Variational Inference (wrap-up), Approximate Inference via Sampling

CS772A: Probabilistic Machine Learning

Piyush Rai

VI using ELBO's gradients

- For simple locally conjugate models, VI updates are usually easy
  - Sometimes, can find the optimal 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. $\boldsymbol{\Theta}$

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

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

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

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

- The ELBO's gradient w.r.t. $\boldsymbol{\phi}$

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

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

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

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

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 $\log q(\mathbf{Z}|\phi)$
  - 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 $q(\mathbf{Z}|\phi)$, 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)

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 $q(\boldsymbol{w}|\boldsymbol{\phi}) = \mathcal{N}(\boldsymbol{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$, so $\boldsymbol{\phi} = \{\boldsymbol{\mu}, \boldsymbol{\Sigma}\}$
- Suppose our ELBO has a difficult expectation term $\mathbb{E}_q[f(\boldsymbol{w})]$
- However, note that we need ELBO gradient, not ELBO itself. Let's use the trick
- Reparametrize $\boldsymbol{w}$ as $\boldsymbol{w} = \boldsymbol{\mu} + \mathbf{L}\mathbf{v}$ where $\mathbf{v} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$

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

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

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

- 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

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

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

$\frac{\partial f}{\partial \boldsymbol{w}} \frac{\partial \boldsymbol{w}}{\partial \boldsymbol{\mu}}$

Chain Rule

$\frac{\partial f}{\partial \boldsymbol{w}} \frac{\partial \boldsymbol{w}}{\partial \mathbf{L}}$

- 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 g may be difficult to find for general distributions
  - Recent work on generalized reparametrizations*
- Also, the transformation function g needs to be invertible (difficult/expensive)
  - Recent work on implicit reparametrized gradients#
- Assumes that we can directly draw samples from $p(\epsilon)$. 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 $q(\boldsymbol{Z}|\boldsymbol{\phi})$
- Assume a transformation T such that $\boldsymbol{u} = T(\boldsymbol{Z})$ s.t. $\boldsymbol{u} \in \mathbb{R}^D$ (unconstrained) then

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

- Recall the original ELBO expression $\mathcal{L}(\boldsymbol{\phi}) = \mathbb{E}_{q(\boldsymbol{Z}|\boldsymbol{\phi})} \left[\log \frac{p(\boldsymbol{D},\boldsymbol{Z})}{q(\boldsymbol{Z}|\boldsymbol{\phi})} \right]$
- Assuming $q(\boldsymbol{u}|\psi) = \mathcal{N}(\boldsymbol{u}|\mu, \Sigma)$ and using $\boldsymbol{Z} = T^{-1}(\boldsymbol{u})$, the ELBO becomes

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

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

- Optimize $\mathcal{L}(\psi)$ w.r.t. ψ to get $q(\boldsymbol{u}|\psi)$ as a Gaussian and use distribution transformation equation to get $q(\boldsymbol{Z}|\boldsymbol{\phi})$

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 $q(\boldsymbol{Z}|\boldsymbol{\phi})$
- 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 $\boldsymbol{\phi_1}, \boldsymbol{\phi_2}, \ldots, \boldsymbol{\phi_M}$ "tied" via a shared prior

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

- 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.

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

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

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

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

*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

Sampling for Approximate Inference

- Some typical tasks that we have to solve in probabilistic/fully-Bayesian inference

Posterior distribution

$$p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}|\theta)p(\theta)}{p(\mathcal{D})} = \frac{p(\mathcal{D}|\theta)p(\theta)}{\int p(\mathcal{D}|\theta)p(\theta)d\theta}$$

Posterior predictive distribution

$$p(\mathcal{D}^{new}|\mathcal{D}) = \int p(\mathcal{D}^{new}|\theta)p(\theta|\mathcal{D})d\theta = \mathbb{E}_{p(\theta|\mathcal{D})}[p(\mathcal{D}^{new}|\theta)]$$

Needed for model selection (and in computing posterior too)

