

Discrete Latent Variables

Art Sobolev

Research Scientist at Samsung AI Center
@art_sobolev | artem.sobolev.name



1. Why Discreteness?

2. The Problem

3. Relaxations

4. Variance Reduction

5. Conclusion

Why Discreteness?

- ▶ Easier to interpret discrete categories than continuous spectrum
 - ▶ **Example:** Discrete Variational Autoencoder
 - ▶ Assume observations can be described by some binary (or categorical) code
 - ▶ We want to learn both encoder and decoder for such code and observations
- ▶ Allow the model to make a discrete choice
 - ▶ **Example:** Hard Attention
 - ▶ An attention module generates binary mask of where to look at
 - ▶ The network classifies masked images
 - ▶ We want attention module to attend only important areas of the image
- ▶ Sometimes you need discrete predictions to have certain properties
 - ▶ **Example:** GANs for text
 - ▶ Generator outputs discrete text
 - ▶ Discriminator takes discrete text as input and classifies how real it is
 - ▶ We want the generator to output text that fools the discriminator

- ▶ Easier to interpret discrete categories than continuous spectrum
 - ▶ **Example:** Discrete Variational Autoencoder
 - ▶ Assume observations can be described by some binary (or categorical) code
 - ▶ We want to learn both encoder and decoder for such code and observations
- ▶ Allow the model to make a discrete choice
 - ▶ **Example:** Hard Attention
 - ▶ An attention module generates binary mask of where to look at
 - ▶ The network classifies masked images
 - ▶ We want attention module to attend only important areas of the image
- ▶ Sometimes you need discrete predictions to have certain properties
 - ▶ **Example:** GANs for text
 - ▶ Generator outputs discrete text
 - ▶ Discriminator takes discrete text as input and classifies how real it is
 - ▶ We want the generator to output text that fools the discriminator

- ▶ Easier to interpret discrete categories than continuous spectrum
 - ▶ **Example:** Discrete Variational Autoencoder
 - ▶ Assume observations can be described by some binary (or categorical) code
 - ▶ We want to learn both encoder and decoder for such code and observations
- ▶ Allow the model to make a discrete choice
 - ▶ **Example:** Hard Attention
 - ▶ An attention module generates binary mask of where to look at
 - ▶ The network classifies masked images
 - ▶ We want attention module to attend only important areas of the image
- ▶ Sometimes you need discrete predictions to have certain properties
 - ▶ **Example:** GANs for text
 - ▶ Generator outputs discrete text
 - ▶ Discriminator takes discrete text as input and classifies how real it is
 - ▶ We want the generator to output text that fools the discriminator

- ▶ Easier to interpret discrete categories than continuous spectrum
 - ▶ **Example:** Discrete Variational Autoencoder
 - ▶ Assume observations can be described by some binary (or categorical) code
 - ▶ We want to learn both encoder and decoder for such code and observations
- ▶ Allow the model to make a discrete choice
 - ▶ **Example:** Hard Attention
 - ▶ An attention module generates binary mask of where to look at
 - ▶ The network classifies masked images
 - ▶ We want attention module to attend only important areas of the image
- ▶ Sometimes you need discrete predictions to have certain properties
 - ▶ **Example:** GANs for text
 - ▶ Generator outputs discrete text
 - ▶ Discriminator takes discrete text as input and classifies how real it is
 - ▶ We want the generator to output text that fools the discriminator

- ▶ Easier to interpret discrete categories than continuous spectrum
 - ▶ **Example:** Discrete Variational Autoencoder
 - ▶ Assume observations can be described by some binary (or categorical) code
 - ▶ We want to learn both encoder and decoder for such code and observations
- ▶ Allow the model to make a discrete choice
 - ▶ **Example:** Hard Attention
 - ▶ An attention module generates binary mask of where to look at
 - ▶ The network classifies masked images
 - ▶ We want attention module to attend only important areas of the image
- ▶ Sometimes you need discrete predictions to have certain properties
 - ▶ **Example:** GANs for text
 - ▶ Generator outputs discrete text
 - ▶ Discriminator takes discrete text as input and classifies how real it is
 - ▶ We want the generator to output text that fools the discriminator

