_{i=1}^{k} \wp(x_i)$.
3. The samples can then be drawn from one of $k$ points based on the distribution $p_d$:
   $$p_d(x_i) = \frac{\wp(x_i)}{\widehat{z}}$$
   The new distribution has point masses at the samples $x_1, x_2, \ldots, x_k$. As the number of grows larger, the approximation will get better

While a controllable approximation that works well in one dimension, how well does such a discretization method scale to higher dimensions. The number of discrete points scales exponentially[19] in the number of dimensions as $k^n$ ($n$ being the dimensionality). Thus, discretization is not feasible in higher dimensions. Another factor in favour of Monte Carlo methods, vis-a-vis numerical techniques is that the statement (7.37) for the Monte Carlo estimator is really independent of the dimensionality $n$.

### Rejection Sampling

Rejection sampling, which dates back to von Neumann in 1950's assumes that in addition to the decomposition $p(\mathbf{x}) = \frac{z}{\wp(\mathbf{x})}$ with $\wp(x)$ easy to compute and $\frac{1}{Z}$ is hard to compute, we also have a proposal distribution $q(\mathbf{x})$ that is relatively easy to (exactly) sample from. $q$ could be one of Gaussians, Cauchy, or some other member of the exponential family.

$$q(\mathbf{x}) = \frac{z_q}{\widehat{q}(\mathbf{x})}$$

[19]This problem also goes under the name of the *curse of dimensionality* - the task that is easy in a single dimension becomes extremely complex at higher dimensions.

Rejection sampling also assumes that you have a constant $M$ such that $\widehat{q}(\mathbf{x})$ scaled by the constant yields an upper bound for the original distribution $\wp(\mathbf{x})$.

$$\wp(\mathbf{x}) \leq M\widehat{q}(\mathbf{x})$$

The way rejection sampling works is:

1. First generate a sample $\mathbf{y} \sim q(\mathbf{x})$.
2. Secondly, sample $u$ from a uniform distribution between 0 and $M\widehat{q}(\mathbf{y})$: $u \sim U[0, M\widehat{q}(\mathbf{y})]$.
    - If $u < \wp(\mathbf{y})$, then accept $\mathbf{y}$ as a realization of $\wp(\mathbf{x})$.
    - Otherwise, reject $\mathbf{y}$.

We will see that this random procedure itself induces a distribution $p_{rej}$ over $\mathbf{x}$ that happens to be the same as $\wp(\mathbf{x})$. For any $\mathbf{y}$ that is an output of the rejection sampling procedure, its probability of being generated by the procedure is the product of the probability $q(\mathbf{y})$ of choosing $\mathbf{y}$ and the probability $\frac{\wp(\mathbf{y})}{M\widehat{q}(\mathbf{y})}$ of accepting $\mathbf{y}$:

$$p_{gen}^{rej}(\mathbf{y}) = \frac{1}{z_{p_{gen}^{rej}}} q(\mathbf{y}) \left( \frac{\wp(\mathbf{y})}{M\widehat{q}(\mathbf{y})} \right) = \frac{1}{z_{p_{gen}^{rej}} z_q} \wp(\mathbf{y})$$

where, the normalization constant $\frac{1}{z_{p_{gen}^{rej}}}$ is defined as

$$z_{p_{gen}^{rej}} = \int_{\mathbf{y}'} q(\mathbf{y}') \left( \frac{\wp(\mathbf{y}')}{M\widehat{q}(\mathbf{y}')} \right) d\mathbf{y}' = \int_{\mathbf{y}'} \frac{1}{z_q} \wp(\mathbf{y}') d\mathbf{y}' = \frac{z_p}{z_q}$$

Combined, these two equalities mean that

$$p_{gen}^{rej}(\mathbf{y}) = \frac{1}{z_{p_{gen}^{rej}} z_q} \wp(\mathbf{y}) = \frac{z_q}{z_{p_{gen}^{rej}} z_q z_p} \wp(\mathbf{y}) = \frac{1}{z_p} \wp(\mathbf{y}) = p(\mathbf{y})$$

In practice, it crucially matters how small you can make the reject region, since you would not like to spend too many sampling cycles to generate each sample $\mathbf{y}$. So it will be best to choose a $\widehat{q}$ that follows $\wp$ very closely. A measure of this 'following closely' is the ratios of the area $A_{acc}$ under $\wp(\mathbf{x})$ to the area $A_{tot}$ under $M\widehat{q}(\mathbf{x})$. This can be thought of as the acceptance probability $p_{acc}^{rej}$.

$$p_{acc}^{rej} = \frac{\displaystyle\int_{\mathbf{x}} \wp(\mathbf{x}) d\mathbf{x}}{\displaystyle\int_{\mathbf{x}} M\widehat{q}(\mathbf{x}) d\mathbf{x}} = \frac{A_{acc}}{M z_q} = \frac{A_{acc}}{A_{tot}}$$

One of the concerns with rejection sampling in high dimensions is that since $p_{acc}^{rej} \propto \frac{1}{M}$, the number of attempts before getting a single sample will scale as $M$. For instance, suppose $\mathbf{X}$ and $\mathbf{Y}$ are independent Gaussians with slightly different variances - $p(\mathbf{x}) = \mathcal{N}(\mathbf{x}, \mathbf{0}, \sigma_p^2 I)$ and $q(\mathbf{y}) = \mathcal{N}(\mathbf{y}, \mathbf{0}, \sigma_q^2 I)$. where $\rho \in (0, 1.1]$. For rejection sampling, we will need to choose an $M > \frac{\wp(\mathbf{x})}{\widehat{q}(\mathbf{x})}$ for every $\mathbf{x} \in \Re^n$. For this Gaussian, we will require that $M > \frac{\wp(\mathbf{0})}{\widehat{q}(\mathbf{0})} = \exp\left\{n \log \frac{\sigma_q}{\sigma_p}\right\}$. Note that if $\sigma_q$ is even slightly larger than $\sigma_p$, then $M$ will increase exponentially with $n$. That is, we will have to do exponentially many rejection trials before getting a sample accepted.

In summary, while rejection sampling is useful in low dimensions, it breaks down in higher dimensions. Markov chain monte carlo (MCMC) builds on rejection sampling. While rejection sampling in memoriless - in the sense that rejected samples are naively abandoned, MCMC preserves rejected samples using memory.

### Importance Sampling

Importance sampling is a more general form of Monte Carlo sampling method. Recall that Monte Carlo sampling was to solve the problem

$$E[f] = \int p(\mathbf{x}) f(\mathbf{x}) d\mathbf{x} \tag{7.38}$$

and the Monte Carlo estimate collects iid samples $\overline{\mathbf{x}} = (\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(m)})$ from $p(.)$ and computes the weighted sum

$$\widehat{f} = \frac{1}{m} \sum_{i=1}^{m} f(\mathbf{x}^{(i)})$$

The idea of importance sampling is to generate samples $\overline{\mathbf{y}}$ from an alternative $q$ which is somewhat easier to sample than $p$. However, how do we make use of $\overline{\mathbf{y}}$ in estimation? Assuming that $q(\mathbf{x}) > 0$ whenever $f(\mathbf{x})p(\mathbf{x}) > 0$, we rewrite the expression for $E[f]$ as

$$E[f] = \int \frac{p(\mathbf{x}) f(\mathbf{x})}{q(\mathbf{x})} q(\mathbf{x}) d\mathbf{x}$$

Defining $g(\mathbf{x}) = \frac{p(\mathbf{x}) f(\mathbf{x})}{q(\mathbf{x})}$, we just re-express

$$E_p[f] = E_q[g]$$

This motivates the study of the Monte Carlo estimate $\widehat{g}$ based on samples $\overline{\mathbf{y}}$ from $q$.

$$\widehat{g} = \frac{1}{m} \sum_{i=1}^{m} g(\mathbf{y}^{(i)}) = \frac{1}{m} \sum_{i=1}^{m} \frac{p(\mathbf{y}^{(i)}) f(\mathbf{y}^{(i)})}{q(\mathbf{y}^{(i)})} \tag{7.39}$$

The estimate $\widehat{g}$ is called the importance sampling estimate. The terms $\frac{p(\mathbf{y}^{(i)})}{q(\mathbf{y}^{(i)})}$ are called *importance weights*. It can be shown that $\widehat{g}$ is unbiased, that is,

$$E_p[f] = E_{\overline{\mathbf{y}}}[\widehat{g}]$$

Like for the case of the Monte Carlo estimate, it can also be shown that if $g$ has finite variance then,

$$var(\widehat{g}) = \sum_{i=1}^{m} \frac{var\left(g(\mathbf{y}^{(i)})\right)}{m} \tag{7.40}$$

or equivalently, the spread for $var(\widehat{g})$ is $\frac{1}{\sqrt{m}}$ times the spread for $\widehat{g}$.

Importance sampling is useful when $q$ is easier to sample than $p$. Backing off a bit, it may happen that $q$ is itself of the form

$$q(\mathbf{y}) = \frac{1}{z_q}\widehat{q}(\mathbf{y})$$

where $\widehat{q}(\mathbf{y})$ is easy to compute while the normalization constant $z_q$ is not. This is especially true in the case of graphical models, for which $\widehat{q}(\mathbf{y})$ is simply the product of some compatibility functions or exponentiated weighted sum of sufficient statistics (exponential family). Similarly, it is often that $p(\mathbf{x}) = \frac{1}{z_p}\wp(\mathbf{x})$. In such a case, we can write

$$\widehat{g} = \frac{1}{m}\sum_{i=1}^{m} \frac{p(\mathbf{y}^{(i)})f(\mathbf{y}^{(i)})}{q(\mathbf{y}^{(i)})} = \frac{1}{m}\sum_{i=1}^{m} \frac{\wp(\mathbf{y}^{(i)})f(\mathbf{y}^{(i)})}{\widehat{q}(\mathbf{y}^{(i)})}\frac{z_q}{z_p}$$

For this formulation, $\frac{\wp(\mathbf{y}^{(i)})}{\widehat{q}(\mathbf{y}^{(i)})}$ are called the importance weights $im(\mathbf{y}^{(i)})$. Also,

$$z_{p/q} = \frac{z_p}{z_q} = \frac{1}{z_q}\int \wp(\mathbf{x})d\mathbf{x} = \int \wp(\mathbf{x})\frac{q(\mathbf{x})}{\widehat{q}(\mathbf{x})} = \int \frac{\wp(\mathbf{x})}{\widehat{q}(\mathbf{x})}q(\mathbf{x})d\mathbf{x}$$

which is again the expectation under $q$ of $im(\mathbf{y})$. The Monte Carlo estimate $\widehat{z}_{p/q}$ for $z_{p/q} = \frac{z_p}{z_q}$ is

$$\widehat{z}_{p/q} = \frac{1}{m}\sum_{i=1}^{m} im(\mathbf{y}^{(i)}) = \frac{1}{m}\sum_{i=1}^{m}\frac{\wp(\mathbf{y}^{(i)})}{\widehat{q}(\mathbf{y}^{(i)})}$$

This gives you a modified Monte Carlo estimate, which can be contrasted against (7.39).

$$\widehat{g}' = \frac{\frac{1}{m}\sum_{i=1}^{m}\frac{p(\mathbf{y}^{(i)})f(\mathbf{y}^{(i)})}{q(\mathbf{y}^{(i)})}}{\frac{1}{m}\sum_{i=1}^{m}\frac{\wp(\mathbf{y}^{(i)})}{\widehat{q}(\mathbf{y}^{(i)})}} = \frac{\sum_{i=1}^{m} f(\mathbf{y}^{(i)})im(\mathbf{y}^{(i)})}{\sum_{i=1}^{m} im(\mathbf{y}^{(i)}} \tag{7.41}$$

This modified Monte Carlo estimate uses samples $\overline{\mathbf{y}}$ from $q$ and also does not require explict computation of $p$. Rather it needs to compute only $\wp$.

Importance sampling considerably widens the scope of Monte Carlo methods, since it provides flexibility of choosing $q$. In fact, even if you could sample from $p$, it can be useful to use a $q$, especially when $f$ lies in the tail region(s) of $p$. Since $q$ lets you reshape what you sample from, the modified $g$ can help shift the mass over to regions of $p$ such that getting information from $f$ becomes much more likely. In practice, adapting $q$ to the shape of $f$ can also lead to reduction in the variance of $\widehat{g}$.

Finally, importance sampling can be useful even when you can draw samples from $p(.)$. For example, say $X_i \in \{0,1\}$, for $1 \leq i \leq n$ are $n$ binary random variables with $p(X_i = 1) = \epsilon$ for a very small $\epsilon$, close to 0. Let the $X_i$'s be independent (though the example could hold for many non-independent $X_i$'s as well). Let us say, we are interested in estimating

$$p\left(\frac{\sum\limits_{i=1}^{n} x_i}{n}\right) \geq 0.5$$

As can be seen, that $\frac{\sum\limits_{i=1}^{n} x_i}{n} \geq 0.5$ is a very rare event. Importance sampling can be used to model such rare events which are otherwise computationally infeasible to simulate.

Like in the case of rejection sampling and numerical techniques, there are problems that importance sampling techniques have in high dimensional spaces.

### Markov Chain Monte Carlo

Markov Chain Monte Carlo (MCMC) methods are a suite of methods based on setting up a markov chain to generate samples from a target distribution

$$p(\mathbf{x}) = \frac{1}{z}\wp(\mathbf{x})$$

where $z$ may not be easy to compute. The basic idea here is to set up a Markov chain $X_1 \rightarrow X_2 \rightarrow \ldots X_n$ and a sequence of distributions $q$ so that as $t \rightarrow \infty$, $q(.|X^{(t)}) \rightarrow p$. While these methods have a flavour of rejection sampling, they are not memoryless; rejected samples are not entirely discarded. Introduced by physicists Metropolis, Rosenbluth, Teller in 1953, these methods were generalized by Hastings in 1970.

The Metropolis-Hastings algorithm is one of the more popular examples from this class. It is outlined in Figure 7.12. It builds on the rejection sampling technique, with two main differences. More importantly, a sample is not discarded if rejected; the value of $X_{t+1}$ is set to $X_t$. Secondly, the acceptance probability determines if it is more likely to go $\mathbf{X}^{(t)} \rightarrow \mathbf{y}$ or $\mathbf{y} \rightarrow \mathbf{X}^{(t)}$.

```
**Input**: A target distribution $p(\mathbf{x}) = \frac{1}{z}\wp(\mathbf{x})$.
**Initialize**: $\mathbf{X}^{(1)} \sim q(\mathbf{x})$ for some arbitrary $q$.
**for** $t = 1, 2, \ldots$ **do**
    1. Sample $\mathbf{y}$ from the conditional probability distribution $q(.|\mathbf{X}^{(t)})$.
    2. Sample $u \sim Uniform[0,1]$.
    3. Define the acceptance probability as
       $$A(\mathbf{X}^{(t)}, \mathbf{y}) = \min\left\{1, \frac{\wp(\mathbf{y})q(\mathbf{X}^{(t)}|\mathbf{y})}{\wp(\mathbf{X}^{(t)})q(\mathbf{y}|\mathbf{X}^{(t)})}\right\}$$
    4. Set
       $$\mathbf{X}^{(t+1)} = \begin{cases} \mathbf{y}, & \text{if } u \leq A(\mathbf{X}^{(t)}, \mathbf{y}) \quad //\text{Accept} \\ \mathbf{X}^{(t)} & \text{otherwise} \quad //\text{Reject, but do not discard.} \end{cases}$$
**end for**
```

Figure 7.12: The Metropolis-Hastings algorithm.

In the special case of symmetry, that is if $q(\mathbf{y}|\mathbf{X}^{(t)}) = q(\mathbf{X}^{(t)}|\mathbf{y})$,

$$A(\mathbf{X}^{(t)}, \mathbf{y}) = \min\left\{1, \frac{\wp(\mathbf{y})}{\wp(\mathbf{X}^{(t)})}\right\}$$

the algorithm acts like a gradient algorithm with some randomness. To see this, note that if $\wp(\mathbf{y}) \geq \wp(\mathbf{X}^{(t)})$, the algorithm will always accept. Else it accepts with probability $\frac{\wp(\mathbf{y})}{\wp(\mathbf{X}^{(t)}} < 1$; the logic is that you do not always want to reject even if you were to go downhills, since you might want to wriggle out of local modes. So you reject, but probabilistically. Thus, the algorithm always tries to sample from regions of the space where the density is more.

We will eventually see that if $q(\mathbf{y}|\mathbf{X}^{(t)}) \rightarrow p(\mathbf{y})$ as $t \rightarrow \infty$. Enroute to proving this, we will require to understand the limiting behaviour of Markov chains as number of states goes to $\infty$.

1. The Metropolis-Hastings algorithm in Figure 7.12 generates a first order Makov chain. Assume for simplicity that the Markov chain is finite state and that $\mathbf{X}^{(i)} \in \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_k\}$. For many graphical models, $k$ could be exponential in the number of nodes in the graphical model. For the Markov chain $\mathbf{X}^{(1)}, \mathbf{X}^{(2)}, \mathbf{X}^{(3)}, \ldots$ generated by the algorithm in Figure 7.12, $\mathbf{X}^{(1)} \sim q(.)$ while the transition $\mathbf{X}^{(t)} \rightarrow \mathbf{X}^{(t+1)}$ is specified by the homogeneous (*i.e.* fixed across time steps) conditional distribution

$$\Gamma(\mathbf{X}^{(t+1)}|\mathbf{X}^{(t)}) = A(\mathbf{X}^{(t)}, \mathbf{y})q(\mathbf{y}|\mathbf{X}^{(t)})$$

That is, the transition function for the algorithm is the combination of the original proposal distribution and the probability of acceptance in the accept/reject step. Then, by the steps (1) and (4) of the algorithm in Figure 7.12,

$$q(\mathbf{X}^{(t+1)}) = \sum_{\mathbf{X}^{(t)}} q(\mathbf{X}^{(t)})\Gamma(\mathbf{X}^{(t+1)}|\mathbf{X}^{(t)})$$

In matrix notation, this translates to

$$\mathbf{q}_{t+1} = \mathbf{q}_t^T T \tag{7.42}$$

where, $\Gamma[i,j] = \Gamma(\mathbf{X}^{(t+1)} = \mathbf{x}_j|\mathbf{X}^{(t)} = \mathbf{x}_i)$ and $\mathbf{q}_t[i] = q(\mathbf{X}^{(t)} = \mathbf{x}_i)$. By its very definition, $\Gamma$ is row-stochastic, that is,

$$\Gamma\mathbf{1} = \mathbf{1}$$

We would like $\mathbf{q} \rightarrow \mathbf{p}$, where $\mathbf{p}$ is the targetted probability vector. The following sequence of arguments will help arrive at conditions under which this will happen.

2. Let $\mathbf{r}$ be the fix point of update equation (7.42). Then $\mathbf{r}$ is also invariant[20] with respect to $\Gamma$. Thus, once the distribution $\mathbf{q}_t$ hits an invariant $\mathbf{r}$, it stays in $\mathbf{q}_t$.

