Mid-sem exam solutions review, Variational Inference (wrap-up)

CS772A: Probabilistic Machine Learning

Piyush Rai

Mid-sem exam: Q1

1. Consider the Bernoulli observation model with likelihood $p(y_n|\theta) = \text{Bernoulli}(y_n|\theta) = \theta^{y_n}(1-\theta)^{1-y_n}$ and prior $p(\theta|a,b) = \text{Beta}(\theta|a,b) = \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)}\theta^{a-1}(1-\theta)^{b-1}$, where a, b are the hyperparameters of the prior on $\theta \in (0,1)$. Given training data $\boldsymbol{y} = \{y_1, y_2, \ldots, y_N\}$, derive the expression of the objective function which is optimized to compute the point estimates of a, b. You don't need to solve the optimization problem.

To get the point estimates of a, b, we need to maximize the marginal likelihood given by $p(\boldsymbol{y}|a,b) = \int p(\boldsymbol{y}|\theta)p(\theta|a,b)d\theta$. For this model, $p(\boldsymbol{y}|\theta) = \theta^{N_1}(1-\theta)^{N_0}$ where N_1 and N_1 denote the number of observations with value 1 and 0, respectively, and $p(\theta|a,b) = \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)}\theta^{a-1}(1-\theta)^{b-1}$.

Thus $p(\boldsymbol{y}|a,b) = \int \theta^{N_1}(1-\theta)^{N_0} \times \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)}\theta^{a-1}(1-\theta)^{b-1}d\theta = \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)}\int \theta^{N_1+a-1}(1-\theta)^{N_0+b-1}d\theta$. The integral is the normalization constant of $\text{Beta}(N_1+a, N_0+b)$ which is equal to $\frac{\Gamma(N_1+a)\Gamma(N_0+b)}{\Gamma(N+a+b)}$.

Thus, $p(\boldsymbol{y}|a,b) = \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)}\frac{\Gamma(N_1+a)\Gamma(N_0+b)}{\Gamma(N+a+b)}$

Also equal to the ratio of the normalization constants of the Beta posterior and Beta prior

Mid-sem exam: Q2

2. Recall that Laplace's approximation $p(\theta|\mathcal{D}) \approx \mathcal{N}(\theta|\theta_{MAP}, \Lambda^{-1})$ is based on a second order Taylor expansion of log of joint distribution of data and unknowns $\log p(\mathcal{D},\theta) \approx \log p(\mathcal{D},\theta_{MAP}) - \frac{1}{2}(\theta-\theta_{MAP})^\top \Lambda(\theta-\theta_{MAP})$ where $\Lambda = -\nabla^2_{\theta=\theta_{MAP}} \log p(\mathcal{D},\theta)$. Derive the expression for the marginal likelihood $p(\mathcal{D})$ when using Laplace's approximation. Note: For an r.v. $\boldsymbol{x} \in \mathbb{R}^D$, $\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu},\boldsymbol{\Sigma}) = \frac{1}{\sqrt{(2\pi)^D|\boldsymbol{\Sigma}|}}\exp\{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^\top\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\}$
(Apologies for the confusion regarding the sign. I was trying to write the quadratic term compactly and replaced the second derivative by Λ, not paying attention to the signs, which led to the confusion. For this question, all students who showed the basic steps of the solution given below will get full credit.)

$$p(\mathcal{D}) = \int p(\mathcal{D},\theta)d\theta = \int \exp(\log p(\mathcal{D},\theta))d\theta \approx \int \exp\left(\log p(\mathcal{D},\theta_{MAP}) - \frac{1}{2}(\theta-\theta_{MAP})^\top\Lambda(\theta-\theta_{MAP}\right)d\theta$$

Note that $\int \exp\left(-\frac{1}{2}(\theta-\theta_{MAP})^\top\Lambda(\theta-\theta_{MAP}\right)$ is just the normalization constant of our Laplace's approximation $p(\theta|\mathcal{D}) \approx \mathcal{N}(\theta|\theta_{MAP},\Lambda^{-1})$, and this constant is equal to $(2\pi)^{D/2}\det(\Lambda^{-1})^{1/2}$ Therefore, $p(\mathcal{D}) \approx \exp(\log p(\mathcal{D},\theta_{MAP}))(2\pi)^{D/2}\det(\Lambda^{-1})^{1/2}$.

Mid-sem exam: Q3

3. Consider a toy latent variable model with a latent $z_n \in \mathbb{R}$ for each observation $x_n \in \mathbb{R}$. Assume likelihood $p(x_n|z_n,\theta) = \frac{\exp(\theta x_n z_n)}{C_1}$, prior $p(z_n|\phi) = \frac{\exp(\phi z_n^2)}{C_2}$, where C_1, C_2 are constants of proportionality. Denote $\Theta = (\theta, \phi)$ and suppose the CP $p(z_n|\Theta, x_n) = \mathcal{N}(z_n|\mu_{z_n}, \sigma^2_{z_n})$. Can expected complete data log-likelihood (CLL) be computed in exact form? If yes, derive its expression. If no, then state why it can't be.

The CLL (writing it for a single observation x_n; the overall CLL will simply be a sum over n) will be $\log p(x_n, z_n|\Theta) = \log[p(x_n|z_n,\theta)p(z_n|\phi)] = \log p(x_n|z_n,\theta) + \log p(z_n|\phi)$. Substituting the expressions for the likelihood and prior, we get $\log p(x_n, z_n|\Theta) = \theta x_n z_n + \phi z_n^2 - \log C_1 - \log C_2$.

Therefore, the expected CLL will be $\mathbb{E}[\log p(x_n, z_n|\Theta)] = \theta x_n \mathbb{E}[z_n] + \phi \mathbb{E}[z_n^2] - \log C_1 - \log C_2$. The only expectations we need here are $\mathbb{E}[z_n] = \mu_{z_n}$ and $\mathbb{E}[z_n^2] = \text{var}(z_n) + \mathbb{E}[z_n]^2 = \sigma^2_{z_n} + \mu^2_{z_n}$.