- ▶ Easier to interpret discrete categories than continuous spectrum
 - ▶ **Example:** Discrete Variational Autoencoder
 - ▶ Assume observations can be described by some binary (or categorical) code
 - ▶ We want to learn both encoder and decoder for such code and observations
- ▶ Allow the model to make a discrete choice
 - ▶ **Example:** Hard Attention
 - ▶ An attention module generates binary mask of where to look at
 - ▶ The network classifies masked images
 - ▶ We want attention module to attend only important areas of the image
- ▶ Sometimes you need discrete predictions to have certain properties
 - ▶ **Example:** GANs for text
 - ▶ Generator outputs discrete text
 - ▶ Discriminator takes discrete text as input and classifies how real it is
 - ▶ We want the generator to output text that fools the discriminator

- ▶ Easier to interpret discrete categories than continuous spectrum
 - ▶ **Example:** Discrete Variational Autoencoder
 - ▶ Assume observations can be described by some binary (or categorical) code
 - ▶ We want to learn both encoder and decoder for such code and observations
- ▶ Allow the model to make a discrete choice
 - ▶ **Example:** Hard Attention
 - ▶ An attention module generates binary mask of where to look at
 - ▶ The network classifies masked images
 - ▶ We want attention module to attend only important areas of the image
- ▶ Sometimes you need discrete predictions to have certain properties
 - ▶ **Example:** GANs for text
 - ▶ Generator outputs discrete text
 - ▶ Discriminator takes discrete text as input and classifies how real it is
 - ▶ We want the generator to output text that fools the discriminator

The Problem

Typically problems boil down to optimizing an objective of the following form

$$\mathcal{L}(\phi) = \mathbb{E}_{q_{\phi}(\mathbf{z})} f(\mathbf{z}) \rightarrow \max_{\phi}$$

- ▶ We consider models where the objective is differentiable w.r.t. ϕ
 - ▶ Hence the gradient $\frac{\partial}{\partial \phi} \mathcal{L}(\phi)$ exists
- ▶ The expectation is expensive to compute due to large number of summands, turn to *Stochastic Optimization*
- ▶ Stochastic Optimization requires a stochastic (unbiased) estimate $g(\mathbf{z}, \phi)$ of the true gradient:

$$\mathbb{E}_{q_{\phi}(\mathbf{z})} g(\mathbf{z}, \phi) = \frac{\partial}{\partial \phi} \mathcal{L}(\phi)$$

- ▶ No continuous reparametrization is possible for \mathbf{z}
 - ▶ Because \mathbf{z} takes finitely many different values

Typically problems boil down to optimizing an objective of the following form

$$\mathcal{L}(\phi) = \mathbb{E}_{q_{\phi}(\mathbf{z})} f(\mathbf{z}) \rightarrow \max_{\phi}$$

- ▶ We consider models where the objective is differentiable w.r.t. ϕ
 - ▶ Hence the gradient $\frac{\partial}{\partial \phi} \mathcal{L}(\phi)$ exists
- ▶ The expectation is expensive to compute due to large number of summands, turn to *Stochastic Optimization*
- ▶ Stochastic Optimization requires a stochastic (unbiased) estimate $g(\mathbf{z}, \phi)$ of the true gradient:

$$\mathbb{E}_{q_{\phi}(\mathbf{z})} g(\mathbf{z}, \phi) = \frac{\partial}{\partial \phi} \mathcal{L}(\phi)$$

- ▶ No continuous reparametrization is possible for \mathbf{z}
 - ▶ Because \mathbf{z} takes finitely many different values

Typically problems boil down to optimizing an objective of the following form

$$\mathcal{L}(\phi) = \mathbb{E}_{q_{\phi}(\mathbf{z})} f(\mathbf{z}) \rightarrow \max_{\phi}$$

