STA414
Lecture Notes

Yuchen Wang

February 9, 2020

Contents

1 Introduction

2 Introduction to Probabilistic Models

2.1 Overview of probabilistic models

In general, we have random variables $X = (X_1, \ldots, X_N)$ that are either *observed* or *unobserved*. Need a model that captures the relationship between these variables. The approach of probabilistic generative models is to relate all variables by a learned joint probability distribution $p_\theta(X_1, \ldots, X_N)$. We assume there is a true joint p_*, which we are trying to learn with a model p_θ.
Assume we have the joint probability $p(X, C, Y)$

Regression

$$p(Y|X) = \frac{p(X,Y)}{p(X)} = \frac{p(X,Y)}{\int p(X,Y)\,dY}$$

Classification / Clustering

$$p(C|X) = \frac{p(X,C)}{\sum_C p(X,C)}$$

Now that we have a distribution over class labels, we have a choice for how to assign the class label:

1. $c^* = \arg\max_c p(C = c|X)$
2. Sample the class assignment from our distribution, $c^* \sim p(C|X)$
3. Output the class assignment (however we chose it) along with its density under our distribution $(c^*, p(C = c^*|X))$. Can inform us of the model's uncertainty or confidence of the prediction.

Latent/hidden Variables Variables which are never observed in the dataset.

Operations on Probabilistic Models

- **Generate Data** For this we will need to know how to sample from the model
- **Estimate Likelihood** When all variables are either observed or marginalized, the result is a single real number which is the probability of all variables taking on whose specific values.
- **Inference**: Compute expected value of some variables given others which are either observed or marginalized.
- **Learning**: Set the parameters of the joint distribution given some observed data to maximize the probability of the observed data.

Goals of joint distributions

1. Facilitate **efficient computation** of marginal and conditional distributions
2. Have **compact representation** so the size of the parameterization scales well for joint distributions over many variables.

Joint Dimensionality Suppose n is the number of variables and k is the number of states of each variable. Then dimensionality of our parameters is k^n.

2.2 Sufficient statistics

Theorem 2.1 (Fisher-Neyman factorization theorem)**.** If $f_\theta(x)$ is a pdf, then T is sufficient for θ if and only if nonnegative functions g and h can be found such that

$$f_\theta(x) = h(x)g_\theta(T(x))$$

i.e. the density f can be factored into a product such that one factor h does not depend on θ and the other factor, which does depend on θ, depends on x only through $T(x)$.

Definition 2.1 (Statistic and Sufficient statistic)**.** A statistic is a (possibly vector valued) deterministic function of a (set of) random variable(s). A sufficient statistic is a statistic that conveys exactly the same information about the data generating process that created the data as the entire data itself. Formally, we say that $T(X)$ is a sufficient statistic for X if

$$T(x^{(1)}) = T(x^{(2)}) \implies L(\theta; x^{(1)}) = L(\theta; x^{(2)}) \quad \forall\theta$$

where L is the likelihood function.
Alternatively,

$$P(\theta|T(X)) = P(\theta|X)$$

Equivalently (by the Neyman factorization theorem) we can write

$$P(\theta|T(X)) = h(x, T(x))g(T(x), \theta)$$

Example 2.1 (Bernoulli Trials)**.** We observe N iid coin flips.
Model: $p(H) = \theta, P(T) = 1 - \theta$
Likelihood: $l(\theta; D) = \log\theta \sum_n x^{(n)} + \log(1-\theta)\sum_n (1 - x^{(n)})$

$$\begin{aligned} l(\theta; D) &= \log\theta \sum_n x^{(n)} + \log(1-\theta)\sum_n (1 - x^{(n)}) \\ &= \log\theta N_H + \log(1-\theta)N_T \end{aligned}$$

Notice that our likelihood depends on $N_H = \sum_n x^{(n)}$ (and N_T).
$\implies$ If we know this summary statistic $T(x) = \sum_n x^{(n)}$, then we know everything that is useful from our sample to do inference.

$$l(\theta; D) = T(X)\log\theta + (N - T(X))\log(1-\theta)$$

Then we take the derivative and set it to 0 to find the maximum

$$\Rightarrow \frac{\partial \ell}{\partial \theta} = \frac{T(X)}{\theta} - \frac{N - T(X)}{1 - \theta}$$
$$\Rightarrow \hat{\theta} = \frac{T(X)}{N}$$

This is our maximum likelihood estimation of the parameters $\theta, \theta^{\star}_{MLE}$.
sufficient statistics: counts

Example 2.2 (Multinomial)**.** We observe M iid die rolls (K-sided).
Model: $p(k) = \theta_k, \sum_k \theta_k = 1$
Likelihood: $l(\theta; D) = \sum_k N_k \log \theta_k$
Take derivatives and set to zero (enforcing $\sum \theta_k = 1$):

$$\frac{\partial \ell}{\partial \theta_k} = \frac{N_k}{\theta_k} - M$$
$$\Rightarrow \theta^*_k = \frac{N_k}{M}$$

sufficient statistics: counts

Example 2.3 (exponential family of distributions)**.** The result of the previous example distributions show that the MLE are just normalized counts. However, the simplicity of the sufficient statistics and MLE are due to those being members of the exponential family. In general, exponential family members have simple sufficient statistics and MLE for the natural statistics η

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

where:

- η are the parameters
- $T(x)$ are the sufficient statistics
- $h(x)$ is the base measure
- $A(\eta)$ is the normalizing constant

with log-likelihood

$$\begin{aligned} l(\eta; D) &= \log p(D; \eta) \\ &= (\sum_n \log h(x_n)) - NA(\eta) + (\eta^T \sum_n T(x_n)) \end{aligned}$$

Finding the derivative and setting to zero, we get

$$\eta_{MLE} = \frac{1}{N} \sum_n T(x_n)$$

which is the normalized counts of the data.

