

Probabilistic Graphical Models: Representation, Inference and Learning

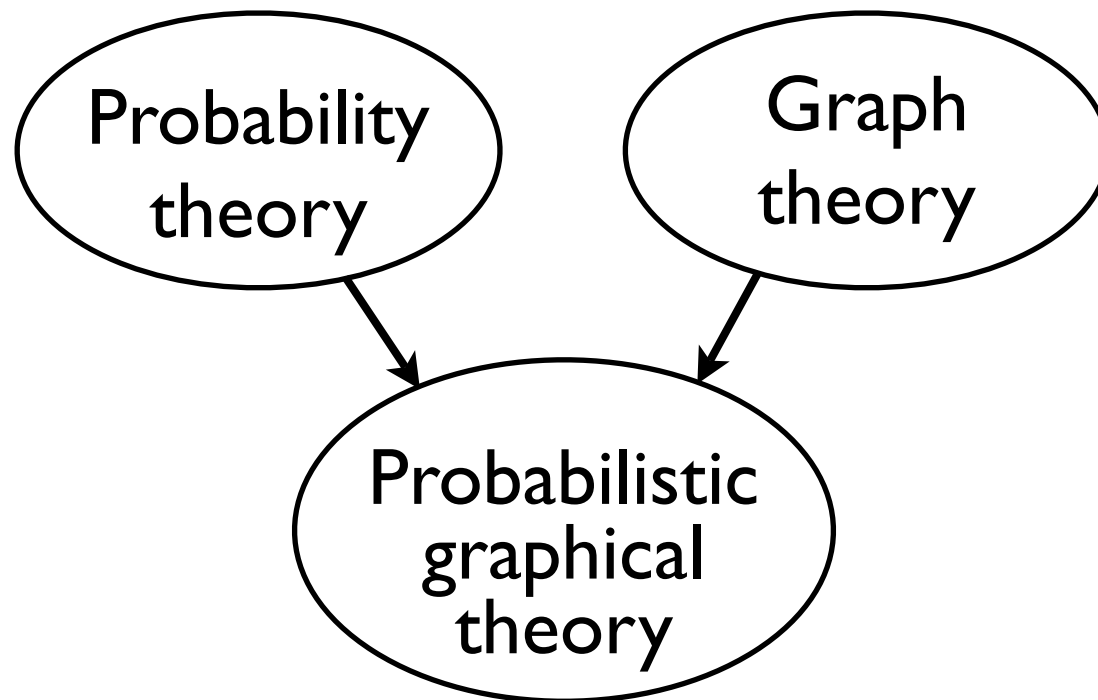
Aaron Courville
Université de Montréal

Overview

- Graphs and probabilities
 - Conditional independence
 - Directed graphs
 - Examples
- Inference
 - Belief propagation
 - Approximate inference
- Learning
 - Maximum likelihood I: fully observable case
 - Maximum likelihood II: Expectation Maximum

Probabilistic Graphical Models

- Graphs endowed with a probability distribution
 - nodes represent random variables and the edges encode conditional independence assumptions
- Graphical model express sets of conditional independence via graph structure (and conditional independence is useful)
- Graph structure plus associated parameters define joint probability distribution of the set of nodes/variables



Graphical Models

- Graphical models come in two main flavors:
 1. Directed graphical models (a.k.a Bayes Net, Belief Networks):
 - consists of a set of nodes with arrows (directed edges) between some of the nodes
 - arrows encode factorized conditional probability distributions
 2. Undirected graphical models (a.k.a Markov random fields):
 - consists of a set of nodes with undirected edges between some of the nodes
 - edges (or more accurately the lack of edges) encode conditional independence.
- Today, we will focus exclusive on directed graphs.

Probability Review: Marginal Independence

Definition: X is marginally independent of Y if for all (i, j)

$$\begin{aligned}P(X = x_i, Y = y_j) &= P(X = x_i)P(Y = y_j) \\ P(X, Y) &= P(X)P(Y)\end{aligned}$$

This is equivalent to the expression:

$$P(X | Y) = P(X) \quad P(Y | X) = P(Y)$$

Why? Recall from the probability product rule

$$P(X, Y) = P(X | Y)P(Y) = P(Y | X)P(X) = P(X)P(Y)$$

Probability Review: Conditional Independence

Definition: X is conditionally independent of Y given Z if the probability distribution governing X is independent of the value of Y , given the value of Z : for all (i, j, k)

$$\begin{aligned}P(X = x_i, Y = y_j \mid Z = z_k) &= P(X = x_i \mid Z = z_k)P(Y = y_j \mid Z = z_k) \\P(X, Y \mid Z) &= P(X \mid Z)P(Y \mid Z)\end{aligned}$$

Or equivalently (by the product rule):

$$P(X \mid Y, Z) = P(X \mid Z) \quad P(Y \mid X, Z) = P(Y \mid Z)$$

Why? Recall from the probability product rule

$$P(X, Y, Z) = P(X \mid Y, Z)P(Y \mid Z)P(Z) = P(X \mid Z)P(Y \mid Z)P(Z)$$

Example: $P(\textit{Thunder} \mid \textit{Rain}, \textit{Lightning}) = P(\textit{Thunder} \mid \textit{Lightning})$

Directed Graphical Models

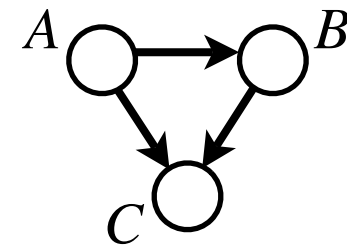
- Set of nodes with arrows (directed edges) between some of the nodes
 - Nodes represent random variables
 - Arrows encode factorized conditional probability distributions

- Consider an arbitrary joint distribution:

$$P(A, B, C)$$

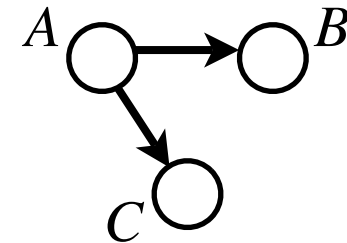
- By application of the product rule:

$$\begin{aligned} P(A, B, C) &= P(A)P(B, C | A) \\ &= P(A)P(B | A)P(C | A, B) \end{aligned}$$

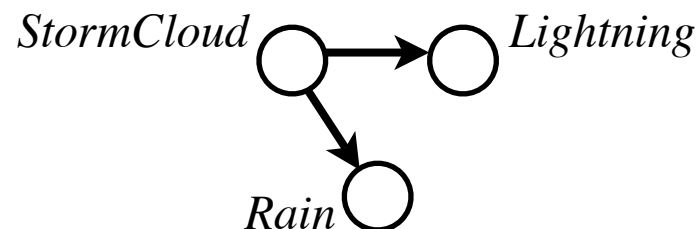


- What if C is conditionally independent from B :

$$P(A, B, C) = P(A)P(B | A)P(C | A)$$



E.g., $P(\text{Rain}, \text{Lightning} | \text{StormCloud}) = P(\text{Rain} | \text{StormCloud})P(\text{Lightning} | \text{StormCloud})$

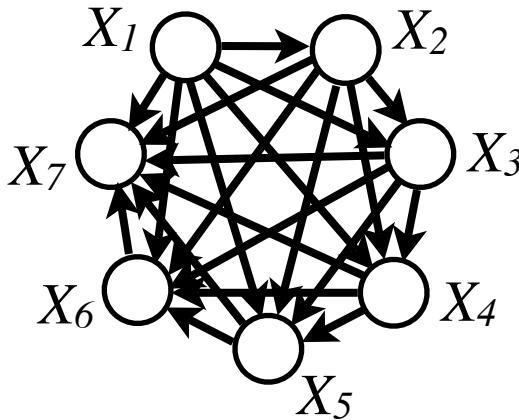


The General Case

- Consider the arbitrary joint distribution: $P(X_1, \dots, X_d)$
- By successive application of the product rule

$$P(X_1, \dots, X_d) = P(X_1)P(X_2 | X_1) \cdots P(X_d | X_1, \dots, X_{d-1})$$

- Can be represented by a graph in which each node has links from all lower-numbered nodes (i.e. *a fully connected graph*)



- *Directed acyclic graph*: NO DIRECTED CYCLES \Rightarrow can number nodes so that there are no links from higher numbered to lower numbered nodes.

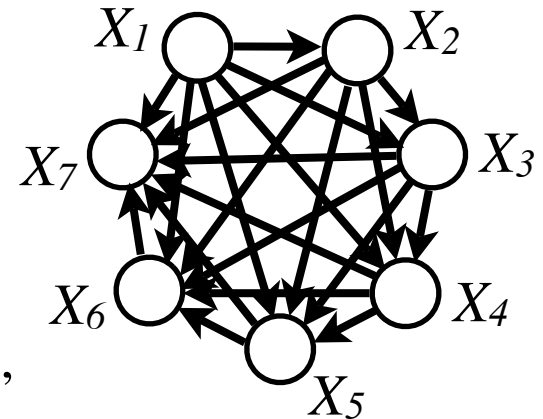
Directed Acyclic Graphs (DAGs)

- General factorization:

$$P(X_1, \dots, X_d) = \prod_{i=1}^d P(X_i \mid Pa_i)$$

where Pa_i denotes the immediate parents of node i ,

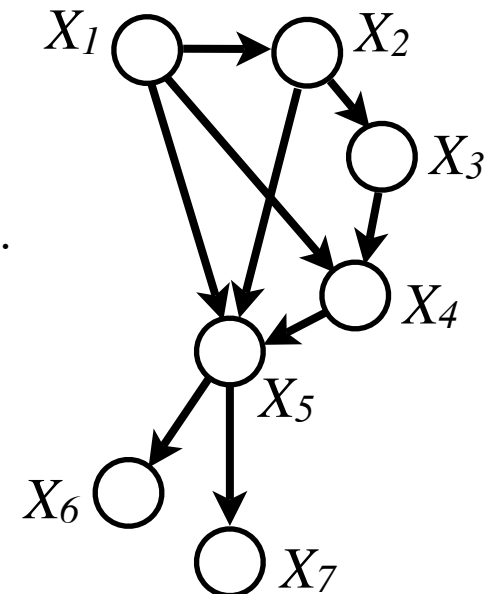
- i.e. all the nodes with arrows leading to node i .



- Node X_i is conditionally independent of its non-descendants given its immediate parents (missing links imply conditional independence).

- Some terminology:

- Parents of node i = nodes with arrows **leading to** node i .
- Antecedents = parents, parents of parents, etc.
- Children of node i = nodes with arrows **leading from** node i .
- Descendants = children, children of children, etc.



Importance of Ordering

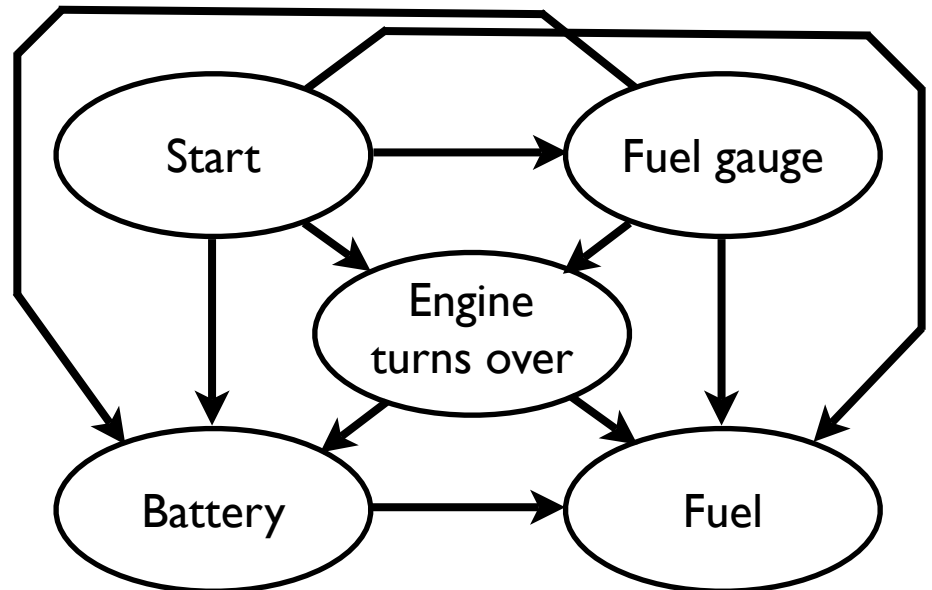
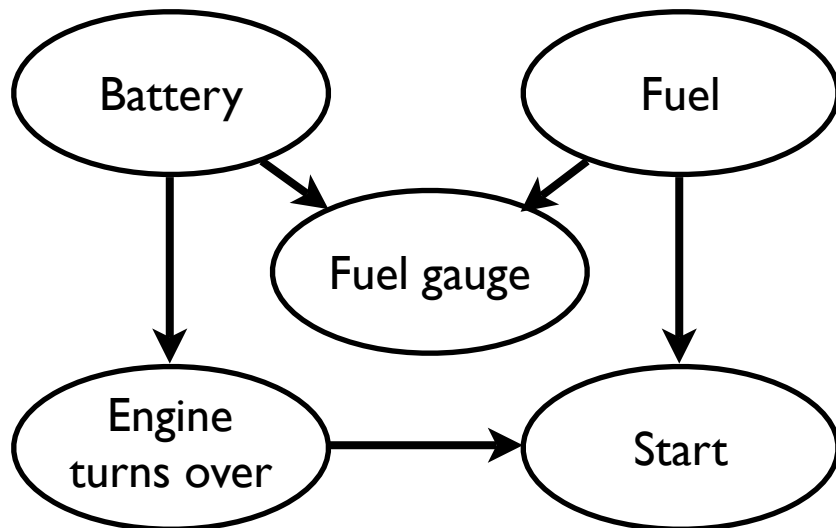
- Since *every* joint probability distribution can be decomposed as

$$P(X_1, \dots, X_d) = \prod_{i=1}^d P(X_i \mid Pa_i)$$

- then every joint probability can be represented as a graphical model
 - with arbitrary ordering over the variables
 - i.e. any decomposition will generate a valid graph