3. In order to have an invariant vector, the matrix must be non-negative ($\Gamma \geq 0$) and must be row-stochastic, which it is. The matrix $\Gamma$ is not symmetric in general. The Perroon Forbenius theorem states that for any non-negative, row-stochastic matrix $A$, its spectral radius $\rho(A) = \max_{i=1,2,\ldots,n} |\lambda_i(A)|$ satisfies the condition $\rho(A) = 1$ and that it has a left eigenvector $\mathbf{v} \geq \mathbf{0}$ such that $\mathbf{v}^T A = \mathbf{v}$. Since the matrix $\Gamma$ is both non-negative and row-stochastic, a consequence of this theorem is that $\mathbf{r} = \frac{1}{\sum_{i=1}^{k} v_k}\mathbf{v}$ will be invariant with respect to $\Gamma$.

4. The above conditions and arguments state in principle that there is potentially a fix point for (7.42). In general, the fix point need not be unique; for a diagonal matrix, there are several invariant distributions. It can shown that an irreducible matrix $\Gamma$ (*i.e.*, every state is reachable from every other state with strictly positive probability) will have a unique invariant distribution $\mathbf{r}$.

[20]A vector $\mathbf{r} \in \Re^k$ is invariant with respect to matrix $\Gamma$ if $\mathbf{r}$ represents a probability distribution (*i.e.*, $\mathbf{r}^T\mathbf{1} = \mathbf{1}$ and $\mathbf{r} \geq \mathbf{0}$) and is fixed under the updates of $\Gamma$, that is $\mathbf{r}^T\Gamma = \mathbf{r}$. As an example, you can verify that for random walks on a graph $\mathcal{G} =< \mathcal{V}, \mathcal{E} >$ with 0 jump probability, $\mathbf{r}$ such that $v_i = \frac{d_i}{\sum_{i\in\mathcal{V}} d_i}$ is invariant, $d_i$ being the degree of vertex $i$.

5. But will the algorithm in Figure 7.12 converge to the fix point? To enable convergence, another necessary condition is that the Markov chain should be aperiodic (so that $q_t$ does not keep toggling between values) or equivalently, have a period of 1.

6. With all the armour discussed so far, we state the following theorem, central to the correct convergence of the algorithm in Figure 7.12:

   **Theorem 98** *For any finite-chain irreducible aperiodic Markov Chain, the sequence* $\mathbf{q}_{t+1} = \mathbf{q}_t^T \Gamma$ *converges, for an arbitrary* $\mathbf{q}_1$, *to the unique invariant distribution* $\mathbf{r}$ *of the chain.*

   The next few steps are dedicated to proving that (i) the Metropolis-Hastings algorithm satisfies the pre-requisites for this theorem under certain conditions and (ii) that the target distribution is indeed invariant with respect to the Markov chain in the Metropolis-Hastings algorithm.

7. The next step is to establish that the matrix $\Gamma$ defined as

   $$\Gamma(\mathbf{X}^{(t+1)}|\mathbf{X}^{(t)}) = A(\mathbf{X}^{(t)}, \mathbf{y})q(\mathbf{y}|\mathbf{X}^{(t)})$$

   is irreducible and aperiodic. The Metropolis-Hastings algorithm is very general, allowing fairly arbitrary proposal distribution. Both these properties can be established if $q(\mathbf{y}|\mathbf{X}^{(t)}) > 0$ for all $\mathbf{y}$, so that $\Gamma(\mathbf{X}^{(t+1)}|\mathbf{X}^{(t)}) > 0$ for all $\mathbf{y}$. For complex models such as graphical models, the $q$ should be designed so as to make that task of sampling from $q$ efficient. Gibbs sampling is one such example.

8. The final step is in showing that the target distribution $\mathbf{p}$ is invariant for the Markov chain $\Gamma$. That is, for every $\mathbf{y}$,

   $$\sum_{\mathbf{x}} p(\mathbf{x})\Gamma(\mathbf{y}|\mathbf{x}) = p(\mathbf{y})$$

   This can be shown by first noting that the Metropolis Hastings Markov chain $\Gamma$ satisfies the *detailed balance condition* with respect to $\mathbf{p}$, which means

   $$p(\mathbf{x})\Gamma(\mathbf{y}|\mathbf{x}) = p(\mathbf{y})\Gamma(\mathbf{x}|\mathbf{y})$$

   This can be proved by simply substituting for $\Gamma(\mathbf{y}|\mathbf{x})$. This leads to the desired result

   $$\sum_{\mathbf{x}} p(\mathbf{x})\Gamma(\mathbf{y}|\mathbf{x}) = \sum_{\mathbf{x}} p(\mathbf{y})\Gamma(\mathbf{x}|\mathbf{y}) = p(\mathbf{y})$$

   since $\Gamma$ is row-stochastic.

While the algorithm in Figure 7.12 works in principle, it can be very slow, taking small steps into valleys. In practice, many refinements are made to the algorithm to make it work fast.

**Gibbs Sampling**

Gibbs sampling is a simple subclass of Metropolis-Hastings algorithm in which the transitions $\Gamma$ are intuitive and easy to draw from. It is especially relevant for graphical models, which have a large state space of size $c^n$ if each random variable can take $c$ values and the graph has $n$ nodes. Let $\mathcal{G} =< \mathcal{V}, \mathcal{E} >$ be a graph with $n$ nodes. The Gibbs sampling procedure involves two main steps, as outlined in Figure 7.13 It is very natural to think of Gibbs sampling in terms of

```
**Input**: A target distribution $p(\mathbf{x}) = \frac{1}{z}\wp(\mathbf{x})$ over a graph $\mathcal{G} =< \mathcal{V}, \mathcal{E} >$.
**Initialize**: $\mathbf{X}^{(1)} = \mathbf{x}^{(1)}$ for some arbitrary $q$.
**for** step $t = 1, 2, \ldots$ **do**
  **for** $i = 1, \ldots, n$ **do**
    Sample: $x_i^{(t+1)} \sim p(x_i \mid \mathbf{x}_{1,i-1}^{(t+1)} \cup \mathbf{x}_{i+1,n}^{(t)}$.
  **end for**
**end for**
```

Figure 7.13: The Gibbs sampling algorithm.

the graphical model; the sampling is very much simplified given the conditional independence property - that a node is indepent of all other nodes, given its Markov blanket. Thus, each distribution in the sampling step is the probability of a node given its Markov blanket. This procedure depends on the ordering of nodes, though. Though a very popular inference procedure for graphical models, it could take an extemely long time to converge.

For discrete random variables, the graph of configurations is in general a hypercube. While MCMC potentially allows jumps from one node to another in the hypercube, Gibbs sampling restricts every step to be along an edge of the hypercube. This can lead to poor convergence. There are many smarter algorithms that take larger, but calculated steps in the hypercube, leading to better convergence, without incurring excessive computational cost for individual steps.

## 7.3 Factor Graphs

We have seen that both directed and undirected models can be specified in one of two equivalent ways each; conditional independence and factorization. The semantics for directed and undirected models are however different. While the two specifications are equivalent, factorization can be specified at different levels of granularity or could be defined using different forms of potential functions and is therefore richer. For instance, factorization could be over maximal cliques or over smaller cliques or even over edges, while all these specifications could map to the same conditional independence properties demanding specialization in the conditional independence assertions.

Consider a triangular graph, given by the adjacency matrix

$$A = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

The factorization for this graph could be presented as

$$p(x_1, x_2, x_3) \propto \phi_{123}(x_1, x_2, x_3)$$

However, the following factorization is also a valid one for the graph

$$p(x_1, x_2, x_3) \propto \phi_{12}(x_1, x_2)\phi_{23}(x_2, x_3)\phi_{13}(x_1, x_3)$$

Such problems interest statisticians a lot, since existence of interactions between variables influence diagnosis and/or treatments in the medical domain, for example. Is it possible to graphically differentiate between the two factorizations? Factor graphs precisely serve this purpose. In some sense, there is more information being embedded in the second factorization; the bonafide triplet interaction has been decomposed into pairwise interactions. And this information is represented as 'factor nodes' in factor graphs. Corresponding to each factor in the factorization, factor graphs host a node. The nodes in the original graphical model are also retained. But edges are changed; there will be an edge between the factor node and each node whose random variable the factor is a function of.

**Definition 55** *Given a graphical model $\mathcal{G} =< \mathcal{V}, \mathcal{E} >$, with a factorization $p(\mathbf{x}) = \prod_{a \in \mathcal{F}} \phi_a(\mathbf{x}_a)$ with $\mathcal{F} \subseteq 2^{\mathcal{V}}$, that is compatible with the independence assumptions asserted by the edges $\mathcal{E}$, the factor graph $\mathcal{G}_f =< \mathcal{V} \cup \mathcal{F}, \mathcal{E}_f >$ is defined by $\mathcal{E}_f = \{(q, a) \mid q \in \mathcal{V}, a \in \mathcal{F}, q \in a\}$ The factor graph $\mathcal{G}_f$ is said to encode the factorization of $\mathcal{G}$.*

The set of factor nodes $\mathcal{F}$ is a set of placeholders for particular terms in the factorization. Factor graphs can be looked upon as a 'graphical way' of representing hypergraphs (which can have a single edge spanning multiple vertices), where each hyperedge can be thought of as spanning all vertices that figure together in some potential function in the factorization.

As an example, the factor graph for $p(x_1, x_2, x_3) \propto \phi_{123}(x_1, x_2, x_3)$ will be given by $\mathcal{V} = \{1, 2, 3\}$, $\mathcal{F} = \{a\}$ and $\mathcal{E}_f = \{(1, a), (2, a), (3, a)\}$. Whereas, the factor graph for $p(x_1, x_2, x_3) \propto \phi_{12}(x_1, x_2)\phi_{13}(x_1, x_3)\phi_{23}(x_2, x_3)$ will be given by $\mathcal{V} = \{1, 2, 3\}$, $\mathcal{F} = \{a, b, c\}$ and $\mathcal{E}_f = \{(1, a), (2, a), (1, b), (3, b), (2, c), (3, c)\}$.

Any undirected/directed graph without any specialized factorization specified can also be converted into a factor graph. Thus, if $\mathcal{V} = \{1, 2, 3, 4, 5, 6, 7, 8\}$, $\mathcal{E} = \{(1, 2), (2, 3), (3, 4), (2, 5), (3, 4), (4, 5), (4, 7), (5, 7), (5, 6), (6, 8), (7, 8)\}$, then $\mathcal{G}_f$ can be read off the maximal cliques; $\mathcal{F} = \{a, b, c, d, e\}$ and $\mathcal{E}_f = \{(1, a), (2, a), (3, a), (2, b), (5, b), (3, c), ($
As another example, consider a hidden markov model with $\mathcal{V} = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$

and $\mathcal{E} = \{(1,2),(1,6),(2,3),(2,7),(3,4),(3,8),(4,5),(4,9),(5,10)\}$. Then the factor graph will have $\mathcal{F} = \{a,b,c,d,p,q,r,s\}$ and $\mathcal{E}_f = \{(1,a),(2,a),(2,b),(3,b),(3,c),(4,c),(4,d),(5,d),(1,p),(6,p),$ As a final example, consider a markov decision process, which is a purely directed model and which has five 'control' variables in addition to the 10 for the HMM described above (decision process because there is a control that helps you decide the behaviour of the evolution across states in the HMM). It will be specified by $\mathcal{V} = \{1,2,3,4,5,6,7,8,9,10,11,12,13,14,15\}$ and $\mathcal{E} = \{(1,2),(1,6),(11,1),(2,3),(2,7),(12,2),(3,4),(3,8),(13,3),(4,5),(4,9),(14,4),(5,10),(15,5)\}$. What will be the factor graph corresponding to this graph? Note that, even though there are no directed triangles in this graph, nodes 1 throug 5 have two parents each and the corresponding factors are conditional probabilities of the form $p(x_1|x_6,x_{11})$, $p(x_2|x_7,x_{12})$, *etc.* Thus, the factor graph will have a node for each such factor connected to three nodes each.

All the factor graph examples considered thus far are factor trees. The sum-product and max-prodyct algorithms are exact on factor trees. Though they assume slightly different forms, the essential ideas are the same. There is no consistent reason for the factor graph being a better representation than the graphical model representation.

## 7.4 Exponential Family

The exponential family captures many common discrete[21] and continuous[22] graphical model formulations at an abstract level. It provides a rich (though not exhaustive) toolbox of models. The multinomial distribution which models random variables taking $k$ discrete values is what we have looked at so far. Gaussian is one of the most widely used continuous distributions. The poisson distribution helps model distribution over random variables that can take any integral value (as against just $k$ discrete values).

**Definition 56** *For a given vector of functions $\mathbf{f}(\mathbf{x}) = [f_1(\mathbf{x}),\ldots,f_k(\mathbf{x})]$ and a parameter vector $\eta \in \Re^k$, the exponential family of distributions is defined as*

$$p(\mathbf{x},\eta) = h(\mathbf{x})\exp\left\{\eta^T\mathbf{f}(\mathbf{x}) - A(\eta)\right\} \tag{7.43}$$

*where the $h(\mathbf{x})$ is a conventional reference function and $A(\eta)$ is the log normalization constant*[23] *designed as*

$$A(\eta) = \log\left[\int_{\mathbf{x}\in Range(\mathbf{X})} \exp\{\eta^T\mathbf{f}(\mathbf{x})\}h(\mathbf{x})d\mathbf{x}\right]$$

*The domain of the parameters $\eta$ will be restricted to $Domain(\eta) = \left\{\eta \in \Re^k \mid A(\eta) < +\infty\right\}$. $A(\eta)$ plays a fundamental role in hypothesis testing, parameter estimation, etc.,*

[21]Density with respect to the Counting measure.
[22]Density with respect to the Lebesgue measure.
[23]The same as $\log Z$ for graphical models.

*though often not very easy to estimate. The central component here is the log-linear form. The parametrization $\eta \in \Re^k$, is also called the canonical parametrization. The function $\mathbf{f}(\mathbf{x})$ is a sufficient statistic function.*

As an example, the Gaussian density function can be expressed in the exponential form. If $X \sim \mathcal{N}(\mu, \sigma^2)$ is a univariate Gaussian distribution, then its normal parametrization is

$$p(x, \mu, \sigma^2) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left\{-\frac{1}{2\sigma^2}(x-\mu)^2\right\}$$

This density can be manipulated and shown to be a member of the exponential family

$$p(x, \mu, \sigma^2) = \frac{1}{\sqrt{2\pi}} \exp\left\{\frac{\mu}{\sigma^2}x - \frac{1}{2\sigma^2}x^2 - \frac{1}{2\sigma^2}\mu^2 - \log\sigma\right\} = p(x, \eta)$$

The parameter vector for the exponential form is $\eta = \left[\frac{\mu}{\sigma^2}, -\frac{1}{2\sigma^2}\right]$ while the feature function vector is $\mathbf{f}(x) = [x, x^2]$. The log normalization constant is $A(\eta) = \frac{1}{2\sigma^2}\mu^2 + \log\sigma \equiv -\frac{\eta_1^2}{4\eta_2} + \frac{1}{2}\log(-2\eta_2)$, which determines $Domain(\eta) = \left\{\eta \in \Re^2 \mid \eta_2 < 0\right\}$. Finally $h(\mathbf{x}) = \frac{1}{\sqrt{2\pi}}$ for this example. The number of degrees of freedom (reflected through two parameters in the moment parametrization) will precisely be the number of canonical parameters. The canonical parametrization extended[24] to the multivariate Gaussian counterpart will be discussed in Section 7.5.

As a second example, consider the bernoulli distribution. A bernoulli random variable $X \in \{0, 1\}$ can model the outcome of any coin-flipping experiment. It can be expressed as

$$p(x, \mu) = \mu^x(1-\mu)^{1-x}$$

where $\mu$ is the probability of $X = 1$. Rearranging the terms, we get the exponential form for bernoulli distribution as

$$p(x, \mu) = \exp\left\{\left(\log\frac{\mu}{1-\mu}\right)x + \log(1-\mu)\right\} = p(x, \eta)$$

with parameter vector specified as $\eta = \left[\log\frac{\mu}{1-\mu}\right]$ which gives $\mu$ as a logistic function (log of likelihood ratio or log-odds ratio) of $\eta_1$ and $\mathbf{f}(x) = [x]$. The log normalization constant is $A(\eta) = -\log(1-\mu) \equiv \log\{1 + e^{\eta_1}\}$[25] while $h(x) = 1$. Also, $Domain(\eta) = \Re$. In general, if you started coupling together multiple distributions and try expressing them in exponential form, determining $\eta$ could become a very hard problem.

As a third example, consider the poisson[26] random variable $X \in \mathcal{N}$ (set of natural numbers). Its density function is given by

$$p(x, \mu) = \frac{\mu^x e^{-\mu}}{x!}$$

[24]EXERCISE.

[25]EXERCISE.

[26]When London was being bombed in World war 2, experts were trying to model where the bomb would next fall using a 2D poisson distribution.

Poisson distributions are often used to model events, such as the ringing of a telephone. The mean parameter $\mu$ controls the density or frequency with which the event is happening. The canonical parametrization for this distribution can be obtained using a routine rewrite as

$$p(x,\mu) = \frac{1}{x!}\exp\{(\log\mu)x - \mu\}$$

where $h(x) = \frac{1}{x!}$, $\eta = [\log\mu]$, $\mathbf{f}(x) = [x]$, $A(\eta) = \mu = e^{\eta_1}$ with $Domain(\eta) = \Re$.

In all examples considered so far, we algebriacally converted the density function from a moment parametrization to a canonical representation as a member of the exponential family. How about going backwards from a canonical form to moment parametrization? The vector of moment parameters is defined as $\mu = [\mu_1, \mu_2, \ldots \mu_k] = [E[f_1(\mathbf{X})], E[f_2(\mathbf{X})], \ldots, E[f_k(\mathbf{X})]]$. That is, we can get the moment parameters by taking expectations of the sufficient statistics $f_i(\mathbf{X})$ that sit in the exponent of the canonical form. And the expecation of the $i^{th}$ component of this function can be proved to be the same as $\frac{\partial A}{\partial \eta_i}$. That is

$$\nabla A(\eta)_i = \frac{\partial A}{\partial \eta_i} = E[\mathbf{f}(\mathbf{X})] = \mu_i \tag{7.44}$$

Further, it can be proved that

$$\nabla^2 A(\eta)_{ij} = \frac{\partial^2 A}{\partial\eta_i\partial\eta_j} = cov\{f_i(\mathbf{X}), f_j(\mathbf{X})\} = E\left[(f_i(\mathbf{X}) - \mu_i)(f_j(\mathbf{X}) - \mu_j)\right] = \mu_{ij}$$

The proof of the two above statements are straightforward and use the property that $A(\eta)$ is infinitely differentiable. To illustrate this, we will revisit the canonical parametrization for the Gaussian example.

$$p(x,\mu,\sigma^2) = \frac{1}{\sqrt{2\pi}}\exp\left\{\frac{\mu}{\sigma^2}x - \frac{1}{2\sigma^2}x^2 - \frac{1}{2\sigma^2}\mu^2 - \log\sigma\right\} = p(x,\eta)$$

The moments can be derived to me $\mu_1 = E[x] = \mu$ and $\mu_2 = E[x^2] = \sigma^2 + \mu$. Similarly, for the poisson distribution, the moment parameter is simply $[\mu]$ as also $var(X) = \mu$. For the bernoulli distribution, $E(X) = \mu$ and $var(X) = (1-\mu)\mu$.

