Topics in Probabilistic Modeling and Inference
(CS698X, Spring 2019)
Homework 3
Due Date: April 02, 2019 (11:59pm)

Instructions:

- Only electronic submissions will be accepted. Your main PDF writeup must be typeset in LaTeX (please also refer to the "Additional Instructions" below).
- Your submission will have two parts: The main PDF writeup (to be submitted via Gradescope `https://www.gradescope.com/`) and the code for the programming part (to be submitted via this Dropbox link: `https://tinyurl.com/cs698x-sp19-hw3`). Both parts must be submitted by the deadline to receive full credit (**delay in submitting either part would incur late penalty for both parts**). We will be accepting late submissions upto 72 hours after the deadline (with every 24 hours delay incurring a 10% late penalty). We won't be able to accept submissions after that.

Additional Instructions

- We have provided a LaTeX template file `hw3sol.tex` to help typeset your PDF writeup. There is also a style file `pmi.sty` that contain shortcuts to many of the useful LaTeX commends for doing things such as boldfaced/calligraphic fonts for letters, various mathematical/greek symbols, etc., and others. Use of these shortcuts is recommended (but not necessary).
- Your answer to every question should begin on a new page. The provided template is designed to do this automatically. However, if it fails to do so, use the `\clearpage` option in LaTeX before starting the answer to a new question, to *enforce* this.
- While submitting your assignment on the Gradescope website, you will have to specify on which page(s) is question 1 answered, on which page(s) is question 2 answered etc. To do this properly, first ensure that the answer to each question starts on a different page.
- Be careful to flush all your floats (figures, tables) corresponding to question n before starting the answer to question $n+1$ otherwise, while grading, we might miss your important parts of your answers.
- Your solutions must appear in proper order in the PDF file i.e. solution to question n must be complete in the PDF file (including all plots, tables, proofs etc) before you present a solution to question $n+1$.
- For the programming part, all the code and README should be zipped together and submitted as a single file named `yourrollnumber.zip`. Please DO NOT submit the data provided.

Problem 1 (10 marks)

Consider approximating an expectation $\mathbb{E}[f] = \int f(\boldsymbol{z})p(\boldsymbol{z})d\boldsymbol{z}$ using S samples $\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(L)}$ drawn i.i.d. from $p(\boldsymbol{z})$. Denote the approximated expectation as $\hat{f} = \frac{1}{S}\sum_{s=1}^{S} f(\boldsymbol{z}^{(\ell)})$. Show that this approximation is unbiased, i.e., $\mathbb{E}[\hat{f}] = \mathbb{E}[f]$. Also show that the variance of this approximation is given by var$[\hat{f}] = \frac{1}{S}\mathbb{E}[(f - \mathbb{E}[f])^2]$, i.e., the well-known result that the Monte-Carlo estimate's variance goes down as S increases.

Problem 2 (20 marks)

Consider linear regression with likelihood defined by Student t distribution $p(y_n|\boldsymbol{x}_n, \boldsymbol{w}, \sigma^2, \nu) = \mathcal{T}(y_n|\boldsymbol{w}^\top\boldsymbol{x}_n, \sigma^2, \nu)$ and a Gaussian prior on the weights $\boldsymbol{w}$, i.e., $p(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{w}|0, \rho^2\mathbf{I}_D)$. A Student t likelihood is often better than a Gaussian likelihood since it models outliers better (since it is a heavy-tailed distribution). Assume we are given N training examples, $(\mathbf{X}, \boldsymbol{y}) = \{(\boldsymbol{x}_n, y_n)\}_{n=1}^N$ to infer $\boldsymbol{w}$.

Unfortunately, the Student t likelihood is not conjugate to the Gaussian prior! However, thankfully, the Student t distribution can be expressed in the following "infinite mixture" form

$$\mathcal{T}(y|\mu, \sigma^2, \nu) = \int \mathcal{N}(y|\mu, \sigma^2/z)\text{Gamma}(z|\frac{\nu}{2}, \frac{\nu}{2})dz$$

The above is called a "Gaussian scale mixture) (note that variance is also called the scale). Essentially, we obtain Student t by taking infinite many Gaussians, each with a different variance σ^2/z, where z is another latent variable that we have introduced, and then integrating out z.