- However **not all decompositions are created equal.**

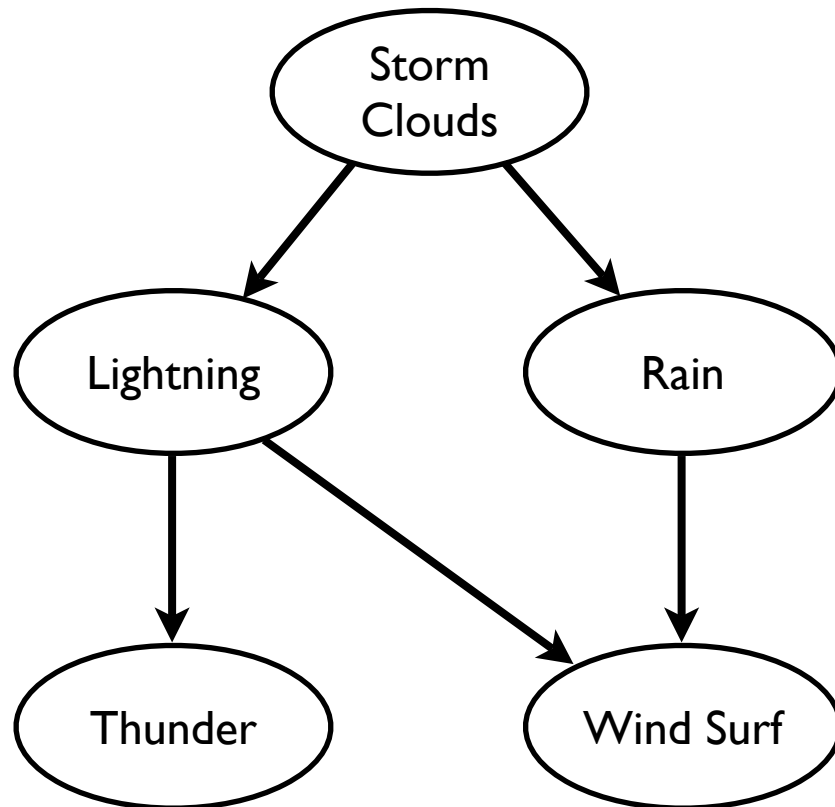
- An illustration:



DAG Example (Bayes Net)

A conditional probability distribution (CPD) is associated with each node.

- Defines $P(X_i | Pa_i)$.
- Specifies how the parents interact.



CPD for node: *Lightning*

Pa	$P(L SC)$	$P(\sim L SC)$
SC	0.7	0.3
$\sim SC$	0	1.0

CPD for node: *Wind Surf*

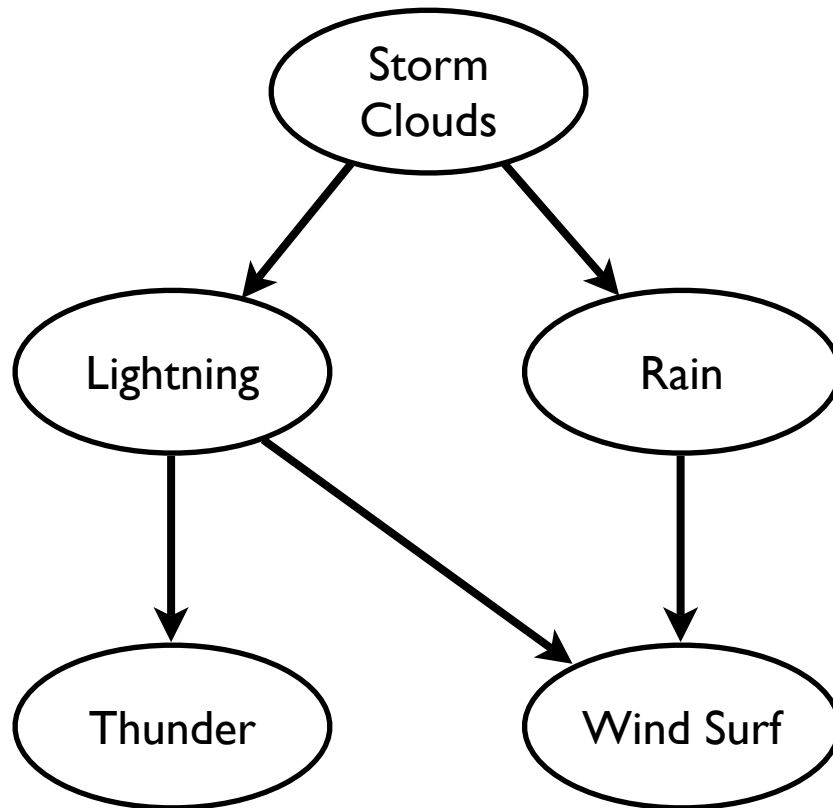
Pa	$P(WS Pa)$	$P(\sim WS Pa)$
L, R	0	1.0
L, $\sim R$	0	1.0
$\sim L$, R	0.2	0.8
$\sim L$, $\sim R$	0.9	0.1

$$\begin{aligned}
 P(SC, L, R, T, WS) &= \prod_{i=1}^d P(\text{node}_i | Pa_i) \\
 &= P(SC)P(L | SC)P(R | SC)P(T | L)P(WS | L, R)
 \end{aligned}$$

Example I: Bayes Net (cont.)

Let's count the parameters

- In the full joint distribution: $2^5 - 1 = 31$
- Taking advantage of conditional independences: 11



CPD for node: *Lightning*

Pa	$P(L SC)$	$P(\sim L SC)$
SC	0.7	0.3
$\sim SC$	0	1.0

CPD for node: *Wind Surf*

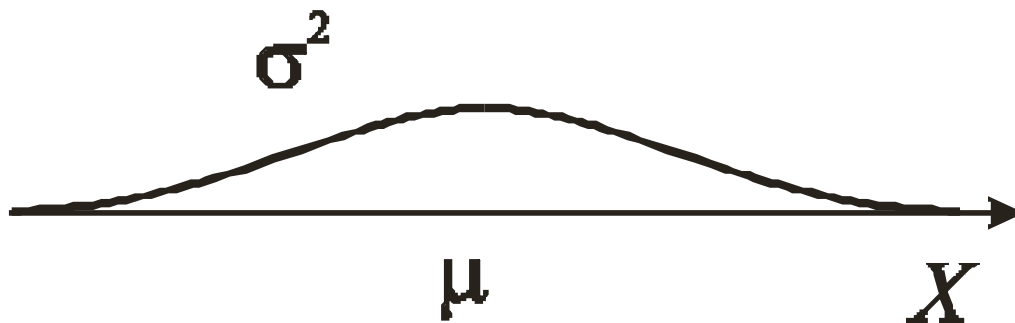
Pa	$P(WS Pa)$	$P(\sim WS Pa)$
L, R	0	1.0
L, $\sim R$	0	1.0
$\sim L$, R	0.2	0.8
$\sim L$, $\sim R$	0.9	0.1

$$P(SC, L, R, T, WS) = P(SC)P(L | SC)P(R | SC)P(T | L)P(WS | L, R)$$

Example: Univariate Gaussian

- Consider a univariate Gaussian random variable as a (simple) graphical model.

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



Graphical model:

$X \circ$

Example II: Mixture of Gaussians

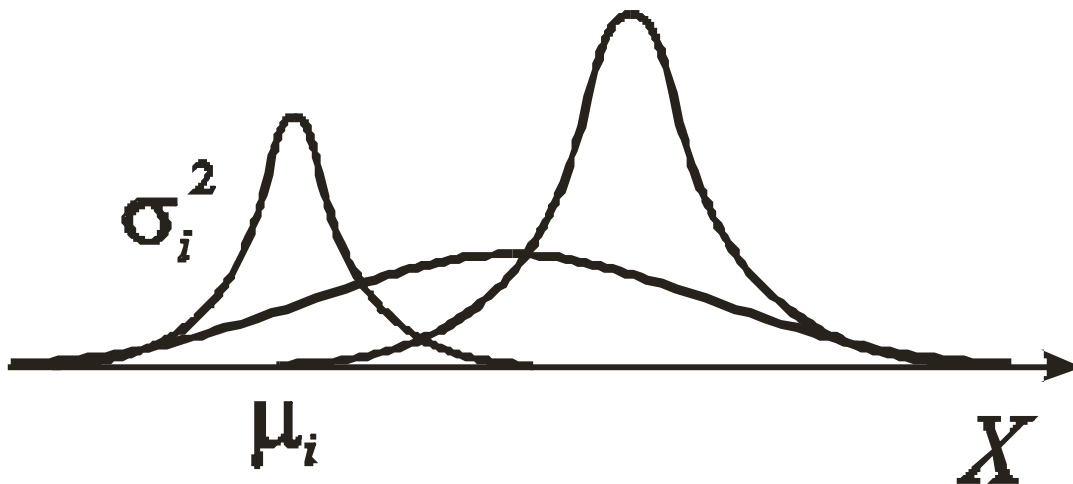
- Now let's consider a random variable distributed according to a mixture of Gaussians.
- **Conditional distributions:**

$$P(I = i) = w_i$$
$$p(X = x | I = i) = \mathcal{N}(x | \mu_i, \sigma_i^2) = \frac{1}{\sigma_i \sqrt{2\pi}} \exp\left(-\frac{(x - \mu_i)^2}{2\sigma_i^2}\right)$$

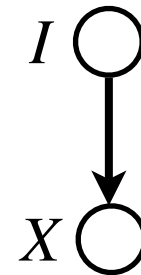
where I is an index over the Gaussian components in the mixture and the mixing proportion, w_i , is the marginal probability that X is generated by mixture component i .

- **Marginal distributions:**

$$p(X = x) = \sum_i p(X = x | I = i)P(I = i) = \sum_i w_i \mathcal{N}(x | \mu_i, \sigma_i^2)$$



Graphical model:



Example III: Naïve Bayes

- Consider the classification problem:

X
 Y

Training data:

	Sky	Temp	Humid	Wind	Water	Forecst	EnjoySpt
	Sunny	Warm	Normal	Strong	Warm	Same	Yes
	Sunny	Warm	High	Strong	Warm	Same	Yes
	Rainy	Cold	High	Strong	Warm	Change	No
	Sunny	Warm	High	Strong	Cool	Change	Yes

- For a new example X^{new} , We want to determine $P(Y | X^{new})$ so that we can classify Y^{new} as $\text{argmax}_j P(Y = j | X^{new})$.
- One way to do this is to use Bayes Rule:

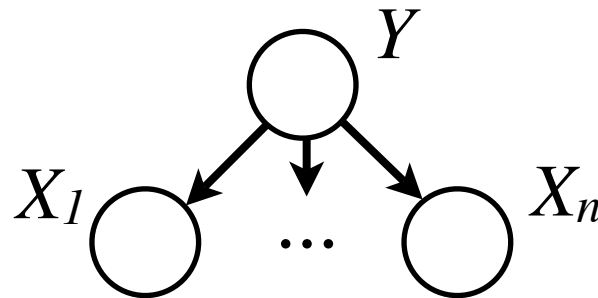
$$P(Y | X) = \frac{P(X | Y)P(Y)}{P(X)}$$

Example III: Naïve Bayes

- Using Bayes rule, to compute $P(Y | X)$ we need to represent $P(X | Y)$ and $P(Y)$.
 - For even marginally large X , the full joint $P(X_1, X_2, \dots, X_n | Y)$ is impractical
 - We would require a parameter for each unique combination of X_1, \dots, X_n .
- Naïve Bayes assumption:

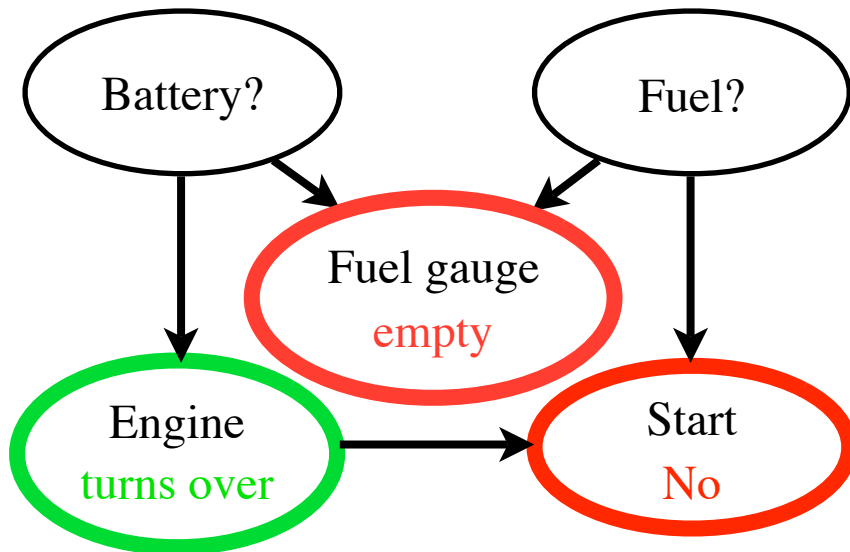
$$P(X | Y) = P(X_1, \dots, X_n | Y) = \prod_i P(X_i | Y)$$

- X_i and X_j ($i \neq j$) are conditionally independent given Y .