Mid-sem exam: Q4

4. Consider Bayesian linear regression with likelihood $p(y_n|\boldsymbol{x}_n, \boldsymbol{w}, \beta) = \mathcal{N}(y_n|\boldsymbol{w}^\top \boldsymbol{x}_n, \beta^{-1})$ and prior $p(\boldsymbol{w}|\lambda) = \mathcal{N}(\boldsymbol{w}|\mathbf{0}, \lambda^{-1}\mathbf{I}_D)$, N training examples $\mathcal{D} = \{\boldsymbol{x}_n, y_n\}_{n=1}^N$, and the following two ways of computing the posterior distribution of $\boldsymbol{w}$, assuming the hyperparameters λ, β as known: (1) You compute the posterior $p(\boldsymbol{w}|\mathcal{D}, \lambda, \beta)$ using all the N training examples at once, and (2) You compute this posterior in multiple rounds of updates where, in each round, you are given a single training example $(\boldsymbol{x}_n, y_n)$ which you use to update the posterior in an online manner, taking the previous round's posterior as the prior. Will both approaches (1) and (2) give the same final posterior $p(\boldsymbol{w}|\mathcal{D}, \lambda, \beta)$? Justify your answer formally using necessary equations. Also answer the same question if the likelihood model is of the form $p(y_n|\boldsymbol{x}_n, \boldsymbol{w}, \beta) = \mathcal{N}(y_n|\text{NN}(\boldsymbol{x}_n; \mathbf{W}), \beta^{-1})$ where $\text{NN}(\boldsymbol{x}_n; \mathbf{W})$ denotes a deep neural network with weights $\mathbf{W}$ and we have a $\mathcal{N}(0, \lambda^{-1})$ on each of the weights of the neural network. Note: If you need but don't remember Gaussian posterior expression, you may use linear Gaussian model reverse conditional result given in Question 8.

The posterior of $\boldsymbol{w}$ for Bayesian linear regression is $p(\boldsymbol{w}|\mathcal{D}, \lambda, \beta) = \mathcal{N}(\boldsymbol{w}|\mu, \Lambda^{-1})$ where $\Lambda = \mathbf{X}^\top\mathbf{X} + \frac{\lambda}{\beta}\mathbf{I}_D = \sum_{n=1}^N \boldsymbol{x}_n\boldsymbol{x}_n^\top + \frac{\lambda}{\beta}\mathbf{I}_D$ and $\mu = (\mathbf{X}^\top\mathbf{X} + \frac{\lambda}{\beta}\mathbf{I}_D)^{-1}\mathbf{X}^\top\boldsymbol{y} = \Lambda^{-1}\sum_{n=1}^N y_n\boldsymbol{x}_n$. Note that both Λ and μ_N have forms that depend on sums of quantities ($\boldsymbol{x}_n\boldsymbol{x}_n^\top$ for Λ and $y_n\boldsymbol{x}_n$ for μ) and these sums can be computed online. Thus the "batch" as well as "online" updates to the posterior will yield the same answer.

For a (Bayesian) deep neural net ,however, we won't have such a nice and exact form of the posterior (we may have a Gaussian posterior if using Laplace's approximation but it still won't have a form where the mean and precision matrix will be defined using simple summations over the observations). Thus the batch and online updates won't give the same answer in case of Bayesian deep neural net.

Mid-sem exam: Q5

5. The definition of KL divergence between two distributions $p_{\theta_1}(x)$ and $p_{\theta_2}(x)$ is given by $\mathrm{KL}[p_{\theta_1}||p_{\theta_2}] = \int p_{\theta_1}(x) \log \frac{p_{\theta_1}(x)}{p_{\theta_2}(x)} dx = \mathbb{E}_{p_{\theta_1}(x)}[\log p_{\theta_1}(x) - \log p_{\theta_2}(x)]$, where their parameters are θ_1 and θ_2, respectively.

   Now consider a distribution $p_\theta(x)$ with parameters $\theta \in \mathbb{R}^D$, and suppose we introduce a small additive perturbation δ to θ as $\theta' = \theta + \delta$ which changes the distribution to $p_{\theta'}(x)$. Show that the KL divergence $\mathrm{KL}[p_\theta||p_{\theta'}]$ is equal to the squared Mahalanobis distance between θ and θ', i.e., $(\theta' - \theta)^\top \mathbf{W}(\theta' - \theta)$ where $\mathbf{W}$ is the Malalanobis distance metric. Solution hint: Use the idea of Taylor expansion.

   Let's do a second order Taylor expansion of $\log p_{\theta'}(x)$ around θ. Since $\theta' = \theta + \delta$, the Taylor expansion would be $\log p_{\theta'}(x) = \log p_\theta(x) + (\theta' - \theta)^\top \nabla \log p_\theta(x) + (\theta' - \theta)^\top \nabla^2 \log p_\theta(x)(\theta' - \theta)$. Thus

$$\begin{aligned} \mathrm{KL}[p_\theta||p_{\theta'}] &= \mathbb{E}_{p_\theta(x)}[\log p_\theta(x) - \log p_{\theta'}(x)] \\ &= \mathbb{E}_{p_\theta(x)}[-(\theta' - \theta)^\top \nabla \log p_\theta(x) - (\theta' - \theta)^\top \nabla^2 \log p_\theta(x)(\theta' - \theta)] \end{aligned}$$

   Using the fact that $\mathbb{E}_{p_\theta(x)}[\nabla \log p_\theta(x)] = 0$ (expectation of the score function is zero), we have $\mathrm{KL}[p_\theta||p_{\theta'}] = \delta^\top \mathbf{F}(\theta)\delta$ where $\mathbf{F}(\theta) = -\mathbb{E}_{p_\theta(x)}[\nabla^2 \log p_\theta(x)] = \mathbb{E}_{p_\theta(x)}[\nabla \log p_\theta(x) \nabla \log p_\theta(x)^\top]$, which is indeed the squared Mahalanobis distance between θ and θ' with $\mathbf{W} = \mathbf{F}(\theta)$.

   A side note (not required for the solution): This notion of distance between two distributions is central to the idea of "natural gradients" used in natural gradient descent (NGD) algorithms which is often used when learning the parameters of probability distributions (e.g., in MLE/MAP/EM,VI) where it is a more sensible way of updating the parameters instead of standard gradient descent. Basically, in NGD, we use $\mathbf{F}(\theta)$ to define the learning rate parameter of GD so that we don't use the same learning rate in all dimensions and our learning rate also takes into account the information about the shape of the distribution. We will study this later in this course.

Mid-sem exam: Q6

6. Suppose we have a clean image denoted by a vector $\boldsymbol{x}_0 \in \mathbb{R}^D$ of its pixel intensities. We gradually corrupt $\boldsymbol{x}_0$ in a series of steps $t = 1, 2, 3, \ldots$ using $p(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) = \mathcal{N}(\boldsymbol{x}_t|\sqrt{1-\beta_t}\boldsymbol{x}_{t-1}, \beta_t\mathbf{I})$ where $\beta_t \in (0,1)$. Note that this is equivalent to writing $\boldsymbol{x}_t = \sqrt{1-\beta_t}\boldsymbol{x}_{t-1} + \sqrt{\beta_t}\epsilon_t$ where indepedent noise $\sqrt{\beta_t}\epsilon_t$ with $p(\epsilon_t) = \mathcal{N}(\epsilon_t|\mathbf{0},\mathbf{I})$ is added to the scaled version $\sqrt{1-\beta_t}\boldsymbol{x}_{t-1}$ of the corrupted image at step $t-1$. Is it possible to get $\boldsymbol{x}_t$ from $\boldsymbol{x}_0$ in a single step of corruption using a distribution $p(\boldsymbol{x}_t|\boldsymbol{x}_0)$? If yes, write down what ths distribution will be along with its parameters. If not, explain clearly why it is not possible. Note: If you need, you may use the result that if $p(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ and $\boldsymbol{y} = \mathbf{A}\boldsymbol{x} + \boldsymbol{b}$ then $p(\boldsymbol{y}) = \mathcal{N}(\boldsymbol{y}|\mathbf{A}\boldsymbol{\mu} + \boldsymbol{b}, \mathbf{A}^\top\boldsymbol{\Sigma}\mathbf{A})$