Use this idea to develop a sampling based inference procedure to infer $\boldsymbol{w}$. Although this would have been otherwise hard due to lack of conjugacy in this case, if we explicitly also keep the variables $z_1, \ldots, z_N$ in the model, this will give us an "augmented" model that has conjugacy with a simple inference procedure!

Essentially, in this augmented model, we can consider the joint distribution of the output y_n and the augmented variable z_n, e.g., instead of $\mathcal{T}(y|\mu, \sigma^2, \nu)$, we will consider $p(y, z|\mu, \sigma^2, \nu) = \mathcal{N}(y|\mu, \sigma^2/z)\text{Gamma}(z|\frac{\nu}{2}, \frac{\nu}{2})$.

In our linear regression problem, since the z_n's that we will introduce for each $\mathcal{T}(y_n|\boldsymbol{w}^\top\boldsymbol{x}_n, \sigma^2, \nu)$ aren't known, these need to be inferred as well, along with our main variable of interest $\boldsymbol{w}$. To do so, construct a Gibbs sampler for $p(\boldsymbol{w}, z_1, \ldots, z_N|\mathbf{X}, \boldsymbol{y})$. Derive the conditional posteriors of all the unknowns and clearly write down their expressions of their parameters. Assume all other unknowns (σ^2, ν, ρ^2) to be known.

Avoid very detailed steps in the derivations. If some updates are easy to obtain using standard formulae (e.g., Gaussian posterior updates), please feel free to use those.

Problem 3 (30 marks)

Consider the Latent Dirichlet Allocation (LDA) model

$$\begin{aligned}
\phi_k &\sim \text{Dirichlet}(\eta, \ldots, \eta), \quad k = 1, \ldots, K \\
\theta_d &\sim \text{Dirichlet}(\alpha, \ldots, \alpha), \quad d = 1, \ldots, D \\
\boldsymbol{z}_{d,n} &\sim \text{multinoulli}(\theta_d), \quad n = 1, \ldots, N_d \\
\boldsymbol{w}_{d,n} &\sim \text{multinoulli}(\phi_{\boldsymbol{z}_{d,n}})
\end{aligned}$$

In the above, ϕ_k denotes the V dim. topic vector for topic k (assuming vocabulary of V unique words), θ_d denotes the K dim. topic proportion vector for document d, and the number of words in document d is N_d.

Your task is to derive a Gibbs sampler for the word-topic assignment variable $z_{d,n}$ (for each word in each document). Your sampler should not sample β_k, θ_d but only be sampling the $z_{d,n}$'s from the conditional posterior (CP). Derive and clearly write down the the expressions for the CP that the Gibbs sampler requires in this case, and sketch the overall Gibbs sampler. Important: Note of the expressions should contain θ_d and ϕ_k. Also briefly justify why your expression for CP makes intuitive sense.

Suppose, in addition, we are also interested in computing the posterior expectation $\mathbb{E}[\theta_d]$ for each document and the posterior expectation $\mathbb{E}[\phi_k]$ for each topic, using the information in the collected samples of **Z**. Suggest a way and give the proper expressions (approximation is fine) that compute these quantities, and give an intuitive meaning of the final expressions for $\mathbb{E}[\theta_d]$ and $\mathbb{E}[\phi_k]$.

Problem 4 (20 marks)

Consider an $N \times M$ matrix $\mathbf{X}$ with each entry X_{nm} a count value, modeled as

$$\begin{aligned} p(X_{nm}|\boldsymbol{u}_n, \boldsymbol{v}_m) &= \text{Poisson}(X_{nm}|\boldsymbol{u}_n^\top \boldsymbol{v}_m) \\ p(u_{nk}|a,b) &= \text{Gamma}(u_{nk}|a,b) \\ p(v_{mk}|c,d) &= \text{Gamma}(v_{mk}|c,d) \end{aligned}$$

In the above, $\boldsymbol{u}_n \in \mathbb{R}_+^K$, $\boldsymbol{v}_m \in \mathbb{R}_+^K$, and the Gamma distribution is assumed to have the shape and rate parameterization. The above is essentially a gamma-Poisson matrix factorization model for count data.