Conditioning on Evidence

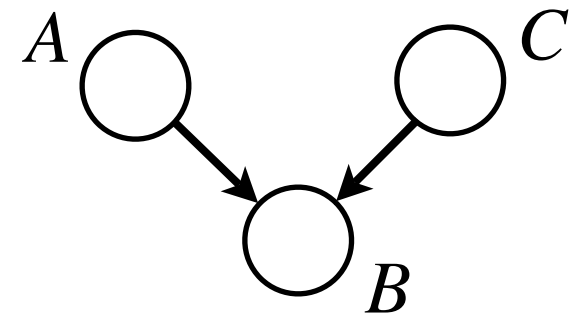
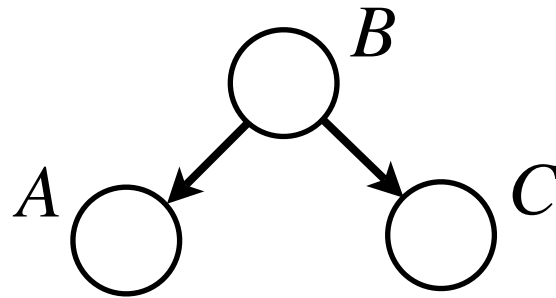
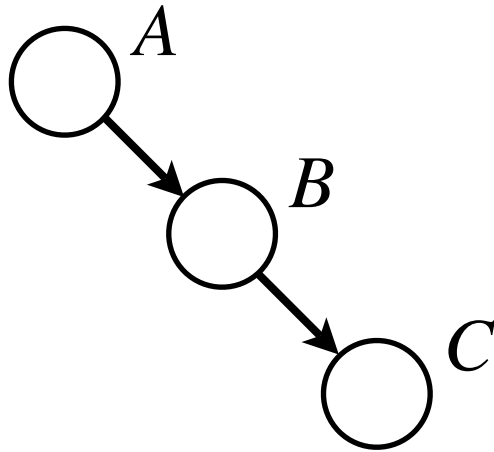
- Much of the point of Bayes Nets (and Graphical models, more generally) is to use data or observed variables to *infer* the values of hidden (or latent) variables.
- We can condition on the observed variables and compute conditional distributions over the latent variables.



Note: hidden variables may have a specific interpretation (such as Battery and Fuel), or may be introduced to permit a richer class of distributions (as in neural network hidden units).

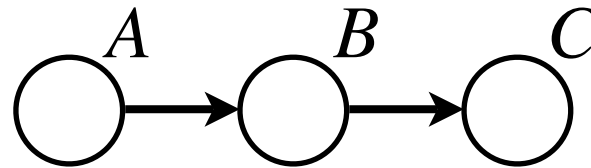
Markov Properties

- We have said that the conditional independence properties are represented directly from the graph.
- The pattern of conditional independence turn out to be extremely important for determining how to calculate the conditional distributions over variables of interest.
- We will now consider three simple independence structures that form the basic building blocks of all independence relationships representable by directed graphical models.



Markov Properties I

- Consider the joint distribution, $P(A,B,C)$ specified by the graph:

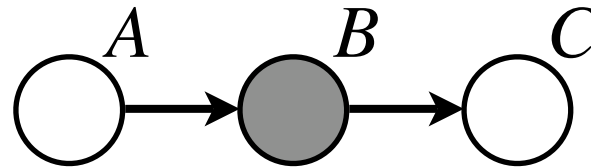


- Note the missing edge from A to C .
- Node B is “head-to-tail” with respect to the path A - B - C .
- Joint distribution:

$$\begin{aligned} P(A, B, C) &= P(A)P(B | A)P(C | A, B) \\ &= P(A)P(B | A)P(C | B) \end{aligned}$$

Markov Properties I (cont.)

- Suppose we condition on the node B :



- A and C are conditionally independent, given B : $A \perp\!\!\!\perp C \mid B$

$$\begin{aligned} P(A, C \mid B) &= \frac{P(A, B, C)}{P(B)} \\ &= \frac{P(A)P(B \mid A)P(C \mid B)}{P(B)} && \text{[By definition of joint]} \\ &= \frac{P(A)P(B \mid A)}{P(B)} P(C \mid B) && \text{[By Bayes rule]} \\ &= P(A \mid B)P(C \mid B) \end{aligned}$$

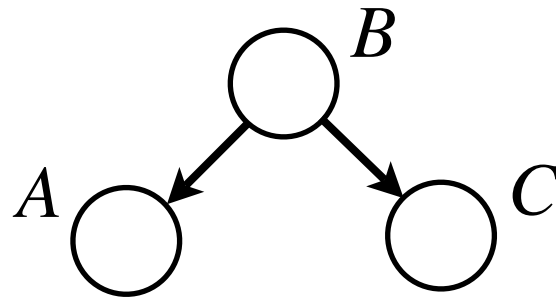
- Note that if B is not observed: $A \not\perp\!\!\!\perp C \mid \emptyset$

$$P(A, C) = \sum_B P(A, C \mid B)P(B) = \sum_B P(A \mid B)P(C \mid B)P(B)$$

- Informally: observing B “blocks the path” from A to C .

Markov Properties II

- Consider the graph:

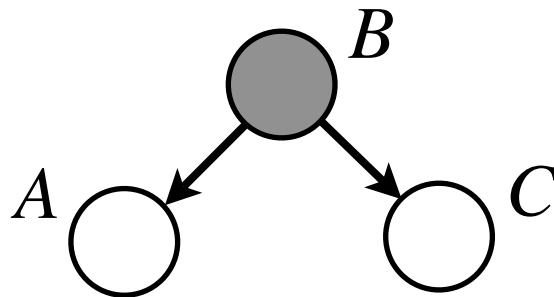


- Again, note missing edge from A to C .
- Node B is “tail-to-tail” with respect to the path A - B - C .
- Joint distribution:

$$\begin{aligned} P(A, B, C) &= P(B)P(A | B)P(C | A, B) \\ &= P(B)P(A | B)P(C | B) \end{aligned}$$

Markov Properties II (cont.)

- Again, let's condition on node B :



- A and C are conditionally independent, given B: $A \perp\!\!\!\perp C \mid B$

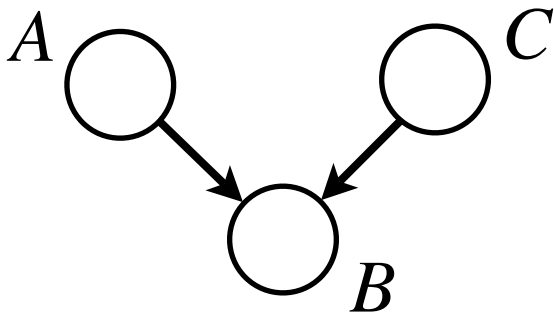
$$P(A, C \mid B) = P(A \mid B)P(C \mid B)$$

- Again, if B is not observed: $A \not\perp\!\!\!\perp C \mid \emptyset$

$$P(A, C) = \sum_B P(A, C \mid B)P(B) = \sum_B P(A \mid B)P(C \mid B)P(B)$$

- Informally: observing B “blocks the path” from A to C .

Markov Properties III

- Consider the graph: 

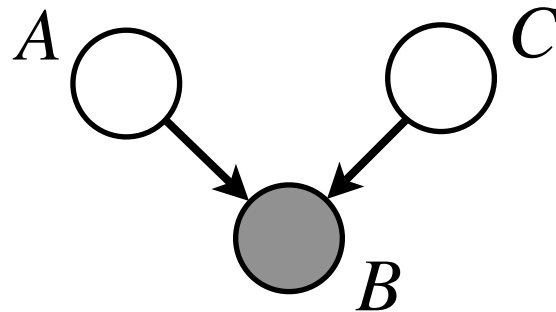
- Once again, note missing edge from A to C .
- Node B is “head-to-head” with respect to the path A - B - C .
- Joint distribution: $P(A, B, C) = P(A)P(C)P(B | A, C)$
- If B is not observed, we have: $A \perp\!\!\!\perp C | \emptyset$

$$\begin{aligned} P(A, C) &= \sum_B P(A)P(C)P(B | A, C) \\ &= P(A)P(C) \sum_B P(B | A, C) \\ &= P(A)P(C) \end{aligned}$$

Markov Properties III (cont.)

The Famous: “Explaining Away Effect”

- Suppose, once again, we condition on node B :



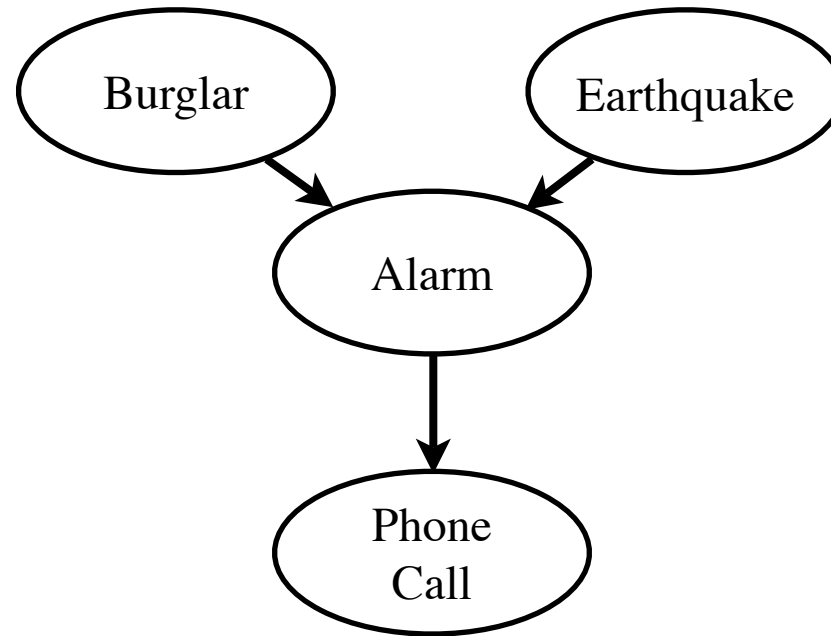
- A and C are *not* conditionally independent, given B : $A \not\perp C \mid B$

$$\begin{aligned} P(A, C \mid B) &= P(A \mid B, C)P(C \mid B) \\ &= P(C \mid B, A)P(A \mid B) \\ &\neq P(A \mid B)P(C \mid B) \end{aligned}$$

- Informally, an *unobserved* head-to-head node B “blocks the path” from A to C , but once B is observed the path is unblocked.
- **Important Note:** observation of *any descendent* of B also “unblocks the path.”

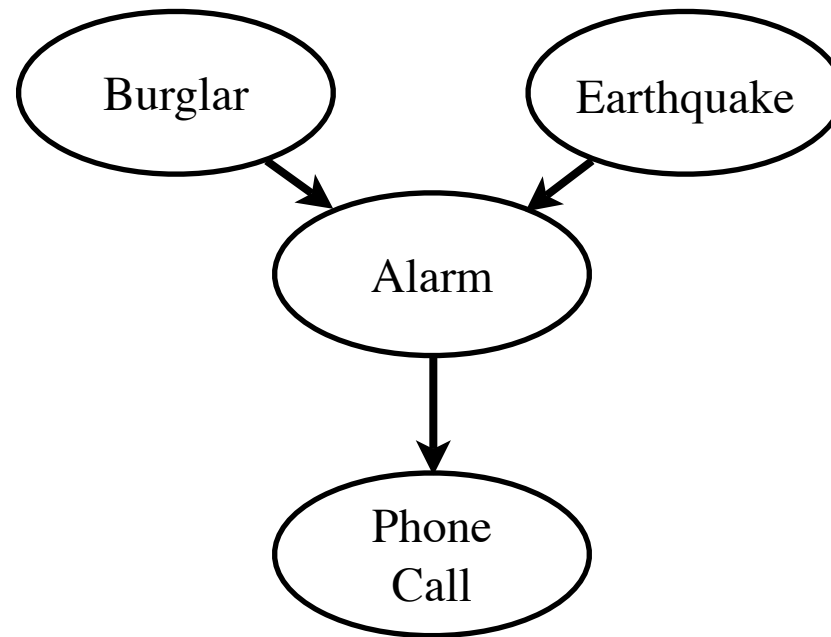
Explaining Away: An Illustration

- Consider the “Burglar Alarm” problem:



- Your house has a twitchy burglar alarm that is sometimes triggered by earthquakes.
- Earth doesn't care whether your house is currently being burgled
- While on vacation, one of your neighbours call and tells you that your home's burglar alarm is ringing!