- ▶ We consider models where the objective is differentiable w.r.t. ϕ
 - ▶ Hence the gradient $\frac{\partial}{\partial \phi} \mathcal{L}(\phi)$ exists
- ▶ The expectation is expensive to compute due to large number of summands, turn to *Stochastic Optimization*
- ▶ Stochastic Optimization requires a stochastic (unbiased) estimate $g(\mathbf{z}, \phi)$ of the true gradient:

$$\mathbb{E}_{q_{\phi}(\mathbf{z})} g(\mathbf{z}, \phi) = \frac{\partial}{\partial \phi} \mathcal{L}(\phi)$$

- ▶ No continuous reparametrization is possible for \mathbf{z}
 - ▶ Because \mathbf{z} takes finitely many different values

Typically problems boil down to optimizing an objective of the following form

$$\mathcal{L}(\phi) = \mathbb{E}_{q_{\phi}(\mathbf{z})} f(\mathbf{z}) \rightarrow \max_{\phi}$$

- ▶ We consider models where the objective is differentiable w.r.t. ϕ
 - ▶ Hence the gradient $\frac{\partial}{\partial \phi} \mathcal{L}(\phi)$ exists
- ▶ The expectation is expensive to compute due to large number of summands, turn to *Stochastic Optimization*
- ▶ Stochastic Optimization requires a stochastic (unbiased) estimate $g(\mathbf{z}, \phi)$ of the true gradient:

$$\mathbb{E}_{q_{\phi}(\mathbf{z})} g(\mathbf{z}, \phi) = \frac{\partial}{\partial \phi} \mathcal{L}(\phi)$$

- ▶ No continuous reparametrization is possible for \mathbf{z}
 - ▶ Because \mathbf{z} takes finitely many different values

Typically problems boil down to optimizing an objective of the following form

$$\mathcal{L}(\phi) = \mathbb{E}_{q_{\phi}(\mathbf{z})} f(\mathbf{z}) \rightarrow \max_{\phi}$$

- ▶ We consider models where the objective is differentiable w.r.t. ϕ
 - ▶ Hence the gradient $\frac{\partial}{\partial \phi} \mathcal{L}(\phi)$ exists
- ▶ The expectation is expensive to compute due to large number of summands, turn to *Stochastic Optimization*
- ▶ Stochastic Optimization requires a stochastic (unbiased) estimate $g(\mathbf{z}, \phi)$ of the true gradient:

$$\mathbb{E}_{q_{\phi}(\mathbf{z})} g(\mathbf{z}, \phi) = \frac{\partial}{\partial \phi} \mathcal{L}(\phi)$$

- ▶ No continuous reparametrization is possible for \mathbf{z}
 - ▶ Because \mathbf{z} takes finitely many different values

$$\frac{\partial}{\partial \phi} \mathbb{E}_{q_{\phi}(\mathbf{z})} f(\mathbf{z}) = \frac{\partial}{\partial \phi} \sum_{\mathbf{z}} q_{\phi}(\mathbf{z}) f(\mathbf{z}) = \sum_{\mathbf{z}} \frac{\partial}{\partial \phi} q_{\phi}(\mathbf{z}) f(\mathbf{z})$$

Now note that $q_{\phi}(\mathbf{z}) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}) = \frac{\partial}{\partial \phi} q_{\phi}(\mathbf{z})$ (log-derivative trick)

$$= \sum_{\mathbf{z}} q_{\phi}(\mathbf{z}) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}) f(\mathbf{z}) = \mathbb{E}_{q_{\phi}(\mathbf{z})} \left[\frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}) f(\mathbf{z}) \right]$$

In practice this expectation is intractable, thus we resort to Monte Carlo estimation:

$$g(\mathbf{z}_{1:M}, \phi) := \frac{1}{M} \sum_{m=1}^M f(\mathbf{z}_m) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}_m), \quad \mathbf{z}_m \sim q_{\phi}(\mathbf{z})$$

This g is called the **REINFORCE** (aka log-derivative trick or score-function) estimator.