Example 2.4. univariate normal We have N i.i.d. samples $\{x_i\}_1^N$
Model: $p(x|\theta) = \frac{1}{\sqrt{2\pi}\sigma} \exp -\frac{1}{2\sigma^2}(x-\mu)^2$
Gaussian distribution is a member of the exponential family, so we can put it into a natural form

$$p(x|\theta) = \frac{1}{\sqrt{2\pi}\sigma} \exp\{-\frac{1}{2\sigma^2}\mu^2\} \exp\{\begin{bmatrix}\frac{\mu}{\sigma^2} & \frac{-1}{2\sigma^2}\end{bmatrix} \begin{bmatrix}x \\ x^2\end{bmatrix}\}$$

From here, it is clear that the natural parameters and the sufficient statistics are

- $\eta = \begin{bmatrix}\frac{\mu}{\sigma^2} \\ \frac{-1}{2\sigma^2}\end{bmatrix}$
- $T(x) = \begin{bmatrix}x \\ x^2\end{bmatrix}$

re-writing in terms of η

$$p(x|\eta) = (\sqrt{2\pi})^{-\frac{1}{2}} \cdot (-2\eta_2)^{\frac{1}{2}} \cdot \exp\left\{\frac{\eta_1^2}{4\eta_2}\right\} \cdot \exp\left\{\eta^T T(x)\right\}$$

noting that

- $h(x) = (\sqrt{2\pi})^{-\frac{1}{2}}$
- $A(\eta) = (-2\eta_2)^{\frac{1}{2}} \cdot \exp\left\{\frac{\eta_1^2}{\eta_2}\right\}$

3 Directed Graphical Models

3.1 Decision Theory

We care about probabilities because they help us make decisions.
Denote action by a, state by s, value function by $V(s)$, utility function by $u(a)$, then

$$a^* = \underset{a}{argmax} \underbrace{E_{p(s|a,knowledge)}[V(s)]}_{u(a)}$$

3.2 Joint Distributions

Theorem 3.1 (chain rule of probability)**.** The joint distribution of N random variables can be computed by the chain rule

$$p\left(x_{1,\ldots,N}\right) = p\left(x_1\right) p\left(x_2|x_1\right) p\left(x_3|x_2,x_1\right) \ldots p\left(x_n|x_{n-1:1}\right)$$

This is true for any joint distribution over any random variables (assuming full dependence between variables). More formally, in probability the chain rule for two random variables is

$$p(x,y) = p(x|y)p(y)$$

and for N random variables

$$p(x_1, x_2, \ldots, x_N) = \prod_{j=1}^{N} p(x_j|x_1, x_2, \ldots, x_{j-1})$$

for any ordering of the variables.

3.2.1 Number of parameters in a joint distribution

$$p(x_1, \ldots, x_A | y_1, \ldots, y_B)$$

parameters = {(# possible states of $x_{1:A}$) - 1} $\times$ (# possible states of $y_{1:B}$)
binary: $(2^A - 1) \times 2^B$.

3.2.2 Conditional Independence

Definition 3.1 (conditionally independence)**.** Two random variables A, B are conditionally independent given a third variable C, denoted

$$X_A \perp X_B | X_C$$

if

$$\begin{aligned} &\iff p(X_A, X_B | X_C) = p(X_A | X_C) p(X_B | X_C) \\ &\iff p(X_A | X_B, X_C) = p(X_A | X_C) \\ &\iff p(X_B | X_A, X_C) = p(X_B | X_C) \end{aligned}$$

for all X_C.

3.3 Directed acyclic graphical models (DAGM)

Graphical models Probabilistic graphical models are a concise way to specify and reason about conditional independencies, without worrying about the detailed form of the distribution. There are three flavours:

- Undirected
- Factor graphs
- Directed

Definition 3.2 (directed acyclic graphical model)**.** A directed acyclic graphical model implies a restricted factorization of the joint distribution. In a DAG, variables are represented by nodes, and dependence are represented by edges.
Recall that for N random variables,

$$p(x_1, x_2, \ldots, x_N) = \prod_{j=1}^{N} p(x_j | x_1, x_2, \ldots, x_{j-1})$$

for any ordering of the variables.
In the context of DAG, we can write

$$p(x_1, x_2, \ldots, x_N) = \prod_{i=1}^{N} p(x_i | parents(x_i))$$

where $parents(x_i)$ is the set of nodes with edges pointing to x_i.

In other words, the joint distribution of a DAGM factors into a product of local conditional distributions, where each node (a random variable) is conditionally dependent on its parent nodes(s), which could be empty.