Derive a Gibbs sampler for the above model. In particular, you need to derive the conditional posteriors for u_{nk} and v_{mk}. Assume the hyperparameters a, b, c, d to be known.

A useful result that you will need: Given K independent Poisson r.v.'s $x_1, \ldots, x_K$ s.t. $x_k \sim \text{Poisson}(\lambda_k)$, their sum $x = \sum_{k=1}^K x_k$ is also Poisson distributed, i.e., $x \sim \text{Poisson}(\lambda)$ where $\lambda = \sum_{k=1}^K \lambda_k$. The converse is also true. Based on this, a count-valued r.v. x can be thought of as a sum of smaller count-valued r.v.'s $x_1, \ldots, x_K$.

Problem 5 (20+20 = 40 marks)

(Part 1: Implementing A Rejection Sampler) Consider a distribution $p(x) = \frac{\tilde{p}(x)}{Z}$ where $\tilde{p}(x) = \mathcal{N}(x|20, 10) + \mathcal{N}(x|50, 5) + \mathcal{N}(x|80, 20)$. Implement a rejection sampler that generates 10,000 samples from $p(x)$ using a Gaussian proposal $q(x) = \mathcal{N}(x|50, 30)$. To do rejection sampling, you also need to find an M s.t. $Mq(x) \geq \tilde{p}(x)$, $\forall x$. Note that a choice of $M = \max_x \frac{\tilde{p}(x)}{q(x)}$ will guarantee this (note that a larger M can also be used but will lead to larger rejection rate). You can find M by computing this maxima over a sufficiently large number of x values from a pre-specified range (e.g., [-50,100]). What value of M do you get?

Given this M, run your rejection sampler, and on the same figure, show (in different colors), the original unnormalized distribution $\tilde{p}(x)$, your proposal $q(x)$, and a histogram plot of the accepted samples. What's the acceptance rate? Does it make sense based on the value of M you found? Submit your code as well as the figure.

(Part 2: Implementing MH Sampling for 2-D Gaussian) In this problem, your task is to implement MH sampling to generate random samples from a 2-D Gaussian $p(\boldsymbol{z}) = \mathcal{N}\left(\begin{bmatrix}4\\4\end{bmatrix}, \begin{bmatrix}1 & 0.8\\0.8 & 1\end{bmatrix}\right)$.

To sample from $p(\boldsymbol{z})$, you will use a proposal distribution $q(\boldsymbol{z}^{(t)}|\boldsymbol{z}^{(t-1)}) = \mathcal{N}\left(\boldsymbol{z}^{(t-1)}, \begin{bmatrix}\sigma^2 & 0\\0 & \sigma^2\end{bmatrix}\right)$ and play with different values of the proposal distribution's variance σ^2. For generating the candidate sample from the proposal distribution, you can use existing functions from MATLAB/Python/R (whichever language you are using for the implementation), but you must not use the actual distribution $p(\boldsymbol{z})$ to generate the samples.

You will experiment with the following values of $\sigma^2 : 0.01, 1, 100$. For each of these cases, run the MH sampler long enough to collect 10,000 samples and show the plots of the generated samples on a 2-D plane for 100 samples, 1000 samples, and 10,000 samples (similar to the plots of slide 14, lecture-15).

Looking at the plots, which of the 3 proposals ($q(\boldsymbol{z}^{(t)}|\boldsymbol{z}^{(t-1)})$ with $\sigma^2 : 0.01, 1, 100$) seems the best choice to you? What is the rejection rate in each of these cases (rejection rate is the ratio of number of samples rejected and the total number of candidate samples generated)? Submit the code as well as the plots.