Marginal likelihood

$$p(\mathcal{D}|m) = \int p(\mathcal{D}|\theta)p(\theta|m)d\theta = \mathbb{E}_{p(\theta|m)}[p(\mathcal{D}|\theta)]$$

Needed in EM

Expected complete data log-likelihood

$$\text{Exp-CLL} = \int p(\boldsymbol{z}|\theta, \boldsymbol{x})p(\boldsymbol{x}, \boldsymbol{z}|\theta)d\boldsymbol{z} = \mathbb{E}_{p(\boldsymbol{z}|\theta,\boldsymbol{x})}[p(\boldsymbol{x}, \boldsymbol{z}|\theta)]$$

Needed in VI

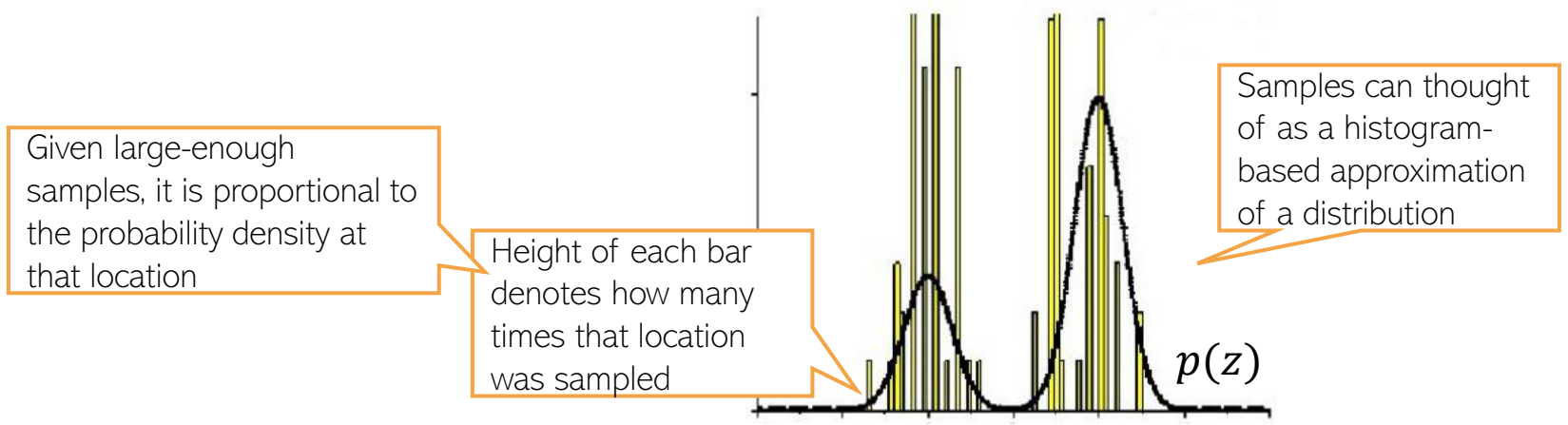
Evidence lower bound (ELBO)

$$\mathcal{L}(q) = \mathbb{E}_q[\log p(\boldsymbol{x}, \boldsymbol{z})] - \mathbb{E}_q[\log p(\boldsymbol{z})]$$

- Sampling methods provide a general way to (approximately) solve these problems
- More general than VI methods which only approximate the posterior distribution

Approximating a Prob. Distribution using Samples

- Can approximate any distribution using a set of randomly drawn samples from it



- The samples can also be used for computing expectations (Monte-Carlo averaging)
- Usually straightforward to generate samples if it is a simple/standard distribution
- The interesting bit: Even if the distribution is "difficult" (e.g., an intractable posterior), it is often possible to generate random samples from such a distribution, as we will see.