The plot thickens...

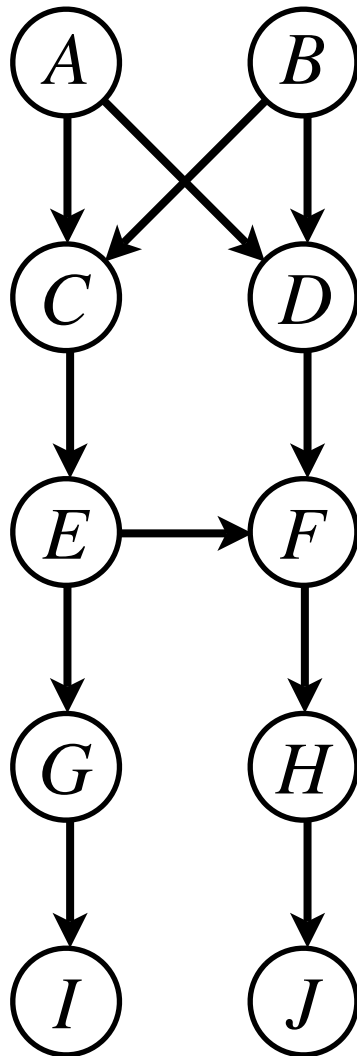


- But now suppose you learn that there was a small earthquake in your neighbourhood. Whew! Probably not a burglar after all.
- The Earthquake “explains away” the hypothetical burglar.
- So Burglar must not be conditionally independent of Earthquake given Alarm (or Phone Call).

d-Separation

- In a general DAG, consider three *groups* of nodes A , B , C .
- To determine whether A is conditional independent of C given B , consider all possible paths from any node in A to any node in C .
- Any such path is blocked if there is a node X which is head-to-tail or tail-to-tail with respect to the path and $X \in B$ or if the node is head-to-head and neither the node, *nor any of its descendants*, is in B .
- Note that a particular node may, for example, be head-to-head with respect to one particular path and head-to-tail with respect to a different path
- **Theorem** [Verma & Pearl, 1998]: If, given B , all possible paths from A to C are blocked (we say that B d-separates A and C), then A is conditionally independent of C , given B .
- Note: Variables may actually be independent even when not d-separated.

d-Separation Example



Conditional Independence

$C \perp\!\!\!\perp D ?$

$C \perp\!\!\!\perp D \mid A ?$

$C \perp\!\!\!\perp D \mid A, B ?$

$C \perp\!\!\!\perp D \mid A, B, J ?$

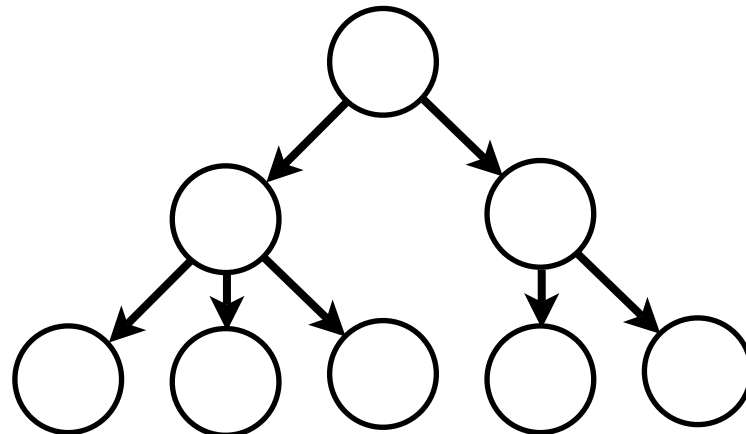
$C \perp\!\!\!\perp D \mid A, B, E, J ?$

Inference in a DAG

- Inference: calculate $P(X | Y)$ for some variable or sets of variables of interest X and observed variable or set of variables Y .
- If Z are the set of remaining variables in the graph:

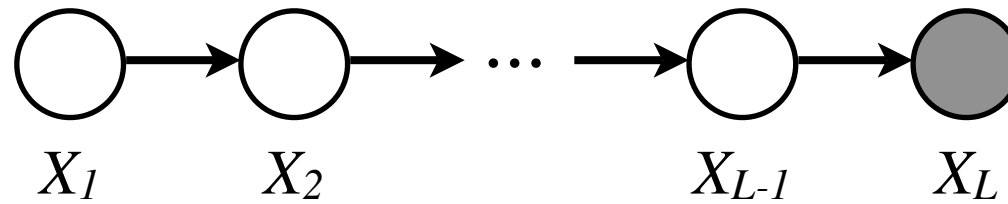
$$P(X | Y) = \sum_Z P(X | Z, Y)P(Z | Y)$$

- The bad news: exact inference in Bayes Nets (DAGs) is NP-hard!
- But exact inference is still tractable in some cases.
- Let's look at a special class of networks: *trees / forests*
 - Each node has at most one parent.



Example: Markov Chain

- Consider the graph: (a Markov Chain)



$$P(X_1, \dots, X_L) = P(X_1)P(X_2 | X_1) \dots P(X_L | X_{L-1})$$

- What if we want to find $P(X_1 | X_L)$?
- Direct evaluation gives:

$$P(X_1 | X_L) = \frac{\sum_{X_2} \dots \sum_{X_{L-1}} P(X_1 \dots X_L)}{\sum_{X_1} \sum_{X_2} \dots \sum_{X_{L-1}} P(X_1 \dots X_L)}$$

- For variables having M states, the denominator involves summing over M^{L-1} terms (exponential in the length of the chain).

Example: Markov Chain (cont.)

- Using the conditional independence structure we can reorder the summations in the denominator to give

$$\sum_{X_1} \sum_{X_2} \cdots \sum_{X_{L-1}} P(X_1 \dots X_L) = \sum_{X_{L-1}} P(X_L | X_{L-1}) \cdots \sum_{X_2} P(X_3 | X_2) \sum_{X_1} P(X_2 | X_1) P(X_1)$$

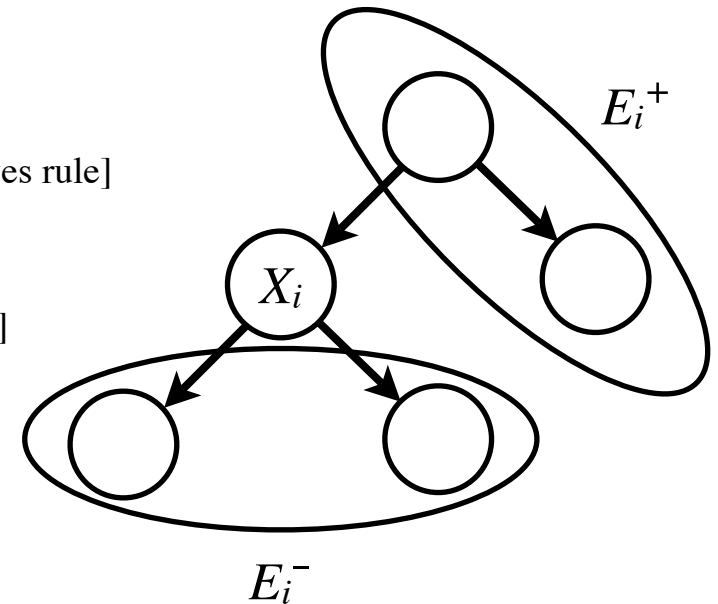
which involves on the order of LM^2 summations (linear in the length of the chain) - similarly for the numerator

- This is known as *variable elimination*, also can be interpreted as a sequence of messages being passed.

Belief Propagation: Decomposing Probabilities

- Suppose we want to compute $P(X_i | E)$ where E is some set of *evidence variables*.
 - Evidence variables are those that are observable, offering “evidence” with respect to the values of the latent variables.
- Let’s split E into two parts:
 - E_i^- is the part consisting of assignments to variables in the subtree rooted at X_i .
 - E_i^+ is the rest of it.

$$\begin{aligned}
 P(X_i | E) &= P(X_i | E_i^-, E_i^+) \\
 &= \frac{P(E_i^- | X_i, E_i^+) P(X_i | E_i^+)}{P(E_i^- | E_i^+)} && \text{[by Bayes rule]} \\
 &= \frac{P(E_i^- | X_i) P(X_i | E_i^+)}{P(E_i^- | E_i^+)} && \text{[by cond. ind.]} \\
 &= \alpha \pi(X_i) \lambda(X_i)
 \end{aligned}$$



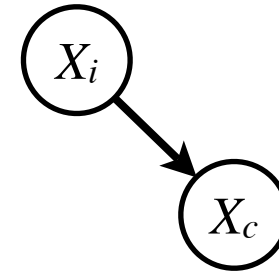
where $\pi(X_i) = P(X_i | E_i^+)$, $\lambda(X_i) = P(E_i^- | X_i)$
 and α is a constant independent of X_i .

Computing $\lambda(X_i)$ for leaves

- Starting at the leaves of the tree, recursively compute $\lambda(X_i) = P(E_i^- \mid X_i)$ for all X_i .
- If X_i is a leaf:
 - If X_i is in E (X_i is observed): $\lambda(X_i) = 1$ if X_i matches E , 0 otherwise.
 - If X_i is not in E : E_i^- is the null set, so $\lambda(X_i) = 1$ (constant).
- For simplicity, and without loss of generality, we will assume that *all* variables in E (the evidence set) are leaves in the tree.
 - How can we do this? In the case of a tree, conditioning on E d-separates the graph at that point.

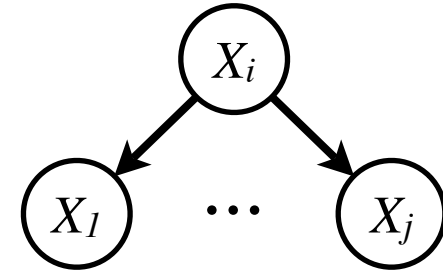
Computing $\lambda(X_i)$ for non-leaves I

- Suppose X_i has one child, X_c .



$$\begin{aligned}\lambda(X_i) &= P(E_i^- | X_i) \\ &= \sum_j P(E_i^-, X_c = j | X_i) \\ &= \sum_j P(X_c = j | X_i) P(E_i^- | X_i, X_c = j) \\ &= \sum_j P(X_c = j | X_i) P(E_i^- | X_c = j) \\ &= \sum_j P(X_c = j | X_i) \lambda(X_c = j)\end{aligned}$$

Computing $\lambda(X_i)$ for non-leaves II

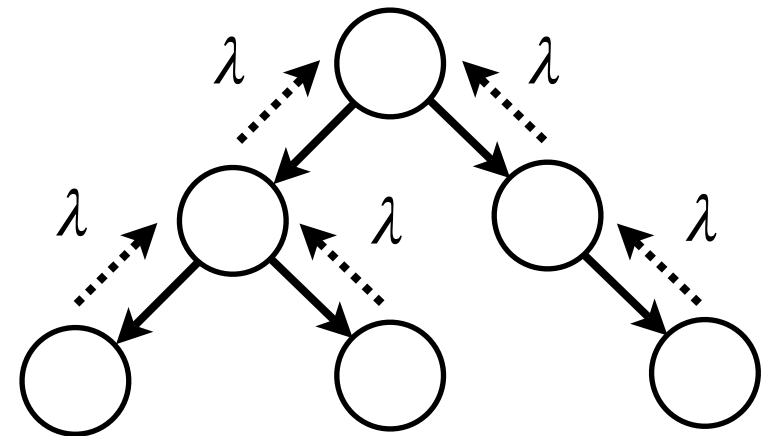


- Now suppose X_i has a set of children C .
- Since X_i d-separates each of its subtrees, the contribution of each subtree to $\lambda(X_i)$ is independent.

$$\lambda(X_i) = P(E_i^- | X_i) = \prod_{X_j \in C} \lambda_j(X_i) = \prod_{X_j \in C} \left[\sum_k P(X_j = k | X_i) \lambda(X_j) \right]$$

where $\lambda_j(X_i)$ is the contribution to $\lambda(X_i)$ of the evidence lying in the subtree rooted at one of X_i 's children X_j .