Starting with $\boldsymbol{x}_t = \sqrt{1-\beta_t}\boldsymbol{x}_{t-1} + \sqrt{\beta_t}\epsilon_t$, substituting $\boldsymbol{x}_{t-1} = \sqrt{1-\beta_{t-1}}\boldsymbol{x}_{t-2} + \sqrt{\beta_{t-1}}\epsilon_{t-1}$ gives us

$$\begin{aligned}\boldsymbol{x}_t &= \sqrt{1-\beta_t}(\sqrt{1-\beta_{t-1}}\boldsymbol{x}_{t-2} + \sqrt{\beta_{t-1}}\epsilon_{t-1}) + \sqrt{\beta_t}\epsilon_t \\ &= \sqrt{1-\beta_t}\sqrt{1-\beta_{t-1}}\boldsymbol{x}_{t-2} + \sqrt{\beta_{t-1} - \beta_t\beta_{t-1}}\epsilon_{t-1} + \sqrt{\beta_t}\epsilon_t\end{aligned}$$

Note that the green terms denote sum of two independent Gaussian random variables with zero means and with covariances equal to $(\beta_{t-1} - \beta_t\beta_{t-1})\mathbf{I}_D$ and $\beta_t\mathbf{I}_D$. This sum will give another zero mean Gaussian random variable scaled by a constant. Repeating the same t times, we can see that $\boldsymbol{x}_t$ represented as a sum of $\prod_{i=0}^t \sqrt{1-\beta_{t-i}}\boldsymbol{x}_0$ and a zero mean Gaussian random variable scaled by a constant. This sum will also be a Gaussian with mean equal to $\prod_{i=0}^t \sqrt{1-\beta_{t-i}}\boldsymbol{x}_0$ and variance which will depend on all the β_t's, and thus $p(\boldsymbol{x}_t|\boldsymbol{x}_0)$ will be a Gaussian. So we can get the corrupted image at time step t using a single step starting with $\boldsymbol{x}_0$ instead of doing it for t steps. This idea is actually used in the forward (corruption) process of Diffusion Models which is one of the most popular deep generative models nowadays that we will study later in the course.

Mid-sem exam: Q7

7. For a linear model of the form $y = \boldsymbol{w}^\top \boldsymbol{x}$, we can regularize $\boldsymbol{w}$ using ℓ_1 regularization or a using Laplace prior which is of the form $p(\boldsymbol{w}) \propto \exp(-|\boldsymbol{w}|)$ to learn a sparse vector $\boldsymbol{w}$ for identifying the relevant features from the inputs. Doing the same for a Gaussian Process (GP), which is a nonlinear model, can be done using a kernel function. Suggest one such kernel function which can do the same thing, clearly write down its mathematical expression, and briefly explain how we can learn this kernel function.

We can use the ARD kernel. The ARD kernel is similar to the RBF kernel but has a different bandwidth parameter for each feature. In particular, the ARD kernel between two D dimensional inputs $\boldsymbol{x}_n$ and $\boldsymbol{x}_m$ can be written as

$$k(\boldsymbol{x}_n, \boldsymbol{x}_m) = \exp\left(-\sum_{d=1}^{D} \frac{(x_{nd} - x_{md})^2}{\gamma_d}\right)$$

If a feature (say the d^{th} feature in the input) is not relevant for the problem , the corresponding bandwidth parameter γ_d (which can be learned by MLE-II) will tend to take a very high value and this feature will not contribute to the above summation, and therefore won't contribute to the similarity computation.

Mid-sem exam: Q8

8. Consider N observations $\boldsymbol{y} = \{y_n\}_{n=1}^N$ assumed generated i.i.d. from a Student-t distribution $p(y_n|\mu,\sigma^2,\nu) = \mathcal{T}(y_n|\mu,\sigma^2,\nu)$, where μ is the "location" parameter (akin to mean), σ^2 is the "scale" parameter (akin to spread/variance), and ν is the degrees of freedom. Using $\mathcal{T}(y|\mu,\sigma^2,\nu) = \int \mathcal{N}(y|\mu,\sigma^2/z)\text{Gamma}(z|\frac{\nu}{2},\frac{\nu}{2})dz$, and assuming a prior $p(\mu) = \mathcal{N}(\mu|0,\tau^2)$, compute CP of μ and z_n.

   Note: $\mathcal{N}(x|\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}}\exp\{-\frac{(x-\mu)^2}{2\sigma^2}\}$ and $\text{Gamma}(x|a,b) = \frac{b^a}{\Gamma(b)}x^{a-1}\exp(-bx)$. Also, if needed, you may use the result that if $p(\boldsymbol{x}|\boldsymbol{z}) = \mathcal{N}(\boldsymbol{x}|\mathbf{A}\boldsymbol{z}+\boldsymbol{b},\mathbf{L}^{-1})$, $p(\boldsymbol{z}) = \mathcal{N}(\boldsymbol{\mu},\boldsymbol{\Lambda}^{-1})$, then the "reverse conditional" is given by $p(\boldsymbol{z}|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{z}|\boldsymbol{\Sigma}\left\{\mathbf{A}^\top\mathbf{L}(\boldsymbol{x}-\boldsymbol{b})+\boldsymbol{\Lambda\mu}\right\},\boldsymbol{\Sigma})$, where $\boldsymbol{\Sigma} = (\boldsymbol{\Lambda}+\mathbf{A}^\top\mathbf{L}\mathbf{A})^{-1}$

   The model has N additional latent variables $\boldsymbol{z} = \{z_1, z_2, \ldots, z_N\}$ and augmenting the model with these latent variables, the original Student-t distribution based likelihood for y_n becomes a Gaussian likelihood when conditioned on z_n.