Remark 3.1. We are conditioning on parent nodes as opposed to every node. Therefore, the model that represents this distribution is exponential in the fan-in of each node (the number of nodes in the parent set), instead of in N.

Grouping variables We can always group variables together into one bigger variable:

$$p(x_i, x_{\pi_i}) = p(x_{\pi_i})p(x_i|x_{\pi_i})$$

3.3.1 Conditional Independence in DAGM

The simplest conditional independence relationship encoded in a Bayesian network can be stated as follows: a node is independent of its ancestors given its parents:

$$x_i \perp x_{\tilde{\pi}_i}|x_{\pi_i}$$

In general, missing edges imply conditional independence.

Definition 3.3 (D-Separation)**.** D-separation, or directed-separation is a notion of connectedness in DAGMs in which two (sets of) variables may or may not be connected conditioned on a third (set of) variable(s).
D-connection implies conditional dependence and d-separation implies conditional independence.
In particular, we say that

$$x_A \perp x_B|x_C$$

if every variable in A is d-separated from every variable in B conditioned on all the variables in C.
To check if an independence is true, we can cycle through each node in A, do a depth-first search to read every node in B, and examine the path between them. If all of the paths are d-separated(i.e. conditionally independent), then

$$x_A \perp x_B|x_C$$

Example 3.1. Chain

Question: When we condition on y, are x and z independent?
Answer:
From the graph, we get

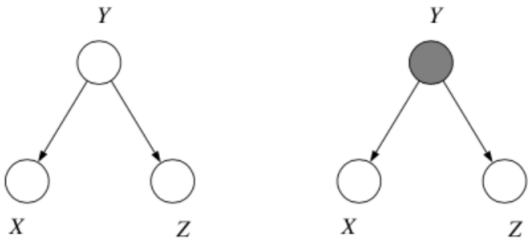
$$P(x, y, z) = P(x)P(y|x)P(z|y)$$

Then

$$\begin{aligned} P(z|x, y) &= \frac{P(x, y, z)}{P(x, y)} \\ &= \frac{P(x)P(y|x)P(z|y)}{P(x)P(y|x)} \\ &= P(z|y) \end{aligned}$$

which implies $x \perp z|y$.

Example 3.2. Common Cause



Where we think of y as the "common cause" of the two independent effects x and z.
Question: When we condition on y, are x and z independent?
Answer: From the graph, we get

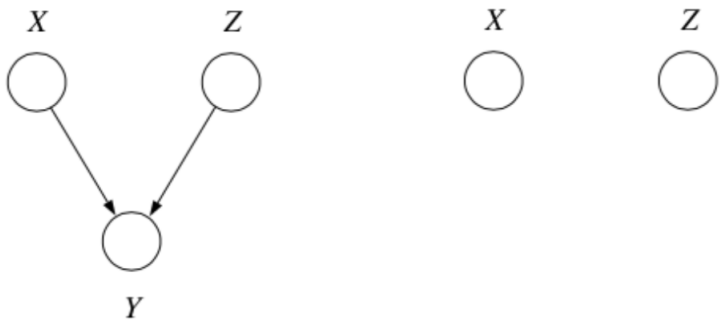
$$P(x, y, z) = P(y)P(x|y)P(z|y)$$

which implies

$$\begin{aligned} P(x, z|y) &= \frac{P(x, y, z)}{P(y)} \\ &= \frac{P(y)P(x|y)P(z|y)}{P(y)} \\ &= P(x|y)P(z|y) \end{aligned}$$

which implies $x \perp z|y$.

Example 3.3. Explaining Away



Question: $x \perp z$?
Answer:

$$\begin{aligned} P(x,z) &= \sum_y P(x,y,z) \\ &= \sum_y P(x)P(z)P(y|x,z) \\ &= P(x)P(z)\sum_y P(y|x,z) \\ &= P(x)P(z) \end{aligned}$$

which implies that $x \perp z$.
Question: $x \perp z|y$?
Answer:

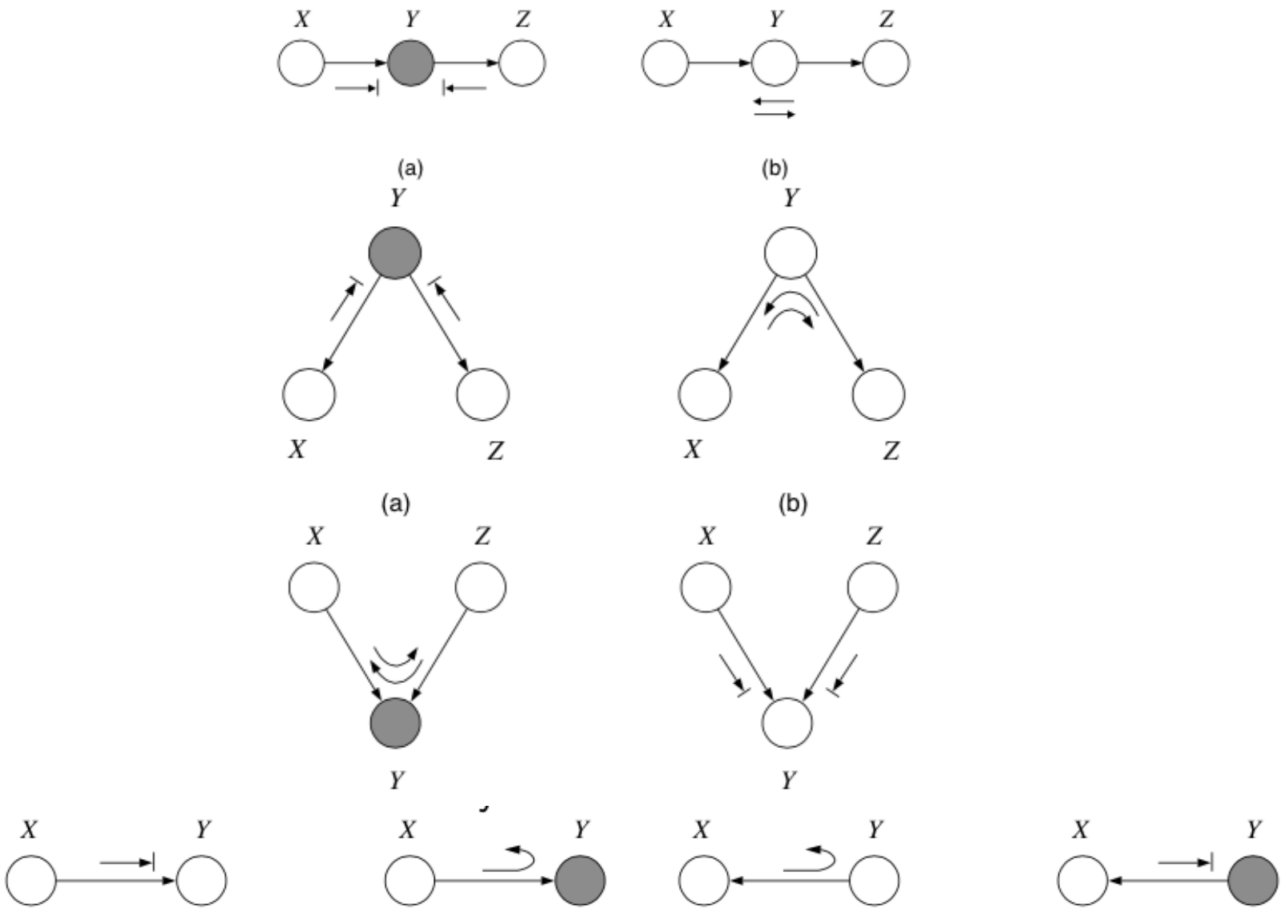
$$\begin{aligned} p(z|x,y) &= \frac{p(x)p(z)p(y|x,z)}{p(x)p(y|x)} \\ &= \frac{p(z)p(y|x,z)}{p(y|x)} \\ &= p(z|y) \end{aligned}$$

which implies $x \not\perp |y$.
In fact, x and z are marginally independent, but given y they are conditionally dependent. This important effect is called explaining away.

Theorem 3.2 (Bayes-Balls Algorithm)**.** In general, the algorithm works as follows:

1. Shades all nodes x_C
2. Place "balls" at each node in x_A (or x_B)
3. Let the "balls" "bounce" around according to some rules
4. If any of the balls reach any of the nodes in x_B from x_A (or x_A from x_B), then $x_A \not\perp x_B|x_C$; otherwise $x_A \perp x_B|x_C$.

The rules are as follows:



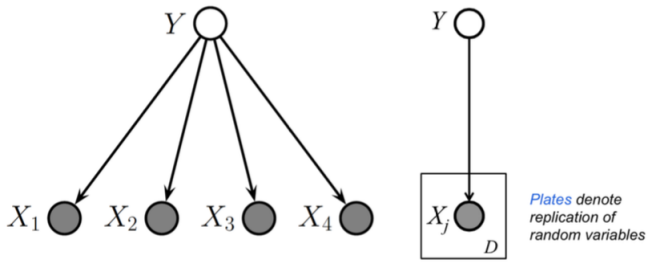
where arrows indicate paths the balls can travel, and arrows with bars indicate paths the balls cannot travel.

3.3.2 Example of a DAGM: Markov Chain

Markov chains are a stochastic model describing a sequence of possible events in which the probability of each event depends only on the state attained in the previous event.
i.e. Conditional on the present state of the system, its future and past states are independent.

3.3.3 Plates

Because Bayesian methods treat parameters as random variables, we would like to include them in the graphical model. One way to do this is to repeat all the iid observations explicitly and show the parameter only once. A better way is to use plates, in which repeated quantities that are iid are put in a box.



The rules of plates:

1. Repeat every structure in a box a number of times given by the integer in the corner of the box, updating the plate index variable as you go. Duplicate every arrow going into the plate and every arrow leaving the plate by connecting the arrows to each copy of the structure.
2. Plates can be nested, in which case their arrows get duplicated also.
3. Plates can also intersect, in which case the nodes at the intersection have multiple indices and get duplicated a number of times equal to the product of the duplication numbers on all the plates containing them.

3.3.4 Unobserved Variables

Certain variables in our models may be unobserved, either some of the time or always, at training time or at test time.