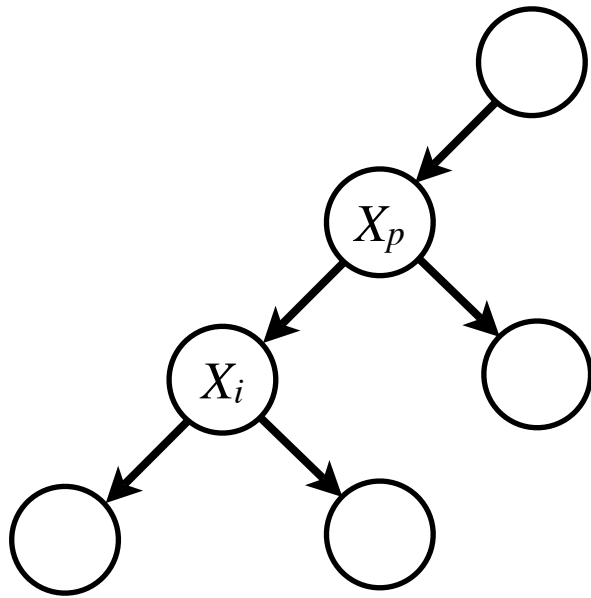
- We now have a way to recursively compute all the $\lambda(X_i)$'s.
- Each node in the network passes a little “ λ message” to its parent.



The other half of the problem

- Remember $P(X_i | E) = \alpha \pi(X_i) \lambda(X_i)$. Now that we have all the $\lambda(X_i)$'s, what about the $\pi(X_i)$'s?
 - recall: $\pi(X_i) = P(X_i | E_i^+)$
- At the root node X_r , E_r^+ is the null set, so $\pi(X_r) = P(X_r)$.
 - since we also know $\lambda(X_i)$, we can compute the final $P(X_r | E)$!
- So for an arbitrary X_i with parent X_p , we can recursively compute $\pi(X_i)$ from $\pi(X_p)$ and/or $P(X_p | E)$.
- How do we do this?

Computing $\pi(X_i)$

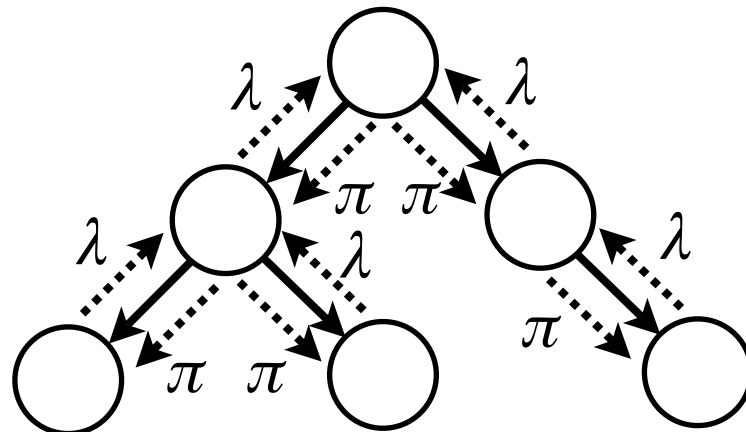


$$\begin{aligned}\pi(X_i) &= P(X_i | E_i^+) \\ &= \sum_j P(X_i, X_p = j | E_j^+) \\ &= \sum_j P(X_i | X_p = j, E_i^+) P(X_p = j | E_i^+) \\ &= \sum_j P(X_i | X_p = j) P(X_p = j | E_i^+) \\ &= \sum_j P(X_i | X_p = j) \frac{P(X_p = j | E)}{\lambda_i(X_p = j)} \\ &= \sum_j P(X_i | X_p = j) \pi_i(X_p = j)\end{aligned}$$

where $\pi_i(X_p)$ is defined as $\frac{P(X_p|E)}{\lambda_i(X_p)}$.

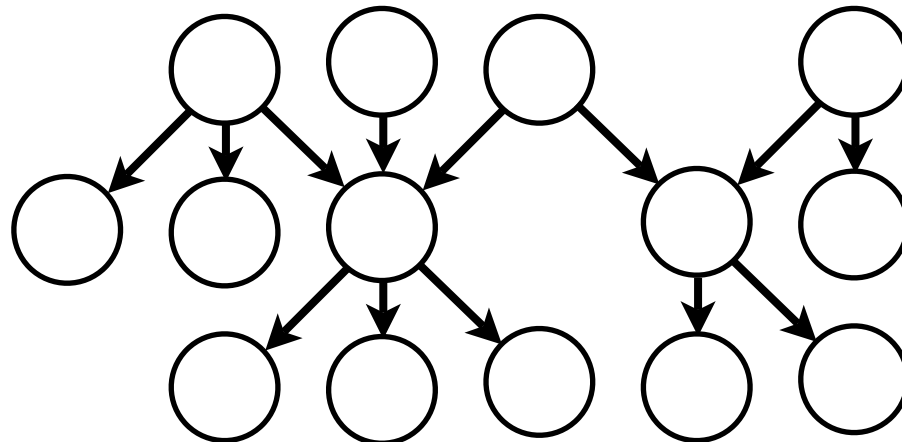
Belief Propagation Wrap-up

- Using a top down strategy, we can compute all the $\pi(X_i)$'s and, in turn, all the $P(X_i | E)$'s.
- Nodes can be thought of as autonomous processors passing λ and π messages to their neighbors.
- What if you have a conjunctive distribution, e.g., $P(A, B | C)$ instead of just marginal distributions $P(A | C)$ and $P(B | C)$?
 - Use the chain rule: $P(A, B | C) = P(A | C)P(B | A, C)$
 - Each of these latter probabilities can be computed using belief propagation.



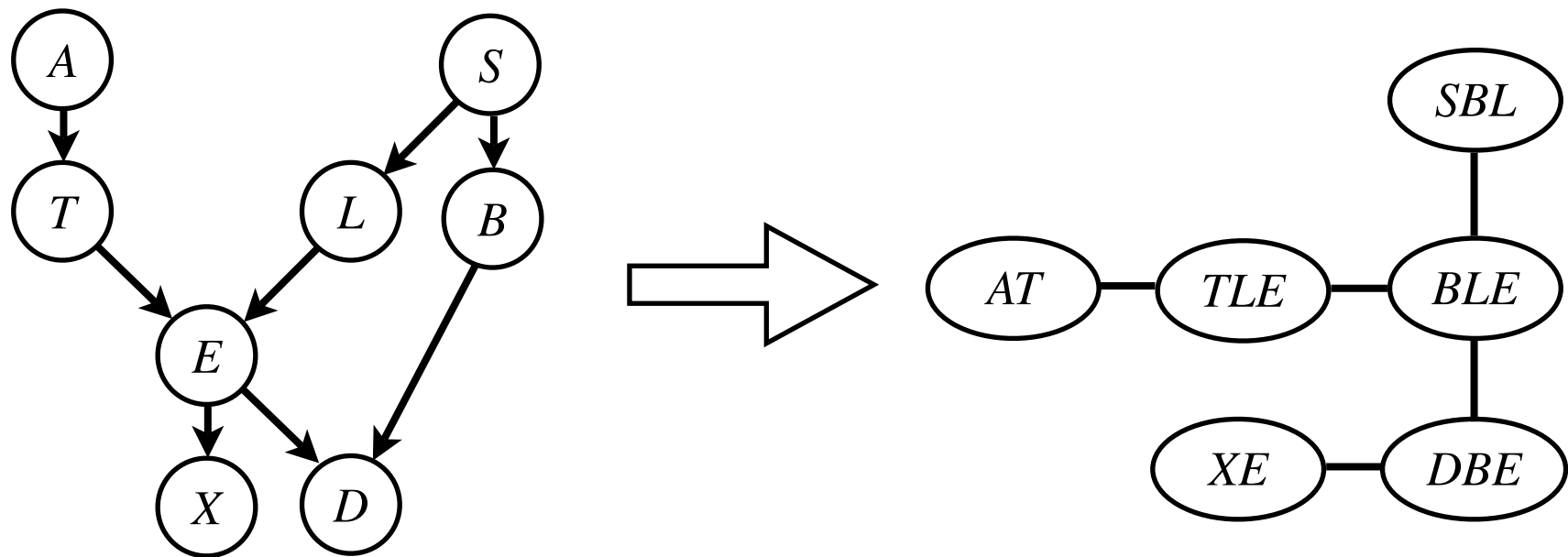
Generalizing Belief Propagation

- Great! For tree structured graphs, we have an inference algorithm with time complexity linear in the number of nodes.
- All this is fine, but what if you don't have a tree structured graph?
- The method we discussed can be generalized to *polytrees*:
 - undirected versions of the graphs are still trees, but nodes can have more than one parent.
 - **do not contain any undirected cycles.**



Dealing with Cycles

- Can deal with undirected cycles in a graph by the *junction tree algorithm*.
- An arbitrary directed acyclic graph can be transformed via some graph theoretic procedure into a tree structure with nodes containing clusters of variables (cliques). Belief propagation can be performed on the new tree.



- Note: In the worst case, the junction tree nodes must take on exponentially many combinations of values, but can work well in practice.

Approximate Inference Methods

- For densely connected graphs, exact inference may be intractable.
- There are 3 widely used approximation schemes
 - Pretend the graph is a tree: “loopy belief propagation”
 - Markov chain Monte Carlo (MCMC): simulate lots of values of the variables and use the sample frequencies to approximate probabilities.
 - Variational inference: Approximate the graph with a simplified the graph (with no cycles) and fit a set of “variational” parameters such that the probabilities computed with the simple structure are “close” to those in the original graph.

Learning in DAGs

- Two things could be learned:
 - Graph structure
 - Parameters governing the conditional probability distribution
- Learning the structure often involves a search over candidate structures and a method to score each structure.
 - In practice, it is often difficult to extract the conditional independence relationships that make DAGs so appealing in the first place.
 - MCMC methods are also used to search over the space of structures.
- We will focus on the problem of learning the parameters.

Learning the Parameters

- Consider the parameter set $\theta = (\theta_1, \dots, \theta_n)$ which govern the conditional probability distributions $P(X_i | Parents_i, \theta)$.
- One way to learn the parameters is to maximize the likelihood (or probability) of the evidence (observed variables):

$$\begin{aligned}\theta_{ML} &= \arg \max_{\theta} [\ln P(V | \theta)] \\ &= \arg \max_{\theta} \left[\ln \sum_H P(V, H | \theta) \right]\end{aligned}$$

- When there are no latent variables, the situation is much easier:
 - If we can condition on all the variables, the graph factors by d-separation and we can estimate the parameters for all the $P(X_i | Parents_i, \theta)$ independently (Bayes Classifier).
- Otherwise the summation over H (inside the logarithm) may be intractable.

Expectation-Maximization Algorithm

- E-step (expectation): evaluate the posterior distribution $P(H | V, \theta^{old})$ using current estimate, θ_{old} , of the parameters.
- M-step (maximization): re-estimate θ by maximizing the expected *complete-data* log-likelihood:

$$\theta_{new} = \arg \max_{\theta} \left\{ \sum_H P(H | V, \theta_{old}) \ln P(V, H | \theta) \right\}$$

- Note that the log and the summation have been exchanged - this will often make the summation tractable.
- Iterate E and M steps until convergence.
- Guaranteed to converge to a local optimum with linear convergence rate

Understanding EM

1. Phase E (estimation):

Auxiliary Function:
$$Q(\theta, \theta_{t-1}) = E_Z \left[\sum_i \log(P_\theta(y_i, Z)) \mid \theta_{t-1}, D \right]$$

(où $D = \{y_1 \dots y_n\}$ et la distribution de Z est conditionnée sur la connaissance de D , en utilisant les paramètres θ_{t-1}).

2. Phase M (maximisation):

$$\theta_t \leftarrow \operatorname{argmax}_\theta Q(\theta, \theta_{t-1})$$

Understanding EM (cont.)

D'où vient la fonction auxiliaire Q ?

On va utiliser Q pour borner la vraisemblance et on va ensuite optimiser θ par rapport à cette borne. Soit $L(\theta)$ la log-vraisemblance obtenue avec les paramètres θ . Donc

$$\begin{aligned} L(\theta) - L(\theta_{t-1}) &= \sum_i \log \left\{ \frac{\sum_j P_\theta(Z = j, Y = y_i)}{P_{\theta_{t-1}}(Y = y_i)} \right\} \\ &= \sum_i \log \left\{ \frac{\sum_j P_\theta(Z = j, Y = y_i)}{P_{\theta_{t-1}}(Y = y_i)} \frac{P_{\theta_{t-1}}(Z = j|Y = y_i)}{P_{\theta_{t-1}}(Z = j|Y = y_i)} \right\} \\ &\geq \sum_i \sum_j P_{\theta_{t-1}}(Z = j|Y = y_i) \log \left\{ \frac{P_\theta(Z = j, Y = y_i)}{P_{\theta_{t-1}}(Y = y_i) P_{\theta_{t-1}}(Z = j|Y = y_i)} \right\} \\ &= \sum_i E_Z[\log(P_\theta(Z, y_i)) | \theta_{t-1}, D] + \text{constante en } \theta \end{aligned}$$

où l'on a utilisé l'inégalité de Jensen pour le log ($\log(E[X]) \geq E[\log X]$).