Including these latent variables in our model, the joint distribution of the data and unknowns can be written as $p(\boldsymbol{y},\boldsymbol{z},\mu|\tau^2,\sigma^2,\nu) = p(\boldsymbol{y}|\boldsymbol{z},\mu,\sigma^2)p(\boldsymbol{z}|\nu)p(\mu|\tau^2) = \prod_{n=1}^N[p(y_n|z_n,\mu,\sigma^2)p(z_n|\nu)]p(\mu|\tau^2)$. CP of z_n is proportional to the product of terms containing z_n, i.e.,

$$p(z_n|y_n,\mu,\sigma^2,\nu) \propto p(y_n|z_n,\mu,\sigma^2)p(z_n|\nu)$$

which is a product of Gaussian and gamma with mean of the Gaussian fixed at μ. Therefore, we have conjugacy and it is easy to see that the CP will also be Gamma $\left(z_n|\frac{\nu}{2}+\frac{1}{2},\frac{\nu}{2}+\frac{(y_n-\mu)^2}{2\sigma^2}\right)$. Likewise, the CP of μ will be propotional to the product of terms containing μ, i.e.,

$$p(\mu|\boldsymbol{y},\tau^2) \propto \prod_{n=1}^N[p(y_n|\mu,z_n,\sigma^2)]p(\mu|\tau^2)$$

Mid-sem exam: Q8 (contd)

$$p(\mu|\boldsymbol{y},\tau^2) \propto \prod_{n=1}^{N}[p(y_n|\mu,z_n,\sigma^2)]p(\mu|\tau^2)$$

This is basically a Gaussian observation model with each of the N observations $\boldsymbol{y} = \{y_n\}_{n=1}^N$ having Gaussian likelihood $p(y_n|\mu,z_n,\sigma^2) = \mathcal{N}(y_n|\mu,\sigma^2/z_n)$ with μ having $\mathcal{N}(\mu|0,\tau^2)$ prior. Note that we can collectively define the likelihood of the vector $\boldsymbol{y}$ as an N-dimensional Gaussian $\mathcal{N}(\boldsymbol{y}|\boldsymbol{a}\mu,\mathbf{L}^{-1})$ where $\boldsymbol{a}$ is a column vector of all 1s, and $\mathbf{L}^{-1}$ is a diagonal matrix with the n-th diagonal entry equal to σ^2/z_n. Using this likelihood $\mathcal{N}(\boldsymbol{y}|\boldsymbol{a}\mu,\mathbf{L}^{-1})$ and the prior $\mathcal{N}(\mu|0,\tau^2)$, and the reverse conditional formula, we can see that the CP of μ will be $p(\mu|\boldsymbol{y},\tau^2) = \mathcal{N}(\mu|\mu_N,\sigma_N^2)$, where

$$\begin{aligned}\frac{1}{\sigma_N^2} &= \frac{1}{\tau^2} + \frac{1}{\sigma^2}\sum_{n=1}^{N} z_n \\ \mu_N &= \frac{\sigma_N^2}{\sigma^2}\sum_{n=1}^{N} y_n z_n\end{aligned}$$

Recap: Variational Inference (VI)

Variational distribution

Variational parameters

- Assuming $p(\boldsymbol{Z}|\boldsymbol{\mathcal{D}},\Theta)$ is intractable, VI approximates it by a distr $q(\boldsymbol{Z}|\phi)$ or $q_\phi(\boldsymbol{Z})$

KL minimization

$$\phi^* = \text{argmin}_\phi \, \text{KL}[q_\phi(\boldsymbol{Z})||p(\boldsymbol{Z}|\boldsymbol{\mathcal{D}},\Theta)]$$

ELBO maximization

$$\phi^* = \text{argmax}_\phi \mathbb{E}_{q_\phi(\boldsymbol{Z})}[\log p(\boldsymbol{\mathcal{D}}|\boldsymbol{Z},\Theta)] - \text{KL}[q_\phi(\boldsymbol{Z})||p(\boldsymbol{Z}|\Theta)]$$
$$= \text{argmax}_\phi \mathbb{E}_{q_\phi(\boldsymbol{Z})}\left[\log p(\boldsymbol{\mathcal{D}},\boldsymbol{Z}|\Theta) - \log q_\phi(\boldsymbol{Z})\right] = \text{argmax}_\phi \, \mathcal{L}(\phi,\Theta)$$

Can use gradient-based optimization to learn the parameters of the variational distribution

$$\phi_{t+1} = \phi_t + \eta_t \nabla_{\phi=\phi_t}\mathcal{L}(\phi,\Theta)$$

Mean-field assumption on the variational distribution

$$q(\boldsymbol{Z}|\phi) = \prod_{i=1}^{M} q(\boldsymbol{Z}_i|\phi_i)$$

$\mathbb{E}_{i\neq j}$ denotes expectations w.r.t. $\prod_{i\neq j} q(Z_i|\phi_i)$

$$q_j^*(\boldsymbol{Z}_j) = \frac{\exp(\mathbb{E}_{i\neq j}[\log p(\boldsymbol{\mathcal{D}},\boldsymbol{Z}|\Theta)])}{\int \exp(\mathbb{E}_{i\neq j}[\log p(\boldsymbol{\mathcal{D}},\boldsymbol{Z}|\Theta)]\, d\boldsymbol{Z}_j}$$

Equivalent to writing $\log q_j^*(\boldsymbol{Z}_j) = \mathbb{E}_{i\neq j}[\log p(\boldsymbol{\mathcal{D}},\boldsymbol{Z}|\Theta)] + \text{const}$

Variational EM

- In LVMs, latent vars $\boldsymbol{Z}$ and parameters $\boldsymbol{\Theta}$ both may be unknown. In such cases, we can use variational EM (VEM). Same as EM except VEM uses VI to approx. CP of $\boldsymbol{Z}$
- VEM alternates between the following two steps
  - Maximize the ELBO w.r.t. ϕ (gives the variational approximation $q(\boldsymbol{Z})$ of CP of $\boldsymbol{Z}$)

$$\phi^{(t)} = \text{argmax}_\phi \, \mathbb{E}_{q_\phi(\boldsymbol{Z})}\left[\log p\left(\mathcal{D}, \boldsymbol{Z} \middle| \Theta^{(t-1)}\right) - \log q_\phi(\boldsymbol{Z})\right]$$

  - Maximize the ELBO w.r.t. $\boldsymbol{\Theta}$ (gives us point estimate of $\boldsymbol{\Theta}$)

$$\Theta^{(t)} = \text{argmax}_\Theta \, \mathbb{E}_{q_{\phi^{(t)}}(\boldsymbol{Z})}\left[\log p(\mathcal{D}, \boldsymbol{Z}|\Theta) - \log q_{\phi^{(t)}}(\boldsymbol{Z})\right]$$

$$= \text{argmax}_\Theta \, \mathbb{E}_{q_{\phi^{(t)}}(\boldsymbol{Z})}[\log p(\mathcal{D}, \boldsymbol{Z}|\Theta)]$$

This looks very similar to the expected CLL with the CP replaced by its variational approximation

- Note: If we want posterior for $\boldsymbol{\Theta}$ as well, treat it similar to $\boldsymbol{Z}$ and apply variational approximation (instead of using VEM) if the posterior isn't tractable

VI for models without "latent variables"