## 7.5 A Look at Modeling Continuous Random Variables

A normal or gaussian distribution is one of the most widely (ab)used probability distributions. They come up in the central limit theorem in the sums of independent or weakly dependent random variables, whose normalized sum coverges to a gaussian (justifying their wide use). They are very often used to model noise. Example graphical models that use gaussian distribution are Gaussian graphical models, Gauss-Markov time series, *etc.*

Another reason for their wide use is computational. In the case of continuous random variables, messages are functions rather than vectors. In general, this can often force us to quantize the messages. But in the case of gaussians (and in general for exponential models which we will shortly see), the message passing can be performed in terms of sufficient statistics. Kalman filters are a special case of message passing that pass sufficient statistics as messages.

The form of the density function for Gaussian distribution, with $\mathbf{x}, \mu \in \Re^n$ and $\Sigma \in \Re^{n\times n}$ is

$$p(\mathbf{x}, \mu, \Sigma) = \frac{1}{(2\pi)^{\frac{1}{2}}\sqrt{det(\Sigma)}} \exp\left\{\frac{1}{2}(\mathbf{x}-\mu)^T\Sigma^{-1}(\mathbf{x}-\mu)\right\} \tag{7.45}$$

where $\mu$ is the mean vector and $\Sigma \succ \mathbf{0}$ is the covariance matrix[27]. It can be verified that $\mu$ is indeed the mean vector; $\mu = E[\mathbf{X}] = \int_{\Re^n} \mathbf{x}p(\mathbf{x}, \mu, \Sigma)d\mathbf{x}$. Similarly, it can be verified that $\Sigma = E[(\mathbf{X}-\mu)(\mathbf{X}-\mu)^T]$. The quantity $\frac{1}{(2\pi)^{\frac{1}{2}}\sqrt{det(\Sigma)}}$ is the normalization constant and can be computed as $\int_{\Re^n} \exp\left\{-\frac{1}{2}(\mathbf{x}-\mu)^T\Sigma^{-1}(\mathbf{x}-\mu)\right\}d\mathbf{x}$. If $n = 1$, the integral is easy to compute. For computing integrals for multivariate problem, it is a good idea to reduce the matrix $\Sigma$ by diagonalization.

What sits in the exponent of the density function is a quandratic term. The contours of constant probability are ellipsoids, with shape determined by $\Sigma^{-1}$ and center as $\mu$ (which does not affect the shape of the ellipsoid). For example, if $\Sigma^{-1}$ is diagonal, the axes of the ellipsoid will be aligned along the coordinate axes.

The parametrization $(\mu, \Sigma)$ is called the *moment parametrization* of the Gaussian; $\mu$ is the first order moment of $\mathbf{x}$ while $\Sigma$ is the matrix of second order centered moment. There is another *canonical parametrization* of the Gaussian which is related to the sum-product algorithm. The parameters are a new vector and a new matrix:

$$\eta = \Sigma^{-1}\mu$$

and a new matrix

$$\Omega = \Sigma^{-1}$$

With a bit of algebra, the Gaussian density can be re-expressed in terms of the canonical parameters as:

$$p(\mathbf{x}, \eta, \Omega) = \exp\left\{\eta^T\mathbf{x} - \frac{1}{2}\mathbf{x}^T\Omega\mathbf{x} + \mathbf{x}\right\} \tag{7.46}$$

[27]Recall that a matrix is positive definite if all its eigenvalues are strictly positive or equivalently, $\forall \mathbf{z} \neq \mathbf{0},\ \mathbf{z}^T\Sigma\mathbf{x} > 0$.

where

$$\mathbf{x} = -\frac{1}{2}\left\{n\log(2\pi) - \log|\Omega| + \eta^T\Omega^{-1}\eta\right\}$$

is a constant, analogous to the normalization constant in the moment parametrization of the Gaussian density in (7.45). The parametrization for (7.45) is more naturally suited for graphical models, because, as we will see, the canonical parameters can be directly related to the compatibility functions when you start breaking up the Gaussian into a graphical model-like factorization. Typically, graphical model factorizations invoke canonical parameters. The above parametrization is often referred to as $\mathcal{N}(\eta, \Omega)$.

Equation (7.46 gives a quadratic form in the exponent and is a special case of the exponential family representation, which happens to be a very general concept. Note that all logarithms are to the base $e$. The quadratic term in the exponent can be rewritten as a trace:

$$\mathbf{x}^T\Omega\mathbf{x} = trace(\Omega\mathbf{x}\mathbf{x}^T)$$

where $trace(A)$ is the sum of the diagonal entries of $A$ and happens to be linear. The Gaussian density using this transformation is obtained in its log-normal form, which has an exponent linear in its features of $\mathbf{x}$ and $\mathbf{x}\mathbf{x}^T$.

It can be shown that linear functions of Gaussian random vectors are Gaussian; $\sum_{i=0}^{n} Y_i \sim \mathcal{N}(0,(n+1)\sigma^2)$ if each $Y_i \sim \mathcal{N}(0,\sigma^2)$. Also, if $X_i = \sum_{j=0}^{n} Y_i$ then $p(x_{i+1} \mid x_i) = \exp\left\{-\frac{1}{2\sigma^2}(x_{i+1}-x_i)^2\right\}$ and the random variables $X_i$ form a markov chain with factorization

$$p(\mathbf{x}) = \exp\left\{-\frac{1}{2\sigma^2}x_0^2\right\}\prod_{i=1}^{n}\exp\left\{-\frac{1}{2\sigma^2}(x_{i+1}-x_i)^2\right\}$$

We can verify that for this markov chain, $E[X_1] = E_{X_0}\left[E_{X_1|X_0}[X_1 \mid X_0]\right] = E_{X_0}\left[X_0 + E[Y_0]\right] = 0$ and in general $E[X_i] = 0$. Also, $E[X_1X_2] = 2\sigma^2$.

The canonical parameters for $p(\mathbf{x})$ can be read off the factorization as $\eta = \mathbf{0}$ and a tridiagonal, sparse

$$\Omega = \begin{bmatrix} 2 & -1 & 0 & \dots & 0 \\ -1 & 2 & -1 & \dots & 0 \\ 0 & -1 & 2 & \dots & 0 \\ 0 & 0 & -1 & \dots & 0 \\ . & . & . & \dots & . \\ . & . & . & \dots & -1 \\ 0 & 0 & 0 & \dots & 2 \end{bmatrix}$$

The matrix $\Omega$ is sparse, because the graph is sparse (just a markov chain). Missing edges translate to zeros; for any graph, if there is no edge between $X_i$ and $X_j$, $\Omega_{ij} = 0$.

Factor analysis is another example application of gaussian density functions. Principal component analysis and Karhunen Loeve transform are limiting cases of factor analysis. The motivation here is dimensionality reduction for cases when $\mathbf{Y} \in \Re^n$ and $n$ is very large, such as the size of an image for a naive $1-d$ raster scanned pixel vector representation. The complete vector need not really capture the intrinsic dimensionality of the object being described (such as a face). Factor analysis is useful if you think that the data should be lying in some lower ($d << n$) dimensional space[28]. Let $\omega_1, \omega_2, \ldots, \omega_d \in \Re^n$ be $d$ (linearly independent) basis vectors. Consider the linear subspace $M = \left\{ \mathbf{y} \in \Re^n \middle| \mathbf{y} = \sum_{i=1}^{d} x_i\omega_i,\ x_i \in \Re,\ \omega_i \in \Re^n \right\}$. Factor analysis tries to induce a probabilitic distribution over the subspace $M$, by defining $\mathbf{X} \sim \mathcal{N}(\mathbf{0}_d, I_{d\times d})$ and

$$\mathcal{Y}_{n\times 1} = \mu_{n\times 1} + \Omega_{n\times d}\mathbf{X}_{d\times 1} + \mathbf{K}_{n\times 1}$$

where the $i^{th}$ column of $\Omega$ is $\omega_i$, $\mathbf{K} \sim \mathcal{N}(\mathbf{0}, D)$ is gaussian noise (capturing the assumption that the model may not be exactly correct) and $\mu \in \Re^d$ is the shift (in subspace $\Re^n$) vector. Though very naive, it has not stopped researchers and especially practitioners from using it successfully. The components $\omega_i$'s are extracted from a large database using eigenvalue analysis (such as eigenface analysis in image processing literature).

The graphical model representation for factor analysis is very simple; $\mathcal{V} = \{X_1, X_2, \ldots, X_d, \mathbf{Y}\}$ and $\mathcal{E} = \{(X_1, \mathbf{Y}), (X_2, \mathbf{Y}), \ldots, (X_d, \mathbf{Y})\}$. The $X_i$'s are marginally independent as indicated by $\mathcal{G}$. Further, we can infer that $E[\mathbf{Y}] = \mu$ and

$$\Sigma = E\left[(\mathbf{Y}-\mu)(\mathbf{Y}-\mu)^T\right] = E\left[(\Omega\mathbf{X}+\mathbf{K})(\Omega\mathbf{X}+\mathbf{K})^T\right] = \Omega\Omega^T + D$$

Finally, you can show that

$$\begin{pmatrix} \mathbf{X}_{d\times 1} \\ \mathbf{Y}_{n\times 1} \end{pmatrix} \sim \mathcal{N}\left( \begin{pmatrix} \mathbf{0}_{d\times 1} \\ \mu_{n\times 1} \end{pmatrix}, \begin{pmatrix} I_{d\times d} & \Omega^T_{d\times n} \\ \Omega_{n\times d} & \left(\Omega\Omega^T + D\right)_{n\times n} \end{pmatrix} \right)$$

In practice, it is useful to infer a distribution for $\mathbf{X}$ (distribution on weights for different eigenfaces) used to synthesize a particular observation $\mathbf{Y} = \widehat{\mathbf{y}}$ (such as a face image). That is, it can be required to infer $p(\mathbf{X} \mid \widehat{\mathbf{y}})$. Fortunately[29], this turns out to be a Guassian and this can be inferred using the Bayes rule and

[28]There is a lot of interest these days in identifying the manifold (especially non-linear subspaces, such as spheres) in which the data lies. In the classical technique of factor analysis, we make a very restrictive assumption that the data lies in some *linear subspace*.

[29]The conditional distribution need not always stay in the same family. But for Gaussians, the conditional distribution stays in the same family.

algebraic operations on matrices. In particular, the following property turns out to be useful. If

$$\begin{pmatrix} \mathbf{X} \\ \mathbf{Y} \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix}, \begin{pmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{pmatrix}\right)$$

then, $\mu_{\mathbf{X}|\mathbf{y}} = \mu_1 + \Sigma_{12}\Sigma_{22}^{-1}(\mathbf{y} - \mu_2)$, $Var(\mathbf{X} \mid \mathbf{y}) = \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}$ and $Var(\mathbf{X} \mid \mathbf{y}) \preceq \Sigma_{11} = Var(\mathbf{X})$. The last expression indicates that observation over $\mathbf{Y}$ should reduce uncertainity over $\mathbf{X}$.

## 7.6 Exponential Family and Graphical Models

We will now discuss the exponential family in the setting of graphical models. Particularly, we will discuss how to stitch together exponential models on graphs. Many, if not all graphical models take an exponential form. We will mainly discuss undirected graphs. Recall from (7.12), the standard factorization for an undirected graphical model

$$p(\mathbf{x}) = \frac{1}{Z}\prod_{\mathcal{C}\in\Pi}\phi_{\mathcal{C}}(\mathbf{x}_{\mathcal{C}})$$

where $\phi_{\mathcal{C}}$ is a compatibility or potential function and $\Pi$ is the set of all maximal cliques. All we do in exponential form is rewrite it in the log form to get an additive decomposition. This does not always hold, since will need that all $\phi_{\mathcal{C}}$ are non-negative in order to take their logs.

$$p(\mathbf{x}) = \exp\left\{\sum_{\mathcal{C}\in\Pi}\log\phi_{\mathcal{C}}(\mathbf{x}_{\mathcal{C}}) - \log Z\right\}$$

The above additive decomposition is essentially in exponential form, though not quite in the exponential family form (we still need to specify the canonical parameters). As an example, consider a Gaussian graphical model with $\mathbf{X} \sim \mathcal{N}(\mathbf{0}, \Omega)$ (canonical parametrization) defined on a tree $\mathcal{G} =< \mathcal{V}, \mathcal{E} >$ with noisy observations $\mathcal{Y} = C\mathcal{X} + \mathcal{V}$, $\mathcal{V} \sim \mathcal{N}(\mathbf{0}, R)$. Then the conditional distribution $p(\mathbf{X} \mid \mathbf{Y} = \mathbf{y})$ is also Gaussian, which can be decomposed according to the tree structure of $\mathcal{G}$.

$$p(\mathbf{x}|\mathbf{y}) \propto \prod_{s\in\mathcal{V}} p(y_s|x_s) \prod_{(s,t)\in\mathcal{E}} \exp\left\{-\frac{1}{2}[x_s\ x_t]J_{st}[x_s\ x_t]^T\right\}$$

where,

$$J_{st} = \begin{bmatrix} \Omega_{s(t)} & \Omega_{ts} \\ \Omega_{st} & \Omega_{t(s)} \end{bmatrix}$$

with $\Omega_{s(t)}$ specified so that $\Omega_{ss} = \sum_{t \in \mathcal{N}(s)} \Omega_{s(t)}$. In this representation, it is possible to do message passing using the canonical parameters in the sum/product updates (the basis of Kalman filters).

You could also have bernoulli random variables at every node (for example, to model the spread of a contagious disease in a social network of people) and one might be interested in building a probability distribution over the social network. Then, with every node $X_s \sim Ber(\lambda_s) \in \{0,1\}$ (canonical parametrization) and choosing[30] $\phi_{st} = \exp\{\lambda_{st} f_{st}(x_s, x_t)\}$ where, $f_{st} = x_s x_t + (1-x_s)(1-x_t)$ defined so that it takes value 1 iff $x_s = x_t$ (both are diseased or healthy) and is 0 otherwise, we have the following factorization:

$$p(\mathbf{x}) = \prod_{s \in \mathcal{V}} \exp\{\lambda_s x_s\} \prod_{(s,t) \in \mathcal{E}} \exp\{\lambda_{st} f_{st}(x_s, x_t)\} \tag{7.47}$$

If $\lambda_{st} = 0$, it indicates no coupling between the related individuals. For a reasonably contagious disease, we expect $\lambda_{st}$ to be non-negative atleast. The value of $\theta_s$ indicates the belief that an individual $X_s$ was intially diseased; higher the value, more will be the belief. Though not directly reflecting the marginal probability, $\lambda_s$ is an indicator of the log-odds ratio.

Let us consider another instance - a problem of counting the vertex coverings in a graph using the sum-product formalism. We will associate a random variable $X_i \in \{0,1\}$ with each node in the graph. Then, the following exponential form for $p(\mathbf{x})$ will serve our purpose:

$$p(\mathbf{x}, \eta) = \exp\left\{\eta \sum_{i=1}^{n} x_i - A(\eta)\right\} \prod_{(s,t) \in \mathcal{E}} (1 - (1-x_s)(1-x_t))$$

where $h(\mathbf{x}) = \prod_{(s,t) \in \mathcal{E}} (1 - (1-x_s)(1-x_t))$ ensures that we indeed have a vertex cover and $\mathbf{f}(\mathbf{x}) = \sum_{i=1}^{n} x_i$. It can be proved that as $\eta \rightarrow -\infty$ (which means you are increasingly penalizing larger coverings), the distribution goes to the minimum cardinality vertex covering (*i.e.*, $\sum_{i=1}^{n} x_i$).

If a graphical model were to be used to represent a language model for spell-checking or validity, *etc.* of an input string $\mathbf{x}$ of length upto $n$, you can have a substantially fewer number of parameters than if you were to have a naive potential over all 26 characters in an alphabet leading to a table of size $26^n$. This parameter reduction can be achieved by having indicator feature functions

[30] Remember that in a graphical model, we are free to choose the form of the potential functions so that they match the semantics. Here we pick edge potentials that reflect the coupling between health of related individuals, thereby shaping the distribution.

$\mathbf{f}(\mathbf{x})$ corresponding to interesting (and not all) substrings such as 'ion', 'ing'. To model such apriori information about 'striking patterns', it is useful to think of graphical models in the following *reduced parametrization* form, where feature function $f_\alpha$ can represent some such information as a feature function:

$$p(\mathbf{x}, \eta) = h(\mathbf{x}) \exp\left\{ \sum_{\mathcal{C} \in \Pi} \sum_{\alpha \in I_{\mathcal{C}}} \eta_\alpha f_\alpha(\mathbf{x}_{\mathcal{C}}) - A(\eta) \right\} \tag{7.48}$$

where $I_{\mathcal{C}}$ is the index for clique $\mathcal{C}$ so that $f_\alpha(\mathbf{x}_{\mathcal{C}})$ corresponds to only those interesting features that are local in terms of clique $\mathcal{C}$.

### 7.6.1 Exponential Models and Maximum Entropy

The data fitting principle of maximum entropy, which is suitable for learning models from data leads naturally to graphical models in exponential form and also gives nice semantics to the weights in the exponential family. There are many problems with constraints on distributions, but where the information is not complete. For instance, if we knew that for two binary random variables $X, Y \in \{0, 1\}$ and for their paired observations, 2/3 times $X$ takes value 0, 3/4 times, $Y$ takes value 1 and if you are asked, to specify the fraction of times $(X, Y) = (0, 0)$, what would you answer with the insufficient specification? Or going back to our diseased network model, if you are given observations on healths of similarly networked people and were to translate these observations into constraints on the moments (and/or the joint quantities), how would you determine the parameters of the probability distribution? There could be many distributions/parameters that are consistent with the observations. What principle should be adopted for making a good choice of the parameters/distribution?

More concretely, say you are given some (feature) functions $f_\alpha(\mathbf{x})$ with $\alpha \in I$ and are also given some observations $\widehat{\mu}_\alpha$ on the expected values of the functions, called *empirical moments*. The observations could come either from data or from physical constraints. We are interested in the family of distributions that are consistent with these constraints.

$$\mathcal{P}(\widehat{\mu}) = \left\{ p(.) \middle| \sum_{\mathbf{x}} p(\mathbf{x}) f_\alpha(\mathbf{x}) = \widehat{\mu_\alpha} \quad \forall \alpha \in I \right\}$$

Note that the family could be an empty set if the set of constraints are inconsistent. However, we will assume that the constraints are consistent. We are interested in choosing a particular $\widehat{p}$ from $\mathcal{P}(\mu)$ in a principled manner. Maximum entropy is one such principled method. Entropy is, roughly speaking, a measure of uncertainity or randomness. It has played a vital role in physics, chemisty, statistics, computer science, communication[31], *etc.*

[31] Entropy plays a fundamental role in deciding how far you could compress a sequence of bits.

**Definition 57** *The entropy $H(p)$ of the distribution $p$ on a random variable (vector) $\mathbf{X}$ is given by the expected value of the log of the distribution. Depending on whether the random variable (vector) is continuous or discrete, we will have two different definitions of expectation (and therefore for the entropy). For discrete random variable (vector) $\mathbf{X}$,*

$$H(p) = -\sum_{\mathbf{x}} p(\mathbf{x}) \log p(\mathbf{x})$$

*whereas, for continuous valued random variable $\mathbf{X}$,*

$$H(p) = -\int_{\mathbf{x}} p(\mathbf{x}) \log p(\mathbf{x}) d\mathbf{x}$$

It can be easily proved that $H(p) \geq 0$ (convention being that $0 \log 0 = 0$). $H(p) = 0$ if and only if $X$ is deterministically always some $a$, making $p(a) = 1$ and $p(x) = 0$, $\forall x \neq a$. $H(p)$ is maximal for the uniform distribution.