$$\frac{\partial}{\partial \phi} \mathbb{E}_{q_{\phi}(\mathbf{z})} f(\mathbf{z}) = \frac{\partial}{\partial \phi} \sum_{\mathbf{z}} q_{\phi}(\mathbf{z}) f(\mathbf{z}) = \sum_{\mathbf{z}} \frac{\partial}{\partial \phi} q_{\phi}(\mathbf{z}) f(\mathbf{z})$$

Now note that $q_{\phi}(\mathbf{z}) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}) = \frac{\partial}{\partial \phi} q_{\phi}(\mathbf{z})$ (log-derivative trick)

$$= \sum_{\mathbf{z}} q_{\phi}(\mathbf{z}) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}) f(\mathbf{z}) = \mathbb{E}_{q_{\phi}(\mathbf{z})} \left[\frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}) f(\mathbf{z}) \right]$$

In practice this expectation is intractable, thus we resort to Monte Carlo estimation:

$$g(\mathbf{z}_{1:M}, \phi) := \frac{1}{M} \sum_{m=1}^M f(\mathbf{z}_m) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}_m), \quad \mathbf{z}_m \sim q_{\phi}(\mathbf{z})$$

This g is called the **REINFORCE** (aka log-derivative trick or score-function) estimator.

$$\frac{\partial}{\partial \phi} \mathbb{E}_{q_{\phi}(\mathbf{z})} f(\mathbf{z}) = \frac{\partial}{\partial \phi} \sum_{\mathbf{z}} q_{\phi}(\mathbf{z}) f(\mathbf{z}) = \sum_{\mathbf{z}} \frac{\partial}{\partial \phi} q_{\phi}(\mathbf{z}) f(\mathbf{z})$$

Now note that $q_{\phi}(\mathbf{z}) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}) = \frac{\partial}{\partial \phi} q_{\phi}(\mathbf{z})$ (log-derivative trick)

$$= \sum_{\mathbf{z}} q_{\phi}(\mathbf{z}) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}) f(\mathbf{z}) = \mathbb{E}_{q_{\phi}(\mathbf{z})} \left[\frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}) f(\mathbf{z}) \right]$$

In practice this expectation is intractable, thus we resort to Monte Carlo estimation:

$$g(\mathbf{z}_{1:M}, \phi) := \frac{1}{M} \sum_{m=1}^M f(\mathbf{z}_m) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}_m), \quad \mathbf{z}_m \sim q_{\phi}(\mathbf{z})$$

This g is called the **REINFORCE** (aka log-derivative trick or score-function) estimator.

$$\frac{\partial}{\partial \phi} \mathbb{E}_{q_{\phi}(\mathbf{z})} f(\mathbf{z}) = \frac{\partial}{\partial \phi} \sum_{\mathbf{z}} q_{\phi}(\mathbf{z}) f(\mathbf{z}) = \sum_{\mathbf{z}} \frac{\partial}{\partial \phi} q_{\phi}(\mathbf{z}) f(\mathbf{z})$$

Now note that $q_{\phi}(\mathbf{z}) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}) = \frac{\partial}{\partial \phi} q_{\phi}(\mathbf{z})$ (log-derivative trick)

$$= \sum_{\mathbf{z}} q_{\phi}(\mathbf{z}) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}) f(\mathbf{z}) = \mathbb{E}_{q_{\phi}(\mathbf{z})} \left[\frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}) f(\mathbf{z}) \right]$$

In practice this expectation is intractable, thus we resort to Monte Carlo estimation:

$$g(\mathbf{z}_{1:M}, \phi) := \frac{1}{M} \sum_{m=1}^M f(\mathbf{z}_m) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}_m), \quad \mathbf{z}_m \sim q_{\phi}(\mathbf{z})$$

This g is called the **REINFORCE** (aka log-derivative trick or score-function) estimator.

$$\frac{\partial}{\partial \phi} \mathbb{E}_{q_{\phi}(\mathbf{z})} f(\mathbf{z}) = \frac{\partial}{\partial \phi} \sum_{\mathbf{z}} q_{\phi}(\mathbf{z}) f(\mathbf{z}) = \sum_{\mathbf{z}} \frac{\partial}{\partial \phi} q_{\phi}(\mathbf{z}) f(\mathbf{z})$$