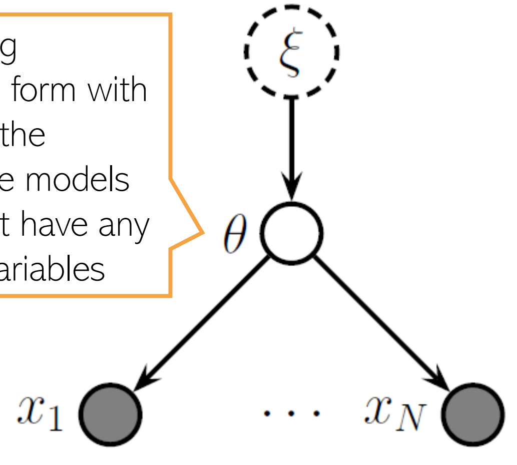
- Suppose we have a "fully observed" case (no missing data/latent variables but just some unknown global parameters $\boldsymbol{\theta}$ and known hyperparams ξ)
- A simple example of the model is shown in the figure below

$$p(\mathcal{D}, \boldsymbol{\theta}|\boldsymbol{\xi}) = p(\boldsymbol{\theta}|\boldsymbol{\xi}) \prod_{n=1}^{N} p(\boldsymbol{x}_n|\boldsymbol{\theta})$$

If this CP is intractable, we can use VI to approximate this

$$p(\boldsymbol{\theta}|\mathcal{D}, \boldsymbol{\xi}) = \frac{p(\mathcal{D}|\boldsymbol{\theta})p(\boldsymbol{\theta}|\boldsymbol{\xi})}{p(\mathcal{D}|\boldsymbol{\xi})}$$

Even supervised learning problems may have this form with $\boldsymbol{\theta}$ being the weights of the generative/discriminative models and the models may not have any missing data or latent variables



- If ξ are also unknown then one way would be to alternate like Variational EM
  - Approximating the CP $p(\theta|\mathcal{D}, \xi)$ using VI
  - Using MLE-II to get point estimates of the hyperparameters ξ

Mean-Field VI

- Since $\log q_j^*(\boldsymbol{Z}_j) = \mathbb{E}_{i\neq j}[\log p(\boldsymbol{X},\boldsymbol{Z})]$ + const = $\mathbb{E}_{i\neq j}\left[\log p(\boldsymbol{X},\boldsymbol{Z}_j,\boldsymbol{Z}_{-j})\right]$ + const

$$\log q_j^*(\mathbf{Z}_j) = \mathbb{E}_{i\neq j}[\log p(\mathbf{Z}_j|\mathbf{X},\mathbf{Z}_{-j})] + \text{const}$$

For any model

- Thus opt variational distr $q_j^*(\boldsymbol{Z}_j)$ basically requires expectations of CP $p(\boldsymbol{Z}_j|\boldsymbol{X},\boldsymbol{Z}_{-j})$
- For locally conjugate models, we know CP is easy and is an exp-fam distr of the form

$$p(\mathbf{Z}_j|\mathbf{X},\mathbf{Z}_{-j}) \quad = \quad h(\mathbf{Z}_j)\exp\left[\eta(\mathbf{X},\mathbf{Z}_{-j})^\top\mathbf{Z}_j - A(\eta(\mathbf{X},\mathbf{Z}_{-j}))\right]$$

- Using the above, we can rewrite the optimal variational distribution as follows

$$\log q_j^*(\mathbf{Z}_j) \quad = \quad \mathbb{E}_{i\neq j}\left[\log\left(h(\mathbf{Z}_j)\exp\left[\eta(\mathbf{X},\mathbf{Z}_{-j})^\top\mathbf{Z}_j - A(\eta(\mathbf{X},\mathbf{Z}_{-j}))\right]\right)\right] + \text{const}$$

$$\Longrightarrow q_j^*(\mathbf{Z}_j) \quad \propto \quad h(\mathbf{Z}_j)\exp\left[\mathbb{E}_{i\neq j}[\eta(\mathbf{X},\mathbf{Z}_{-j})]^\top\mathbf{Z}_j\right] \qquad \text{(verify)}$$

- Thus, with local conj, we just require expectation of nat. params. of CP of $\boldsymbol{Z}_j$

Making VI Faster for LVMs: Stochastic VI (SVI)

- Many LVMs have local latent variables $\boldsymbol{Z} = \{\boldsymbol{z}_1, \boldsymbol{z}_2, \dots, \boldsymbol{z}_N\}$ and global params $\boldsymbol{\Theta}$
- VI updates of local and global variables depend on each other (similar to EM)
- This makes things slow (for VI and also for EM) especially when N is large
  - We must update $q(\boldsymbol{z}_n|\boldsymbol{\phi}_n)$, i.e., compute $\boldsymbol{\phi}_n$, for each latent variable before updating $\boldsymbol{\Theta}$
- Also need all the data $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_N\}$ in memory to do these updates
- Stochastic VI* is an efficient way using minibatches of data
- In each iteration, SVI takes a minibatch $\mathcal{B}$ of $|\mathcal{B}| \ll N$ data points, updates $q(\boldsymbol{z}_n|\boldsymbol{\phi}_n)$ examples in that minibatches and approximates the ELBO as follows

Optimize this approximate ELBO w.r.t. $\boldsymbol{\Theta}$ (note: this is an unbiased estimate*)

$$\tilde{\mathcal{L}}(\phi, \Theta) = \frac{N}{B} \sum_{x_i \in \mathcal{B}} \mathbb{E}_{q(\boldsymbol{z}_i|\phi_i)}[\log p(\boldsymbol{x}_i|\boldsymbol{z}_i, \Theta)] - \mathrm{KL}[q_\phi(\boldsymbol{z}_i)||p(\boldsymbol{z}_i|\Theta)]$$

*Stochastic Variational Inference (Hoffman et al, 2013)

Making VI Faster for LVMs: Amortized VI

- Instead of computing the optimal $\boldsymbol{\phi}_n$ for each $\boldsymbol{q(z_n|\phi_n)}$, learn a function to do so

$$q(z_n|\phi_n) \approx q(z_n|\hat{\phi}_n) \quad \text{where} \quad \hat{\phi}_n = \text{NN}_\phi(x_n)$$

- Function is usually a neural network with weights $\boldsymbol{\phi}$
  - Usually referred to as "inference network" or "recognition model"
- Amortization: We are shifting the cost of finding $\boldsymbol{\phi}_n$ for each data point to finding the weights $\boldsymbol{\phi}$ of the neural network shared by all data points
- Can also combine amortized VI with stochastic VI
  - Each iteration only uses a minibatch to optimize NN weights $\boldsymbol{\phi}$ and global params $\boldsymbol{\Theta}$
- ELBO expression remains the same but $q(z_n|\phi_n)$ is replaced by $q(z_n|\text{NN}_\phi(x_n))$
- Amortized VI quality can be poor but it is fast and can give a quick solution
  - We can refine this solution other methods (e.g., using sampling; will see later)
  - This refinement based approach is called "semi-amortized VI"

