Course Notes for Bayesian Models for Machine Learning

John Paisley
Department of Electrical Engineering
Columbia University

Fall 2016

Contents

EECS E6720 Bayesian Models for Machine Learning

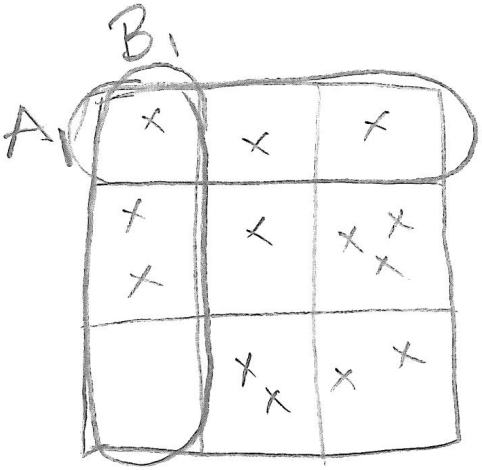
Columbia University, Fall 2016

Lecture 1, 9/8/2016

Instructor: John Paisley

- "Bayes rule" pops out of basic manipulations of probability distributions. Let's reach it through a very simple example.

Example



Call this entire space Ω

A_i is the ith column (defined arbitrarily)

B_i is the ith row (also defined arbitrarily)

- We have points lying in this space. We pick one of these points uniformly at random.
- In this case, calculating probabilities is simply a matter of counting.

$$P(x \in A_1) = \frac{\#A_1}{\#\Omega}, \qquad P(x \in B_1) = \frac{\#B_1}{\#\Omega}$$

- What about $P(x \in A_1 | x \in B_1)$? This is the probability that $x \in A_1$ given that I know $x \in B_1$. This is called a *conditional probability*.

$$P(x \in A_1 | x \in B_1) = \frac{\#(A_1 \cap B_1)}{\#B_1} = \frac{\#(A_1 \cap B_1)}{\#\Omega}\frac{\#\Omega}{\#B_1} = \frac{P(x \in A_1 \,\&\, x \in B_1)}{P(x \in B_1)}$$

- We've simply multiplied and divided by the same thing, but already we've made a general statement.

A more general statement

- Let A and B be two events, then

$$P(A|B) = \frac{P(A,B)}{P(B)} \qquad \Rightarrow \qquad P(A|B)P(B) = P(A,B)$$

- We have some names for these terms,

$$\begin{aligned} P(A|B) &: \text{ conditional probability distribution} \\ P(A,B) &: \text{ joint probability distribution} \\ P(B) &: \text{ marginal probability distribution} \end{aligned}$$

- This last one could be tricky, since it's also just "the probability of B." However, we can use the same counting approach as before to interpret $P(B)$ as a distribution arrived at by integrating (or marginalizing) something out. From the previous example,

$$P(B) = \frac{\#B}{\#\Omega} = \frac{\sum_{i=1}^{3} \#(A_i \cap B)}{\#\Omega} = \sum_{i=1}^{3} \frac{\#(A_i \cap B)}{\#\Omega} = \sum_{i=1}^{3} P(A_i, B)$$

- Side note: In general, the summation and each A_i have a very strict requirement. Each A_i has to be disjoint (no overlaps) and the union of all the A_i has to equal Ω (the entire space).

Getting to Bayes rule

- We're a few easy steps away. We showed that

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

By symmetry we could just as easily have shown that

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

And therefore,

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} = \frac{P(B|A)P(A)}{\sum_i P(A_i, B)} = \frac{P(B|A)P(A)}{\sum_i P(B|A_i)P(A_i)}$$

- This equality is called *Bayes rule*. As you can see, it has a few ways it can be written.

Bayes rule

- And so we have that $P(A|B) = \dfrac{P(B|A)P(A)}{P(B)}$

- These values each have a name:

$$\text{posterior} = \frac{\text{likelihood} \times \text{prior}}{\text{evidence}}$$

- Imagine that we don't know A, but we get some information about it in the form of B. Bayes rule tells us a principled way to incorporate this information in our belief about A.

Example (medical test)

- We have two binary values, A and B:

$$A = \begin{cases} 1 & \text{person has disease} \\ 0 & \text{no disease} \end{cases} \qquad B = \begin{cases} 1 & \text{test for disease "positive"} \\ 0 & \text{test is "negative"} \end{cases}$$

- A person tests "positive" for the disease. What's the probability he has it?

- We want $P(A = 1|B = 1)$. Does Bayes rule help? Bayes rule says that

$$\begin{aligned} P(A = 1|B = 1) &= \frac{P(B = 1|A = 1)P(A = 1)}{P(B = 1)} \\ &= \frac{P(B = 1|A = 1)P(A = 1)}{P(B = 1|A = 1)P(A = 1) + P(B = 1|A = 0)P(A = 0)} \end{aligned}$$

- Image we can estimate that

$$P(B = 1|A = 1) = 0.95, \quad P(B = 1|A = 0) = 0.05, \quad P(A = 1) = 0.01 = 1 - P(A = 0)$$

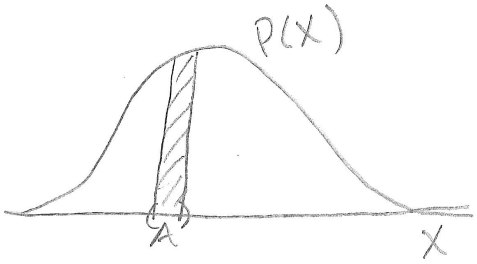
Then plugging in, we have that $P(A = 1|B = 1) = 0.16$

Continuous space

- We've been talking about discrete distributions so far. That is, the number of possible values the unknowns can take is finite.

- When values are in a continuous space, we switch to continuous distributions.

Example



$x \in \mathbb{R}$, $p(x)$ is its density (represented by the switch to lower case)

$p(x) \geq 0$ and $\int p(x)dx = 1$

$P(x \in A) = \int_A p(x)dx$

- The same rules apply as for discrete random variables

- $p(x|y) = \dfrac{p(x,y)}{p(y)}, \qquad p(y) = \displaystyle\int p(x,y)dx$
- $p(x|y)p(y) = p(x,y) = p(y|x)p(x)$

- This leads to *Bayes rule* for continuous random variables

$$p(y|x) \;=\; \frac{p(x|y)p(y)}{p(x)} \;=\; \frac{p(x|y)p(y)}{\int p(x|y)p(y)dy}$$

- The difference is that we are dealing with continuous functions.

Bayesian modeling

- Applying Bayes rule to the unknown variables of a data modeling problem is called Bayesian modeling.
- In a simple, generic form we can write this process as

  $x \sim p(x|y) \;\leftarrow$ The data-generating distribution. This is the model of the data.

  $y \sim p(y) \;\leftarrow$ The model prior distribution. This is what we think about y a priori.

- We want to learn y. We write that

$$p(y|x) \propto p(x|y)p(y)$$

- In the above line, we don't know $p(y|x)$ and want to calculate it. It answers the question "Having seen x, what can we say about y?" The right two terms $p(x|y)p(y)$ we do know because we have *defined* it. The symbol $\propto$ is read as "is distributed as."
- This is the general form of the problems discussed in this class. However, it is non-trivial because
  1. $p(x|y)$ can be quite complicated
  2. $p(y|x)$ can be intractable because the integral in the normalizing constant doesn't have a closed-form solution, thus requiring an algorithm to approximate

- These two issues will make up the focus of this class: Defining various models on the structure of the data-generating phenomenon, and defining inference algorithms for learning the posterior distribution of that model's variables.

Simple example: beta-Bernoulli model

- We have a sequence of observations $X_1, \ldots, X_N$, where $X_i = 1$ indicates "success" and $X_i = 0$ indicates "failure." Think of them as results of flipping a coin.
- We hypothesize that each X_i is generated by flipping a biased coin, where $P(X_i = 1|\pi) = \pi$.

- We further assume the X_i are *independent and identically distributed* (iid). This means the X_i are *conditionally independent* given π. Mathematically this means we can write

$$P(X_1, \ldots, X_N|\pi) = \prod_{i=1}^{N} P(X_i|\pi)$$

- Since $P(X_i|\pi) = \pi^{X_i}(1-\pi)^{1-X_i}$, we can write

$$P(X_1, \ldots, X_N|\pi) = \prod_{i=1}^{N} \pi^{X_i}(1-\pi)^{1-X_i}$$

- We are interested in the posterior distribution of π given $X_1, \ldots, X_N$, i.e.,

$$\begin{aligned} P(\pi|X_1, \ldots, X_N) &\propto P(X_1, \ldots, X_N|\pi)p(\pi) \\ &\propto \prod_{i=1}^{N} \pi^{X_i}(1-\pi)^{1-X_i}p(\pi) \end{aligned}$$

- We've come across the next significant problem: What do we set $p(\pi)$ to?

A first try

- Let $p(\pi) = Uniform(0,1) \;\Rightarrow\; p(\pi) = \mathbb{1}(0 \leq \pi \leq 1)$
- Then by Bayes rule,

$$p(\pi|X_1, \ldots, X_N) = \frac{\pi^{(\sum_i X_i+1)-1}(1-\pi)^{(N-\sum_i X_i+1)-1}}{\int_0^1 \pi^{\sum_i X_i}(1-\pi)^{N-\sum_i X_i} d\pi}$$

- The normalizing constant is tricky, but fortunately mathematicians have solved it,

$$p(\pi|X_1, \ldots, X_N) = \frac{\Gamma(N+2)}{\Gamma(1+\sum_i X_i)\Gamma(1+N-\sum_i X_i)} \pi^{\sum_i X_i+1-1}(1-\pi)^{N-\sum_i X_i+1-1}$$

  ($\Gamma(\cdot)$ is called the "gamma function")

- This is a very common distribution called a *beta distribution*:

$$Beta(a,b) = \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)} \pi^{a-1}(1-\pi)^{b-1}$$

- In the case of the above posterior distribution, $a = 1 + \sum_i X_i$ and $b = 1 + N - \sum_i X_i$.
- Notice that when $a = b = 1$, $Beta(1,1) = Uniform(0,1)$ which was the prior we chose.

A conjugate prior

- The beta distribution looks a lot like the likelihood term. Also, because it has the $Uniform(0,1)$ distribution as a special case, it would give us a wider range of prior beliefs that we could express. What if we try the beta distribution as the prior for π?

$$\begin{aligned} p(\pi|X_1,\ldots,X_N) &\propto p(X_1,\ldots,X_N|\pi)p(\pi) \\ &\propto \left[\pi^{\sum_i X_i}(1-\pi)^{N-\sum_i X_i}\right]\left[\frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)}\pi^{a-1}(1-\pi)^{b-1}\right] \\ &\propto \pi^{a+\sum_i X_i-1}(1-\pi)^{b+N-\sum_i X_i-1} \end{aligned}$$

- We can notice a few things here:

  1. I threw away $\frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)}$ in the last line. This is because it doesn't depend on π, and so it can be absorbed in the normalizing constant. In other words, the normalizing constant is the integral of the numerator. If we divide the second line by this integral, we can immediately cancel out $\frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)}$. The result is the third line divided by the integral of the third line. The only reason to do something like this is for convenience and to make things look simpler.

  2. In this case, we know what to divide this by to make it a probability distribution on π. The last line is proportional to a $Beta(a+\sum_i X_i, b+N-\sum_i X_i)$ distribution.

- Therefore, $p(\pi|X_1,\ldots,X_N) = Beta(a+\sum_i X_i, b+N-\sum_i X_i)$.

- Notice that for this problem, when we select the prior $p(\pi)$ to be a beta distribution, we find that the posterior distribution is also a beta distribution with different parameters. This is because the beta distribution is a *conjugate prior* for this problem.

- We say that a distribution is a "conjugate prior" for a particular variable in a likelihood, or that it is "conjugate to the likelihood," if the posterior is in the same distribution family as the prior, but has updated parameters. Conjugate priors are very convenient, since we only need to collect *sufficient statistics* from the data in order to transition from the prior to the posterior. For example, in the problem above we only need to count the total number of "successes" ($\sum_i X_i$) and "failures" ($N-\sum_i X_i$).

- will see many conjugate priors in this course and how they can result in fast inference algorithms for learning models.

What do we gain by being Bayesian?

- For the previous modeling problem with a beta prior, consider the expectation and variance of π under the posterior distribution.

$$\mathbb{E}[\pi] = \frac{a+\sum_i X_i}{a+b+N}, \qquad Var(\pi) = \frac{(a+\sum_i X_i)(b+N-\sum_i X_i)}{(a+b+N)^2(a+b+N-1)}$$

- Notice that as N increases,

    1. The expectation converges to the empirical success rate.

    2. The variance decreases like $1/N$, i.e., it's going to zero.

- Compare this with *maximum likelihood*, in which we seek a point estimate (on specific value) of π to maximize the likelihood term only

$$\pi = \arg\max_{\pi} \; p(X_1, \ldots, X_N|\pi) \; = \; \frac{1}{N} \sum_{i=1}^{N} X_i$$

- The Bayesian approach is capturing our uncertainty about the quantity we are interested in. Maximum likelihood does not do this.

- As we get more and more data, the Bayesian and ML approaches agree more and more. However, Bayesian methods allow for a smooth transition from uncertainty to certainty.

EECS E6720 Bayesian Models for Machine Learning

Columbia University, Fall 2016

Lecture 2, 9/15/2016

Instructor: John Paisley

- Next, we look at another instance of a conjugate prior. To help motivate the practical usefulness of the distributions we will consider, we discuss this prior in the context of a regression problem.

Problem setup

- We're often given a data set of the form $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$ where $x \in \mathbb{R}^d$ and $y \in \mathbb{R}$. The goal is to learn a prediction rule from this data so that, given a new $\hat{x}$ we can predict its associated an unobserved $\hat{y}$. This is called a regression problem.

- We often refer to x as "covariates" or "features" and y as a "response."

Linear regression

- One model for this problem is to assume a linear relationship between the inputs x and outputs y such that

$$y_i = x_i^T w + \epsilon_i$$

The term ϵ_i accounts for the fact that we usually can't find a w such that $y_i = x_i^T w$ for all i. The model value of interest is w and we simply set $\epsilon_i = y_i - x_i^T w$. Further assumptions let $w \in \mathbb{R}^d$ and $\epsilon_i \overset{iid}{\sim} \text{Normal}(0, \sigma^2)$.

Likelihood term for y

- Using only this information, we have an implied likelihood term of y given X and w (where $X = \{x_i\}$).

- First, notice that for each i

$$y_i \overset{ind}{\sim} \text{Normal}(x_i^T w, \sigma^2)$$

where we recall that $\text{Normal}(y_i | x_i^T w, \sigma^2) = (2\pi\sigma^2)^{\frac{1}{2}} \exp\{-\frac{1}{2\sigma^2}(y_i - x_i^T w)^2\}$.

- Because of the conditional independence assumption of y_i given w, we can write the likelihood as

$$p(y_1, \ldots, y_N | w, X) = \prod_{i=1}^{N} p(y_i | w, x_i)$$

- The vector w is the model parameter that is unknown as we want to learn it. Bayesian linear regression takes the additional step of treating w as a random variable with a prior distribution. Hence in the Bayesian setting we refer to w as a model variable instead of a model parameter.

- After defining a prior $p(w)$, we use Bayes rule to learn a posterior distribution on w:

$$p(w|X, \vec{y}) = \frac{p(\vec{y}|X, w)p(w)}{\int_{\mathbb{R}^d} p(\vec{y}|X, w)p(w)dw} = \frac{\prod_{i=1}^{N} p(y_i|x_i, w)p(w)}{\int_{\mathbb{R}^d} \prod_{i=1}^{N} p(y_i|x_i, w)p(w)dw}$$

- Question: Why is X always being conditioned on? (i.e., on the RHS of |)

- Answer: Look at what we have distributions on.
  - The model assumes a generative distribution for y
  - We put a prior distribution on w
  - We haven't assumed any distribution on x_i
  - In short, this results from the *model assumptions* that we have made

Prior on w

- We still have to define the prior distribution on w. *Out of convenience* we define

$$w \sim \text{Normal}(0, \lambda^{-1} I)$$

Why do we do this?
Because it's conjugate to the likelihood. As a result, the posterior distribution of w is a recognizable distribution with known parameters that can be easily calculated from the data.

Posterior of w

- We next calculate the posterior distribution of w for this model. The steps we take are as follows:

Using Bayes rule we have that

$$p(w|X, \vec{y}) \propto \prod_{i=1}^{N} \text{Normal}(y_i | x_i^T w, \sigma^2) \text{Normal}(w | 0, \lambda^{-1} I)$$

By direction plugging in using the form of a Gaussian, we have that

$$p(w|X, \vec{y}) \propto \left[\prod_{i=1}^{N} \mathrm{e}^{-\frac{1}{2\sigma^2}(y_i - x_i^T w)^2} \right] \left[\mathrm{e}^{-\frac{\lambda}{2} w^T w} \right]$$

We don't need to include the constants out front $(2\pi\sigma^2)^{-\frac{1}{2}}$ and $(\lambda/(2\pi))^{\frac{1}{2}}$ because they don't depend on the unknown part of the distribution, w. When writing out the normalizing constant as the integral of the numerator, one can see that these terms appear in the numerator and denominator and cancel out. The only reason we do this is for convenience (it's less to write).

I'll work through the full derivation below. You can skip to Step 7 for the final result.

1. The first step to calculate $p(w|X,\vec{y})$ is to combine into one exponential by summing the terms in the exponent,

$$p(w|X,\vec{y}) \propto \mathrm{e}^{-\frac{\lambda}{2}w^Tw-\frac{1}{2\sigma^2}\sum_{i=1}^N(y_i-x_i^Tw)^2}$$

2. In the next step, we expand the square to write an equation quadratic in w

$$p(w|X,\vec{y}) \propto \exp\Big\{-\tfrac{1}{2}\Big[w^T(\lambda I+\tfrac{1}{\sigma^2}\textstyle\sum_i x_ix_i^T)w-2w^T(\tfrac{1}{\sigma^2}\sum_i y_ix_i)+\tfrac{1}{\sigma^2}\sum_i y_i^2\Big]\Big\}$$

3. Next, we write

$$\begin{aligned} p(w|X,\vec{y}) &\propto \exp\Big\{-\tfrac{1}{2}\Big[w^T(\lambda I+\tfrac{1}{\sigma^2}\textstyle\sum_i x_ix_i^T)w-2w^T(\tfrac{1}{\sigma^2}\sum_i y_ix_i)\Big]\Big\}\exp\Big\{-\tfrac{1}{2}\Big[\tfrac{1}{\sigma^2}\textstyle\sum_i y_i^2\Big]\Big\} \\ &\propto \exp\Big\{-\tfrac{1}{2}\Big[w^T(\lambda I+\tfrac{1}{\sigma^2}\textstyle\sum_i x_ix_i^T)w-2w^T(\tfrac{1}{\sigma^2}\sum_i y_ix_i)\Big]\Big\} \end{aligned}$$

   In the first line, we just moved things around, but didn't change the actual function. We did this to justify the second line, which again takes advantage of the fact that we only care about proportionality. If "$\propto$" were instead "$=$," this would of course be mathematically wrong. However, "$\propto$" allows us to treat $\exp\{-\frac{1}{2}[\sigma^{-2}\sum_i y_i^2]\}$ as a pre-multiplying constant w.r.t. w that will cancel out when we divide the numerator by its integral over w to get the posterior distribution and return to "$=$." We get rid of it just for convenience.

4. In the fourth step, we invert this process by multiplying by a term of our choosing that's constant with respect to w. The goal is to complete the square in the exponential term. We choose to multiply this term by the somewhat arbitrary looking term

$$\exp\Big\{-\tfrac{1}{2}\Big[(\tfrac{1}{\sigma^2}\textstyle\sum_i y_ix_i)^T(\lambda I+\tfrac{1}{\sigma^2}\sum_i x_ix_i^T)^{-1}(\tfrac{1}{\sigma^2}\sum_i y_ix_i)\Big]\Big\}$$

   However, notice that this doesn't involve w, so by multiplying with it we aren't violating the proportionality w.r.t. w. Think of this as working with a function of w, and scaling that function up and down until we reach the point that it integrates to 1.

5. By multiplying the last term in Step 3 with the term in Step 4 and shuffling things around, we get a long term that I will break down as follows

$$p(w|X,\vec{y}) \propto \exp\Big\{-\tfrac{1}{2}(w-\mu)^T\Sigma^{-1}(w-\mu)\Big\}$$

   where

$$\Sigma=(\lambda I+\tfrac{1}{\sigma^2}\textstyle\sum_i x_ix_i^T)^{-1} \quad \text{and} \quad \mu=\Sigma(\tfrac{1}{\sigma^2}\sum_i y_ix_i)$$

   You can verify this by plugging in for μ and Σ, expanding the quadratic term, pulling out the constant and seeing that the result is the multiplication of Step 3 with Step 4.

6. Finally, we want to calculate $p(w|X, \vec{y})$ exactly. We're hoping we can find a closed form solution to the integral in

$$p(w|X, \vec{y}) = \frac{\exp\{-\frac{1}{2}(w-\mu)^T\Sigma^{-1}(w-\mu)\}}{\int \exp\{-\frac{1}{2}(w-\mu)^T\Sigma^{-1}(w-\mu)\}dw}$$

where I use the same definition of μ and Σ as in Step 5. Approaching this problem from a purely mathematical standpoint, this is not easy at all. However, mathematicians and statisticians have been collecting known probability distributions for our reference. At some point, someone actually solved this integral and now we have the solution without having to recalculate the integral each time.

Specifically, we know that a d-dimensional multivariate Gaussian distribution with mean μ and covariance Σ has the form

$$p(w|\mu, \Sigma) = (2\pi)^{-\frac{d}{2}}|\Sigma|^{-\frac{1}{2}}\mathrm{e}^{-\frac{1}{2}(w-\mu)^T\Sigma^{-1}(w-\mu)}$$

We know from other people's calculations that this function of w integrates to 1. This function is also proportional to the function in Step 5. In Step 6 we're solving the integral in order to find the constant to multiply Step 5 with to make the function integrate to 1. Therefore, we know this constant is $(2\pi)^{-\frac{d}{2}}|\Sigma|^{-\frac{1}{2}}$ and

$$\int \exp\{-\frac{1}{2}(w-\mu)^T\Sigma^{-1}(w-\mu)\}dw = (2\pi)^{\frac{d}{2}}|\Sigma|^{\frac{1}{2}}$$

It's an argument based on logic and based on trust that previous proofs that the nonnegative function $(2\pi)^{-\frac{d}{2}}|\Sigma|^{-\frac{1}{2}}\mathrm{e}^{-\frac{1}{2}(w-\mu)^T\Sigma^{-1}(w-\mu)}$ integrates to 1 are correct, which is the stopping point for this class (and 99.99% of all "Bayesians" including myself).

7. Therefore, we have shown that the posterior distribution $p(w|X, \vec{y})$ is a multivariate Gaussian with mean μ and covariance Σ, where

$$\Sigma = (\lambda I + \tfrac{1}{\sigma^2}\textstyle\sum_i x_i x_i^T)^{-1} \qquad \mu = \Sigma(\tfrac{1}{\sigma^2}\textstyle\sum_i y_i x_i)$$

Notice that this is in the same family as the prior $p(w)$, which was also a multivariate Gaussian. Therefore we chose a *conjugate prior* for w.

Making predictions

- For many applications, the posterior distribution of a model's variables gives useful information. For example, it often provides information on the structure underlying the data, or cause and effect type relationships.

- For example, in the linear regression model $y = x^T w + \epsilon$ we've been discussing, the vector w tells us the relationship between the inputs of x and output y. e.g., if w_k is positive the model will lead us to believe that increasing the kth dimension of x will increase y and by how much. We might then use this information to figure out which dimensions we need to "improve."

- Often, however, we just want to make predictions on new data using the model of the old data.

Predictive distribution

- For Bayesian linear regression, we ultimately want to predict a new $\hat{y}$ given its associated $\hat{x}$ and all previously observed (x, y) pairs. In other words, we wish to form the predictive distribution

$$p(\hat{y}|\hat{x}, \vec{y}, X) = \int_{\mathbb{R}^d} p(\hat{y}|\hat{x}, w)p(w|\vec{y}, X)dw$$

- Notice that the predictive distribution can be written as a marginal distribution over the model variables.

- However, implied in the left hand side is the model definition, which tells us which variables to integrate over on the right hand side, as well as the specific distributions to plug in.

- That is, just writing a predictive distribution $p(\hat{y}|\hat{x}, \vec{y}, X)$ is not enough information to know how to represent it as a marginal distribution, where marginalization (i.e., integration) takes place over the model variables. It is necessary to know in advance what the underlying model is in order to know what model variables to integrate over.

- Let's step back and think more generally about what we're doing when we construct a predictive probability distribution.

  Bayesian model: The Bayesian modeling problem is summarized in the following sequence.

$$\begin{aligned} \text{Model of data:} \quad & X \sim p(X|\theta) \\ \text{Model prior:} \quad & \theta \sim p(\theta) \\ \text{Model posterior:} \quad & p(\theta|X) = p(X|\theta)p(\theta)/p(X) \end{aligned}$$

  Predictions: The goal of making predictions is to use past data to predict the future under the same modeling assumption. This is why we choose to model data in the first place: We assume there is an underlying rule governing the way data is created whether its seen or unseen data. By defining a model we define a rule. Seeing some data lets us get a better sense of what that rule should be, letting us learn from the past. Therefore, when making predictions we assume future data follows the same rule as past data,

$$\text{Future data:} \quad \hat{x} \sim p(x|\theta) \qquad \leftarrow \text{the same model as the past data}$$

  We don't know θ, but the past data X, along with our prior believe about θ, gives us information about it. In fact we have a posterior distribution on it $p(\theta|X)$. We can use this to "score" or "weight" each possible setting of θ. This is the marginalization procedure,

$$\text{Predictive distribution:} \quad p(\hat{x}|X) = \int p(\hat{x}|\theta)p(\theta|X)d\theta$$

  So by marginalizing, we are considering the infinite number of possible settings for θ in the likelihood and weighting it by our posterior belief of how probable that setting is. For certain probability distributions, we can solve this integral analytically.

- The Bayesian linear regression model we've been discussing is one example where we can solve for the predictive distribution. Our goal is to solve the integral

$$p(\hat{y}|\hat{x},\vec{y},X) = \int \underbrace{(2\pi\sigma^2)^{-\frac{1}{2}}\mathrm{e}^{-\frac{1}{2\sigma^2}(\hat{y}-\hat{x}^T w)^2}}_{=\,p(\hat{y}|\hat{x},w)} \underbrace{(2\pi)^{-\frac{d}{2}}|\Sigma|^{-\frac{1}{2}}\mathrm{e}^{-\frac{1}{2}(w-\mu)^T\Sigma^{-1}(w-\mu)}}_{=\,p(w|\vec{y},X)} dw$$

  Notice that we keep all the terms this time because we are writing an equality here, not a proportionality. The specific values of μ and Σ for the regression problem were derived above.

- Below I'll derive this predictive distribution. The trick this time will be to multiply and divide by the same specifically chosen value. You can skip to Step 5 for the result.

  1. First, we expand both squares and pull everything not depending on w out in front of the integral

$$\begin{aligned} p(\hat{y}|\hat{x},\vec{y},X) &= (2\pi\sigma^2)^{-\frac{1}{2}}(2\pi)^{-\frac{d}{2}}|\Sigma|^{-\frac{1}{2}}\mathrm{e}^{-\frac{1}{2\sigma^2}\hat{y}^2-\frac{1}{2}\mu^T\Sigma^{-1}\mu} \\ &\quad \times \int \mathrm{e}^{-\frac{1}{2}\left(w^T(\hat{x}\hat{x}^T/\sigma^2+\Sigma^{-1})w-2w^T(\hat{x}\hat{y}/\sigma^2+\Sigma^{-1}\mu)\right)}dw \end{aligned}$$

  2. Next we multiply and divide by a constant w.r.t. w. We want to choose a constant to result in the integral equaling 1. We notice that we can complete the square in the exponent, allowing us to put it into the form of a Gaussian distribution. Therefore, multiply and divide by the term

$$(2\pi)^{-\frac{d}{2}}\left|\sigma^{-2}\hat{x}\hat{x}^T+\Sigma^{-1}\right|^{\frac{1}{2}}\mathrm{e}^{-\frac{1}{2}(\hat{x}\hat{y}/\sigma^2+\Sigma^{-1}\mu)^T(\hat{x}\hat{x}^T/\sigma^2+\Sigma^{-1})^{-1}(\hat{x}\hat{y}/\sigma^2+\Sigma^{-1}\mu)}$$

  3. If we put this extra term in the numerator to the right of the integral and the term in the denominator to the left of the integral, we can complete the square in the right-most exponent and see that a Gaussian results. Therefore we know the integral equals 1 and we can simply remove this term. The result is

$$p(\hat{y}|\hat{x},\vec{y},X) = \frac{(2\pi\sigma^2)^{-\frac{1}{2}}(2\pi)^{-\frac{d}{2}}|\Sigma|^{-\frac{1}{2}}\mathrm{e}^{-\frac{1}{2\sigma^2}\hat{y}^2-\frac{1}{2}\mu^T\Sigma^{-1}\mu}}{(2\pi)^{-\frac{d}{2}}\left|\sigma^{-2}\hat{x}\hat{x}^T+\Sigma^{-1}\right|^{\frac{1}{2}}\mathrm{e}^{-\frac{1}{2}(\hat{x}\hat{y}/\sigma^2+\Sigma^{-1}\mu)^T(\hat{x}\hat{x}^T/\sigma^2+\Sigma^{-1})^{-1}(\hat{x}\hat{y}/\sigma^2+\Sigma^{-1}\mu)}}$$

  4. The final step uses two important matrix equalities that are often useful. In the context of the equation above, these are

$$\left|\sigma^{-2}\hat{x}\hat{x}^T+\Sigma^{-1}\right| = |\Sigma^{-1}|(1+\hat{x}^T\Sigma\hat{x}/\sigma^2) = |\Sigma|^{-1}(1+\hat{x}^T\Sigma\hat{x}/\sigma^2)$$

  plugging this in above, several terms cancel out and we end up multiplying the ratio of exponents by $(2\pi)^{-\frac{1}{2}}(\sigma^2+\hat{x}^T\Sigma\hat{x})^{-\frac{1}{2}}$ The second useful equality is called the matrix inversion lemma. This allows us to say that

$$(\hat{x}\hat{x}^T/\sigma^2+\Sigma^{-1})^{-1} = \Sigma+\Sigma\hat{x}(\sigma^2+\hat{x}^T\Sigma\hat{x})^{-1}\hat{x}^T\Sigma$$

  Notice that the inverted term is a scalar, not a matrix. We plug this in where it appears in the exponent, and combine into one exponential. The bookkeeping is fairly involved, but we get terms that cancel and can write the resulting quadratic term as a square. As a result, the exponential term is $\mathrm{e}^{-\frac{1}{2(\sigma^2+\hat{x}^T\Sigma\hat{x})}(\hat{y}-\hat{x}^T\mu)^2}$.

5. As a result, we have calculated that

$$p(\hat{y}|\hat{x}, \vec{y}, X) = (2\pi)^{-\frac{1}{2}}(\sigma^2 + \hat{x}^T\Sigma\hat{x})^{-\frac{1}{2}}\mathrm{e}^{-\frac{1}{2(\sigma^2+\hat{x}^T\Sigma\hat{x})}(\hat{y}-\hat{x}^T\mu)^2}$$

Notice that this is a univariate Normal distribution with mean $\hat{x}^T\mu$ and variance $\sigma^2+\hat{x}^T\Sigma\hat{x}$. Again, the terms μ and Σ are calculated using the prior parameters and the data $\vec{y}$ and X.

Another modeling example

- Let's look at another Bayesian modeling example that is slightly more complicated.

Problem setup

- Again we're given a data set $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$ where $x \in \Omega_x$, which is an arbitrary space, and this time $y \in \{0, 1\}$. The goal is to *classify* a new x by assigning a label to it. That is, we want to predict the unknown value of $y \in \{0, 1\}$ associated with a new input x.

Bayesian classifier model

- This approach to classification assumes a generative process for *both* y and x. For the model, we will assume that each (x, y) pair is generated i.i.d. as follows

$$\text{Model:} \quad y \overset{iid}{\sim} \text{Bernoulli}(\pi), \quad x|y \overset{ind}{\sim} p(x|\theta_y)$$

To draw x, we need to know y, which is what $x|y$ is meant to convey.

- As is evident, the joint distribution on (x, y) is not a distribution that can be factorized into separate distributions on x and y. Rather, it must be factorized as

$$p(x, y|\Theta, \pi) = p(x|y, \Theta)p(y|\pi)$$

This factorization says that we can first sample a value for y without knowing x, after which we sample x from a distribution that depends on the value of y and some parameters Θ. This is also what the generative process above says. In fact, there we further assume $\Theta = \{\theta_0, \theta_1\}$ and let y pick out the correct parameters from this set.

- We can think of this as follows, imagine x contains the text of an email and y indicates whether it is spam or not. The generative model assumes that spam and non-spam emails are generated from the same distribution family, but with different parameters. All spam emails share one parameter, θ_1, and all non-spam emails share another parameter θ_0. The data is generated by first choosing whether a spam or non-spam email will be generated by flipping a coin with bias π, and then generating the email itself from a distribution using the class-specific parameter.

- It should be clear that x doesn't have to be an email, but something else, for example several statistics from a patient during a health checkup that can be used to diagnose an illness. The x for this problem will be in a different domain than for the email problem, which is why it is beneficial to keep the distribution and space of x general.

Prior distributions

- The model variables are the class probability $\pi \in (0, 1)$ and class-specific variables θ_1, θ_0. We next choose prior distributions for these. We select for π out of convenience and write a generic prior for each θ

$$\pi \sim \text{Beta}(a, b), \qquad \theta_0, \theta_1 \overset{iid}{\sim} p(\theta)$$

We are making one assumption here about θ, that the class-specific variables are drawn *independently* from the same prior distribution. This will let us simplify the joint likelihood of the model for posterior computation.

Posterior computation

- Again we want to find the posterior, but this time of π, θ_0 and θ_1. Using Bayes rule,

$$p(\pi, \theta_1, \theta_0 | \vec{y}, X) = \frac{p(X, \vec{y} | \pi, \theta_1, \theta_0) p(\pi, \theta_1, \theta_0)}{\int_{\Omega_\theta} \int_{\Omega_\theta} \int_0^1 p(X, \vec{y} | \pi, \theta_1, \theta_0) p(\pi, \theta_1, \theta_0) d\pi d\theta_1 d\theta_0}$$

- This time we have three variables instead of one. However, looking closer we will see that these distributions factorize nicely. (This is the exception, not the rule.)

- Prior: The prior can be written $p(\pi, \theta_1, \theta_0) = p(\pi)p(\theta_1)p(\theta_0)$. Of course $p(\theta_1)$ and $p(\theta_0)$ are the same exact distribution, but evaluated at different points.

- Likelihood: By the modeling assumption, we can write the likelihood as

$$\begin{aligned} p(X, \vec{y} | \pi, \theta_1, \theta_0) &= \prod_{i=1}^N p(x_i, y_i | \pi, \theta_1, \theta_0) \\ &= \prod_{i=1}^N p(x_i | y_i, \pi, \theta_1, \theta_0) p(y_i | \pi, \theta_1, \theta_0) \\ &= \prod_{i=1}^N p(x_i | \theta_{y_i}) p(y_i | \pi) \end{aligned}$$

The product is from the independence assumption. The transition from the first to second line is a rule of probability that's always true. The transition from the second to third line simplifies the to reflect the dependence structure assumed by the model.

- Finally, we can return to Bayes rule and try computing it. In the numerator, we group the multiplications across the three variables

$$p(\pi, \theta_1, \theta_0 | \vec{y}, X) \propto \left[\prod_{i:y_i=1} p(x_i | \theta_1) p(\theta_1) \right] \left[\prod_{i:y_i=0} p(x_i | \theta_0) p(\theta_0) \right] \left[\prod_{i=1}^N p(y_i | \pi) p(\pi) \right]$$

- The normalizing constant is the integral of this term. Notice that since we can write this as the product of three separate functions of over each model variable, the posterior can be written as

$$p(\pi, \theta_1, \theta_0 | \vec{y}, X) = \frac{\prod_{i:y_i=1} p(x_i|\theta_1)p(\theta_1)}{\int \prod_{i:y_i=1} p(x_i|\theta_1)p(\theta_1)d\theta_1} \cdot \frac{\prod_{i:y_i=0} p(x_i|\theta_0)p(\theta_0)}{\int \prod_{i:y_i=0} p(x_i|\theta_0)p(\theta_0)d\theta_0} \cdot \frac{\prod_{i=1}^{N} p(y_i|\pi)p(\pi)}{\int \prod_{i=1}^{N} p(y_i|\pi)p(\pi)d\pi}$$

  However, this is just the product of three instances of Bayes rule and is equivalently saying

$$p(\pi, \theta_1, \theta_0 | \vec{y}, X) = p(\theta_1 | \{x_i : y_i = 1\}) p(\theta_0 | \{x_i : y_i = 0\}) p(\pi | \vec{y})$$

- Last week we worked through $p(\pi|\vec{y})$. We saw that this was the posterior of a beta-Bernoulli process,

$$p(\pi|\vec{y}) \propto \prod_{i=1}^{N} p(y_i|\pi)p(\pi) \quad \longrightarrow \quad p(\pi|\vec{y}) = \text{Beta}(a + \textstyle\sum_i y_i, b + N - \sum_i y_i)$$

  That is, for the class probability, we simply sum the number of times our data was in each class and use those values as the parameters in the beta distribution.

- The other class-conditional posterior distributions on θ_0 and θ_1 are problem-specific. We notice the feature of these distributions is that we partition the data set into two groups, those belonging to class 1 and those in class 0. Then, we wish to solve (for, e.g., class 1)

$$p(\theta_1 | \{x_i : y_i = 1\}) \propto \prod_{i:y_i=1} p(x_i|\theta_1)p(\theta_1)$$

  In many cases we can do this. For example, when $x \in \mathbb{R}^d$ and $p(x|\theta)$ is a Gaussian, we can pick a conjugate prior for the mean and covariance.

Naive Bayes classifier

- When x is a complex multidimensional object, a simplifying assumption about $p(x|\theta)$ is sometimes made on how this can be further factorized. To give an overview, let x be composed of m pieces of information possibly in different domains $x = \{x^{(1)}, \ldots, x^{(m)}\}$. For example, some $x^{(j)}$ could real numbers and some non-negative integers. A naive Bayes classification model makes the assumption that the variables θ can be separate into m groups as well and that

$$p(x|\theta) = \prod_{j=1}^{m} p(x^{(j)}|\theta^{(j)})$$

- Separate priors are selected for each $\theta^{(j)}$, possibly in different distribution families, and we assume $\theta_0^{(j)}, \theta_1^{(0)} \overset{iid}{\sim} p_j(\theta^{(j)})$.

- The Bayes classifier is then computed as above, after which one finds (for class 1, for example)

$$p(\theta_1 | \{x_i : y_i = 1\}) = \prod_{j=1}^{m} p_j(\theta_1^{(j)} | \{x_i^{(j)} : y_i = 1\})$$

- Spam detection: To make this more concrete, consider the problem of spam detection. In the spam detector model we will consider here, a pair (x, y) consists of a label $y \in \{0, 1\}$ indicating "spam" or "not spam" and x is a v dimensional vector of word counts for a vocabulary of size v. Thus $x(j)$ is a count of how many times word j (e.g., j maps to "deal") appears in the email. We need to define a distribution on x as its model, which can also be thought of as a joint probability distribution on all the values in x,

$$p(x|\theta) \quad \Leftrightarrow \quad p(x(1), \ldots, x(v)|\theta)$$

Naive Bayes takes the additional modeling step of breaking the correlation structure between these values

$$p(x(1), \ldots, x(v)|\theta) = \prod_{j=1}^{v} p(x(j)|\theta(j)) \quad \leftarrow \text{a modeling assumption – we define this}$$

From a generative perspective, we are saying that for an observation x_i, each "piece" of x_i is generated independently from its own distribution given its class. The assumption that $x_i(j)$ and $x_i(j')$ are independent given its class variable θ_{y_i} is what is "naive" about this model.

We use Bayes rule to calculate the posterior of π exactly as before. Using it to calculate the posterior of each θ_y can be split into the process of calculating the posterior of each $\theta_y(j)$ separately. For example, in spam detection we can define $p(x(j)|\theta_y(j))$ to be a Poisson distribution, and $p(\theta_y(j))$ to be a gamma distribution. These two distributions are conjugate and so the posterior of $\theta_y(j)$ will also be gamma. Because of independence assumptions, solving for the posterior of θ_y can be separated into v independent applications of Bayes rule, one for each $\theta_y(j)$.

Predictive distribution

- As with Bayesian linear regression, a major application of the Bayes classifier is to predicting the labels of new data. That is, given a *training* set $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$ of labeled data we want to learn a model so that we can predict the unobserved label $\hat{y} \in \{0, 1\}$ of a newly observed $\hat{x}$.

- Again we want to form the predictive distribution $p(\hat{y}|\hat{x}, X, \vec{y})$. This can be done, but is not as straightforward as before. We'll try two ways. The first way will lead to a dead-end. The second approach will work.

First attempt

- We want to calculate

$$p(\hat{y}|\hat{x}, X, \vec{y}) = \int_{\Omega_\theta} \int_{\Omega_\theta} \int_{0}^{1} p(\hat{y}|\hat{x}, \theta_1, \theta_0, \pi) p(\theta_1, \theta_0, \pi|X, \vec{y}) d\pi d\theta_1 d\theta_0$$

There are two distributions we need to know. The second one is the posterior that we previously calculated and showed for this model can be factorized as

$$p(\theta_1, \theta_0, \pi|X, \vec{y}) = p(\theta_1|X, \vec{y}) p(\theta_0|X, \vec{y}) p(\pi|\vec{y})$$

The first term is trickier and we can't simply follow the procedure for the Bayesian linear regression model. The reason is that we are asked for a distribution on $\hat{y}$ conditioned on both π and $\hat{x}$ plus the class-specific variables. However, according to our model we have

$$y \sim \text{Bernoulli}(\pi), \qquad x|y \sim p(x|\theta_y)$$

The generative model for the data gives us a marginal distribution on y and a conditional distribution on x. Therefore, unlike before we can't just go to the model to find out what to plug in for $p(\hat{y}|\hat{x}, \theta_1, \theta_0, \pi)$. However, we notice that these distributions in combination with Bayes rule provide us with all the information we need.

- The distribution $p(\hat{y}|\hat{x}, \theta_1, \theta_0, \pi)$ can be interpreted as the *posterior* distribution of $\hat{y}$ given $\hat{x}$ and the model variables. By Bayes rule,

$$p(\hat{y}|\hat{x}, \theta_1, \theta_0, \pi) = \frac{p(\hat{x}|\theta_{\hat{y}})p(\hat{y}|\pi)}{p(\hat{x}|\theta_1)p(\hat{y}=1|\pi) + p(\hat{x}|\theta_0)p(\hat{y}=0|\pi)}$$

  Notice that because $\hat{y}$ is a discrete variable, the integral turns into a sum over the set of possible values $\hat{y}$ can take.

- We can swap $p(\hat{y}|\hat{x}, \theta_1, \theta_0, \pi)$ with the Bayes rule version of it in the integral to form the predictive distribution. We're left with

$$p(\hat{y}|\hat{x}, X, \vec{y}) = \int_{\Omega_\theta}\int_{\Omega_\theta}\int_0^1 \frac{p(\hat{x}|\theta_{\hat{y}})p(\hat{y}|\pi)p(\theta_1|X, \vec{y})p(\theta_0|X, \vec{y})p(\pi|\vec{y})}{p(\hat{x}|\theta_1)p(\hat{y}=1|\pi) + p(\hat{x}|\theta_0)p(\hat{y}=0|\pi)} d\pi d\theta_1 d\theta_0$$

  However, we now hit a dead end. Even though we have actual functions we can plug in everywhere, the problem now is that there is a sum in the denominator. For example, if we plug in the functions involving π we can see that the integral is not tractable for this variable and so we can't get an analytic function for the predictive distribution.