Now note that $q_{\phi}(\mathbf{z}) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}) = \frac{\partial}{\partial \phi} q_{\phi}(\mathbf{z})$ (log-derivative trick)

$$= \sum_{\mathbf{z}} q_{\phi}(\mathbf{z}) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}) f(\mathbf{z}) = \mathbb{E}_{q_{\phi}(\mathbf{z})} \left[\frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}) f(\mathbf{z}) \right]$$

In practice this expectation is intractable, thus we resort to Monte Carlo estimation:

$$g(\mathbf{z}_{1:M}, \phi) := \frac{1}{M} \sum_{m=1}^M f(\mathbf{z}_m) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}_m), \quad \mathbf{z}_m \sim q_{\phi}(\mathbf{z})$$

This g is called the **REINFORCE** (aka log-derivative trick or score-function) estimator.

$$\frac{\partial}{\partial \phi} \mathbb{E}_{q_{\phi}(\mathbf{z})} f(\mathbf{z}) = \frac{\partial}{\partial \phi} \sum_{\mathbf{z}} q_{\phi}(\mathbf{z}) f(\mathbf{z}) = \sum_{\mathbf{z}} \frac{\partial}{\partial \phi} q_{\phi}(\mathbf{z}) f(\mathbf{z})$$

Now note that $q_{\phi}(\mathbf{z}) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}) = \frac{\partial}{\partial \phi} q_{\phi}(\mathbf{z})$ (log-derivative trick)

$$= \sum_{\mathbf{z}} q_{\phi}(\mathbf{z}) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}) f(\mathbf{z}) = \mathbb{E}_{q_{\phi}(\mathbf{z})} \left[\frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}) f(\mathbf{z}) \right]$$

In practice this expectation is intractable, thus we resort to Monte Carlo estimation:

$$g(\mathbf{z}_{1:M}, \phi) := \frac{1}{M} \sum_{m=1}^M f(\mathbf{z}_m) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}_m), \quad \mathbf{z}_m \sim q_{\phi}(\mathbf{z})$$

This g is called the **REINFORCE** (aka log-derivative trick or score-function) estimator.

$$\frac{\partial}{\partial \phi} \mathbb{E}_{q_{\phi}(\mathbf{z})} f(\mathbf{z}) = \frac{\partial}{\partial \phi} \sum_{\mathbf{z}} q_{\phi}(\mathbf{z}) f(\mathbf{z}) = \sum_{\mathbf{z}} \frac{\partial}{\partial \phi} q_{\phi}(\mathbf{z}) f(\mathbf{z})$$

Now note that $q_{\phi}(\mathbf{z}) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}) = \frac{\partial}{\partial \phi} q_{\phi}(\mathbf{z})$ (log-derivative trick)

$$= \sum_{\mathbf{z}} q_{\phi}(\mathbf{z}) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}) f(\mathbf{z}) = \mathbb{E}_{q_{\phi}(\mathbf{z})} \left[\frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}) f(\mathbf{z}) \right]$$

In practice this expectation is intractable, thus we resort to Monte Carlo estimation:

$$g(\mathbf{z}_{1:M}, \phi) := \frac{1}{M} \sum_{m=1}^M f(\mathbf{z}_m) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}_m), \quad \mathbf{z}_m \sim q_{\phi}(\mathbf{z})$$

This g is called the **REINFORCE** (aka log-derivative trick or score-function) estimator.

$$\frac{\partial}{\partial \phi} \mathbb{E}_{q_{\phi}(\mathbf{z})} f(\mathbf{z}) = \frac{\partial}{\partial \phi} \sum_{\mathbf{z}} q_{\phi}(\mathbf{z}) f(\mathbf{z}) = \sum_{\mathbf{z}} \frac{\partial}{\partial \phi} q_{\phi}(\mathbf{z}) f(\mathbf{z})$$