VI using ELBO's gradients

- For simple locally conjugate models, VI updates are usually easy
  - Sometimes, can find the optimal $\boldsymbol{q}$ even without taking the ELBO's gradients
- For complex models, we have to use the more general gradient-based approach
- Consider the setting when we have latent variables $\boldsymbol{Z}$ and parameters $\boldsymbol{\Theta}$
- The ELBO's gradient w.r.t. model parameters $\boldsymbol{\Theta}$

$$\nabla_\Theta \mathcal{L}(\phi, \Theta) = \nabla_\Theta \mathbb{E}_{q_\phi(\boldsymbol{Z})}[\log p(\boldsymbol{\mathcal{D}}, \boldsymbol{Z}|\Theta) - \log q_\phi(\boldsymbol{Z})]$$

$$= \mathbb{E}_{q_\phi(\boldsymbol{Z})}\left[\nabla_\Theta \{\log p(\boldsymbol{\mathcal{D}}, \boldsymbol{Z}|\Theta) - \log q_\phi(\boldsymbol{Z})\}\right]$$

Gradient can go inside expectation since $q(Z)$ doesn't depend on Θ

Monte-Carlo approximation using samples of $q_\phi(\boldsymbol{Z})$ is straightforward here

- The ELBO's gradient w.r.t. parameters $\boldsymbol{\phi}$ of the variational distribution(s)

$$\nabla_\phi \mathcal{L}(\phi, \Theta) = \nabla_\phi \mathbb{E}_{q_\phi(\boldsymbol{Z})}[\log p(\boldsymbol{\mathcal{D}}, \boldsymbol{Z}|\Theta) - \log q_\phi(\boldsymbol{Z})]$$

$$\neq \mathbb{E}_{q_\phi(\boldsymbol{Z})}\left[\nabla_\phi \{\log p(\boldsymbol{\mathcal{D}}, \boldsymbol{Z}|\Theta) - \log q_\phi(\boldsymbol{Z})\}\right]$$

Gradient can't go inside expectation since $q(Z)$ depends on ϕ

Monte-Carlo approximation using samples of $q_\phi(\boldsymbol{Z})$ is NOT as straightforward

Black-Box Variational Inference (BBVI)

- Black-box Var. Inference* (BBVI) approximates ELBO derivatives using Monte-Carlo
- Uses the following identity for the ELBO's derivative

$$\begin{aligned}\nabla_\phi \mathcal{L}(q) &= \nabla_\phi \mathbb{E}_q[\log p(\mathbf{X},\mathbf{Z}) - \log q(\mathbf{Z}|\phi)] \\ &= \mathbb{E}_q[\nabla_\phi \log q(\mathbf{Z}|\phi)(\log p(\mathbf{X},\mathbf{Z}) - \log q(\mathbf{Z}|\phi))] \quad \text{(proof on next slide}\end{aligned}$$

- Thus ELBO gradient can be written solely in terms of expec. of gradient of $\log q(\boldsymbol{Z}|\boldsymbol{\phi})$
  - Required gradients don't depend on the model; only on chosen var. distribution (hence "black-box")
- Given S samples $\{Z_s\}_{s=1}^S$ from $q(\boldsymbol{Z}|\boldsymbol{\phi})$, we can get (noisy) gradient as follows

$$\nabla_\phi \mathcal{L}(q) \approx \frac{1}{S}\sum_{s=1}^{S} \nabla_\phi \log q(\mathbf{Z}_s|\phi)(\log p(\mathbf{X},\mathbf{Z}_s) - \log q(\mathbf{Z}_s|\phi))$$

- Above is also called the "score function" based gradient (also REINFORCE method)

Gradient of a log-likelihood or log-probability function w.r.t. its params is called score function; hence the name

*Black Box Variational Inference - Ranganath et al (2014)

Proof of BBVI Identity

- The ELBO gradient can be written as

$$\begin{aligned}
\nabla_\phi \mathcal{L}(q) &= \nabla_\phi \int (\log p(\mathbf{X},\mathbf{Z}) - \log q(\mathbf{Z}|\phi)) q(\mathbf{Z}|\phi) d\mathbf{Z} \\
&= \int \nabla_\phi [(\log p(\mathbf{X},\mathbf{Z}) - \log q(\mathbf{Z}|\phi)) q(\mathbf{Z}|\phi)] d\mathbf{Z} \qquad (\nabla \text{ and } \int \text{ interchangeable; dominated convergence theorem}) \\
&= \int \nabla_\phi [(\log p(\mathbf{X},\mathbf{Z}) - \log q(\mathbf{Z}|\phi))] q(\mathbf{Z}|\phi) + \nabla_\phi q(\mathbf{Z}|\phi)[(\log p(\mathbf{X},\mathbf{Z}) - \log q(\mathbf{Z}|\phi))] d\mathbf{Z} \\
&= \mathbb{E}_q[-\nabla_\phi \log q(\mathbf{Z}|\phi)] + \int \nabla_\phi q(\mathbf{Z}|\phi)[(\log p(\mathbf{X},\mathbf{Z}) - \log q(\mathbf{Z}|\phi))] d\mathbf{Z}
\end{aligned}$$

- Note that $\mathbb{E}_q[\nabla_\phi \log q(\mathbf{Z}|\phi)] = \mathbb{E}_q\left[\frac{\nabla_\phi q(\mathbf{Z}|\phi)}{q(\mathbf{Z}|\phi)}\right] = \int \nabla_\phi q(\mathbf{Z}|\phi) d\mathbf{Z} = \nabla_\phi \int q(\mathbf{Z}|\phi) d\mathbf{Z} = \nabla_\phi 1 = 0$
- Also note that $\nabla_\phi q(\mathbf{Z}|\phi) = \nabla_\phi [\log q(\mathbf{Z}|\phi)] q(\mathbf{Z}|\phi)$, using which

$$\begin{aligned}
\int \nabla_\phi q(\mathbf{Z}|\phi)[(\log p(\mathbf{X},\mathbf{Z}) - \log q(\mathbf{Z}|\phi))] d\mathbf{Z} &= \int \nabla_\phi \log q(\mathbf{Z}|\phi)[(\log p(\mathbf{X},\mathbf{Z}) - \log q(\mathbf{Z}|\phi))] q(\mathbf{Z}|\phi) d\mathbf{Z} \\
&= \mathbb{E}_q[\nabla_\phi \log q(\mathbf{Z}|\phi)(\log p(\mathbf{X},\mathbf{Z}) - \log q(\mathbf{Z}|\phi))]
\end{aligned}$$

- Therefore $\nabla_\phi \mathcal{L}(q) = \mathbb{E}_q[\nabla_\phi \log q(\mathbf{Z}|\phi)(\log p(\mathbf{X},\mathbf{Z}) - \log q(\mathbf{Z}|\phi))]$