Second attempt

- Even though the first attempt didn't work, we still have other options. We first notice that $\hat{y} \in \{0, 1\}$, meaning we can only query the predictive distribution at these two values. Above we used Bayes rule to make progress toward the final calculation. Now we use it again, but on the marginal distribution itself:

$$\begin{aligned} p(\hat{y}|\hat{x}, X, \vec{y}) &= \frac{p(\hat{x}|\hat{y}, X, \vec{y})p(\hat{y}|X, \vec{y})}{p(\hat{x}|\hat{y}=1, X, \vec{y})p(\hat{y}=1|X, \vec{y}) + p(\hat{x}|\hat{y}=0, X, \vec{y})p(\hat{y}=0|X, \vec{y})} \\ &= \frac{p(\hat{x}|\hat{y}, X, \vec{y})p(\hat{y}|\vec{y})}{p(\hat{x}|\hat{y}=1, X, \vec{y})p(\hat{y}=1|\vec{y}) + p(\hat{x}|\hat{y}=0, X, \vec{y})p(\hat{y}=0|\vec{y})} \end{aligned}$$

- Again we need to know what to plug in for the likelihood $p(\hat{x}|\hat{y}, X, \vec{y})$ and prior $p(\hat{y}|\vec{y})$. The first thing we should think is that these are both marginal distributions and can be represented as an integral over model variables.

- Likelihood: First,

$$\begin{aligned} p(\hat{x}|\hat{y}, X, \vec{y}) &= \int_{\Omega_\theta}\int_{\Omega_\theta} p(\hat{x}|\hat{y}, \theta_1, \theta_0)p(\theta_1, \theta_0|X, \vec{y}) d\theta_1 d\theta_0 \\ &= \int_{\Omega_\theta} p(\hat{x}|\theta_{\hat{y}})p(\theta_{\hat{y}}|\{x_i : y_i = \hat{y}\}) d\theta_{\hat{y}} \end{aligned}$$

  Notice that π isn't involved because, according to the model, conditioned on $\hat{y}$, $\hat{x}$ is independent of π. Also, the second line shows how $\hat{y}$ really picks out either θ_1 or θ_0 and so we only need to

integrate one of them. This marginal distribution is problem-specific, but as is usual, distributions are often selected so that it can be solved. Then, we would solve this integral for $\hat{y} = 1$ and $\hat{y} = 0$ and use these two values in Bayes rule above. Since we haven't defined $p(x|\theta)$ or $p(\theta)$ we have to stop here.

- Prior: The other term we need is $p(\hat{y}|\vec{y})$, which also has a marginal representation according to our model,

$$p(\hat{y}|\vec{y}) = \int_0^1 p(\hat{y}|\pi)p(\pi|\vec{y})d\pi$$

We already know from the model definition that

$$p(\hat{y}|\pi) = \pi^{\hat{y}}(1-\pi)^{1-\hat{y}}$$

and from Bayes rule that the posterior distribution

$$p(\pi|\vec{y}) = \frac{\Gamma(a+b+N)}{\Gamma(a+\sum_i y_i - 1)\Gamma(b+N-\sum_i y_i)-1}\pi^{a+\sum_i y_i}(1-\pi)^{b+N-\sum_i y_i}$$

We can also solve this integral, which we can write as

$$\int_0^1 p(\hat{y}|\pi)p(\pi|\vec{y})d\pi \quad = \quad \frac{\Gamma(a+b+N)}{\Gamma(a+\sum_i y_i)\Gamma(b+N-\sum_i y_i)} \int_0^1 \pi^{\hat{y}+a+\sum_i y_i - 1}(1-\pi)^{1-\hat{y}+b+N-\sum_i y_i - 1}d\pi$$

To do this, multiply and divide by the value $\frac{\Gamma(1+a+b+N)}{\Gamma(\hat{y}+a+\sum_i y_i)\Gamma(1-\hat{y}+b+N-\sum_i y_i)}$. In the numerator, put this value inside the integral. In the denominator, put it outside. Then we can notice that the integral is over a beta distribution and equals 1, so the integral disappears.

The result is

$$p(\hat{y}|\vec{y}) = \frac{\Gamma(a+b+N)\Gamma(\hat{y}+a+\sum_i y_i)\Gamma(1-\hat{y}+b+N-\sum_i y_i)}{\Gamma(a+\sum_i y_i)\Gamma(b+N-\sum_i y_i)\Gamma(1+a+b+N)}$$

One final useful property comes into play that is worth memorizing: For the gamma function, $\Gamma(x) = (x-1)\Gamma(x-1)$. Using this property on the evaluation of $p(\hat{y}|\vec{y})$ at $\hat{y} = 1$ and $\hat{y} = 0$, we see that

$$p(\hat{y} = 1|\vec{y}) = \frac{a+\sum_i y_i}{a+b+N}, \qquad p(\hat{y} = 0|\vec{y}) = \frac{b+N-\sum_i y_i}{a+b+N}$$

We now notice that we now have all we need to calculate the predictive distribution $p(\hat{y}|\hat{x}, X, \vec{y})$.

- Naive Bayes extension: For the naive Bayes extension we only need to modify the likelihood term. Using the notation from the discussion on naive Bayes above, we want to calculate

$$\begin{aligned} p(\hat{x}|\hat{y}, X, \vec{y}) &= \int_{\Omega_\theta} \prod_{j=1}^{v} p(\hat{x}(j)|\theta_{\hat{y}}(j))p(\theta_{\hat{y}}(j)|\{x_i(j) : y_i = \hat{y}\})d\theta_{\hat{y}} \\ &= \prod_{j=1}^{v} \int p(\hat{x}(j)|\theta_{\hat{y}}(j))p(\theta_{\hat{y}}(j)|\{x_i(j) : y_i = \hat{y}\})d\theta_{\hat{y}}(j) \end{aligned}$$

In other words, we have v separate marginal distributions to calculate, and the overall marginal distribution is the product of each individual one. We've made the problem much easier for ourselves, which is why someone might opt to be "naive" when doing Bayesian classification.

EECS E6720 Bayesian Models for Machine Learning

Columbia University, Fall 2016

Lecture 3, 9/22/2016

Instructor: John Paisley

- Up to this point we've been discussing problems that are simple enough that we can calculate all posterior distributions in closed form. That is, we've assumed that given a model $X \sim p(X|\theta)$ and prior $\theta \sim p(\theta)$, we can analytically calculate the posterior using Bayes rule,

$$p(\theta|X) = \frac{p(X|\theta)p(\theta)}{\int p(X|\theta)p(\theta)d\theta}. \tag{1}$$

  Since we have defined the model and priors, the joint likelihood term in the numerator is always known. Therefore, we specifically have assumed that the normalizing constant

$$p(X) = \int p(X|\theta)p(\theta)d\theta \tag{2}$$

  is an integral we can analytically solve.

- In practice, this is the exception and not the rule for machine learning applications. Again, since we will always be able to write an analytic form for $p(X|\theta)p(\theta)$ because we define it, this is equivalent to saying that the integral in the denominator is usually intractable.

- Why should we care? We know that $p(\theta|X) = \frac{1}{Z}p(X,\theta)$ for some number Z. This just scales the joint likelihood function $p(X,\theta)$ up or down. Imagine we want to calcualte the expectation of some function over its posterior, $\mathbb{E}_{p(\theta|X)}[f(\theta)] = \int f(\theta)p(\theta|X)d\theta$. This is equal to $\frac{1}{Z}\int f(\theta)p(X,\theta)d\theta$. What if this is an integral we can calculate? The problem here is that Z can be expected to vary over such a large range that, without knowing Z we are no closer to knowing anything about $\mathbb{E}_{p(\theta|X)}[f(\theta)]$.

- In this lecture we will discuss two methods for approximating posterior distributions. The first method is called the Laplace approximation and is useful when the number of model variables is relatively small. The second method is called Gibbs sampling, which is an instance of a wide variety of posterior inference algorithms collectively referred to as Markov chain Monte Carlo (MCMC) methods. Gibbs sampling is much more widely used than the Laplace approximation and is still useful in the case of models that have very many variables.

Example: Logistic regression

- Logistic regression is one instance of linear regression applied to the classification problem. To recall the setup, we have data in the form of labeled pairs $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$ where $x \in \mathbb{R}^d$ and $y \in \{0, 1\}$. The goal is to use this information to help us predict the label of a new y given a new x.

- Model: As with linear regression, we assume that the output depends on the linear relationship between x and a weight vector $w \in \mathbb{R}^d$. This time the relationship is

$$y_i \overset{ind}{\sim} \text{Bernoulli}(\sigma(x_i^T w)), \qquad \sigma(x_i^T w) = \frac{\mathrm{e}^{x_i^T w}}{1 + \mathrm{e}^{x_i^T w}} \tag{3}$$

  Last lecture we saw another modeling approach to binary classification using a Bayes classifier. There is no "correct" choice of model, only more and less appropriate models for the problem at hand; all models make a simplifying approximation about the underlying function generating of the data—the first one being that such a function exists. To find out which model is "better," you can try both on your data and see which performs better in practice.

- Prior: We assume a prior distribution on w of the form

$$w \sim \text{Normal}(0, cI). \tag{4}$$

  Again, this is a choice that we make and a different prior can be defined, perhaps one that makes stronger assumptions about what types of vectors we should learn in the posterior. For example, a prior that is only defined on $\mathbb{R}_+^d$ would *require* that the posterior be defined on $\mathbb{R}_+^d$ as well (simply look at Bayes rule to see this). With the normal prior there is no such requirement.

- Posterior: The posterior distribution on w is found using Bayes rule

$$p(w|X, \vec{y}) = \frac{\prod_{i=1}^N p(y_i|x_i, w)p(w)}{\int \prod_{i=1}^N p(y_i|x_i, w)p(w)dw} \tag{5}$$

  When we plug in $p(y_i|x_i, w) = \left(\frac{\mathrm{e}^{x_i^T w}}{1+\mathrm{e}^{x_i^T w}}\right)^{y_i} \left(1 - \frac{\mathrm{e}^{x_i^T w}}{1+\mathrm{e}^{x_i^T w}}\right)^{1-y_i}$, we quickly find that the integral in the denominator can't be solved. Therefore, we can't get the posterior distribution. Furthermore, imagine that there existed an oracle that could magically gave us the value of the solution to this integral. We would still likely hit a roadblock later when trying to solve integrals, for example to calculate the predictive distribution. This is why many researchers like to keep things conjugate if they can. When they can't they often resort to approximations. We will next discuss one simple method to approximate this posterior distribution.

Laplace approximations

- Return to the generic model: $X \sim p(X|\theta)$ and $\theta \sim p(\theta)$. We make no assumptions about the forms of either of these distributions. Laplace approximations treat the posterior as being approximately Gaussian. That is, the goal of the Laplace approximation is to set

$$p(\theta|X) \approx \text{Normal}(\mu, \Sigma) \tag{6}$$

  for some good setting of μ and Σ. There are different ways one might find these parameter values; the Laplace approximation is one approach.

- First, it turns out that a normal approximation to the posterior can arise automatically from a specific approximation to the joint likelihood $p(X, \theta) = p(X|\theta)p(\theta)$. That is, we will approximate the numerator and denominator in Bayes rule, after which the normal distribution will pop out.

- The first step is to write Bayes rule slightly differently

$$p(\theta|X) = \frac{\mathrm{e}^{\ln p(X,\theta)}}{\int \mathrm{e}^{\ln p(X,\theta)} d\theta} \tag{7}$$

- Next, rather than focus on $p(X, \theta)$, we focus on $\ln p(X, \theta)$. Notice that in more abstract terms we can just think of this as a function of θ. The interpretation of this function as the log of the joint likelihood is simply added later. We want an approximation of $\ln p(X, \theta)$ that is a good one, but also general enough that we don't need to worry about the fact that we don't actually know what this function is yet since we haven't defined anything other than abstract probability distributions.

- The Laplace approximation uses a 2nd order Taylor expansion as an approximation. Recall that we can approximate a function $f(\theta)$ with its 2nd order Taylor expansion as follows:

$$f(\theta) \approx f(\theta_0) + (\theta - \theta_0)^T \nabla f(\theta_0) + \frac{1}{2}(\theta - \theta_0)^T \nabla^2 f(\theta_0)(\theta - \theta_0) \tag{8}$$

  The point θ_0 is any arbitrarily chosen point in the domain of f (i.e., chosen among values that θ is allowed to take). The shorthand $\nabla f(\theta_0)$ means we evaluate the gradient of $f(\theta_0)$ at θ_0. For our problem $f(\theta) = \ln p(X, \theta)$.

- The point θ_0 is critical since the 2nd order Taylor approximation is very accurate around this point, and much less accurate as we move away from it. Therefore, we must decide a good setting for θ_0.

- For the Laplace approximation we are able to find a good point, and also a convenient one. Specifically, we set

$$\theta_0 = \theta_{\text{MAP}} = \arg\max_{\theta} \; p(\theta|X) = \arg\max_{\theta} \; \frac{p(X|\theta)p(\theta)}{p(X)} = \arg\max_{\theta} \; \ln p(X, \theta) \tag{9}$$

  This is called the *maximum a posteriori* solution. Notice that it is the point that is the most probable (or, has the highest density) according to the posterior distribution. However, the whole reason we're on this endeavor is that we don't know the posteriori. Fortunately the maximum point of $p(\theta|X)$ is the same as for $p(X, \theta)$; the constant $p(X)$ doesn't depend on θ so it just scales the function and doesn't change the point at which it's maximized. Also notice that the $\ln$ function doesn't change the location of the maximum, which is all that we're interested in.

- Aside about MAP inference: MAP inference is the process of finding, for example, θ_{MAP} in the problem above. It first converts the joint likelihood to the log joint likelihood, $\ln p(X, \theta)$, making the observation that the monotonicity of the log transformation preserves all local maximum of $p(X, \theta)$. It then initializes θ and iteratively modifies θ to maximize $\ln p(X, \theta)$. One pervasive approach to this are the gradient ascent methods. For example, with steepest ascent, we iteratively update $\theta \leftarrow \theta + \rho \nabla_\theta \ln p(X, \theta)$ until it reaches a local maximum of $\ln p(X, \theta)$. $\rho \in [0, 1]$ scales the gradient and is usually small.

- And so for the Laplace approximation, the first step is to run an iterative algorithm to find θ_{MAP}. Then, using this as θ_0 in the 2nd order Taylor expansion we notice a nice simplification occurs,

$$\ln p(X,\theta) \approx \ln p(X,\theta_{\text{MAP}}) + \underbrace{(\theta-\theta_{\text{MAP}})^T \nabla \ln p(X,\theta_{\text{MAP}})}_{=0} + \frac{1}{2}(\theta-\theta_{\text{MAP}})^T \nabla^2 \ln p(X,\theta_{\text{MAP}})(\theta-\theta_{\text{MAP}}) \tag{10}$$

The first order term cancels out because the gradient of $\ln p(X,\theta)$ equals zero at the location of its maximum point, which is precisely what we get when we run the MAP algorithm to find θ_{MAP}.

- Directly plugging in this approximation,

$$p(\theta|X) \approx \frac{p(X,\theta_{\text{MAP}}) e^{-\frac{1}{2}(\theta-\theta_{\text{MAP}})^T(-\nabla^2 \ln p(X,\theta_{\text{MAP}}))(\theta-\theta_{\text{MAP}})}}{\int p(X,\theta_{\text{MAP}}) e^{-\frac{1}{2}(\theta-\theta_{\text{MAP}})^T(-\nabla^2 \ln p(X,\theta_{\text{MAP}}))(\theta-\theta_{\text{MAP}})} d\theta} \tag{11}$$

The terms $p(X,\theta_{\text{MAP}})$ cancel out and we are left with something we can immediately recognize as a multivariate normal distribution. In particular, we see that

$$p(\theta|X) \approx \text{Normal}(\theta_{\text{MAP}}, (-\nabla^2 \ln p(X,\theta_{\text{MAP}}))^{-1}) \tag{12}$$

We set the mean of our posterior approximation to the MAP solution θ_{MAP} and the covariance to the inverse of the negative of the Hessian of $\ln p(X,\theta)$ evaluated at θ_{MAP}. Therefore, after running the MAP algorithm, we calculate the matrix of second derivatives by hand and plug $\theta = \theta_{\text{MAP}}$ into the resulting matrix-organized set of functions of θ. We use this matrix to approximate the covariance. Notice that, from a computational perspective, the measure of uncertainty that we get on top of MAP from an approximate posterior distribution comes almost for free when $\dim(\theta)$ is small.

Logistic regression revisited

- We next derive the Laplace approximation for the logistic regression problem. We're going to approximate $p(w|\vec{y},X) \approx N(\mu,\Sigma)$. We first write the joint likelihood,

$$p(\vec{y},w|X) \propto \prod_{i=1}^{N} \sigma(x_i^T w)^{y_i}(1-\sigma(x_i^T w))^{1-y_i} e^{-\frac{1}{2c}w^T w}, \qquad \sigma(x_i^T w) = \frac{e^{x_i^T w}}{1+e^{x_i^T w}} \tag{13}$$

We initialize $w^{(0)} = 0$ and update $w^{(t)} = w^{(t-1)} + \rho \nabla_w \ln p(\vec{y},w|X)|_{w^{(t-1)}}$ using gradient ascent. The derivative is directly calculated to be

$$\nabla_w \ln p(\vec{y},w|X) = \sum_{i=1}^{N}(y_i - \sigma(x_i^T w))x_i - \frac{1}{c}w \tag{14}$$

We simply plug the most recent value for w into this function to get the direction in which to step. When the algorithm converges to a point "close enough" to the maximum value, found by monitoring how much $\ln p(\vec{y},w|X)$ is increasing with each step, we take the final value of $w^{(T)}$ at this iteration (call it T) and set w_{MAP} to be this vector and therefore the mean of the approximate posterior distribution of w. So $\mu = w_{\text{MAP}} \equiv w^{(T)}$.

To find the covariance, we directly calculate that

$$-\Sigma^{-1} = \nabla_w^2 \ln p(\vec{y},w|X) = -\sum_{i=1}^{N} \sigma(x_i^T w)(1-\sigma(x_i^T w)) x_i x_i^T - \frac{1}{c} I \tag{15}$$

We plug the value of w_{MAP} into this function and invert the negative of it to obtain the the covariance matrix of the normal distribution used to approximate $p(w|\vec{y}, X)$.

Another modeling example

- Let's look at another example of a model where we can't calculate the posterior analytically, but for which the Laplace approximation isn't a feasible option.

- Setup: We are given a set of measurements $\mathcal{D} = \{r_{ij}\}$ where the index pair $(i, j) \in \Omega$. That is, we let $i = 1, \ldots, N$ and $j = 1, \ldots, M$, but not every (i, j) is measured (i.e., in Ω). For example, r_{ij} could be the rating given by user i to object j (think Netflix, Yelp, Amazon, etc.), in which case $|\Omega| \ll NM$. We want to come up with a model to predict those $r_{ij} \notin \mathcal{D}$.

- Model: A typical model for this uses matrix factorization (we will not directly use the matrix notation here) and a low rank assumption. Here, we assume each user i is represented by a vector $u_i \in \mathbb{R}^d$ and each object j by a vector $v_j \in \mathbb{R}^d$. The low rank assumption comes from $d \ll \min(N, M)$. Then we let

$$r_{ij} \sim \text{Normal}(u_i^T v_j, \sigma^2) \qquad \text{for all } (i, j) \in \Omega \tag{16}$$

  Comment: This model does not seem ideal for all cases since it assumes $r_{ij} \in \mathbb{R}$. For example, in the context of user/object ratings alarm bells should be going off because we are modeling a finite set of positive integers as continuous. In this case where $r_{ij} \notin \mathbb{R}$ we are making a simplifying modeling assumption. This won't lead to any bugs during inference time, and the justification for it is in the performance. Still, we will later see how a few simple modifications can make us more comfortable with the modeling assumptions while still being very close to this model.

- Priors: We will assume that $u_i \overset{iid}{\sim} \text{Normal}(0, cI)$ and $v_j \overset{iid}{\sim} \text{Normal}(0, cI)$.

- Posterior: Let U contain all u_i, V contain all v_i and R contain all measured r_{ij}. Using Bayes rule, we want to calculate

$$p(U, V|R) = \frac{p(R|U, V)p(U, V)}{\int \cdots \int p(R|U, V)p(U, V)du_1 \cdots du_N dv_1 \cdots dv_M} \tag{17}$$

  Because of the definitions we made in the model and the priors, conditional independence lets us write $p(R|U, V) = \prod_{(i,j)\in\Omega} p(r_{ij}|u_i, v_j)$ and $p(U, V) = \prod_i p(u_i) \prod_j p(v_j)$.

- The problem is that the denominator is intractable and so we can't solve Bayes rule. Just like before we hit a roadblock. However, unlike before it seems that Laplace can't help us here. Even though we could find a local optimal solution for U_{MAP} and V_{MAP} to approximate a large, $Nd+Md$ length mean vector, the covariance matrix obtained from the Hessian would be $(Nd + Md) \times (Nd + Md)$. This can easily exceed one billion values to estimate!

- Step back: Let's take a step back and talk about what our objectives typically are with Bayes rule in contexts like these. Let $p(U, V|R)$ be the posterior distribution on the model variables. What we really want is the predictive distribution $p(r_{ij}|R)$ for some $(i, j) \notin \Omega$. Or, at least, we would like to calculate the expectation $\mathbb{E}[r_{ij}|R]$ under this posterior where we integrate out all of our uncertainty in the model variables. For now let's just focus on the expectation.

- We develop this further in the context of the above problem to make it concrete, but observe that this is a very general problem in Bayesian modeling. Using a notational shorthand for this problem, the expectation of a new r_{ij} under the predictive distribution is

$$\begin{aligned}
\mathbb{E}[r_{ij}|R] &= \int r_{ij} p(r_{ij}|R) dr_{ij} && (18)\\
&= \int r_{ij} \int\int p(r_{ij}|U,V) p(U,V|R) dU dV dr_{ij} && (19)\\
&= \int\int \underbrace{\left[\int r_{ij} p(r_{ij}|U,V) dr_{ij}\right]}_{= u_i^T v_j} p(U,V|R) dU dV && (20)
\end{aligned}$$

  Notice that we've swapped integrals above. Since $p(r_{ij}|U,V) = \text{Normal}(u_i^T v_j, \sigma^2)$, the bracketed term is equal to $u_i^T v_j$. Therefore, we ultimately want $\mathbb{E}[u_i^T v_j|R] = \int\int (u_i^T v_j) p(U,V|R) dU dV$ where the expectation is under the posterior distribution $p(U,V|R)$. Since we don't know this distribution, we are forced to seek other options. As we will see, one of these options employs Monte Carlo integration.

- Monte Carlo integration: MC integration is a very general technique. Imagine we want to calculate $\mathbb{E}_{p(X)}[f(x)]$ but run into a mathematical problem in doing so. We can approximate this expectation by first sampling $X_1, \ldots, X_N \overset{iid}{\sim} p(X)$ and then approximating

$$\mathbb{E}_{p(X)}[f(X)] \approx \frac{1}{N} \sum_{n=1}^{N} f(X_n) \tag{21}$$

  As $N \to \infty$ this approximation converges to the true expectation.

- So if we could find a way to get i.i.d. samples

$$(U^{(1)}, V^{(1)}), \ldots, (U^{(N)}, V^{(N)}) \overset{iid}{\sim} p(U,V|R) \tag{22}$$

  we could say

$$\mathbb{E}[u_i^T v_j|R] \approx \frac{1}{N} \sum_{n=1}^{N} \langle u_i^{(n)}, v_j^{(n)} \rangle \tag{23}$$

- In fact, having access to these samples would allow us to calculate much more than an approximation to $\mathbb{E}[r_{ij}|R]$. For example, if we wanted a measure of confidence in this prediction, we could calculate the variance under the predictive distribution, $Var(r_{ij}|R) = \mathbb{E}[r_{ij}^2|R] - \mathbb{E}[r_{ij}|R]^2$. Using similar reasoning as above, we would find that

$$Var(r_{ij}|R) \approx \sigma^2 + \frac{1}{N} \sum_{n=1}^{N} \left(\langle u_i^{(n)}, v_j^{(n)} \rangle\right)^2 - \left(\frac{1}{N} \sum_{n=1}^{N} \langle u_i^{(n)}, v_j^{(n)} \rangle\right)^2 \tag{24}$$

  Many other calculations of interest under the predictive distribution can also be obtained this way. Therefore, a lot can be done if we can somehow get access to i.i.d. samples from the posterior distribution on the model $p(U,V|R)$.

- A major procedure for (approximately) getting these samples is called *Markov chain Monte Carlo (MCMC)* sampling. We'll next discuss one instance of this general technique, and arguably the easiest, called *Gibbs sampling*.

Gibbs sampling

- We will present the *recipe* for Gibbs sampling. Unfortunately a discussion on the *correctness* of the recipe is beyond the scope of this class. Every statistics department will have courses that focus heavily on MCMC methods, and to truly appreciate MCMC and it's theoretical correctness, it's worthwhile to take one of those classes. This is in contrast to *variational* inference methods, which are generally not taught in statistics departments and will be discussed more fully later in this course. Variational inference is a development of machine learning research.

- To use simpler and more general notation, imagine we have a model with two variables, θ_1 and θ_2. We have data from this model, $X \sim p(X|\theta_1, \theta_2)$ and independent priors $p(\theta_1)$ and $p(\theta_2)$. We find that we can't calculate $p(\theta_1, \theta_2|X)$ because the normalizing constant is an intractable integral. This is the typical story thus far.

- However, Gibbs sampling makes the added assumption that we *can* calculate $p(\theta_1|X, \theta_2)$ and $p(\theta_2|X, \theta_1)$ using Bayes rule. In other words, if we had access to all of the model variables except for one of them, then we could calculate the posterior distribution of this one unknown variable given the data and all other variables.

- Gibbs sampling uses this property of the tractability of all *conditional* posterior distributions to get samples from the unknown *full* posterior distribution of all model variables. Below, we write the Gibbs sampling procedure for this generic model.

  Gibbs sampling algorithm for the toy problem

  1. Randomly initialize $\theta_1^{(0)}$ and sample $\theta_2^{(0)} \sim p(\theta_2|X, \theta_1^{(0)})$
  2. For step $t = 1, \ldots, T$
     (a) Sample $\theta_1^{(t)} \sim p(\theta_1|X, \theta_2^{(t-1)})$
     (b) Sample $\theta_2^{(t)} \sim p(\theta_2|X, \theta_1^{(t)})$

- In other words, we first initialize all the variables in some way. Step 1 above is one possible initialization method. Then, we iterate through each variable and update it by sampling from its posterior distribution conditioned on the *current values* of everything els. MCMC theory proves that we get (approximate) samples from $p(\theta_1, \theta_2|X)$ this way, and Gibbs sampling MCMC requires us to be able to calculate the conditional posterior distributions.

- Discussion: First, notice that the conditional posterior uses the *most recent* values for all variables. Therefore, when sampling a particular θ_i during iteration t, some variables we condition on will use their value at iteration $t-1$ (because they haven't been updated yet during iteration t), while others will use their new value at iteration t (because they were sampled before sampling θ_i during the current iteration). What Gibbs sampling does *not* do is partition the variables into two sets and sample the new set at step t conditioned on the old set from $t-1$.

- To give some MCMC background and how it is run in practice (in the context of the toy model):

  1. MCMC is a special case of a Markov chain where the stationary distribution is the desired full posterior distribution. Therefore, we are only guaranteed to get *one* sample from the full posterior in the infinite limit. (It is necessary to take a class devoted to these methods to appreciate this fully.)

  2. In practice, one first runs MCMC (here, the Gibbs sampler) for a large number of steps, let that number be T_1, and then approximates the first sample from the posterior distribution as $(\theta_1^{(T_1)}, \theta_2^{(T_1)})$. The first $T_1 - 1$ steps are called the "burn-in" phase and is intended to minimize the impact of the initialization, which likely does not start the chain off with a good setting for the model variables. The first $T_1 - 1$ iterations "fix" this bad initialization.

  3. Next we want to get a second sample from the posterior. However, remember that we need the samples to be i.i.d. for Monte Carlo integration. Clearly any two adjacent samples $(\theta_1^{(t)}, \theta_2^{(t)})$ and $(\theta_1^{(t+1)}, \theta_2^{(t+1)})$ are not independent of each other, since the values at t were used to get the values at $t+1$. Therefore, just like with the burn-in phase, we run the chain for several more iterations until step T_2 so that $(\theta_1^{(T_1)}, \theta_2^{(T_1)})$ and $(\theta_1^{(T_2)}, \theta_2^{(T_2)})$ are almost totally uncorrelated. This continues again until step T_3, then T_4, etc. Usually, $T_i - T_{i-1}$ for $i > 1$ is picked to be a constant number much less than T_1.

  4. Good numbers for T_1 and $T_i - T_{i-1}$ are not immediately obvious. MCMC textbooks give useful heuristics. In machine learning applications these are usually picked arbitrarily in practice and are usually determined more by the desired number of samples within a pre-allotted running time than by the statistical properties of the Markov chain.

  5. We then make the approximation that

  $$(\theta_1^{(T_1)}, \theta_2^{(T_1)}), (\theta_1^{(T_2)}, \theta_2^{(T_2)}), \ldots, (\theta_1^{(T_S)}, \theta_2^{(T_S)}) \overset{iid}{\sim} p(\theta_1, \theta_2|X) \tag{25}$$

  Notice that we're simply throwing away the updates for many of the iterations. By writing out the integral form of the expectation below, we see we can use these samples for Monte Carlo integration to approximate

  $$\mathbb{E}[f(\hat{X})|X] \approx \frac{1}{S}\sum_{s=1}^{S} \mathbb{E}[f(\hat{X})|\theta_1^{(T_s)}, \theta_2^{(T_s)}] \tag{26}$$

  To give the full derivation (letting $\Theta = (\theta_1, \theta_2)$:

  $$\begin{aligned}\mathbb{E}[f(\hat{X})|X] &= \int f(\hat{X})p(\hat{X}|X)d\hat{X} = \int f(\hat{X}) \int p(\hat{X}|\Theta)p(\Theta|X)d\Theta d\hat{X} \\ &\approx \int f(\hat{X})\frac{1}{S}\sum_{s=1}^{S} p(\hat{X}|\Theta^{(s)})d\hat{X} = \frac{1}{S}\sum_{s=1}^{S}\int f(\hat{X})p(\hat{X}|\Theta^{(s)})d\hat{X}\end{aligned}$$

  This last term equals $\frac{1}{S}\sum_{s=1}^{S}\mathbb{E}[f(\hat{X})|\theta_1^{(s)}, \theta_2^{(s)}]$ and we just re-index the samples.

  6. The reason that MCMC, despite its precise theory, is still only an approximate method is the fact that T_1 and $T_i - T_{i-1}$ are finite. Therefore, at step T_1 we are *not* sampling from the stationary distribution of the Markov chain (remember that the "stationary distribution" is the desired posterior distribution), and the samples at steps T_i and T_{i-1} are *not* 100% uncorrelated with each other, and therefore not independent. Nevertheless, MCMC often works extremely well in practice.

7. Much of the development of MCMC techniques not covered in this class—in addition to accounting for models in which we don't have all conditional posteriors—are on ways to sample such that the values of T_1 and $T_i - T_{i-1}$ can be reduced without sacrificing the quality of the samples (i.e., their approximate independence and closeness to the posterior distribution).

- Returning to the original model we were discussing, we want $p(U, V|R)$ where U is the set of N "user" vectors and V the set of M "object" vectors, and R is the set of measured values $\{r_{ij} : (i,j) \in \Omega\}$ often corresponding to a rating given by user i to object j.

- We can't find this posterior exactly. A next step is to check if we can calculate all the *conditional* posteriors exactly. If so, we can approximately sample from $p(U, V|R)$ using Gibbs sampling.

- We will check for one u_i. Clearly this then establishes the case for all u_i, and by symmetry we can perform the same exact derivation for all v_j. The conditional posterior of u_i is

$$p(u_i|V, R) \propto \prod_{j:(i,j)\in\Omega} p(r_{ij}|u_i, v_j)p(u_i) \tag{27}$$

  1. Notice with this that we decided to write the likelihood in terms of r_{ij} only and not v_j.
  2. Also, notice that on the LHS we condition on all V and R.
  3. On the RHS we use the structure of the model we defined: We only need to consider the r_{ij} for a fixed i and over those j such that $r_{ij} \in \mathcal{D}$. Also, the likelihood of these r_{ij} are independent given u_i and V. For a particular r_{ij}, conditioning on all V is fine to write out in the abstract, but when we actually write out the distribution we will see that, according to our model assumption, r_{ij} only depends on u_i and v_j, and no other u or v.
  4. Therefore, $p(r_{ij}|u_i, V) = p(r_{ij}|u_i, v_j)$. Conditioning on more than is necessary is just superfluous and makes our parsing of the distributions and the dependencies they induce among the model variables less transparent—it is good practice to only condition on what is necessary according to the model and prior definitions.

- Using the likelihood and prior defined for this problem, we have

$$p(u_i|V, R) \propto \prod_{j:(i,j)\in\Omega} \text{Normal}(r_{ij}|u_i^T v_j, \sigma^2)\text{Normal}(u_i|0, cI) \tag{28}$$

  We have actually already calculated this last lecture for the Bayesian linear regression problem. In fact, from the perspective of u_i it's identically the same problem! From that calculation, we know that

$$p(u_i|V, R) = \text{Normal}(\mu_{u_i}, \Sigma_{u_i}) \tag{29}$$

$$\Sigma_{u_i} = \left(\frac{1}{c}I + \frac{1}{\sigma^2}\sum_{j:(i,j)\in\Omega} v_j v_j^T\right)^{-1} \qquad \mu_{u_i} = \Sigma_{u_i}\left(\frac{1}{\sigma^2}\sum_{j:(i,j)\in\Omega} r_{ij}v_j\right)$$

- By symmetry, we get a similar posterior distribution for each v_j. The resulting Gibbs sampling algorithm for this model is the following.

Gibbs sampling algorithm for the matrix factorization model

1. Randomly initialize each $u_i^{(0)}$ and $v_j^{(0)}$
2. For step $t = 1, \ldots, T$
   (a) For $i = 1, \ldots, N$ sample a new u_i from its conditional posterior distribution,

$$u_i^{(t)} \sim \text{Normal}(\mu_{u_i}^{(t)}, \Sigma_{u_i}^{(t)})$$

   where at iteration t we use the parameters

$$\Sigma_{u_i}^{(t)} = \left(\frac{1}{c} I + \frac{1}{\sigma^2} \sum_{j:(i,j)\in\Omega} v_j^{(t-1)} (v_j^{(t-1)})^T \right)^{-1} \qquad \mu_{u_i}^{(t)} = \Sigma_{u_i}^{(t)} \left(\frac{1}{\sigma^2} \sum_{j:(i,j)\in\Omega} r_{ij} v_j^{(t-1)} \right)$$

   (b) For $j = 1, \ldots, M$ sample a new v_j from its conditional posterior distribution,

$$v_j^{(t)} \sim \text{Normal}(\mu_{v_j}^{(t)}, \Sigma_{v_j}^{(t)})$$

   where at iteration t we use the parameters

$$\Sigma_{v_j}^{(t)} = \left(\frac{1}{c} I + \frac{1}{\sigma^2} \sum_{i:(i,j)\in\Omega} u_i^{(t)} (u_i^{(t)})^T \right)^{-1} \qquad \mu_{v_j}^{(t)} = \Sigma_{v_j}^{(t)} \left(\frac{1}{\sigma^2} \sum_{i:(i,j)\in\Omega} r_{ij} u_i^{(t)} \right)$$

- Notice that, because we choose to update all u_i first in each iteration, their conditional posteriors use the values of v_j in the *previous* iteration—those are the most recent values for v_j. When we then update all v_j, we use the values of u_i in the *current* iteration because these are the most recent values. This might give the impression that the order in which we update each variable will make a difference. However, *order does not matter*. We could have chosen to update all v_j first during each iteration. Or we could alternate between u_i and v_j in an arbitrary way. All that matters is that, when calculating the conditional posterior, we condition on the most recent values for all variables.

EECS E6720 Bayesian Models for Machine Learning

Columbia University, Fall 2016

Lecture 4, 10/6/2016

Instructor: John Paisley

- So far, we've been talking about posterior distributions where:

    1. The prior is conjugate to the likelihood. In this case we can calculate the posterior distribution exactly by giving the distribution name and its parameters.

    2. We can make a simple approximation using the Laplace's method. In this case, we approximate the posterior of the model variables with a multivariate Gaussian and use the MAP solution and the Hessian of the log joint likelihood evaluated at the MAP solution.

    3. We can calculate all conditional posterior distributions of the model variables and approximately sample i.i.d. from the posterior distribution using MCMC Gibbs sampling.

- In machine learning applications, the first approach is relatively rare since the model's are usually too complex. The second approach is also not very common since there are usually too many variables to estimate a full covariance matrix for them. The third approach *is* commonly used, but it suffers from poor scalability to large data sets, which is what machine learning problems typically encounter. A recent focus of research interest is on fixing this problem.

- There is a fourth major technique used widely in the machine learning community called *variational inference*. Unlike MCMC methods, which sample new values of the model variables in each iteration, variational methods set up an objective function (not unlike MAP) and optimize this objective function over parameters. As we will see in future lectures, these parameters correspond to parameters of distributions over the model variables. Variational methods try to find parameters to these distributions such that the result is a close approximation to the desired, but intractable posterior distribution.

- This will be made more precise in the next few lectures. However, to lay the groundwork for understanding variational inference (VI) it is necessary to go over the *Expectation-Maximization* (EM) algorithm in its full generality. The goal will be to present VI as a very simple but interesting modification of EM.

- Therefore, for the rest of this lecture we will return to the world of learning *point estimates* of model variables, instead of trying to learn their *posterior distributions*.

Maximum a posteriori (MAP) inference

- Recall the general modeling setup:

$$\begin{aligned} \text{Model:} \quad & X \sim p(X|\theta) \\ \text{Prior:} \quad & \theta \sim p(\theta) \end{aligned}$$

- We assume that we can't calculate the posterior $p(\theta|X)$ exactly, so instead we use MAP inference to find the value of θ that is a maximum of $p(\theta|X)$.

$$\theta_{\text{MAP}} = \arg\max_{\theta} \ \ln p(X, \theta) \tag{1}$$

(When $\ln p(X, \theta)$ is non-convex, we can usually only hope to find a local maximum. This is typically the case.)

- MAP inference is an optimization problem with a Bayesian interpretation of the objective function and what is being optimized.
  - Ultimately, we want to find the point θ that maximizes, or at least locally maximizes, the objective function $\ln p(X, \theta)$.
  - Equivalently, we want to search for a θ such that $\nabla_\theta \ln p(X, \theta) = 0$, since the derivative at any local maximum will equal zero. We also need to find this point in a way that assures us we haven't found a local *minimum*, since the gradient here will be zero as well.
  - Sometimes we can do this in closed form; we can take the derivative and solve for θ. In this case the problem is solved and no iterative optimization algorithm is necessary.
  - Often we can't find a solution for θ to the equation $\nabla_\theta \ln p(X, \theta) = 0$. In this case, the typical approach is to use gradient methods, where we iteratively update θ according to the rule

$$\theta' \ \leftarrow \ \theta + \rho B \nabla_\theta \ln p(X, \theta). \tag{2}$$

    The value of $\rho > 0$ is a step size and the matrix B is positive semi-definite. When $B = I$, the identity matrix, this is steepest ascent. When B is the inverse of the negative Hessian of $\ln p(X, \theta)$, this is Newton's method.
  - What we should set ρ and B to are not solved problems. This is part of the "art" of optimization and there are several ways of figuring out what these should be, and they may change with each update of θ to take into account properties of the objective function $\ln p(X, \theta)$ at the current point.
  - In a sense, we want to have closed-form solutions. We want to say that, for such-and-such variable, the best setting for it is equal to something we can write out exactly, rather than something we stumble upon after being told which way to move by 50 different gradients (the number 50 is just for example).
  - The EM algorithm is a very clever way to do this. Before discussing EM in general, let's first look at one model for which we might give up hope if we wanted to do MAP by directly doing gradient ascent on the log joint likelihood as described above.

Probit regression

- Setup: The setup is the same as for logistic regression: We have a data set, $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$ where $x \in \mathbb{R}^d$ is a feature vector and $y \in \{0, 1\}$ is its class label. We want to use this information to help predict the value of y for a new x.

- Model and prior: The probit model is similar to the logistic regression model. We assume that the labels are Bernoulli random variables that depend on the dot product of the corresponding features with a weight vector, which we assume has a Gaussian prior,

$$y_i \sim \text{Bernoulli}(\Phi(x_i^T w/\sigma)), \qquad w \sim \text{Normal}(0, \lambda^{-1} I) \tag{3}$$

  The value $\sigma > 0$ is a parameter we can change.

  As with the logistic regression model, the probit regression model is a discriminative classifier because it conditions on x throughout instead of modeling it with a distribution. However, where logistic regression uses the sigmoid function to map $x_i^T w$ to a number between 0 and 1, probit regression uses the cumulative distribution function (CDF) of a standard normal distribution,

$$\Phi(x_i^T w/\sigma) = \int_{-\infty}^{x_i^T w/\sigma} (2\pi)^{-\frac{1}{2}} \mathrm{e}^{-\frac{1}{2}s^2} ds \tag{4}$$

  Notice that, as with the sigmoid function discussed in an earlier lecture, as $x_i^T w$ increases to $+\infty$, the function $\Phi(x_i^T w/\sigma)$ increases to 1. As $x_i^T w$ decreases to $-\infty$, the function $\Phi(x_i^T w/\sigma)$ decreases to 0. (In fact, it should be clear that any function that has the properties of a CDF is a candidate for plugging into this Bernoulli distribution.)

- MAP inference: Ideally, we would do posterior inference for w. However, applying Bayes rule we quickly find that we run into the same problem as for logistic regression: The normalizing constant is not a tractable integral. A next step would be to learn the MAP solution for w,

$$\begin{aligned} w_{\text{MAP}} &= \arg\max_w \ \ln p(\vec{y}, w|X) \qquad (5) \\ &= \arg\max_w \ \ln p(w) + \sum_{i=1}^N \ln p(y_i|w, x_i) \\ &= \arg\max_w \ -\frac{\lambda}{2} w^T w + \sum_{i=1}^N y_i \ln \Phi(x_i^T w/\sigma) + (1 - y_i) \ln(1 - \Phi(x_i^T w/\sigma)) \end{aligned}$$

  This requires us to calculate $\nabla_w \ln p(\vec{y}, w|X)$. One of the terms in this gradient is

$$\nabla_w \ln \Phi(x_i^T w/\sigma) = \nabla_w \ln \int_{-\infty}^{x_i^T w/\sigma} (2\pi)^{-\frac{1}{2}} \mathrm{e}^{-\frac{1}{2}s^2} ds \tag{6}$$

  Taking the full gradient of $\ln p(\vec{y}, w|X)$, we can see that, not only would we not be able to solve for w, but it's not even clear what the analytic solution to $\nabla_w \ln \Phi(x_i^T w/\sigma)$ is! Therefore, we can't even get a closed form vector representation for $\ln p(\vec{y}, w|X)$ that tells us which direction to move in to update w.

- So the question is, is probit regression even harder to work with that logistic regression? Is it totally hopeless? We will see that the EM algorithm can be used to optimize $\ln p(\vec{y}, w|X)$ in a clever way that involves closed form updates and results in an algorithm not more complicated (and arguably cleaner and nicer) than gradient ascent for logistic regression.

Setting up the expectation-maximization (EM) algorithm

- In the following discussion of the EM algorithm, we return to the general model setting, where

$$X \sim p(X|\theta), \qquad \theta \sim p(\theta).$$

  Again, the goal is to do MAP inference, where we maximize $\ln p(X, \theta)$ over θ. We will then derive an EM algorithm for probit regression to make it more specific.