Now note that $q_{\phi}(\mathbf{z}) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}) = \frac{\partial}{\partial \phi} q_{\phi}(\mathbf{z})$ (log-derivative trick)

$$= \sum_{\mathbf{z}} q_{\phi}(\mathbf{z}) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}) f(\mathbf{z}) = \mathbb{E}_{q_{\phi}(\mathbf{z})} \left[\frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}) f(\mathbf{z}) \right]$$

In practice this expectation is intractable, thus we resort to Monte Carlo estimation:

$$g(\mathbf{z}_{1:M}, \phi) := \frac{1}{M} \sum_{m=1}^M f(\mathbf{z}_m) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}_m), \quad \mathbf{z}_m \sim q_{\phi}(\mathbf{z})$$

This g is called the **REINFORCE** (aka log-derivative trick or score-function) estimator.

$$\frac{1}{M} \sum_{m=1}^M f(\mathbf{z}_m) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}_m), \quad \mathbf{z}_m \sim q_{\phi}(\mathbf{z})$$

- Works for our case, discreteness does not get in the way
 - f is not even required to be continuous
- Typically has large variance
- Requires sophisticated *Variance Reduction* methods
 - Just taking bigger M gives only a modest improvement

$$\text{Var} \left[\frac{1}{M} \sum_{m=1}^M f(\mathbf{z}_m) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}_m) \right] = \frac{1}{M} \text{Var} \left[f(\mathbf{z}) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}) \right]$$

- Hence the "typical error" would decrease in proportion to $1/\sqrt{M}$.
- In practice a single sample ($M = 1$) is often used.

$$\frac{1}{M} \sum_{m=1}^M f(\mathbf{z}_m) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}_m), \quad \mathbf{z}_m \sim q_{\phi}(\mathbf{z})$$

- ▶ Works for our case, discreteness does not get in the way
 - ▶ f is not even required to be continuous
- ▶ Typically has large variance
- ▶ Requires sophisticated *Variance Reduction* methods
 - ▶ Just taking bigger M gives only a modest improvement

$$\text{Var} \left[\frac{1}{M} \sum_{m=1}^M f(\mathbf{z}_m) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}_m) \right] = \frac{1}{M} \text{Var} \left[f(\mathbf{z}) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}) \right]$$

- ▶ Hence the "typical error" would decrease in proportion to $1/\sqrt{M}$.
- ▶ In practice a single sample ($M = 1$) is often used.

$$\frac{1}{M} \sum_{m=1}^M f(\mathbf{z}_m) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}_m), \quad \mathbf{z}_m \sim q_{\phi}(\mathbf{z})$$

- ▶ Works for our case, discreteness does not get in the way
 - ▶ f is not even required to be continuous
- ▶ Typically has large variance
- ▶ Requires sophisticated *Variance Reduction* methods
 - ▶ Just taking bigger M gives only a modest improvement

$$\text{Var} \left[\frac{1}{M} \sum_{m=1}^M f(\mathbf{z}_m) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}_m) \right] = \frac{1}{M} \text{Var} \left[f(\mathbf{z}) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}) \right]$$

- ▶ Hence the "typical error" would decrease in proportion to $1/\sqrt{M}$.
- ▶ In practice a single sample ($M = 1$) is often used.

$$\frac{1}{M} \sum_{m=1}^M f(\mathbf{z}_m) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}_m), \quad \mathbf{z}_m \sim q_{\phi}(\mathbf{z})$$

- Works for our case, discreteness does not get in the way
 - f is not even required to be continuous
- Typically has large variance
- Requires sophisticated *Variance Reduction* methods
 - Just taking bigger M gives only a modest improvement

$$\text{Var} \left[\frac{1}{M} \sum_{m=1}^M f(\mathbf{z}_m) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}_m) \right] = \frac{1}{M} \text{Var} \left[f(\mathbf{z}) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}) \right]$$

- Hence the "typical error" would decrease in proportion to $1/\sqrt{M}$.
- In practice a single sample ($M = 1$) is often used.