The Empirical Distribution

- Sampling based approx. can be formally represented using an empirical distribution
- Given L points/samples $\boldsymbol{z}^{(1)}, \boldsymbol{z}^{(2)}, \ldots, \boldsymbol{z}^{(L)}$, empirical distr. defined by these is

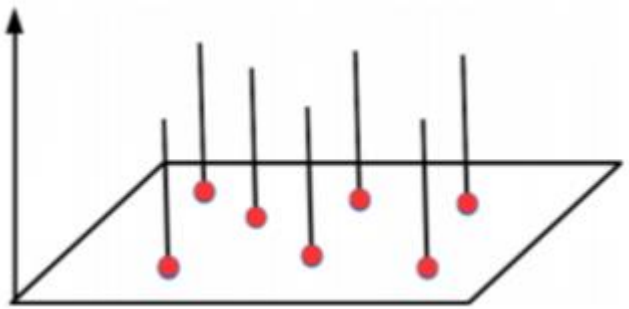
Dirac Distribution with finite support at $\boldsymbol{z}^{(1)}, \boldsymbol{z}^{(2)}, \ldots, \boldsymbol{z}^{(L)}$

Weights sum to 1

Weight of point $z^{(\ell)}$

$$p_L(A) = \sum_{\ell=1}^{L} w_\ell \delta_{\boldsymbol{z}^{(\ell)}}(A)$$

Can think of A as being the area over which we want to evaluate the distribution



Dirac Distribution

$$\delta_{\boldsymbol{z}}(A) = \begin{cases} 0 & \text{if} \quad \boldsymbol{z} \notin A \\ 1 & \text{if} \quad \boldsymbol{z} \in A \end{cases}$$

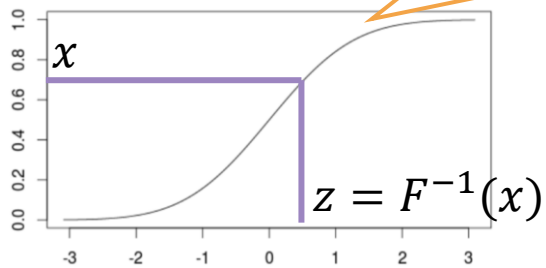
Sampling: Some Basic Methods

$$p(z) = q(x)\left|\frac{\partial x}{\partial z}\right|$$

Determinant of Jacobian

- Most of these basic methods are based on the idea of transformation
  - Generate a random sample x from a distribution $q(x)$ which is easy to sample from
  - Apply a transformation on x to make it random sample z from a complex distr $p(z)$

$F(z)$: CDF of $p(z)$



- Some popular examples of transformation methods
  - Inverse CDF method

$$x \sim \text{Unif}(0,1) \Rightarrow z = \text{Inv-CDF}_{p(z)}(x) \sim p(z)$$

  - Reparametrization method

$$x \sim \mathcal{N}(0,1) \Rightarrow z = \mu + \sigma x \sim \mathcal{N}(\mu, \sigma^2)$$

  - Box-Mueller method: Given (x_1, x_2) from $\mathbf{Unif}(0,1)$, generate (z_1, z_2) from $\mathcal{N}(0, \mathbf{I}_2)$

$$z_1 = \sqrt{-2\ln x_1}\cos(2\pi x_2), \quad z_1 = \sqrt{-2\ln x_1}\sin(2\pi x_2)$$

- Transformation Methods are simple but have limitations
  - Mostly limited to standard distributions and/or distributions with very few variables

Rejection Sampling

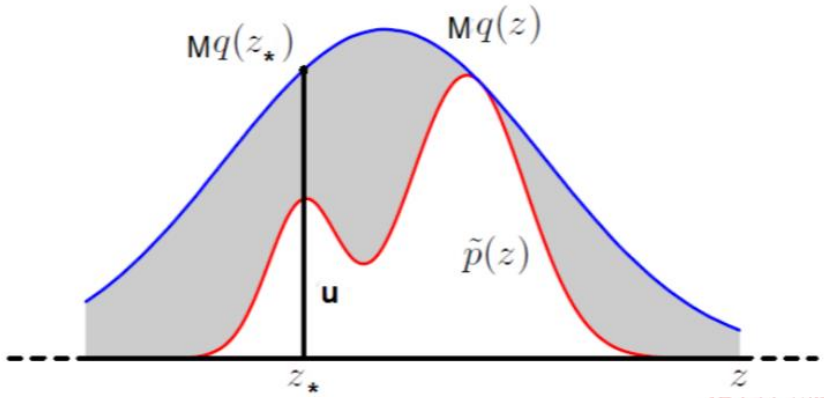
- Goal: Generate a random sample from a distribution of the form $p(z) = \frac{\tilde{p}(z)}{Z_p}$, assuming
  - We can only evaluate the value of numerator $\tilde{p}(z)$ for any z
  - The denominator (normalization constant) Z_p is intractable and we don't know its value