- EM starts with an assumption: Assume there is some other variable ϕ that we can *introduce* to the model so that the marginal distribution is unchanged. That is

$$p(X, \theta) = \int p(X, \theta, \phi) d\phi$$

  - Note that we can always manage to do this. The purpose for EM will be to pick a variable ϕ that is useful.
  - In the next lecture we will see that, when coming at it from the other direction, this comes directly out of the model itself. That is, in this lecture we are starting with $p(X, \theta)$ and finding that it's problematic, leading us down this EM path. Often we *start* with a more complex model definition $p(X, \theta, \phi)$ and then decide to optimize the marginal distribution of this $p(X, \theta)$. In this case we know ϕ because we defined it in the larger model.
  - Again, one case starts with $p(X, \theta)$ and expands it (the hard case), and one case starts with $p(X, \theta, \phi)$ and collapses it (the easy case). Traditionally an introduction of EM starts with the hard case.

- The next steps are all directed towards finding an equivalent representation for $\ln p(X, \theta)$ that involves $\ln p(X, \theta, \phi)$. The reason why this new representation should help us in any way to optimize $\ln p(X, \theta)$ more easily will come later. For now, we are only manipulating equations.

- The first step is to use a basic rule of probability

$$p(X, \theta) p(\phi|X, \theta) = p(X, \theta, \phi) \tag{7}$$

  and taking logarithms and reorganizing, we therefore have

$$\ln p(X, \theta) = \ln p(X, \theta, \phi) - \ln p(\phi|X, \theta) \tag{8}$$

  Notice that the LHS of this equation is the original objective function we want to maximize. We can now interpret this as the marginal log joint likelihood of the extended log joint likelihood on the RHS. Also subtracted on the RHS is the posterior distribution of ϕ. In practice we will find this with Bayes rule. From this we can guess (and be correct) that if $p(\phi|X, \theta)$ is not solvable, the EM algorithm we're heading towards isn't going to make optimizing $\ln p(X, \theta)$ easier.

- The next step toward the EM "master equation" is to introduce a probability distribution $q(\phi)$ such that q is defined on the same support as ϕ. That is, if $\phi \in \mathbb{R}$, then q is defined on $\mathbb{R}$. If $\phi \in \{1, 2, 3, 4, 5\}$ then q is a 5-dimensional probability vector on those integers. For now, this is the only requirement we make on $q(\phi)$. Later, EM will tell us what we should set $q(\phi)$ to.

- We then use $q(\phi)$ in the following sequence of equalities:

$$q(\phi) \ln p(X, \theta) = q(\phi) \ln p(X, \theta, \phi) - q(\phi) \ln p(\phi | X, \theta) \tag{9}$$

Comment: We simply multiply the LHS and RHS by the same thing. It doesn't matter that this thing is a function. For each value of ϕ the equality holds.

$$\int q(\phi) \ln p(X, \theta) d\phi = \int q(\phi) \ln p(X, \theta, \phi) d\phi - \int q(\phi) \ln p(\phi | X, \theta) d\phi \tag{10}$$

Comment: An integral can be thought of as an infinitesimal summation. For each infinitesimal bit we add on the LHS, we add an infinitesimal bit on the RHS that equals it.

$$\begin{aligned} \ln p(X, \theta) \quad = \quad & \int q(\phi) \ln p(X, \theta, \phi) d\phi - \int q(\phi) \ln q(\phi) d\phi \\ & - \int q(\phi) \ln p(\phi | X, \theta) d\phi + \int q(\phi) \ln q(\phi) d\phi \end{aligned} \tag{11}$$

Comment: On the LHS, we notice that $\ln p(X, \theta)$ doesn't involve ϕ in any way. Therefore it's a constant from the perspective of the integral and can be brought out front. The resulting integral is of a probability distribution. Even though we haven't defined $q(\phi)$ yet, we know it's integral has to equal to, hence the LHS. For the RHS we have added and subtracted the same thing, leaving the result unchanged. However, we do this because the result will give us insights as to how to design the EM algorithm and prove it works.

$$\ln p(X, \theta) = \int q(\phi) \ln \frac{p(X, \theta, \phi)}{q(\phi)} d\phi + \int q(\phi) \ln \frac{q(\phi)}{p(\phi | X, \theta)} d\phi \tag{12}$$

Comment: This last equality is the EM master equation we have been aiming to arrive at. This equation will tell us (1) what we should do, and (2) why it will work.

- Notice that the goal has been more than just to arrive at an equality for $\ln p(X, \theta)$. We already had that with $\ln p(X, \theta) = \ln p(X, \theta, \phi) - \ln p(\phi | X, \theta)$. However, this equality doesn't immediately help us. We will see that the final equation above does give us a potentially useful algorithm.

- Let's look at the EM equation term-by-term.

  1. $\ln p(X, \theta)$: This is simply the objective function we want to optimize.

  2. $\mathcal{L}(\theta) := \int q(\phi) \ln \frac{p(X, \theta, \phi)}{q(\phi)} d\phi$: This is a function only of θ since we integrate ϕ out. Of course, the function $\mathcal{L}$ we end up with does depend on what $q(\phi)$ is.

  3. $\mathrm{KL}(q \| p) := \int q(\phi) \ln \frac{q(\phi)}{p(\phi | X, \theta)} d\phi$: This term is called the Kullback-Leibler divergence. It is very important and worth looking at more closely.

Kullback-Leibler (KL) divergence

- The KL-divergence is a function of two probability distributions. That is, the input to $\mathrm{KL}(q \| p)$ includes probability distributions $q(x)$ and $p(x)$ such that they are defined on the same support (i.e., the same values for x can be input to both of them). The output is a number.

- The KL-divergence is a similarity measure between q and p in that $\text{KL}(q\|p) \geq 0$ for all q and p and $\text{KL}(q\|p) = 0$ only in the case where $q(x) = p(x)$ for all x—that is, when q and p are exactly the same distribution.

- The KL-divergence is not technically a distance measure because, for that to be the case, we would need that for any three distributions q, p and f, $\text{KL}(q\|f) + \text{KL}(f\|p) \geq \text{KL}(q\|p)$ (called the triangle inequality), and this can be shown to not always be true.

- We can show that $\text{KL}(q\|p) \geq 0$ by using the concavity of $\ln(\cdot)$. Because $\ln(\cdot)$ is concave, by Jensen's inequality we can say that for any distribution $q(x)$ on x, $\mathbb{E}_q[\ln x] \leq \ln \mathbb{E}_q[x]$. This inequality itself requires a proof, which will be given in any convex optimization class. For now we will only use this property of the $\ln$ function. First, notice that

$$0 = \ln 1 = \ln \int p(x)dx = \ln \int q(x)\frac{p(x)}{q(x)}dx \tag{13}$$

The last in this seemingly trivial sequence of equalities can be interpreted as $\ln \mathbb{E}_q[\frac{p(x)}{q(x)}]$. From Jensen's inequality and the concavity of $\ln$,

$$0 \;=\; \ln \mathbb{E}_q\Big[\frac{p(x)}{q(x)}\Big] \;\geq\; \mathbb{E}_q\left[\ln \frac{p(x)}{q(x)}\right] \;=\; \int q(x) \ln \frac{p(x)}{q(x)}dx \tag{14}$$

And we immediately have the non-negativity of KL because

$$0 \geq \int q(x) \ln \frac{p(x)}{q(x)}dx \qquad \Leftrightarrow \qquad 0 \leq \int q(x) \ln \frac{q(x)}{p(x)}dx \;:=\; \text{KL}(q\|p)$$

Discussion on the EM equality

- Before presenting the EM algorithm and then showing why it works, we make two observations about the equality

$$\ln p(X,\theta) = \underbrace{\int q(\phi) \ln \frac{p(X,\theta,\phi)}{q(\phi)}d\phi}_{=\,\mathcal{L}(\theta)} + \underbrace{\int q(\phi) \ln \frac{q(\phi)}{p(\phi|X,\theta)}d\phi}_{=\text{KL}(q\|p)}$$

1. For any *fixed* value of θ, changing $q(\phi)$ doesn't change what the RHS adds up to because the LHS only changes with θ. All that is changing with $q(\phi)$ is the breakdown of how much the first and second term of the RHS contribute to the fixed total.

2. Since $\text{KL}(q\|p) \geq 0$, the term $\mathcal{L}(\theta) \leq \ln p(X,\theta)$ and we only have $\mathcal{L}(\theta) = \ln p(X,\theta)$ when $q(\phi) = p(\phi|X,\theta)$.

3. One might be tempted to ask: If we know $p(\phi|X,\theta)$, can we just define $q(\phi) := p(\phi|X,\theta)$ and plug back in? The answer is yes, but this will defeat the purpose of EM. Notice that if we plug in this value for $q(\phi)$, we will get out that

$$\ln p(X,\theta) = \int p(\phi|X,\theta) \ln \frac{p(X,\theta,\phi)}{p(\phi|X,\theta)}d\phi$$

We've simply found another way to write $\ln p(X,\theta)$ as a function only of θ, and if there's no easy solution for θ for the LHS, there won't be one for the RHS either. The trick is to keep $q(\phi)$ as a separate object from θ, and use one of them when updating the other.

The EM algorithm

- Using the RHS of the EM equality one can devise a means for iteratively updating θ, and then prove that it is monotonically increasing $\ln p(X, \theta)$. That is, in the iterative algorithm below, we will get out a sequence $\theta_1, \theta_2, \theta_3, \ldots$. We first focus on how to get this sequence, then we will show that, using this procedure, the generated sequence has the property that

$$\ln p(X, \theta_1) \leq \ln p(X, \theta_2) \leq \ln p(X, \theta_3) \leq \cdots \tag{15}$$

- The EM algorithm is traditionally broken into two steps, an "E" (expectation) step and an "M" (maximization) step. Each iteration of the algorithm consists of one E and one M step, which are then repeated in the next iteration.

  E-Step at iteration t: Set $q_t(\phi) = p(\phi|X, \theta_{t-1})$ and calculate

$$\mathcal{L}_t(\theta) = \int q_t(\phi) \ln p(X, \theta, \phi) d\phi - \int q_t(\phi) \ln q_t(\phi) d\phi \tag{16}$$

  Notice that $q_t(\phi)$ is the conditional posterior using the value of θ from the previous iteration. Also, notice that the second term in $\mathcal{L}_t(\theta)$ is simply a constant w.r.t. θ.

  M-Step at iteration t: Treat $\mathcal{L}_t(\theta)$ as a function of θ and find the value of θ that maximizes it,

$$\theta_t \;=\; \arg\max_{\theta} \; \mathcal{L}_t(\theta) \tag{17}$$

  Because the second term in $\mathcal{L}_t(\theta)$ doesn't factor into this maximization, in practice it is only necessary to calculate $\int q_t(\phi) \ln p(X, \theta, \phi) d\phi$ and maximize this over θ.

- Before we analyze what this is doing, there are two crucial assumptions underlying this algorithm that would make it preferable to directly optimizing $\ln p(X, \theta)$ using gradient methods:

  1. First, we assume we can calculate $p(\phi|X, \theta)$ in closed form using Bayes rule. If we can't then we're already stuck when trying to update $\mathcal{L}(\theta)$, which requires this distribution.

  2. Second, in the M-step we have to optimize over $\mathcal{L}_t(\theta)$, which is a function of θ. The original thing we want to optimize over, $\ln p(X, \theta)$, is also a function of θ. If optimizing $\mathcal{L}_t(\theta)$ is no easier than $\ln p(X, \theta)$, then there's not much point to this.

     That is, if we can't solve $\nabla_\theta \mathcal{L}_t(\theta) = 0$ for θ analytically, then we're back where we started. While the following statement might not be true 100% of the time (but I can't think of a counter-example offhand), in big-picture terms if you need to use gradient methods to optimize $\mathcal{L}_t(\theta)$, you might as well just use gradient methods to optimize $\ln p(X, \theta)$ instead.

- Therefore, just like Gibbs sampling required an additional assumption about the model for it to be useful (that the conditional posterior distributions of all variables are in closed form), EM also makes assumptions about the model before claiming to be useful (#1 and #2 above). Of course, EM will technically work if #2 above isn't satisfied, which is something to at least keep in mind.

Analysis of the EM algorithm

- The last part of this discussion on EM will show that the sequence $\theta_1, \theta_2, \ldots$ we get from this procedure is monotonically increasing $\ln p(X, \theta)$. First, recall that

$$\mathcal{L}_t(\theta) = \int q_t(\phi) \ln \frac{p(X, \theta, \phi)}{q_t(\phi)} d\phi, \qquad \text{KL}(q_t(\phi) \| p(\phi|X, \theta)) = \int q_t(\phi) \ln \frac{q_t(\phi)}{p(\phi|X, \theta)} d\phi$$

  The following sequence of inequalities shows that $\ln p(X, \theta_{t-1}) \leq \ln p(X, \theta_t)$, followed by our elaboration on each line. In the transition from iteration $t-1$ to t we have that

$$\begin{aligned}
\ln p(X, \theta_{t-1}) &= \mathcal{L}_t(\theta_{t-1}) + \text{KL}(q_t(\phi) \| p(\phi|X, \theta_{t-1})) && (18)\\
&= \mathcal{L}_t(\theta_{t-1}) && (19)\\
&\leq \mathcal{L}_t(\theta_t) && (20)\\
&\leq \mathcal{L}_t(\theta_t) + \text{KL}(q_t(\phi) \| p(\phi|X, \theta_t)) && (21)\\
&= \ln p(X, \theta_t) && (22)
\end{aligned}$$

  1. The first line is simply the EM master equation using $q(\phi) \leftarrow q_t(\phi) = p(\phi|X, \theta_{t-1})$ and evaluating the functions at $\theta = \theta_{t-1}$.
  2. Since $q_t(\phi) = p(\phi|X, \theta_{t-1})$, the KL-divergence equals zero, so we can remove it. These two lines constitute what we do for the E-step.
  3. We know that $\mathcal{L}_t(\theta_{t-1}) \leq \mathcal{L}_t(\theta_t)$ because $\theta_t = \arg\max_\theta \mathcal{L}_t(\theta)$. This is the M-step.
  4. The next question is how $\mathcal{L}_t(\theta_t)$ relates to $\ln p(X, \theta_t)$. In the discussion above we have already been able to show that $\mathcal{L}(\theta) \leq \ln p(X, \theta)$. However, to see this again explicitly, we add a strategically chosen non-negative number to $\mathcal{L}_t(\theta_t)$. Specifically, we add the KL-divergence $\text{KL}(q_t(\phi) \| p(\phi|X, \theta_t))$. When we do this, we see that the second to last line is again simply the EM equation for $\ln p(X, \theta_t)$.

- Also worth emphasizing here is that $\text{KL}(q_t(\phi) \| p(\phi|X, \theta_t)) > 0$ because $q_t(\phi) \neq p(\phi|X, \theta_t)$ since $q_t(\phi) = p(\phi|X, \theta_{t-1})$ and we can assume $\theta_{t-1} \neq \theta_t$ (otherwise the algorithm has converged). Therefore, we can return to the E-step by updating $q(\phi)$ at iteration $t+1$ to account for the new value of θ. Hence there is a natural loop we can iterate back and forth between, where we update $q(\phi)$ and then update θ. The above inequalities show that any single completion of this cycle will find a new θ that is better than the old one in terms of maximizing $\ln p(X, \theta)$.

- The final question is whether the sequence $\theta_1, \theta_2, \ldots$ is converging to a local optimal solution of $\ln p(X, \theta)$. It is easy to think how we could have $\ln p(X, \theta_t) \leq \ln p(X, \theta_{t+1})$ for all t, but also have θ_∞ not be at the top of one of the "hills" of $\ln p(X, \theta)$. It can be shown that EM does converge to one of these peaks, but the proof is significantly more complicated and so we skip it in this class. Suffice it to say that EM will give the MAP solution (or a local optimal of it) for any interesting model you will come across.

The EM algorithm for probit regression

- Next, we derive an EM algorithm for the probit regression model. In this model the weights w correspond to θ. We first need to find a good ϕ that will make things work out nicely. It's worth

dwelling on this for a while before jumping to the algorithm. Even though this qualifies as a straightforward application of EM, this is a good example of how the generic view taken above doesn't quite make deriving an EM algorithm for a specific problem an automatic procedure.

- For this model, the joint likelihood factorizes as

$$p(\vec{y}, w|X) = p(w)\prod_{i=1}^{N} p(y_i|w, x_i) \tag{23}$$

- With EM, our goal is to find a random variable, ϕ such that

$$\int p(\vec{y}, w, \phi|X)d\phi = p(\vec{y}, w|X) \tag{24}$$

- Actually, here we will see how our above simplified version of EM, where we had one θ and one ϕ, can be generalized. Instead, let $\phi = (\phi_1, \ldots, \phi_N)$. We pick these such that

$$\int p(\vec{y}, w, \phi|X)d\phi = \prod_{i=1}^{N}\int p(y_i, \phi_i, w|X)d\phi_i = p(\vec{y}, w|X) \tag{25}$$

  Therefore, we assume that the variables ϕ_i are independent from each other and each (x_i, y_i) pair has a ϕ_i associated with it in some way.

- Notice that by assuming that the ϕ_i are independent of each other and only dependent on (x_i, y_i) and w, the posterior on the vector $\phi = (\phi_1, \ldots, \phi_N)$ factorizes as well

$$\begin{aligned} p(\phi|\vec{y}, w, X) &= \frac{\prod_{i=1}^{N} p(y_i|\phi_i, w, x_i)p(\phi_i|w, x_i)}{\prod_{i=1}^{N}\int p(y_i|\phi_i, w, x_i)p(\phi_i|w, x_i)d\phi_i} \\ &= \prod_{i=1}^{N}\frac{p(y_i|\phi_i, w, x_i)p(\phi_i|w, x_i)}{\int p(y_i|\phi_i, w, x_i)p(\phi_i|w, x_i)d\phi_i} \\ &= \prod_{i=1}^{N} p(\phi_i|y_i, w, x_i) \end{aligned} \tag{26}$$

- How does this impact the EM equation? We have that

$$\ln p(\vec{y}, w|X) = \ln p(w) + \sum_{i=1}^{N}\ln p(y_i|w, x_i) \tag{27}$$

  For a single $\ln p(y_i, w|x_i)$ we have from the derivation above that

$$\ln p(y_i, w|x_i) = \int q(\phi_i)\ln\frac{p(y_i, \phi_i|w, x_i)}{q(\phi_i)}d\phi_i + \int q(\phi_i)\ln\frac{q(\phi_i)}{p(\phi_i|y_i, w, x_i)}d\phi_i \tag{28}$$

  and so summing the LHS and RHS over i we have that

$$\ln p(\vec{y}, w|X) = \ln p(w) + \sum_{i=1}^{N}\int q(\phi_i)\ln\frac{p(y_i, \phi_i|w, x_i)}{q(\phi_i)}d\phi_i + \sum_{i=1}^{N}\int q(\phi_i)\ln\frac{q(\phi_i)}{p(\phi_i|y_i, w, x_i)}d\phi_i \tag{29}$$

- To belabor the point, we could have gotten to this last EM equation from the other direction, and it can be worthwhile to see it from this perspective as well w.r.t. choosing q. Let ϕ be the latent variable we introduce—possibly a vector. From EM we know that

$$\ln p(\vec{y}, w|X) = \int q(\phi) \ln \frac{p(\vec{y}, \phi, w|X)}{q(\phi)} d\phi + \int q(\phi) \ln \frac{q(\phi)}{p(\phi|\vec{y}, w, X)} d\phi \tag{30}$$

- Next, assume that we pick $\phi = (\phi_1, \ldots, \phi_N)$ such that $p(\vec{y}, \phi, w|X) = \prod_{i=1}^N p(y_i, \phi_i, w|x_i)$. Then we showed above that $p(\phi|\vec{y}, w, X) = \prod_{i=1}^N p(\phi_i|y_i, w, x_i)$. From the EM algorithm, we know that we need to use this posterior to select q,

$$q(\phi) = p(\phi|\vec{y}, w, X) = \prod_{i=1}^N p(\phi_i|y_i, w, x_i) \tag{31}$$

  This implies a *factorization* of the $q(\phi)$ distribution (which results in the above EM equation)

$$q(\phi) = \prod_{i=1}^N q(\phi_i) \qquad \text{and} \qquad q(\phi_i) = p(\phi_i|y_i, w, x_i) \tag{32}$$

- Therefore, our goal is threefold:

  1. Define ϕ_i such that

$$\int p(y_i, \phi_i|w, x_i) d\phi_i = p(y_i|w, x_i) = \Phi\Big(\frac{x_i^T w}{\sigma}\Big)^{y_i} \left[1 - \Phi\Big(\frac{x_i^T w}{\sigma}\Big)\right]^{1-y_i}$$

  2. Derive a closed-form posterior distribution $p(\phi_i|y_i, x_i, w)$ using Bayes rule.

  3. Calculate the expectation $\mathcal{L}(w)$ and find that we can analytically find the value of w that maximizes it.

- In general, once #1 is done, #2 and #3 follow immediately (or on the flipside, we immediately realize that the ϕ we picked won't work and we should try again). It's step #1 that requires the insights that come from experience working with models, and that can't be reduced to a set of instructions like #2 and #3 can. Therefore, #1 is going to appear like it's coming out of nowhere below, while #2 and #3 will be more straightforward.

- Step 1: Consider the following expanded model

$$y_i = \mathbb{1}(\phi_i > 0), \quad \phi_i \sim \text{Normal}(x_i^T w, \sigma^2) \tag{33}$$

  Notice that even though we have an indicator function for y_i, we can still view this as a (deterministic) probability distribution: $p(y_i = 1|\phi_i) = \mathbb{1}(\phi_i > 0)$. In other words, y_i isn't random given ϕ_i, but it makes perfect mathematical sense to write the joint likelihood

$$\begin{aligned} p(y_i = 1, \phi_i|w, x_i) &= p(y_i = 1|\phi_i) p(\phi_i|w, x_i) \\ &= \mathbb{1}(\phi_i > 0)(2\pi\sigma^2)^{-\frac{1}{2}} \mathrm{e}^{-\frac{1}{2\sigma^2}(\phi_i - x_i^T w)^2} \end{aligned} \tag{34}$$

For $y_i = 0$, we use the indicator $\mathbb{1}(\phi_i \leq 0)$ instead. Next we need to calculate the marginal distribution $p(y_i|w, x_i)$. Clearly

$$\begin{aligned}\int p(y_i = 1, \phi_i|w, x_i)d\phi_i &= \int_{-\infty}^{\infty} \mathbb{1}(\phi_i > 0)(2\pi\sigma^2)^{-\frac{1}{2}}\mathrm{e}^{-\frac{1}{2\sigma^2}(\phi_i - x_i^T w)^2} \\ &= \int_0^{\infty} (2\pi\sigma^2)^{-\frac{1}{2}}\mathrm{e}^{-\frac{1}{2\sigma^2}(\phi_i - x_i^T w)^2} \\ &= P(\phi_i > 0) \end{aligned} \tag{35}$$

However, remember that we need to show that this equals $\Phi(x_i^T w/\sigma) = \int_{-\infty}^{x_i^T w/\sigma}(2\pi)^{-\frac{1}{2}}\mathrm{e}^{-\frac{1}{2}s^2}ds$. To do this, we first observe that we can draw a random variable $\phi_i \sim \text{Normal}(x_i^T w, \sigma^2)$ as follows

$$\phi_i = x_i^T w + \sigma s, \qquad s \sim \text{Normal}(0, 1) \tag{36}$$

Therefore, the probability

$$P(\phi_i > 0) = P(x_i^T w + \sigma s > 0) = P(s > -x_i^T w/\sigma)$$

The final step is to recognize that $P(s > -x_i^T w/\sigma) = P(s \leq x_i^T w/\sigma)$ since s is a standard normal distribution symmetric around zero. Therefore,

$$\int p(y_i = 1, \phi_i|w, x_i)d\phi_i = P(\phi_i > 0) = P(s \leq x_i^T w/\sigma) = \Phi(x_i^T w/\sigma) \tag{37}$$

- We have therefore found a hierarchical expansion of the probit regression model,

$$y_i = \mathbb{1}(\phi_i > 0), \quad \phi_i \sim \text{Normal}(x_i^T w, \sigma^2), \quad w \sim \text{Normal}(0, \lambda^{-1}I)$$

If we integrate out all the ϕ_i in this model, we return to the original probit regression model. Before moving on, it's worth discussing this a little more. First, there's nothing inherently "right" or "correct" about a probit model. Because we picked that as our desired model, we had to do some work to get to this hierarchical representation for EM. However, the probit model was only picked because it "makes sense" to do it.

The point I am making is that, in my opinion, the above model makes a lot of sense too—we could have made *this* model our starting point. In that case, we could have done MAP on w and the vector ϕ. Or, we could have decided to *integrate out* all uncertainty in ϕ and do MAP only for w. As shown below, EM would then give us conditional posterior distributions on the ϕ_i rather than a point estimate. In general, the more variables you can integrate out the better (one reason is that the model will avoid over-fitting). Coming at EM from this perspective, the problem is much easier since we already know the "hidden" variable ϕ to integrate out—after all, it was part of the definition of the model in this case.

- Step 2: We've found a latent variable ϕ_i that gives the correct marginal distribution. Next we need to calculate the posterior $p(\phi_i|y_i, x_i, w)$. By Bayes rule,

$$\begin{aligned} p(\phi_i|y_i, x_i, w) &= \frac{p(y_i|\phi_i)p(\phi_i|x_i, w)}{\int p(y_i|\phi_i)p(\phi_i|x_i, w)d\phi_i} \\ &= \frac{\mathbb{1}\{\text{sign}(\phi_i) = 2y_i - 1\}\mathrm{e}^{-\frac{1}{2}(\phi_i - x_i^T w)^2}}{\int_{-\infty}^{\infty} \mathbb{1}\{\text{sign}(\phi_i) = 2y_i - 1\}\mathrm{e}^{-\frac{1}{2}(\phi_i - x_i^T w)^2}d\phi_i} \end{aligned} \tag{38}$$

Unfortunately the indicator doesn't look nice, but all it is saying is that ϕ_i must be positive if $y_i = 1$ and must be negative if $y_i = 0$. This distribution is called a *truncated normal distribution*.

- If $y_i = 1$, then the truncated normal distribution $TN_1(x_i^T w, \sigma^2)$ is defined to be the part of Normal$(x_i^T w, \sigma^2)$ defined on $\mathbb{R}_+$ re-normalized to give a probability distribution. In other words, it's the distribution of a Gaussian random variable *conditioned on* knowledge that it is positive. The reverse holds when $y_i = 0$, in which case we can write $TN_0(x_i^T w, \sigma^2)$ defined on $\mathbb{R}_-$.

- Step 3: Finally, we need to calculate

$$\mathcal{L}(w) = \ln p(w) + \sum_{i=1}^{N} \mathbb{E}_q[\ln p(y_i, \phi_i | w, x_i)] + \text{constant} \tag{39}$$

Since the joint distribution including the extra ϕ variables is

$$p(\vec{y}, \phi, w | X) = p(w) \prod_{i=1}^{N} p(y_i|\phi_i) p(\phi_i | x_i, w) \tag{40}$$

we have that

$$\mathcal{L}(w) = -\frac{\lambda}{2} w^T w + \sum_{i=1}^{N} \underbrace{\mathbb{E}_q[\ln \mathbb{1}\{\text{sign}(\phi_i) = 2y_i - 1\}]}_{=0} - \frac{1}{2\sigma^2} \mathbb{E}_q[(\phi_i - x_i^T w)^2] + \text{const.} \tag{41}$$

One of the expectations always equals zero, since $q(\phi_i) = TN_{y_i}(x_i^T w, \sigma^2)$, and so it is only integrating over values of ϕ_i for which $\mathbb{1}\{\text{sign}(\phi_i) = 2y_i - 1\} = 1$. Also, if we expand the square of the rightmost term, we can put anything not involving w into the constant, since we want to maximize $\mathcal{L}(w)$ over w. As a result, we want to maximize

$$\mathcal{L}(w) = -\frac{\lambda}{2} w^T w - \sum_{i=1}^{N} \frac{1}{2\sigma^2} (w^T x_i x_i^T w - 2 w^T x_i \mathbb{E}_q[\phi_i]) + \text{constant} \tag{42}$$

Solving for $\nabla_w \mathcal{L}(w) = 0$, we find that

$$w = \arg\max_w \ \mathcal{L}(w) \qquad \Leftrightarrow \qquad w = \left(\lambda I + \sum_{i=1}^{N} x_i x_i^T / \sigma^2 \right)^{-1} \left(\sum_{i=1}^{N} x_i \mathbb{E}_q[\phi_i] / \sigma^2 \right) \tag{43}$$

We just need to know what $\mathbb{E}_q[\phi_i]$ is under the conditional posterior $q(\phi_i) = TN_{y_i}(x_i^T w, \sigma^2)$. Looking this up in a textbook or on Wikipedia, we find that

$$\mathbb{E}_q[\phi_i] = \begin{cases} x_i^T w + \sigma \times \dfrac{\Phi'(-x_i^T w/\sigma)}{1 - \Phi(-x_i^T w/\sigma)} & \text{if } \ y_i = 1 \\ \\ x_i^T w + \sigma \times \dfrac{-\Phi'(-x_i^T w/\sigma)}{\Phi(-x_i^T w/\sigma)} & \text{if } \ y_i = 0 \end{cases} \tag{44}$$

The function $\Phi'(s)$ is the probability density function (PDF) of a Normal$(0, 1)$ distribution evaluated at s. $\Phi(s)$, as before, is the CDF of a Normal$(0, 1)$ distribution evaluated at s. These can be quickly evaluated by calling a built-in function.

- Notice that, even though it took several steps to get to a final EM algorithm, we have everything we want:

    1. An expression for the conditional posterior distribution $p(\phi_i|y_i, w, x_i)$ as a known distribution (even though it's an atypical distribution, it's still one we know how to take the expectation with respect to, which is all that matters here)

    2. A closed form expression for updating w that we can evaluate quickly in code without having to use iterative gradient methods.

- It might take a few readings of the above to appreciate all they nuances of the EM algorithm for probit regression and why it's correct (i.e., maximizes $\ln p(\vec{y}, w|X)$ over w). However, the final algorithm can be summarized very easily.

An EM algorithm for probit regression

1. Initialize w_0 to a vector of all zeros.

2. For iteration $t = 1, \ldots, T$

    (a) E-Step: Calculate the vector $\mathbb{E}_{q_t}[\phi] = \big(\mathbb{E}_{q_t}[\phi_1], \ldots, \mathbb{E}_{q_t}[\phi_N]\big)$, where

$$\mathbb{E}_{q_t}[\phi_i] = \begin{cases} x_i^T w_{t-1} + \sigma \times \dfrac{\Phi'(-x_i^T w_{t-1}/\sigma)}{1 - \Phi(-x_i^T w_{t-1}/\sigma)} & \text{if } y_i = 1 \\ x_i^T w_{t-1} + \sigma \times \dfrac{-\Phi'(-x_i^T w_{t-1}/\sigma)}{\Phi(-x_i^T w_{t-1}/\sigma)} & \text{if } y_i = 0 \end{cases}$$

    (b) M-Step: Update the vector w using the expectations above in the following equation

$$w_t = \left(\lambda I + \sum_{i=1}^{N} x_i x_i^T/\sigma^2\right)^{-1} \left(\sum_{i=1}^{N} x_i \mathbb{E}_{q_t}[\phi_i]/\sigma^2\right)$$

    (c) Calculate $\ln p(\vec{y}, w_t|X)$ using the equation

$$\begin{aligned} \ln p(\vec{y}, w_t|X) &= \frac{d}{2}\ln\left(\frac{\lambda}{2\pi}\right) - \frac{\lambda}{2} w_t^T w_t \\ &\quad + \sum_{i=1}^{N} y_i \ln \Phi(x_i^T w_t/\sigma) + \sum_{i=1}^{N} (1 - y_i) \ln(1 - \Phi(x_i^T w_t/\sigma)) \end{aligned}$$

- Part 2c can be used to assess convergence and therefore determine what T should be. Practically speaking, it can also be used to make sure your implementation is correct, since we know from the earlier proof that $\ln p(\vec{y}, w_t|X)$ must be monotonically increasing in t. If you find that this is not the case with your implementation, then you can be sure there is a bug somewhere.

EECS E6720 Bayesian Models for Machine Learning

Columbia University, Fall 2016

Lecture 5, 10/13/2016

Instructor: John Paisley

- Last week we talked about using the EM algorithm for MAP inference. Today, we will apply EM to a sequence of models to see where EM breaks down and variational inference (VI) takes over. The purpose is to discuss EM in more detail and also see how it closely relates to variational methods. Variational inference provides a general way to approximate the full posterior distribution of all model variables, and is a new technique to add to the arsenal, along with the Laplace approximation and MCMC methods (of which we only discussed Gibbs sampling).

The "Core Model"

- We are more interested in the general EM and VI inference techniques, but I think they're easier to differentiate through an easy modeling example. Therefore, we will build on the basic linear regression model discussed in an earlier lecture. Our development of this model will not be the main thing of interest. Instead, we will be more interested in what we can do with each version of this model in the context of EM and VI.

- Remember that we're given pairs $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$ where $x \in \mathbb{R}^d$ and $y \in \mathbb{R}$ and we model this data as

$$y_i \overset{ind}{\sim} \text{Normal}(x_i^T w, \alpha^{-1}), \qquad w \sim \text{Normal}(0, \lambda^{-1} I) \tag{1}$$

  There is no prior distribution on x, and so we are in the "discriminative" setting. This is in contrast to the Bayes classifier where we had priors on x and so were in the "generative" setting.

- Posterior calculation: Earlier, we saw how we could calculate the posterior distribution of this model in closed form,

$$p(w|x, y) = \frac{p(y|w, x)p(w)}{p(y|x)} = \text{Normal}(\mu, \Sigma) \tag{2}$$

  where

$$\Sigma = \Big(\lambda I + \alpha \sum_{i=1}^N x_i x_i^T\Big)^{-1}, \qquad \mu = \Sigma\Big(\sum_{i=1}^N \alpha y_i x_i\Big).$$

  (This time, we let the index-free x and y represent all x_i and y_i, respectively.)

- Therefore, for this simple model the inference problem is very easy. No iterative algorithms are needed and we can directly go to the full posterior of all model variables (in this case, only w) in one step. We next develop this model in a way where this is not possible.

- We notice that there are two model parameters that need to be set and will likely have a significant impact on $p(w|y, x)$. That is, even though we have a mathematically correct answer for this posterior distribution, it is only correct under the assumptions we make about the values of α and λ. (And as emphasized before, on the assumption of the model to begin with, which is a simplifying assumption about the data-generating processes underlying a real-world phenomenon.)

- Therefore, even though we have solved for $p(w|y, x)$, we can't claim that we've solved the problem of modeling (x, y) with a linear regression model—we probably can never claim that.

- Since α corresponds to the inverse noise variance, it arguably has the greater impact on our predictions, so let's address this one first:

    - The default solution in the wider machine learning community is to use "cross-validation." In short, that means we try a bunch of values for α and see which one is best according to some problem we're interested in.

    - In the Bayesian modeling world, the default solution is to put a prior distribution on α and try to learn it using posterior inference.

Core model (version 2.0)

- We expand the core model hierarchically by putting a prior distribution on α. Out of convenience we pick a conjugate gamma prior,

$$y_i \overset{ind}{\sim} \text{Normal}(x_i^T w, \alpha^{-1}), \qquad w \sim \text{Normal}(0, \lambda^{-1} I), \qquad \alpha \sim \text{Gamma}(a, b) \tag{3}$$

- As was the case with w, by adding a prior on α, we are committing ourselves to one of the following:

    1. Try to learn the full posterior of α and w (exactly or approximately)
    2. Learn a point estimate of the variables via MAP inference
    3. Integrate out (or marginalize) some variables to learn the others (e.g. with point estimate)

- Let's consider each of these:

    1. The full posterior distribution of α and w is

$$p(w, \alpha|y, x) = \frac{p(y|w, \alpha, x)p(w)p(\alpha)}{\int\int p(y|w, \alpha, x)p(w)p(\alpha)dwd\alpha} \tag{4}$$

    When we try to calculate the normalizing constant, we see that one integral can be performed (either over α or w), but this makes the other integral intractable. In this case, the interactions between w and α in the model are such that we can't find the full posterior.

(This is not always the case, e.g., in the first homework.) To approximate this posterior, we could use Laplace or MCMC sampling (Gibbs sampling will work in this case). However, since we're more interested in introducing new inference techniques than in solving specific models now, let's consider other options.

2. We could also simply do MAP over w and α,

$$\begin{aligned} w, \alpha &= \arg\max_{w,\alpha} \ln p(y, w, \alpha|x) \\ &= \arg\max_{w,\alpha} \ln p(y|w, \alpha, x) + \ln p(w) + \ln p(\alpha) \end{aligned} \tag{5}$$

We can take the derivative of this with respect to w or α and solve for each variable in closed form (but not both at once). This would lead to a "coordinate ascent" algorithm, where we iterate between maximizing w.r.t. w or α holding the other one fixed.

However, this in some sense is unsatisfying because we're giving up all measures of uncertainty and learning a point estimate of all variables. In any case, I argue that it should be unsatisfying in light of option #3.

3. Another option is to integrate out α. That is, to calculate the marginal likelihood

$$p(y, w|x) = \int p(y, w, \alpha|x) d\alpha \tag{6}$$

We have a few options after doing this. The first one to try is to do posterior inference for $p(w|y, x)$ using Bayes rule. However, notice that a different model is implied here. When we write $p(w|y, x) \propto p(y|w, x)p(w)$, we see that the term $p(y|w, x)$ isn't a Gaussian anymore since it's found by integrating out α. That is, $p(y|w, x) = \int p(y, \alpha|w, x) d\alpha$, which gives a student-t distribution, not a Gaussian like in the core model.[1] Therefore, the prior and likelihood aren't conjugate anymore.

Another option would be to maximize $p(y, w|x)$ over w using MAP. However, using the student-t marginal $p(y|w, x)$, we see that $\nabla_w \ln p(y|w, x)p(w) = 0$ does not have a closed form solution for w. Therefore, we would have to use gradient methods if we wanted to directly maximize $p(y, w|x)$ over w. However, this setup is reminiscent of EM...

- Let's now focus on option #3. We want to maximize $p(y, w|x) = \int p(y, w, \alpha|x) d\alpha$ over w. Doing so directly working with $\ln p(y, w|x)$ requires gradient methods, which is something we might want to avoid. (This is not a blanket statement and optimization researchers would probably disagree. But if we have an option that gives closed form updates, we would rather use that.)

- Remember from our discussion on the EM algorithm that we want to maximize a joint likelihood, e.g., $p(y, w|x)$, but we want to avoid gradient methods. To this end, we need to find a hidden variable, e.g., α, such that $p(y, w|x) = \int p(y, w, \alpha|x) d\alpha$. Our initial discussion started with defining the model $p(y, w|x)$ and the tricky part was finding the α. In this case, the latent variable is actually part of the original model definition and we are then finding a point estimate of the marginal distribution of that model.

[1] In that model, α was a parameter so we didn't bother to write $p(y|w, x, \alpha)$, but that's technically what $p(y|w, x)$ corresponds to in the core model, whereas now $p(y|w, x)$ corresponds to the above integral over α.

1. The first direction was hard: Start with a desired marginal distribution and introduce a variable that gives that marginal distribution after integrating it out.

2. This new direction is easy: Start with the expanded model as a definition and then try to maximize over the marginal. We care about the distributions in the larger model, not the distribution that comes from marginalizing. "Whatever $p(y, w|x) = \int p(y, w, \alpha|x)d\alpha$ ends up being is what it ends up being." We're just going to maximize it, possibly without even knowing what it is, since EM doesn't require us to ever solve this integral in order to optimize the marginal over w.

EM for the core model (version 2.0)

- Therefore, we're going to find a point estimate of w to maximize the marginal distribution $p(y, w|x) = \int p(y, w, \alpha|x)d\alpha$ of the second version of the core model. We'll do so by treating α as the extra variable in an EM setting. The EM equation in this case is

$$\ln p(y, w|x) = \underbrace{\int q(\alpha) \ln \frac{p(y, w, \alpha|x)}{q(\alpha)} d\alpha}_{\mathcal{L}(w)} + \underbrace{\int q(\alpha) \ln \frac{q(\alpha)}{p(\alpha|y, w, x)} d\alpha}_{\mathrm{KL}(q\|p)} \tag{7}$$

E-Step

Can we find $p(\alpha|y, w, x)$?

$$\begin{aligned} p(\alpha|y, w, x) &\propto \prod_{i=1}^{N} p(y_i|\alpha, w, x_i)p(\alpha) && (8) \\ &\propto \underbrace{\alpha^{\frac{N}{2}} \mathrm{e}^{-\frac{\alpha}{2}\sum_{i=1}^{N}(y_i - x_i^T w)^2}}_{\text{product of normal likelihoods}} \times \underbrace{\alpha^{a-1}\mathrm{e}^{-b\alpha}}_{\text{gamma prior}} \\ &= \mathrm{Gamma}\Big(a + \frac{N}{2}, b + \frac{1}{2}\sum_{i=1}^{N}(y_i - x_i^T w)^2\Big) && (9) \end{aligned}$$

So we can set $q_t(\alpha) = p(\alpha|y, w_{t-1}, x)$ at iteration t. We then calculate the expectation,

$$\begin{aligned} \mathcal{L}_t(w) &= \mathbb{E}_q[\ln p(y, \alpha|w, x)p(w)] - \mathbb{E}_q[\ln q(\alpha)] && (10) \\ &= -\frac{\mathbb{E}_{q_t}[\alpha]}{2}\sum_{i=1}^{N}(y_i - x_i^T w)^2 - \frac{\lambda}{2}w^T w + \text{constant w.r.t. } w \end{aligned}$$

M-Step

We can find q and complete the E-step by calculating an analytic function $\mathcal{L}(w)$. Next, we need to see if we can maximize $\mathcal{L}(w)$ in closed form. If not, then we're likely no better off than we were with $\ln p(y, w|x)$. Differentiating $\nabla_w \mathcal{L}_t(w)$ and setting to zero, we find that

$$w_t = \Big(\lambda I + \mathbb{E}_{q_t}[\alpha]\sum_{i=1}^{N} x_i x_i^T\Big)^{-1}\Big(\sum_{i=1}^{N}\mathbb{E}_{q_t}[\alpha]y_i x_i\Big) \tag{11}$$

The expectation of α is of a gamma random variable with the parameters from the E-step, so

$$\mathbb{E}_{q_t}[\alpha] = \frac{a + \frac{N}{2}}{b + \frac{1}{2}\sum_{i=1}^{N}(y_i - x_i^T w_{t-1})^2} \tag{12}$$

- Notice that if we plug this expression for $\mathbb{E}_{q_t}[\alpha]$ directly into the update of w_t, the EM algorithm is giving us a way of iteratively updating w using the previous value.

- Also notice that, while this is a "pure" EM problem along the lines of what we discussed last time, the interpretation here feels slightly different:

    1. Last week, the latent variable we introduced was just a stepping stone to get the point estimate we wanted. We didn't motivate that new variable or its q distribution as being something interesting itself.

    2. In this model, the latent variable α has a clear interpretation as relating to the observation noise. This was originally an important model parameter that we decided to put a prior distribution on. Therefore, we have a clear picture of what the "introduced" variable α means here, since it was introduced at the point of the model formulation and not at the point of inference. Therefore, we can think of $q(\alpha) = p(\alpha|y, w, x)$ as being interesting in its own right since it's the conditional posterior distribution of an important model variable.

    3. Therefore, using EM for this problem we make a compromise: We can't get the full posterior $p(w, \alpha|y, x)$, but we don't want to make point estimates for these two variables either, so we settle for a point estimate of w and a *conditional* posterior of α.

    4. Also notice that *we could have done the reverse*: We could have learned a point estimate of α and a conditional posterior $q(w)$ distribution. In this case, we would be doing MAP for $p(y, \alpha|x) = \int p(y, w, \alpha|x)dw$. For this specific model, the EM algorithm would work perfectly well since $q(w) = p(w|y, x, \alpha)$ is a multivariate Gaussian, $\mathcal{L}(\alpha)$ is analytic and maximized in closed form. However, w is clearly not just a stepping stone to learn α! In fact, we could interpret EM here as allowing us to calculate the posterior $p(w|y, x)$ of the original "core model" while additionally helping us *set* the parameter α.