Benefits of BBVI

- Recall that BBVI approximates the ELBO gradients by the Monte Carlo expectations

$$\nabla_\phi \mathcal{L}(q) \approx \frac{1}{S}\sum_{s=1}^{S} \nabla_\phi \log q(\mathbf{Z}_s|\phi)(\log p(\mathbf{X},\mathbf{Z}_s) - \log q(\mathbf{Z}_s|\phi))$$

- Enables applying VI for a wide variety of probabilistic models
- Can also work with small minibatches of data rather than full data
- BBVI has very few requirements
  - Should be able to sample from $q(\boldsymbol{Z}|\boldsymbol{\phi})$ (usually sampling routines exists!)
  - Should be able to compute $\nabla_{\boldsymbol{\phi}} \log q(\boldsymbol{Z}|\boldsymbol{\phi})$ (automatic differentiation methods exist!)
  - Should be able to evaluate $\log p(\boldsymbol{X},\boldsymbol{Z})$ and $\log q(\boldsymbol{Z}|\boldsymbol{\phi})$ for any value of $\boldsymbol{Z}$
- Some tricks needed to control the variance in the Monte Carlo estimate of the ELBO gradient (if interested in the details, please refer to the BBVI paper)

Reparametrization Trick

- Another Monte-Carlo approx. of ELBO grad (with often lower var than BBVI gradient)
- Suppose we want to compute ELBO's gradient $\nabla_\phi \mathbb{E}_{q_\phi(\mathbf{Z})}[\log p(\mathbf{X},\mathbf{Z}) - \log q_\phi(\mathbf{Z})]$
- Assume a deterministic transformation g

$$\boldsymbol{Z} = g(\epsilon, \phi) \quad \text{where} \quad \epsilon \sim p(\epsilon)$$

Assumed to not depend on ϕ

- With this reparametrization, and using LOTUS rule, the ELBO's gradient would be

$$\nabla_\phi \mathbb{E}_{p(\epsilon)}[\log p(\mathbf{X}, g(\epsilon,\phi)) - \log q_\phi(g(\epsilon,\phi))] = \mathbb{E}_{p(\epsilon)} \nabla_\phi [\log p(\mathbf{X}, g(\epsilon,\phi)) - \log q_\phi(g(\epsilon,\phi))]$$

- Given S i.i.d. random samples $\{\epsilon_s\}_{s=1}^S$ from $p(\epsilon)$, we can get a Monte-Carlo approx.

$$\nabla_\phi \mathbb{E}_{q_\phi(\mathbf{Z})}[\log p(\mathbf{X},\mathbf{Z}) - \log q_\phi(\mathbf{Z})] \approx \frac{1}{S}\sum_{s=1}^{S}[\nabla_\phi \log p(\mathbf{X}, g(\epsilon_s,\phi)) - \nabla_\phi \log q_\phi(g(\epsilon_s,\phi))]$$

- Such gradients are called pathwise gradients* (since we took a "path" from ϵ to $\boldsymbol{Z}$)

*Autoencoding Variational Bayes - Kingma and Welling (2013), Stochastic Backpropagation and Approximate Inference in Deep Generative Models- Rezende et al (2014)

Reparametrization Trick: An Example

- Suppose our variational distribution is $q(\boldsymbol{w}|\phi) = \mathcal{N}(\boldsymbol{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$, so $\phi = \{\boldsymbol{\mu}, \boldsymbol{\Sigma}\}$
- Suppose our ELBO has a difficult expectation term $\mathbb{E}_q[f(\boldsymbol{w})]$

Or $\phi = \{\boldsymbol{\mu}, \mathbf{L}\}$ where $\mathbf{L} = \text{chol}(\Sigma)$

- However, note that we need ELBO gradient, not ELBO itself. Let's use the trick
- Reparametrize $\boldsymbol{w}$ as $\boldsymbol{w} = \boldsymbol{\mu} + \mathbf{Lv}$ where $\mathbf{v} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$

Note that we will still have $q(\boldsymbol{w}|\phi) = \mathcal{N}(\boldsymbol{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$

$$\nabla_{\mu,\mathbf{L}}\mathbb{E}_{\mathcal{N}(\mathbf{w}|\mu,\Sigma)}[f(\mathbf{w})] = \nabla_{\mu,\mathbf{L}}\mathbb{E}_{\mathcal{N}(\mathbf{v}|0,\mathbf{I})}[f(\mu + \mathbf{Lv})] = \mathbb{E}_{\mathcal{N}(\mathbf{v}|0,\mathbf{I})}[\nabla_{\mu,\mathbf{L}} f(\mu + \mathbf{Lv})]$$

- The above is now straightforward
  - Easily take derivatives of $f(\boldsymbol{w})$ w.r.t. variational params $\boldsymbol{\mu}, \mathbf{L}$
  - Replace exp. by Monte-Carlo averaging using samples of $\mathbf{v}$ from $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$

Often even one or very few samples suffice

$$\begin{aligned}\nabla_{\mu}\mathbb{E}_{\mathcal{N}(\mathbf{w}|\mu,\Sigma)}[f(\mathbf{w})] &= \mathbb{E}_{\mathcal{N}(\mathbf{v}|0,\mathbf{I})}[\nabla_{\mu} f(\mu + \mathbf{Lv})] \approx \nabla_{\mu} f(\mu + \mathbf{Lv}_s) \rightarrow \frac{\partial f}{\partial \boldsymbol{w}}\frac{\partial \boldsymbol{w}}{\partial \boldsymbol{\mu}} \\ \nabla_{\mathbf{L}}\mathbb{E}_{\mathcal{N}(\mathbf{w}|\mu,\Sigma)}[f(\mathbf{w})] &= \mathbb{E}_{\mathcal{N}(\mathbf{v}|0,\mathbf{I})}[\nabla_{\mathbf{L}} f(\mu + \mathbf{Lv})] \approx \nabla_{\mathbf{L}} f(\mu + \mathbf{Lv}_s) \rightarrow \frac{\partial f}{\partial \boldsymbol{w}}\frac{\partial \boldsymbol{w}}{\partial \mathbf{L}}\end{aligned}$$

Chain Rule

- Std. reparam. trick assumes differentiability (recent work on removing this req).