Partially unobserved variables If variables are **occasionally unobserved** then they are missing data, e.g. undefined inputs, missing class labels, erroneous target values. In this case, we can still model the joint distribution, but we marginalize the missing values: $l(\theta; D) = \sum_{complete} \log p(x^c, y^c|\theta) + \sum_{missing} \log p(x^m|\theta)$

$$\begin{aligned} l(\theta; D) &= \sum_{complete} \log p(x^c, y^c|\theta) + \sum_{missing} \log p(x^m|\theta) \\ &= \sum_{complete} \log p(x^c, y^c|\theta) + \sum_{missing} \log \sum_y p(x^m, y|\theta) \end{aligned}$$

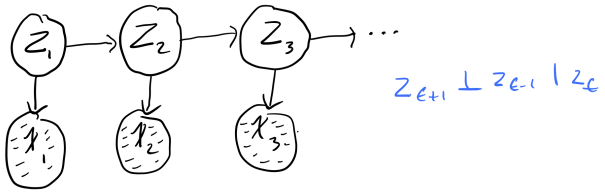
Latent variables What to do when a variable z is always unobserved?
Depends on where it appears in our model. If we never condition on it when computing the probability of the variables we do observe, then we can just forget about it and integrate it out.

Mixture models What if the class is unobserved? Then we sum it out

$$p(x|\theta) = \sum_{k=1}^{K} p(z = k|\theta_z) p(x|z = k, \theta_k)$$

We can use the Bayes' rule to compute the posterior probability of the mixture component given some data:

Hidden Markov Models (HMMs) Hidden Markov Model (HMM) is a statistical Markov model in which the system being modelled is assumed to be a Markov process with unobserved (i.e. hidden) states. It is a very popular type of latent variable model.



where

- Z_t are hidden states taking on one of K discrete values
- X_t are observed variables taking on values in any space

The joint probability represented by the graph factorizes according to

$$p(X_{1:T}, Z_{1:T}) = p(Z_{1:T})p(X_{1:T}|Z_{1:T}) = p(Z_1)\prod_{t=2}^{T} p(Z_t|Z_{t-1})\prod_{t=1}^{T} p(X_T|Z_t)$$

4 Exact Inference

4.1 Inference as Conditional Distribution

Notation 4.1.

$$\begin{aligned} X_E &= \text{ the observed evidence} \\ X_F &= \text{ the unobserved variable we want to infer} \\ X_R = X - \{X_F, X_E\} &= \text{ Remaining variables, extraneous to query} \end{aligned}$$

where X_R is the set of random variables in our model that are neither part of the query nor the evidence.

Definition 4.1 (exact inference)**.** The exact inference task is defined as: Given a fully parameterized DAG model over variables χ, $X_F \subseteq \chi, X_E \subseteq \chi$ s.t. $X_E \cap X_F = \emptyset$ and $\mathbf{e} \in val(X_E)$:

$$\text{compute } P(X_F|X_E = \mathbf{e})$$

X_E can be empty in which case we're after $P(X_F)$.

4.2 Variable elimination

Variable elimination is a simple and general exact inference algorithm in any probabilistic graphical model.

Simple Example: Chain

$$A \to B \to C \to D$$

where we want to compute $P(D)$, with no observations for other variables.
We have

$$X_F = \{D\}, X_E = \{\}, X_R = \{A, B, C\}$$

This graphical model describes the factorization of the joint distribution as:

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

If the goal is to compute the marginal distribution $p(D)$ with no observed variables, then we marginalize over all variables but D:

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

Sum naively: $\mathcal{O}(k^n)$

$$\begin{aligned} p(D) &= \sum_{A,B,C} p(A, B, C, D) \\ &= \sum_C \sum_B \sum_A p(A)p(B|A)p(C|B)p(D|C) \end{aligned}$$

Elimination ordering: $\mathcal{O}(nk^2)$

$$\begin{aligned} p(D) &= \sum_{A,B,C} p(A, B, C, D) \\ &= \sum_C p(D|C) \sum_B p(C|B) \sum_A p(A)p(B|A) \\ &= \sum_C p(D|C) \sum_B p(C|B)P(B) \\ &= \sum_C p(D|C)p(C) \end{aligned}$$

Remark 4.1. Computing the joint is NP-hard. We can catch up computations that are otherwise computed exponentially many times, but this depends on having a good variable elimination ordering.

4.2.1 Sum-Product Inference

Sum-product inference algorithm can be used to compute $P(Y)$ for directed and undirected models: $\forall Y$,

$$\tau(Y) = \sum_z \prod_{\phi \in \Phi} \phi(scope[\phi] \cap Z, scope[\phi] \cap Y)$$

where Φ is a set of potentials or factors.

We want to marginalize out variables in Z since they are extraneous

e.g. for $\phi(A, B, C)$, $\text{scope}[\phi] = \{A, B, C\}$

Directed models Φ is given by the conditional probability distributions for all variables

$$\Phi = \{\phi_{x_i}\}_{i=1}^{N} = \{p(x_i|parents(x_i))\}_{i=1}^{N}$$

where the sum is over the set $Z = X - X_F$. The resulting term $\tau(Y)$ will automatically be normalized.

For DAG, we have $\phi = p$

Undirected models Φ is given by the set of unormalized potentials. Therefore, we must normalize the resulting $\tau(Y)$ by $\sum_Y \tau(y)$.