Core model (version 3.0)

- If we were doing EM for version 2.0 of the core model, we would probably want to maximize point-wise over α and learn $q(w)$ as discussed in point #4 above. However, this discussion is directed primarily towards connecting EM with VI and so the development in this next version would not work out the way I want it to.

- We've integrated out α, but what about λ? We can again update the model by adding a conveniently chosen prior here,

$$y_i \overset{ind}{\sim} \text{Normal}(x_i^T w, \alpha^{-1}), \quad w \sim \text{Normal}(0, \lambda^{-1} I), \quad \alpha \sim \text{Gamma}(a, b), \quad \lambda \sim \text{Gamma}(e, f)$$

- Again, we can try to learn this model using Laplace, or MCMC sampling (Gibbs sampling works here), or point estimates of w, α and λ, or point estimates of some variables and marginalization over others...

- Let's again look into maximizing w over the the marginal distribution

$$p(y, w|x) = \int \int p(y, w, \alpha, \lambda|x) d\alpha d\lambda \tag{13}$$

EM for the core model (version 3.0)

- We now have two latent variables, but again we can write

$$\ln p(y, w|x) = \int\int q(\alpha, \lambda) \ln \frac{p(y, w, \alpha, \lambda|x)}{q(\alpha, \lambda)} d\alpha d\lambda + \int\int q(\alpha, \lambda) \ln \frac{q(\alpha, \lambda)}{p(\alpha, \lambda|y, w, x)} d\alpha d\lambda \tag{14}$$

- Side comment: Notice that the left hand side again contains $\ln p(y, w|x)$, but the right hand side is different. Does this mean that EM for versions 2.0 and 3.0 are equivalent? The answer is emphatically *no*. This is simply because

$$\int \underbrace{p(y, w, \alpha|x)}_{\text{model \#2}} d\alpha \;\neq\; \int\int \underbrace{p(y, w, \alpha, \lambda|x)}_{\text{model \#3}} d\alpha d\lambda \tag{15}$$

  Even though both of these can be written as $p(y, w|x)$, they are of *different models*. There is always an assumed model underlying a joint likelihood and simply writing $p(y, w|x)$ is not enough information to know what it is.

- Therefore, the EM algorithm we can derive for doing MAP of w in this model is over a different objective function than the previous model: If we were to actually calculate the marginal distributions $p(y, w|x)$ for version 2.0 and 3.0, we would see that they are different functions.

E-Step

According to the rules of EM, we need to set $q(\alpha, \lambda) = p(\alpha, \lambda|y, w, x)$. Using Bayes rule,

$$\begin{aligned} p(\alpha, \lambda|y, w, x) &= \frac{p(y|w, \alpha)p(\alpha)p(w|\lambda)p(\lambda)}{\int\int p(y|w, \alpha)p(\alpha)p(w|\lambda)p(\lambda)d\alpha d\lambda} \\ &= \underbrace{\frac{p(y, \alpha|w)}{\int p(y, \alpha|w, x)d\alpha}}_{= p(\alpha|y,w,x)} \cdot \underbrace{\frac{p(w, \lambda)}{\int p(w, \lambda)d\lambda}}_{= p(\lambda|w)} \end{aligned} \tag{16}$$

The conditional posterior of α is found exactly the same way, and the conditional posterior of λ is found in a similar way,

$$p(\alpha|y, w, x) = \text{Gamma}\Big(a + \frac{N}{2}, b + \frac{1}{2}\sum_{i=1}^{N}(y_i - x_i^T w)^2\Big), \quad p(\lambda|w) = \text{Gamma}\Big(e + \frac{d}{2}, f + \frac{1}{2}w^T w\Big)$$

M-Step

Using the same exact method as for the previous version, we can compute $\mathcal{L}(w)$ and maximize over w to find that

$$w_t = \Big(\mathbb{E}_{q_t}[\lambda]I + \mathbb{E}_{q_t}[\alpha]\sum_{i=1}^{N} x_i x_i^T\Big)^{-1}\Big(\sum_{i=1}^{N}\mathbb{E}_{q_t}[\alpha]y_i x_i\Big) \tag{17}$$

The difference is that we have expectations over both α and λ since these are the variables being marginalized.

- We notice that marginalizing two variables works out because we can still calculate their full conditional posteriors. In this case, the posterior has the form

$$q(\alpha, \lambda) = p(\alpha, \lambda | y, w, x) = p(\alpha | y, w, x) p(\lambda | w) \tag{18}$$

and so we say the posterior "factorizes." Therefore, q factorizes as well

$$q(\alpha, \lambda) = q(\alpha) q(\lambda) \tag{19}$$

where $q(\alpha) = p(\alpha | y, w, x)$ and $q(\lambda) = p(\lambda | w)$. When we calculate expectations over both q distributions, we can focus only on the relevant ones. That is,

$$\mathbb{E}_q[\alpha] = \int \int \alpha q(\alpha) q(\lambda) d\alpha \lambda = \int \alpha q(\alpha) d\alpha \tag{20}$$

Technically, during the E-step we are performing the first integral. However, because α and λ are *conditionally independent* given w, we simply integrate out λ to give the expectation of α restricted to its conditional posterior. *This is not the rule.* It is a by-product of the fact that $p(\alpha, \lambda | y, w, x)$ factorizes the way it does. In a way, this factorization of the (conditional) posterior makes life easier—something to keep in mind when we discuss variational inference next.

- As with the previous version, the output of this EM algorithm is a point estimate of w and conditional posteriors on the other variables. We've again made a compromise between the desired full posterior and a point estimate of everything.

- Before discussing how variational methods allow us to approximate the full posterior of all variables, it's worth quickly pointing out another way to approximate the full posterior distribution based on what we've done. This can further highlight how there is no single solution to approximate posterior inference (e.g., Laplace, MCMC, VI, etc.)

    - The output of this EM algorithm are conditional posterior distributions on α and λ and a MAP estimate of w. If we wanted to get some approximation of the posterior distribution of w, notice that we could do a Laplace approximation on $\ln p(y, w|x)$ at the MAP estimate. We now have this! Therefore, we only need to calculate the Hessian $\nabla^2_w \ln p(y, w|x)$ and evaluate it at w_{MAP} in order to get a Gaussian approximation of the posterior of w.

    - In this case, we will be approximating the posterior distribution

$$p(\alpha, \lambda, w | y, x) \approx q(\alpha) q(\lambda) q(w) \tag{21}$$

where

$$\begin{aligned} q(\alpha) = p(\alpha | y, x, w_{\text{MAP}}), \quad q(\lambda) = p(\lambda | w_{\text{MAP}}), \\ q(w) = \text{Normal}(w_{\text{MAP}}, (-\nabla^2_w \ln p(y, w_{\text{MAP}}, |x))^{-1}) \end{aligned} \tag{22}$$

Notice that to do this we will need to actually calculate $p(y, w|x)$, while we didn't need to in order to optimize it over w using EM.

    - This approach of approximating the full posterior with a factorized distribution over its variables will appear again as a major component of variational inference. In the above approach there doesn't appear to be a single, unified objective function that we optimize in order to get these q. We simply combine two techniques: Two of the q are found with EM and the last one with Laplace, and w_{MAP} is what connects them. Variational inference learns this factorized q using a single objective function that is closely related to EM.

From EM to variational inference

- Let's try to be clever. We were able to get conditional posteriors on α and λ by integrating them out and running EM. Can we add w to this? That is, what does it mean to do EM for the marginal distribution

$$p(y|x) = \int\int\int p(y, w, \alpha, \lambda) d\alpha d\lambda dw \tag{23}$$

  First, we notice that there aren't any free parameters in the marginal distribution on the left side. Still, this is a marginal distribution of *something* (the data), and so we can write

$$\begin{aligned} \ln p(y|x) &= \int\int\int q(\alpha, \lambda, w) \ln \frac{p(y, w, \alpha, \lambda|x)}{q(\alpha, \lambda, w)} d\alpha d\lambda dw \; + \\ & \int\int\int q(\alpha, \lambda, w) \ln \frac{q(\alpha, \lambda, w)}{p(\alpha, \lambda, w|y, x)} d\alpha d\lambda dw \end{aligned} \tag{24}$$

- However, this scenario feels different. There is nothing to optimize over the left hand side, so there is no M-step that can be performed. As for the E-step, we need to set

$$q(\alpha, \lambda, w) = p(\alpha, \lambda, w|y, x) \tag{25}$$

- Finally we can see the big problem. Trying to work through EM in this instance requires us to calculate the full posterior distribution of all model variables. This was the problem to begin with and so, even though there is no M-step to perform, we can't get to that point anyway because we can't complete the E-step. Before moving on, I think the following digression is worthwhile.

- Digression: A side comment relating EM to the original core model.

  This is purely a digression. At this point we've laid enough groundwork that we can do something potentially useful very fast. We saw with the core model,

$$y_i \overset{ind}{\sim} \text{Normal}(x_i^T w, \alpha^{-1}), \qquad w \sim \text{Normal}(0, \lambda^{-1} I)$$

  that we could calculate the posterior in closed form and that it was a Gaussian. For this model, we can also write the marginal likelihood $p(y|x) = \int p(y, w|x) dw$ and set up an EM-like equality,

$$\ln p(y|x) = \int q(w) \ln \frac{p(y, w|x)}{q(w)} dw + \int q(w) \ln \frac{q(w)}{p(w|y, x)} dw$$

  Again there is nothing to optimize over on the left hand side. However, we can solve for $q(w)$ in this case because the model is simple enough that we know $p(w|y, x)$. Therefore,

$$p(y|x) = \exp\Big\{ \int p(w|y, x) \ln \frac{p(y, w|x)}{p(w|y, x)} dw \Big\}$$

  Actually we didn't need EM to make this statement (Bayes rule is enough), but given the context of our discussion at this point, I thought I would mention it.

- Back to the main discussion: We have nothing to optimize in the marginal distribution $p(y|x)$ and we can't update $q(\alpha, \lambda, w)$ as EM tells us we must because we don't know the full posterior

distribution of these model variables (which was the problem to begin with). The question is whether the equation:

$$\ln p(y|x) = \int q(\alpha, \lambda, w) \ln \frac{p(y, w, \alpha, \lambda|x)}{q(\alpha, \lambda, w)} d\alpha d\lambda dw + \int q(\alpha, \lambda, w) \ln \frac{q(\alpha, \lambda, w)}{p(\alpha, \lambda, w|y, x)} d\alpha d\lambda dw$$

tells us anything interesting. The answer is yes, and it will lead to a new approximate posterior inference technique called variational inference (VI). I'll refer to this as the VI equation below.

- Before we discuss the three terms in this equation, remember how we set up the EM problem:
  - When we were arriving at this equation we simply said that q is *some* distribution on the model variable we wanted to integrate out in the marginal likelihood. That is, this equation is true for every q we can define, so long as it is defined on the correct space corresponding to what values the variables can take.
  - Then, because we wanted to maximize the LHS (back when it had something we could optimize over, that is), EM told us our *one option* for setting q in order to be able to modify our point estimate such that we can guarantee that we're monotonically increasing the marginal likelihood.

- However, in the VI equation scenario there is nothing to maximize on the LHS. Therefore, this strict requirement made by EM on what we can set q to equal seems irrelevant. To make the transition from EM to VI, we still find this equality to be useful. However, we shift our perspective: With EM we were focused on the LHS and on making sure we were always improving it via the RHS. With VI we are more interested in what is going on with the q distributions, and whether the freedom we have to pick q can be useful.

- In light of this new "VI perspective," let's break down the three terms in the VI equation, keeping in mind that we are not going to be so strict about the distribution $q(\alpha, \lambda, w)$ that we choose:
  - $\ln p(y|x)$: This is the marginal likelihood of the data given the model. We don't know what this is, and in fact it's because of this that we're discussing any of these things to being with. That's because the target posterior is

$$p(\alpha, \lambda, w|y, x) = \frac{p(y, \alpha, \lambda, w|x)}{p(y|x)} \tag{26}$$

    and since we can't solve the integral in the denominator to find $p(y|x)$, we have to use an inference algorithm. Therefore, we can't write out an analytic expression for $\ln p(y|x)$. However, we do know one crucial fact: $\ln p(y|x)$ *is constant*. That is, given the model definition and the data, there are no other degrees of freedom for $p(y|x)$, and so whatever value it takes, it's something that will never change no matter what we do on the RHS of the VI equation.
  - Next, consider the far right term. This is $\mathrm{KL}(q\|p)$ where q is some distribution we're going to pick. This is the KL-divergence between our chosen q and the full posterior. Again, we don't actually know what this is, so we can't give a number for KL in this case. However, we again know two crucial facts about the KL-divergence: *It is always non-negative and only equals zero when* $q = p$. Therefore, this term is a similarity measure between our chosen q distribution and the true, full posterior that we want.

- Finally there is the middle term. Fortunately, this is something we can calculate!—that is, provided that we pick a distribution for $q(\alpha, \lambda, w)$ that helps us toward this end. This is because we have defined the model, so we have defined how to write the function $p(y, w, \alpha, \lambda|x)$. We just need to calculate the integrals.

- Our goal is to pick a $q(\alpha, \lambda, w)$ that is close to the posterior distribution $p(\alpha, \lambda, w|y, x)$. We saw previously how the Laplace approximation tries to do this by letting q be a multivariate Gaussian. However, we didn't really have an objective function we were trying to optimize there. Rather we were solving a second order Taylor approximation problem.

- A natural measure of closeness between $q(\alpha, \lambda, w)$ and $p(\alpha, \lambda, w|y, x)$ is the KL-divergence. Since this measure is not symmetric (i.e., $\text{KL}(q\|p) \neq \text{KL}(p\|q)$), we have two options. Variational inference optimizes $\text{KL}(q\|p)$ using the VI equation, which we equivalently write as

$$\ln p(y|x) = \underbrace{\mathbb{E}_q[\ln p(y, w, \alpha, \lambda|x)] - \mathbb{E}_q[\ln q(\alpha, \lambda, w)]}_{\equiv\ \mathcal{L}\ \leftarrow\ \text{“variational objective function”}} \ +\ \text{KL}(q\|p(\alpha, \lambda, w|y, x))$$

- How can we optimize $\text{KL}(q\|p)$ using this equation? After all, we don't know two of the three terms, including the KL term we're now setting out to optimize. The key insight is that we don't need to know these terms: The LHS is constant and the KL divergence is non-negative. Therefore, $\mathcal{L} + \text{KL}$ must add up to the same number for every possible q that we define. By finding a q that maximizes $\mathcal{L}$, we are equivalently finding a q that minimizes the KL divergence between it and the target posterior distribution because $\text{KL} \geq 0$.

- Therefore, as mentioned, VI is purely interested in the q distribution. And in this case we have a good reason to interpret this q distribution as an approximation to the full posterior distribution.

Variational inference for the core model (version 3.0)

- This leads us to a variational inference algorithm for the last model we have been discussing. This entails the following steps:

  1. We need to define a q distribution *family* for α, λ and w. Compare this with EM in which we were told what to set q equal to.

  2. We then need to construct the *variational objective function*

  $$\mathcal{L} = \mathbb{E}_q[\ln p(y, w, \alpha, \lambda|x)] - \mathbb{E}_q[\ln q(\alpha, \lambda, w)] \tag{27}$$

  After doing this, w, α and λ will be gone since they're integrated out. All that will remain are the parameters of q.

  3. We define the distribution family, meaning we don't actually say what it's parameters are. How do we know what to set these too? Our goal is to maximize $\mathcal{L}$ over these parameters because the VI equation tells us that doing so will minimize the KL-divergence between q and the target posterior. Therefore, as its name implies, we treat $\mathcal{L}$ as an objective function to be maximizes over the parameters of q.

- In the EM algorithm, $\mathcal{L}$ was a function over a model variable—the one in the marginal likelihood we were doing MAP inference for. The parameters of the q distribution were always known because we set them to the conditional posterior distribution. Now, it's the parameters of q that are unknown and $\mathcal{L}$ is a function of these parameters. Therefore, even though it's the same $\mathcal{L}$ in EM and VI, $\mathcal{L}$ is a function of different things in these two methods. This is another subtle difference between VI and EM that really makes them two distinct inference techniques.

- And so it remains to define what $q(\alpha, \lambda, w)$ actually is. We need to pick a recognizable distribution so we can carry out the integral and optimize over its parameters. In all cases this entails some sort of simplification. By far the most common is the "mean-field" assumption (a physics term where these ideas originate from). In this approach we write

$$q(\alpha, \lambda, w) = q(\alpha)q(\lambda)q(w) \tag{28}$$

  and pick individual distributions for each variable. Why did we split across these three? Basically because these were the three variable "units" of the prior (notice that w is a vector, so we don't necessarily split across every variable, although we certainly could define $q(w) = \prod_j q(w_j)$).

- In the next lecture we will see how to find the optimal distribution family of each q. However, we are free to define them however we want and the algorithm will still work, and so we define them to be in the same family as the prior. (These also happen to be the optimal choices.)

$$q(\alpha) = \text{Gamma}(a', b'), \qquad q(\lambda) = \text{Gamma}(e', f'), \qquad q(w) = \text{Normal}(\mu', \Sigma') \tag{29}$$

  We just don't know what a', b', e', f', μ' and Σ' are. To find this we first calculate $\mathcal{L}$.

- Before doing that, why do we pick this q distribution and what is it doing?

  - We pick it, like how we pick many things, purely out of convenience. It's easy to define a distribution on a variable that is in the same family as the prior. By way of comparison, if we wanted to pick a 2-dimensional distribution $q(\alpha, \lambda)$ that wasn't factorizable, what could we pick? (What multivariate non-negative distributions are there readily at hand? The multivariate Gaussian isn't one of them since it is on all $\mathbb{R}^d$, not $\mathbb{R}^d_+$.) Likewise, how can we define a distribution, e.g., $q(\alpha, w)$? What distributions are there that we can easily write down where one dimension is in $\mathbb{R}_+$ and the others in $\mathbb{R}^d$? It's just easier to pick distributions individually for each variable.

  - The other consideration is that we need to be able to calculate $\mathcal{L}$. Picking complicated distributions on large sets of variables may produce an expectation (i.e., an integral) that is not solvable and then we're stuck with an objective we can't even write out, let alone easily optimize.

  - By making this factorization, we are assuming that all variables are independent *in the posterior*. In the core model version 2.0, we saw that α and λ are *conditionally* independent given w, which let us write $q(\alpha, \lambda) = q(\alpha)q(\lambda)$. In that case that was true, but now this is no longer true. That is, for every possible choice of the individual q distributions, we will always have

$$q(\alpha)q(\lambda)q(w) \neq p(\alpha, \lambda, w|y, x) \tag{30}$$

    Therefore, the KL-divergence will always be > 0.

- So now that we've defined q, we need to calculate $\mathcal{L}$. Next week we will see how we can update q in some situations without actually going through $\mathcal{L}$. That is, there is sometimes a trick that can be done where we can go straight to the parameter updates without calculating $\mathcal{L}$.

- However, the default is to calculate $\mathcal{L}$. Unfortunately this is not a pleasant task, but it's the most failsafe approach. The variational objective is

$$\begin{aligned}
\mathcal{L} &= \mathbb{E}_q[\ln p(y, w, \alpha, \lambda|x)] - \mathbb{E}_q[\ln q(\alpha, \lambda, w)] \\
&= -\frac{\mathbb{E}_q[\alpha]}{2}\sum_{i=1}^{N}\mathbb{E}_q[(y_i - x_i^T w)^2] + \frac{1}{2}\mathbb{E}_q[\ln \alpha] - \frac{\mathbb{E}_q[\lambda]}{2}\mathbb{E}_q[w^T w] + \frac{d}{2}\mathbb{E}_q[\ln \lambda] \\
&\quad + (a-1)\mathbb{E}_q[\ln \alpha] - b\mathbb{E}_q[\alpha] + (e-1)\mathbb{E}_q[\ln \lambda] - f\mathbb{E}_q[\lambda] + \text{constant} \\
&\quad - \mathbb{E}_q[\ln q(\alpha)] - \mathbb{E}_q[\ln q(\lambda)] - \mathbb{E}_q[\ln q(w)] \qquad (31)
\end{aligned}$$

- Notice that the last line is the sum of the entropies of each individual q distribution, which is a result of the factorization. At the very least we need to pick q so that we can calculate its entropy.

- The first two lines contain the expected log joint likelihood. Notice that a convenient result of the factorization is that q assumes all variables are independent. Therefore we have simplifications such as

$$\mathbb{E}_q\Big[\frac{\alpha}{2}\sum_{i=1}^{N}(y_i - x_i^T w)^2\Big] = \frac{\mathbb{E}_q[\alpha]}{2}\sum_{i=1}^{N}\mathbb{E}_q[(y_i - x_i^T w)^2] \qquad (32)$$

  And the expectations use only the part of q relevant to the variable being integrated over. This makes calculating $\mathcal{L}$ much easier.

- We will stop here for now. The function $\mathcal{L}$ will be nasty looking. However, after calculating it we can take derivatives and optimize over a', b', e', f', μ' and Σ'. It's not obvious at first sight, but after setting the respective derivatives to zero and solving, the final algorithm will actually be very simple and intuitively satisfying (in my opinion). We'll discuss this later.

- For now, the important take-home message is that by maximizing $\mathcal{L}$ over these six parameters, we are finding a *point estimate* of $q(\alpha)q(\lambda)q(w)$ such that it is an approximation of $p(\alpha, \lambda, w|y, x)$. That is, we get a point estimate of a probability distribution that approximates the posterior.

EECS E6720 Bayesian Models for Machine Learning

Columbia University, Fall 2016

Lecture 6, 10/20/2016

Instructor: John Paisley

Variational inference review (simple notation)

- For this fast review, we compress this into a simple notation. We have data X generated from a model with parameters θ. These parameters are themselves generated from a prior distribution (and so called "variables" from now on), giving the hierarchical representation

$$X \sim p(X|\theta), \quad \theta \sim p(\theta) \tag{1}$$

- We want the posterior distribution

$$p(\theta|X) = \frac{p(X|\theta)p(\theta)}{p(X)}, \tag{2}$$

but it's intractable. This can be because the prior and likelihood term are not conjugate. Just as common, the variables θ could actually be a large set of variables, the data X a complicated set of data, and the distributions $p(X|\theta)$ and $p(\theta)$ quite complicated in the dependency structure they induce on these variables.

- We approximate $p(\theta|X)$ with a distribution $q(\theta)$ and construct the equality

$$\underbrace{\ln p(X)}_{\text{constant}} = \underbrace{\int q(\theta) \ln \frac{p(X,\theta)}{q(\theta)} d\theta}_{\text{variational objective } \mathcal{L}} + \underbrace{\int q(\theta) \ln \frac{q(\theta)}{p(\theta|X)} d\theta}_{\text{KL-divergence} \geq 0} \tag{3}$$

- We've discussed how we can't actually compute the first and last terms in general, but that by maximizing $\mathcal{L}$ with respect to the parameters of $q(\theta)$, we are minimizing the KL-divergence between $q(\theta)$ and $p(\theta|X)$, which is like a distance measure between the two.

- Therefore, we can view $q(\theta)$ as an approximation of $p(\theta|X)$. The key thing is that we must be able to calculate

$$\mathcal{L} = \int q(\theta) \ln p(X,\theta) d\theta - \int q(\theta) \ln q(\theta) d\theta \tag{4}$$

Fortunately we often can do this since we define the joint likelihood $p(X,\theta)$ and the distribution $q(\theta)$.

- In fact, the above oversimplifies the problem somewhat. In reality, we have a set of parameters $\theta = (\theta_1, \ldots, \theta_m)$ which, along with their prior distributions, defines a joint likelihood on the data

$$p(X, \theta_1, \ldots, \theta_m) \tag{5}$$

  Notice that, as written, I haven't given you enough information to know how $p(X, \theta_1, \ldots, \theta_m)$ should factorize into a product of conditional distributions. The following discussion doesn't change at all based on this factorization, so I will just keep using $p(X, \theta_1, \ldots, \theta_m)$.

- Question: How should we pick $q(\theta_1, \ldots, \theta_m)$?
  Answer: Last week we discussed factorizing q using a "mean-field" assumption.

"Mean-field" assumption

- The "mean-field" assumption gets its name from physics, where these techniques were first developed in the context of a problem where this name makes sense. It consists of the following steps:

  1. Split θ into groups, usually (but not always) according to the "units" in which they are drawn from the prior. Here we've already assumed that to be $\theta_1, \ldots, \theta_m$. Notice that we could have $\theta_1 \in \mathbb{R}^d$ and $\theta_2 \in \mathbb{R}$, so it's not correct to think of each θ_i as being in the same space.

  2. Define $q(\theta_i|\psi_i)$ for variables θ_i. That is, q_i is the distribution family defined for θ_i and ψ_i is its parameters. Often you will just see "$q(\theta_i)$" for short, but for now we'll write it in this more complicated way to keep it more transparent.

  3. Let $q(\theta_1, \ldots, \theta_m) = \prod_{i=1}^m q(\theta_i|\psi_i)$ be the distribution we choose to approximate the posterior $p(\theta_1, \ldots, \theta_m|X)$

- Using this q distribution, the variational objective is then computed as

$$\mathcal{L} = \int \Big(\prod_{i=1}^m q(\theta_i|\psi_i) \Big) \ln p(X, \theta_1, \ldots, \theta_m) d\theta_1 \cdots d\theta_m - \sum_{i=1}^m \int q(\theta_i|\psi_i) \ln q(\theta_i|\psi_i) d\theta_i \tag{6}$$

- Assuming that we can calculate all of these integrals, the result is a function $\mathcal{L}(\psi_1, \ldots, \psi_m)$ that we try to maximize over all ψ_i.

Example (direct method)

- We will refer to the process of explicitly defining each q_i and calculating $\mathcal{L}$ as the "direct method." We show an example of this from the model discussed last week.

- We have data $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$ with $x \in \mathbb{R}^d$ and $y \in \mathbb{R}$ and the model

$$y_i \sim \text{Normal}(x_i^T w, \alpha^{-1}), \quad w \sim \text{Normal}(0, \lambda^{-1} I), \quad \alpha \sim \text{Gamma}(a, b) \tag{7}$$

- Based on the definition of this model, the joint likelihood factorizes as

$$p(y, w, \alpha|x) = p(\alpha)p(w)\prod_{i=1}^{N} p(y_i|x_i, w, \alpha) \tag{8}$$

- To approximate the full posterior, we pick the distribution

$$q(w, \alpha) = q(\alpha)q(w) = \text{Gamma}(\alpha|a', b')\text{Normal}(w|\mu', \Sigma') \tag{9}$$

- We the calculate the variational objective function

$$\begin{aligned}
\mathcal{L}(a', b', \mu', \Sigma') &= \int_0^\infty \int_{\mathbb{R}^d} q(w, \alpha) \ln \frac{p(y, w, \alpha|X)}{q(w, \alpha)} dw d\alpha \\
&= \int q(\alpha) \ln p(\alpha) d\alpha + \int q(w) \ln p(w) dw \\
&\quad + \sum_{i=1}^{N} \int \int q(\alpha)q(w) \ln p(y_i|x_i, w, \alpha) dw d\alpha \\
&\quad - \int q(\alpha) \ln q(\alpha) d\alpha - \int q(w) \ln q(w) dw
\end{aligned} \tag{10}$$

- When we write out the distributions involved, we see that this will require us to calculate $\mathbb{E}[\alpha]$, $\mathbb{E}[\ln \alpha]$, $\mathbb{E}[w]$ and $\mathbb{E}[ww^T]$. This can be looked up when necessary. If we were to actually solve these integrals, we would get something that looks like this:

$$\begin{aligned}
\mathcal{L}(a', b', \mu', \Sigma') &= (a-1)(\psi(a') - \ln b') - b\frac{a'}{b'} + \text{constant} \\
&\quad -\frac{\lambda}{2}(\mu'^T\mu' + \text{tr}(\Sigma')) + \text{constant} \\
&\quad +\frac{N}{2}(\psi(a') - \ln b') - \sum_{i=1}^{N} \frac{1}{2}\frac{a'}{b'}\Big((y_i - x_i^T\mu')^2 + x_i^T\Sigma' x_i\Big) + \text{constant} \\
&\quad +a' - \ln b' + \ln \Gamma(a') + (1-a')\psi(a') \\
&\quad +\frac{1}{2}\ln|\Sigma'| + \text{constant}
\end{aligned} \tag{11}$$

  The only reason this is being written out here is to give an idea of how complicated the variational objective function can look (remember how simple the model is). It's just here to make things 100% concrete, but you don't have to actually parse this if you don't want to. Of course in practice it might be unavoidable that you have to calculate at this level of detail.

- In this function, $\psi(\cdot)$ is a special function called the digamma function, and notice that the priors parameters are included without the $'$.

- Generally speaking, when optimizing $\mathcal{L}$ we want to update *all* parameters of a single q_i distribution at a time. Therefore, we want to update the pair (a', b') together and (μ', Σ') together.

- We'll discuss a faster method later, but the fail-safe method is to directly calculate $\mathcal{L}(a', b', \mu', \Sigma')$ and then solve "two equations, two unknowns" problem $\partial\mathcal{L}/\partial a' = 0$ and $\partial\mathcal{L}/\partial b' = 0$ for (a', b') and similarly for (μ', Σ').

Picking and then solving $q(\theta_i|\psi_i)$

- We return to the general problem where we have data X and model variables $\theta_1, \ldots, \theta_m$. After defining how $q(\theta_1, \ldots, \theta_m)$ factorizes, the two key problems are

  1. Picking the distribution family of $q(\theta_i|\psi_i)$ (e.g., Gaussian, gamma, etc.)
  2. Finding a way to update the parameters ψ_i to find the best $q(\theta_i|\psi_i)$ in its family

- Previously, we solved #1 arbitrarily and #2 through brute force calculation of $\mathcal{L}$. We have two natural questions that arise at this point:

  Q1: Is there a way to pick a *best* family for each q_i? (We'll use the shorthand $q_i \Leftrightarrow q(\theta_i|\psi_i)$.)

  Q2: Is there a faster way to solve for ψ_i that bypasses calculating $\mathcal{L}$?

  The answer to both of these questions is "yes" (at least in principle) and we can get the solutions to both questions at once! We will now show how to do this.

- General setup: Exactly as before, we have a joint likelihood $p(X, \theta_1, \ldots, \theta_m)$ and we approximate the posterior distribution $p(\theta_1, \ldots, \theta_m|X) \approx \prod_{i=1}^m q(\theta_i|\psi_i)$.

- Let's now focus on the q distribution for a specific variable, say $q(\theta_i|\psi_i)$. Our goal is to think about the variational objective function only in the context of this q_i distribution and see what it tells us,

$$\mathcal{L} = \int q(\theta_i|\psi_i) \underbrace{\left[\int \Big(\prod_{j \neq i} q(\theta_j|\psi_j) \Big) \ln p(X, \theta_1, \ldots, \theta_m) d\theta_{j \neq i} \right]}_{= \mathbb{E}_{q_{j \neq i}}[\ln p(X, \theta_1, \ldots, \theta_m)]} d\theta_i \tag{12}$$

$$- \int q(\theta_i|\psi_i) \ln q(\theta_i|\psi_i) d\theta_i - \sum_{j \neq i} \int q(\theta_j|\psi_j) \ln q(\theta_j|\psi_j) d\theta_j \tag{13}$$

- As indicated, the term $\mathbb{E}_{q_{j \neq i}}[\ln p(X, \theta_1, \ldots, \theta_m)]$ is the expectation over all θ_j for $j \neq i$. Therefore, the result of this expectation is a function of θ_i and of ψ_j for all $j \neq i$.

- We haven't defined any other q_j yet either, but we'll see that this doesn't matter for choosing the family of q_i.

- The next step is to manipulate how we write the variational objective function. First,

$$\begin{aligned} \mathcal{L} &= \int q(\theta_i|\psi_i) \mathbb{E}_{q_{j \neq i}}[\ln p(X, \theta_1, \ldots, \theta_m)] d\theta_i - \int q(\theta_i|\psi_i) \ln q(\theta_i|\psi_i) d\theta_i + \text{const. w.r.t. } \theta_i \\ &= \int q(\theta_i|\psi_i) \ln \frac{\mathrm{e}^{\mathbb{E}_{q_{j \neq i}}[\ln p(X, \theta_1, \ldots, \theta_m)]}}{q(\theta_i|\psi_i)} d\theta_i + \text{const. w.r.t. } \theta_i \end{aligned} \tag{14}$$

- Next we add and subtract $\ln Z$ and define

$$Z = \int \mathrm{e}^{\mathbb{E}_{q_{j \neq i}}[\ln p(X, \theta_1, \ldots, \theta_m)]} d\theta_i \tag{15}$$

This gives

$$\mathcal{L} = \int q(\theta_i|\psi_i) \ln \frac{\frac{1}{Z}\mathrm{e}^{\mathbb{E}_{q_{j\neq i}}[\ln p(X,\theta_1,\ldots,\theta_m)]}}{q(\theta_i|\psi_i)} d\theta_i + \ln Z + \text{const. w.r.t. } \theta_i \tag{16}$$

- What is $\frac{1}{Z}\mathrm{e}^{\mathbb{E}_{q_{j\neq i}}[\ln p(X,\theta_1,\ldots,\theta_m)]}$? Notice that we can think of it as a probability distribution on θ_i. As we will now see, this is very relevant, since our goal is to pick $q(\theta_i|\psi_i)$ such that, if we only focus on this term and ignore all other q_j, we maximize

$$\int q(\theta_i|\psi_i) \ln \frac{\frac{1}{Z}\mathrm{e}^{\mathbb{E}_{q_{j\neq i}}[\ln p(X,\theta_1,\ldots,\theta_m)]}}{q(\theta_i|\psi_i)} d\theta_i = -\mathrm{KL}\big(q_i \| \tfrac{1}{Z}\mathrm{e}^{\mathbb{E}_{q_{j\neq i}}[\ln p(X,\theta_1,\ldots,\theta_m)]}\big) \tag{17}$$

- Remember that $KL \geq 0$ and so $-\mathrm{KL} \leq 0$. We know when KL is minimized, so we equivalently know when the *negative* KL is maximized, that is, we now know we should set

$$q(\theta_i|\psi_i) = \frac{1}{Z}\mathrm{e}^{\mathbb{E}_{q_{j\neq i}}[\ln p(X,\theta_1,\ldots,\theta_m)]} \tag{18}$$

- Notice a few things about this:

  1. This gives the optimal family for $q(\theta_i|\psi_i)$. It doesn't matter what the other q_j are or their ψ_j parameters.

  2. This also gives the optimal parameters ψ_i of $q(\theta_i|\psi_i)$ for given settings of the other q_j. If we know all other q_j except for q_i, there's nothing unknown in this optimal $q(\theta_i|\psi_i)$. Put another way, we haven't just written out a distribution family above, we've written out a specific distribution including all its parameters.

  3. If we don't know what the family of the other q_j are (e.g., Gaussian, gamma, etc.) we still can say the *family* of $q(\theta_i|\psi_i)$. That is, the above equation will allow us to say something like "q_i should be defined to be a gamma distribution."

  4. Since our choice of i was arbitrary, we therefore know what *every* q_i should be set to by looking at the *form* of $\exp\{\mathbb{E}_{q_{j\neq i}}[\ln p(X, \theta_1, \ldots, \theta_m)]\}$ for each i. Therefore, this is the first step in variational inference: To iterate between each θ_i and decide what q_i distribution should be defined for it.

  5. We can write this abstractly, but if the integral is intractable (in the numerator or the denominator), then this does us no good. Does this work in general? The answer is "yes," for a large class of models (to be discussed in a later lecture).

- In words, this says that to find the optimal q distribution for θ_i:

  1. Take the log of the joint likelihood

  2. Take the expectation of all variables using their respective q distribution except for $q(\theta_i|\psi_i)$

  3. Exponentiate the result and normalize

General variational inference algorithm

- Given data X and a joint likelihood $p(X, \theta_1, \ldots, \theta_m)$
  - For iteration $t = 1, 2, \ldots$
    1. For model variable index $i = 1, \ldots, m$ set
$$q(\theta_i|\psi_i) = \frac{\mathrm{e}^{\mathbb{E}_{q_{j\neq i}}[\ln p(X,\theta_1,\ldots,\theta_m)]}}{\int \mathrm{e}^{\mathbb{E}_{q_{j\neq i}}[\ln p(X,\theta_1,\ldots,\theta_m)]} d\theta_i}$$
    2. Evaluate the variational objective function using the updated q
$$\mathcal{L}_t = \mathbb{E}_q[\ln p(X, \theta_1, \ldots, \theta_m)] - \sum_{i=1}^{m} \mathbb{E}_{q_i}[\ln q(\theta_i|\psi_i)]$$
    3. If the marginal increase in $\mathcal{L}_t$ compared with $\mathcal{L}_{t-1}$ is "small," terminate, otherwise continue to the next iteration.

- As with Gibbs sampling, we always use the most recent parameters for all $q(\theta_j|\psi_j)$ when updating q_i. Let's look at a concrete example.

Example (optimal method)

- We return to the original example
$$y_i \sim \text{Normal}(x_i^T w, \alpha^{-1}), \quad w \sim \text{Normal}(0, \lambda^{-1} I), \quad \alpha \sim \text{Gamma}(a, b) \tag{19}$$

- We want to approximate the posterior $p(w, \alpha|y, x)$ with $q(\alpha, w)$ using variational inference and a mean-field assumption. Therefore, the first step is to choose the factorization of q. Again we choose
$$p(\alpha, w|y, x) \approx q(\alpha, w) \equiv q(\alpha)q(w)$$

- Next, we want to learn what distribution family we should set $q(\alpha)$ and $q(w)$ to be. For example, should $q(\alpha)$ be Gaussian? Should it be gamma? Poisson? etc.

- $\boldsymbol{q(\alpha)}$: We know from the general approach that we can find $q(\alpha)$ as follows:
$$\begin{aligned} q(\alpha) &\propto \exp\left\{\mathbb{E}_{q(w)}[\ln p(y|x, \alpha, w) + \ln p(\alpha) + \ln p(w)]\right\} \\ &\propto \exp\left\{\mathbb{E}_{q(w)}[\ln p(y|x, \alpha, w)] + \ln p(\alpha)\right\} \end{aligned} \tag{20}$$

  Notice that we can remove any terms not involving α, therefore $\ln p(w)$ is removed. Also, the expectation is only over w, therefore the expectation doesn't impact $\ln p(\alpha)$. Then,
$$\begin{aligned} q(\alpha) &\propto \exp\left\{\sum_{i=1}^{N} \mathbb{E}_{q(w)}[\ln p(y_i|x_i, \alpha, w)]\right\} p(\alpha) \\ &\propto \left[\prod_{i=1}^{N} \alpha^{\frac{1}{2}} \mathrm{e}^{-\frac{\alpha}{2}\mathbb{E}_{q(w)}[(y_i - x_i^T w)^2]}\right] \alpha^{a-1} \mathrm{e}^{-b\alpha} \end{aligned} \tag{21}$$

- Since we haven't defined $q(w)$ yet, we can't take this expectation. However, notice that we have enough information to be able to say that

$$q(\alpha) = \text{Gamma}(\alpha|a', b'), \quad a' = a + \frac{N}{2}, \quad b' = b + \frac{1}{2}\sum_{i=1}^{N} \mathbb{E}_{q(w)}[(y_i - x_i^T w)^2] \tag{22}$$

- $\boldsymbol{q(w)}$: Next, we perform similar operations to find the optimal $q(w)$,

$$\begin{aligned} q(w) &\propto \exp\left\{\mathbb{E}_{q(\alpha)}[\ln p(y|x, \alpha, w) + \ln p(\alpha) + \ln p(w)]\right\} \\ &\propto \exp\left\{\mathbb{E}_{q(\alpha)}[\ln p(y|x, \alpha, w)] + \ln p(w)\right\} \end{aligned} \tag{23}$$

  Again we can remove any terms not involving w, and so $\ln p(\alpha)$ is removed. Also, since the expectation is only over α, we don't have an expectation for the term $\ln p(w)$. Again, we continue:

$$\begin{aligned} q(w) &\propto \exp\left\{\sum_{i=1}^{N} \mathbb{E}_{q(\alpha)}[\ln p(y_i|x_i, \alpha, w)]\right\} p(w) \\ &\propto \Big[\prod_{i=1}^{N} \mathrm{e}^{\frac{1}{2}\mathbb{E}[\ln \alpha]} \mathrm{e}^{-(\mathbb{E}_{q(\alpha)}[\alpha]/2)(y_i - x_i^T w)^2}\Big] \mathrm{e}^{-\frac{\lambda}{2} w^T w} \end{aligned} \tag{24}$$

  First notice that we can simply ignore $\mathrm{e}^{\frac{1}{2}\mathbb{E}[\ln \alpha]}$ because it will be canceled out when we compute the normalizing constant.

- We already saw this type of proportionality before when we calculated the posterior of w in the Bayesian linear regression problem. As a result, we know that

$$q(w) = \text{Normal}(w|\mu', \Sigma') \tag{25}$$

$$\Sigma' = \Big(\lambda I + \mathbb{E}_{q(\alpha)}[\alpha] \sum_{i=1}^{N} x_i x_i^T\Big)^{-1}, \qquad \mu' = \Sigma'\Big(\mathbb{E}_{q(\alpha)}[\alpha] \sum_{i=1}^{N} y_i x_i\Big)$$

- Notice that by cycling through each parameter and learning its q distribution, we also learn what the expectations are when learning other q distributions. That is, when we found that $q(\alpha)$ was a gamma distribution, we weren't able to say what the expectation with respect to $q(w)$ was because we didn't know what distribution to use for $q(w)$. Now we know it's Gaussian and so we can retroactively solve for the expectation.

- Similarly, because we first found that $q(\alpha)$ was a gamma distribution, we know what the expectation is that we should use when we update $q(w)$. This is the general pattern. We first find what the distributions are for each q_i, and after we have them all, we will know what all the expectations will be.

- For example, for this problem

$$\mathbb{E}_{q(\alpha)}[\alpha] = a'/b' \tag{26}$$

$$\mathbb{E}_{q(w)}[(y_i - x_i^T w)^2] = (y_i - x_i^T \mu')^2 + x_i^T \Sigma' x_i \tag{27}$$

- Let's look at the final variational inference algorithm for this problem.

VI algorithm for Bayesian linear regression with unknown noise precision

Inputs: Data and definitions $q(\alpha) = \text{Gamma}(\alpha|a', b')$ and $q(w) = \text{Normal}(w|\mu', \Sigma')$

Output: Values for a', b', μ' and Σ'

1. Initialize a'_0, b'_0, μ'_0 and Σ'_0 in some way
2. For iteration $t = 1, \ldots, T$
   - Update $q(\alpha)$ by setting

$$\begin{aligned} a'_t &= a + \frac{N}{2} \\ b'_t &= b + \frac{1}{2}\sum_{i=1}^{N}(y_i - x_i^T\mu'_{t-1})^2 + x_i^T\Sigma'_{t-1}x_i \end{aligned}$$

   - Update $q(w)$ by setting

$$\begin{aligned} \Sigma'_t &= \Big(\lambda I + \frac{a'_t}{b'_t}\sum_{i=1}^{N} x_i x_i^T\Big)^{-1} \\ \mu'_t &= \Sigma'_t\Big(\frac{a'_t}{b'_t}\sum_{i=1}^{N} y_i x_i\Big) \end{aligned}$$

   - Evaluate $\mathcal{L}(a'_t, b'_t, \mu'_t, \Sigma'_t)$ to assess convergence (i.e., decide T).

- Notice that this is exactly the solution we would have found if we solved the system of equations

$$\frac{\partial\mathcal{L}(a', b', \mu'_{t-1}, \Sigma'_{t-1})}{\partial a'} = 0, \quad \frac{\partial\mathcal{L}(a', b', \mu'_{t-1}, \Sigma'_{t-1})}{\partial b'} = 0$$

  to find a'_t and b'_t. We also would find this update for μ'_t and Σ'_t by solving the system

$$\nabla_{\mu'}\mathcal{L}(a'_t, b'_t, \mu', \Sigma') = 0, \quad \nabla_{\Sigma'}\mathcal{L}(a'_t, b'_t, \mu', \Sigma') = 0$$

- You can also see the difference between VI for this model and EM for maximizing $\ln p(y, w|x)$ using α as the marginalized variable, which we discussed earlier. In EM, the update for the point estimate is $w_t = \mu'_t$ using μ'_t from above. When we wanted to update $q(\alpha)$, we again had a gamma distribution, but because there was no distribution on w, we simply plugged in w_t where μ'_t appears and remove Σ'_t (again, because there is no uncertainty about w).