The principle of maximum entropy can be now defined:

**Definition 58** *Given* $\mathcal{P}(\widehat{\mu}) = \left\{ p(.) \left| \sum_{\mathbf{x}} p(\mathbf{x}) f_\alpha(\mathbf{x}) = \widehat{\mu_\alpha} \quad \forall \alpha \in I \right. \right\}$, *choose*

$$\widehat{p} = \underset{p \in \mathcal{P}(\widehat{\mu})}{\text{argmax}} H(p)$$

The intuition here is to balance across the constraints. It is also called the *principle of least commitment* since the goal is to simultaneously respect the data and not commit to anything more.

**Theorem 99** *The maximum entropy solution exists and is unique. The unique solution takes the form*

$$p(\mathbf{x}) \propto \exp \left\{ \sum_{\alpha \in I} \eta_\alpha f_\alpha(\mathbf{x}) \right\} \tag{7.49}$$

*Proof:* The first thing to note is that $\mathcal{P}(\widehat{\mu})$ is a convex[32] (linear), closed[33] and bounded set of distributions. Further, $H(p)$ is continuous, and this, in conjunction with the nature of $\mathcal{P}(\widehat{\mu})$, guarantees that a maximum will be attained. Also, $H(p)$ is strictly concave[34], implying that the maximum will be unique.

The canonical parameters are actually the Langrange multipliers. Consider the Lagrangian $L(p, \eta, \lambda)$:

$$L(p, \eta, \lambda) = H(p) + \sum_{\alpha \in I} \eta_\alpha \left[ \widehat{\mu_\alpha} - \sum_{\mathbf{x}} p(\mathbf{x}) f_\alpha(\mathbf{x}) \right] + \lambda [1 - \sum_{\mathbf{x}} p(\mathbf{x})]$$

[32]What if the set is empty?

[33]Which means the optimum cannot escape to the boundary.

[34]Since $\frac{\partial^2 H}{\partial p^2} = -\frac{1}{p} < 0$.

The KKT necessary and sufficient conditons (see (4.88) on page 296) for optimality of $(\widehat{p(\mathbf{x})}, \widehat{\eta}, \widehat{\lambda})$ yield:

1. $\frac{\partial L}{\partial p} = 0 \Rightarrow \log \widehat{p(\mathbf{x})} - \sum_{\alpha \in I} \widehat{\eta_\alpha} f_\alpha(\mathbf{x}) - \widehat{\lambda} = 0$. That is,

$$\widehat{p(\mathbf{x})} = \exp\left\{\sum_{\alpha \in I} \widehat{\eta_\alpha} f_\alpha(\mathbf{x})\right\} e^{\widehat{\lambda}} \tag{7.50}$$

2. $\sum_{\mathbf{x}} \widehat{p(\mathbf{x})} = 1$. Substituting for $p(\mathbf{x})$ from (7.50), we get

$$e^{\widehat{\lambda}} = \frac{1}{\sum_{\mathbf{x}} \exp\left\{\sum_{\alpha \in I} \widehat{\eta_\alpha} f_\alpha(\mathbf{x})\right\}}$$

   which is a constant.

This proves that $p(\mathbf{x}) \propto \exp\left\{\sum_{\alpha \in I} \eta_\alpha f_\alpha(\mathbf{x})\right\}$. $\square$

This result shows how exponential families can arise in a fairly natural way. It can be further shown that a slight generalization of the maximum entropy principle, called the *Kullkack-Leibler divergence* very naturally leads to the maximum likelihood principle.

**Definition 59** *The Kullkack-Leibler (KL) divergence $D(p||q)$ between two distributions $p(.)$ and $q(.)$ is given by the expectation over $p(.)$ of the log-likelihood ratio of $p(.)$ over $q(.)$.*

$$D(p||q) = \sum_{\mathbf{x}} p(\mathbf{x}) \log \frac{p(\mathbf{x})}{q(\mathbf{x})}$$

*For distributions $p(.)$ and $q(.)$ over continuous valued random variables*

$$D(p||q) = \int_{\mathbf{x}} p(\mathbf{x}) \log \frac{p(\mathbf{x})}{q(\mathbf{x})} d\mathbf{x}$$

Like any distance, the KL divergence is always non-negative. Also, $p \equiv q$ if and only if $D(p||q) = 0$. However, by inspection, we can infer that $D(p||q)$ is assymetric, unlike metrics. That is $D(p||q) \neq D(q||p)$. If $q(.)$ were the uniform distribution $D(p||q) = H(p) + c$, where $c$ is some constant.

We will next define a slight generalization of the maximum entropy principle. Recall the definition of $\mathcal{P}(\widehat{\mu})$. Now let $q$ be some additional prior or reference distribution. The generalized definition goes as:

**Definition 60** *Given* $\mathcal{P}(\widehat{\mu}) = \left\{ p(.) \middle| \sum_{\mathbf{x}} p(\mathbf{x}) f_\alpha(\mathbf{x}) = \widehat{\mu_\alpha} \quad \forall \alpha \in I \right\}$ *and a reference distribution* $q(\mathbf{x})$*, choose*

$$\widehat{p} = \underset{p \in \mathcal{P}(\widehat{\mu})}{\operatorname{argmin}} D(p||q)$$

The interpretation here is: get as close as possible to the reference distribution, while respecting the data in the form of the constraint set $\mathcal{P}(\widehat{\mu})$. For the maximum entropy principle, the reference distribution happened to be the uniform distribution. Note that we have an 'argmin' here instead of an 'argmax'.

**Theorem 100** *The solution to the minimum KL divergence problem in definition 60 exists and is unique. The unique solution takes the form*

$$p(\mathbf{x}) \propto q(\mathbf{x}) \exp\left\{ \sum_{\alpha \in I} \eta_\alpha f_\alpha(\mathbf{x}) \right\}$$

*The reference distribution* $q(\mathbf{x})$ *plays the role of the function* $h(\mathbf{x})$ *in the exponential form.*

*Proof:* The proof of existence and uniqueness here are the same as that for the counterpart in the proof of theorem 99. The only change will be the KKT necessary and sufficient conditions for optimality of $(\widehat{p(\mathbf{x})}, \widehat{\eta}, \widehat{\lambda})$ (see (4.88) on page 296).

Consider the Lagrangian $L(p, \eta, \lambda)$:

$$L(p, \eta, \lambda) = D(p||q) + \sum_{\alpha \in I} \eta_\alpha \left[ \widehat{\mu_\alpha} - \sum_{\mathbf{x}} p(\mathbf{x}) f_\alpha(\mathbf{x}) \right] + \lambda[1 - \sum_{\mathbf{x}} p(\mathbf{x})]$$

The KKT necessary and sufficient conditons yield:

1. $\frac{\partial L}{\partial p} = 0 \Rightarrow \log \widehat{p(\mathbf{x})} - \sum_{\alpha \in I} \widehat{\eta_\alpha} f_\alpha(\mathbf{x}) - \widehat{\lambda} - \log q(\mathbf{x}) = 0$. That is,

$$\widehat{p(\mathbf{x})} = q(\mathbf{x}) \exp\left\{ \sum_{\alpha \in I} \widehat{\eta_\alpha} f_\alpha(\mathbf{x}) \right\} e^{\widehat{\lambda}} \tag{7.51}$$

2. $\sum_{\mathbf{x}} \widehat{p(\mathbf{x})} = 1$. Substituting for $p(\mathbf{x})$ from (7.50), we get

$$e^{\widehat{\lambda}} = \frac{1}{\sum_{\mathbf{x}} q(\mathbf{x}) \exp\left\{ \sum_{\alpha \in I} \widehat{\eta_\alpha} f_\alpha(\mathbf{x}) \right\}}$$

which is a constant.

This proves that $p(\mathbf{x}) \propto q(\mathbf{x}) \exp\left\{ \sum_{\alpha \in I} \eta_\alpha f_\alpha(\mathbf{x}) \right\}$. $\square$

## 7.7 Model fitting

Thus far we have discussed graphical models and inference in these models. But how do learn the potential functions associated with a model or the canonical parameters associated with the exponential form, given some data from the real world that describes the phenomena? This opens up a new interesting fundamental question: 'how to infer models from data?'. We will also need a measure of how 'good' a model is and this is often driven by the end goal. These discussions forms the topic of discussion for this section.

Consider a coin tossing experiment, in which $X \sim Ber(\mu)$ where $\mu = \Pr(X = 1)$. Let us say we observe a sequence of coin tosses: $[x^{(1)}, x^{(2)}, \ldots, x^{(m)}]$. Can we use this data to infer something about $\mu$? There are two primary ways to do this.

1. The Bayesian school of thought views the Bayes rule as the fundamental method for inventing model from the data:

$$p(\theta \mid \overline{\mathbf{x}}) = \frac{p(\overline{\mathbf{x}} \mid \theta)p(\theta)}{p(\overline{\mathbf{x}})}$$

where $\theta$ is the underlying model parameter, $p(\overline{\mathbf{x}} \mid \theta)$ is the likelihood term, $p(\theta \mid \overline{\mathbf{x}})$ is the posterior that summarizes the remaining uncertainty in $\theta$, given that you have observed the data $\overline{\mathbf{x}}$. This simple statement has consequences on the way you might view parameters; by imposing a distribution over $\theta$, you are encoding the belief that the parameter itself is a random quantity. This requires viewing all parameters as random variables. The Bayesian technique views $\mu$ as a random quantity following a $Beta(a, b)$ distribution[35] on $[0, 1]$.

$$\mu \sim Beta(a, b) \propto \mu^{a-1}(1 - \mu)^{b-1}$$

The Bayesian tries to model the inherent uncertainity in the statistician about the value of the parameter ($\mu$). The Bayesian thus has a more or less fixed procedure for folding in the data into the model.

2. The frequentist's way of measuring probabilities is as numbers measured as outcomes over repeated trials as against the subjective notion of probability adopted by the Bayesians. The frequentist would object to the Bayesian view on several counts: (i) the subjectivity of the procedure; is there a sound justification for the choice of $\mu \sim Beta(a, b)$ and for the particular choice of $a$ and $b$? (ii) that different results for the same data set could be obtained by different statisticians adhering to different choices of the prior. The frequentist belief is that one should be completely objective with the data; the data should speak so that you get the same model, no matter who builds the model. It does not make sense for the probability

[35]The shape of the Beta distribution depends on the value of the parameters - it could be either uniform or bell-shaped.

of a coin throwing up heads to be a random variable. However, the frequentist does not have any fixed procedure for inventing a model from the data. The frequentist thus considers several estimators for the parameters. One estimator is the sample mean of the data. Any estimator is assessed for its properties such as bias, variability, *etc.* Variability asks: How much would the estimated value vary if the data sample changed? How will the estimate behave if we repeated the experiement several times?

Both Bayesian and frequentist views are compatible with graphical models. Also, the two views agree that there could be different models for different (coin-tossing) experiments. The Bayesian view often gives you a good way of generating good procedures (that is, procedures with good properties) whereas the frequentist view often gives you a very good way of evaluating a procedure.

### 7.7.1 Sufficient Statistics

Let $\mathbf{X}$ be a random variable. In the present discussion, we will often assume it to correspond to a vector of observations (that is data). Any function $\tau(\mathbf{X})$ is called a statistic. From the Bayesian point of view, a statistic $\tau(\mathbf{X})$ is *sufficient* for the parameter variable $\theta$ if $\theta \perp \mathbf{X} \mid \tau(\mathbf{X})$. This relation can be expressed graphically as a markov chain with $\mathcal{V} = \{\mathbf{X}, \tau(\mathbf{X}), \theta\}$, $\mathcal{E} = \{(\mathbf{X}, \tau(\mathbf{X})), (\tau(\mathbf{X}), \theta)$. Intuitively, this means that the function of the observations $\tau(\mathbf{X})$ is what is essential in data to explain the characteristics of $\theta$ and that all the dependence between $\mathbf{X}$ and $\theta$ is being mediated by $\tau(\mathbf{X})$. $\tau(\mathbf{X})$ is typically much smaller than $\mathbf{X}$ itself.

From the frequentist point of view, $\theta$ is an unknown fixed parametrization that we would like to estimate. The sufficiency for a frequentist is that

$$p(\mathbf{x} \mid \tau(\mathbf{x}); \theta) = p(\mathbf{x} \mid \tau(\mathbf{x}))$$

That is, the family of distributions specified by the parametrization $\theta$ becomes independent of $\theta$ when conditioned on the sufficient statistic $\tau(\mathbf{x})$; it indicates that we have conditioned on sufficient information to capture any information from $\theta$.

A view of sufficiency unified across the Bayesian and frequetist perspectives can be obtained by treating it as a statement about factorization:

$$p(\mathbf{x}, \tau(\mathbf{x}), \theta) = \phi_1(\tau(\mathbf{x}), \theta)\phi_2(\tau(\mathbf{x}), \mathbf{x})$$

It can be proved that this is indeed a unified view of sufficiency, consistent with the Bayesian and frequentist views. Though there is a difference in interpretations, it amounts to the same factorization. Note the similarity of the factorization here with that for general graphical models. As we will see, for the graphical model family, sufficiency properties can be read out, just on the basis of their factorization. Further, the sufficiency property stands our clearly for exponential family. Given iid data[36] $\overline{\mathbf{x}} = (\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(n)})$ each from some

[36]This assumption is common-place because it makes computations very simple and decomposable. It is often not the case, since there could be dependence in the sampling.

exponential family, and given an $\eta$ (which corresponds to the $\theta$ being discussed thus far in the context of model estimation), we can easily infer that

$$p(\overline{\mathbf{x}}_{1,m};\eta) = \prod_{i=1}^{m} p(\mathbf{x}^{(i)};\eta) = \underbrace{\left(\prod_{i=1}^{m} h(\mathbf{x}^{(i)})\right)}_{\phi_2(\tau(\mathbf{x}),\mathbf{x})} \underbrace{\exp\left\{\eta^T \left(\sum_{i=1}^{m} \mathbf{f}(\mathbf{x}^{(i)})\right) - mA(\eta)\right\}}_{\phi_1(\tau(\mathbf{x}),\eta)}$$

This algebra tells us that the quantity $\tau(\overline{\mathbf{x}}) = \sum_{i=1}^{m} \mathbf{f}(\mathbf{x}^{(i)})$ is the sufficient statistic. Note that $\phi_1$ can potentially depened on $\tau(\mathbf{x})$, though it does not depend in this case. The key to estimating $\eta$ are the sufficient statistics $\tau(\overline{\mathbf{x}})$. We will also realize that the prefactor $h(\overline{\mathbf{x}})$ does not play a significant role here.

The quantity

$$\overline{\mu} = \frac{1}{m}\sum_{i=1}^{m} \mathbf{f}(\mathbf{x}^{(i)}) = \sum_{\mathbf{x}} \wp(\mathbf{x})\mathbf{f}(\mathbf{x})$$

is called the *empirical moment parameter* vector, where the empirical distribution $\wp(\mathbf{x})$ is obtained by placing point mass at data points.

$$\wp(\mathbf{x}) = \frac{1}{m}\sum_{i=1}^{m} \delta(\mathbf{x} - \mathbf{x}^{(i)})$$

The empirical moment parameters are a special type of empirical moments discussed on page 439. Note that the empirical moment parameters are simply the sufficient statistics scaled down by the number of data points. The empirical moment parameter vector can be contrasted against the moment parameter vector $\mu = [E[f_1(\mathbf{X})], E[f_2(\mathbf{X})], \ldots, E[f_k(\mathbf{X})]]$ for exponential families, as defined on page 433. The two differ in the probability distributions with respect to which they compute their expectations; while the former computes with respect to the empirical distribution, the latter computes with respect to the true distribution.

As an example, consider the Ising model, defined on a graph $\mathcal{G} = <\mathcal{V},\mathcal{E}>$, which is a member of the exponential family. It is distribution is very similar to that of the disease network model (7.47), only difference is that only the first term $x_s x_t$ is retained in the definition of $f_{st}(\mathbf{x})$.

$$p(\mathbf{x},\eta) \propto \exp\left\{\sum_{s\in\mathcal{V}} \eta_s x_s + \sum_{(s,t)\in\mathcal{E}} \eta_{st} x_s x_t\right\}$$

By appropriately identifying the feature function vector $\mathbf{f}$ and $\eta$ of size $|\mathcal{V}| + |\mathcal{E}|$ each, we can determine the empirical moments to be

$$\widehat{\mu}_s = \frac{1}{m}\sum_{i=1}^{m} x_s^i$$

and

$$\widehat{\mu}_{st} = \frac{1}{m}\sum_{i=1}^{m} x_s^i x_t^i$$

which are just the marginal counts.

What about empirical moments for the reduced parametrization discussed earlier in (7.48) on page 439?

$$p(\mathbf{x}, \eta) = h(\mathbf{x}) \exp\left\{\sum_{\mathcal{C}\in\Pi}\sum_{\alpha\in I_{\mathcal{C}}} \eta_\alpha f_\alpha(\mathbf{x}_{\mathcal{C}}) - A(\eta)\right\}$$

Given iid data $\overline{\mathbf{x}} = \left(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(m)}\right)$, the following factorization holds if each $\mathbf{x}^{(i)}$ follows the distribution specified by the reduced parametrization. Since the apriori information is captured using indicative feature functions, you can have a reduced set of sufficient statistics.

$$p(\overline{\mathbf{x}}_{1,m}; \eta) = \prod_{i=1}^{m} p(\mathbf{x}^{(i)}; \eta) = \underbrace{\left(\prod_{i=1}^{m} h(\mathbf{x}^{(i)})\right)}_{\phi_2(\tau(\mathbf{x}),\mathbf{x})} \underbrace{\exp\left\{\sum_{\alpha} \eta_\alpha \left(\sum_{i=1}^{m}\sum_{\mathcal{C}\in\Pi|\alpha\in I_{\mathcal{C}}} f_\alpha(\mathbf{x}^{(i)})\right) - mA(\eta)\right\}}_{\phi_1(\tau(\mathbf{x}),\eta)}$$

The form of the empirical moments in this case pools over all cliques that are relevant for a particular feature type $f_\alpha$ (for instance, in a grid, some features may be commonly defined/parametrized across all vertical edges while others across all horizontal edges) and then pools over all data:

$$\widehat{\mu}_\alpha = \sum_{i=1}^{m}\sum_{\mathcal{C}\in\Pi|\alpha\in I_{\mathcal{C}}} f_\alpha(\mathbf{x}^{(i)})$$

The reduced parametrization assumes some kind of homogenity in the model. More precisely, it makes the statistical assumption that the parameter $\eta_\alpha$ is independent of the exact position in $\mathcal{G}$ and is determined by the local graphical structure. This parametrization assuming homogenity is often desirable since it gets a more stable estimator with less variance even using relatively less data. In fact, the reduced parametrization yields a new exponential family with much lower dimensions, in which the parameters $\eta_\alpha$ enter in a purely linear way[37]. The issues of when and how to pool data are important ones in modeling.