Should have the same support as $p(z)$

- Assume a proposal distribution $q(z)$ we can generate samples from, and

$$Mq(z) \geq \tilde{p}(z) \qquad \forall z \quad (\text{where } M > 0 \text{ is some const.})$$

- Rejection Sampling then works as follows
  - Sample a random variable z_* from $q(z)$
  - Sampling a uniform r.v. $u \sim \text{Unif}[0, Mq(z_*)]$
  - If $u \leq \tilde{p}(z_*)$ then accept z_*, otherwise reject it



- All accepted z_*'s will be random samples from $p(z)$. Proof on next slide

Rejection Sampling

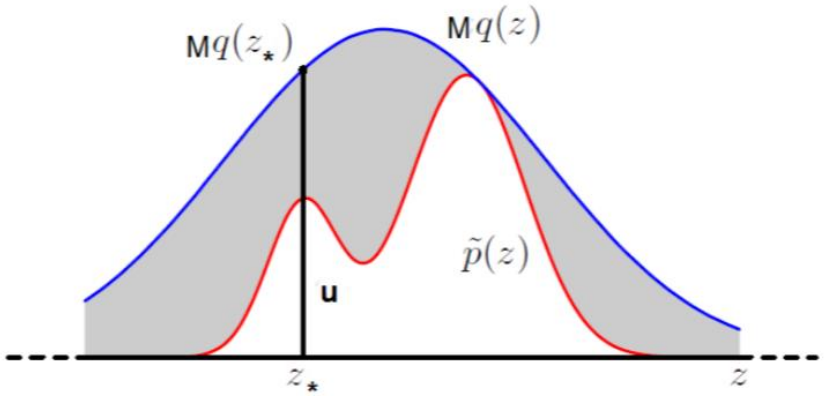
- Why $z \sim q(z)$ + accept/reject rule is equivalent to $z \sim p(z)$?
- Let's look at the pdf of the z's that were accepted, i.e., $p(z|\text{accept})$

$$p(\text{accept}|z) = \int_0^{\tilde{p}(z)} \frac{1}{Mq(z)} du = \frac{\tilde{p}(z)}{Mq(z)}$$

$$p(z, \text{accept}) = q(z)p(\text{accept}|z) = \frac{\tilde{p}(z)}{M}$$

$$p(\text{accept}) = \int \frac{\tilde{p}(z)}{M} dz = \frac{Z_p}{M}$$

$$p(z|\text{accept}) = \frac{p(z, \text{accept})}{p(\text{accept})} = \frac{\tilde{p}(z)}{Z_p} = p(z)$$



Computing Expectations via Monte Carlo Sampling

- Often we are interested in computing expectations of the form

$$\mathbb{E}[f] = \int f(z)p(z)dz$$

where $f(z)$ is some function of the random variable $z \sim p(z)$

- A simple approx. scheme to compute the above expectation: Monte Carlo integration
  - Generate L independent samples from $p(z)$: $\{z^{(\ell)}\}_{\ell=1}^{L} \sim p(z)$

Assuming we know how to sample from $p(z)$

  - Approximate the expectation by the following empirical average

$$\mathbb{E}[f] \approx \hat{f} = \frac{1}{L}\sum_{\ell=1}^{L} f(z^{(\ell)})$$

- Since the samples are independent of each other, we can show the following (exercise)

Unbiased expectation

$$\mathbb{E}[\hat{f}] = \mathbb{E}[f] \quad \text{and} \quad \text{var}[\hat{f}] = \frac{1}{L}\text{var}[f] = \frac{1}{L}\mathbb{E}[(f - \mathbb{E}[f])^2]$$