*Autoencoding Variational Bayes - Kingma and Welling (2013), Stochastic Backpropagation and Approximate Inference in Deep Generative Models- Rezende et al (2014)

Reparametrization Trick: Some Comments

- Standard Reparametrization Trick assumes the model to be differentiable

$$\nabla_\phi \mathbb{E}_{q_\phi(\mathbf{Z})}[\log p(\mathbf{X},\mathbf{Z}) - \log q_\phi(\mathbf{Z})] = \mathbb{E}_{p(\epsilon)}[\nabla_\phi \log p(\mathbf{X}, g(\epsilon,\phi)) - \nabla_\phi \log q_\phi(g(\epsilon,\phi))]$$

- In contrast, BBVI (score function gradients) only required $q(\boldsymbol{Z})$ to be differentiable
- Thus rep. trick often isn't applicable, e.g., when $\boldsymbol{Z}$ is discrete (e.g., binary /categorical)
  - Recent work on continuous relaxation† of discrete variables† (e.g., Gumbel Softmax for categorical)
- The transformation function $\boldsymbol{g}$ may be difficult to find for general distributions
  - Recent work on generalized reparametrizations*
- Also, the transformation function $\boldsymbol{g}$ needs to be invertible (difficult/expensive)
  - Recent work on implicit reparametrized gradients#
- Assumes that we can directly draw samples from $\boldsymbol{p}(\boldsymbol{\epsilon})$. If not, then rep. trick isn't valid@

†Categorical Reparameterization with Gumbel-Softmax (Jang et al, 2017), * The Generalized Reparameterization Gradient (Ruiz et al, 2016), # Implicit Reparameterization Gradients (Figurnov et al, 2018), @ Reparameterization Gradients through Acceptance-Rejection Sampling Algorithms (Naesseth et al, 2016)

Automatic Differentiation Variational Inference

- Suppose $\boldsymbol{Z}$ is D-dim r.v. with constraints (e.g., non-negativity) and distribution $q(\boldsymbol{Z}|\phi)$
- Assume a transformation T such that $\boldsymbol{u} = T(\boldsymbol{Z})$ s.t. $\boldsymbol{u} \in \mathbb{R}^D$ (unconstrained) then

$$q(\boldsymbol{u}) = q(\boldsymbol{Z}) \left| \det\left(\frac{\partial \boldsymbol{Z}}{\partial \boldsymbol{u}}\right) \right|$$

- Assuming $q(\boldsymbol{u}|\psi) = \mathcal{N}(\boldsymbol{u}|\mu, \Sigma)$, the ELBO becomes

Original ELBO for $q(\boldsymbol{Z}|\phi)$

$$\mathcal{L}(\phi) = \mathbb{E}_{q(\boldsymbol{Z}|\phi)}[\log p(\mathcal{D}|\boldsymbol{Z}) + \log p(\boldsymbol{Z})] + \mathrm{H}(q(\boldsymbol{Z}|\phi))$$

Transformed, equivalent ELBO

Easier to optimize since $q(u|\psi)$ is Gaussian

$$\mathcal{L}(\psi) = \mathbb{E}_{q(\boldsymbol{u}|\psi)}\left[\log p(\mathcal{D}|T^{-1}(\boldsymbol{u})) + \log p\left(T^{-1}(\boldsymbol{u})\right) + \log\left|\det\left(\frac{\partial \boldsymbol{Z}}{\partial \boldsymbol{u}}\right)\right|\right] + \mathrm{H}(q(\boldsymbol{u}|\psi))$$

- We can optimize the above ELBO w.r.t. ψ to get $q(\boldsymbol{u}|\psi)$ as a Gaussian
- The transformed density $q(\boldsymbol{Z}|\phi)$ can be found using the transformation equation

Structured Variational Inference

- Here "structured" may refer to anything that makes VI approx. more expressive, e.g.,
  - Removing the independence assumption of mean-field VI
  - In general, learning more complex forms for the variational approximation family $q(\boldsymbol{Z}|\phi)$
- To remove the mean-field assumption in VI, various approaches exist
  - Structured mean-field (Saul et al, 1996)
  - Hierarchical VI (Ranganath et al, 2016): Variational params $\phi_1, \phi_2, \ldots, \phi_M$ "tied" via a shared prior

$$q(\boldsymbol{z}_1, \ldots, \boldsymbol{z}_M|\theta) = \int \left[\prod_{m=1}^{M} q(\boldsymbol{z}_m|\phi_m) \right] p(\phi|\theta) d\phi$$

- Recent work on learning more expressive variational approx. for general VI
  - Boosting or mixture of simpler distributions, e.g., $q(\boldsymbol{Z}) = \sum_{c=1}^{C} \rho_c q_c(\boldsymbol{Z})$

Even simple unimodal components will give a multimodal $q(\boldsymbol{Z})$

  - Normalizing flows*: Turn a simple var. distr. into a complex one via series of invertible transfor.

$$\mathbf{z}_K = f_K \circ \cdots \circ f_1(\mathbf{z}_0), \quad \mathbf{z}_0 \sim q_0(\mathbf{z}_0),$$

A simple unimodal variational distribution (e.g.. $\mathcal{N}(0, I)$)

$$\mathbf{z}_K \sim q_K(\mathbf{z}_K) = q_0(\mathbf{z}_0) \prod_{k=1}^{K} \left| \det \frac{\partial f_k}{\partial \mathbf{z}_{k-1}} \right|^{-1}$$

A much more complex (e.g., multimodal) variational distribution obtained via the flow idea

*Variational Inference with Normalizing Flows (Rezende and Mohamed, 2015)

Other Divergence Measures

- VI minimizes $KL(q||p)$ but other divergences can be minimized as well
  - Recall that VI with minimization of $KL(q||p)$ leads to underestimated variances
- A general form of divergence is Renyi's α-divergence defined as

$$D_\alpha^R(p(\mathbf{Z})||q(\mathbf{Z})) = \frac{1}{\alpha - 1} \log \int p(\mathbf{Z})^\alpha q(\mathbf{Z})^{1-\alpha} d\mathbf{Z}$$

- $KL(p||q)$ is a special case with $\alpha \to 1$ (can verify using L'Hopital rule of taking limits)
- An even more general form of divergence is f-Divergence

$$D_f(p(\mathbf{Z})||q(\mathbf{Z})) = \int q(\mathbf{Z}) f\left(\frac{p(\mathbf{Z})}{q(\mathbf{Z})}\right) d\mathbf{Z}$$

- Many recent variational inference algorithms are based on minimizing such divergences

Variational Inference: Some Comments

- Many probabilistic models nowadays rely on VI to do approx. inference
- Even mean-field with locally-conjugacy used in lots of models
  - This + SVI gives excellent scalability as well on large datasets
- Progress in various areas has made VI very popular and widely applicable
  - Stochastic Optimization (e.g., SGD)
  - Automatic Differentiation
  - Monte-Carlo gradient of ELBO

We covered many of the threads being explored in recent work but a lot of work still being done in this area

- Note: Most of these ideas apply also to Variational EM
- Many VI and advanced VI algos are implemented in probabilistic prog. packages (e.g., Tensorflow Probability, PyTorch, etc), making VI easy even for complex models
- Still a very active area of research, especially for doing VI in complex models
  - Models with discrete latent variables
  - Reducing the variance in Monte-Carlo estimate of ELBO gradients
  - More expressive variational distribution for better approximation