## 7.7.2 Maximum Likelihood

The likelihood function interests both Bayesian and frequentists alike. Recall that likelihood $p(\overline{\mathbf{x}} \mid \theta)$ was one part of the Bayes rule. The frequentist interpretation of this quantity is as 'the likelihood of a fixed $\overline{\mathbf{x}}$ as $\theta$ varies' whereas the Bayesian interpretation of this quantity is as 'the conditional probability of

[37]A widely used different class of exponential families is called *curved exponential families*, in which the parameters $\eta_\alpha$ enter in non-linear ways.

a varying $\overline{\mathbf{x}}$ given a fixed $\theta$.' The frequentist thus views likelihood as a function of theta $L(\theta)$ that measures, in some sense, the goodness of $\theta$ as it is varied for the particular sample of data in hand. It is often useful to talk about the log-likelihood instead[38].

**Definition 61** *Given a sample $\overline{\mathbf{x}} = \left\{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(m)}\right\}$, the log-likelihood (LL) is defined as*

$$LL(\theta; \overline{\mathbf{x}}) = \frac{1}{m} \log p(\overline{\mathbf{x}} \mid \theta)$$

*Note that the $\frac{1}{m}$ term does not really change the optimization and its usage is conventionally just to normalize the log-likelihood.*

The maximum likelihood estimate (MLE) is defined next.

**Definition 62** *Given a sample $\overline{\mathbf{x}} = \left\{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(m)}\right\}$, the maximum likelihood estimate (MLE) $\widehat{\theta}_{MLE}$ of $\theta$ is defined as*

$$\widehat{\theta}_{MLE} = \underset{\theta}{\operatorname{argmax}} \; LL(\theta; \overline{\mathbf{x}})$$

*As the name suggests, this principle treats log-likelihood as a measure of goodness of the parameter and picks that value which maximizes the LL. Though not against Bayesian principles, MLE has been of greater interest to the frequentists. The estimate $\widehat{\theta}_{MLE}$ is called a point estimate, since the method gives you just a single point.*

Let us try to fit (that is, adjust the parameters of) a scalar Gaussian $\mathcal{N}(\mu, \sigma^2)$ to some data $\left\{x^{(1)}, x^{(2)}, \ldots, x^{(m)}\right\}$ using the MLE principle. We will assume that the data is iid. This will mean that the joint distribution over the data will factorize into individual data instances, given the parameters.

$$LL(\mu; \overline{\mathbf{x}}) = \frac{1}{m} \sum_{i=1}^{m} \log p(x^{(i)} \mid \mu) = -\frac{1}{2m} \log 2\pi - \frac{1}{2m} \sum_{i=1}^{M} (x^{(i)} - \mu)^2$$

The LL is a quadratic in $\mu$ (that is $\theta$), which achieves its maximum at

$$\widehat{\mu}_{MLE} = \frac{1}{m} \sum_{i=1}^{m} x^{(i)}$$

as expected. The data determines the shape of the likelihood and MLE looks for the point $\widehat{\mu}_{MLE}$ that maximizes the LL.

This example of parameter estimation for the simple Gaussian is one of the motivations for using MLE; MLE corresponds to an intuitively appealing estimation for the mean parameter. Secondly MLE has good asymptotic properties.

[38]Since it is a monotonic transformation between the likelihood and log-likelihood, it does not really matter much whether we look at the former or the latter. It is usually more convenient to look at the log-likelihood (LL).

A frequentist takes more interest in the asymptotic properties of the estimator. We can look upon $\widehat{\theta}_{MLE}$ as a random variable, since the data $\overline{\mathbf{x}}$ was itself random and $\widehat{\theta}_{MLE}$ is a function of the data. As the amount of data grows, any reasonable estimator should behave 'well'. One yardstick of good behaviour of the estimator is *consistency*, that is, if there was some true underlying $\theta^*$, we would like

$$\widehat{\theta}_{MLE} \overset{m\to\infty}{\to} \theta^*$$

Under most conditions, MLE estimators are consistent. Generally, it can be worrisome if the estimator does not converge to the true answer even with infinite amount of data.

Another yardstick of good behaviour is that asymptotically, the estimator's spread around the true value of the parameter must be Gaussian-like. MLE estimators honor this feature as well; with increasing $m$, the distribution of $\widehat{\theta}_{MLE}$ tends to be Gaussian with the true mean $\theta^*$ and the spread $\Sigma$ given by the Fisher information matrix[39].

In small sample regimes (that is, if you do not have too much data), MLE does not behave well. In such settings, the frequentist adds a penalty term, called the *regularization* term (such as $\mu^2$ in the above example) to the likelihood objective to somehow prevent the estimator from drifting away based on a small $m$.

## 7.7.3 What the Bayesians Do

The Bayesians focus more on the posterior $p(\theta \mid \overline{\mathbf{x}})$, which is the likelihood multiplied by a prior.

$$p(\theta \mid \overline{\mathbf{x}}) \propto p(\overline{\mathbf{x}} \mid \theta)p(\theta)$$

For example, let $x^{(i)} \sim \mathcal{N}(\mu, 1)$ for $1 \leq i \leq m$ and let $\mu \sim \mathcal{N}(\nu, \sigma^2)$. The parameter $\mu$ is a random variable, whereas the *hyperparameters* $\mu'$ and $\sigma$ are fixed. The choice of the values $\mu'$ and $\sigma$ could significantly influence the posterior. There is a class of models[40] called the 'hierarchical Bayes models' that put a prior on the hyper-parameters, priors again on the hyper-hyper-parameters and so on. It stops (for pragmatic reasons) when the abstration has been sufficiently performed so that it matters little what the fixed values of the hyper$^n$-parameters are.

The posterior for this example, takes the form

$$p(\theta \mid \overline{\mathbf{x}}_{1,m}) \propto \exp\left\{-\frac{1}{2}\sum_{i=1}^{m}(x^{(i)} - \mu)^2\right\} \exp\left\{-\frac{1}{2\sigma^2}(\mu - \nu)^2\right\}$$

The Bayesian is not interested just in the point estimate, but moreso in the entire posterior distribution. In this case (a special instance of Kalman filters) the posterior again turns out to be a Gaussian; $\mu_{\overline{\mathbf{x}}_{1,m}} \sim \mathcal{N}\left(E(\mu \mid \overline{\mathbf{x}}_{1,m}), Var(\mu \mid \overline{\mathbf{x}}_{1,m})\right)$.

[39]Related to this is the Cramer-Rao lower bound.

[40]The broader area is called *Robust Bayesian statistics*.

A little calculation should convince you that $\mu_{\overline{\mathbf{x}}_{1,m}} \sim \mathcal{N}\left(\frac{m\sigma^2}{m\sigma^2+1}\widehat{\mu}_{MLE} + \frac{1}{m\sigma^2+1}\nu, Var(\mu \mid \overline{\mathbf{x}}_{1,m})\right)$. How does the posterior behave as $m \rightarrow \infty$? We can easily see that with increasing amount of data, the likelihood will completely swamp the prior. That is, $E(\mu \mid \overline{\mathbf{x}}_{1,m}) \overset{m\rightarrow\infty}{\rightarrow} \widehat{\mu}_{MLE}$. So the choice of the hyper-parameters is irrelevant as the size of the data goes to $\infty$. Thus, MLE behaves well asymptotically even from the Bayesian point of view. Unlike the frequentist, the Bayesian does not need to explicitly handle the small sample regimes.

In general, the posterior might not have a finite parametrization. In such cases, the Bayesians do compute point estimates. Examples of point estimates computed by Bayesians are:

1. *Bayes estimate:* This is the mean of the posterior:
$$\widehat{\theta}_{Bayes} = \int \theta p(\theta \mid \overline{\mathbf{x}}_{1,m})d\theta$$
2. *MAP estimate:* This is the mode of the posterior:
$$\widehat{\theta}_{MAP} = \underset{\theta}{\text{argmax}}\ \frac{1}{m}\log\{p(\theta \mid \overline{\mathbf{x}}_{1,m})\}$$
The MAP estimate is very much related to the MLE. Invoking the Bayes rule and ignoring the term involving $\log p(\overline{\mathbf{x}}_{1,m})$,
$$\begin{aligned}\widehat{\theta}_{MAP} &= \underset{\theta}{\text{argmax}}\ \frac{1}{m}\log\{p(\overline{\mathbf{x}}_{1,m} \mid \theta)\} + \frac{1}{m}\log\{p(\theta)\} \qquad (7.52)\\ &\underset{\text{iid}}{=} \underset{\theta}{\text{argmax}}\ \frac{1}{m}\sum_{i=1}^{m}\log\left\{p(x^{(i)} \mid \theta)\right\} + \frac{1}{m}\log\{p(\theta)\}\end{aligned}$$
The first term in the decomposition is the data likelihood term, while the second term is the prior term. Let us study this decomposition.
   (a) This decomposition suggests that if the prior is uniform, $\widehat{\theta}_{MAP} = \widehat{\theta}_{MLE}$.
   (b) Secondly, in the asymptotic case, as $m \rightarrow \infty$, the prior component fades away (since it has no dependence on $m$ in the numerator) in contrast to the likelihood term[41]. In fact, as $m \rightarrow \infty$, $\widehat{\theta}_{MAP} \rightarrow \widehat{\theta}_{MLE}$.
   (c) If the prior is assumed to be Gaussian, then the prior term will assume the form of a quadratic penalty. This is the same as a quadratic regularization term for MLE. With different choices of priors on $\theta$, you get different regularizations. For example, a Laplacian prior on $\theta$ results in $L1$ regularization. For example, Lasso is L1 regularization for a regression problem. How the choice of regularization affects these estimators is an area of active research.

[41]One could use the central limit theorem or law of large numbers to study the behaviour of the likelihood term as $m \rightarrow \infty$.

### 7.7.4 Model Fitting for Exponential Family

We will initiate the discussion of model fitting in exponential families by looking at maximum likelihood (ML) estimation. Just as for most other properties of exponential models, there is something crisper that we can state about the ML properties of exponential models. The optimization conditions will turn out to take special and easily interpretable forms.

Recall the specification of any member of the exponential family from (7.43).

$$p(\mathbf{x} \mid \eta) = h(\mathbf{x}) \exp\left\{\eta^T \mathbf{f}(\mathbf{x}) - A(\eta)\right\}$$

We will now see how the empirical moment parameters, defined on page 445 and discussed subsequently become relevant for parameter estimation. Given iid observations $\overline{\mathbf{x}} = \left\{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(n)}\right\}$, the normalized LL for this distribution decouples as a sum

$$\begin{aligned}
LL(\eta; \overline{\mathbf{x}}) &= \frac{1}{m} \log p(\overline{\mathbf{x}} \mid \eta) && (7.53)\\
&= \frac{1}{m}\left\{\eta^T \sum_{i=1}^{m} \mathbf{f}(\mathbf{x}^{(i)}) - mA(\eta)\right\} + \log h(\overline{\mathbf{x}}) && (7.54)\\
&= \left\{\frac{1}{m}\eta^T \sum_{i=1}^{m} \mathbf{f}(\mathbf{x}^{(i)})\right\} - A(\eta) + \log h(\overline{\mathbf{x}}) && (7.55)\\
&= \left\{\eta^T \overline{\mu}(\overline{\mathbf{x}})\right\} - A(\eta) + \log h(\overline{\mathbf{x}}) && (7.56)
\end{aligned}$$

where $\overline{\mu}(\overline{\mathbf{x}}) = \frac{1}{m}\sum_{i=1}^{m} \mathbf{f}(\mathbf{x}^{(i)})$ is the vector of empirical moment parameters. Since $h(\overline{\mathbf{x}})$ is not a function of the parameter (and instead, is a constant offset), it follows that the MLE for the exponential family takes the form

$$\widehat{\mu}_{ML} \in \underset{\eta \in Domain(\eta)}{\text{argmax}} \left\{\eta^T \overline{\mu}(\overline{\mathbf{x}}) - A(\eta)\right\}$$

This is a very crisp formulation of sufficiency; given data, all you need to compute are $\overline{\mu}(\overline{\mathbf{x}})$, forget the data $\overline{\mathbf{x}}$ and focus on solving the optimization problem. Since $A(\eta)$ is infinitely differentiable and since the remainder of the objective is linear, we could make use of the first order necessary optimality conditions from (4.44) on page 270:

$$\nabla LL(\eta; \overline{\mathbf{x}}) = 0$$

That is,

$$\overline{\mu}(\overline{\mathbf{x}}) - \nabla A(\eta) = \overline{\mu}(\overline{\mathbf{x}}) - E(\mathbf{f}(\mathbf{X})) = 0$$

where, we may recall from (7.44) on page 433 that $\nabla A(\eta)$ is the vector of moments predicted by the model. This important condition

$$\overline{\mu}(\overline{\mathbf{x}}) = E(\mathbf{f}(\mathbf{X})) \tag{7.57}$$

is called the *moment matching condition*[42] and it states that for maximizing the likelihood, we need to match the empirical moments to the model moments. In general, these coupled $d$ equations are complex and non-linear ones, that are not very easy to solve, though there are some exceptions. The necessary conditions suggest that the parameters should be learnt in such a way that if you drew samples from the model, they should look like the given data.

As example, let $X$ be a bernoulli random variable. Recollect the specification of the bernoulli distribution as a member of exponential family from page 7.4: $\eta = \log \frac{\mu}{1-\mu}$, $f(x) = x$ and $A(\eta) = \log\{1 + e^{\eta}\}$. This leads to the following moment matching conditions

$$\overline{\mu} = \frac{1}{m}\sum_{i=1}^{m} x^{(i)} = A'(\eta) = \frac{e^{\eta}}{1+e^{\eta}}$$

The empirical moments is the fraction of heads that come up in the $m$ coin tosses of the experiment. The expression for $\eta$ will be

$$\eta = \log\left(\frac{\overline{\mu}}{1-\overline{\mu}}\right)$$

which corresponds to the logs-odd ratio. When does this have a solution? It has a solution if $\overline{\mu} \in (0,1)$ but not if $i = 1$ or $i = 0$, which can happen if all the coin tosses landed up with heads or tails[43]. If $\overline{\mu} \in (0, \frac{1}{2})$, $\eta < 0$ and if $\overline{\mu} \in (\frac{1}{2}, 1)$, $\eta > 0$. So if $\overline{\mu} = 1$, strictly speaking the MLE does not exist. Note that if we were to use the original formulation of the bernoulli distribution on page 432, moment matching would have yielded the MLE as $\mu = \overline{\mu}$. For large amounts of data, using the weak law of large numbers, you can be convinced that this loss can be recovered using even the exponential formulation.

As another example, let us revist the multivariate normal distribution in the canonical form first uncovered in (7.46):

$$p(\mathbf{x}, \eta, \Omega) = \exp\left\{\eta^T\mathbf{x} - \frac{1}{2}trace(\Omega\mathbf{x}\mathbf{x}^T) + \mathbf{x}\right\}$$

where

$$\eta = \Sigma^{-1}\mu$$

and

$$\Omega = \Sigma^{-1}$$

[42]This is related to moment matching rooted in classical statistics. Though not related directly to maximum likelihood, these two boil down to the same criteria for the exponential family.

[43]This is not impossible, especially if the tossing is done by some magician such as Persi Warren Diaconis or by a machine built by him.

Moment matching will yield the model first order and second order moments in terms of the empirical mean and the empirical second order moment matrix respectively. That is,

$$E[\mathbf{x}] = \mu = \frac{1}{m}\sum_{i=1}^{m}\mathbf{x}^{(i)} = \overline{\mu}$$

and

$$E[\mathbf{x}\mathbf{x}^T] = \Sigma + \mu\mu^T = \frac{1}{m}\sum_{i=1}^{m}\mathbf{x}^{(i)}\left(\mathbf{x}^{(i)}\right)^T$$

Solving this system, we get $\Omega_{ML}$ as the inverse of the sample covariance matrix

$$\Omega_{ML} = \left(\frac{1}{m}\sum_{i=1}^{m}\mathbf{x}^{(i)}\left(\mathbf{x}^{(i)}\right)^T - \overline{\mu}\mu^T\right)^{-1}$$

(which exists if the sample covariance matrix has full rank) and

$$\eta_{ML} = \Omega_{ML}\overline{\mu}$$

In practice (such as in text mining problems), it can happen that many features are never observed and consequently, the corresponding empirical moment parameters will be 0. Similarly, for the multivariate Gaussian example, if you do not have sufficient data, the sample covariance matrix may not have full rank. More precisely, we need to pay heed to $Domain(\eta)$, since the optimal values could reside on the boundary. This is addressed using regularization through a penalized log-likelihood objective.

### 7.7.5 Maximum Likelihood for Graphical Models

For general large graphical models, the moment matching is not at all easy to solve. It has therefore been a practice to often resort to iterative algorithms for solving the set of moment matching equations. We will illustrate this difficulty through the Ising model (*c.f.* page 445), which is a model on an undirected graph $\mathcal{G} = <\mathcal{V}, \mathcal{E}>$.

$$p(\mathbf{x}, \eta) = \exp\left\{\sum_{s\in\mathcal{V}}\eta_s x_s + \sum_{(s,t)\in\mathcal{E}}\eta_{st}x_s x_t - A(\eta)\right\}$$

Let us say we have iid data $\overline{\mathbf{x}} = (\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(m)})$. The empirical moments are

$$\widehat{\mu}_s = \frac{1}{m}\sum_{i=1}^{m}x_s^i$$

and

$$\widehat{\mu}_{st} = \frac{1}{m}\sum_{i=1}^{m}x_s^i x_t^i$$

The $d = |\mathcal{V}| + |\mathcal{E}|$ moment matching equations will be

$$\sum_{\mathbf{x}\in\{0,1\}^{|\mathcal{V}|}} \exp\left\{\sum_{s\in\mathcal{V}} \eta_s x_s + \sum_{(s,t)\in\mathcal{E}} \eta_{st} x_s x_t - A(\eta)\right\} x_s = \widehat{\mu}_s$$

and

$$\sum_{\mathbf{x}\in\{0,1\}^{|\mathcal{V}|}} \exp\left\{\sum_{s\in\mathcal{V}} \eta_s x_s + \sum_{(s,t)\in\mathcal{E}} \eta_{st} x_s x_t - A(\eta)\right\} x_s x_t = \widehat{\mu}_{st}$$

The task is to estimate the values of the $|\mathcal{V}| + |\mathcal{E}|$ sized vector $\eta$. The log-normalization constant $A(\eta)$ is in turn very complex and involves

$$A(\eta) = \sum_{x\in\{0,1\}^{|\mathcal{V}|}} \exp\left\{\sum_{s\in\mathcal{V}} \eta_s x_s + \sum_{(s,t)\in\mathcal{E}} \eta_{st} x_s x_t - A(\eta)\right\}$$

In general, though the equations are intuitive (asserting that parameters be picked to match the data), solving them is very very complex. What if the graph were a tree? Then the solution can be obtained efficiently, as we will see subsequently. Also, $A(\eta)$ as well as the marginal $p(\mathbf{x};\eta)$ could be computed in polynomial time leading to efficient inference as well. Thus, solving the moment matching equations is closely linked to the graph structure. In general, solving the estimation problem inevitably involves solution to the marginalization problem, which can at least be performed efficiently for tree structured models.