Variance in our estimate decreases as L increases

Computing Expectations via Importance Sampling

- How to compute Monte Carlo expec. if we don't know how to sample from $p(z)$?
- One way is to use transformation methods or rejection sampling
- Another way is to use Importance Sampling (assuming $p(z)$ can be evaluated at least)
  - Generate L indep samples from a proposal $q(z)$ we know how sample from: $\{z^{(\ell)}\}_{\ell=1}^{L} \sim q(z)$
  - Now approximate the expectation as follows

$$\mathbb{E}[f] = \int f(z)p(z)dz = \int f(z)\frac{p(z)}{q(z)}q(z)dz \approx \frac{1}{L}\sum_{\ell=1}^{L} f(z^{(\ell)})\frac{p(z^{(\ell)})}{q(z^{(\ell)})}$$

- This is basically "weighted" Monte Carlo integration
  - $w^{(\ell)} = \frac{p(z^{(\ell)})}{q(z^{(\ell)})}$ denotes the importance weight of each sample $z^{(\ell)}$
- IS works even when we can only evaluate $p(z) = \frac{\tilde{p}(z)}{Z_p}$ up to a prop. constant

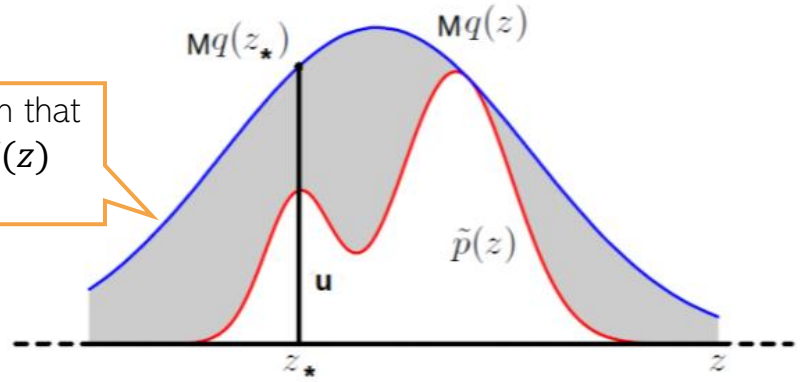
See PRML 11.1.4

- Note: Monte Carlo and Importance Sampling are NOT sampling methods!
  - These are only uses for computing expectations (approximately)

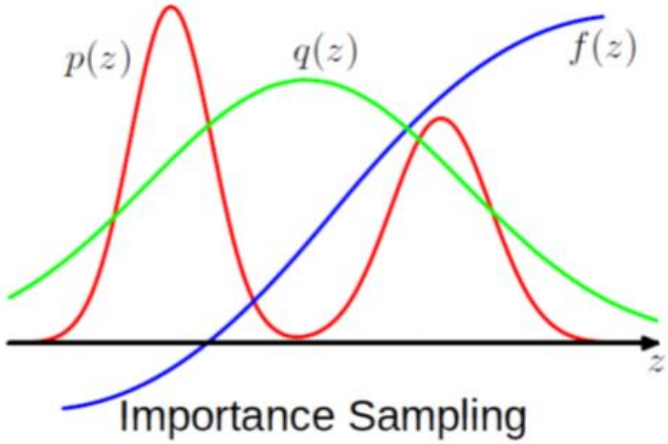
Limitations of the Basic Methods

- Transformation based methods: Usually limited to drawing from standard distributions
- Rejection Sampling and Importance Sampling: Require good proposal distributions

$q(z)$ should be such that $Mq(z)$ envelopes $\tilde{p}(z)$ everywhere



Rejection Sampling



Importance Sampling

$$\mathbb{E}[f] \approx \frac{1}{L}\sum_{\ell=1}^{L} f(z^{(\ell)})\frac{p(z^{(\ell)})}{q(z^{(\ell)})}$$

Ideally, would like $q(z)$ to give samples from where $p(z)$ is large or $f(z)p(z)$ is large