Mixture of Multivariate Gaussians

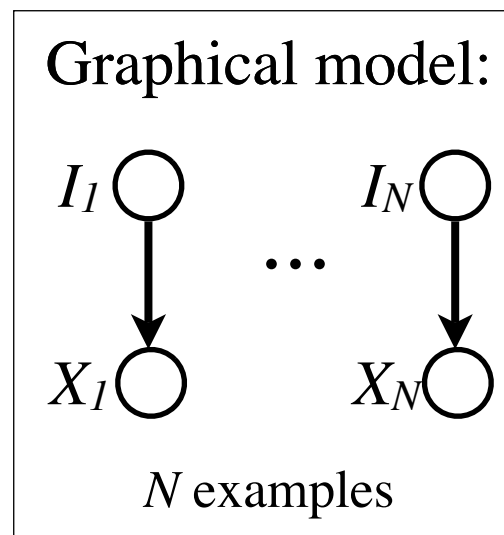
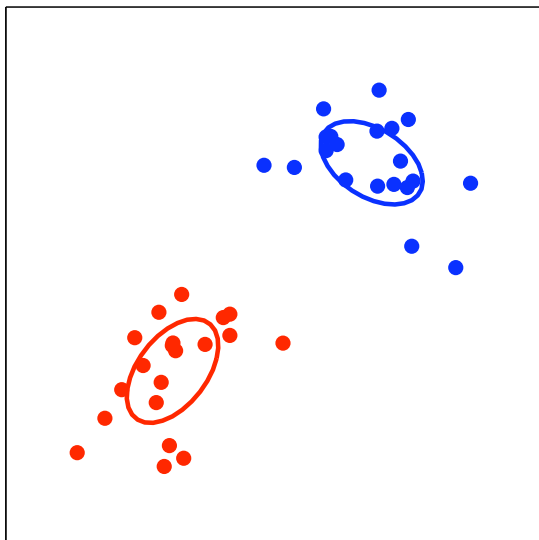
- Now let's consider a random variable distributed according to a mixture of Gaussians.
- **Conditional distributions for a D -dimensional X :**

$$P(I = i) = w_i$$

$$p(X = x | I = i) = \mathcal{N}(x | \mu_i, \Sigma_i) = \frac{1}{(2\pi)^{D/2} |\Sigma_i|^{1/2}} \exp\left(-\frac{1}{2}(x - \mu_i)^T \Sigma_i^{-1} (x - \mu_i)\right)$$

where I is an index over the *multivariate* Gaussian components in the mixture and the mixing proportion, w_i , is the marginal probability that X is generated by mixture component i .

- **Marginal distributions:** $p(X = x) = \sum_i p(X = x | I = i)P(I = i) = \sum_i w_i \mathcal{N}(x | \mu_i, \Sigma)$



E Step: Mixture of Gaussians

- Calculate $P(I_n | X_n, \theta)$ for each observed example X_n

$$X_n = [X_{1,n}, \dots, X_{d,n}, \dots, X_{D,n}]^T$$

$$\begin{aligned} P(I_n = i | X_n, \theta) &= \frac{P(I_n = i | w_i)P(X_n | I_n = i, \mu_i, \Sigma_i)}{P(X_n)} \\ &= \frac{P(I_n = i | w_i)P(X_n | I_n = i, \mu_i, \Sigma_i)}{\sum_i P(I_n = i | w_i)P(X_n | I_n = i, \mu_i, \Sigma_i)} \\ &= \frac{w_i \mathcal{N}(X_n | \mu_i, \Sigma_i)}{\sum_j w_j \mathcal{N}(X_n | \mu_j, \Sigma_j)} \end{aligned}$$

Where $\theta = \{\theta_1, \dots, \theta_i, \dots, \theta_K\}$, $\theta_i = \{w_i, \mu_i, \Sigma_i\}$ and $\mathcal{N}(\cdot)$ is the multivariate Gaussian probability density function.

M Step: Mixture of Gaussians

- For mixtures of Gaussians:

$$\theta \leftarrow \arg \max_{\theta'} \left\{ \sum_n \sum_i P(I_n = i | X_n = x_n, \theta) \ln P(X_n = x_n, I_n = i | \theta') \right\}$$

- We already computed $P(I_n = i | X_n = x_n, \theta)$ in the E step and we can decompose the joint $P(X_n = x_n, I_n = i | \theta')$:

$$\begin{aligned} \sum_n \sum_i P(i_n | x_n, \theta) \ln p(x_n, i_n | \theta') &= \sum_n \sum_i P(i_n | x_n, \theta) \ln p(x_n | i_n, \theta') P(i_n | \theta') \\ &= \sum_n \sum_i P(i_n | x_n, \theta) \ln w'_i + \sum_n \sum_i P(i_n | x_n, \theta) \ln \mathcal{N}(x_n | \mu'_i, \Sigma'_i) \end{aligned}$$

- Now we maximize this expression w.r.t θ' (*on to the M step*)

M Step: Mixture of Gaussians (cont.)

- Let's consider updating w_i : (subject to the constraint $\sum_i w_i = 1$)

$$\frac{\partial}{\partial w'_i} \left[\sum_n \sum_i P(i_n | x_n, \theta) \ln w'_i + \lambda \left(\sum_i w'_i - 1 \right) \right] = 0$$

$$\sum_{n=1}^N \frac{1}{w'_i} P(i_n | x_n, \theta) + \lambda = 0$$

$$w_i \leftarrow \frac{1}{N} \sum_{n=1}^N P(i_n | x_n, \theta)$$

M Step: Mixture of Gaussians (cont.)

- Now consider updating the mean vectors μ_i :

$$\frac{\partial}{\partial \mu'_i} \left[\sum_n \sum_i P(i_n | x_n, \theta) \ln \mathcal{N}(x_n | \mu'_i, \Sigma'_i) \right] = 0$$

$$\frac{\partial}{\partial \mu'_i} \left[\sum_n \sum_i P(i_n | x_n, \theta) \left(-\frac{1}{2} \ln(|\Sigma'_i|) - \frac{1}{2} (x_n - \mu_i)^T \Sigma'^{-1}_i (x_n - \mu_i) \right) \right] = 0$$

$$\mu_i \leftarrow \frac{\sum_{n=1}^N P(i_n | x_n, \theta) x_n}{\sum_{n=1}^N P(i_n | x_n, \theta)}$$

M Step: Mixture of Gaussians (cont.)

- Finally, let's consider updating the covariance matrices Σ_i :

$$\frac{\partial}{\partial \Sigma'_i} \left[\sum_n \sum_i P(i_n | x_n, \theta) \ln \mathcal{N}(x_n | \mu'_i, \Sigma'_i) \right] = 0$$

$$\frac{\partial}{\partial \Sigma'_i} \left[\sum_n \sum_i P(i_n | x_n, \theta) \left(-\frac{1}{2} \ln(|\Sigma'_i|) - \frac{1}{2} (x_n - \mu_i)^T \Sigma'^{-1}_i (x_n - \mu_i) \right) \right] = 0$$

This is the new μ_i

$$\Sigma_i \leftarrow \frac{\sum_{n=1}^N P(i_n | x_n, \theta) (x_n - \mu_i)(x_n - \mu_i)^T}{\sum_{n=1}^N P(i_n | x_n, \theta)}$$

EM Gaussian Mix: Summary

- Given observed X_1 to X_N and hidden variables I_1 to I_N (mixture component) iterate E and M steps until convergence.
- E step: for each data point n compute

$$P(i_n | x_n, \theta) = \frac{w_i \mathcal{N}(x_n | \mu_i, \Sigma_i)}{\sum_{j=1}^K w_j \mathcal{N}(x_n | \mu_j, \Sigma_j)}$$

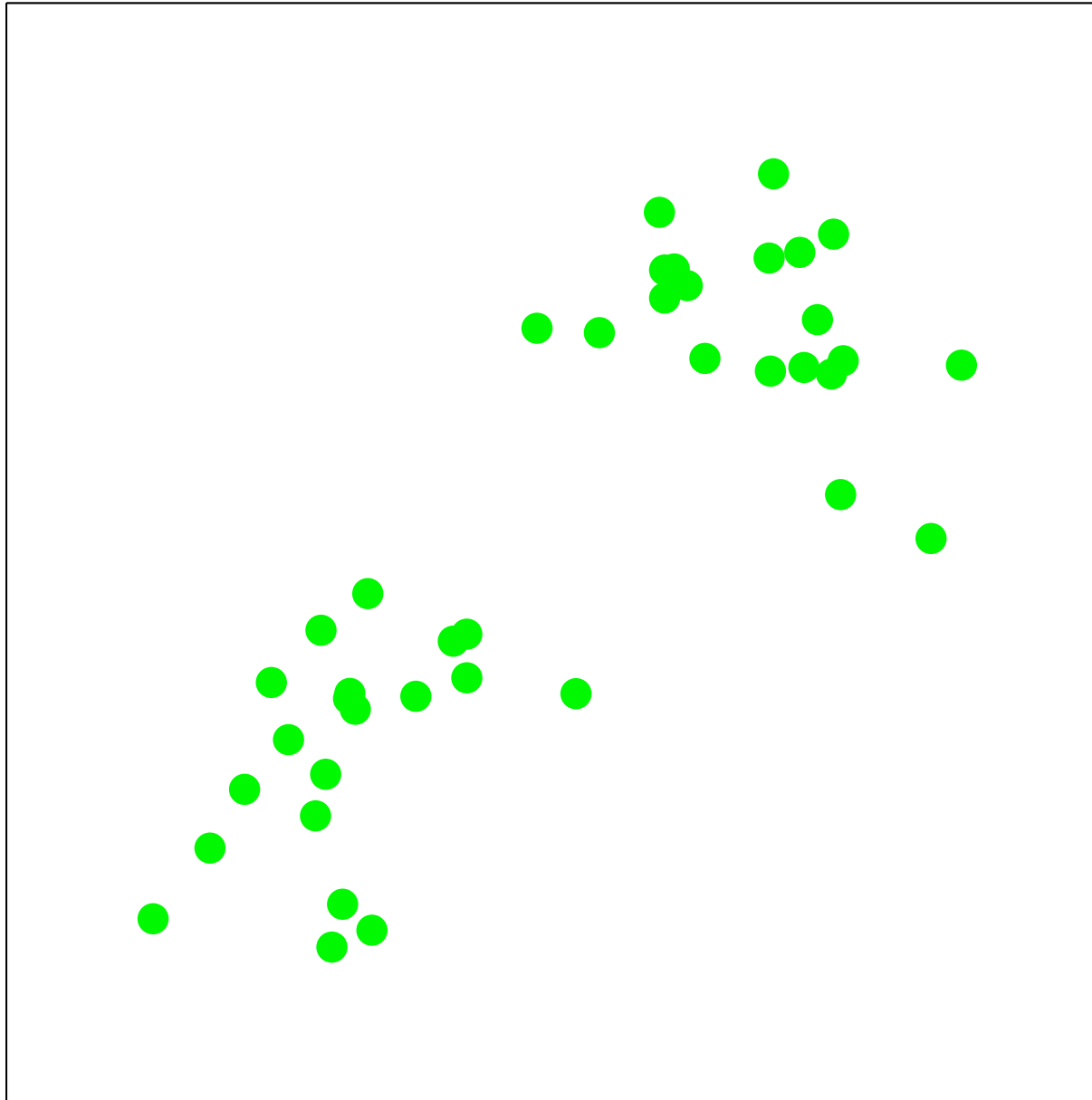
- M step: Update the parameters of component i (from 1 to K) with