Let us consider some graphical models, whose estimation problems are easy. Consider a markov chain with $\mathcal{V} = \{X_1, X_2, X_3, X_4\}$, $\mathcal{E} = \{(X_1, X_2), (X_2, X_3), (X_3, X_4)\}$. Though it can be thought of as a directed model, we will choose an exponential factorization that will link it to the general exponential machinery. Let each $X_i$ have $k$ possible states. Also, let

$$\lambda_{st} = \sum i = 1, j = 1^k \lambda_{st,ij}(\delta(x_s, i)\delta(x_t, j))$$

where $\delta(x, i)\delta(y, j)$ is an indicator feature function for each fixed $(i, j)$ pair and is expressible as a $k \times k$ matrix. $\lambda_{st,ij}$'s are the canonical parameters. We can then adopt the exponential form

$$p(\mathbf{x};\lambda) = \exp\left\{\lambda_{12}(x_1, x_2) + \lambda_{23}(x_2, x_3) + \lambda_{34}(x_3, x_4)\right\}$$

Given data $\overline{\mathbf{x}} = (\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(m)})$, the empirical moment parameters (sufficient statistics scaled down by $m$) can be computed to be

$$\overline{\mu}_{st}(x_s, x_t) = \frac{1}{m}\sum_{i=1}^{m} \delta(x_s = x_s^{(i)})\delta(x_t = x_t^{(i)})$$

for $(s, t) \in \{(1, 2), (2, 3), (3, 4)\}$ and

$$\overline{\mu}_s(x_s) = \frac{1}{m}\sum_{i=1}^{m} \delta(x_s = x_s^{(i)})$$

for $s \in \{1,2,3,4\}$.

These simply correspond to the empirical marginal probability $\wp(x_s, x_t)$. The moment matching conditions can be satisfied[44] by the following assignments to the canonical parameters:

$$\widehat{\lambda}_{12,12} = \log \overline{\mu}_{12}(x_1, x_2)$$

$$\widehat{\lambda}_{23,23} = \log \frac{\overline{\mu}_{23}(x_2, x_3)}{\overline{\mu}_2(x_2)}$$

$$\widehat{\lambda}_{34,34} = \log \frac{\overline{\mu}_{34}(x_3, x_4)}{\overline{\mu}_3(x_3)}$$

Thus, the canonical parameters can be computed in closed form. The task is similarly simple for trees - the basic intuition has been captured in this markov chain example. However, the simple solution suggested in this four node example does not hold for a cyclic graph having 4 nodes with edges defined as $\mathcal{E} = \{(X_1, X_2), (X_2, X_3), (X_3, X_4), (X_4, X_1)\}$.

The task is also equally simple, with closed form solutions for the class of decomposable graphs (which could be graphs with cycles), defined on page 415 in definition 54.

### 7.7.6 Iterative Algorithms

We will now move to more general graphs and describe iterative algorithms for solving fixed point equations. We will assume that the potential functions are defined on maximal cliques.

$$p(\mathbf{x}, \phi) \propto \prod_{\mathcal{C}\in\Pi} \phi_{\mathcal{C}}(\mathbf{x}_{\mathcal{C}}) \equiv \exp\left\{\sum_{\mathcal{C}\in\Pi} \theta_{\mathcal{C}}(\mathbf{x}_{\mathcal{C}})\right\}$$

The general setting for maximum likelihood optimization is that we have to solve a set of fixed point equations

$$\mathbf{F}(\eta) = E_\eta[\mathbf{f}(\mathbf{X})] - \overline{\mu} = \mathbf{0}$$

(such as $\widehat{\mu}_\alpha = \sum_{i=1}^{m} \sum_{\mathcal{C}\in\Pi|\alpha\in I_{\mathcal{C}}} f_\alpha(\mathbf{x}^{(i)})$ in the case of reduced parametrization - *c.f.* page 446). For this system, we will investigate an iterative solution of the form

$$\eta^{(t+1)} = \eta^{(t)} + f\left(\eta^{(t)}\right)$$

so that at the fixed point $\eta^{(t^*)}$, $f\left(\eta^{(t^*)}\right) = 0$ and therefore $\eta^{(t^*+1)} = \eta^{(t^*)}$. $t$ is the time step.

This general algorithm is called Iterative Proportional Fitting (IPF) and goes as follows for a general undirected graph $\mathcal{G} = <\mathcal{V}, \mathcal{E}>$:

[44]EXERCISE: Prove.

1. Start with some initial factorization (which could be uniform)

$$p(\mathbf{x}) \propto \prod_{\mathcal{C}\in\Pi} \phi_{\mathcal{C}}^{(0)}(\mathbf{x}_{\mathcal{C}}) \equiv \exp\left\{\sum_{\mathcal{C}\in\Pi} \theta_{\mathcal{C}}^{(0)}(\mathbf{x}_{\mathcal{C}})\right\}$$

2. For $t = 1$ onwards, let $\mathcal{C}$' range stepwise over the cliques in $\Pi$. Update

$$\phi_{\mathcal{C}'}^{(t+1)}(\mathbf{x}_{\mathcal{C}'}) = \phi_{\mathcal{C}'}^{(t)}(\mathbf{x}_{\mathcal{C}'})\frac{\overline{\mu}_{\mathcal{C}'}(\mathbf{x}_{\mathcal{C}'})}{\mu_{\mathcal{C}'}^{(t)}(\mathbf{x}_{\mathcal{C}'})} \tag{7.58}$$

where, $\mu_{\mathcal{C}'}^{(t)}(\mathbf{x}_{\mathcal{C}'})$ is the current model marginal, computed as[45]

$$\mu_{\mathcal{C}'}^{(t)}(\mathbf{x}_{\mathcal{C}'}) = \frac{1}{Z_{\phi^{(t)}}}\sum_{\mathbf{x}_{\mathcal{V}\setminus\mathcal{C}'}} p(\mathbf{x};\phi^{(t)})$$

While the model marginals can be computed efficiently for tree structured or even decomposable graphs using message passing formalisms, we may have to resort to approximate inferencing for general graphs. $\overline{\mu}_{\mathcal{C}'}$ are the empirical marginals that are precomputed from data $\overline{\mathbf{x}}$ as

$$\overline{\mu}_{\mathcal{C}'}(\mathbf{x}_{\mathcal{C}'}) = \frac{1}{m}\sum_{i=1}^{m}\delta\left(\mathbf{x}_{\mathcal{C}'},\mathbf{x}_{\mathcal{C}'}^{(i)}\right)$$

Equivalently one may also use the following update rule:

$$\theta_{\mathcal{C}'}^{(t+1)}(\mathbf{x}_{\mathcal{C}'}) = \theta_{\mathcal{C}'}^{(t)}(\mathbf{x}_{\mathcal{C}'}) + \log\frac{\overline{\mu}_{\mathcal{C}'}(\mathbf{x}_{\mathcal{C}'})}{\mu_{\mathcal{C}'}^{(t)}(\mathbf{x}_{\mathcal{C}'})} \tag{7.59}$$

The algorithm is called *iterative proportional fitting* because at every step, the potentials are updated proportional to how well the empirical moment parameters fit the model moments for a clique $\mathcal{C}'$.

### Convergence of the iterative algorithm

It is easy to see that at the fix point $t = t^*$, the algorithm will yield the MLE $\phi^{(t^*)}$ since, for all $\mathcal{C}\in\Pi$,

$$\mathbf{F}(\phi^{(t^*)}) = \mu_{\mathcal{C}}^{(t^*)}(\mathbf{x}_{\mathcal{C}}) - \overline{\mu}_{\mathcal{C}}(\mathbf{x}_{\mathcal{C}}) = 0$$

[45]You could formulate this problem using a set of $\delta_{\mathcal{C}}$ feature functions, and $\lambda_{\mathcal{C},\mathbf{v}_{\mathcal{C}}}$ canonical parameters, as was adopted in the example on page 453. Here, $\mathbf{v}_{\mathcal{C}}$ is a configuration vector that could be assumed by variables on clique $\mathcal{C}$.

But we also need to show that the updates will always converge to the fix point. We first observe that after updating using step (7.59) (or equivalently, (7.58)), the moment matching will be achieved for clique $\mathcal{C}'$.

$$\mu_{\mathcal{C}'}^{(t+1)}(\mathbf{x}_{\mathcal{C}'}) = \overline{\mu}_{\mathcal{C}'}(\mathbf{x}_{\mathcal{C}'})$$

Why is this so? By definition,

$$\mu_{\mathcal{C}'}^{(t+1)}(\mathbf{x}_{\mathcal{C}'}) = \frac{1}{Z_{\theta^{(t+1)}}} \sum_{\mathbf{x}_{\mathcal{V}\setminus\mathcal{C}'}} p(\mathbf{x};\theta^{(t+1)}) = \frac{1}{Z_{\theta^{(t+1)}}} \sum_{\mathbf{x}_{\mathcal{V}\setminus\mathcal{C}'}} \exp\left\{\theta_{\mathcal{C}'}^{(t)}(\mathbf{x}_{\mathcal{C}'}) + \sum_{\mathcal{C}\in\Pi,\mathcal{C}\neq\mathcal{C}'} \theta_{\mathcal{C}}^{(t)}(\mathbf{x}_{\mathcal{C}})\right\}$$

That is, every factor that is not in $\mathcal{C}'$ is not changing with respect to the previous step. Expanding upon this, using the update equation (7.59),

$$\mu_{\mathcal{C}'}^{(t+1)}(\mathbf{x}_{\mathcal{C}'}) = \frac{1}{Z_{\theta^{(t+1)}}} \sum_{\mathbf{x}_{\mathcal{V}\setminus\mathcal{C}'}} \exp\left\{\theta_{\mathcal{C}'}^{(t)}(\mathbf{x}_{\mathcal{C}'}) + \log\frac{\overline{\mu}_{\mathcal{C}'}(\mathbf{x}_{\mathcal{C}'})}{\mu_{\mathcal{C}'}^{(t)}(\mathbf{x}_{\mathcal{C}'})} + \sum_{\mathcal{C}\in\Pi,\mathcal{C}\neq\mathcal{C}'} \theta_{\mathcal{C}}^{(t)}(\mathbf{x}_{\mathcal{C}})\right\} = \frac{1}{Z_{\theta^{(t+1)}}} \frac{\overline{\mu}_{\mathcal{C}'}(\mathbf{x}_{\mathcal{C}'})}{\mu_{\mathcal{C}'}^{(t)}(\mathbf{x}_{\mathcal{C}'})} \sum_{\mathbf{x}_{\mathcal{V}\setminus\mathcal{C}'}} \mathrm{e}$$

since $\overline{\mu}_{\mathcal{C}'}(\mathbf{x}_{\mathcal{C}'})$ and $\mu_{\mathcal{C}'}^{(t)}(\mathbf{x}_{\mathcal{C}'})$ are independent of the inner summation. This leads to

$$\mu_{\mathcal{C}'}^{(t+1)}(\mathbf{x}_{\mathcal{C}'}) = \frac{Z_{\theta^{(t)}}}{Z_{\theta^{(t+1)}}} \frac{\overline{\mu}_{\mathcal{C}'}(\mathbf{x}_{\mathcal{C}'})}{\mu_{\mathcal{C}'}^{(t)}(\mathbf{x}_{\mathcal{C}'})} \sum_{\mathbf{x}_{\mathcal{V}\setminus\mathcal{C}'}} p(\mathbf{x}_{\mathcal{C}'};\theta^{(t)}$$

By definition,

$$\mu_{\mathcal{C}'}^{(t)}(\mathbf{x}_{\mathcal{C}'}) = \sum_{\mathbf{x}_{\mathcal{V}\setminus\mathcal{C}'}} p(\mathbf{x}_{\mathcal{C}'};\theta^{(t)}$$

Thus,

$$\mu_{\mathcal{C}'}^{(t+1)}(\mathbf{x}_{\mathcal{C}'}) = \frac{Z_{\theta^{(t)}}}{Z_{\theta^{(t+1)}}} \overline{\mu}_{\mathcal{C}'}(\mathbf{x}_{\mathcal{C}'})$$

Since both empirical and model moments are expected to sum to 1 across all cliques in the graph, we should have

$$1 = \sum_{\mathcal{C}'\in\Pi} \mu_{\mathcal{C}'}^{(t+1)}(\mathbf{x}_{\mathcal{C}'}) = \sum_{\mathcal{C}'\in\Pi} \frac{Z_{\theta^{(t)}}}{Z_{\theta^{(t+1)}}} \overline{\mu}_{\mathcal{C}'}(\mathbf{x}_{\mathcal{C}'}) = \frac{Z_{\theta^{(t)}}}{Z_{\theta^{(t+1)}}}$$

Consequently, we will have that once the update step (7.59) is applied, moment matching will hold on $\mathcal{C}'$.

$$\mu_{\mathcal{C}'}^{(t+1)}(\mathbf{x}_{\mathcal{C}'}) = \overline{\mu}_{\mathcal{C}'}(\mathbf{x}_{\mathcal{C}'})$$

This takes us one step forward. But are these step updates guranteed to lead to convergence? To see that convergence holds, we will note that the IPF algorithm is an instance of the coordinate-wise ascent algorithm (*c.f.* page 301). This becomes obvious by stating the log-likelihood objective

$$LL(\theta;\overline{\mathbf{x}}) = \frac{1}{m}\left(\sum_{\mathcal{C}\in\Pi} \theta_{\mathcal{C}} \overline{\mu}_{\mathcal{C}'}(\mathbf{x}_{\mathcal{C}'})\right) - A(\{\theta_{\mathcal{C}} \mid \mathcal{C}\in\Pi\})$$

and viewing it as a function of $\theta_{\mathcal{C}'}$

$$g(\theta_{\mathcal{C}'}) = LL\left(\theta_{\mathcal{C}'}; \left\{\theta_{\mathcal{C}} = \theta_{\mathcal{C}}^{(t)} \mid \mathcal{C} \neq \mathcal{C}'\right\}, \overline{\mathbf{x}}\right)$$

The first order necessary optimality condition for $g(\theta_{\mathcal{C}'})$ precisely yield the moment matching equation for $\mathcal{C}'$. Since $g(\theta_{\mathcal{C}'})$ can be shown to be concave, the first order necessary conditions are also sufficient. The fact that the application of coordinate descent in this setting will lead to convergence follows from the fact that $LL(\theta; \overline{\mathbf{x}})$ is strongly concave.

### 7.7.7 Maximum Entropy Revisted

The IPF algorithm exploited the idea of matching moments in order to maximize likelihood. Recall from Section 7.6.1, the maximum entropy principle that seeks to find $\wp(.) \in \mathcal{P}(\widehat{\mu})$ that maximizes $H(p)$. $\mathcal{P}$ is the set of all distributions that satisfy empirical moment constraints $\widehat{\mu_\alpha}$:

$$\mathcal{P}(\widehat{\mu}) = \left\{ p(.) \left| \sum_{\mathbf{x}} p(\mathbf{x}) f_\alpha(\mathbf{x}) = \widehat{\mu_\alpha} \quad \forall \alpha \in I \right. \right\}$$

The problem of maximum entropy was also shown to be equivalent to the problem of minimizing the KL divergence between $p$ and the reference uniform distribution $u$:

$$\widehat{p}_{ME} = \underset{p \in \mathcal{P}(\widehat{\mu})}{\operatorname{argmax}}\ H(p) = \underset{p \in \mathcal{P}(\widehat{\mu})}{\operatorname{argmin}}\ D(p||u) \tag{7.60}$$

On the other hand, the problem of maximum likelihood estimation problem is specified as

$$\widehat{p}_{MLE} = \underset{\eta}{\operatorname{argmax}}\ \frac{1}{m} \sum_{i=1}^{m} \log p(\mathbf{x}^{(i)}; \eta)$$

To help us establish the connection between the two, we will define the data $\overline{\mathbf{x}}$ driven empirical distribution as

$$\wp_{\overline{\mathbf{x}}}(\mathbf{x}) = \frac{1}{m} \sum_{i=1}^{m} \delta(\mathbf{x} = \mathbf{x}^{(i)})$$

This definition yields an alternative expression for the MLE as the minimizer of the KL divergence between the empirical distribution and the model distribution.

$$\begin{aligned}
\widehat{p}_{MLE} &= \underset{\eta}{\operatorname{argmax}} \left( \frac{1}{m} \sum_{i=1}^{m} \wp_{\overline{\mathbf{x}}} \log p(\mathbf{x}^{(i)}; \eta) \right) && (7.61)\\
&= \underset{\eta}{\operatorname{argmax}} \left( \frac{1}{m} \sum_{i=1}^{m} \wp_{\overline{\mathbf{x}}} \log p(\mathbf{x}^{(i)}; \eta) \right) + \left( \frac{1}{m} \sum_{i=1}^{m} \wp_{\overline{\mathbf{x}}} \log \wp_{\overline{\mathbf{x}}} \right) && (7.62)\\
&= \underset{\eta}{\operatorname{argmax}} - D\left(\wp_{\overline{\mathbf{x}}} || p(\overline{\mathbf{x}}; \eta)\right)\\
&= \underset{\eta}{\operatorname{argmin}} D\left(\wp_{\overline{\mathbf{x}}} || p(\overline{\mathbf{x}}; \eta)\right)\\
&= \underset{p \in \mathbf{E}(\mathbf{f})}{\operatorname{argmin}} D\left(\wp_{\overline{\mathbf{x}}} || p(.)\right) && (7.63)
\end{aligned}$$

where, $\mathbf{E}(\mathbf{f})$ is the exponential family of distributions having $\mathbf{f}$ as the vector of feature functions that also figure in the specification of constraints in $\mathcal{P}(\overline{\mu})$:

$$\mathbf{E}(\mathbf{f}) = \left\{ p(.) \left| p(\mathbf{x}; \eta) \propto \exp\left\{ \eta^T \mathbf{f}(\mathbf{x}) \right\} \right. \right\}$$

We will now show the equivalence of specifications in (7.60) and (7.63). The discussion so far will be relevant in our proof. On the one hand, we have seen in (7.49) in theorem 99 that the constraint in $\mathbf{E}(\mathbf{f})$ is satisified by any solution to (7.60). While on the other hand, we know that the moment matching conditions for (7.63) in (7.57) are precisely the constraints in $\mathcal{P}(\overline{\mu})$.

**Theorem 101** $\widehat{p}_{MLE} = \widehat{p}_{ML}$ *for exponential families, where:*

$$\widehat{p}_{MLE} = \underset{p \in \mathbf{E}(\mathbf{f})}{\operatorname{argmin}} D\left(\wp_{\overline{\mathbf{x}}} || p(.)\right)$$

*and*

$$\widehat{p}_{ME} = \underset{p \in \mathcal{P}(\widehat{\mu})}{\operatorname{argmin}}\ D(p||u)$$

*Two differences between these formulations are:*

1. *While $\widehat{p}_{MLE}$ involves optimization over the second argument in the KL divergence, $\widehat{p}_{ME}$ involves optimization over the first argument.*

2. *The entry point of the data is also toggled in the two; while $\widehat{p}_{ME}$ has data entering through constraint set $\mathcal{P}$, $\widehat{p}_{MLE}$ has data entering through the cost function.*

*This is characteristic of dual problems.*

*Proof:* This can be proved by first showing that the problem in (7.60) is the dual of (7.63). Next we will need to apply duality theory in Theorem 82, which states that for convex cost function and convex inequality constraint set, the KKT conditions are necessary and sufficient conditions for zero duality gap. It can be proved that (7.60) is convex and that the parameters for $\widehat{p}_{MLE}$ are solutions to the KKT conditions.