Example 4.1 (Directed Graph)**.**

$$p(C, D, I, G, S, L, H, J) = p(C)p(D|C)p(I)p(G|D, I)p(L|G)p(S|I)p(J|S, L)p(H|J, G)$$

We can write the conditional distributions as factors

$$\Phi = \{\phi(C), \phi(C, D), \phi(I), \phi(G, D, I), \phi(L, G), \phi(S, I), \phi(J, S, L), \phi(H, J, G)$$

If we are interested in inferring the probability of getting a job, $p(J)$, we can perform exact inference on the joint distribution by marginalizing according to a specific variable elimination

ordering: $\prec \{C, D, I, H, G, S, L\}$

$$
\begin{aligned}
p(J) &= \sum_L \sum_S \phi(J,L,S) \sum_G \phi(L,G) \sum_H \phi(H,G,J) \sum_I \phi(S,I)\phi(I) \sum_D \phi(G,D,I) \underbrace{\sum_C \phi(C)\phi(C,D)}_{\tau(D)} \\
&= \sum_L \sum_S \phi(J,L,S) \sum_G \phi(L,G) \sum_H \phi(H,G,J) \sum_I \phi(S,I)\phi(I) \underbrace{\sum_D \phi(G,D,I)\tau(D)}_{\tau(G,I)} \\
&= \sum_L \sum_S \phi(J,L,S) \sum_G \phi(L,G) \sum_H \phi(H,G,J) \sum_I \underbrace{\phi(S,I)\phi(I)\tau(G,I)}_{\tau(S,G)} \\
&= \sum_L \sum_S \phi(J,L,S) \sum_G \phi(L,G)\tau(S,G) \underbrace{\sum_H \phi(H,G,J)}_{\tau(G,J)} \\
&= \sum_L \sum_S \phi(J,L,S) \underbrace{\sum_G \phi(L,G)\tau(S,G)\tau(G,J)}_{\tau(J,L,S)} \\
&= \sum_L \underbrace{\sum_S \phi(J,L,S)\tau(J,L,S)}_{\tau(J,L)} \\
&= \underbrace{\sum_L \tau(J,L)}_{\tau(J)} \\
&= \tau(J)
\end{aligned}
$$

Fact 4.1 (Complexity of variable elimination ordering)**.** The complexity of the VE algorithm is

$$\mathcal{O}(mk^{N_{max}})$$

- m is the number of initial factors $= |\Phi|$
- k is the number of states each random variable takes (assumed to be equal here)
- N_i is the number of random variables inside each sum $\sum_i$
- $N_{max} = argmax_i N_i$ is the number of random variables inside the largest sum.

5 Message passing, Hidden Markov Models, and Sampling

Inference in Trees Tree is a general family of graphs for which the optimal elimination ordering is trivial to find, and which has linear cost in the number of nodes.

5.1 Message Passing & Belief Propagation

What if we want to compute the marginal of every variable in a graph: $p(x_i)\,\forall x_i \in X$? Run variable elimination separately for each variable x_i is computationally expensive.

Joint distribution for undirected graph models For an undirected graph $G = (V, E)$,

$$P(X_{1:n}) = \frac{1}{Z}\prod_i \phi(x_i) \prod_{(j,k)\in E} \phi_{j,k}(x_j, x_k)$$

Message-passing Belief propagation is based on message-passing of "message" between neighboring vertices of the graph. The message sent from variable j to $i \in N(j)$ is

$$m_{j\to i}(x_i) = \sum_{x_j} \phi_j(x_j)\phi_{ij}(x_i, x_j) \prod_{k\in N(j)\neq i} m_{k\to j}(x_j)$$

Theorem 5.1 (Belief propagation algorithm)**.** As follows:

1. Choose root r arbitrarily
2. Pass messages from leaves to r
3. Pass messages from r to leaves
4. Compute $p(x_i) \propto \phi_i(x_i) \prod_{j\in N(i)} m_{j\to i}(x_i),\ \forall i$

5.2 Hidden Markov models

5.2.1 Sequential data

$$x_{1:T} = \{x_1, \ldots, x_T\}$$

Recall the general joint factorization via the chain rule

$$p(x_{1:T}) = \prod_{t=1}^{T} p(x_t|x_{t-1}, \ldots, x_1)$$

First order Markov chain

$$p(x_t|x_{1:t-1}) = p(x_t|x_{t-1})$$

This assumption greatly simplifies the factors in the joint distribution

$$p(x_{1:T}) = \prod_{t=1}^{T} p(x_t|x_{t-1})$$

Definition 5.1 (stationary/time-homogenous Markov chain)**.** As follows

- **Stationary Markov chain:** the distribution generating the data does not change through time:
$$p(x_t|x_{t-1}) = p(x_{t+k}|x_{t-1+k}) \quad \forall t, k$$
- **Non-stationary Markov chain:** the distribution generating the data is a function of time.