$$w_i \leftarrow \frac{1}{N} \sum_{n=1}^N P(i_n | x_n, \theta)$$

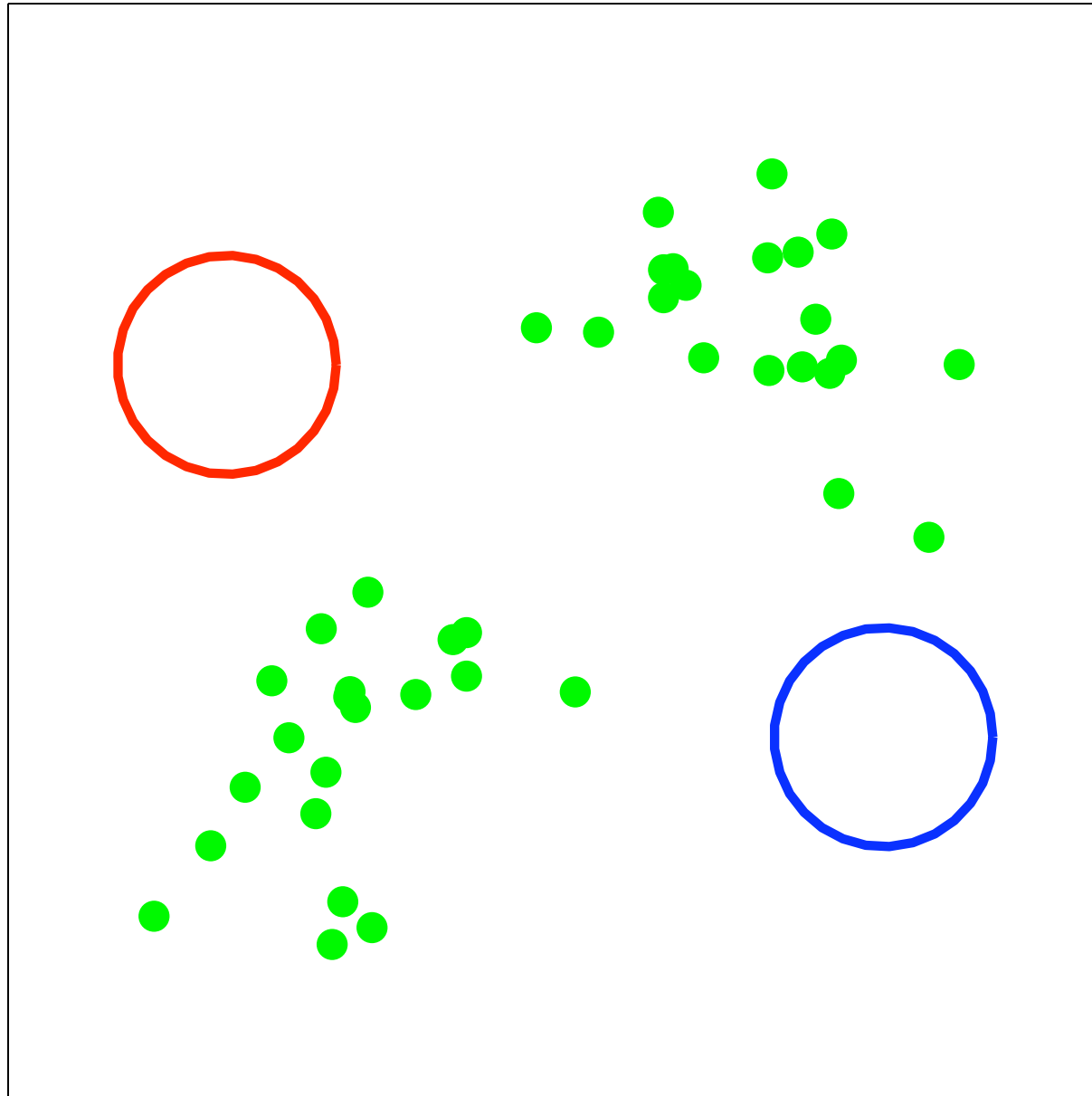
$$\mu_i \leftarrow \frac{\sum_{n=1}^N P(i_n | x_n, \theta) x_n}{\sum_{n=1}^N P(i_n | x_n, \theta)}$$

$$\Sigma_i \leftarrow \frac{\sum_{n=1}^N P(i_n | x_n, \theta) (x_n - \mu_i)(x_n - \mu_i)^T}{\sum_{n=1}^N P(i_n | x_n, \theta)}$$

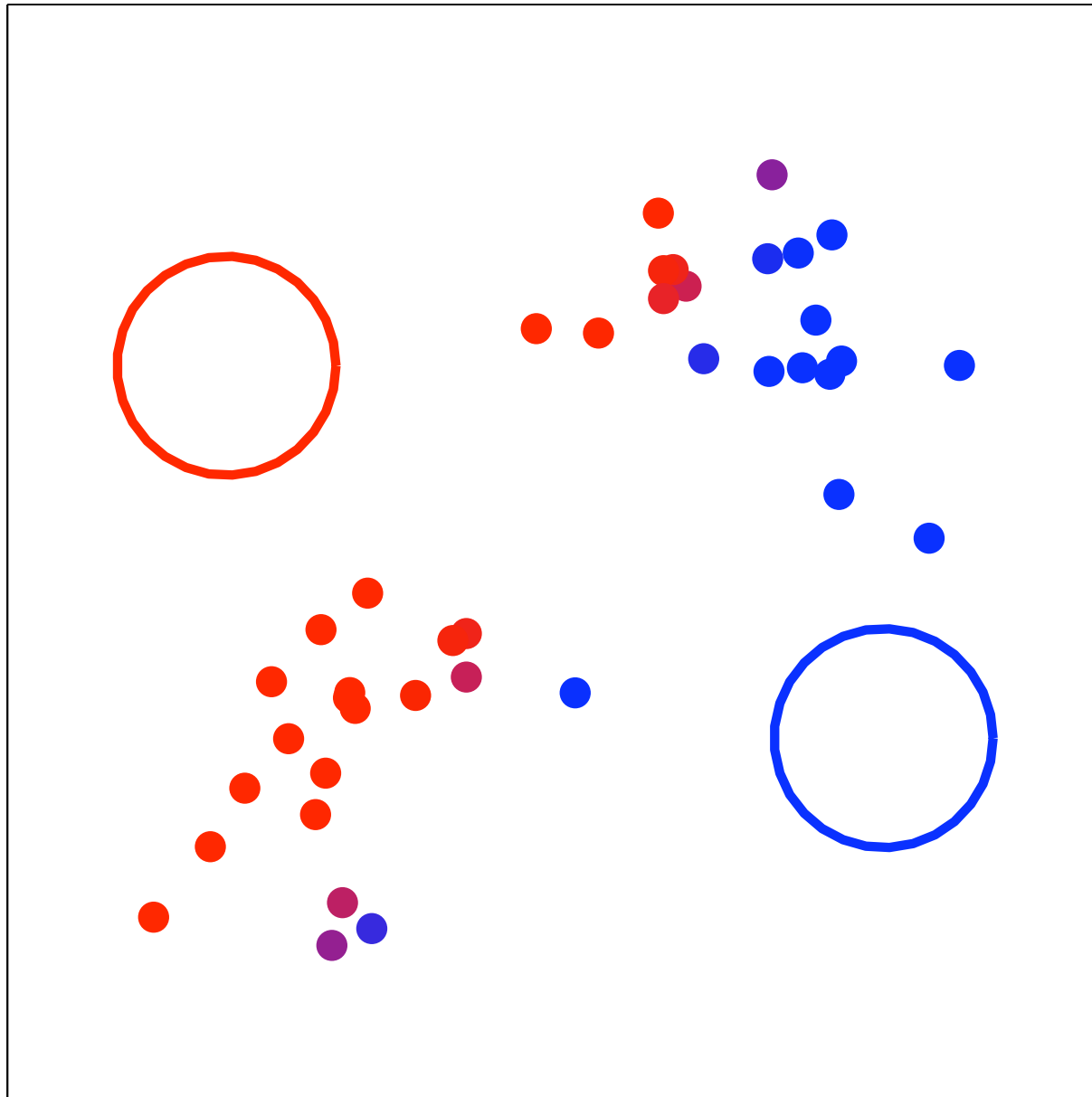
EM: Clustering with Mixtures of Gaussians



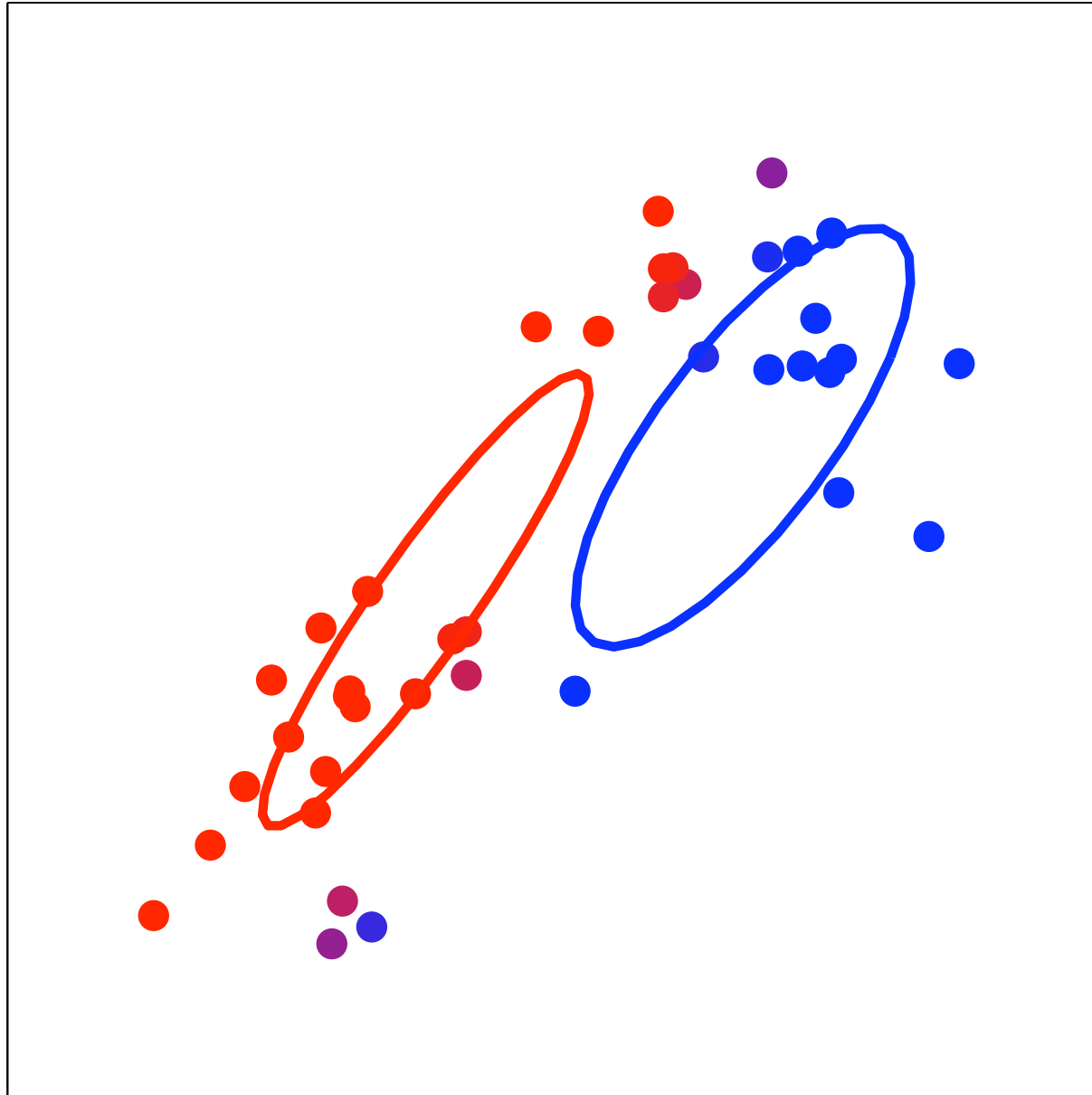
Clustering with Mixtures of Gaussians (cont.)



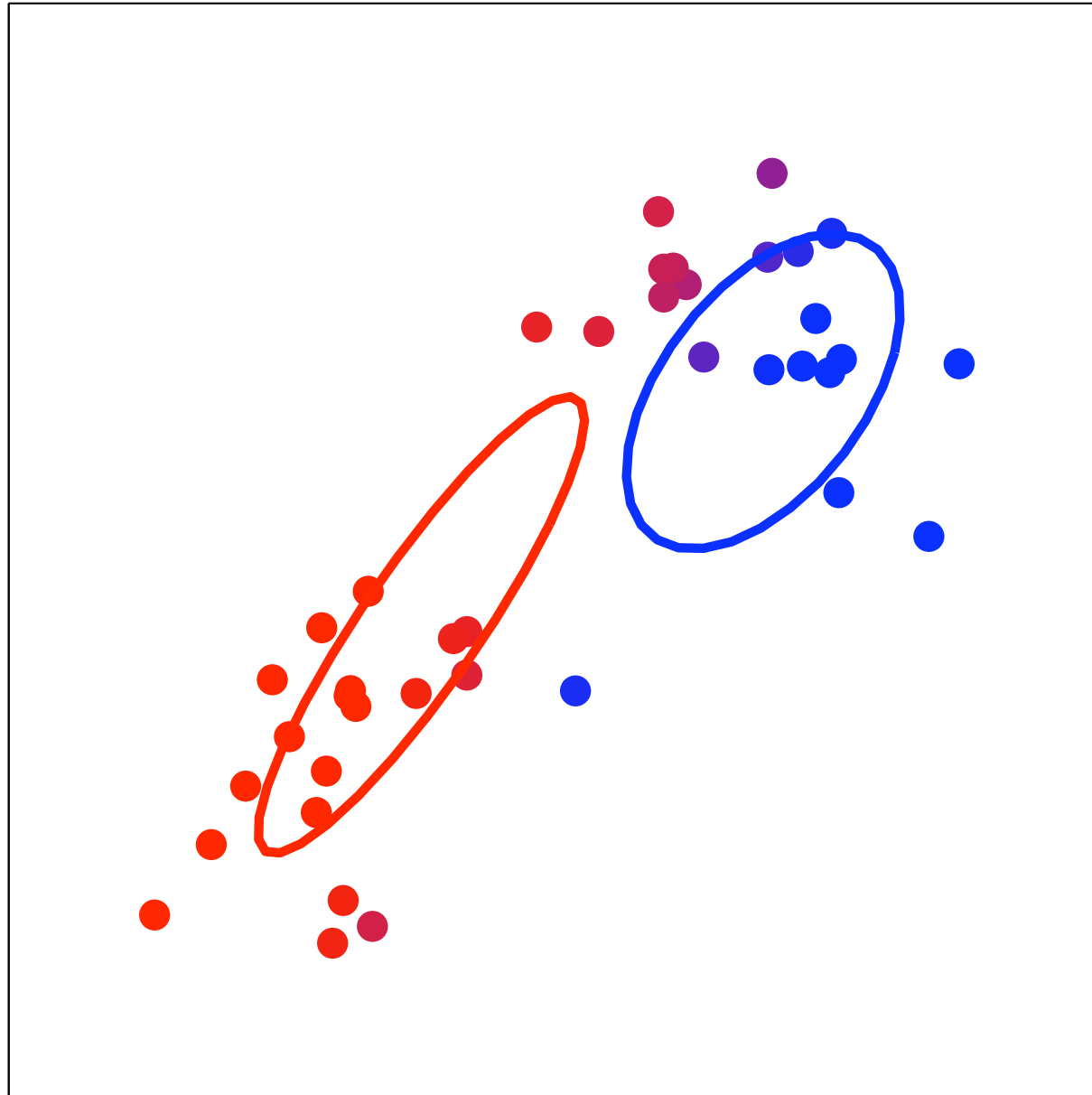
Clustering with Mixtures of Gaussians (cont.)