The theorem can also be proved by invoking the so called *pythagorean theorem*[46] for general class of distance that includes distributions. In this particular case, it can be shown that for all $\widehat{p} \in \mathcal{P}(\overline{\mu}) \cap \mathbf{E}(\mathbf{f})$ and for all $p_{\mathcal{P}} \in \mathcal{P}(\overline{\mu})$ and $p_{\mathcal{E}} \in \mathcal{E}(\mathbf{f})$,

$$D(p_{\mathcal{P}}||p_{\mathbf{E}}) = D(p_{\mathcal{P}}||\widehat{p}) + D(\widehat{p}||p_{\mathbf{E}})$$

If $\widehat{q}, \widehat{p} \in \mathcal{P}(\overline{\mu}) \cap \mathbf{E}(\mathbf{f})$, then it will turn out by simple application of the theorem that $D(\widehat{q}||\widehat{p}) + D(\widehat{p}||\widehat{q}) = 0$, which implies that $\widehat{p} = \widehat{q}$. That is, $\mathcal{P}(\overline{\mu}) \cap \mathbf{E}(\mathbf{f})$ is a singleton and $\widehat{p}$ should correspond to both $\widehat{p}_{MLE}$ and $\widehat{p}_{ML}$. $\square$

## 7.8 Learning with Incomplete Observations

Thus far, we have focussed on learning parameters for graphical models using complete observations; the underlying model was $p(\mathbf{x};\theta)$ and the observations (data) were $\overline{\mathbf{x}} = (\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(n)})$. An example of such a learning task was presented in the case of Markov chains on page 453. Consider the hidden markov model from page 430 with $\mathcal{V} = \{X_1, X_2, X_3, X_4, X_5, Y_1, Y_2, Y_3, Y_4, Y_5\}$ and $\mathcal{E} = \{(X_1, X_2), (X_1, Y_1), (X_2, X_3), (X_2, Y_2), (X_3, X_4), (X_3, Y_3), (X_4, X_5), (X_4, Y_4), (X_5, Y_5)\}$. where nodes $\{X_1, X_2, X_3, X_4, X_5\}$ are hidden variables and nodes $\{Y_1, Y_2, Y_3, Y_4, Y_5\}$ are the observed variables.

Mixture models are another set of popular example, which feature hidden variables. Many real world problems are characterized by distinct classes/subpopulations that the data falls into and can be modeled using mixture models.

**Definition 63** *Let $Z \in \{z_1, z_2, \ldots, z_k\}$ be a multinomial variable indicating mixture component. Let $\mathbf{X}$ be a random variable (vector), whose distribution is specified, conditioned on different values $z_i$ of $Z$ as*

$$p(\mathbf{x}|z_i;\theta_i) \sim f_i(x;\theta_i)$$

*Then the finite mixture model is defined as*

$$p(\mathbf{x}) \sum_{i=1}^{k} p(z_i) f_i(\mathbf{x};\theta_i)$$

*with $k$ being the number of mixture components, $Z$ called the mixture indicator component, $f_i(\mathbf{x};\theta_i)$ termed as the density of the $i^{th}$ mixture component with parameters $\theta_i$. The quantities $p(z_i) = \pi_i$ are also called mixing weights, representing the proportion of the population in subpopulation $i$. Thus, $\pi = [\pi_1, \pi_2, \ldots, \pi_k]$ and $\theta = [\theta_1, \theta_2, \ldots, \theta_k]$ are the paramaters of a finite mixture model, with a fixed value of $k$. As a graphical model, the mixture model can be represented as a two node graph: $\mathcal{G} =< \mathcal{V}, \mathcal{E} >$ with $\mathcal{V} = \{\mathbf{X}, Z\}$ and $\mathcal{E} = \{(\mathbf{X}, Z)\}$.*

[46]The right angle here is not the conventional one, but a notional one.

As an example, the density of each mixture component could be Gaussian with $\theta_i = (\mu_i, \Sigma_i)$.

$$f_i(\mathbf{x}; \theta_i) \sim \mathcal{N}(\mu_i, \Sigma_i)$$

The distribution $p(\mathbf{x})$ is then called a mixture of Gaussians. In general, it not Gaussian itself.

How can the parameters be learnt in the presence of incomplete data? In the case of the HMM example, we might be provided only with observations $\overline{\mathbf{y}}$ for $\mathbf{Y}$ and expected to learn the parameters. Or in the case of mixture models, we might be presented only with instances of $\mathbf{X}$ in the form of $\overline{\mathbf{x}}$ and required to learn parameters $\pi$ and $\theta$. We will illustrate parameter learning for the mixture model problem.

## 7.8.1 Parameter Estimation for Mixture Models

It can be shown that learning for mixture models is an easy problem if the data is fully observed in the form $(\overline{\mathbf{x}}, \overline{z}) = \left[(\mathbf{x}^{(1)}, z^{(1)}), (\mathbf{x}^{(2)}, z^{(2)}), \ldots, (\mathbf{x}^{(m)}, z^{(m)})\right]$. The joint distribution can be decomposed as

$$p(\mathbf{x}, z; \theta) = p(z)p(\mathbf{x} \mid z, \theta)$$

If $p(\mathbf{x} \mid z, \theta)$ is Gaussian and since $p(z)$ is multinomial, the joint will be in exponential form with Gaussian and multinomial sufficient statistics. The maximum likelihood estimation will boil down to moment matching with respect to these sufficient statistics, leading to an easy estimation problem.

In the incomplete data setting, we are given only $\overline{\mathbf{x}}$ while observations $\overline{z}$ on the mixture components are hidden. The likelihood can still be expressed and maximized:

$$LL(\pi, \theta; \overline{\mathbf{x}}) = \frac{1}{m}\sum_{i=1}^{m} \log p(\mathbf{x}^{(i)}; \theta) = \frac{1}{m}\sum_{i=1}^{m} \log \left[\sum_{j=1}^{k} \pi_j f_j(\mathbf{x}^{(i)}; \theta_j)\right]$$

subject to the constraints that $\pi_j \geq 0$ and $\sum_{j=1}^{k} \pi_j = 1$.

Unfortunately, log cannot be distributed over a summation and that creates the main bottleneck. In case the densities are Gaussians, the objective to be maximized will be

$$LL(\pi, \mu, \Sigma; \overline{\mathbf{x}}) = \frac{1}{m}\sum_{i=1}^{m} \log \left[\sum_{j=1}^{k} \pi_j \mathcal{N}\left(\mathbf{x}^{(i)}; \mu_j, \Sigma_j\right)\right]$$

subject to the constraints that $\pi_j \geq 0$ and $\sum_{j=1}^{k} \pi_j = 1$.

1. **M-step:** Writing down the KKT necessary and sufficient optimality conditions (see (4.88) on page 296) for this maximization problem, subject to its associated inequality and linear equality constraints yields:

   (a) For $\mu_j$

$$\mu_j = \frac{\sum_{i=1}^{m} p(z_j \mid \mathbf{x}^{(i)}, \mu, \Sigma)\mathbf{x}^{(i)}}{\sum_{i=1}^{m}\sum_{j=1}^{k} p(z_j \mid \mathbf{x}^{(i)}, \mu, \Sigma)} \tag{7.64}$$

   (b) And for $\Sigma_j$

$$\Sigma'_j = \frac{\sum_{i=1}^{m} p(z_j \mid \mathbf{x}^{(i)}, \mu, \Sigma)\left(\mathbf{x}^{(i)} - \mu_j\right)\left(\mathbf{x}^{(i)} - \mu_j\right)^T}{\sum_{i=1}^{m}\sum_{j=1}^{k} p(z_j \mid \mathbf{x}^{(i)}, \mu, \Sigma)} \tag{7.65}$$

   These steps are called the $M - steps$ or the maximization steps, since they are obtained as necessary and sufficient conditions of optimality for a maximization problem.

2. **E-step:** The posterior $p(z_j \mid \mathbf{x}^{(i)}, \mu, \Sigma)$ and the prior $\pi_j$ in (7.65) and (7.64) can be determined using Bayes rule as

   (a)
$$p(z_j \mid \mathbf{x}^{(i)}, \mu, \Sigma) = \frac{\pi_j f(\mathbf{x}; \mu_j, \Sigma_j)}{\sum_{a=1}^{k} \pi_a f(\mathbf{x}; \mu_a, \Sigma_a)}$$

   (b)
$$\pi_j = \frac{1}{m}\sum_{i=1}^{m} p(z_j \mid \mathbf{x}^{(i)}, \mu, \Sigma)$$

The problem is that we do not get a closed form solution here; what we obtain are a set of coupled, non-linear equations and need to iterate between these steps to arrive at the fix point. This is where the expectation maximization (EM) algoriothm comes in. We now will specify the EM algorithm in a more general setting.

### 7.8.2 Expectation Maximization

Let $\mathbf{X}$ be a set of observed variables and $\mathbf{Z}$ be a set of hidden variables for some statistical model. Let $\overline{\mathbf{x}}$ be $m$ observations on $\mathbf{X}$. In this general setting, we really need not assume that the samples in $\overline{\mathbf{x}}$ are iid (though you could). We will assume that the MLE problem would have been easy[47] if $\overline{\mathbf{z}}$ was observed data for the hidden variables $\mathbf{Z}$ (such as in the case of the mixture model). The complete data log-likelihood would have been:

$$LL(\theta; \overline{\mathbf{x}}, \overline{\mathbf{z}}) = \frac{1}{m} \log p(\overline{\mathbf{x}}, \overline{\mathbf{z}}; \theta)$$

Given a predictive distribution $q(\mathbf{z}|\mathbf{x})$, the expected complete data log-likelihood is a function of the observed $\overline{\mathbf{x}}$ and $\theta$ and is defined as

$$LL_E(\theta; \overline{\mathbf{x}}) = \sum_{\mathbf{z}} q(\mathbf{z}|\overline{\mathbf{x}}) \log p(\overline{\mathbf{x}}, \overline{\mathbf{z}}; \theta) \tag{7.66}$$

The expected complete data log-likelihood is an auxilliary function that gives a lower bound on the actual log-likelihood we want to optimize for. The actual log-likelihood in this general setting will be:

$$LL(\theta; \overline{\mathbf{x}}) = \frac{1}{m} \log \left\{ \sum_{\mathbf{z}} p(\overline{\mathbf{x}}, \mathbf{z}; \theta) \right\}$$

For example, the actual log-likelihood with iid assumption will be:

$$LL(\theta; \overline{\mathbf{x}}) = \frac{1}{m} \sum_{i=1}^{m} \log \left\{ \sum_{\mathbf{z}} p(\mathbf{x}^{(i)}, \mathbf{z}; \theta) \right\}$$

**Theorem 102** *For all $\theta$ and every possible distribution $q(\mathbf{z}|\mathbf{x})$, following holds:*

$$LL(\theta; \overline{\mathbf{x}}) \geq LL_E(\theta; \overline{\mathbf{x}}) + \frac{1}{m} H(q) \tag{7.67}$$

*Equality holds if and only if*

$$q(\mathbf{z}|\mathbf{x}) = p(\mathbf{z}|\mathbf{x}; \theta)$$

*Proof:* First of all

$$LL(\theta; \overline{\mathbf{x}}) = \frac{1}{m} \log \left\{ \sum_{\mathbf{z}} q(\mathbf{z}|\overline{\mathbf{x}}) \frac{p(\overline{\mathbf{x}}, \mathbf{z}; \theta)}{q(\mathbf{z}|\overline{\mathbf{x}})} \right\}$$

[47] The trick in such a setting is to identify the model, $\mathbf{X}$ and $\mathbf{Z}$ so that you make the MLE problem easy in the presence of complete data.

Using the Jensen's inequality (since *log* is a strictly convex function),

$$LL(\theta;\overline{\mathbf{x}}) \geq \underbrace{\frac{1}{m}\sum_{\mathbf{z}} q(\mathbf{z}|\overline{\mathbf{x}}) \log p(\overline{\mathbf{x}},\mathbf{z};\theta)}_{LL_E(\theta;\overline{\mathbf{x}})} \underbrace{-\frac{1}{m}\sum_{\mathbf{z}} q(\mathbf{z}|\overline{\mathbf{x}}) \log q(\mathbf{z}|\overline{\mathbf{x}})}_{H(q)}$$

Equality holds if and only if $\frac{p(\overline{\mathbf{x}},\mathbf{z};\theta)}{q(\mathbf{z}|\overline{\mathbf{x}})}$ is a constant, that is,

$$q(\mathbf{z}|\overline{\mathbf{x}}) \propto p(\overline{\mathbf{x}},\mathbf{z};\theta) = p(\mathbf{z}|\overline{\mathbf{x}};\theta)p(\overline{\mathbf{x}};\theta) \propto p(\mathbf{z}|\overline{\mathbf{x}};\theta)$$

This can happen if and only if $q(\mathbf{z}|\overline{\mathbf{x}}) = p(\mathbf{z}|\overline{\mathbf{x}};\theta)$. □

A consequence of theorem 102 is that

$$\max_{\theta} LL(\theta;\overline{\mathbf{x}}) = \max_{\theta} \max_{q} LL_E(\theta;\overline{\mathbf{x}}) + \frac{1}{m}H(q) \tag{7.68}$$

**The EM algorithm is simply coordinate ascent on the auxilliary function $LL_E(\theta;\overline{\mathbf{x}}) + \frac{1}{m}H(q)$.** The expectation and maximization steps at time instance $t$ can be easily identified for the formulation in (7.68) as

1. **Expectation Step:**

$$q^{(t+1)} = \underset{q}{\operatorname{argmax}}\, LL_E(\theta^{(t)};\overline{\mathbf{x}}) + \frac{1}{m}H(q) = \underset{q}{\operatorname{argmax}} -D\left(q(\mathbf{z}|\mathbf{x})||p(\mathbf{z}|\mathbf{x};\theta^{(t)})\right) + \log\left\{\mathbf{x};\theta^{(t)}\right\} \tag{7.69}$$

Since, $LL_E(\theta^{(t)};\overline{\mathbf{x}}) + \frac{1}{m}H(q) \leq \log\left\{\mathbf{x};\theta^{(t)}\right\}$ by theorem 102, the maximum value is attained in (7.69) for $q(\mathbf{z}|\mathbf{x}) = p(\mathbf{z}|\mathbf{x};\theta^{(t)})$. Thus, the E-step can be summarized by

$$q^{(t+1)}(\mathbf{x}|\mathbf{x}) = p(\mathbf{z}|\mathbf{x};\theta^{(t)}) \tag{7.70}$$

The E-step can involve procedures such as sum-product for obtaining marginals and/or conditions, if the distribution is defined on a graphical model, to obtain $p(\mathbf{z}|\mathbf{x};\theta^{(t)})$.

2. **Maximization Step:**

$$\theta^{(t+1)} = \underset{\theta}{\operatorname{argmax}}\, LL_E(\theta;\overline{\mathbf{x}}) + \frac{1}{m}H(q^{(t+1)})$$

Since the maximization is over $\theta$ and since $H(q)$ is independent of $\theta$, we can rewrite the M-step to be

$$\theta^{(t+1)} = \underset{\theta}{\text{argmax}}\ LL_E(\theta;\overline{\mathbf{x}}) = \underset{\theta}{\text{argmax}}\ \sum_{\mathbf{z}} q(\mathbf{z}|\overline{\mathbf{x}}) \log p(\overline{\mathbf{x}},\overline{\mathbf{z}};\theta) \tag{7.71}$$

In essence, the M-step looks very much like an ordinary maximum likelihood estimation problem, but using predicted values of **z**. The M-step may not have a closed form solution, in which case, it may be required to resort to iterative techniques such as IPF (7.7.6).

Let us take some examples to illustrate the generic EM procedure outlined here. It is particularly useful if the term $\log p(\overline{\mathbf{x}},\overline{\mathbf{z}};\theta)$ were to split up into smaller terms (such as sum of sufficient statistics in the case of exponential models). Consider the Gauss Markov process specified by $Z_{t+1} = \theta Z_T + W_t$, where $Z_0 \sim \mathcal{N}(0,1)$ and $W_t \sim \mathcal{N}(0,1)$. Let $\theta \in \Re$ be the parameter to be estimated. The graphical model representation is $\mathcal{V} = \{Z_1, Z_2, Z_2, \ldots, Z_n, X_1, X_2, \ldots, X_n\}$ and $\mathcal{E} = \{(Z_1,Z_2),(Z_1,X_1),(Z_2,Z_3),(Z_2,Z_2),\ldots,(Z_{n-1},Z_n),(Z_{n-1},X_{n-1}),(Z_n,X_n)\}$.

Say what we observe are noisy, indirect observations $X_t = Z_t + V_t$, $V_t \sim \mathcal{N}(0,\sigma_2)$ being the observation noise. Let $\overline{\mathbf{x}}$ be a set of $m$ iid observations for $\mathbf{X}$ while $\mathbf{Z}$ remains hidden. Both $\mathbf{X}$ and $\mathbf{Z}$ are vectors of random variables of size $n$ each. Then,

$$\begin{aligned} LL(\theta;\overline{\mathbf{x}}) &= \frac{1}{m}\sum_{i=1}^{m} p(x^{(i)};\theta) \\ &= \frac{1}{m}\sum_{i=1}^{m}\log\left\{\int_{\mathbf{z}}\prod_{t=1}^{n}\frac{1}{\sqrt{2\pi}\sigma}\exp\left\{-\frac{1}{2}\frac{(x_t^{(i)}-z_t)^2}{\sigma^2}p(\mathbf{z};\theta)d\mathbf{z}\right\}\right\} \end{aligned} \tag{7.72}$$

which is a mess! In contrast, the lower-bound component $LL_E$ allows us to move the integration outside the logarithm, enabling more simplification:

$$\begin{aligned} LL_E(\theta;\overline{\mathbf{x}}) &= \frac{1}{m}\int_{\mathbf{z}} q(\mathbf{z}|\overline{\mathbf{x}})\log p(\overline{\mathbf{x}}\mathbf{z};\theta)d\mathbf{z} \\ &= \frac{1}{m}\int_{\mathbf{z}} q(\mathbf{z}|\overline{\mathbf{x}})\sum_{i=1}^{m}\log p(\mathbf{x}^{(i)}\mathbf{z};\theta)d\mathbf{z} \\ &= \frac{1}{m}\int_{\mathbf{z}} q(\mathbf{z}|\overline{\mathbf{x}})\sum_{i=1}^{m}\left[\log p(\mathbf{z},\theta)+\sum_{t=1}^{n}\log\left(\frac{1}{\sqrt{2\pi\sigma^2}}\right)-\frac{1}{2\sigma^2}(x_t^{(i)}-z_t)^2\right]d\mathbf{z} \end{aligned}$$

As can be seen above, $p(\mathbf{x}^{(i)}|\mathbf{z};\theta)$ is independent of $\theta$ and therefore, the term $q(\mathbf{z}|\overline{\mathbf{x}})\log p(\mathbf{x}^{(i)}|\mathbf{z};\theta)$ can be brought out as a separate constant (since that part of the integral will not change with $\theta$). This leads to the following simplified expression for $LL_E$

$$\begin{aligned}
LL_E(\theta;\overline{\mathbf{x}}) &= \frac{1}{m}\int_{\mathbf{z}} q(\mathbf{z}|\overline{\mathbf{x}})\log p(\overline{\mathbf{x}}\mathbf{z};\theta)d\mathbf{z} \\
&= C + \frac{1}{m}\int_{\mathbf{z}} q(\mathbf{z}|\overline{\mathbf{x}})\left[\log p(z_1) + \log p(z_2|z_1;\theta) + \ldots + \log p(z_n|z_{n-1};\theta)\right]d\mathbf{z} \\
&= C' + \frac{1}{m}\int_{\mathbf{z}} q(\mathbf{z}|\overline{\mathbf{x}})\left[\log p(z_2|z_1;\theta) + \ldots + \log p(z_n|z_{n-1};\theta)\right]d\mathbf{z}
\end{aligned}$$

In the E-step, the Kalman filter can be used to compute $p(\mathbf{z}|\mathbf{x};\theta^{(t+1)})$ in terms of $\theta^{(t)}$. In the M-step, first order necessary optimality conditions on $LL_E$ will yield $\theta^{(t+1)}$.