- Comment: Sometimes researchers will define a q "distribution" that is a point mass – that is, they will let $q(w) = \delta_{w'}$ for example. For this distribution, $P(w = w') = 1$. When we "integrate" using this $q(w)$ to calculate $\mathcal{L}$, we find that we simply replace w with w' and do a point estimate of w' (and ignore the technical issue that the entropy of $\delta_{w'}$ is $-\infty$). This is sometimes done when the integral using any other $q(w)$ is intractable, which is not the case here.

VI algorithm for probit regression

- Recall the setup (slightly modified): We have data $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$ where $x \in \mathbb{R}^d$ and $y \in \{-1, +1\}$ (previously $\{0, 1\}$, but notice that it's no difference below). Including a hidden variable ϕ_i for each (x_i, y_i) pair, the probit regression model is

$$y_i = \text{sign}(\phi_i), \quad \phi_i \sim \text{Normal}(x_i^T w, \sigma^2), \quad w \sim \text{Normal}(0, \lambda^{-1} I) \tag{28}$$

- Previously, we derived an EM algorithm for maximizing the marginal distribution $p(y, w|x)$ under this model. Let's now look at a variational inference algorithm.

- The unknowns this time are w and the vector $\phi = (\phi_1, \ldots, \phi_N)$. We set up the variational inference equation

$$\ln p(y|x) = \int q(w, \phi) \ln \frac{p(y, w, \phi|x)}{q(w, \phi)} dw d\phi + \int q(w, \phi) \ln \frac{q(w, \phi)}{p(w, \phi|y, x)} dw d\phi \tag{29}$$

- From this setup, we can see that we're designing $q(w, \phi)$ to approximate the full posterior of both w and the extra variables ϕ together. We pick the factorization

$$q(w, \phi) = q(w) \prod_{i=1}^N q(\phi_i) \tag{30}$$

The first problem using our "optimal approach" is to find out what these distributions should be. First, notice that the joint likelihood factorizes as

$$p(y, w, \phi|x) = p(w) \prod_{i=1}^N p(y_i|\phi_i) p(\phi_i|w, x_i) \tag{31}$$

The "distribution" $p(y_i|\phi_i)$ has no randomness in it: $p(y_i|\phi_i) = \mathbb{1}\{y_i = \text{sign}(\phi_i)\}$.

- $\boldsymbol{q(\phi_i)}$ **:** Following the rules, we take the log of $p(y, w, \phi|x)$ and the expectation with respect to $q(w)$ and $q(\phi_j)$ for $j \neq i$, and then exponentiate. As a result, we can write that

$$\begin{aligned} q(\phi_i) \quad \propto \quad & \exp\left\{ \mathbb{E}_{q(w)}[\ln p(w)] + \sum_{j \neq i} \mathbb{E}_{q(\phi_j)}[\ln p(y_j|\phi_j) + \ln p(\phi_j|w, x_j)] \right\} \times \\ & \exp\left\{ \ln p(y_i|\phi_i) + \mathbb{E}_{q(w)}[\ln p(\phi_i|w, x_i) \right\} \end{aligned} \tag{32}$$

Notice that the first line doesn't contain anything having to do with ϕ_i, therefore it's a constant as far as the proportionality is concerned and can be ignored. As a result,

$$\begin{aligned} q(\phi_i) \quad & \propto \quad \mathbb{1}\{y_i = \text{sign}(\phi_i)\} \exp\{-\tfrac{1}{2\sigma^2} \mathbb{E}_{q(w)}[(\phi_i - x_i^T w)^2]\} \\ & \propto \quad \mathbb{1}\{y_i = \text{sign}(\phi_i)\} \exp\{-\tfrac{1}{2\sigma^2} [(\phi_i - x_i^T \mathbb{E}_{q(w)}[w])^2 + x_i^T \mathbb{E}_{q(w)}[w w^T] x_i]\} \\ & \propto \quad \mathbb{1}\{y_i = \text{sign}(\phi_i)\} \exp\{-\tfrac{1}{2\sigma^2} (\phi_i - x_i^T \mathbb{E}_{q(w)}[w])^2\} \exp\{-\tfrac{1}{2\sigma^2} x_i^T \mathbb{E}_{q(w)}[w w^T] x_i\} \\ & \propto \quad \mathbb{1}\{y_i = \text{sign}(\phi_i)\} \exp\{-\tfrac{1}{2\sigma^2} (\phi_i - x_i^T \mathbb{E}_{q(w)}[w])^2\} \end{aligned} \tag{33}$$

This is a truncated normal on the half of $\mathbb{R}$ defined by y_i. Therefore, the optimal $q(\phi_i)$ is

$$q(\phi_i) = \text{TN}_{y_i}(\mu'_{\phi_i}, \sigma^2), \qquad \mu'_{\phi_i} = x_i^T \mathbb{E}_{q(w)}[w] \tag{34}$$

- $\boldsymbol{q(w)}$ **:** We use the same approach to find the optimal q distribution of w,

$$\begin{aligned} q(w) &\propto \exp\Big\{ \ln p(w) + \sum_{i=1}^{N} \mathbb{E}_{q(\phi_i)}[\ln p(y_j|\phi_j) + \ln p(\phi_j|w, x_j)] \Big\} \\ &\propto \exp\Big\{ \ln p(w) + \sum_{i=1}^{N} \mathbb{E}_{q(\phi_i)}[\ln p(\phi_j|w, x_j)] \Big\} \\ &\propto \mathrm{e}^{-\frac{\lambda}{2} w^T w} \prod_{i=1}^{N} \mathrm{e}^{-\frac{1}{2\sigma^2}(\mathbb{E}_{q(\phi_i)}[\phi_i] - x_i^T w)^2} \end{aligned} \tag{35}$$

  This is exactly the Bayesian linear regression problem we have discussed. The difference is that because we don't observe ϕ_i, we end up using the expectation of this latent variable using its $q(\phi_i)$ distribution. As a result,

$$q(w) = \text{Normal}(\mu', \Sigma') \tag{36}$$

$$\Sigma' = \Big(\lambda I + \frac{1}{\sigma^2} \sum_{i=1}^{N} x_i x_i^T\Big)^{-1}, \qquad \mu' = \Sigma' \Big(\frac{1}{\sigma^2} \sum_{i=1}^{N} \mathbb{E}_{q(\phi_i)}[\phi_i] x_i\Big)$$

- Now that we've defined the q distributions, we can write out the relevant expectations,

$$\mathbb{E}_q[\phi_i] = \begin{cases} x_i^T \mu'_{\phi_i} + \sigma \times \dfrac{\Phi'(-x_i^T \mu'_{\phi_i}/\sigma)}{1 - \Phi(-x_i^T \mu'_{\phi_i}/\sigma)} & \text{if } \; y_i = +1 \\ x_i^T \mu'_{\phi_i} + \sigma \times \dfrac{-\Phi'(-x_i^T \mu'_{\phi_i}/\sigma)}{\Phi(-x_i^T \mu'_{\phi_i}/\sigma)} & \text{if } \; y_i = -1 \end{cases}$$

$$\mathbb{E}_{q(w)}[w] = \mu'$$

- As before, Φ is the CDF of a standard normal and Φ' is its PDF.

- This gives the following two steps that can be iterated,
  - At iteration t
    1. Update μ'_{ϕ_i} for each i using the μ' from iteration $t-1$ as in Equation (34)
    2. Update μ' and Σ' using all μ'_{ϕ_i} just updated in Step 1 as in Equation (36)
  - Assess convergence by evaluating the variational objective $\mathcal{L}(\mu', \Sigma', \mu'_{\phi_1}, \ldots, \mu'_{\phi_N})$ using the values of these variational parameters from iteration t.

- As a (painful) exercise, you can try deriving this variational inference algorithm using the "direct method" of first calculating $\mathcal{L}$ and then taking derivatives, setting to zero and solving. This will make it very easy to appreciate how much easier this optimal approach is. Also, if we weren't able to find the optimal q distributions first, and instead defined some other distribution for $q(\phi_i)$, the direct method most likely would lead to a dead end.

EECS E6720 Bayesian Models for Machine Learning

Columbia University, Fall 2016

Lecture 7, 10/27/2016

Instructor: John Paisley

- Let's look at another example of a standard model that is easily learned with variational inference.

Latent Dirichlet allocation (LDA)

- LDA is a Bayesian approach to topic modeling and one of the fundamental models in machine learning. It is popular because it has a wide range of applications and is easy to develop and build into larger systems.

  Setup: We have discrete grouped data, $x_d = \{x_{d1}, \ldots, x_{d,N_d}\}$, where d indexes group number. Each $x_{di} \in \{1, \ldots, V\}$, meaning each observation takes one of V values and x_d is a particular collection of values disjoint from all other $x_{d'}$.

  Example: The classic example where LDA is used is in document modeling. We'll assume we're working in that context from now on.

  - In this case, d would index a particular document (e.g., a blog post, an article in a newspaper, etc.) and V would be the size of the vocabulary of words.
  - The value x_{di} is the index of the ith word in the dth document. For example, $x_{di} = 7241$ might mean that the ith word in document d is "government," meaning that the number 7241 maps to the word "government" in a vocabulary that we have constructed.
  - When we say the "ith word in the dth document," this doesn't have to refer to the literal order. LDA doesn't model word order, so any re-ordering (and thus re-indexing) of the words will be viewed in exactly the same way by LDA.
  - Also, as an aside, there is significant pre-processing prior to getting each x_d. The vocabulary size V and the words comprising that vocabulary is selected in advance, with overly common words like "the" removed and words that are very rare also removed. This pre-processing will have an impact on performance, but we will assume that this task has already been done.

Topic modeling: The idea behind topic modeling is to model the D documents $x_1, \ldots, x_D$ as being generated from a set of K "topics," $\beta_1, \ldots, \beta_K$. Every document shares these topics, but has its own document-specific model variable that dictates how they are used.

- β_k : a V-dimensional probability distribution on the V words for topic k
- θ_d : a K-dimensional probability distribution on the topics for document d

Model: Given the topics, $\beta_1, \ldots, \beta_K$ and the distribution θ_d on them for document d, generate all data in document d as follows:

$$c_{di} \sim \text{Discrete}(\theta_d), \quad x_{di} \sim \text{Discrete}(\beta_{c_{di}}). \tag{1}$$

Notice that we have introduced an additional latent variable to the model:

- c_{di} picks out the topic that the ith word in document d belongs to. Notice that if $x_{di} = x_{di'}$ (i.e., the same word appears multiple times in a document) it's not necessarily the case that $c_{di} = c_{di'}$. The same word can have significant probability in multiple topics.
- x_{di} is generated using the topic indexed by c_{di}, and hence the subscript on β.

Priors: We don't know θ_d or β_k (or c_{di} for that matter, but we know it's distribution given θ_d). Therefore, we need to put prior distributions on them. There are many distributions we could pick. LDA uses the following:

$$\theta_d \overset{iid}{\sim} \text{Dirichlet}(\alpha), \quad \beta_k \overset{iid}{\sim} \text{Dirichlet}(\gamma) \tag{2}$$

Since θ_d and β_k are all finite probability vectors used in a discrete (or from another perspective, multinomial) distribution, LDA uses a conditionally conjugate Dirichlet prior for each of them. This makes variational inference very easy.

A prior discussion on the posterior: Before we get into the variational inference algorithm for LDA, what do we hope to find? What will β_k and θ_d tell us that's useful?

- β_k : If we look at the approximate posterior distribution of β_k and see which dimensions of it are expected to be large, then the high probability words should all relate to a coherent theme. For example, the three most probable words might be "government," "politics," and "congress." We would then call this a "politics" topic. In papers on topic modeling, you will often see lists of words. This is exactly what is being done: Each list is showing the 5 or 10 most probable words (dimensions) according to a specific topic (β_k).
- θ_d : This will tell us the fraction of each topic appearing in a particular document. Because we can assign meaning to each β_k after the fact, we can then say, e.g., "this document is 75% politics and 25% technology," etc., because $\theta_{d,1} = 0.75$ and β_1 is the "politics" topic.
- In general, both β_k and θ_d will be highly sparse, meaning only a fraction of values will be significantly nonzero.

Posterior calculation: We use Bayes rule to try to calculate the posterior,

$$\begin{aligned} p(\beta, \theta, c|x) &\propto p(x|\beta, \theta, c)p(\beta, \theta, c) \\ &\propto p(x|\beta, \theta, c)p(c|\beta, \theta)p(\beta, \theta) \\ &\propto p(x|\beta, c)p(c|\theta)p(\beta)p(\theta) \end{aligned} \tag{3}$$

The first two lines are simply true statements about Bayes rule and how probabilities factorize. The last line takes into consideration the dependency structure of LDA to remove unnecessary conditioning. By the assumed independence structure, this further breaks down as follows:

$$p(\beta, \theta, c|x) \propto \Big[\prod_{d=1}^{D}\prod_{i=1}^{N_d} p(x_{di}|\beta, c_{di})p(c_{di}|\theta_d)\Big]\Big[\prod_{k=1}^{K} p(\beta_k)\Big]\Big[\prod_{d=1}^{D} p(\theta_d)\Big] \tag{4}$$

- Not surprisingly, this can't be normalized and so we need an inference algorithm to approximate the posterior. Notice that there are quite a few variables we want to learn with this model. For example, for each word x_{di} in the data set, there is an associated topic indicator c_{di} that we need to learn. Therefore the number of variables is much larger than what we've discussed before.

Variational inference for LDA

- We will use our previous discussion of the "optimal method" for variational inference to approximate the posterior of the LDA model.

- Step 1: Using the mean-field assumption, we need to pick a factorization of $q(\beta, \theta, c) \approx p(\beta, \theta, c|x)$. We split these variables according to how they are generated in the prior. This makes learning q much easier.

$$q(\beta, \theta, c) = \Big[\prod_{k=1}^{K} q(\beta_k)\Big]\Big[\prod_{d=1}^{D} q(\theta_d)\Big]\Big[\prod_{d=1}^{D}\prod_{i=1}^{N_d} q(c_{di})\Big] \tag{5}$$

  Step 2: Next we need to select the distribution family for each q. For this model we will be able to find the optimal distributions. Remember from our previous discussion that, for a given variable, we can find the optimal q distribution as follows:

  1. Take the log of the complete joint likelihood
  2. Take the expectation of this using all other q distributions except the one of interest
  3. Exponentiate the result and normalize over the variable of interest

- The potential concern with this was that we don't know any of the q distributions to begin with, so when we take the expectation with respect to "all other" q, we don't know what these expectations are! However, recall that we don't need to know the actual values of the expectations in order to find the family of the q distribution being considered. Therefore, by completing one loop of this procedure we reach the point where we can go back and calculate the expectations that we left undefined previously.

- However, before we can do this, we need to know how to write the joint likelihood for LDA.

Joint likelihood of LDA

- The posterior $p(\beta, \theta, c|x) \propto p(x, \beta, \theta, c)$, which is what we calculated before,

$$p(x, \beta, \theta, c) = \Big[\prod_{d=1}^{D}\prod_{i=1}^{N_d} p(x_{di}|\beta, c_{di})p(c_{di}|\theta_d)\Big]\Big[\prod_{k=1}^{K} p(\beta_k)\Big]\Big[\prod_{d=1}^{D} p(\theta_d)\Big] \tag{6}$$

- These probabilities are all easy to write, but we run into a problem with the term

$$p(x_{di}|\beta, c_{di}) = \prod_{v=1}^{V} \beta_{c_{di}}(v)^{\mathbb{1}(x_{di}=v)} \tag{7}$$

  Notice that this function picks out the correct dimension of β according to the value that x_{di} takes. However, it's hard to do variational inference for c_{di} as written.

- This is another example where notation can help make things easier to derive. Notice that

$$p(x_{di}|\beta, c_{di}) = \prod_{v=1}^{V} \beta_{c_{di}}(v)^{\mathbb{1}(x_{di}=v)} = \prod_{k=1}^{K}\Big[\prod_{v=1}^{V} \beta_k(v)^{\mathbb{1}(x_{di}=v)}\Big]^{\mathbb{1}(c_{di}=k)} \tag{8}$$

  In both cases, c_{di} picks out the correct topic vector β_k, while x_{di} picks out the correct dimension of the selected vector.

- To keep the notation clean, we write

$$p(x, \beta, \theta, c) = \Big[\prod_{d=1}^{D}\prod_{i=1}^{N_d}\prod_{k=1}^{K} \big(p(x_{di}|\beta_k)\theta_{dk}\big)^{\mathbb{1}(c_{di}=k)}\Big]\Big[\prod_{k=1}^{K} p(\beta_k)\Big]\Big[\prod_{d=1}^{D} p(\theta_d)\Big] \tag{9}$$

  Notice that we directly use $\theta_{dk} = p(c_{di} = k|\theta_d)$, but leave the rest in $p(\cdot)$ notation.

- Therefore, the log of the joint likelihood can be written as

$$\begin{aligned}\ln p(x, \beta, \theta, c) \;\; = \;\; & \sum_{d=1}^{D}\sum_{i=1}^{N_d}\sum_{k=1}^{K} \mathbb{1}(c_{di} = k) \ln p(x_{di}|\beta_k) + \mathbb{1}(c_{di} = k) \ln \theta_{dk} \\ & + \sum_{k=1}^{K} \ln p(\beta_k) + \sum_{d=1}^{D} \ln p(\theta_d)\end{aligned} \tag{10}$$

- The log joint likelihood is a key equation and should never be out of sight when deriving variational inference algorithms.

- As a general rule, when we want to find a q distribution, we can throw away anything not involving the variable being updated. This will make working with the log joint likelihood less intimidating and require much less writing.

- To emphasize this point, let's look at a made-up toy example.

  Toy example

- Imagine a model with joint likelihood $p(x, c, a, b)$ where x is the data and c, a, b are model variables. Then using the factorization

$$q(c, a, b) = q(c)q(a)q(b)$$

  we have that

$$q(c) \propto \mathrm{e}^{\mathbb{E}_{-q}[\ln p(x,c,a,b)]}$$

  We use the shorthand notation "$-q$" to indicate all q distributions except for the one being considered at the moment.

- What if $\ln p(x, c, a, b) = ax + bc + xc^2 + ab$? This is a completely made up log joint likelihood and doesn't actually correspond to anything. However, this is just to highlight the point. In this case,

$$\begin{aligned} q(c) &\propto \mathrm{e}^{\mathbb{E}[a]x+\mathbb{E}[b]c+xc^2+\mathbb{E}[a]\mathbb{E}[b]} \\ &\propto \mathrm{e}^{\mathbb{E}[a]x+\mathbb{E}[a]\mathbb{E}[b]}\mathrm{e}^{\mathbb{E}[b]c+xc^2} \\ &\propto \mathrm{e}^{\mathbb{E}[b]c+xc^2} \end{aligned}$$

- This last line is because

$$q(c) = \frac{\mathrm{e}^{\mathbb{E}[a]x+\mathbb{E}[a]\mathbb{E}[b]}\mathrm{e}^{\mathbb{E}[b]c+xc^2}}{\int \mathrm{e}^{\mathbb{E}[a]x+\mathbb{E}[a]\mathbb{E}[b]}\mathrm{e}^{\mathbb{E}[b]c+xc^2}dc} = \frac{\mathrm{e}^{\mathbb{E}[b]c+xc^2}}{\int \mathrm{e}^{\mathbb{E}[b]c+xc^2}dc}$$

- As a side comment, we can write $\mathbb{E}[ab] = \mathbb{E}[a]\mathbb{E}[b]$ because the expectation uses $q(a, b) = q(a)q(b)$, so they're independent.

- The take-home message here is that, when updating a q distribution for a particular model variable, we only need to look at the terms in the log joint likelihood that involve this variable.

- The log joint likelihood will be the sum of many things, and anything not involving this variable can be absorbed in the normalizing constant and so ignored. For this model, that means we can simply ignore $ax + ab$ in the log joint likelihood when we find $q(c)$.

- This is just a toy example to drive home a point. But for complicated models, this way of simplifying things can make deriving the VI algorithm seem like a much less daunting task.

- For quick reference, in deriving the VI algorithm for LDA, we doing this with the joint likelihood

$$\begin{aligned} \ln p(x, \beta, \theta, c) &= \sum_{d=1}^{D}\sum_{i=1}^{N_d}\sum_{k=1}^{K} \mathbb{1}(c_{di} = k) \ln p(x_{di}|\beta_k) + \mathbb{1}(c_{di} = k) \ln \theta_{dk} \\ &\quad + \sum_{k=1}^{K} \ln p(\beta_k) + \sum_{d=1}^{D} \ln p(\theta_d) \end{aligned} \tag{11}$$

$q(c_{di})$: Indicator of which topic word x_{di} came from

- To find this q distribution, we can focus only on terms in the log joint likelihood involving c_{di}. Therefore,

$$\begin{aligned} q(c_{di}) &\propto \mathrm{e}^{\sum_{k=1}^{K} \mathbb{1}(c_{di}=k)\left(\mathbb{E}_{-q}[\ln p(x_{di}|\beta_k)]+\mathbb{E}_{-q}[\ln \theta_{dk}]\right)} \\ &\propto \prod_{k=1}^{K}\left[\mathrm{e}^{\mathbb{E}_{-q}[\ln p(x_{di}|\beta_k)]+\mathbb{E}_{-q}[\ln \theta_{dk}]}\right]^{\mathbb{1}(c_{di}=k)} \end{aligned} \tag{12}$$

- We want to normalize this over c_{di}. Since $c_{di} \in \{1, \ldots, K\}$, the integral in the denominator turns into a sum,

$$q(c_{di}) = \frac{\prod_{k=1}^{K}\left[\mathrm{e}^{\mathbb{E}_{-q}[\ln p(x_{di}|\beta_k)]+\mathbb{E}_{-q}[\ln \theta_{dk}]}\right]^{\mathbb{1}(c_{di}=k)}}{\sum_{c_{di}=1}^{K}\prod_{k=1}^{K}\left[\mathrm{e}^{\mathbb{E}_{-q}[\ln p(x_{di}|\beta_k)]+\mathbb{E}_{-q}[\ln \theta_{dk}]}\right]^{\mathbb{1}(c_{di}=k)}} \tag{13}$$

- This is another way of writing

$$q(c_{di}) = \prod_{k=1}^{K}\left[\frac{\mathrm{e}^{\mathbb{E}_{-q}[\ln p(x_{di}|\beta_k)]+\mathbb{E}_{-q}[\ln \theta_{dk}]}}{\sum_{j=1}^{K}\mathrm{e}^{\mathbb{E}_{-q}[\ln p(x_{di}|\beta_j)]+\mathbb{E}_{-q}[\ln \theta_{d,j}]}}\right]^{\mathbb{1}(c_{di}=k)} \tag{14}$$

- Notice that this is simply a discrete distribution,

$$q(c_{di}) = \text{Discrete}(\phi_{di}), \quad \phi_{di}(k) = \frac{\mathrm{e}^{\mathbb{E}_{-q}[\ln p(x_{di}|\beta_k)]+\mathbb{E}_{-q}[\ln \theta_{dk}]}}{\sum_{j=1}^{K}\mathrm{e}^{\mathbb{E}_{-q}[\ln p(x_{di}|\beta_j)]+\mathbb{E}_{-q}[\ln \theta_{d,j}]}} \tag{15}$$

- Compare this with Gibbs sampling. Remember that we sample from the conditional posterior distribution,

$$p(c_{di} = k|x_{di}, \beta, \theta_d) = \frac{p(x_{di}|\beta_k)\theta_{dk}}{\sum_{j=1}^{K} p(x_{di}|\beta_j)\theta_{d,j}}$$

  where β and θ_d are the most recent samples of these variables.

- For variational inference, we swap

$$p(x_{di}|\beta_k) \Rightarrow \mathrm{e}^{\mathbb{E}_{-q}[\ln p(x_{di}|\beta_k)]} \quad \text{and} \quad \theta_{dk} \Rightarrow \mathrm{e}^{\mathbb{E}_{-q}[\ln \theta_{dk}]}$$

  Rather than sampling, we then simply keep this "approximate conditional posterior" as the q distribution for c_{di}.

- Notice that while $a = \mathrm{e}^{\ln a}$, using expectations, $\mathbb{E}[a] \neq \mathrm{e}^{\mathbb{E}[\ln a]}$.

- We don't know what $\mathbb{E}[\ln \theta_{dk}]$ and $\mathbb{E}[\ln p(x_{di}|\beta_k)] = \mathbb{E}[\ln \beta_{k,x_{di}}]$ are yet, but that doesn't change what the optimal form of $q(c_{di})$ is. Notice that, if we can use the same logic to find $q(\beta_k)$ and $q(\theta_d)$, then after one cycle through the model variables we will be able to come back to $q(c_{di})$ and explicitly calculate these expectations.

- Also notice that, since d and i are arbitrary, we have solved $q(c_{di})$ for all d and i.

$q(\theta_d)$: The distribution on topics for document d

- To find $q(\theta_d)$, we focus only on terms in the log joint likelihood involving this variable

$$\begin{aligned} q(\theta_d) &\propto \mathrm{e}^{\sum_{i,k} \mathbb{E}_{-q}[\mathbb{1}(c_{di}=k)] \ln \theta_{dk} + \ln p(\theta_d)} \\ &\propto \prod_{k=1}^{K} \theta_{dk}^{\sum_{i=1}^{N_d} \mathbb{E}_{-q}[\mathbb{1}(c_{di}=k)] + \alpha - 1} \end{aligned} \tag{16}$$

- The term $\alpha - 1$ comes from the Dirichlet prior $p(\theta_d)$. We want to normalize this over θ_d subject to $\theta_{dk} \geq 0$ and $\sum_k \theta_{dk} = 1$. This was a problem on the first homework, where we saw that the solution is

$$q(\theta_d) = \text{Dirichlet}(\alpha_{d1}, \ldots, \alpha_{dK}), \quad \alpha_{dk} = \alpha + \sum_{i=1}^{N_d} \underbrace{\mathbb{E}_{-q}[\mathbb{1}(c_{di} = k)]}_{\phi_{di}(k)} \tag{17}$$

- The expectation of an indicator of an event is simply the probability of that event. Therefore,

$$\mathbb{E}_{-q}[\mathbb{1}(c_{di} = k)] \;=\; \sum_{j=1}^{K} q(c_{di} = j)\mathbb{1}(c_{di} = k) \;=\; q(c_{di} = k) \tag{18}$$

  Since we previously calculated this q distribution, we are able to solve this expectation and use $\phi_{di}(k)$ as defined above for $q(c_{di})$.

- Notice that the parameters of the Dirichlet use the expected histogram of the allocations of all words from document d. Therefore, if 75% of words are expected to come from topic 1, then $q(\theta_d)$ will reflect this by having $\mathbb{E}_q[\theta_{d,1}] \approx 0.75$. However, q will also capture an approximation of the uncertainty of this value under its posterior distribution.

- Again compare with Gibbs sampling, where we have the conditional posterior

$$p(\theta_d | c_d) \propto \underbrace{\Big[\prod_{i=1}^{N_d} \prod_{k=1}^{K} \theta_{dk}^{\mathbb{1}(c_{di}=k)}\Big]}_{p(c_d|\theta_d)} \underbrace{\Big[\prod_{k=1}^{K} \theta_{dk}^{\alpha-1}\Big]}_{p(\theta_d)} \tag{19}$$

- In this case,

$$p(\theta_d | c_d) = \text{Dirichlet}(\alpha_{d1}, \ldots, \alpha_{dK}), \quad \alpha_{dk} = \alpha + \sum_{i=1}^{N_d} \mathbb{1}(c_{di} = k)$$

  where c_{di} is the most recent sample of this variable. In this case, we use the *empirical* histogram (rather than the expected histogram) constructed from the most recent samples. Gibbs sampling then samples a new vector θ_d from this distribution, while variational inference keeps the approximate conditional posterior as the approximation to the full posterior of this variable.

- Again, because d is arbitrary, we've solved for all θ_d.

$q(\beta_k)$: The topics

- Finally, we learn the q distributions for the topics themselves. Following the same procedure as for $q(\theta_d)$ and $q(c_{di})$, we have

$$\begin{aligned} q(\beta_k) &\propto \mathrm{e}^{\sum_{d,i} \mathbb{E}_{-q}[\mathbb{1}(c_{di}=kk)] \ln p(x_{di}|\beta_k) + \ln p(\beta_k)} \\ &\propto p(\beta_k) \prod_{d=1}^{D} \prod_{i=1}^{N_d} p(x_{di}|\beta_k)^{\mathbb{E}_{-q}[\mathbb{1}(c_{di}=k)]} \end{aligned} \tag{20}$$

- Since $p(x_{di}|\beta_k) = \prod_{v=1}^{V} \beta_{kv}^{\mathbb{1}(x_{di}=v)}$,

$$q(\beta_k) \propto \prod_{v=1}^{V} \beta_{kv}^{\sum_{d,i} \mathbb{E}_{-q}[\mathbb{1}(c_{di}=k)]\mathbb{1}(x_{di}=v)+\gamma-1} \tag{21}$$

- Normalizing over the probability vector β_k,

$$q(\beta_k) = \text{Dirichlet}(\gamma_{k,1}, \ldots, \gamma_{k,V}), \quad \gamma_{k,v} = \sum_{d=1}^{D} \sum_{i=1}^{N_d} \mathbb{E}_{-q}[\mathbb{1}(c_{di} = k)]\mathbb{1}(x_{di} = v) + \gamma \tag{22}$$

- Once again, we set $\mathbb{E}_{-q}[\mathbb{1}(c_{di} = k)] = \phi_{di}(k)$ as defined in the update of $q(c_{di})$.

- We can interpret $\sum_{d=1}^{D} \sum_{i=1}^{N_d} \phi_{di}(k)\mathbb{1}(x_{di} = v)$ as follows:

  1. $\phi_{di}(k)$: The probability that word x_{di} came from topic k according to $q(c_{di})$
  2. $\mathbb{1}(x_{di} = v)$: An indicator of what the ith word in document d corresponds to

- So if $v \rightarrow$ "government" then this sum is the expected total number of times we see the word "government" come from topic k given the model's q distributions at the current iteration.

- Because we solved for $q(c_{di})$ first, and the updates of $q(\theta_d)$ and $q(\beta_k)$ only used $q(c_{di})$, we were able to input the correct variational parameters to update these last two distributions. Finally, we need to address how to solve the expectations in $q(c_{di})$ in order to update this distribution, and therefore explicitly say what each ϕ_{di} equals in the updates of $q(\theta_d)$ and $q(\beta_k)$.

- That is, we have the question, how do we actually calculate

$$\mathbb{E}_{-q}[\ln \theta_{dk}] \quad \text{and} \quad \mathbb{E}_{-q}[\ln \beta_{kv}] \; ?$$

- The quick answer is, since we know these expectations are both with respect to Dirichlet distributions on θ and β, we can go to Wikipedia and look up the solution. However, we can also derive this using a clever technique.

- To do so, we will find that understanding properties of exponential family distributions gives a general way to find many interesting expectations.

Exponential family distributions

- We take a quick detour to discuss exponential family distributions. Almost all distributions we have (and will) discuss are in the "exponential family," for example, Gaussian, beta, Dirichlet, gamma, Poisson, multinomial, etc. distributions.

- This means they can be written in the form

$$p(x|\eta) = h(x)\mathrm{e}^{\eta^T t(x) - A(\eta)} \tag{23}$$

  where these terms are called as follows:

  1. η is the natural parameter vector
  2. $t(x)$ is the sufficient statistic vector
  3. $h(x)$ is the base measure (a function of x)
  4. $A(\eta)$ is the log-normalizer (a function of η)

- Q: Is there a general way to find $\mathbb{E}[t(x)]$?

- A: Yes, the derivation follows: Since

$$\int p(x|\eta)dx = 1 \quad \Rightarrow \quad \nabla_\eta \int p(x|\eta)dx = 0 \tag{24}$$

  we have that

$$\begin{aligned}
\nabla_\eta \int p(x|\eta)dx &= 0 && (25)\\
&\Downarrow \\
\int (t(x) - \nabla_\eta A(\eta))p(x|\eta)dx &= 0 && (26)\\
&\Downarrow \\
\int t(x)p(x|\eta)dx &= \int \nabla_\eta A(\eta)p(x|\eta)dx && (27)\\
&\Downarrow \\
\mathbb{E}[t(x)] &= \nabla_\eta A(\eta) && (28)
\end{aligned}$$

- We will find that, in many cases, the expectations we want to take in variational inference are of a sufficient statistic. We show this for the Dirichlet example.

- Dirichlet example: The density of the Dirichlet distribution can be put into the exponential family form as follows

$$\begin{aligned}
p(x|\alpha_1, \ldots, \alpha_k) &= \frac{\Gamma(\sum_i \alpha_i)}{\prod_i \Gamma(\alpha_i)} \prod_{i=1}^k x_i^{\alpha_i - 1} && (29)\\
&= \Big(\prod_{i=1}^k x_i^{-1}\Big) \mathrm{e}^{\sum_{i=1}^k \alpha_i \ln x_i - \left(\sum_i \ln \Gamma(\alpha_i) - \ln \Gamma(\sum_i \alpha_i)\right)} && (30)
\end{aligned}$$

- Matching terms with the generic exponential family distribution form, we have
  - $h(x) = \prod_i x_i^{-1}$
  - $t(x) = [\ln x_1, \ldots, \ln x_k]^T$
  - $\eta = [\alpha_1, \ldots, \alpha_k]^T$
  - $A(\eta) = \sum_i \ln \Gamma(\alpha_i) - \ln \Gamma(\sum_i \alpha_i)$

  Therefore, $\mathbb{E}[t(x)] = \nabla_\eta A(\eta)$ implies that, for each i,

$$\mathbb{E}[\ln x_i] = \partial A / \partial \alpha_i \quad \rightarrow \quad \mathbb{E}[\ln x_i] = \frac{\partial \ln \Gamma(\alpha_i)}{\partial \alpha_i} - \frac{\partial \ln \Gamma(\sum_j \alpha_j)}{\partial \alpha_i} \tag{31}$$

  These derivatives appear often enough that they have been given a symbol, $\psi(\cdot)$, called a digamma function. Therefore, $\mathbb{E}[\ln x_i] = \psi(\alpha_i) - \psi(\sum_j \alpha_j)$. This can be evaluated in languages such as Matlab using a built-in function.

- The final variational inference algorithm for LDA is given below.

Variational inference for latent Dirichlet allocation (LDA)

1. Define $q(c_{di}) = \text{Discrete}(\phi_{di})$, $q(\theta_d) = \text{Dirichlet}(\alpha_d)$ and $q(\beta_k) = \text{Dirichlet}(\gamma_k)$. Initialize each $\phi_{di}^{(0)}$, $\alpha_d^{(0)}$ and $\gamma_k^{(0)}$ in some way.
2. for iteration $t = 1, \ldots, T$
   (a) For each d and i, set

$$\phi_{di}^{(t)}(k) = \frac{\mathrm{e}^{\psi(\gamma_{k,x_{di}}^{(t-1)}) - \psi(\sum_v \gamma_{k,v}^{(t-1)}) + \psi(\alpha_{d,k}^{(t-1)}) - \psi(\sum_j \alpha_{d,j}^{(t-1)})}}{\sum_{m=1}^K \mathrm{e}^{\psi(\gamma_{m,x_{di}}^{(t-1)}) - \psi(\sum_v \gamma_{m,v}^{(t-1)}) + \psi(\alpha_{d,m}^{(t-1)}) - \psi(\sum_j \alpha_{d,j}^{(t-1)})}}$$

   (b) For each d and $k = 1, \ldots, K$, set

$$\alpha_{dk}^{(t)} = \alpha + \sum_{i=1}^{N_d} \phi_{di}^{(t)}(k)$$

   (c) For each k and $v = 1, \ldots, V$, set

$$\gamma_{kv}^{(t)} = \gamma + \sum_{d=1}^{D} \sum_{i=1}^{N_d} \phi_{di}^{(t)}(k) \mathbb{1}(x_{di} = v)$$

3. Using all variational parameter updates after iteration t, evaluate

$$\mathcal{L}_t = \mathbb{E}_q[\ln p(x, c, \beta, \theta)] - \sum_{d,i} \mathbb{E}_q[\ln q(c_{di})] - \sum_d \mathbb{E}_q[\ln q(\theta_d)] - \sum_k \mathbb{E}_q[\ln q(\beta_k)]$$

   to assess convergence. This function must be monotonically increasing as a function of t.

EECS E6720 Bayesian Models for Machine Learning

Columbia University, Fall 2016

Lecture 8, 11/3/2016

Instructor: John Paisley

- We next look at scalable variational inference in the context of conjugate exponential family models. We first review these types of models.

Conjugate exponential family (CEF) models

- General setup: We have a model of data X with variables $\theta_1, \ldots, \theta_m$. In combination with their priors, this defines a joint likelihood $p(X, \theta_1, \ldots, \theta_m)$.

- A major class of models are called "conjugate exponential family" (CEF) models. The features of this type of model are:

  1. All distributions on variables and data defined in the model are in the exponential family.
  2. If we pick one variable, θ_i, it's conditional posterior $p(\theta_i|X, \theta_{-i})$ is in the same family as the prior distribution on θ_i. That is, all priors are conjugate to the likelihood of their variable.

- Example: Let's return the the earlier toy model,

$$y_i \sim N(x_i^T w, \alpha^{-1}), \quad w \sim N(0, \lambda^{-1} I), \quad \alpha \sim Gam(a, b), \quad \lambda \sim Gam(c, d)$$

  Checks:

  1. The Gaussian and gamma distributions are exponential family distributions
  2. $p(w|-)$ is Gaussian; $p(\alpha|-)$ is gamma; $p(\lambda|-)$ is gamma

  So it is a CEF model.

- The reason we care about CEF models is that (1) They are commonly used in machine learning, (2) We can make general statements about variational inference for these types of models.

- From Bishop (slightly modified), we have a conjugate likelihood-prior pair if they can be written as follows:

$$p(x|\eta) = h(x)g(\eta)e^{\eta^T t(x)}, \quad p(\eta|\xi, \nu) = f(\xi, \nu)g(\eta)^\nu \mathrm{e}^{\eta^T \xi} \tag{1}$$

Because we gave names to the terms in $p(x|\eta)$ last time, and we are claiming that both $p(x|\eta)$ and $p(\eta|\xi,\nu)$ are exponential family distributions we should map the names to $p(\eta|\xi,\nu)$. In these two distributions, the base measures are $h(x)$ and $g(\eta)^\nu$ respectively. The natural parameters are η and ξ respectively. The log normalizers are $\ln g(\eta)$ and $\ln f(\xi,\nu)$ respectively. And the sufficient statistics are $t(x)$ and η respectively. However, this isn't the perspective we take with the prior.

Without belaboring the point too much further. Notice that the log normalizer of $p(x|\eta)$ is the same as the log base measure of $p(\eta|\xi,\nu)$. We'll move on to an example now.

- To give a simple, more general example, let

$$x \sim p(x|\theta_1,\theta_2), \quad \theta_1 \sim p(\theta_1|\theta_3), \quad \theta_2 \sim p(\theta_2), \quad \theta_3 \sim p(\theta_3) \tag{2}$$

Let's focus on θ_1. Using a slightly new notation for this problem, we have

Likelihood:

$$p(x,\theta_2|\theta_1) = h(x,\theta_2)\mathrm{e}^{\eta(\theta_1)^T t(x,\theta_2) - nA(\eta(\theta_1))}$$

Conjugate prior:

$$p(\theta_1|\theta_3) = f(\xi(\theta_3),\nu)\mathrm{e}^{\eta(\theta_1)^T \xi(\theta_3) - \nu A(\eta(\theta_1))}$$

Posterior:

$$\begin{aligned} p(\theta_1|x,\theta_2,\theta_3) &\propto p(x|\theta_1,\theta_2)p(\theta_1|\theta_3) \\ &\propto \mathrm{e}^{\eta(\theta_1)^T(t(x,\theta_2)+\xi(\theta_3)) - (n+\nu)A(\eta(\theta_1))} \end{aligned} \tag{3}$$

The notation $\eta(\theta_1)$ simply accounts for the fact that θ_1 as defined by us might not correspond to the natural parameter form.

- However, notice that if we want to make this a distribution on θ_1, it's the same form as the prior. That is, we can say that the posterior distribution

$$p(\theta_1|x,\theta_2,\theta_3) = f(\xi',\eta')\mathrm{e}^{\eta(\theta_1)^T\xi' - \nu' A(\eta(\theta_1))} \tag{4}$$

$$\xi' = t(x,\theta_2) + \xi(\theta_3), \quad \nu' = n + \nu$$

- How does this relate to variational inference? Recall that the optimal $q(\theta_1)$ is found by:

1. Taking the log of the joint likelihood
2. Taking the expectation wrt all q distributions except for $q(\theta_1)$
3. Exponentiating the result
4. Normalizing it as a function of θ_1

- In a CEF model this optimal q distribution always is in the same family as the prior and has expectations over the terms in the exponent. In the context of the above abstract model, we can already see that

$$q(\theta_1) \propto \exp\{\eta(\theta_1)^T(\mathbb{E}_q[t(x,\theta_2)] + \mathbb{E}_q[\xi(\theta_3)]) - (n+\nu)A(\eta(\theta_1))\} \tag{5}$$

That is,

$$q(\theta_1) = f(\xi', \eta')\mathrm{e}^{\eta(\theta_1)^T\xi' - \nu' A(\eta(\theta_1))} \tag{6}$$
$$\xi' = \mathbb{E}[t(x, \theta_2)] + \mathbb{E}[\xi(\theta_3)], \quad \nu' = n + \nu$$

- Let's be a little less abstract (but not totally concrete) and look at this in the context of a model *framework*.

Mixed membership models (and latent factor models)

- Mixed membership models can be broken down into the following components. (Latent factor models can as well.)

1. Grouped data $x_1, \ldots, x_D$

   The data is naturally grouped, and modeled as a group. Here, the group is not precise, but can be thought of as being more complex than a simple point in $\mathbb{R}^d$. For example, each group could be a person, and the data could be various vital statistics about that person. The group could also be an audio signal from which is extracted a large set of time-varying features. In topic modeling, the group is the document, and the data is the set of words making up that document.

2. Global variables β

   These are the model variables that directly impact every group of data. For example, in LDA they are the set of topics from which every word in the data set is drawn.

3. Local variables $z_1, \ldots, z_D$

   These mirror the grouped data: Model variables in z_i only interact with data in group x_i. Given the global variables, z_i in no way impacts $x_{i'}$ for $i' \neq i$. The variables in z_i can be complicated and have their own hierarchical dependencies. For example, in LDA $z_i = \{\theta_i, c_{i1}, \ldots, c_{in_i}\}$, where θ_i is the distribution on topics and c_{ij} is the indicator of the topic for the jth word in x_i. The c's are generated from a distribution parameterized by θ.

- Mixed membership models and latent factor models are distinguished in the way z and β combine to generate x probabilistically, but these details aren't necessary for the following discussion. LDA is the most common mixed membership model for discrete data.

- An important property of this type of model is that the joint likelihood factorizes in a nice way,

$$p(x, z, \beta) = p(\beta)\prod_{i=1}^{D} p(x_i, z_i|\beta) \quad \Big(= p(\beta)\prod_{i=1}^{D} p(x_i|z_i, \beta)p(z_i)\Big) \tag{7}$$

For example, for LDA this is

$$p(x, z, \beta) = \Big(\prod_k p(\beta_k)\Big)\prod_{i=1}^{D} p(\theta_i)\prod_{j=1}^{n_i} p(x_{ij}|\beta_{c_{ij}})p(c_{ij}|\theta_i)$$

- For the generic model, it follows that the log joint likelihood is

$$\ln p(x, z, \beta) = \sum_{i=1}^{D} \ln p(x_i, z_i|\beta) \; + \; \ln p(\beta) \tag{8}$$

- Since we can't find the exact posterior distribution of all the model variables, we approximate with $q(z, \beta)$ using variational inference. We pick the factorization

$$q(z, \beta) = q(\beta) \prod_{i=1}^{D} q(z_i) \tag{9}$$

  These factorizations can then sub-factorize. For example, with LDA, $q(z_i) = q(\theta_i) \prod_j q(c_{ij})$ (using the definition of z_i from earlier). However, at the level at which we are talking about this model framework, we must leave the factorization as written above.