Difficult to guarantee so if z is high-dimensional

- In general, difficult to find good prop. distr. especially when z is high-dim
- More sophisticated sampling methods like MCMC work well in such high-dim spaces

Markov Chain Monte Carlo (MCMC)

- Goal: Generate samples from some target distribution $p(\boldsymbol{z}) = \frac{\tilde{p}(\boldsymbol{z})}{Z_p}$

If the target is a posterior, it will be conditioned on data, i.e., $p(\boldsymbol{z}|\boldsymbol{x})$

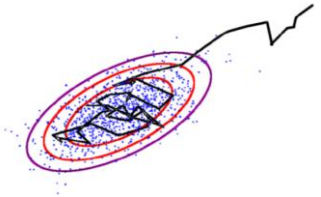
$\boldsymbol{z}$ usually is high-dim

- Assume we can evaluate $p(\boldsymbol{z})$ at least up to a proportionality constant

Means we can at least evaluate $\tilde{p}(\boldsymbol{z})$

- MCMC uses a Markov Chain which, when converged, starts giving samples from $p(\boldsymbol{z})$

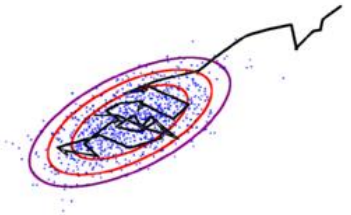
$$\underbrace{\boldsymbol{z}^{(1)} \rightarrow \boldsymbol{z}^{(2)} \rightarrow \boldsymbol{z}^{(3)} \rightarrow}_{\text{initial samples typically garbage}} \ldots \rightarrow \underbrace{\boldsymbol{z}^{(L-2)} \rightarrow \boldsymbol{z}^{(L-1)} \rightarrow \boldsymbol{z}^{(L)}}_{\text{after convergence, actual samples from } p(\boldsymbol{z})}$$



- Given current sample $\boldsymbol{z}^{(\ell)}$ from the chain, MCMC generates the next sample $\boldsymbol{z}^{(\ell+1)}$ as
  - Use a proposal distribution $q(\boldsymbol{z}|\boldsymbol{z}^{(\ell)})$ to generate a candidate sample $\boldsymbol{z}_*$
  - Accept/reject $\boldsymbol{z}_*$ as the next sample based on an acceptance criterion (will see later)
  - If accepted, set $\boldsymbol{z}^{(\ell+1)} = \boldsymbol{z}_*$. If rejected, set $\boldsymbol{z}^{(\ell+1)} = \boldsymbol{z}^{(\ell)}$
- Important: The proposal distribution $q(\boldsymbol{z}|\boldsymbol{z}^{(\ell)})$ depends on the previous sample $\boldsymbol{z}^{(\ell)}$

Should also have the same support as $p(\boldsymbol{z})$

MCMC: The Basic Scheme



- The chain run infinitely long (i.e., upon convergence) will give ONE sample from $p(z)$
- But we usually require several samples to approximate $p(z)$
- This is done as follows
  - Start the chain at an initial $z^{(0)}$
  - Using the proposal $q(z|z^{(\ell)})$, run the chain long enough, say T_1 steps
  - Discard the first $T_1 - 1$ samples (called "burn-in" samples) and take last sample $z^{(T_1)}$
  - Continue from $z^{(T_1)}$ up to T_2 steps, discard intermediate samples, take last sample $z^{(T_2)}$
    - This discarding (called "thinning") helps ensure that $z^{(T_1)}$ and $z^{(T_2)}$ are uncorrelated
  - Repeat the same for a total of S times
  - In the end, we now have S *approximately independent* samples from $p(z)$
- Note: Good choices for T_1 and $T_i - T_{i-1}$ (thinning gap) are usually based on heuristics

MCMC is exact in theory but approximate in practice since we can't run the chain for infinitely long in practice

Thus we say that the samples are approximately from the target distribution

Will treat it as our first sample from $p(z)$

Requirement for Monte Carlo approximation