Higher-order Markov chains second order:

$$p(x_t|x_{1:t-1}) = p(x_t|x_{t-1}, x_{t-2})$$

m-order:

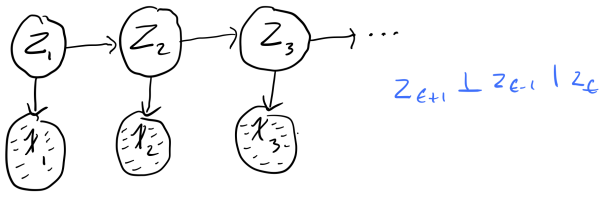
$$p(x_t|x_{1:t-1}) = p(x_t|x_{t-m:t-1})$$

Parameterization How does the order of temporal dependence affect the number of parameters in our model?
Assume x is a discrete random variable with k states.

1. $x_t : k - 1$, as the last state is implicit.
2. first-order chain: $k(k-1)$, as we need k numbers of parameters for each parameter of x_t
3. m-order chain: $k^m(k-1)$, as we need k^m number of parameters for each parameter of x_t

5.2.2 Hidden Markov Models

Hidden Markov Model (HMM) hide the temporal dependence by keeping it in the unobserved state. For each observation x_t, we associate a corresponding unobserved hidden/latent variable z_t



The joint probability represented by the graph factorizes according to

$$p(X_{1:T}, Z_{1:T}) = p(Z_{1:T})p(X_{1:T}|Z_{1:T}) = p(Z_1)\prod_{t=2}^{T} p(Z_t|Z_{t-1})\prod_{t=1}^{T} p(X_T|Z_t)$$

Unlike simple Markov chains, the observations are not limited by a Markov assumption of any order, i.e. x_t isn't necessarily independent of any other observation, no matter how many other observations we make.

Definition 5.2 (Parameterization of a hidden Markov model)**.** Assuming we have a homogeneous model, we only have to learn three distributions

1. **Initial distribution:** $\pi(i) = p(z_1 = i)$. The probability of the first hidden variable being in state i (often denoted by π.)
2. **Transition distribution:** $T(i, j) = p(z_{t+1} = j|z_t = i)$, $i \in \{1, \ldots, k\}$. The probability of moving from hidden state i to hidden state j.
3. **Emission probability:** $\epsilon_i(x_t) = p(x_t|z_t = i)$. The probability of an observed random variable x_t given the state of the hidden variable that "emitted" it.

5.2.3 Inference in HMMs

HMMs are just tree-structured DAGs, meaning that inference in them is linear in the number of time steps. We can do exact inference in them.

Main tasks we perform with HMMs

1. Compute the probability of a latent sequence given an observation sequence. (e.g. compute $p(z_i|x_{1:t})\forall i$ with the Forward-Backward algorithm)
2. Compute the marginal likelihood $p(x_1, x_2, \ldots, x_T)$ in order to fit parameters
3. Infer the most likely sequence of hidden states (i.e. compute $Z^* = \underset{z_{1:T}}{argmax}\, p(z_{1:T}|x_{1:T})$ using the Viterbi algorithm)

Assuming that we know the initial $p(z_1)$, transition $p(z_t|z_{t-1})$, and emission $p(x_t|z_t)\forall t \in [1, T]$ This task of hidden state inference breaks down into the following:

- **Filtering:** compute posterior over current hidden state, $p(z_t|x_{1:t})$
- **Prediction:** compute posterior over future hidden state, $p(z_{t+k}|x_{1:t})$
- **Smoothing:** compute posterior over past hidden state, $p(z_n|x_{1:t}) \quad 1 < n < t$

Prediction Let's first take a look at the example where $k = 2$, then

$$\begin{aligned} p(z_{t+2}|x_{1:t}) &= \sum_{z_{t+1}} \sum_{z_t} p(z_t, z_{t+1}, z_{t+2}|x_{1:t}) \\ &= \sum_{z_{t+1}} \sum_{z_t} p(z_t|x_{1:t}) p(z_{t+1}|z_t, x_{1:t}) p(z_{t+2}|z_{t+1}, z_t, x_{1:t}) \\ &= \sum_{z_{t+1}} \sum_{z_t} p(z_t|x_{1:t}) p(z_{t+1}|z_t) p(z_{t+2}|z_{t+1}) \end{aligned}$$

Theorem 5.2 (Forward-backward algorithm)**.** The Forward-backward algorithm is used to efficiently estimate the latent sequence given an observation sequence under a HMM. That is, we want to compute

$$p(z_t|x_{1:T}) \quad \forall t \in [1, T]$$

It is computed in two parts, and then multiplied together:

- **Forward Filtering:** computes $p(z_t, x_{1:t})$
- **Backward Filtering:** computes $p(x_{1+t:T}|z_t)$

Note that

$$\begin{aligned} p(z_t|x_{1:T}) &\propto p(z_t, x_{1:T}) \\ &= p(z_t, x_{1:t}) p(x_{t+1:T}|z_t, x_{1:t}) \\ &= \underbrace{p(z_t, x_{1:t})}_{forward\, recursion} \underbrace{p(x_{t+1:T}|z_t)}_{backward\, recursion} \end{aligned}$$

Forward Filtering

$$\begin{aligned} p(z_t, x_{1:t}) &= \sum_{z_{t-1}=1}^{k} p(z_{t-1}, z_t, x_{1:t}) \\ &= \sum_{z_{t-1}=1}^{k} p(x_t|z_{t-1}, z_t, x_{1:t-1}) p(z_t|z_{t-1}, x_{1:t-1}) p(z_{t-1}, x_{1:t-1}) \end{aligned}$$