- The variational objective function is then computed,

$$\mathcal{L} = \sum_{i=1}^{D} \mathbb{E}_q \left[\ln \frac{p(x_i, z_i|\beta)}{q(z_i)} \right] + \mathbb{E}_q \left[\ln \frac{p(\beta)}{q(\beta)} \right] \tag{10}$$

- One typical way for optimizing this objective function is called "batch" variational inference. This algorithm is as follows:

Batch variational inference

1. For $i = 1, \ldots, D$, optimize $q(z_i)$
2. Optimize $q(\beta)$
3. Repeat

- The general coding structure therefore involves an inner loop to optimize the q distribution for each group of data. Then the q distribution for the global variables is optimized, usually very quickly using the statistics collected during the inner loop. This is done in an outer loop, which counts how many iterations we run.

- For LDA, since $q(z_i) = q(\theta_i) \prod_j q(c_{ij})$, the inner loop itself may use an algorithm where we iterate back and forth several times between updating $q(\theta_i)$ and updating each $q(c_{ij})$.

- This can lead to scalability issues. For example, what if D is massive? What if we have millions of documents we want to do topic modeling on? Even if learning each $q(z_i)$ is fast (e.g., a fraction of a second), that fraction multiplied several million times can lead to an algorithm that takes over 24 hours to run a single iteration!

- Looking at the structure of the model, we can obviously speed this up with parallelization. That is, in Step 1 of the algorithm, we can break the groups into subsets and send that information out to

different processors. For example, if we have 100 computers, each computer can be responsible for optimizing 1% of the $q(z_i)$. This can make the algorithm run about 100 times faster.

- Another way to speed up variational inference in this context is to use techniques from optimization. Notice that $\mathcal{L}$ is an objective function like any other, just with a Bayesian interpretation of the result. Since $\mathcal{L}$ for mixed membership models such as LDA can be written as a sum over groups, we can use stochastic optimization to optimize it.

- We will talk about stochastic optimization next at a very high level. It's the type of technique that needs to be studies more fully in an optimization class, since there is a lot that is known about it. We will just state the procedure here and analyze it in the context of variational inference for CEF models.

- One thing to notice in the following is that it is not an either/or choice between stochastic inference and parallelization. The stochastic algorithm described below can be parallelized very easily to provide a massive speed-up of inference.

Stochastically optimizing $\mathcal{L}$

- Since the local variables factorize, we get a sum over groups in the variational objective. Again, this means we can generically write

$$\mathcal{L} = \sum_{i=1}^{D} \mathbb{E}_q \left[\ln \frac{p(x_i, z_i|\beta)}{q(z_i)} \right] + \mathbb{E}_q \left[\ln \frac{p(\beta)}{q(\beta)} \right] \tag{11}$$

- This type of objective function is perfect for "stochastic optimization." Operationally speaking, to stochastically optimize $\mathcal{L}$, we do the following at each iteration:

Stochastic variational inference

1. Randomly select a subset of local data, $S_t \subset \{1, \ldots, D\}$
2. Construct the scaled variational objective function

$$\mathcal{L}_t = \frac{D}{|S_t|} \sum_{i \in S_t} \mathbb{E}_q \left[\ln \frac{p(x_i, z_i|\beta)}{q(z_i)} \right] + \mathbb{E}_q \left[\ln \frac{p(\beta)}{q(\beta)} \right] \tag{12}$$

3. Optimize each $q(z_i)$ in $\mathcal{L}_t$ *only*
4. Update the parameters of $q(\beta|\psi)$ using a gradient step (assume ψ is the natural parameter, but it doesn't have to be)

$$q(\beta|\psi) \quad \rightarrow \quad \psi_t = \psi_{t-1} + \rho_t M_t \nabla_\psi \mathcal{L}_t \tag{13}$$

5. Repeat

- Comments:

  1. We can show that $\mathbb{E}[\mathcal{L}_t] = \mathcal{L}$. Rewrite $\mathcal{L}_t$ as

$$\mathcal{L}_t = \frac{D}{|S_t|} \sum_{i=1}^{D} \mathbb{1}(d \in S_t) \mathbb{E}_q \left[\ln \frac{p(x_i, z_i|\beta)}{q(z_i)} \right] + \mathbb{E}_q \left[\ln \frac{p(\beta)}{q(\beta)} \right]$$

  Let $S_t \sim P(S_t)$. This is the distribution of a subset of size $|S_t|$ sampled from $\{1, \ldots, D\}$ uniformly *without* replacement. Then

$$\mathbb{E}_P[\mathcal{L}_t] = \frac{D}{|S_t|} \sum_{i=1}^{D} \underbrace{\mathbb{E}_q[\mathbb{1}(d \in S_t)]}_{P(d \in S_t)} \mathbb{E}_q \left[\ln \frac{p(x_i, z_i|\beta)}{q(z_i)} \right] + \mathbb{E}_q \left[\ln \frac{p(\beta)}{q(\beta)} \right]$$

  We need to calculate $P(d \in S_t)$, which is the probability that the dth item is picked to be in set S_t. We can use combinatorics to calculate this. Since S_t is uniformly created, each set is equally probable. Since there are $\binom{D}{|S_t|}$ different sets, they each have probability $\binom{D}{|S_t|}^{-1}$. Also, we need to count how many of the possible sets contain d. Knowing that d is in the set, there are $\binom{D-1}{|S_t|-1}$ possible sets of size $|S_t| - 1$ that we could add d to create a set of size $|S_t|$ that contains d. Therefore the result follows from the fact that

$$P(d \in S_t) = \binom{D-1}{|S_t|-1} \binom{D}{|S_t|}^{-1} = \frac{|S_t|}{D}.$$

  Since the stochastic update is $\psi_t = \psi_{t-1} + \rho_t M_t \nabla_\psi \mathcal{L}_t$, we have shown that

$$\mathbb{E}_P[\psi_t] = \psi_{t-1} + \rho_t M_t \mathbb{E}_P[\nabla_\psi \mathcal{L}_t] = \psi_{t-1} + \rho_t M_t \nabla_\psi \mathcal{L}$$

  In other words, the stochastic gradient is *unbiased*. In combination with the second comment, this is important for proving that when stochastic optimization converges, it finds a local optimal solution of $\mathcal{L}$, containing all of the data.

  2. We must have $\sum_{t=1}^{\infty} \rho_t = \infty$ and $\sum_{t=1}^{\infty} \rho_t^2 < \infty$ for this method to provably converge to a local optimal solution of $\mathcal{L}$. Often, people will choose $\rho_t = \frac{1}{(t_0+t)^\kappa}$ with $\kappa \in (\frac{1}{2}, 1]$, since this can be shown to satisfy these requirements.

Example: Stochastic inference for LDA

- Before we look more in detail into $\psi_t = \psi_{t-1} + \rho_t M_t \nabla_\psi \mathcal{L}_t$ in general for CEF models, let's look at the final algorithm we will get for LDA.

- The local variables are:

  1. $q(\theta_d)$: Distribution on topics for document d

  2. $q(c_{dj})$: Word allocation for word j in document d

- The global variables are:

1. $q(\beta_k)$: Topics (distributions on words) for $k = 1, \ldots, K$. We pick

$$q(\beta_k) = \text{Dirichlet}(\gamma_{k1}, \ldots, \gamma_{kV})$$

SVI for LDA

1. Pick a random subset of documents
2. Learn $q(c_{dj})$ an $q(\theta_d)$ for each document by iterating between updateing these q several times
3. Update γ_k as follows:
   (a) Define $\lambda_d^{(k)} = \sum_{j=1}^{n_d} \mathbb{E}_q[\mathbb{1}(c_{di} = k)]\text{e}_{x_{di}}$ (e_i is vector of all zeros except 1 in dim i)
   (b) Set $\gamma_k^{(t)} = (1 - \rho_t)\gamma_k^{(t-1)} + \rho_t \left(\gamma + \frac{D}{|S_t|} \sum_{d \in S_t} \lambda_d^{(k)}\right)$

- Notice that SVI for LDA takes the old parameters and averages them with the new parameters calculated only over the subset chosen for the current iteration. As the iterations increase, the new information is weighted less and less since ρ_t is decreasing to zero.
- Contrast this with the "batch" inference algorithm
  1. In Step 1, we instead picked all the documents
  2. In Step 3b, we instead set $\gamma_k^{(t)} = \gamma + \sum_{d=1}^{D} \lambda_d^{(k)}$. Equivalently, we just set $\rho_t = 1$ because we fully weight all the new information.
- Let's return to the generic CEF hierarchy to see more of the difference between batch and stochastic variational inference.

Abstract stochastic variational inference

- We're in the general mixed membership setting described above. Also, let β be a natural parameter for notation convenience.
- Likelihood:

$$p(x, z|\beta) = \prod_{i=1}^{D} p(x_i, z_i|\beta) = \left[\prod_{i=1}^{D} h(x_i, z_i)\right] \text{e}^{\beta^T \sum_{i=1}^{D} t(x_i, z_i) - DA(\beta)} \tag{14}$$

- Prior:

$$p(\beta) = f(\xi, \nu)\text{e}^{\beta^T \xi - \nu A(\beta)} \tag{15}$$

- The optimal approximate posterior is in same family as the prior:

$$q(\beta) = f(\xi', \nu')\text{e}^{\beta^T \xi' - \nu' A(\beta)} \tag{16}$$

- Next, we specifically work with the variational objective function. Before, we didn't bother with this because we had the "trick" which allowed us to go straight to the answer. In other words, we could find (ξ', ν') such that $\nabla\mathcal{L} = 0$ without explicitly calculating $\mathcal{L}$.

- Focusing on the entire data set, in the gradient approach we want to set

$$\begin{bmatrix} \xi' \\ \nu' \end{bmatrix} \leftarrow \begin{bmatrix} \xi' \\ \nu' \end{bmatrix} + \rho_t M_t \nabla_{(\xi',\nu')}\mathcal{L} \tag{17}$$

  Therefore, we need to explicitly calculate $\mathcal{L}$.

- If we only focus on the terms in $\mathcal{L}$ involving β, we can say that the full variational objective function is

$$\mathcal{L}_\beta = \sum_{i=1}^{D} \mathbb{E}_q[\ln p(x_i, z_i|\beta)] + \mathbb{E}_q[\ln p(\beta)] - \mathbb{E}_q[\ln q(\beta)] \tag{18}$$

- And plugging in the distributions above, we have

$$\mathcal{L}_\beta = \mathbb{E}_q[\beta]^T \left(\sum_{i=1}^{D} \mathbb{E}[t(x_i, z_i)] + \xi - \xi' \right) - \mathbb{E}_q[A(\beta)](D + \nu - \nu') + \ln f(\xi', \nu') + \text{const.} \tag{19}$$

  The constant is with respect to ξ' and ν'.

- Next we need to calculate $\mathbb{E}_q[\beta]$ and $\mathbb{E}_q[A(\beta)]$ using the q distribution of β. We have already seen how we can do this with exponential family distributions. That is, if we solve the equalities

$$\int \nabla_{\xi'} q(\beta) d\beta = 0, \qquad \int \frac{\partial}{\partial \nu'} q(\beta) d\beta = 0 \tag{20}$$

  we will find that

$$\mathbb{E}_q[\beta] = -\nabla_{\xi'} \ln f(\xi', \nu'), \qquad \mathbb{E}_q[A(\beta)] = \frac{\partial \ln f(\xi', \nu')}{\partial \nu'} \tag{21}$$

- Therefore,

$$\begin{aligned} \mathcal{L}_\beta \;=\; & -\left[\nabla_{\xi'} \ln f(\xi', \nu')\right]^T \left(\sum_{i=1}^{D} \mathbb{E}[t(x_i, z_i)] + \xi - \xi' \right) - \frac{\partial \ln f(\xi', \nu')}{\partial \nu'}(D + \nu - \nu') \\ & - \ln f(\xi', \nu') \;+\; \text{const.} \end{aligned} \tag{22}$$

- Recall that we want to set

$$\nabla_{(\xi',\nu')}\mathcal{L}_\beta = \begin{bmatrix} \nabla_{\xi'}\mathcal{L}_\beta \\ \frac{\partial}{\partial \nu'}\mathcal{L}_\beta \end{bmatrix} = 0$$

- Using the equation for $\mathcal{L}_\beta$ above, we can calculate that

$$\begin{aligned}\nabla_{\xi'}\mathcal{L}_\beta &= -\nabla^2_{\xi'}\ln f(\xi',\nu')\left(\sum_{i=1}^{D}\mathbb{E}_q[t(x_i,z_i)]+\xi-\xi'\right)+\nabla_{\xi'}\ln f(\xi',\nu') \\ &\quad -\frac{\partial^2\ln f(\xi',\nu')}{\partial\nu'\partial\xi}(D+\nu-\nu')-\nabla_{\xi'}\ln f(\xi',\nu') \qquad (23)\end{aligned}$$

$$\begin{aligned}\frac{\partial}{\partial\nu'}\mathcal{L}_\beta &= -\frac{\partial^2\ln f(\xi',\nu')}{\partial\nu'\partial\xi'^T}\left(\sum_{i=1}^{D}\mathbb{E}_q[t(x_i,z_i)]+\xi-\xi'\right)-\frac{\partial^2\ln f(\xi',\nu')}{\partial\nu'^2}(D+\nu-\nu') \\ &\quad +\frac{\partial\ln f(\xi',\nu')}{\partial\nu'}-\frac{\partial\ln f(\xi',\nu')}{\partial\nu'} \qquad (24)\end{aligned}$$

- Notice that in both equations, there are two terms that cancel out. As written, this is a two-equations two-unknowns situation. However, it makes the problem much clearer to write it as follows

$$\nabla_{(\xi',\nu')}\mathcal{L}_\beta=\begin{bmatrix}\nabla_{\xi'}\mathcal{L}_\beta\\ \frac{\partial}{\partial\nu'}\mathcal{L}_\beta\end{bmatrix}=-\begin{bmatrix}\nabla^2_{\xi'}\ln f(\xi',\nu') & \frac{\partial^2\ln f(\xi',\nu')}{\partial\nu'\partial\xi}\\ \frac{\partial^2\ln f(\xi',\nu')}{\partial\nu'\partial\xi'^T} & \frac{\partial^2\ln f(\xi',\nu')}{\partial\nu'^2}\end{bmatrix}\begin{bmatrix}\sum_{i=1}^{D}\mathbb{E}_q[t(x_i,z_i)]+\xi-\xi'\\ D+\nu-\nu'\end{bmatrix}$$

- Again, all we've done is write the previous two equations in matrix vector form.

- The goal is to set the resulting vector to zero. However, notice that, since the preconditioning matrix is negative definite, the only way we can make this matrix-vector product equal zero is by setting the right vector to zero. This means we want to find values of ξ' and ν' such that the right vector is zero. We can simply read this from the vector:

$$\xi'=\xi+\sum_{i=1}^{D}\mathbb{E}[t(x_i,z_i)],\qquad \nu'=\nu+D \qquad (25)$$

- This is exactly the update for these parameters that we derived before by following the "log–expectation–exponential–normalization" rule. It is good to see via another route that this is the correct updates for these parameters. However, the reason we are calculating these things now is to see how stochastic variational inference modifies this.

- Recall that for a parameter to a variational q distribution, ψ, we want to set $\psi_t = \psi_{t-1} + \rho_t M_t \nabla_\psi \mathcal{L}_t$, where $\mathcal{L}_t$ is calculated using the sub-sampled set of groups. Using the above notation, that means that: (Aside: in the matrix below the gradients of f with respect to ξ' and ν' are evaluated at xi'_{t-1} and ν'_{t-1})

$$\begin{bmatrix}\xi'_t\\ \nu'_t\end{bmatrix}=\begin{bmatrix}\xi'_{t-1}\\ \nu'_{t-1}\end{bmatrix}-\rho_t M_t\begin{bmatrix}\nabla^2_{\xi'}\ln f(\xi',\nu') & \frac{\partial^2\ln f(\xi',\nu')}{\partial\nu'\partial\xi}\\ \frac{\partial^2\ln f(\xi',\nu')}{\partial\nu'\partial\xi'^T} & \frac{\partial^2\ln f(\xi',\nu')}{\partial\nu'^2}\end{bmatrix}\begin{bmatrix}\frac{D}{|S_t|}\sum_{i\in S_t}\mathbb{E}[t(x_i,z_i)]+\xi-\xi'_{t-1}\\ D+\nu-\nu'_{t-1}\end{bmatrix} \qquad (26)$$

- We could simply leave it here and pick $M_t = I$. However, if we pick M_t in a clever way, we can get a very "clean" update. We need to pick M_t to be some positive definite matrix. For

example, steepest ascent uses $M_t = I$. Newton's method sets $M_t = -(\nabla^2 \mathcal{L}_t)^{-1}$. For stochastic variational inference, we set

$$M_t = - \begin{bmatrix} \nabla^2_{\xi'} \ln f(\xi', \nu') & \frac{\partial^2 \ln f(\xi', \nu')}{\partial \nu' \partial \xi} \\ \frac{\partial^2 \ln f(\xi', \nu')}{\partial \nu' \partial \xi'^T} & \frac{\partial^2 \ln f(\xi', \nu')}{\partial \nu'^2} \end{bmatrix}^{-1} \tag{27}$$

This is also evaluated at the values at iteration $t - 1$. This preconditioning matrix can be shown to be equivalent to $M_t = \mathbb{E}_q[\nabla^2 \ln q(\beta)]$ evaluated at parameters of q at iteration $t - 1$. This gradient is known as the *natural gradient*. This is a "good" gradient direction for reasons that have been analyzed (and are beyond the scope of this class). Therefore, we pick this M_t not only for convenience, but also motivated by the natural gradient method.

- When we use this value of M_t, we see that the update for ν'_t is always to set it equal to $D + \nu$. For ξ'_t it is

$$\xi'_t = (1 - \rho_t)\xi'_{t-1} + \rho_t \Big(\xi + \frac{D}{|S_t|} \sum_{i \in S_t} \mathbb{E}[t(x_i, z_i)] \Big) \tag{28}$$

- We see that the update is a weighted average of the new sufficient statistics with the old value. Compare this with the update for stochastic LDA above to see that this result generalizes the pattern we observed there.

- Also notice that the function of the scaling parameter is to treat each group of data as though it appears $D/|S_t|$ times in the data set. This is because we only look at $|S_t|/D$ fraction of data—if we look at 1% of the data, we treat each observation (group) as if it appeared 100 times to keep the size of the data set constant.

EECS E6720 Bayesian Models for Machine Learning

Columbia University, Fall 2016

Lecture 9, 11/10/2016

Instructor: John Paisley

Clustering with the Gaussian mixture model (GMM)

- We next look at a fundamental problem in machine learning—clustering data. There are many types of data we might want to cluster. We will focus on the Gaussian mixture model for data clustering, which restricts the data to be in $\mathbb{R}^d$.

- Data: That is, we're given a set of vectors $\{x_1, \ldots, x_n\}$ where each $x_i \in \mathbb{R}^d$.

- Clustering: The main goal of clustering is to partition the data into groups (clusters) such that data within the same cluster are "similar" and data in different clusters are "different." Intuitively, we can think of the Gaussian mixture model as measuring closeness by distance, so each cluster defines a region in a space in the same way we might think of a region in a physical location.

- K-means is one way of clustering data, and arguable the most fundamental, which is why I bring it up here. It's not a probabilistic method and so we won't discuss it, but it's important to know. In this lecture, we will focus on the probabilistic approach to clustering using a Gaussian mixture model.

- Gaussian mixture models assume a probability density of the data that is a weighted sum of K Gaussian distributions. That is, we model the distribution of each data point $x_i \in \mathbb{R}^d$ as follows:

$$x_i \overset{iid}{\sim} p(x|\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}) = \sum_{j=1}^{K} \pi_j \text{Normal}(x|\mu_j, \Lambda_j^{-1}) \tag{1}$$

- The number of Gaussians, K, is set in advance. There are ways to try to assess what it should be, but for now we just assume we have picked a setting. The model parameters are,
  - π : A probability distribution on Gaussians.
  - (μ_j, Λ_j) : The mean and precision of the jth Gaussian

- It might not be clear how to generate x_i from this distribution unlike a single Gaussian. Intuitively, you can think that, in a data set of size n, roughly π_j fraction of that data will come from the jth Gaussian with mean μ_j and precision (inverse covariance) Λ_j.

- For now, let's not think about priors on $\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}$. The likelihood of the data given $\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}$ is

$$p(x_1, \ldots, x_n | \pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}) \;=\; \prod_{i=1}^{n} \left(\sum_{j=1}^{K} \pi_j \frac{|\Lambda_j|^{\frac{1}{2}}}{(2\pi)^{\frac{d}{2}}} \mathrm{e}^{-\frac{1}{2}(x_i - \mu_j)^T \Lambda_j (x_i - \mu_j)} \right) \tag{2}$$

- If we were to try to maximize this likelihood over $\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}$ (i.e., perform "maximum likelihood"), we would not make it very far before we realize that we can't do this simply and in closed form. That is, no derivative of this likelihood can be set to zero and solved for a parameter of interest. We therefore would need to resort to gradient methods if we want to directly maximize $\ln p(x_1, \ldots, x_n | \pi, \boldsymbol{\mu}, \boldsymbol{\Lambda})$.

- However, recall that this is the setting where EM can help, and we will indeed see that that's the case for the GMM.

- Before discussing EM for the GMM, it's important here to point out that the EM algorithm does not equal "maximum likelihood" or "maximum a posteriori." EM is often presented (but not in this class!) as a maximum likelihood technique and it's easy to conflate the two. We've been discussing EM for MAP inference. In the context of the GMM:

  1. Maximum likelihood $\Rightarrow \arg\max \;\; \ln p(x | \pi, \boldsymbol{\mu}, \boldsymbol{\Lambda})$
  2. Maximum a posteriori $\Rightarrow \arg\max \;\; \ln p(x, \pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}) = \arg\max \;\; \ln p(x | \pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}) + \ln p(\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda})$

- EM is simply the procedure of adding "hidden data" or extra latent variables to this such that the marginal is correct. That is, adding c such that $\int p(x, c | \pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}) dc \;=\; p(x | \pi, \boldsymbol{\mu}, \boldsymbol{\Lambda})$ or $\int p(x, c, \pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}) dc \;=\; p(x, \pi, \boldsymbol{\mu}, \boldsymbol{\Lambda})$ as the case may be, and then using these c to get closed form updates for $\pi, \boldsymbol{\mu}$, and $\boldsymbol{\Lambda}$. Therefore, people will sometimes say "maximum likelihood EM" or "MAP-EM" to be more specific. We'll just say "EM," but notice that up to now we've been doing MAP-EM, whereas today we will focus on ML-EM.

ML-EM for the GMM

- Since the GMM is always derived in the maximum likelihood framework for EM, we'll do the same here and not define prior distributions on π, $\boldsymbol{\mu}$ or $\boldsymbol{\Lambda}$. We'll then discuss variational inference where we will need to introduce priors.

- The log likelihood $\ln p(x | \pi, \boldsymbol{\mu}, \boldsymbol{\Lambda})$ is a problem because the sum-log-sum form does not lead to closed form updates of parameters. To avoid gradient methods, we use EM, which means we must introduce an additional variable to the model. We start from the end and define a larger model, then show that it has the correct marginal.

- Model (expanded): For each $x_i \in \mathbb{R}^d$, given π, $\boldsymbol{\mu}$ and $\boldsymbol{\Lambda}$, generate

$$c_i \sim \text{Discrete}(\pi), \qquad x_i \,|\, c_i \sim \text{Normal}(\mu_{c_i}, \Lambda_{c_i}^{-1}) \tag{3}$$

- Notice that this is similar to LDA, where we used an indicator to say which topic a word came from. The function of this indicator is identical here. The value of $c_i \in \{1, \ldots, K\}$ says which

of the K Gaussians generated x_i, as is evident because c_i picks out the parameters of the corresponding Gaussian.

Marginal distribution

- We now need to check that this expanded model has the correct marginal distribution. We have to do this because we set out to maximize

$$\prod_{i=1}^{n} p(x_i|\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}) = \prod_{i=1}^{n} \sum_{j=1}^{n} \pi_j \text{Normal}(x_i|\mu_j, \Lambda_j^{-1}) \tag{4}$$

- The new extended model has the likelihood

$$\begin{aligned} \prod_{i=1}^{n} p(x_i, c_i|\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}) &= \prod_{i=1}^{n} p(x_i|c_i, \boldsymbol{\mu}, \boldsymbol{\Lambda}) p(c_i|\pi) \\ &= \prod_{i=1}^{n} \prod_{j=1}^{K} \left(\pi_j \text{Normal}(x_i|\mu_j, \Lambda_j^{-1})\right)^{\mathbb{1}(c_i=j)} \end{aligned} \tag{5}$$

- Notice that indicators are used to pick out the correct term like in LDA. The next question is, what is

$$p(x|\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}) = \sum_{c_1=1}^{K} \cdots \sum_{c_n=1}^{K} \prod_{i=1}^{n} p(x_i, c_i|\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}) = \prod_{i=1}^{n} \sum_{j=1}^{K} p(x_i, c_i = j|\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}) \tag{6}$$

- It might seem trivial, but we literally just plug in and sum

$$\prod_{i=1}^{n} \sum_{j=1}^{K} \underbrace{p(x_i|c_i = j, \boldsymbol{\mu}, \boldsymbol{\Lambda})}_{= \text{Normal}(\mu_j, \Lambda_j^{-1})} \underbrace{p(c_i = j|\pi)}_{= \pi_j} \tag{7}$$

- This is exactly the marginal we want, so we found our extra variable.

Deriving an EM algorithm for the GMM

- We start where we have to start, with the EM equation

$$\ln p(x|\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}) = \sum_{\boldsymbol{c}} q(\boldsymbol{c}) \ln \frac{p(x, \boldsymbol{c}|\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda})}{q(\boldsymbol{c})} \; + \; \sum_{\boldsymbol{c}} q(\boldsymbol{c}) \ln \frac{q(\boldsymbol{c})}{p(\boldsymbol{c}|x, \pi, \boldsymbol{\mu}, \boldsymbol{\Lambda})} \tag{8}$$

- We make a few comments about this equality:

    1. The extra variables $\boldsymbol{c} = (c_1, \ldots, c_n)$ are discrete. Therefore there is no integral because c isn't in a continuous space. Sums are used in discrete settings and integrals in continuous settings, but the path to the equality is identical. In our previous derivation of this equation, we can literally replace every $\int$ with a Σ and remove the infinitesimals (e.g., dc)

2. The sum is over all possible values of the vector $\boldsymbol{c}$, where each entry $c_i \in \{1, \ldots, K\}$. Therefore, as written, there are K^n vectors we have to sum over. Of course, this is impossible for almost any problem (K^n is massive!) so if we can't simplify the problem, then EM is useless.

3. Despite the combinatorial issue, this is the true, correct equation. We have to start from here and try to simplify. Unlike variational methods where we make approximations with respect to $q(\boldsymbol{c})$—i.e., we define $q(\boldsymbol{c}) = \prod_i q(c_i)$—in EM we must set $q(\boldsymbol{c}) = p(\boldsymbol{c}|x, \pi, \boldsymbol{\mu}, \boldsymbol{\Lambda})$ and so the true conditional posterior on all c_i dictates what we can do about $q(\boldsymbol{c})$.

4. Next we will show that the true conditional posterior factorizes, and so therefore $q(\boldsymbol{c})$ must factorize as well (i.e., there's no approximation in factorizing $q(\boldsymbol{c})$ as above). The posterior tells us that this is the case, and so no approximations are being made w.r.t. $q(\boldsymbol{c})$.

5. Also, as a side comment, notice that even when doing maximum likelihood where no priors are assumed on the model, posterior calculation using Bayes rule still factors into the problem. Therefore, even though we're not doing Bayesian modeling here, Bayesian ideas are still critical for making progress on this problem because we want to use EM and EM needs Bayes rule.

- We recall the steps of the EM algorithm and then work through each step.

  E-Step

  1. Set $q(\boldsymbol{c}) = p(\boldsymbol{c}|x, \pi, \boldsymbol{\mu}, \boldsymbol{\Lambda})$
  2. Calculate $\sum_{\boldsymbol{c}} q(\boldsymbol{c}) \ln p(x, \boldsymbol{c}|\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda})$

  M-Step

  3. Maximize #2 over π, $\boldsymbol{\mu}$ and $\boldsymbol{\Lambda}$

- E-Step (Step 1): Use Bayes rule to calculate the conditional posterior distribution

$$\begin{aligned} p(\boldsymbol{c}|x, \pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}) &\propto p(x|\boldsymbol{c}, \boldsymbol{\mu}, \boldsymbol{\Lambda}) p(\boldsymbol{c}|\pi) \\ &\propto \prod_{i=1}^{n} p(x_i|c_i, \boldsymbol{\mu}, \boldsymbol{\Lambda}) p(c_i|\pi) \end{aligned} \tag{9}$$

- This conditional independence removes the K^n combinatorial issue. We can divide this by a number $Z = \prod_{i=1}^{n} Z_i$ to normalize the entire function of $\boldsymbol{c}$ by normalizing each individual prior-likelihood pair,

$$\begin{aligned} p(\boldsymbol{c}|x, \pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}) &= \frac{1}{Z} \prod_{i=1}^{n} p(x_i|c_i, \boldsymbol{\mu}, \boldsymbol{\Lambda}) p(c_i|\pi) \\ &= \prod_{i=1}^{n} \frac{p(x_i|c_i, \boldsymbol{\mu}, \boldsymbol{\Lambda}) p(c_i|\pi)}{Z_i} \\ &= \prod_{i=1}^{n} p(c_i|x_i, \pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}) \end{aligned} \tag{10}$$

where

$$\begin{aligned} p(c_i = k|x_i, \pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}) &= \frac{p(x_i|c_i = k, \boldsymbol{\mu}, \boldsymbol{\Lambda})p(c_i = k|\pi)}{\sum_{j=1}^K p(x_i|c_i = j, \boldsymbol{\mu}, \boldsymbol{\Lambda})p(c_i = j|\pi)} \\ &= \frac{\pi_k \text{Normal}(x_i|\mu_k, \Lambda_k^{-1})}{\sum_{j=1}^K \pi_j \text{Normal}(x_i|\mu_j, \Lambda_j^{-1})} \end{aligned} \tag{11}$$

- So to summarize

$$q(\boldsymbol{c}) \;=\; p(\boldsymbol{c}|x, \pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}) \;=\; \prod_{i=1}^n \underbrace{p(c_i|x_i, \pi, \boldsymbol{\mu}, \boldsymbol{\Lambda})}_{\to q(c_i)} \;=\; \prod_{i=1}^n q(c_i) \tag{12}$$

- We use notation $q(c_i = j) = \phi_i(j)$. When implementing, we first set $\phi_i(j) \leftarrow \pi_j \text{Normal}(x_i|\mu_j, \Lambda_j^{-1})$ for $j = 1, \ldots, K$ and then normalize the K-dimensional vector ϕ_i by dividing it by its sum.

- E-Step (Step 2): The expectation we need to take is

$$\mathcal{L}(\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}) = \sum_{i=1}^n \mathbb{E}_{q(\boldsymbol{c})}[\ln p(x_i, c_i|\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda})] + \text{const.} \tag{13}$$

  The constant is with respect to $\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}$.

- Because each (x_i, c_i) are conditionally independent of each other. The expectation over $q(\boldsymbol{c})$ reduces to an expectation over $q(c_i)$. It's worth thinking about this. Each of the n expectations above is over the q distribution on the entire vector $\boldsymbol{c} = (c_1, \ldots, c_n)$. However, because the ith term only contains c_i in it, the sum over $c_{i'}$ $i' \neq i$ in $q(\boldsymbol{c})$ causes each $q(c_{i'})$ to disappear because it sums to one. We're then left with:

$$\mathcal{L}(\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}) = \sum_{i=1}^n \mathbb{E}_{q(c_i)}[\ln p(x_i, c_i|\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda})] + \text{const.} \tag{14}$$

- Notice that this takes care of the combinatorial problem. We now only need to sum over $K \cdot n$ terms instead of over K^n terms.

- Continuing,

$$\begin{aligned} \mathcal{L}(\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}) &= \sum_{i=1}^n \sum_{j=1}^K q(c_i = j)[\ln p(x_i|c_i = j, \boldsymbol{\mu}, \boldsymbol{\Lambda}) + \ln p(c_i = j|\pi)] + \text{const.} \\ &= \sum_{i=1}^n \sum_{j=1}^K \phi_i(j) \left(\frac{1}{2} \ln |\Lambda_j| - \frac{1}{2}(x_i - \mu_j)^T \Lambda_j (x_i - \mu_j) + \ln \pi_j \right) + \text{const.} \end{aligned} \tag{15}$$

- (The constant in the second line contains extra terms in addition to what's in the first line.)

- M-Step (Step 3): Finally, we maximize $\mathcal{L}(\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda})$ over its parameters. We hope we can do this by taking derivatives and setting to zero. If not, then EM probably hasn't helped us.

  a) $\nabla_{\mu_j} \mathcal{L} = 0$

$$\begin{aligned} \nabla_{\mu_j} \mathcal{L} &= -\sum_{i=1}^{n} \phi_i(j)(\Lambda_j \mu_j - \Lambda_j x_j) = 0 \qquad (16) \\ &\Downarrow \\ \mu_j &= \frac{1}{n_j} \sum_{i=1}^{n} \phi_i(j) x_i, \qquad n_j = \sum_{i=1}^{n} \phi_i(j) \qquad (17) \end{aligned}$$

  Notice that this is a weighted average of all the points in the data set. The weight is determined by the conditional posterior probability of each point coming from the jth cluster using the parameters from the previous iteration.

  b) $\nabla_{\Lambda_j} \mathcal{L} = 0$

$$\begin{aligned} \nabla_{\Lambda_j} \mathcal{L} &= \sum_{i=1}^{n} \phi_i(j)(\frac{1}{2}\Lambda_j^{-1} - \frac{1}{2}(x_i - \mu_j)(x_i - \mu_j)^T) = 0 \qquad (18) \\ &\Downarrow \\ \Lambda_j^{-1} &= \frac{1}{n_j} \sum_{i=1}^{n} \phi_i(j)(x_i - \mu_j)(x_i - \mu_j)^T, \qquad n_j = \sum_{i=1}^{n} \phi_i(j) \qquad (19) \end{aligned}$$

  We use the equalities $\nabla_\Lambda \ln |\Lambda| = \Lambda^{-1}$ and $x^T \Lambda x = \text{trace}(\Lambda x x^T) \Rightarrow \nabla_\Lambda x^T \Lambda x = x x^T$. Just like the mean μ_j we see the inverse of Λ_j (i.e., the covariance) is equal to a weighted version of the empirical covariance. For this step, we use the most recent update for μ_j.

  c) $\nabla_\pi \mathcal{L} = 0$ subject to $\pi_j \geq 0$ and $\sum_{j=1}^{K} \pi_j = 1$. This step requires Lagrange multipliers to ensure these constraints. We won't review the technique of Lagrange multipliers since it would be too much of a digression, but it's important to know in general. The solution is

$$\pi_j = \frac{n_j}{n}, \qquad n_j = \sum_{i=1}^{n} \phi_i(j) \qquad (20)$$

  The update of the prior probability that a point comes from cluster j is the fraction of points we expect to see from cluster j under the current posterior distribution.

- Notice we don't need to iterate over $\boldsymbol{\mu}$, $\boldsymbol{\Lambda}$, π to maximize $\mathcal{L}$.

- We are left with the following EM algorithm.

A maximum likelihood EM algorithm for the Gaussian mixture model

Input: Data $x_1, \ldots, x_n$, $x \in \mathbb{R}^d$. Number of clusters K.

Output: GMM parameters π, $\boldsymbol{\mu}$, $\boldsymbol{\Lambda}$ and cluster assignment distributions ϕ_i

1. Initialize $\pi^{(0)}$ and each $(\mu_j^{(0)}, \Lambda_j^{(0)})$ in some way.
2. At iteration t,
   (a) E-Step: For $i = 1, \ldots, n$ and $j = 1, \ldots, K$ set

$$\phi_i^{(t)}(j) = \frac{\pi_j^{(t-1)} \text{Normal}(x_i | \mu_j^{(t-1)}, (\Lambda_j^{(t-1)})^{-1})}{\sum_{k=1}^K \pi_k^{(t-1)} \text{Normal}(x_i | \mu_k^{(t-1)}, (\Lambda_k^{(t-1)})^{-1})}$$

3. M-Step: Set

$$\begin{aligned}
n_j^{(t)} &= \sum_{i=1}^n \phi_i^{(t)}(j) \\
\mu_j^{(t)} &= \frac{1}{n_j^{(t)}} \sum_{i=1}^n \phi_i^{(t)}(j) x_i \\
\Lambda_j^{(t)} &= \left(\frac{1}{n_j^{(t)}} \sum_{i=1}^n \phi_i^{(t)}(j)(x_i - \mu_j^{(t)})(x_i - \mu_j^{(t)})^T \right)^{-1} \\
\pi_j^{(t)} &= \frac{n_j^{(t)}}{n}
\end{aligned}$$

4. Calculate $f_t = \ln p(x | \pi^{(t)}, \boldsymbol{\mu}^{(t)}, \boldsymbol{\Lambda}^{(t)})$ to assess convergence of f as a function of t.

Variational inference (VI)

- We next look at VI for the GMM. We therefore need to define priors for π, $\boldsymbol{\mu}$ and $\boldsymbol{\Lambda}$.
- Priors: We have many options for priors, but we choose (out of convenience)

$$\pi \sim \text{Dirichlet}(\alpha), \quad \mu_j \sim \text{Normal}(0, cI), \quad \Lambda_j \sim \text{Wishart}(a, B) \tag{21}$$

- The Wishart distribution
- Since there's a good chance the Wishart distribution is new, we'll review it briefly here. The Wishart distribution is a probability distribution on positive definite matrices. It is the conjugate prior for the precision, Λ_j, of a multivariate Gaussian, which is why we pick it. For a $d \times d$ matrix Λ, the density function is

$$p(\Lambda|a, B) = \frac{|\Lambda|^{\frac{a-d-1}{2}} \mathrm{e}^{-\frac{1}{2}\text{trace}(B\Lambda)}}{2^{\frac{ad}{2}}|B|^{-\frac{a}{2}}\Gamma_d(a/2)} \tag{22}$$

  The parameter requirements are $a > d-1$ and B is a positive definite $d \times d$ matrix. The function $\Gamma_d(a/2)$ is a complicated function of the gamma function,

$$\Gamma_d(a/2) = \pi^{\frac{d(d-1)}{4}} \prod_{j=1}^{d} \Gamma\left(\frac{a}{2} + \frac{1-j}{2}\right)$$

- The important expectations for VI are of sufficient statistics of the Wishart distribution. Since the Wishart distribution is an exponential family distribution, we can use the technique discussed in a previous lecture to find that

$$\mathbb{E}[\Lambda] = aB^{-1}, \quad \mathbb{E}[\ln|\Lambda|] = d\ln 2 - \ln|B| + \sum_{j=1}^{d} \psi(a/2 + (1-j)/2) \tag{23}$$

  $\psi(\cdot)$ is the digamma function discussed previously. It's something you would call a built-in function to evaluate in your code.

- It's worth trying the previously discussed exponential family trick out for $\mathbb{E}[\ln|\Lambda|]$ to see how easily we can derive this nasty expectation.
- Joint and log joint likelihood: The joint likelihood and log joint likelihood can be written as

$$p(x, \boldsymbol{c}, \pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}) = \Big[\prod_{i=1}^{n}\prod_{j=1}^{K} p(x_i|c_i = j, \boldsymbol{\mu}, \boldsymbol{\Lambda})p(c_i = j|\pi)\Big]\Big[\prod_{j=1}^{K} p(\mu_j)p(\Lambda_j)\Big]p(\pi) \tag{24}$$

  Therefore, using the indicator function representation for c_i,

$$\ln p(x, \boldsymbol{c}, \pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}) = \sum_{i=1}^{n}\sum_{j=1}^{K} \mathbb{1}(c_i = j)[\ln p(x_i|\mu_j, \Lambda_j) + \ln \pi_j] + \sum_{j=1}^{K}[\ln p(\mu_j) + \ln p(\Lambda_j)] + \ln p(\pi) \tag{25}$$

- Posterior calculation: Since we can't calculate $p(\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}, \boldsymbol{c}|x)$ we use a variational approximation with mean-field factorization

$$p(\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}, \boldsymbol{c}|x) \approx q(\pi, \boldsymbol{\mu}, \boldsymbol{\Lambda}, \boldsymbol{c}) = q(\pi) \left[\prod_{j=1}^{K} q(\mu_j)q(\Lambda_j)\right] \left[\prod_{i=1}^{n} q(c_i)\right] \tag{26}$$

- Next we need to define the distribution family of each q. We know how to do this, but notice that we can make a CEF argument. All distributions are exponential family distributions (Gaussian, Wishart, Dirichlet, multinomial) and we can verify that the conditional posterior of each model variable is in the same family as the prior, hence there is conditional conjugacy. The model is a conjugate exponential family model and so each q distribution should be set to the same family as the prior.

- We still need to calculate how to update their parameters, so the CEF argument hasn't really reduced the amount of work we have to do. However, we can say at the outset that

  - $q(\pi) = \text{Dirichlet}(\alpha'_1, \ldots, \alpha'_K)$
  - $q(\mu_j) = \text{Normal}(m'_j, \Sigma'_j)$
  - $q(\Lambda_j) = \text{Wishart}(a'_j, B'_j)$
  - $q(c_i) = \text{Multinomial}(\phi_i)$

- The variational inference algorithm will then consist of iterating through each of these q distributions and modifying their parameter values to make their product closer to the true posterior distribution according to their KL divergence.

- Since this is a CEF model, we can use the "log-expectation-exponentiate-normalize" approach we've been discussing over the past several lectures.

- $q(c_i = j)$:

$$\begin{aligned} q(c_i = j) &\propto \mathrm{e}^{\mathbb{E}[\ln p(x_i, c_i=j|\pi,\mu_j,\Lambda_j)]} \\ &\propto \mathrm{e}^{\mathbb{E}[\ln p(x_i|\mu_j,\Lambda_j)]+\mathbb{E}[\ln p(c_i=j|\pi)]} \\ &\propto \mathrm{e}^{\frac{1}{2}\mathbb{E}[\ln|\Lambda_j|]-\frac{1}{2}\mathbb{E}[(x_i-\mu_j)^T\Lambda_j(x_i-\mu_j)]+\mathbb{E}[\ln \pi_j]} \end{aligned} \tag{27}$$

  The expectation is over π, μ_j and Λ_j. They are taken using the q distributions on them, so

$$\mathbb{E}[\ln \pi_j] = \psi(\alpha'_j) - \psi(\textstyle\sum_k \alpha'_k)$$

  Multiplying out we see that

$$\mathbb{E}[(x_i - \mu_j)^T\Lambda_j(x_i - \mu_j)] = (x_i - \mathbb{E}[\mu_j])^T\mathbb{E}[\Lambda_j](x_i - \mathbb{E}[\mu_j]) + \text{trace}(\mathbb{E}[\Lambda_j]\Sigma'_j)$$

  We discussed $\mathbb{E}[\ln|\Lambda_j|]$ above. Σ'_j is from $q(\mu_j)$ and $\mathbb{E}[\mu_j] = m'_j$.