Recall from Section 7.7.7, equation (7.63) that the likelihood maximization problem can be viewed as a problem of minimizing the KL divergence between the empirical distribution and the model distribution.

$$\widehat{p}_{MLE} = \underset{p\in\mathbf{E}(\mathbf{f})}{\operatorname{argmin}} D\left(\wp_{\overline{\mathbf{x}}}||p(\mathbf{x};\theta)\right)$$

While the likelihood maximization perspective lead to a lower-bounding strategy in the form of EM, an alternative upper-bounding strategy can also be adopted to view EM, though it is only the older bound in disguise. Making use of theorem 102, we can prove that for all distributios $q(\mathbf{z}|\mathbf{x})$ and any parameter $\theta$, the following always holds:

$$D\left(\wp(\mathbf{x})||p(\mathbf{x};\theta)\right) \leq D\left(\wp(\mathbf{x})q(\mathbf{z}|\mathbf{x})||p(\mathbf{x},\mathbf{z};\theta)\right) \tag{7.73}$$

This statement says that the KL divergence between the empirical and model distributions that maximum likelihood tries to minimize is upperbounded by the KL divergence between the 'completed' empirical and model distributions. As before, it can also be shown that the bound is tight if and only if $q(\mathbf{z}|\mathbf{x}) = p(\mathbf{z}|\mathbf{x};\theta)$. The E-step will remain the same as before. Only, the M-step will slightly change:

1. **KL divergence perspective of E-Step:**

$$q^{(t+1)}(\mathbf{z}|\mathbf{x}) = \underset{q}{\operatorname{argmin}} D\left(\wp(\mathbf{x})q(\mathbf{z}|\mathbf{x})||p(\mathbf{x},\mathbf{z};\theta^{(t)})\right)$$

2. **KL divergence perspective of M-Step:**

$$\theta^{(t+1)} = \underset{\theta}{\operatorname{argmin}} D\left(\wp(\mathbf{x})q^{(t+1)}(\mathbf{z}|\mathbf{x})||p(\mathbf{x},\mathbf{z};\theta)\right)$$

These modified E and M steps correspond to coordinate descent in constrast to the earlier perspective of coordinate ascent.

## 7.9 Variational Methods for Inference

In contrast to the sampling methods, variational methods are deterministic and fast algorithms that generate good approximations to the problems of computing marginals and MAP configurations. They are involve the reformulation of the quantity of interest (such as log-likelihood, first order moments, marginal distributions, *etc.*) as the solution of some optimization problem. They are useful in several ways:

- The variational formulation often leads to efficient algorithms for determining exact solutions. Many algorithms discussed thus far, could be discovered as efficient techniques for solving the variational optimization problem.
- For many quantities that are hard to compute, the variational perspective leads to approximate solutions. Mean field and loopy sum product algorithms can also be viewed as special cases of approximation through variational inference.
- In contrast to approximate sampling methods, these are faster, deterministic and cheaper (in terms of memory).

We will motivate variational methods using two examples.

1. The first is the mean field algorithm for the Ising model defined on a graph $\mathcal{G} =< \mathcal{V}, \mathcal{E} >$ of binary (0/1) variables, with pairwise interactions between adjacent variables captured through their product $(x_s x_t)$

$$p(\mathbf{x}, \eta) \propto \exp\left\{\sum_{s\in\mathcal{V}} \eta_s x_s + \sum_{(s,t)\in\mathcal{E}} \eta_{st} x_s x_t\right\}$$

The Gibbs sampling for the Ising model derives updates of the form

$$x_i^{(t+1)} = \begin{cases} 1 & \text{if } u \sim uniform[0,1] \leq 1 + \exp\left\{-\eta_i + \sum\limits_{j\in\mathcal{N}(s)} \eta_{ij} x_j^{(t)}\right\} \\ 0 & \text{otherwise} \end{cases}$$

which correspond to a non-deterministic version of the message passing algorithm (owing to $u \sim uniform[0,1]$). The updates are very simple and local, making this a good choice.

The mean field method has its roots in physics. It makes a deterministic to the Gibbs update by replacing each random variable $X_i$ by a deterministic mean parameter $\mu_i \in [0,1]$ (which can be thought of as the probability of $X_i = 1$) and updates $\mu_i$ using

$$\mu_i^{(t+1)} = \left(1 + \exp\left\{-\eta_i + \sum_{j\in\mathcal{N}(s)} \eta_{ij} \mu_j^{(t)}\right\}\right)^{-1}$$

Thus, the mean field algorithm is exactly a message passing algorithm, but has some semantics related to sampling techniques. We will see that the mean field algorithm is a specific instance of variational methods and it can be formulated as coordinate descent on a certain optimization problem and subsequently be analysed for convergence, *etc.*

2. The *loopy sum-product* (also called the loopy belief propagation) method is another instance of variational methods. 'Loopy' here means on graphs with cycles. If the tree width were low, you could create a junction tree and perform message passing, but what if the tree width were large, such as with a grid. This algorithm is the most naive application of the sum-product updates (originally developed for trees in Section 7.2.2) and apply it to graphs with cycles. This naive procedure has had extraordinary success in the fields of signal processing, compute vision, bioinformatics and most importantly, in communication networks, *etc.*

   The message passing algorithm, when applied on a tree, breaks it into subtrees and passes messages between subtrees, which are independent (share no variable). But the moment you add an edge connecting any two subtrees, they are not independent any more. Passing messages around in cycles can lead to over-counting (analogous to gossip spreading in a social network). Thus, the message passing algorithm ceases to remain an exact algorithm and does not even gurantee convergence.

   What turns out to be actually very important is how long are the cycles on which messages are being propagated; for *long cycles, the effects of over-counting can be weakened.* More technically speaking, the behaviour will depend on

   (a) The girth of the graph (length of cycles): For larger girth, you could run the message passing for many iterations before you land up with overcounting.

   (b) The strength of the potentials are, or in other words, how close to independence is the model. For example, in the Ising model itself, based on the coupling induced through terms of the form $x_s x_t$, if the coupling is weak, almost close to independence, the algorithm will be perfect, giving almost exact answers. There is a region of transition, based on strengthening of the coupling terms, beyond which the algorithm breaks down.

3. The idea behind variational methods is to represent the quantity of interest (such as the marginal or mode over a graphical model) as the solution to an optimization problem. For example, the solution to $A\mathbf{x} = \mathbf{b}$ (as in the case of inferencing for Kalman filters) with $A \succ 0$ and $\mathbf{b} \in \Re^n$ can be represented as the solution to

$$\widetilde{\mathbf{x}} = \underset{\mathbf{x}}{\operatorname{argmin}} \frac{1}{2}\mathbf{x}^T A\mathbf{x} - \mathbf{b}^T\mathbf{x}$$

This is precisely a *variational formulation* for the linear system $A\mathbf{x} = \mathbf{b}$. If the system of equations $A\mathbf{x} = \mathbf{b}$ is large[48] the solution $\widetilde{\mathbf{x}} = A^{-1}\mathbf{b}$ may not be easy to compute, in the event of which, iterative (and sometimes approximate) solutions to the optimization problem can be helpful. One of the most succesful techniques for solving such systems (without inverting matrices) is the conjugate gradient method, discussed in Section 4.5.8, that solves the system in exactly $O(n)$ steps.

4. As will be seen on page 432 in Section 7.4, the bernoulli distribution can be expressed as
$$p(\mathbf{x}, \eta) = \exp\{\eta x - A(\eta)\}$$
for $X \in \{0, 1\}$. $A(\eta) = \log(1 + e^{\eta})$. We saw in Section 7.4 that the mean is given by
$$\overline{\mu} = E_{\eta} = \nabla A(\eta) = \frac{e^{\eta}}{1 + e^{\eta}} = (1 + e^{-\eta})^{-1}$$
The key is in noting that $\overline{\mu}$ corresponds to the 'slope of' a supporting hyperplane (see Figure 4.38 on page 271) for $epi(A(\eta))$ in a $(\eta, A(\eta))$ space. Thus, we are interested in all the hyperplanes that lie below $epi(A(\eta))$, with intercept $C$ along the axis for $A(\eta)$:
$$\mu^T\eta - C \leq A(\eta)$$
and want to get as close to a supporting hyperplane as possible
$$C^* = \sup_{\eta} \{\mu^T\eta - A(\eta)\} = A^*(\mu)$$
Borrowing ideas from duality theory (*c.f.* Section 4.4), we call the function $\sup_{\eta} \{\mu^T\eta - A(\eta)\}$ as the dual function $A^*(\mu)$. $C^*$ is the intercept for the supporting hyerplane. In the case of the bernoulli example, we have
$$A^*(\mu) = \sup_{\eta} \{\mu^T\eta - \log 1 + e^{\eta}\} = \begin{cases} (1-\mu)\log(1-\mu) + \mu\log\mu & \text{if } \mu \in (0,1) \\ \infty & \text{otherwise} \end{cases}$$
Under nice conditions (such as under *Slaters constraint qualification* discussed in definition 42 on page 290), the operation of taking duals is symmetric $((A^*)^* = A)$, that is,
$$A(\theta) = \sup_{\mu} \{\mu^T\eta - A^*(\mu)\}$$
Under the conditions of zero duality gap, it should happen that if
$$\widehat{\mu}(\eta) = \operatorname*{argmax}_{\mu} \{\mu^T\eta - A^*(\mu)\}$$

[48]Of course, as we saw in Section 7.5, such a system comes up in the case of solution to Kalman filters, but can be computed efficiently by exploiting the tree structure of the graph in inverting the matrix. The discussion here is for general graphs.

is the primal optimum, then $\widehat{\mu}^T\eta - A^*(\widehat{\mu})$ is the supporting hyperplane to $A(\eta)$, meaning that $\widehat{\mu}$ is the mean we were seeking. This yields a variational representatation for the original problem of finding the mean. The dual itself is also useful in determining the log-normalization constant for problems such as parameter estimation. We can confirm in the simple bernoulli case that indeed

$$\widehat{\mu}(\eta) = \overline{\mu}(\eta) = \frac{e^\eta}{1+e^\eta}$$

We will now generalize the variational formulation of the bernoulli case to the exponential family.

$$p(\mathbf{x};\theta) = \exp\left\{\theta^T\mathbf{f}(\mathbf{x}) - A(\theta)\right\}$$

where, $\mathbf{x} \in \Re^n$ and $\mathbf{f} : \Re^n \to \Re^d$. Let us say we are interested in computing the first order moments

$$\mu = E[\mathbf{f}(\mathbf{x})] \tag{7.74}$$

Following a similar line of argument as for the case of the bernoulli distribution, we define the dual as

$$A^*(\mu) = \sup_{\theta} \left\{\mu^T\theta - A(\mu)\right\}$$

The key ingredients in this calculation are to set the gradient with respect to $\theta$ to $\mathbf{0}$, as a necessary first order condition.

$$\mu - \nabla A(\theta) = \mathbf{0} \tag{7.75}$$

This looks like the moment matching problem that results from maximum likelihood estimation. The only difference is that $\mu$ need not come from data, it is the argument to $A^*(\eta)$. When will (7.75) have a solution? In the simple bernoulli case, we already saw that we will have a solution $\eta = -\log\mu - \log 1 - \mu$ if $\mu \in (0,1)$. As another example, for the univariate Gaussian, $\mathbf{f}(\mathbf{x}) = [x, x^2]$ and $A(\eta) = \frac{1}{2\sigma^2}\mu^2 + \log\sigma \equiv -\frac{\eta_1^2}{4\eta_2} + \frac{1}{2}\log(-2\eta_2)$ (as seen on page 432). The system (7.75) can be shown to have a solution only if $\mu_2 - \mu_1^2 > 0$, that is if the variance is positive. For two examples, the conditions under which solutions exist to (7.75) are extremely simple - it should be possible to generate data using the distribution.

Assuming that a solution $\theta(\mu)$ exists to (7.75), and exploiting the fact that $\theta(\mu)$ satisfies moment matching conditions (7.75)

$$\sum_{\mathbf{x}} p(\mathbf{x};\theta(\mu))\mathbf{f}(\mathbf{x}) = \mu \tag{7.76}$$

and using the property that

$$A(\mu) = \sum_{\mathbf{x}} A(\mu) p(\mathbf{x}; \theta(\mu))$$

the dual will have the form

$$\begin{aligned}
A^*(\mu) &= \theta^T(\mu)\mu - A(\mu) \\
&= \theta^T(\mu)\left(\sum_{\mathbf{x}} p(\mathbf{x};\theta(\mu))\mathbf{f}(\mathbf{x})\right) - A(\mu) \\
&= \sum_{\mathbf{x}} p(\mathbf{x};\theta(\mu))\left(\theta^T(\mu)\mathbf{f}(\mathbf{x}) - A(\mu)\right) \\
&= \sum_{\mathbf{x}} p(\mathbf{x};\theta(\mu)) \log p(\mathbf{x};\theta(\mu)) \\
&= -H(p(\mathbf{x};\theta(\mu))) \qquad (7.77)
\end{aligned}$$

That is, the dual is precisely the negative entropy $-H(p(\mathbf{x};\theta(\mu)))$ of the distribution whose parameters $\theta(\mu)$ are obtained by moment matching. The dual for the bernoulli case which resembled an entropy was by no means a coincidence! If $\mathcal{M}$ is the set of all possible moments that make a solution to the system (7.75 or equivalently 7.76) feasible, that is

$$\mathcal{M} = \left\{ \mu \in \Re^d \,\middle|\, \sum_{\mathbf{x}} p(\mathbf{x};\theta)\mathbf{f}(\mathbf{x}) = \mu \ for\ some\ p(.) \right\}$$

then the dual could also be expressed as

$$A^*(\mu) = \begin{cases} -\max\ H(p(\mathbf{x};\theta(\mu))) & \text{such that } E[\mathbf{f}(\mathbf{x})] = \mu \text{ for } \mu \in \mathcal{M} \\ \infty & \text{otherwise} \end{cases}$$

Another way of characterizing $\mathcal{M}$ is as the set of all first order moments that could be generated by $p(\mathbf{x};\theta)$. In any case, $\mathcal{M}$ is often very had to characterize and loop belief propagation *etc.* are often used to approximate it.

Finally, we will write down the variational problem as a reformulation of (7.74), of which mean field, (loopy) sum-product, Gauss Seidel, Jacobi, etc can be found to be special cases. Writing down the dual of the dual of (7.77), and assuming zero duality gap, we get a reformulation of (7.74):

$$A(\theta) = \sup_{\mu \in \mathcal{M}} \left\{ \mu^T\theta - A^*(\mu) \right\} \qquad (7.78)$$

Again, $A(\theta)$ is a very hard function to compute, mainly because $\mathcal{M}$ is simple to characterize. This maximumization problem is concave, since $A^*(\eta)$ is concave and the constraint set $\mathcal{M}$ is convex. Under zero duality gap conditions (which holds in this case), the optimal point will be achieved at $\widehat{\mu}(\theta) = E[\mathbf{f}(\mathbf{x})]$.

We will us take some examples to illustate the use of variational techniques. For problems of estimating moments, **f** could be the feature functions. For problems of estimating marginals, **f** can be chosen to be the indicator function.

The simplest example is for a two node chain: $\mathcal{V} = \{X_1, X_2\}$, $\mathcal{E} = \{(X_1, X_2)\}$, $X_1, X_2 \in \{0, 1\}$ and

$$p(\mathbf{x}; \theta) \propto \exp\{\theta_1 x_1 + \theta_2 x_2 + \theta_{12} x_1 x_2\}$$

The moments are: $\mu_i = E[X_i] = p(X_i = 1)$ and $\mu_{12} = E[X_1 X_2] = p(X_1 = 1, X_2 = 1)$. The set $\mathcal{M}$ is

$$\mathcal{M} = \left\{ \mu \in \Re^3 \left| \sum_{\mathbf{x}} p(\mathbf{x}; \theta)\mathbf{f}(\mathbf{x}) = \mu \; for\ some\ p(.) \right. \right\} = \{\mu_i \in [0,1],\ 0 \le \mu_{12} \le \min(\mu_1, \mu_2),\ 1 + \mu_{12} - \mu_1 - \mu_2 \ge 0\}$$

Let us next write down the dual in terms of the entropy of the distribution

$$\begin{aligned} A^*(\mu) = -H(p(\mathbf{x}; \mu)) &= \sum_{x_1, x_2} p(x_1, x_2) \log p(x_1, x_2) \\ &= \mu_{12} \log \mu_{12} + (\mu_1 - \mu_{12}) \log(\mu_1 - \mu_{12}) + (\mu_2 - \mu_{12}) \log(\mu_2 - \mu_{12}) \\ &+ (1 + \mu_{12} - \mu_1 - \mu_2) \log(1 + \mu_{12} - \mu_1 - \mu_2) \end{aligned} \tag{7.79}$$

The corresponding variational problem will be

$$A(\theta) = \max_{\mu \in \mathcal{M}} \{\theta_1 \mu_1 + \theta_2 \mu_2 + \theta_{12} \mu_{12} - A^*(\mu)\}$$

Though this can be solved using the method of Lagrange multipliers (*c.f.*, Section 4.4.1), *etc.*, we expect the optimal solution to the variational problem to be

$$\widehat{\mu}_1 = \frac{1}{z} \sum_{x_1 \in \{0,1\}, x_2 \in \{0,1\}} x_1 \exp\{\theta_1 x_1 + \theta_2 x_2 + \theta_{12} x_1 x_2\} = \frac{\exp \theta_1 + \exp \theta_1 + \theta_2 + \theta_{12}}{1 + \exp \theta_1 + \exp \theta_2 + \exp \theta_1 + \theta_2 + \theta_{12}}$$

There are many applications of variational inference to quantity estimation problems that have either no exactly solutions, or that have solutions not computable in polynomial time. Further, variational principles can be used to study how the approximate algorithms behave; whether they have fix points, whether they converge. what answers do they give, *etc.*. For instance, in belief propagation from a variational perspective, the messages correspond to lagrange multipliers (for active constraints in $\mathcal{M}$) that are passed around.

In general, the sets $\mathcal{M}$ are very complex. For example, with a complete 7 node graph, there will be $O(3 \times 10^8)$ constraints. For an $n$ node tree, you will have $4(n-1)$ constraints. The variational principle provides the foundation for many approximate methods such as the Naive mean field algorithm, which restricts optimization to a 'tractable' set of $\mathcal{M}$, such as one for which the joint distribution over a graphical model is factorized by treating the variables as completely independent.