Define $\alpha_t(z_t) := p(z_t, x_{1:t})$, then

$$\alpha_t(z_t) = p(x_t|z_t) \sum_{z_{t-1}=1}^{k} p(z_t|z_{t-1}) \alpha_{t-1}(z_{t-1})$$

If we recurse all the way down to $\alpha_1(z_1)$, we get

$$\alpha_1(z_1) = p(z_1, x_1) = p(z_1) p(x_1|z_1)$$

Backward Filtering

$$\begin{aligned} p(x_{t+1:T}|z_t) &= \sum_{z_{t+1}=1}^{k} p(z_{t+1}, x_{t+1:T}|z_t) \\ &= \sum_{z_{t+1}=1}^{k} p(x_{t+2:T}|z_{t+1}, z_t, x_{t+1}) p(x_{t+1}|z_{t+1}, z_t) p(z_{t+1}|z_t) \\ &= \sum_{z_{t+1}=1}^{k} p(x_{t+2:T}|z_{t+1}) p(x_{t+1}|z_{t+1}) p(z_{t+1}|z_t) \end{aligned}$$

Define $\beta_t(z_t) := p(x_{1+t:T}|z_t)$, then

$$\beta_t(z_t) = \sum_{z_{t+1}}^{k} \beta_{t+1}(z_{t+1}) p(x_{t+1}|z_{t+1}) p(z_{t+1}|z_t)$$

If we recurse all the way down to $\beta_1(z_1)$, we get

$$\beta_1(z_1) = p(x_{3:T}|z_2) p(x_2|z_2) p(z_2|z_1)$$

5.3 Sampling

A sample form a distribution $p(x)$ is a single realization x whose probability distribution is $p(x)$. This contrasts with the alternative usage in statistics, where sample refers to a collection of realization x.

The problems to be solved The aims of Monte Carlo methods are to solve one or both of the following problems

1. To generate samples $\{x^{(r)}\}_{r=1}^{R}$ from a given probability distribution $p(x)$.
2. To estimate expectations of functions, $f(x)$, under distribution $p(x)$:

$$E = E_{x \sim p(x)}[f(x)] = \int f(x) p(x)\, dx$$

5.3.1 Ancestral Sampling

"Sampling in a topological order"
i.e. at each step, sample from any conditional distribution that you haven't visited yet, whose parents have all been sampled. This procedure will always start with the nodes that have no parents.

Example 5.1. In a chain or HMM, you would always start with z_1 and move to the right. In a tree, you would always start from the root.

Generating marginal samples If you are only interested in sampling a particular set of nodes, you can simply sample from all the nodes jointly, then ignore the nodes you don't need.

Generating conditional samples If you want to sample a variable conditional on a node with no parents, that is also easy - you can simple do ancestral sampling starting from the nodes you have.
However, to sample from a DAG conditional on leaf nodes is hard. Finding ways to do this approximately is what a lot of the rest of the course will be about.

5.4 Simple Monto Carlo

Definition 5.3 (simple Monte Carlo)**.** Given $\{x^{(r)}\}_{r=1}^{R} \sim p(x)$, we estimate the expectation $\mathbb{E}_{x\sim p(x)}[f(x)]$ using the average sum, and call it estimator $\hat{E}$:

$$E = \mathbb{E}_{x\sim p(x)}[f(x)] \sim \frac{1}{R}\sum_{r=1}^{R} f(x^{(r)}) = \hat{E}$$

Property 5.1. If the vectors $\{x^{(r)}\}_{r=1}^{R}$ are generated from $p(x)$ then the expectation of $\hat{E}$ is E. In fact, $\hat{E}$ is an unbiased estimator of E.

Proof.

$$\begin{aligned}
\mathbb{E}[\hat{E}]_{x\sim p(\{x^{(r)}\}_{r=1}^{R})} &= \mathbb{E}[\frac{1}{R}\sum_{r=1}^{R} f(x^{(r)})] \\
&= \frac{1}{R}\sum_{r=1}^{R} \mathbb{E}[f(x^{(r)})] \\
&= \frac{1}{R}\sum_{r=1}^{R} \mathop{\mathbb{E}}_{x\sim p(x)}[f(x)] \\
&= \frac{R}{R}\mathop{\mathbb{E}}_{x\sim p(x)}[f(x)] \\
&= E
\end{aligned}$$

■

Property 5.2. As the number of samples of R increases, the variance of $\hat{E}$ will decrease proportional to $\frac{1}{R}$.

Proof.

$$\begin{aligned} Var[\hat{E}] &= Var\left[\frac{1}{R}\sum_{r=1}^{R} f(x^{(r)})\right] \\ &= \frac{1}{R^2} Var\left[\sum_{r=1}^{R} f(x^{(r)})\right] \\ &= \frac{1}{R^2}\sum_{r=1}^{R} Var\left[f(x^{(r)})\right] && \text{(by i.i.d. assumption)} \\ &= \frac{R}{R^2} Var[f(x)] \\ &= \frac{1}{R} Var[f(x)] \end{aligned}$$

■