- $\underline{q(\pi)}$:

$$\begin{aligned} q(\pi) &\propto \mathrm{e}^{\sum_{i=1}^n \mathbb{E}[\ln p(c_j|\pi)] + \ln p(\pi)} \\ &\propto \mathrm{e}^{\sum_{i=1}^n \sum_{j=1}^K \phi_i(j) \ln \pi_j + \sum_{i=1}^K (\alpha-1) \ln \pi_j} \\ &\propto \prod_{j=1}^K \pi_j^{\alpha + \sum_{i=1}^n \phi_i(j) - 1} \end{aligned} \tag{28}$$

  So we set

$$\alpha_j' = \alpha + n_j, \qquad n_j = \sum_{i=1}^n \phi_i(j)$$

- $\underline{q(\mu_j)}$:

$$\begin{aligned} q(\mu_j) &\propto \mathrm{e}^{\sum_{i=1}^n \mathbb{E}[\ln p(x_i|c_i=j,\mu_j,\Lambda_j)] + \ln p(\mu_j)} \\ &\propto \mathrm{e}^{-\frac{1}{2}\sum_{i=1}^n \phi_i(j)(x_i-\mu_j)^T \mathbb{E}[\Lambda_j](x_i-\mu_j) - \frac{1}{2c}\mu_j^T\mu_j} \\ &\propto \mathrm{e}^{-\frac{1}{2}\sum_{i=1}^n (x_i-\mu_j)^T(\phi_i(j)\mathbb{E}[\Lambda_j])(x_i-\mu_j) - \frac{1}{2c}\mu_j^T\mu_j} \end{aligned} \tag{29}$$

  Normalizing this over μ, the result is a Gaussian distribution $q(\mu_j) = \text{Normal}(m_j', \Sigma_j')$ where

$$\Sigma_j' = \left(c^{-1}I + n_j\mathbb{E}[\Lambda_j]\right)^{-1}, \qquad m_j' = \Sigma_j'\left(\mathbb{E}[\Lambda_j]\sum_{i=1}^n \phi_i(j)x_i\right), \qquad n_j = \sum_{i=1}^n \phi_i(j) \tag{30}$$

- $\underline{q(\Lambda_j)}$:

$$\begin{aligned} q(\Lambda_j) &\propto \mathrm{e}^{\sum_{i=1}^n \mathbb{E}[\ln p(x_i|c_i=j,\mu_j,\Lambda_j)] + \ln p(\Lambda_j)} \\ &\propto |\Lambda_j|^{\frac{n_j+a-d-1}{2}} \mathrm{e}^{-\frac{1}{2}\sum_{i=1}^n \phi_i(j)\left[(x_i-\mathbb{E}[\mu_j])^T\Lambda_j(x_i-\mathbb{E}[\mu_j]) + \text{trace}(\Lambda_j\Sigma_j')\right] - \frac{1}{2}\text{trace}(B\Lambda_j)} \end{aligned} \tag{31}$$

  We can see this has a Wishart form, $q(\Lambda_j) = \text{Wishart}(a_j', B_j')$ where

$$a_j' = a + n_j, \qquad B_j' = B + \sum_{i=1}^n \phi_i(j)\left[(x_i + \mathbb{E}[\mu_j])(x_i - \mathbb{E}[\mu_j])^T + \Sigma_j'\right] \tag{32}$$

  And again $n_j = \sum_{i=1}^n \phi_i(j)$. We used the fact that

$$\phi_i(j)(x_i - \mathbb{E}[\mu_j])^T\Lambda_j(x_i - \mathbb{E}[\mu_j]) = \text{tr}(\phi_i(j)(x_i - \mathbb{E}[\mu_j])(x_i - \mathbb{E}[\mu_j])^T\Lambda_j)$$

- In all of these above equations, the expectations are taken using the q distributions, so for example $\mathbb{E}[\mu_j] = m_j'$ using the most recent value of m_j'.
- The result is the following algorithm. I remove the $'$ below because the notation already looks very complicated. I give iteration indexes to these parameters to emphasize the iterative dependence, which makes it look complicated.

A variational inference algorithm for the Gaussian mixture model

Input: Data $x_1, \ldots, x_n$, $x \in \mathbb{R}^d$. Number of clusters K.

Output: Parameters for $q(\pi)$, $q(\mu_j)$, $q(\Lambda_j)$ and $q(c_i)$.

1. Initialize $(\alpha_1^{(0)}, \ldots, \alpha_K^{(0)})$, $(m_j^{(0)}, \Sigma_j^{(0)})$, $(a_j^{(0)}, B_j^{(0)})$ in some way.
2. At iteration t,

   (a) Update $q(c_i)$ for $i = 1, \ldots, n$. This one is complicated so we break it down. Set

   $$\phi_i^{(t)}(j) = \frac{\mathrm{e}^{\frac{1}{2}t_1(j) - \frac{1}{2}t_2(j) - \frac{1}{2}t_3(j) + t_4(j)}}{\sum_{k=1}^K \mathrm{e}^{\frac{1}{2}t_1(k) - \frac{1}{2}t_2(k) - \frac{1}{2}t_3(k) + t_4(k)}}$$

   where

   $$t_1(j) = \sum_{k=1}^d \psi\left(\frac{1 - k + a_j^{(t-1)}}{2}\right) - \ln|B_j^{(t-1)}|$$

   $$t_2(j) = (x_i - m_j^{(t-1)})^T (a_j^{(t-1)} (B_j^{(t-1)})^{-1})(x_i - m_j^{(t-1)})$$

   $$t_3(j) = \text{trace}(a_j^{(t-1)} (B_j^{(t-1)})^{-1} \Sigma_j^{(t-1)}), \qquad t_4(j) = \psi(\alpha_j^{(t-1)}) - \psi(\sum_k \alpha_k^{(t-1)})$$

   Tip: Some of these values can be calculated once and reused for each $q(c_i)$.

   (b) Set $n_j^{(t)} = \sum_{i=1}^n \phi_i^{(t)}(j)$ for $j = 1, \ldots, K$

   (c) Update $q(\pi)$ by setting

   $$\alpha_j^{(t)} = \alpha + n_j^{(t)}$$

   for $j = 1, \ldots, K$.

   (d) Update $q(\mu_j)$ for $j = 1, \ldots, K$ by setting

   $$\Sigma_j^{(t)} = \left(c^{-1} I + n_j^{(t)} a_j^{(t-1)} (B_j^{(t-1)})^{-1}\right)^{-1}, \qquad m_j^{(t)} = \Sigma_j^{(t)} \left(a_j^{(t-1)} (B_j^{(t-1)})^{-1} \sum_{i=1}^n \phi_i^{(t)}(j) x_i\right)$$

   (e) Update $q(\Lambda_j)$ for $j = 1, \ldots, K$ by setting

   $$a_j^{(t)} = a + n_j^{(t)}, \qquad B_j^{(t)} = B + \sum_{i=1}^n \phi_i^{(t)}(j) \left[(x_i - m_j^{(t)})(x_i - m_j^{(t)})^T + \Sigma_j^{(t)}\right]$$

   (f) Calculate the variational objective function to assess convergence as a function of iteration:

   $$\mathcal{L} = \mathbb{E}[\ln p(x, c, \pi, \boldsymbol{\mu}, \boldsymbol{\Lambda})] - \mathbb{E}[\ln q]$$

EECS E6720 Bayesian Models for Machine Learning

Columbia University, Fall 2016

Lecture 10, 11/17/2016

Instructor: John Paisley

Mixture models with Dirichlet priors

- Review: We have data $X = \{x_1, \ldots, x_n\}$. We will assume $x \in \mathbb{R}^d$, but the following discussion doesn't require it. We want to use a mixture model to partition the data into clusters and learn the statistical properties of each cluster.

- Mixture model: Let $p(x|\theta)$ be a probability distribution on the support of x, for example, a distribution on $\mathbb{R}^d$. Let $p(\theta)$ be a prior on θ. A mixture model can be used to generate data as follows:

  - Model: $c_i \sim \text{Discrete}(\pi), \qquad x_i \sim p(x|\theta_{c_i})$

  - Priors: $\pi \sim \text{Dirichlet}(\underbrace{\alpha, \ldots, \alpha}_{K \text{ times}}), \qquad \theta_j \overset{iid}{\sim} p(\theta)$

- Gaussian mixture model (GMM): Last time we saw how the GMM results from $\theta_j = \{\mu_j, \Lambda_j\}$, $p(x|\theta_j) = \text{Normal}(\mu_j, \Lambda_j^{-1})$ and $p(\theta) = p(\mu, \Lambda) = \text{Normal}(\mu|m, (c\Lambda)^{-1})\text{Wishart}(\Lambda|a, B)$.

- A key question is, how many clusters are there? Up to now we've been presetting K to a value and assuming we know what that value should be. While one approach is to do cross-validation, the approach we will discuss next uses Bayesian nonparametrics.

- A Bayesian nonparametric (BNP) prior is a prior distribution like any other, except it involves "infinity" in a way that requires extra theoretical analysis. We won't discuss the theory, but only be interested in the practical aspects of a BNP prior called a "Dirichlet process."

- Dirichlet processes (high-level): Imagine we used the prior

$$\pi \sim \text{Dirichlet}(\underbrace{\tfrac{\alpha}{K}, \ldots, \tfrac{\alpha}{K}}_{K \text{ times}}) \tag{1}$$

  The only difference is that we divide each parameter α (some non-negative, preset number) by the dimensionality of π (K). We then have K possible clusters as before, but we let $K \to \infty$. The result is called a Dirichlet process.

- Again, there are two aspects to thinking about this:

    1. One is theoretical, in which it's necessary to make sure things remain well-defined and where we want to understand what π looks like in this case, etc. There is no concern about the fact that we have an infinite number of θ_j sitting around, and that π is infinite dimensional, so we can't actually do this in practice.

    2. Thus, the second aspect is practical. This essentially addresses the question of how we can do inference for a model where there are an infinite number of parameters. We clearly can't keep an infinite number of things in memory, so this issue boils down to finding equivalent representations that we can use that don't require this infinite number of objects.

- We only focus on the practical and discuss an equivalent representation. Proving equivalence is itself a theoretical question, so #1 and #2 aren't mutually exclusive. We'll give some hints as well about the theoretical ideas as it relates to the Gibbs sampling algorithm we derive next.

- We won't discuss the theory in depth, but the following can help with some intuition about what π looks like as $K \to \infty$ and why it is useful.

    1. First, things remain well-defined. Second, the number of dimensions of π that have probability greater than some small number will be small. Virtually all dimensions will have vanishingly small probabilities and technically an infinite number will have probability equal to zero. Therefore, even though there are an infinite number of clusters, for a finite amount of data only a small number (e.g., 10) will actually be used.

    2. Since the number is variable and learned as part of inference, the Dirichlet process in some sense uncovers the "correct" number of clusters. (The use of "correct" is not at all part of the theory!) The posterior distribution will return a number of "occupied" clusters that is dependent on the data set used.

    3. Ultimately, the *theory* is only used to argue why this prior will be *useful* for our purposes of not presetting K, and not why it's the "correct" way to learn K.

    4. Dirichlet processes are sometimes referred to as sparsity-promoting priors for this reason.

- Again, our concern is practical, so we are interested in how to infer $(\pi, \theta_1, \ldots, \theta_K, c_1, \ldots, c_n)$ as $K \to \infty$.

- We will show how marginalization (of π) can save the day in an MCMC Gibbs sampling setup. Before doing this, notice the difference in advance:

    - In EM, we used "de-marginalization" to help. That is, we wanted to maximize a marginal distribution over some variables and we did this by adding new variables such that they produced the correct marginal.

    - In this lecture, we will have difficulty doing inference over all the model variables, so we will integrate out one of them *permanently* and only sample the remaining ones.

    - Generally speaking, while EM uses one direction, MCMC finds both directions useful.

- Before we derive the generic Gibbs sampling algorithm for the mixture model where $K \to \infty$, let's review the Gibbs sampler for the full model with π from a finite Dirichlet prior with repeated parameter α/K (and so $K < \infty$).

- Recall the Gibbs sampling procedure (in the context of this model)

    1. For each iteration:

        (a) Sample each c_i given the most recent π, θ and other c_{-i}

        (b) Sample each θ_i given the most recent c, π and θ_{-i}

        (c) Sample π given the most recent c and θ

- We use the subscript $-i$ to indicate all variable indexes except for the ith one. We sample each variable from their conditional posteriors. These are:

Gibbs sampling for the finite Dirichlet mixture model

- Sample c_i: The conditional posterior $p(c_i|\pi, \theta, c_{-i}, X)$ does not depend on c_{-j} and X_{-i}. Thus

$$\begin{aligned} p(c_i = j|\pi, x_i, \theta) &\propto p(x_i|\theta, c_i = j)p(c_i = j|\pi) \\ &= \frac{p(x_i|\theta_j)\pi_j}{\sum_{\ell=1}^{K} p(x_i|\theta_\ell)\pi_\ell} \end{aligned} \tag{2}$$

  And so we calculate how likely each cluster is for the data point considered x_i, and pre-weight this be how likely the cluster is in general. This gives a K dimensional vector that we normalize and then use as the parameter of a Discrete distribution to sample a new index value for c_i.

- Sample θ_j: The conditional posterior $p(\theta_j|\theta_{-j}, c, \pi, X)$ only depends on the data in X that has been assigned to cluster j. We can write this as,

$$\begin{aligned} p(\theta_j|X, c) &\propto p(X|\theta_j, c)p(\theta_j) \\ &\propto \left[\prod_{i=1}^{n} p(x_i|\theta_j)^{\mathbb{1}(c_i=j)}\right] p(\theta_j) \end{aligned} \tag{3}$$

  This is problem-specific and the prior is often chosen to be conjugate to the likelihood so we can calculate the posterior in closed form and sample from it.

- Sample π: This is a simple Dirichlet prior—multinomial likelihood setting. The conditional posterior of π only depends on c. Therefore,

$$\begin{aligned} p(\pi|c) &\propto p(c|\pi)p(\pi) \\ &\propto \left[\prod_{i=1}^{n} p(c_i|\pi)\right] p(\pi) \\ &\propto \left[\prod_{i=1}^{n}\prod_{j=1}^{K} \pi_j^{\mathbb{1}(c_i=j)}\right]\left[\prod_{j=1}^{K} \pi_j^{\frac{\alpha}{K}-1}\right] \\ &\propto \prod_{j=1}^{K} \pi_j^{\frac{\alpha}{K}+\sum_{i=1}^{n}\mathbb{1}(c_i=j)-1} \end{aligned} \tag{4}$$

Define $n_j = \sum_{i=1}^{n} \mathbb{1}(c_i = j)$. Then the posterior is

$$p(\pi|c) = \text{Dirichlet}(\tfrac{\alpha}{K} + n_1, \ldots, \tfrac{\alpha}{K} + n_K) \tag{5}$$

- Now we need to address how to sample π and each θ_j as $K \to \infty$. This will be done by integrating out π, which will produce a new method for sampling each c_i.

Discussion on sampling θ

- We will integrate out π in the next section. Therefore we want to sample each θ_j from its conditional posterior given that π is integrated out. In this case, the conditional posterior distribution has exactly the same form as before

$$\begin{aligned} p(\theta_j|X, c, \theta_{-j}) &\propto p(X|\theta_j, c)p(\theta_j) \\ &\propto \left[\prod_{i=1}^{n} p(x_i|\theta_j)^{\mathbb{1}(c_i=j)}\right] p(\theta_j) \end{aligned} \tag{6}$$

- Therefore, the posterior distribution is calculated exactly the same and then θ_j can be sampled from its conditional posterior. However, now since there are an infinite number of θ_j (since $j \in \{1, 2, 3, \ldots\}$, we can't actually do this). We can break this process down into two categories:

  1. $n_j > 0$. There can only be a finite number of j such that this is the case (since the amount of data is finite). Therefore, we definitely need to sample these indices from their conditional posterior.

  2. $n_j = 0$. There are an infinite number of these. We can't actually sample them, however, a crucial observation is that, for a j such that $n_j = 0$, $\mathbb{1}(c_i = j) = 0$ for all i.

     - Therefore, $p(\theta_j|X, c, \theta_{-j}) \propto p(\theta_j)$ for these j. That is, the conditional posterior is equal to the prior (there is no data in cluster j, so there is no prior-to-posterior update!).
     - So even though we can't sample an infinite number of times from this distribution, we *do* know the distribution we *would* sample from if we could.
     - In this case, we would sample an infinite number of i.i.d. samples from the prior $p(\theta)$. This fact will come in handy later.

Discussion on dealing with π by integrating it out

- The solution to working with an infinite dimensional π is to integrate it out (or marginalize it) from the model. Notice that this is the opposite solution as EM, where we expand the model by adding more variables.

- In other words, before we were Gibbs sampling from $p(\pi, c, \theta|X)$. Now we want to Gibbs sample from

$$p(c, \theta|X) = \int p(\pi, c, \theta|X)d\pi = \frac{p(x|c, \theta)p(\theta)}{p(x)} \int p(c|\pi)p(\pi)d\pi \tag{7}$$

- We already showed how to sample θ in this model (exactly the same as before). Next we need to show how to sample each c_i. However, before doing that we briefly discuss what the difference between these two Gibbs samplers is in more detail.
    - Imagine we obtained a large set of samples of the set of model variables $\{\pi^{(t)}, c^{(t)}, \theta^{(t)}\}$ using Gibbs sampling of $p(\pi, c, \theta|X)$. The samples are this triplet for many (say 1000) different values of t as discussed more generally in an earlier lecture.
    - Then, imagine that we threw away each $\pi^{(t)}$ and only kept $\{c^{(t)}, \theta^{(t)}\}$
    - Statistically, the set of "thinned" samples we would have would be identical to if we sampled this pair from the marginalized model $p(c, \theta|X)$.
    - Therefore, if we directly sample this pair of variables from the marginal $p(c, \theta|X)$, we are not sampling from a different model, but only sampling a subset of variables from the same original model.
    - Often (such as in this case) this subset of variables is sufficient to give us information about all we might care about—after all, $\pi^{(t)}$ would only look like the histogram of $c^{(t)}$.

Sampling c_i

- We go back to a finite K, do some calculations, and then let $K \to \infty$.
- Since we integrate out π, sampling c_i turns out to be different from before. By Bayes rule,

$$p(c_i = j|X, \theta, c_{-i}) \;\propto\; \underbrace{p(X|c_i = j, \theta)}_{= p(x_i|\theta_j) \text{ as before}} \underbrace{p(c_i = j|c_{-i})}_{= ?} \tag{8}$$

- The derivation of $p(c_i = j|c_{-i})$ is the main new calculation we need. The result is very interpretable, but requires the following derivation,

$$\begin{aligned}
p(c_i = j|c_{-i}) &= \int p(c_i = j|\pi)p(\pi|c_{-i})d\pi \\
&= \int \pi_j \cdot \text{Dirichlet}(\pi|\tfrac{\alpha}{K} + n_1^{(-i)}, \ldots, \tfrac{\alpha}{K} + n_K^{(-i)})d\pi \\
&\quad \longrightarrow \left(n_j^{(-i)} = \sum_{s \neq i} \mathbb{1}(c_s = j)\right) \\
&= \int \pi_j \frac{\Gamma(\alpha + n - 1)}{\prod_\ell \Gamma(\frac{\alpha}{K} + n_\ell^{(-i)})} \prod_{\ell=1}^{K} \pi_\ell^{\frac{\alpha}{K} + n_\ell^{(-i)} - 1} d\pi \\
&= \frac{\Gamma(\alpha + n - 1)}{\prod_\ell \Gamma(\frac{\alpha}{K} + n_\ell^{(-i)})} \underbrace{\int \pi_j^{\frac{\alpha}{K} + n_j^{(-i)} + 1 - 1} \prod_{\ell \neq j} \pi_\ell^{\frac{\alpha}{K} + n_\ell^{(-i)} - 1} d\pi}_{= \frac{\Gamma\left(\frac{\alpha}{K} + n_j^{(-i)} + 1\right) \prod_{\ell \neq j} \Gamma\left(\frac{\alpha}{K} + n_\ell^{(-i)}\right)}{\Gamma(\alpha + n)}} \\
&= \frac{\Gamma(\alpha + n - 1)}{\Gamma(\alpha + n)} \frac{\Gamma(\frac{\alpha}{K} + n_j^{(-i)} + 1)}{\Gamma(\frac{\alpha}{K} + n_j^{(-i)})}
\end{aligned} \tag{9}$$

- We solved the integral by recognizing that this is the normalizing constant for a Dirichlet distribution with the parameterization indicated in the equation.

- To complete the calculation, we use the property that $\Gamma(y) = (y-1)\Gamma(y-1)$. Applying this equality to $\Gamma(\alpha + n)$ and $\Gamma(\frac{\alpha}{K} + n_j^{(-i)} + 1)$ in the equation above, we find that

$$p(c_i = j | c_{-j}) \;=\; \frac{\frac{\alpha}{K} + n_j^{(-i)}}{\alpha + n - 1} \tag{10}$$

- As a result,

$$p(c_i = j | X, \theta, c_{-i}) \;\propto\; p(x_i|\theta_j) \left(\frac{\frac{\alpha}{K} + n_j^{(-i)}}{\alpha + n - 1} \right) \tag{11}$$

Letting $K \to \infty$

- The problem is that as $K \to \infty$, more and more $n_j^{(-i)} = 0$. When $n_j^{(-i)} > 0$, there's no problem. We discuss these two cases below.

- Case $n_j^{(-i)} > 0$: In the limit $K \to \infty$,

$$p(c_i = j | X, \theta, c_{-i}) \;\propto\; p(x_i|\theta_j) \frac{n_j^{(-i)}}{\alpha + n - 1} \tag{12}$$

- Case $n_j^{(-i)} = 0$: In the limit $K \to \infty$,

$$p(c_i = j | X, \theta, c_{-i}) \;\propto\; 0 \tag{13}$$

- Does this mean that c_i is limited to the existing clusters? No! Instead of asking the probability $c_i = j$ for a particular j in the case where $n_j^{(-i)} = 0$, we ask

$$\begin{aligned} p(c_i = new | X, \theta, c_{-i}) \;&=\; \sum_{j\,:\,n_j^{(-i)}=0} p(c_i = j | X, \theta, c_{-i}) && (14) \\ &\propto\; \lim_{K\to\infty} \sum_{j\,:\,n_j^{(-i)}=0} p(x_i|\theta_j) \frac{\alpha/K}{\alpha + n - 1} && (15) \\ &\propto\; \lim_{K\to\infty} \frac{\alpha}{\alpha + n - 1} \sum_{j\,:\,n_j^{(-i)}=0} \frac{p(x_i|\theta_j)}{K} && (16) \end{aligned}$$

- What is this last limit? Remember that $\theta_j \overset{iid}{\sim} p(\theta)$ for the infinite number of j such that $n_j^{(-i)} = 0$. We can say that as $K \to \infty$ this *Monte Carlo integral* converges to the true integral it approximates when K is finite. Therefore,

$$\lim_{K\to\infty} \sum_{j\,:\,n_j^{(-i)}=0} \frac{p(x_i|\theta_j)}{K} \;=\; \mathbb{E}[p(x_i|\theta)] \;=\; \int p(x_i|\theta) p(\theta) d\theta \tag{17}$$

- Aside about Monte Carlo integration: Imagine we want to calculate $\mathbb{E}[p(x_i|\theta)] = \int p(x_i|\theta)p(\theta)d\theta$. If this integral is intractable, approximate one way is to sample $\theta_j \overset{iid}{\sim} p(\theta)$ for $j = 1, \ldots, K$ times and approximate $\mathbb{E}[p(x_i|\theta)] \approx S_K := \frac{1}{K}\sum_{j=1}^{K} p(x_i|\theta_j)$. Then $\lim_{K\to\infty} S_K = \mathbb{E}[p(x_i|\theta)]$. Next, what happens if we throw away a *finite* number of the evaluations $p(x_i|\theta_j)$ but still divide by K? In this case, nothing changes as $K \to \infty$ and the convergence is still to the same integral.

- To summarize, we have shown that

$$c_i = \begin{cases} j & \text{w.p. } \propto p(x_i|\theta_j)\frac{n_j^{(-i)}}{\alpha+n-1} \text{ if } n_j^{(-i)} > 0 \\ \text{a new index} & \text{w.p. } \propto \frac{\alpha}{\alpha+n-1}\int p(x_i|\theta)p(\theta)d\theta \end{cases}$$

- Finally, if we pick a new index, the question is, what index did we pick? The answer is "who cares"!? In the end, there is a 1-to-1 mapping between the new index j' and new variable $\theta_{j'}$. So we only need to pick the new variable. This is not trivial to prove how to do, but the solution is, if $c_i = j'$ and j' is a new index (i.e., $n_{j'}^{(-i)} = 0$), then

$$\theta_{j'} \mid \{c_i = j'\} \sim p(\theta|x_i) \tag{18}$$

- This gives a general algorithm for sampling from the posterior of a Dirichlet process mixture model. Specific problems that need to be addressed depending on what type of mixture it is (e.g., Gaussian mixture) are calculating the posterior of θ given x and the marginal $\int p(x|\theta)p(\theta)d\theta$.

A marginalized sampling method for the Dirichlet process mixture model

- Initialize in some way, e.g., set all $c_i = 1$ for $i = 1, \ldots, n$, and sample $\theta_1 \sim p(\theta)$.

- At iteration t, re-index clusters to go from 1 to $K^{(t-1)}$, where $K^{(t-1)}$ is the number of occupied clusters after the previous iteration. Sample all variables below using the most recent values of the other variables.

  1. For $i = 1, \ldots, n$
     (a) For all j such that $n_j^{(-i)} > 0$, set
     $$\hat{\phi}_i(j) = p(x_i|\theta_j)n_j^{(-i)}/(\alpha + n - 1)$$
     (b) For a new value j', set
     $$\hat{\phi}_i(j') = \frac{\alpha}{\alpha + n - 1}\int p(x_i|\theta)p(\theta)d\theta$$
     (c) Normalize $\hat{\phi}_i$ and sample the index c_i from a discrete distribution with this parameter.
     (d) If $c_i = j'$, generate $\theta_{j'} \sim p(\theta|x_i)$
  2. For $j = 1, \ldots, K^{(t)}$ generate
     $$\theta_j \sim p(\theta|\{x_i : c_i = j\})$$

  (K_t = # non-zero clusters that are re-indexed after completing Step 1)

Comments:

1. There is a lot of bookkeeping with this algorithm that can very easily lead to coding issues.
2. $n_j^{(-i)}$ needs to be updated after each c_i is sampled. This means that a new cluster is possibly created, or destroyed, after each i in Step 1. It is created if c_i picks a new j'.
3. A cluster is destroyed if the *previous* value of c_i was the *only* instance of this value in the data set. When we check for the new value of c_i, we erase the value previously assigned to it, in which case the cluster no longer exists because x_i was the only observation in that cluster. Then, if c_i joins an existing clusters, the number of clusters is reduced by one. If c_i still creates a new cluster j', a new value of $\theta_{j'}$ must be generated and so the θ variable is still changed.
4. $n_j^{(-1)}$ needs to always be up to date. When a new c_i is sampled, these counts need to be updated.
5. Therefore the "new value j'" in Step 1b can be changing (e.g., for one i, $j' = 10$, for the next i perhaps $j' = 11$ if the previous c_i picked j' when $j' = 10$)
6. Keeping indexing correct is also crucial to getting things to work. After each iteration, it would be good to re-index clusters to start at 1 and increase one-by-one from there.

Generative process

- We've derived a Gibbs sampler for the marginal Dirichlet process. How do we represent the *generative process* that corresponds to this model?

1. Generate an infinite sequence $\theta_1, \theta_2, \theta_3, \ldots$, where $\theta_j \overset{iid}{\sim} p(\theta)$.
2. For the nth observation, define $K_{n-1} = \#\text{unique}(c_1, \ldots, c_{n-1})$ and draw

$$c_n \sim \sum i = 1^{n-1} \frac{1}{\alpha + n - 1} \delta_{c_i} + \frac{\alpha}{\alpha + n - 1} \delta_{(K_{n-1}+1)}$$

3. Generate data $x_n \sim p(x|\theta_{c_n})$.

- Can we say something about the assumptions of this prior? One things we can say is what K_n looks like. This is the number of unique clusters in a data set of size n generated from this model. First, we know that

$$K_n = \sum_{i=1}^{n} \mathbb{1}(c_i \neq c_{j<i}).$$

What is $\mathbb{E}[K_n]$?

$$\mathbb{E}[K_n] = \sum_{i=1}^{n} \mathbb{E}[\mathbb{1}(c_i \neq c_{j<i})] = \sum_{i=1}^{n} Pr(c_i \neq c_{j<i})$$

We can just look at the generative process: We know that $\sum_{i=1}^{n} Pr(c_i \neq c_{j<i}) = \frac{\alpha}{\alpha+n-1}$, so

$$\mathbb{E}[K_n] = \sum_{i=1}^{n} Pr(c_i \neq c_{j<i}) = \sum_{i=1}^{n} \frac{\alpha}{\alpha + n - 1} \approx \alpha \ln(\alpha + n - 1).$$

- One potentially concerning aspect of this prior is that c_i only depends on $c_{1:i-1}$ to generate data, but when we do inference we condition on c_{-i}. Is the algorithm wrong? If not, how do we reconcile this apparent contradiction?

- The answer relates to the concept of *exchangeability*. To this end, we ask what is

$$p(c_1, \ldots, c_n) = p(c_1) \prod_{i=2}^{n} p(c_i | c_1, \ldots, c_{i-1})$$

We can again read these probabilities from the generative process. Let $n_k^{(n)} = \sum_{i=1}^{n} \mathbb{1}(c_i = k)$. Then

$$p(c_1, \ldots, c_n) = \frac{\alpha^{K_n} \prod_{k=1}^{K_n} n_k^{(n)}!}{\prod_{i=1}^{n} \alpha + i - 1}$$

The crucial observation is that $p(c_1, \ldots, c_n)$ is *independent of the ordering*. That is, if we define a permutation of the integers, $\rho(i) \in \{1, \ldots, n\}$ for $i = 1, \ldots, n$, then

$$p(c_1, \ldots, c_n) = p(c_{\rho(1)}, \ldots, c_{\rho(n)})$$

Exchangeability is a heavily-researched theoretical area of probability. For our purposes, we simply conclude that we can reorder the data however we like and treat each c_i as if it were the last one when sampling. (It might be tempting to conclude that we could also *not* reorder c_i and only sample conditioned on the previous values. However, the value of c_i will also impact the probabilities of all future c. Rather than figure out how to write the conditional posterior for a strict ordering, we just treat each c_i as if it were the last one and so it's obvious what the conditional prior should be when using Bayes rule.)

Dirichlet process Gaussian mixture model

- For this mixture model, $p(\theta) = p(\mu|\Lambda)p(\Lambda)$, where

$$\Lambda \sim \text{Wishart}(a, B), \qquad \mu|\Lambda \sim \text{Normal}(m, (c\Lambda)^{-1}) \tag{19}$$

- In variational inference we didn't need this linked prior because we were using an approximation that allowed for tractable calculations. With Gibbs sampling, we need to be more rigorously correct and so in order to calculate the marginal distribution $p(x)$, we need to connect the priors.

- Imagine that $x_1, \ldots, x_s$ were assigned to cluster j, then

$$p(\mu_j, \Lambda_j | x_1, \ldots, x_s) = \text{Normal}(\mu_j | m_j', (c_j' \Lambda_j)^{-1}) \text{Wishart}(\Lambda_j | a_j', B_j') \tag{20}$$

where

$$m_j' = \frac{c}{s+c} m + \frac{1}{s+c} \sum_{i=1}^{s} x_i, \qquad c_j' = s + c$$

$$a_j' = a + s, \qquad B_j' = B + \sum_{i=1}^{s} (x_i - \bar{x})(x_i - \bar{x})^T + \frac{s}{as+1} (\bar{x} - m)(\bar{x} - m)^T$$

We define $\bar{x} = \frac{1}{s} \sum_{i=1}^{s} x_i$.

- When we want to sample a new cluster in which x_i is the only observation, we can also use this posterior distribution. We just don't have any sums and $s = 1$. Notice that one of the terms in B'_j ends up equaling zero.

- Next we need to calculate the marginal distribution under the prior

$$p(x) = \int \int p(x|\mu, \Lambda)p(\mu, \Lambda)d\mu d\Lambda \tag{21}$$

- Again, the prior $p(\mu, \Lambda) = p(\mu|\Lambda)p(\Lambda)$ where

$$p(\mu|\Lambda) = \text{Normal}(m, (c\Lambda)^{-1}), \qquad p(\Lambda) = \text{Wishart}(a, B)$$

- Using this prior and a likelihood $p(x|\mu, \Lambda) = \text{Normal}(\mu, \Lambda^{-1})$, the marginal is

$$p(x) = \left(\frac{c}{\pi(1+c)}\right)^{\frac{d}{2}} \frac{|B + \frac{c}{1+c}(x-m)(x-m)^T|^{-\frac{a+1}{2}}}{|B|^{-\frac{a}{2}}} \frac{\Gamma_d\left(\frac{a+1}{2}\right)}{\Gamma_d\left(\frac{a}{2}\right)} \tag{22}$$

- And where

$$\Gamma_d(y) = \pi^{\frac{d(d-1)}{4}} \prod_{j=1}^{d} \Gamma\left(y + \frac{1-j}{2}\right)$$

- Again, d is the dimensionality of x.

- When implementing this, there might be scaling issues. For example, in the fraction x_1/x_2, the terms x_1 and x_2 might be too large for a computer (i.e., "rounded up" to ∞), but their fraction is something "reasonably sized". This is particularly relevant for the fraction

$$\frac{\Gamma_d\left(\frac{a+1}{2}\right)}{\Gamma_d\left(\frac{a}{2}\right)}$$

- An implementation trick is to represent this as

$$\frac{\Gamma_d\left(\frac{a+1}{2}\right)}{\Gamma_d\left(\frac{a}{2}\right)} = \mathrm{e}^{\sum_{j=1}^{d}\left[\ln\Gamma\left(\frac{a+1}{2}+\frac{1-j}{2}\right)-\ln\Gamma\left(\frac{a}{2}+\frac{1-j}{2}\right)\right]}$$

- By first calculating the value in the exponent, we can work with smaller terms (because of the natural log) and they can cancel each other such that the sum is small. The exponent then returns this final value to the correct "space". This trick is often useful when implementing machine learning algorithms that run into scaling issues (when we know that they shouldn't).

EECS E6720 Bayesian Models for Machine Learning

Columbia University, Fall 2016

Lecture 11, 12/1/2016

Instructor: John Paisley

- We next look at a model for sequential data. In this case, we assume we have a single sequence $(x_1, \ldots, x_T)$ where each $x_t \in \{1, \ldots, V\}$. We want to model the sequential information in this data.

- For simplicity, we will assume we have only one sequence for now. However, in reality we often have many sequences. I'll address the simple modifications that can be made to account for this at the end. For now, assume T is very large so we can have plenty of sequential information in this single sequence.

- Often the data is in $\mathbb{R}^d$. That would require a slightly different modeling approach than what we will discuss below. Incidentally, discrete-valued sequences often arise from what are originally continuous-valued sequences. For example, you can take the original data in $\mathbb{R}^d$ and quantize it by first running a clustering algorithm (K-means is the usual one) and then mapping $x_t \in \mathbb{R}^d$ to a value $x_t \in \{1, \ldots, V\}$, where the value indicates which of the V clusters the original x_t was mapped to.

- Therefore, the following discussing on discrete sequences is useful for many different types of sequential data that aren't initially discrete-valued.

- We will next discuss a model for $x = (x_1, \ldots, x_T)$ called a (discrete) hidden Markov model.

Hidden Markov model (HMM)

- First the high-level description. We associate each observation x_t in the sequence x with a hidden "state" s_t. Therefore, there is a sequence $s = (s_1, \ldots, s_T)$ that corresponds to x where each element in s gives the state for the corresponding element in x. For now, the "state" is a totally abstract thing, much like the "cluster" was an abstract concept in the GMM.

- Often one can hope in advance to learn states that correspond to something meaningful (e.g., machine is "working" or "not working"), but for this lecture take a state to be something that is an abstract mathematical concept.

- There are three variables that define a K-state discrete HMM:

1. π : A K-dimensional probability vector used to generate s_1
2. A : A $K \times K$ Markov transition matrix. A_{ij} is the probability of transitioning to state j given that the current state is i
3. B : A $K \times V$ emission matrix. B_{iv} is the probability of observing $x = v$ given its corresponding state $s = i$.

- Model: Using these model variables, we generate data from the HMM as follows. Again, we have a sequence of data $x = (x_1, \ldots, x_T)$ where $x_t \in \{1, \ldots, V\}$. A K-state discrete HMM models this sequence as follows.
  - Generate state sequence: $s_1 \sim \text{Discrete}(\pi), \quad s_t | s_{t-1} \sim \text{Discrete}(A_{s_{t-1},:})$
  - Generate observation sequence: $x_t | s_t \sim \text{Discrete}(B_{s_t,:})$

- Each row in A is a probability distribution that decides what the next state will be. Therefore, $s_{t-1} \in \{1, \ldots, K\}$ simply picks out the correct row of A to use, which is what the notation is indicating. Similarly with B, the current state s_t picks out the probability distribution (i.e., row of B) to use to generate x_t.

- In this sense, a state corresponds to a probability distribution on an observation. It's worth thinking about how this relates to the mixture model. Each row of B is a data-level distribution much like a Gaussian was for the GMM. Each row of A is like a mixing distribution on which "cluster" (or "state" here) to pick to generate an observation. The HMM is like a mixture model where the "clusters" are not changing, but the *distribution* on the clusters is changing according to a first-order Markov process.

Maximum likelihood Expectation-Maximization (ML-EM)

- In most of what follows, our goal is to find a point estimate of $H = \{\pi, A, B\}$ that maximizes the marginal likelihood

$$\ln p(x|H) = \ln \sum_s p(x, s|H) \tag{1}$$

- Comments:
  1. From now on, we'll use H for π, A, B when they are all conditioned on.
  2. The sum is over all T-length sequences s. Since there are K^T possible sequences, this sum can't be directly done.
  3. However, with EM we never actually need to calculate the marginal explicitly. We only need to be able to write out the joint likelihood over x and s.
  4. We mention that the sum can be performed in a clever way using outputs from either the "forward" or the "backward" algorithms, both of which we will discuss later.

- Joint likelihood: As usual, we have to start here before we can do anything else.

$$\begin{aligned} p(x, s|H) &= p(x|s, B)p(s|A, \pi) \\ &= \left[\prod_{t=1}^{T} p(x_t|s_t, B)\right]\left[p(s_1|\pi)\prod_{t=2}^{T} p(s_t|s_{t-1}, A)\right] \\ &= \left[\prod_{t=1}^{T}\prod_{k=1}^{K}\prod_{v=1}^{V} B_{kv}^{\mathbb{1}(s_t=k)\mathbb{1}(x_t=v)}\right]\left[\prod_{k=1}^{K}\pi_k^{\mathbb{1}(s_1=k)}\right]\left[\prod_{t=2}^{T}\prod_{i=1}^{K}\prod_{j=1}^{K} A_{ij}^{\mathbb{1}(s_{t-1}=i,s_t=j)}\right] \end{aligned} \tag{2}$$

- Comments:

  1. $p(x|s, B)$: Given the state sequence s, x only depends on B. Also, if I know which state each observation came from, then the observations are all independent of each other. This is a result of the model definition. So we can write this likelihood as a product.

  2. $p(s|\pi, A)$: By the chain rule of probability

$$p(s|\pi, A) = p(s_1|\pi, A)\prod_{t=2}^{T} p(s_t|s_1, \ldots, s_{t-1}, \pi, A)$$

     This is *always* true. By the *first-order Markov property* (i.e., the model definition) we can further say that $p(s_t|s_1, \ldots, s_{t-1}, \pi, A) = p(s_t|s_{t-1}, \pi, A)$ and also simplify the conditioning by removing π or A as required.

  3. Notice that we've used indicators again to pick out the correct entries in π, A and B. This is basically always useful for discrete variables/data.

- EM steps: Recall the three main EM steps (the first two are the "E" and the third is the "M" step)

  1. Set $q(s) = p(s|x, H)$
  2. Calculate $\mathcal{L} = \mathbb{E}_q[\ln p(x, s|H)]$
  3. Maximize $\mathcal{L}$ over $H = \{\pi, A, B\}$

- The first step already poses a problem. Recall from the GMM that we had no difficulty learning $q(c)$ where c was the vector of cluster indicators ($c_n = j$ means observation x_n came from cluster j; refer to the previous lecture notes for more details). This is because conditioned on the GMM model variables θ, and the data x, $c_1, \ldots, c_N$ were conditionally independent. Therefore, $p(c|x, \theta) = \prod_{n=1}^{N} p(c_n|x_n, \theta)$ and thus $q(c) = \prod_{n=1}^{N} q(c_n)$ where $q(c_n) = p(c_n|x_n, \theta)$.

- The conditional posterior $p(s|x, H) \neq \prod_{t=1}^{T} p(s_t|x_t, H)$, and so we can't easily solve for $q(s)$ in this way.

- There are K^T different sequences that s could be. In principal we would therefore have to calculate $p(s|x, H) \propto p(x|s, H)p(s|H)$ for each of these. Even though K^T is finite—and so we know that $p(s|x, H)$ is just a multinomial that can be calculated in principal—in practice K^T is way too large to enumerate all values.

- We'll see that we don't actually have to calculate $q(s)$ for each and every sequence s. However, we don't know that yet. Since we can find $q(s)$ in principal, we will address Step 1 above by simply declaring that we've calculated it.

- **Step 1 of EM**: "We have set $q(s) = p(s|x, H)$." Let's pretend this is the case to try and make some progress (and see if we actually don't need this to be the case).

- **Step 2 of EM**: Next take the expectation, $\mathbb{E}_q[\ln p(x, s|H)]$.

$$\begin{aligned}
\mathcal{L} &= \sum_{t=1}^{T}\sum_{k=1}^{K}\sum_{v=1}^{V}\mathbb{E}_q[\mathbb{1}(s_t = k)]\mathbb{1}(x_t = v)\ln B_{kv} && \leftarrow \mathbb{E}_q[\ln p(x|s, B)] \\
&\quad + \sum_{k=1}^{K}\mathbb{E}_q[\mathbb{1}(s_1 = k)]\ln \pi_k && \leftarrow \mathbb{E}_q[\ln p(s_1|\pi)] \\
&\quad + \sum_{t=2}^{T}\sum_{i=1}^{K}\sum_{j=1}^{K}\mathbb{E}_q[\mathbb{1}(s_{t-1} = i, s_t = j)]\ln A_{ij} && \leftarrow \mathbb{E}_q[\ln p(s_2, \ldots, s_T|A)] \qquad (3)
\end{aligned}$$

- There are two key expectations here.

  1. $\mathbb{E}_q[\mathbb{1}(s_t = k)]$: The expectation of an indicator of an event is the probability of that event, $\mathbb{E}_q[\mathbb{1}(s_t = k)] = q(s_t = k)$. To see this,

$$\begin{aligned}
\sum_s q(s)\mathbb{1}(s_t = k) &= \sum_{s_1=1}^{K}\cdots\sum_{s_{t-1}=1}^{K}\sum_{s_{t+1}=1}^{K}\cdots\sum_{s_T=1}^{K} q(s_1, \ldots, s_{t-1}, s_t = k, s_{t+1}, \ldots, s_T) \\
&= p(s_t = k|x, H) \\
&\equiv q(s_t = k) \qquad (4)
\end{aligned}$$

     What is going on here? This expectation is just the *marginal* distribution of $q(s)$ where we integrate (i.e., sum) over every value of $s_{t'}$ except s_t which we set equal to k. In other words, instead of needing to know the entire posterior distribution of a sequence s, we are here only asking for the posterior probability that $s_t = k$ given x and H without regard to what any other value of the sequence is equal to.

  2. $\mathbb{E}_q[\mathbb{1}(s_{t-1} = i, s_t = j)]$: By using exactly the same reasoning as above,

$$\mathbb{E}_q[\mathbb{1}(s_{t-1} = i, s_t = j)] = p(s_{t-1} = i, s_t = j|x, H) \equiv q(s_{t-1} = i, s_t = j) \qquad (5)$$

     Again, we only care about the posterior probability of a specific transition at a specific time, not caring about anything else going on in the sequence.

- We still don't know what these probabilities are, or how to calculate them. However, we have shown that in order to complete the E-step, we never need to know the posterior distribution of the entire sequence. We only need to know marginal posterior distributions on isolated portions of that sequence.