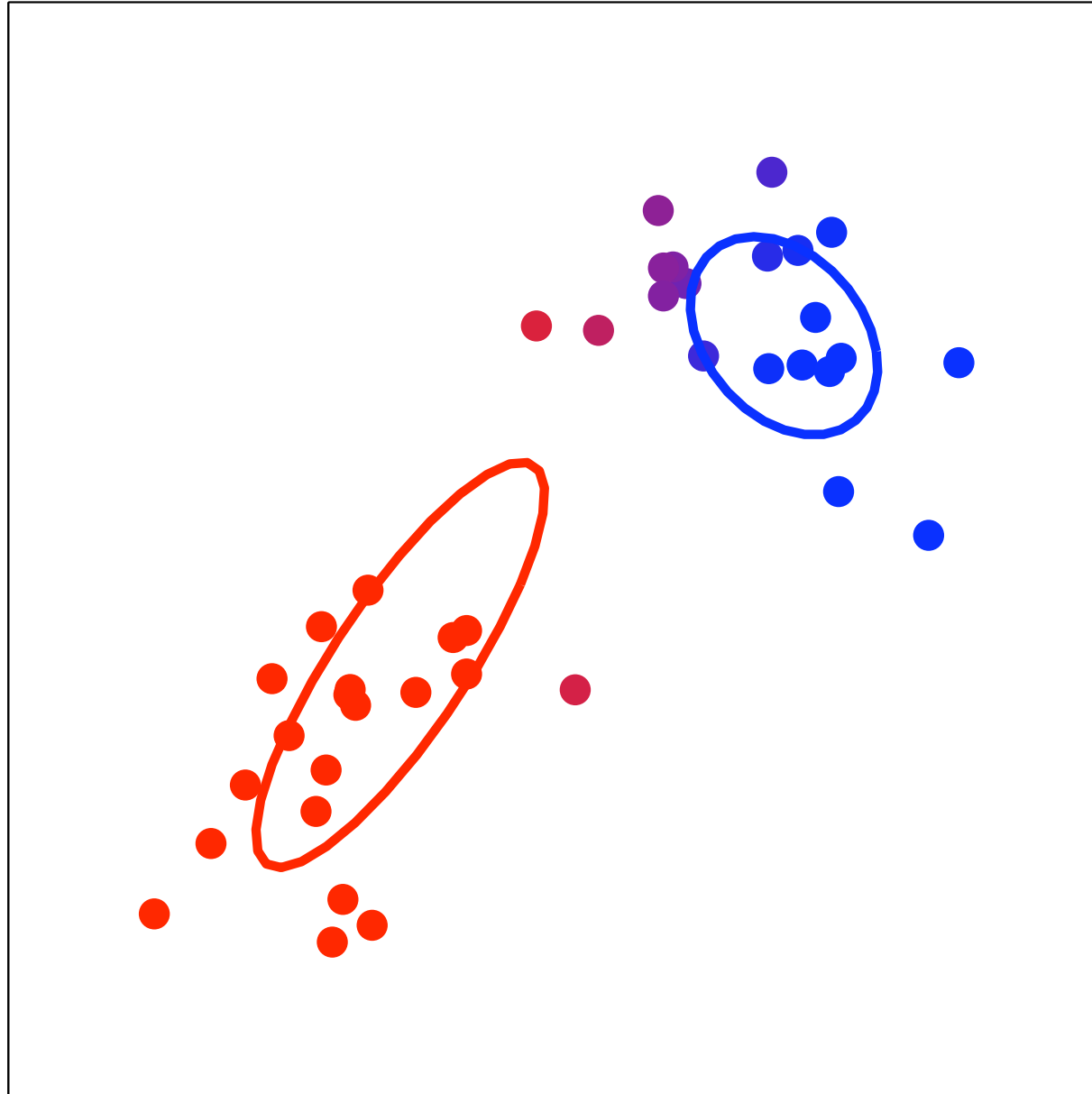
Clustering with Mixtures of Gaussians (cont.)



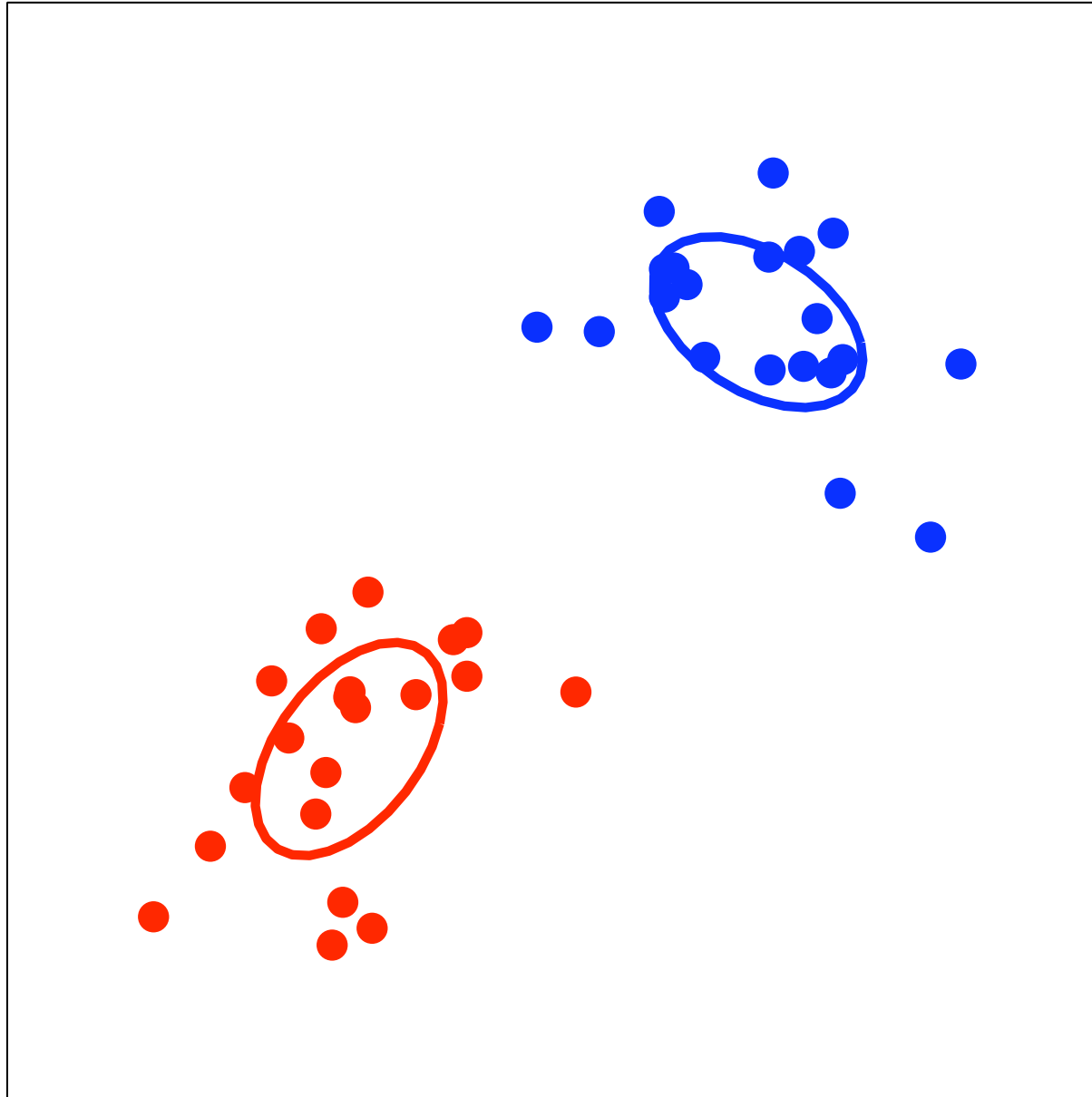
Clustering with Mixtures of Gaussians (cont.)



Clustering with Mixtures of Gaussians (cont.)



Clustering with Mixtures of Gaussians (cont.)



EM II: Clustering Documents with Naïve Bayes

- Consider you have a collection of D unlabeled documents.
- Build an initial naïve Bayes classifier with parameters θ . Use EM to find the maximum likelihood estimation of the parameters.
- Naïve Bayes assumption for document clustering:
 - The probability of a document d_i given class c_j is the product of the probabilities of the words $w_{d_i,k}$ in the document given that class:

$$P(d_i | c_j, \theta) = \prod_k P(w_{d_i,k} | c_j, \theta)$$

- The model parameters are the probabilities of the words w_t given the class c_j : $\theta_{w_t | c_j}$ and the marginal probabilities of the class c_j : θ_{c_j}
- Repeat until convergence:
 - E step: Use the current classifier (θ) to estimate component membership of each unlabeled document, i.e. the probability that each class generated each document $P(c_j | d_i, \theta)$.
 - M step: Re-estimate the classifier (θ) given the estimated component membership of each document.

EM II: Clustering Documents with Naïve Bayes

- E step:

$$\begin{aligned} P(y_i = c_j \mid d_i, \theta) &= \frac{P(c_j \mid \theta) P(d_i \mid c_j, \theta)}{P(d_i \mid \theta)} \\ &= \frac{P(c_j \mid \theta) \prod_{k=1}^{|d_i|} P(w_{d_i, k} \mid c_j, \theta)}{\sum_{r=1}^{\mathcal{C}} P(c_r \mid \theta) \prod_{k=1}^{|d_i|} P(w_{d_i, k} \mid c_r, \theta)} \end{aligned}$$

- M step:

$$\begin{aligned} \theta_{w_t | c_j} &\leftarrow P(w_t \mid c_j, \theta) = \frac{\sum_{i=1}^{|\mathcal{D}|} \text{Num}(w_t, d_i) P(y_i = c_j \mid d_i)}{\sum_{s=1}^{|\mathcal{V}|} \sum_{i=1}^{|\mathcal{D}|} \text{Num}(w_s, d_i) P(y_i = c_j \mid d_i)} \\ \theta_{c_j} &\leftarrow P(c_j \mid \theta) = \frac{\sum_{i=1}^{|\mathcal{D}|} P(y_i = c_j \mid d_i)}{|\mathcal{D}|} \end{aligned}$$

where $|\mathcal{D}|$ is the number of documents, \mathcal{C} is the number of classes, $|d_i|$ is the number of words in document d_i and w_t is the t -th word in the vocabulary of size $|\mathcal{V}|$.

Summary

- Directed acyclic graphical models are a great way to deal with probability distributions over multiple random variables
 - Especially when there are significant conditional independence relationships between the variables.
 - They help clarify the dependency structure
 - The graph structure can be exploited to determine efficient ways to inference.
- In some cases (polytrees), exact inference in a DAG is computationally tractable (linear in the number of nodes).
- Learning is harder with hidden variables.
 - But we have good algorithms such as EM to (more or less) handle these cases.

Acknowledgments

- The material presented here is taken from tutorials, notes and lecture slides from Yoshua Bengio, Christopher Bishop, Andrew Moore, Tom Mitchell and Scott Davies.

Note to other teachers and users of these slides. Andrew and Scott would be delighted if you found this source material useful in giving your own lectures. Feel free to use these slides verbatim, or to modify them to fit your own needs. PowerPoint originals are available. If you make use of a significant portion of these slides in your own lecture, please include this message, or the following link to the source repository of Andrew's tutorials: <http://www.cs.cmu.edu/~awm/tutorials>. Comments and corrections gratefully received.