- That is, we don't need to calculate $p(s|x, H)$, only $p(s_t = k|x, H)$ and $p(s_{t-1} = i, s_t = j|x, H)$ for all t, k, i and j. This is still non-trivial, but we'll see that there is an algorithm that let's us get these values quickly called the *forward-backward algorithm*.

- **Step 3 of EM**: Let's pretend we have $p(s_t = k|x, H)$ and $p(s_{t-1} = i, s_t = j|x, H)$ already calculated and quickly take care of the M-step. Using Lagrange multipliers to ensure that the updates for π, A and B are probability distributions, we can maximize

$$\mathcal{L} = \sum_{t,k,v} q(s_t = k)\mathbb{1}(x_t = v) \ln B_{kv} + \sum_k q(s_1 = k) \ln \pi_k + \sum_{t>1,i,j} q(s_{t-1} = i, s_t = j) \ln A_{ij} \quad (6)$$

  as follows:

$$\pi_k = q(s_1 = k) \quad (7)$$

$$A_{ij} = \frac{\sum_{t=2}^{T} q(s_{t-1} = i, s_t = j)}{\sum_{k=1}^{K} \sum_{t=2}^{T} q(s_{t-1} = i, s_t = k)} \quad (8)$$

$$B_{kv} = \frac{\sum_{t=1}^{T} q(s_t = k)\mathbb{1}(x_t = v)}{\sum_{w=1}^{V} \sum_{t=1}^{T} q(s_t = k)\mathbb{1}(x_t = w)} \quad (9)$$

Calculating $q(s_t = k)$ and $q(s_{t-1} = i, s_t = j)$

- So we now just need to find these two marginal distributions. We will do this in a two-step procedure, first defining them in terms of quantities that again we wish we had, and then presenting an algorithm to find those quantities.

- Calculate $q(s_t = k)$: For this problem we want to set

$$\begin{aligned} q(s_t = k) &= p(s_t = k|x, H) \\ &\propto \underbrace{p(x_{t+1}, \ldots, x_T|s_t = k, H)}_{\equiv \beta_t(k)} \underbrace{p(s_t = k|x_1, \ldots, x_t, H)}_{\equiv \alpha_t(k)} \end{aligned} \quad (10)$$

- We make the definitions:

  - $\beta_t(k) = p(x_{t+1}, \ldots, x_T|s_t = k, H)$, the probability of seeing everything to come after step t given that we are in state k at step t.

  - $\alpha_t(k) = p(s_t = k|x_1, \ldots, x_t, H)$, the posterior probability of being in state k at step t given all the data observed up to, and including, that time point.

- We have simply used Bayes rule and define the two terms in the likelihood and prior as β and α. We could have used Bayes rule in other ways, for example using x_{t+2} and later in β and including x_{t+1} in α. The reason we don't is because we wouldn't be able to get things to work out that way. Ultimately, the goal is to use the rules of probability to keep re-writing things until we reach a form where we can actually start plugging in numbers and find solutions.

- Again we've pushed the solution off to a later point, but notice that, if we could find a way to calculate $\alpha_t(k)$ and $\beta_t(k)$, we could then set

$$q(s_t = k) = \frac{\beta_t(k)\alpha_t(k)}{\sum_{j=1}^{K} \beta_t(j)\alpha_t(j)} \tag{11}$$

- Calculate $q(s_{t-1} = i, s_t = j)$: For this one we again use Bayes rule, followed by additional factorizations,

$$\begin{aligned} q(s_{t-1} = i, s_t = j) &= p(s_{t-1} = i, s_t = j | x, H) \\ &\propto p(x_t, \ldots, x_T | s_{t-1} = i, s_t = j, H) p(s_{t-1} = i, s_t = j | x_1, \ldots, x_{t-1}, H) \\ &\propto p(x_t, \ldots, x_T | s_t = j, H) p(s_t = j | s_{t-1} = i, H) p(s_{t-1} = i | x_1, \ldots, x_{t-1}, H) \\ &\propto \underbrace{p(x_{t+1}, \ldots, x_T | s_t = j, H)}_{\equiv \beta_t(j)} \underbrace{p(x_t | s_t = j, H)}_{B_{j,x_t}} \\ &\quad \times \underbrace{p(s_t = j | s_{t-1} = i, H)}_{A_{ij}} \underbrace{p(s_{t-1} = i | x_1, \ldots, x_{t-1}, H)}_{\equiv \alpha_{t-1}(i)} \end{aligned} \tag{12}$$

- Again, we have used Bayes rule in such a way that we can make progress towards solving for this posterior distribution. Notice that two of the terms are directly from the HMM variables. We can directly plug these values in for the most recent iteration of the algorithm. We don't know the other two probabilities, however, notice that they are exactly what we need to know for $q(s_t = k)$. Therefore, we can use the same definitions and whatever algorithm we develop to solve $q(s_t = k)$, we can use the same result to solve $q(s_{t-1} = i, s_t = j)$.

- Imagining that we have α and β, we can then set

$$q(s_{t-1} = i, s_t = j) = \frac{\beta_t(j) B_{j,x_t} A_{ij} \alpha_{t-1}(i)}{\sum_{r=1}^{K} \sum_{s=1}^{K} \beta_t(r) B_{r,x_t} A_{sr} \alpha_{t-1}(s)} \tag{13}$$

- Think of $q(s_{t-1} = i, s_t = j)$ as the (i, j)-th element in a $K \times K$ matrix that gives the probability of this transition. Since there can only be one transition, the sum of probabilities in this matrix has to equal one. The denominator above makes this be the case.

The forward-backward algorithm

- The algorithm that gives us α_t is called the "forward algorithm" while that which gives β_t is the "backward algorithm." They are recursive algorithms, meaning that to learn α_t we need α_{t-1}, and to learn β_t we need β_{t+1} (hence the directions in the name). Since we can find α_1 and β_T, we can solve for all α_t and β_t.

- Forward algorithm: Here we want to learn the K-dimensional vector α_t at time t, where $\alpha_t(k) = p(s_t = k | x_1, \ldots, x_t, H)$. To do this, we first write this as the marginal of a joint probability distribution and then use Bayes rule on the joint distribution,

$$\underbrace{p(s_t = k|x_1, \ldots, x_t, H)}_{\alpha_t(k)} = \sum_{j=1}^{K} p(s_t = k, s_{t-1} = j|x_1, \ldots, x_t) \tag{14}$$

$$\propto \sum_{j=1}^{K} p(x_t|s_t = k, s_{t-1} = j, H) p(s_t = k, s_{t-1} = j|x_1, \ldots, x_{t-1}, H)$$

$$\propto \sum_{j=1}^{K} \underbrace{p(x_t|s_t = k, H)}_{B_{k,x_t}} \underbrace{p(s_t = k|s_{t-1} = j, H)}_{A_{jk}} \underbrace{p(s_{t-1} = j|x_1, \ldots, x_{t-1}, H)}_{\equiv \alpha_{t-1}(j)}$$

- Therefore, we can set

$$\hat{\alpha}_t(k) = B_{k,x_t} \sum_{j=1}^{K} A_{jk} \alpha_{t-1}(j) \tag{15}$$

$$\alpha_t(k) = \frac{\hat{\alpha}_t(k)}{\sum_{j=1}^{K} \hat{\alpha}_t(j)} \tag{16}$$

- We also notice that we can solve

$$q(s_1 = k) = p(s_1 = k|x_1, H) = \alpha_1(k)$$

  exactly using Bayes rule,

$$\alpha_1(k) = \frac{\pi_k B_{k,x_1}}{\sum_{j=1}^{K} \pi_j B_{j,x_1}} \tag{17}$$

  This means we know the starting point for α_1 and can solve all subsequent α_t.

- Backward algorithm: We now want to learn the K-dimensional vector β_t at time t, where $\beta_t(k) = p(x_{t+1}, \ldots, x_T|s_t = k, H)$. We again write this as the marginal of a joint probability distribution. However, this time we won't need Bayes rule.

$$\underbrace{p(x_{t+1}, \ldots, x_T|s_t = k, H)}_{\beta_t(k)} = \sum_{j=1}^{K} p(x_{t+1}, \ldots, x_T, s_{t+1} = j|s_t = k, H) \tag{18}$$

$$= \sum_{j=1}^{K} p(x_{t+1}, \ldots, x_T|s_{t+1} = j, H) p(s_{t+1} = j|s_t = k, H)$$

$$= \sum_{j=1}^{K} \underbrace{p(x_{t+2}, \ldots, x_T|s_{t+1} = j, H)}_{\equiv \beta_{t+1}(j)} \underbrace{p(x_{t+1}|s_{t+1} = j, H)}_{B_{j,x_{t+1}}} \underbrace{p(s_{t+1} = j|s_t = k, H)}_{A_{kj}}$$

- Notice that this time there is no proportionality. We can just set

$$\beta_t(k) = \sum_{j=1}^{K} \beta_{t+1}(j) B_{j,x_{t+1}} A_{kj} \tag{19}$$

- For the first value, we can then set $\beta_T(j) = 1$ for all j then solve for the previous values. Notice that these numbers can become very small as t becomes less and less leading to computer precision issues. After updating each β_t, you can normalize this vector. Notice that this re-scaling will not change any other updated values.

- Log marginal likelihood: To evaluate convergence, we need to calculate

$$\ln p(x_1, \ldots, x_T|H) = \ln \sum_s p(x, s| \ldots, H) \tag{20}$$

- Recall that since there are K^T different sequences s to sum over, we simply aren't going to calculate this marginal directly.

- However, using the chain rule of probability we can make progress. That is,

$$p(x_1, \ldots, x_T|H) = p(x_1|H) \prod_{t=2}^{T} p(x_t|x_1, \ldots, x_{t-1}, H) \tag{21}$$

- Notice that $p(x_t|x_1, \ldots, x_{t-1}, H)$ is actually something we've calculated in the forward algorithm,

$$p(x_t|x_1, \ldots, x_{t-1}, H) = \sum_{k=1}^{K} \hat{\alpha}_t(k) \tag{22}$$

- We see this by simply plugging in what we defined to be $\hat{\alpha}_t(k)$,

$$\begin{aligned} \sum_{k=1}^{K} \hat{\alpha}_t(k) &= \sum_{k=1}^{K} \sum_{j=1}^{K} p(x_t|s_t = k, H) p(s_t = k|s_{t-1} = j, H) p(s_{t-1} = j|x_1, \ldots, x_{t-1}, H) \\ &= p(x_t|x_1, \ldots, x_{t-1}, H) \end{aligned} \tag{23}$$

- Therefore,

$$\ln p(x_1, \ldots, x_T|H) = \sum_{t=1}^{T} \ln \sum_{k=1}^{K} \hat{\alpha}_t(k) \tag{24}$$

- We can literally compute this value "in passing" during the forward algorithm.

Multiple sequences

- Finally, we show (but don't derive) how to modify the algorithm when we have multiple sequences $x^{(1)}, \ldots, x^{(N)}$ of possibly varying length $T_1, \ldots, T_N$. Hopefully this is a straightforward exercise to derive on your own by now.

- Forward-backward: First, given the HMM variables $H = \{\pi, A, B\}$ forward-backward is run *independently* on each sequence to obtain values $\alpha_t^{(n)}(k)$ and $\beta_t^{(n)}(k)$ for the nth sequence.

- $q(s_t^{(n)} = k)$ and $q(s_{t-1}^{(n)} = i, s_t^{(n)} = j)$: For sequence number n, these probabilities are computed using $\alpha_t^{(n)}(k)$ and $\beta_t^{(n)}(k)$ and π, A, B exactly as before.

- Updating π, A and B: Simply sum over the sequences and normalize

$$\pi_k = \frac{1}{N}\sum_{n=1}^{N} q(s_1^{(n)} = k) \tag{25}$$

$$A_{ij} = \frac{\sum_{n=1}^{N}\sum_{t=2}^{T_n} q(s_{t-1}^{(n)} = i, s_t^{(n)} = j)}{\sum_{n=1}^{N}\sum_{k=1}^{K}\sum_{t=2}^{T_n} q(s_{t-1}^{(n)} = i, s_t^{(n)} = k)} \tag{26}$$

$$B_{kv} = \frac{\sum_{n=1}^{N}\sum_{t=1}^{T_n} q(s_t^{(n)} = k)\mathbb{1}(x_t^{(n)} = v)}{\sum_{n=1}^{N}\sum_{w=1}^{V}\sum_{t=1}^{T_n} q(s_t^{(n)} = k)\mathbb{1}(x_t^{(n)} = w)} \tag{27}$$

MAP-EM inference

- In the last part, we will briefly see how small modifications can be made to derive MAP-EM and variational inference algorithms. In both cases we need to define priors on the HMM parameters, π, A and B. Typically conjugacy motivates these to be

$$A_{k,:} \sim \text{Dirichlet}(\alpha), \quad B_{k,:} \sim \text{Dirichlet}(\gamma), \quad \pi \sim \text{Dirichlet}(\kappa). \tag{28}$$

- For MAP-EM, we simply add $\ln p(H) = \ln p(\pi)\prod_k p(A_{k,:})p(B_{k,:})$ to the EM objective. In this case the update to $p(s)$ is identical to the previous one and the only difference is in the updates to π, A and B above, where terms are added in the numerator and denominator.

- I don't write those updates down, but point out that there are many -1's involved, one in the numerator and K of them in the denominator (so a $-K$ in the denominator) resulting from the Dirichlet prior. This indicates that for MAP-EM to be guaranteed to be well-defined, we need to set all Dirichlet parameters ≥ 1. The MAP solution gives a significantly different meaning to the Dirichlet parameters than what we *a priori* think (where sparsity requires them to be < 1).

- MAP for multinomial parameters with Dirichlet priors in general is a problem. Fortunately VI is easy in this case and fixes this interpretability problem. I'll focus on VI below.

Variational inference

- We want to approximate the posterior of the discrete HMM with Dirichlet priors using variational inference. We therefore first need to define a factorized q distribution. We will used

$$q(s, \pi, A, B) = q(s)q(\pi)\prod_{k=1}^{K} q(A_{k,:})q(B_{k,:}). \tag{29}$$

- Notice that we aren't factorizing over the values in the sequence s. However, the same exact expectations of identities show up here as well, so we can follow the same reasoning to arrive at the same algorithm as for ML-EM (very slightly modified as described below).

- Let's pretend we know $q(s)$. Then the following standard (by now) steps can be followed:

$$\begin{aligned} q(A_{k,:}) &\propto \exp\Big\{\sum_{t=2}^{T}\sum_{j=1}^{K} q(s_{t-1}=k, s_t=j)\ln A_{k,j} + (\alpha-1)\ln A_{k,j}\Big\} \\ &= \text{Dirichlet}\Big(\Big[\alpha + \sum_{t=2}^{T} q(s_{t-1}=k, s_t=j)\Big]_{j=1}^{K}\Big) \end{aligned} \tag{30}$$

- Where the $[\,]$ notation indicates a vector ranging over values of j. Similarly,

$$q(B_{k,:}) = \text{Dirichlet}\Big(\Big[\gamma + \sum_{t=1}^{T} q(s_t=k)\mathbb{1}(x_t=v)\Big]_{v=1}^{V}\Big) \tag{31}$$

$$q(\pi) = \text{Dirichlet}\Big(\Big[\kappa + q(s_1=k)\Big]_{k=1}^{K}\Big) \tag{32}$$

- If there are multiple sequences, the s variables have a sequence index and all q distributions above include an extra summation over the sequence index.

- What about $q(s)$? We won't go into all details, but simply point out that the following changes can be made to the forward-backward algorithm.

$$A_{ij} \to e^{\mathbb{E}_q[\ln A_{ij}]}, \quad B_{kv} \to e^{\mathbb{E}_q[\ln B_{kv}]}, \quad \pi_i \to e^{\mathbb{E}_q[\ln \pi_i]} \tag{33}$$

- This follows the familiar pattern. We also recognize that all expectations are using Dirichlet q distributions, so are of the form $\psi(\cdot) - \psi(\sum \cdot)$. It doesn't impact the algorithm that these modifications don't sum to one and so aren't probability distributions anymore—they're simply plugged in.

- Finally, to calculate the variational objective function $\mathcal{L}$, we note that

$$\begin{aligned} \mathcal{L} &= \int q(\pi)\ln\frac{p(\pi)}{q(\pi)}d\pi + \sum_{i=1}^{K}\int q(A_{i,:})\ln\frac{p(A_{i,:})}{q(A_{i,:})}dA_{i,:} + \sum_{k=1}^{K}\int q(B_{k,:})\ln\frac{p(B_{k,:})}{q(B_{k,:})}dB_{k,:} \\ &\quad + \sum_{t=1}^{T}\ln\sum_{k=1}^{K}\hat{\alpha}_t(k) \end{aligned} \tag{34}$$

- Notice that the term involving $\hat{\alpha}_t(k)$ is just like before, only we've replaced A with $e^{\mathbb{E}_q[\ln A]}$, etc., when calculating it. The other three terms, are the negative KL-divergences between the approximate posterior q and prior p of each variable.

- Finally, to calculate $\mathcal{L}$ when there are multiple sequences, the term involving $\hat{\alpha}_t$ has an additional sequence index and a second sum is done over this index. The first line is unchanged.

EECS E6720 Bayesian Models for Machine Learning

Columbia University, Fall 2016

Lecture 12, 12/8/2016

Instructor: John Paisley

Non-negative matrix factorization

- Goal: We have a $M \times N$ data matrix X where $X_{ij} \geq 0$. We want to approximate this with a product of two non-negative matrices,
  - W: a $M \times K$ matrix, $W_{ij} \geq 0$
  - V: a $K \times N$ matrix, $V_{kj} \geq 0$
  - $X_{ij} \approx (WV)_{ij} = \sum_{k=1}^{K} W_{ik}V_{kj}$

- The questions (as usual) are two-fold:
  1. What objective do we use to measure approximation quality?
  2. What algorithm do we run to learn W and V?

- Lee & Seung's NMF paper is a major (non-Bayesian) step in this direction. They propose two objectives:

$$\text{Squared error: } \arg\min_{W,V} \sum_{i,j} (X_{ij} - (WV)_{ij})^2 \tag{1}$$

$$\text{Divergence penalty: } \arg\min_{W,V} \sum_{i,j} X_{ij} \ln \frac{X_{ij}}{(WV)_{ij}} - X_{ij} + (WV)_{ij} \tag{2}$$

- The key contribution is their fast multiplicative update rules for optimizing these objective functions over W and V.

- For example, the algorithm for the Divergence penalty is to update W and V as follows

$$V_{kj} \leftarrow V_{kj} \frac{\sum_{i=1}^{M} W_{ik} X_{ik}/(WV)_{ij}}{\sum_{i=1}^{M} W_{ik}} \tag{3}$$

$$W_{ik} \leftarrow W_{ik} \frac{\sum_{j=1}^{N} V_{kj} X_{ij}/(WV)_{ij}}{\sum_{j=1}^{N} V_{kj}} \tag{4}$$

- The NMF paper shows that these two updates produce new values of W and V that are monotonically decreasing the Divergence penalty. Similar updates are derived for the Squared error penalty.

- This paper is worth studying very closely for its own sake. We discuss it here to make connections with Bayesian methods. For example, what is this Divergence penalty doing? Imagine we had the following model.

- Model: We have a $M \times N$ matrix X of non-negative values. We model this as

$$X_{ij} \sim \text{Poisson}\left(\sum_{k=1}^{K} W_{ik} V_{kj}\right) \tag{5}$$

  where W_{ik} and V_{kj} are non-negative model variables.

- Recall the Poisson distribution: $X \in \{0, 1, 2, \dots\}$ is Poisson distributed with parameter λ if

$$p(X = x|\lambda) = \frac{\lambda^x}{x!}\mathrm{e}^{-\lambda}, \qquad \mathbb{E}x = Var(x) = \lambda \tag{6}$$

- Joint likelihood: As usual, write this out first before doing anything else

$$\begin{aligned} p(X|W, V) &= \prod_{i=1}^{M}\prod_{j=1}^{N} \text{Poisson}(X_{ij}|(WV)_{ij}) \\ &= \prod_{i=1}^{M}\prod_{j=1}^{N} \frac{(WV)_{ij}^{X_{ij}}}{X_{ij}!}\mathrm{e}^{-(WV)_{ij}} \end{aligned} \tag{7}$$

- Maximum likelihood: Next, consider maximum likelihood for this model. We want to find

$$\arg\max_{W,V} \ \ln p(X|W, V) = \sum_{i,j} X_{ij} \ln(WV)_{ij} - (WV)_{ij} + \text{constant wrt } W, V \tag{8}$$

- Notice that this objective is simply the negative of the Divergence penalty of NMF. Since the NMF algorithm monotonically increases this objective, this algorithm performs maximum likelihood for the Poisson factorization model described above. See the NMF paper for the original proof. We'll next show that the multiplicative updates are equivalent to the EM algorithm.

- This is another perspective of the NMF algorithm for the Divergence penalty. It's not the one they take in the paper, and it's not required for someone to take it either. However, we'll see that by thinking in terms of probability models, we can introduce prior structure and use other optimization (i.e., inference) algorithms to approximate the posterior distribution of these variables.

- Imagine deriving a gradient algorithm for Equation (8). This would be very difficult. This motivated the usefulness of the multiplicative updates of the original NMF paper.

Maximum likelihood EM for Poisson matrix factorization

- Our goal is to derive an EM algorithm for the Poisson likelihood model described above that will have simple and closed form updates. Therefore, we need to find an appropriate latent variable to add that has the correct marginal distribution. To this end, we digress into a property about sums of Poisson distributed random variables.

- Let $Y^{(k)} \sim \text{Poisson}(\lambda_k)$ independently for $k = 1, \ldots, K$. Then define $X = \sum_{k=1}^{K} Y^{(k)}$. It follows that the marginal distribution of X is $X \sim \text{Poisson}(\sum_{k=1}^{K} \lambda_k)$.

  Proof: A random variable Y with distribution $p(Y)$ is uniquely identified by its moment generating function, $\mathbb{E}_p[e^{tY}]$. For a Poisson(λ) distribution

$$\mathbb{E}_p[e^{tY}] = \sum_{y=0}^{\infty} e^{ty} \frac{\lambda^y}{y!} e^{-\lambda} = e^{-\lambda} e^{\lambda e^t} \sum_{y=0}^{\infty} \frac{(\lambda e^t)^y}{y!} e^{-\lambda e^t} = e^{-\lambda(1-e^t)} \tag{9}$$

  The sum equals 1 because it is over a Poisson(λe^t) distribution. Recognizing that $Y^{(k)}$ are generated independently, the following completes the proof,

$$\mathbb{E}[e^{tX}] = \mathbb{E}[e^{t\sum_{k=1}^{K} Y^{(k)}}] = \prod_{k=1}^{K} \mathbb{E}[e^{tY^{(k)}}] = \prod_{k=1}^{K} e^{-\lambda_k(1-e^t)} = e^{-(\sum_{k=1}^{K} \lambda_k)(1-e^t)} \tag{10}$$

  The proof is complete because we calculated these moment generating functions in the context of $Y^{(k)}$ and found that their sum has exactly the same generating function as a Poisson($\sum_{k=1}^{K} \lambda_k$) random variable. Therefore we can say that $\sum_{k=1}^{K} Y^{(k)}$ has this same distribution, which we will show is the correct marginal distribution.

- Extended model: For each element (i, j) in X, we now use the generative model

$$Y_{ij}^{(k)} \sim \text{Poisson}(W_{ik} V_{kj}), \quad X_{ij} | \vec{Y}_{ij} \sim \mathbb{1}\Big(X_{ij} = \sum_{k=1}^{K} Y_{ij}^{(k)}\Big) \tag{11}$$

  Notice that the "distribution" on X_{ij} puts all of its probability mass on the event $X_{ij} = \sum_{k=1}^{K} Y_{ij}^{(k)}$. Therefore, there's nothing random, but we still can say what $p(X|Y)$ is. Also notice that the marginal of X_{ij}—i.e., the distribution on X_{ij} *not* conditioned on Y (so we integrate Y out)—is $X_{ij} \sim \text{Poisson}(\sum_{k=1}^{K} W_{ik} V_{kj})$ as required.

- Joint likelihood: We now have that the joint likelihood including the extra variables Y is

$$p(X, Y | W, V) = \prod_{i=1}^{M} \prod_{j=1}^{N} p(X_{ij} | \vec{Y}_{ij}) \prod_{k=1}^{K} p(Y_{ij}^{(k)} | W_{ik}, V_{kj}) \tag{12}$$

- EM equation: We use the joint likelihood to set up the EM equation

$$\ln p(X|W, V) = \sum_{Y} q(Y) \ln \frac{p(X, Y|W, V)}{q(Y)} + \sum_{Y} q(Y) \ln \frac{q(Y)}{p(Y|X, W, V)} \tag{13}$$

  The q distribution is on all values $Y_{ij}^{(k)}$ for $i = 1, \ldots, M$, $j = 1, \ldots, N$ and $k = 1, \ldots, K$. Hopefully we can simplify this, otherwise we won't get very far.

- E-Step (part 1): The first part of the E-Step is to set $q(Y) = p(Y|X, W, V)$. We simply use Bayes rule and see what progress we can make.

$$\begin{aligned} p(Y|X,W,V) &\propto p(X|Y,W,V)p(Y|W,V) \\ &\propto p(X|Y)p(Y|W,V) \\ &\propto \prod_{i=1}^{M}\prod_{j=1}^{N} p(X_{ij}|\vec{Y}_{ij})p(\vec{Y}_{ij}|W,V) \end{aligned} \tag{14}$$

We have used the conditional independence defined by the model to write this expression. Notice that for each (i, j) pair, we have a "mini" Bayes rule embedded in this problem. That is

$$p(Y|X,W,V) = \prod_{i=1}^{M}\prod_{j=1}^{N} \frac{p(X_{ij}|\vec{Y}_{ij})p(\vec{Y}_{ij}|W,V)}{p(X_{ij}|W,V)} = \prod_{i=1}^{M}\prod_{j=1}^{N} p(\vec{Y}_{ij}|X_{ij},W,V) \tag{15}$$

Thus we know that $q(Y) = \prod_{i,j} q(\vec{Y}_{ij})$ and $q(\vec{Y}_{ij}) = p(\vec{Y}_{ij}|X_{ij}, W, V)$. We just need to see if we can calculate this for one (i, j) pair.

- In words, this is saying that if someone else generates $Y_{ij}^{(k)} \sim \text{Poisson}(W_{ik}V_{kj})$, calculates $X_{ij} = \sum_{k=1}^{K} Y_{ij}^{(k)}$ and then shows me X_{ij} and W and V, what is my posterior belief about $\vec{Y}_{ij}$?

- Again, we solve Bayes rule for this sub-problem. The result is one of the favorites of probability, here unfortunately derived using bogged-down notation from this problem.

$$\begin{aligned} p(\vec{Y}_{ij}|X_{ij},W,V) &= \frac{p(X_{ij}|\vec{Y}_{ij})p(\vec{Y}_{ij}|W,V)}{p(X_{ij}|W,V)} \\ &= \frac{\mathbb{1}(X_{ij} = \sum_{k=1}^{K} Y_{ij}^{(k)}) \prod_{k=1}^{K} \frac{(W_{ik}V_{kj})^{Y_{ij}^{(k)}}}{Y_{ij}^{(k)}!} e^{-W_{ik}V_{kj}}}{\frac{(WV)_{ij}^{X_{ij}}}{X_{ij}!} e^{-(WV)_{ij}}} \end{aligned} \tag{16}$$

In the numerator, we've multiplied the indicator distribution on X_{ij} with the product of K independent Poisson distributions on $\vec{Y}_{ij}$. In the denominator, we use the fact that the marginal distribution on X_{ij} is Poisson$(\sum_{k=1}^{K} W_{ik}V_{kj})$, which we proved above. We simply write out this distribution here. The final key step is to simplify this,

$$\begin{aligned} p(\vec{Y}_{ij}|X_{ij},W,V) &= \mathbb{1}\Big(X_{ij} = \sum_{k=1}^{K} Y_{ij}^{(k)}\Big) \frac{X_{ij}!}{\prod_{k=1}^{K} Y_{ij}^{(k)}!} \Bigg(\underbrace{\frac{W_{ik}V_{kj}}{\sum_{l=1}^{K} W_{il}V_{lj}}}_{\equiv\, \phi_{ij}(k)}\Bigg)^{Y_{ij}^{(k)}} \\ &= \text{Multinomial}(X_{ij}, \phi_{ij}) \end{aligned} \tag{17}$$

- The conditional posterior on the K Poisson random variables in the vector $\vec{Y}_{ij}$ *given that their sum must equal* X_{ij} is a multinomial distribution with probability distribution equal to the normalization of the K parameters in the Poisson prior on $\vec{Y}_{ij}$. While this wouldn't be obvious *a priori*, it's an intuitively reasonable result and a nice one too. Notice that the indicator in front is superfluous since by definition of the multinomial distribution with these parameters this sum must be true. We can therefore ignore it, but notice that it had to be there to start the calculation.

- E-Step 2: Next we calculate the expectation using this $q(Y)$ distribution.

$$\begin{aligned}\mathcal{L} &= \sum_{Y} q(Y) \ln p(X, Y|W, V) \\ &= \sum_{i=1}^{M}\sum_{j=1}^{N}\left\{\mathbb{E}_q[\ln p(X_{ij}|\vec{Y}_{ij})] + \sum_{k=1}^{K}\mathbb{E}_q[\ln p(Y_{ij}^{(k)}|W_{ik}, V_{kj})]\right\} \\ &= \sum_{i=1}^{M}\sum_{j=1}^{N}\sum_{k=1}^{K}\mathbb{E}_q[Y_{ij}^{(k)}]\ln(W_{ik}V_{kj}) - W_{ik}V_{kj} + \text{constant}\end{aligned} \tag{18}$$

- In these equalities, we were able to get rid of $\mathbb{E}_q[\ln p(X_{ij}|\vec{Y}_{ij})] = \mathbb{E}_q[\ln \mathbb{1}(X_{ij} = \sum_{k=1}^{K} Y_{ij}^{(k)})]$ because $q(\vec{Y}_{ij})$ is such that $X_{ij} = \sum_{k=1}^{K} Y_{ij}^{(k)}$ with probability equal to one. Therefore the expectation is entirely over $\ln 1 = 0$. (Thankfully! If any of the $q(\vec{Y}_{ij})$ had nonzero probability of $X_{ij} \neq \sum_{k=1}^{K} Y_{ij}^{(k)}$ then $\mathcal{L} = -\infty$ and we couldn't proceed).

- Notice that given $q(\vec{Y}_{ij}) = \text{Multinomial}(X_{ij}, \phi_{ij})$, we can set $\mathbb{E}_q[Y_{ij}^{(k)}] = X_{ij}\phi_{ij}(k)$.

- M-Step: Finally we take derivatives with respect to W_{ik} and V_{kj} and set to zero.

$$\begin{aligned}\nabla_{W_{ik}}\mathcal{L}(W, V) &= 0 = \sum_{j=1}^{N}\frac{X_{ij}\phi_{ij}(k)}{W_{ik}} - \sum_{j=1}^{N} V_{kj} \\ &\Downarrow \\ W_{ik} &= \frac{\sum_{j=1}^{N} X_{ij}\phi_{ij}(k)}{\sum_{j=1}^{N} V_{kj}}\end{aligned} \tag{19}$$

$$\begin{aligned}\nabla_{V_{kj}}\mathcal{L}(W, V) &= 0 = \sum_{i=1}^{M}\frac{X_{ij}\phi_{ij}(k)}{V_{kj}} - \sum_{i=1}^{M} W_{ik} \\ &\Downarrow \\ V_{kj} &= \frac{\sum_{i=1}^{M} X_{ij}\phi_{ij}(k)}{\sum_{i=1}^{M} W_{ik}}\end{aligned} \tag{20}$$

- Now recalling that $\phi_{ij}(k) = W_{ik}V_{kj}/(WV)_{ij}$, we see that the updates are identical to the multiplicative updates for the NMF algorithm using the divergence penalty. Notice that the values of W and V in ϕ are fixed at the values of the previous iteration for all updates in this step. Also notice that these updates do not maximize $\mathcal{L}$ if done only once. However, they do increase $\mathcal{L}$, which is all that is required. We could continue to iterate between updating W and V for a fixed ϕ, or just make one update to each and then update ϕ.

- The EM algorithm also contains a little more information than the multiplicative algorithm for NMF discussed at the beginning of this lecture. (But it's not obvious at first glance if this extra information is useful at all. If it is it would be for computational efficiency.) As written in the original NMF algorithm, it appears that *all* values of W need to be the most recent ones when updating V and vice-versa for updating W. Just looking at Equations (3) and (4) above isn't

enough to say otherwise. However, we know from EM that the functions of W and V in the numerator correspond to ϕ and so we can keep those at the old values and only update the denominators. Therefore, one could iterate the original NMF algorithm several times only updating the sums in the denominators and holding all other values fixed, and separately update W and V in the numerator and multiplied out front every few iterations.

Variational inference for Poisson matrix factorization

- We have motivated a Bayesian approach to this problem. Next, we again define the model, this time along with its priors, and then discuss a variational inference algorithm for approximating the posterior. This algorithm will introduce new challenges that we haven't faced before, and we'll work through a possible solution that can be made more general.

- Model: We have the matrix X and model it with a K-rank non-negative factorization WV such that

$$X_{ij} \sim \text{Poisson}((WV)_{ij}) \tag{21}$$

- Priors: We use gamma priors for W and V as follows:

$$W_{ik} \sim \text{Gamma}(a, b), \qquad V_{kj} \sim \text{Gamma}(c, d) \tag{22}$$

- Posterior: We approximate the intractable posterior $p(W, V|X)$ with $q(W, V)$ using variational inference. We use the factorization

$$q(W, V) \;=\; \left[\prod_{i=1}^{M}\prod_{k=1}^{K} q(W_{ik})\right]\left[\prod_{j=1}^{N}\prod_{k=1}^{K} q(V_{kj})\right] \tag{23}$$

- Problem: We run into another problem this time in that things are *still* intractable. The variational objective function is

$$\begin{aligned}\mathcal{L} &= \mathbb{E}_q[\ln p(X|W,V)] + \mathbb{E}_q[\ln p(W)p(V)] - \mathbb{E}_q[\ln q] \qquad (24)\\ &= \sum_{i,j} X_{ij} \underbrace{\mathbb{E}_q\Big[\ln \sum_{k=1}^{K} W_{ik}V_{kj}\Big]}_{=\,???} - \sum_{k=1}^{K} \mathbb{E}_q[W_{ik}V_{kj}] + \mathbb{E}_q[\ln p(W)p(V)] - \mathbb{E}_q[\ln q] + \text{const.}\end{aligned}$$

- For this model we can't even calculate $\mathcal{L}$, so how do we take derivatives to update variational parameters for each q? The "optimal method" for learning each q has the same problem, since the problematic expectation still needs to be taken.

- Therefore, we don't have an analytic objective function to work with. This is a very common problem and has a few solutions. One trick is to replace the problem function with another function that approximates it (by lower bounding it) and is tractable.

- Since we want to maximize $\mathcal{L}$, we therefore lower bound the problem function.

- Let's work out this problem for this case using more abstract notation.

- Problem function: $\mathbb{E}_q \ln \sum_k Z_k$, $q = \prod_k q(Z_k)$

- The function $\ln$ is concave. Recall that for a concave function of a random variable, $f(Z)$,

$$f(\mathbb{E}Z) \geq \mathbb{E}f(Z) \tag{25}$$

- Notice that the function $\mathbb{E}_q \ln \sum_k Z_k \leq \ln \sum_k \mathbb{E}_q Z_k$. Therefore, we can't lower bound the problem function this way—instead it's an upper bound. The expectation in $\mathbb{E}_q \ln \sum_k Z_k$ is *not* the expectation corresponding to the function f above.

- Instead, we introduce a brand new discrete K-dimensional probability vector $\phi = [\phi(1), \ldots, \phi(K)]$ and write the trivial equality

$$\ln \sum_k Z_k = \ln \sum_k \phi(k) \frac{Z_k}{\phi(k)} \tag{26}$$

- This is a "trick" in that ϕ doesn't necessarily have a modeling purpose. We're just using math to our advantage. (However, we could also motivate this in a way similar to the previous EM algorithm, where we introduced auxiliary variables.)

- We have written $\ln \sum_k Z_k = \ln \mathbb{E}[Z/\phi]$, where this time the expectation is with respect to the distribution ϕ. Since $\ln$ is concave

$$\ln \sum_k Z_k = \ln \mathbb{E}[Z/\phi] \geq \mathbb{E}[\ln Z/\phi] = \sum_{k=1}^{K} \phi(k) \ln \frac{Z_k}{\phi(k)} \tag{27}$$

- Therefore, in the variational objective function

$$\mathbb{E}_q \ln \sum_{k=1}^{K} Z_k \geq \sum_{k=1}^{K} \phi(k) \mathbb{E}_q \ln Z_k - \sum_{k=1}^{K} \phi(k) \ln \phi(k) \tag{28}$$

- Notice that the problem should be fixed now since these are expectations we usually can take. Picking the lower bound as we did is part of the "art" of this technique. There are other lower bounds we could use. Some are no good because they don't result in analytic expectations. Some might be better than this one in that they are tighter—they approximate the original function more closely. This specific lower bound is probably not the only option.

- The next question is how do we set $\phi = [\phi(1), \ldots, \phi(K)]$? We want to set ϕ so that the bound is as good of an approximation as possible.

- Therefore we want to maximize this lower bound over ϕ. Using Lagrange multipliers, we can find that the lower bound is maximized when

$$\phi(k) = \frac{\mathrm{e}^{\mathbb{E}_q \ln Z_k}}{\sum_{\ell=1}^{K} \mathrm{e}^{\mathbb{E}_q \ln Y_\ell}} \tag{29}$$

- At this point, we have two paths we can go down:
  1. Plug this $\phi(k)$ back into the lower-bounded objective function
  2. Keep $\phi(k)$ as an auxiliary variable and use this update for it

- Path 1: Plugging this value of $\phi(k)$ back in and simplifying, we find

$$\mathbb{E}_q \ln \sum_{k=1}^{K} Z_k \;\geq\; \ln \sum_{k=1}^{K} \mathrm{e}^{\mathbb{E}_q \ln Z_k} \tag{30}$$

- This is the tightest possible lower bound of $\mathbb{E}_q \ln \sum_{k=1}^{K} Z_k$ when we limit ourselves to selecting from the family $\sum_{k=1}^{K} \phi(k) \mathbb{E}_q \ln Z_k - \sum_{k=1}^{K} \phi(k) \ln \phi(k)$

- If we go down this path, then the original problem is modified to

$$\mathcal{L} \geq \sum_{i,j} X_{ij} \ln \sum_{k=1}^{K} \mathrm{e}^{\mathbb{E}_q \ln W_{ik} + \mathbb{E}_q \ln V_{kj}} - \sum_{k=1}^{K} \mathbb{E}_q[W_{ik}]\mathbb{E}_q[V_{kj}] + \mathbb{E}_q[\ln p(W)p(V)] - \mathbb{E}_q[\ln q] \tag{31}$$

- The first term is the only place where an approximation is being made. This path has some pros and cons.

- PROS: We have a closed form objective that is the tightest possible approximation given the lower bound we use. (We'll see that Path 2 actually has this same PRO.)

- CONS: We can't use the optimal method to find q. We need to take derivatives and use gradient methods. That is ok, but like the EM story, if we could avoid doing this it would be preferable.

- Path 2: The second option is to keep ϕ as an auxiliary variable that we also optimize over. Since there is a function $\mathbb{E}_q[\ln \sum_k W_{ik} V_{kj}]$ for each (i, j) pair, we introduce a vector ϕ_{ij} for each of these to lower bound it. This is because the best lower bound will be different for each (i, j). Lower bounding them individually rather than using one shared ϕ will result in a much better approximation (it likely wouldn't work well with a single ϕ since the overall approximation will be bad).

- Therefore, we lower bound the variational objective function as

$$\begin{aligned} \mathcal{L} \;\geq\; & \sum_{i,j} \sum_{k=1}^{K} X_{ij} \left[\phi_{ij}(k) \mathbb{E}_q[\ln W_{ik} + \ln V_{kj}] - \phi_{ij}(k) \ln \phi_{ij}(k)\right] \\ & - \sum_{k=1}^{K} \mathbb{E}_q[W_{ik}]\mathbb{E}_q[V_{kj}] + \mathbb{E}_q[\ln p(W)p(V)] - \mathbb{E}_q[\ln q] \end{aligned} \tag{32}$$

- The advantage of this function is that we can now use the optimal method for finding each q. Also, notice that at the point of convergence, none of the parameters change anymore. Therefore, $\phi_{ij}(k)$ will equal its optimal value. Therefore, Path 2 finds a local optimal of the same function

that Path 1 does. It just does it by taking a few more steps. We had a very similar situation with EM and it's worth independently thinking more about the similarities.

- Let $\ln \hat{p}(X, W, V)$ be the log-joint distribution using the lower bound instead of the true objective,

$$\ln \hat{p}(X, W, V) = \sum_{i,j} \sum_{k=1}^{K} X_{ij} \left[\phi_{ij}(k)(\ln W_{ik} + \ln V_{kj}) - \phi_{ij}(k) \ln \phi_{ij}(k) \right] - \sum_{k=1}^{K} W_{ik} V_{kj} + \mathbb{E}_q[\ln p(W)p(V)]$$

- Using the optimal method for finding q with this lower bound, we have the following algorithm.

- Finding $q(W_{ik})$: By the typical route, we have

$$\begin{aligned} q(W_{ik}) &\propto \mathrm{e}^{\mathbb{E}_{-q}[\ln \hat{p}(X,W,V)]} \\ &\propto W_{ik}^{a+\sum_{j=1}^{N} X_{ij}\phi_{ij}(k)-1} \mathrm{e}^{-(b+\sum_{j=1}^{N} \mathbb{E}_q[V_{kj}])W_{ik}} \end{aligned} \tag{33}$$

  Therefore,

$$q(W_{ik}) = \text{Gamma}\left(a + \sum_{j=1}^{N} X_{ij}\phi_{ij}(k)\,,\; b + \sum_{j=1}^{N} \mathbb{E}_q[V_{kj}] \right) \tag{34}$$

- Finding $q(V_{kj})$: By symmetry we can quickly find that

$$q(V_{kj}) = \text{Gamma}\left(c + \sum_{i=1}^{M} X_{ij}\phi_{ij}(k)\,,\; b + \sum_{i=1}^{M} \mathbb{E}_q[W_{ik}] \right) \tag{35}$$

- Optimizing $\phi_{ij}(k)$: After updating each $q(W_{ik})$ and $q(V_{kj})$, set

$$\phi_{ij}(k) \;=\; \frac{\mathrm{e}^{\mathbb{E}_q[\ln W_{ik}] + \mathbb{E}_q[\ln V_{kj}]}}{\sum_{\ell=1}^{K} \mathrm{e}^{\mathbb{E}_q[\ln W_{i\ell}] + \mathbb{E}_q[\ln V_{\ell j}]}} \tag{36}$$

- To assess convergence, we then evaluate the lower bound of the variational objective function, since this is what we are trying to maximize. We hope that the q distributions then are "about as good as" what we would have gotten optimizing $\mathcal{L}$ directly, since the peaks and valleys of the objective function $\mathcal{L}$ should overlap significantly with those of the lower bound we use.

Comparison with EM

- With EM, updating $\phi_{ij}(k)$ didn't involve expectations since we have a point estimate of W and V. Notice that if we remove these expectations above, the update to $\phi_{ij}(k)$ is identical to EM.

- Using the distributions $q(W_{ik})$ and $q(V_{kj})$, compare $\mathbb{E}_q[W_{ik}]$ and $\mathbb{E}_q[V_{kj}]$ with the updates of EM. We can see a close similarity between the two, only now we have a full probability distribution on these terms. Therefore this model can be considered as the Bayesian approach to NMF with a divergence penalty, rather than just being motivated by it.

- This indicates that our lower bound is doing something similar (or perhaps equivalent) to introducing latent random variables $Y_{ij}^{(k)}$ to the model.