Consider 1-sample estimate

$$g^{\text{REINFORCE}}(\mathbf{z}, \phi) = \underbrace{f(\mathbf{z})}_{\text{scalar}} \underbrace{\frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z})}_{\text{vector}}, \quad \mathbf{z} \sim q_{\phi}(\mathbf{z})$$

- ▶ Gradient estimate points in the direction of increasing probability of a given sample \mathbf{z}
 - ▶ Increases probability of \mathbf{z} if it happened to be good
- ▶ The target function f only enters as a scaling coefficient, and no gradient $\frac{\partial f}{\partial \mathbf{z}}$ is used (unlike the reparametrization trick)
 - ▶ Has no idea where to move probability mass *systematically*
 - ▶ $f(\mathbf{z}) + c$ will give a different estimator
- ▶ **Random search in disguise!** [Rec18]

Consider 1-sample estimate

$$g^{\text{REINFORCE}}(\mathbf{z}, \phi) = \underbrace{f(\mathbf{z})}_{\text{scalar}} \underbrace{\frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z})}_{\text{vector}}, \quad \mathbf{z} \sim q_{\phi}(\mathbf{z})$$

- ▶ Gradient estimate points in the direction of increasing probability of a given sample \mathbf{z}
 - ▶ Increases probability of \mathbf{z} if it happened to be good
- ▶ The target function f only enters as a scaling coefficient, and no gradient $\frac{\partial f}{\partial \mathbf{z}}$ is used (unlike the reparametrization trick)
 - ▶ Has no idea where to move probability mass *systematically*
 - ▶ $f(\mathbf{z}) + c$ will give a different estimator
- ▶ **Random search in disguise!** [Rec18]

Consider 1-sample estimate

$$g^{\text{REINFORCE}}(\mathbf{z}, \phi) = \underbrace{f(\mathbf{z})}_{\text{scalar}} \underbrace{\frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z})}_{\text{vector}}, \quad \mathbf{z} \sim q_{\phi}(\mathbf{z})$$

- ▶ Gradient estimate points in the direction of increasing probability of a given sample \mathbf{z}
 - ▶ Increases probability of \mathbf{z} if it happened to be good
- ▶ The target function f only enters as a scaling coefficient, and no gradient $\frac{\partial f}{\partial \mathbf{z}}$ is used (unlike the reparametrization trick)
 - ▶ Has no idea where to move probability mass *systematically*
 - ▶ $f(\mathbf{z}) + c$ will give a different estimator
- ▶ **Random search in disguise!** [Rec18]

Consider 1-sample estimate

$$g^{\text{REINFORCE}}(\mathbf{z}, \phi) = \underbrace{f(\mathbf{z})}_{\text{scalar}} \underbrace{\frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z})}_{\text{vector}}, \quad \mathbf{z} \sim q_{\phi}(\mathbf{z})$$

- ▶ Gradient estimate points in the direction of increasing probability of a given sample \mathbf{z}
 - ▶ Increases probability of \mathbf{z} if it happened to be good
- ▶ The target function f only enters as a scaling coefficient, and no gradient $\frac{\partial f}{\partial \mathbf{z}}$ is used (unlike the reparametrization trick)
 - ▶ Has no idea where to move probability mass *systematically*
 - ▶ $f(\mathbf{z}) + c$ will give a different estimator
- ▶ Random search in disguise! [Rec18]

Consider 1-sample estimate

$$g^{\text{REINFORCE}}(\mathbf{z}, \phi) = \underbrace{f(\mathbf{z})}_{\text{scalar}} \underbrace{\frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z})}_{\text{vector}}, \quad \mathbf{z} \sim q_{\phi}(\mathbf{z})$$

- ▶ Gradient estimate points in the direction of increasing probability of a given sample \mathbf{z}
 - ▶ Increases probability of \mathbf{z} if it happened to be good
- ▶ The target function f only enters as a scaling coefficient, and no gradient $\frac{\partial f}{\partial \mathbf{z}}$ is used (unlike the reparametrization trick)
 - ▶ Has no idea where to move probability mass *systematically*
 - ▶ $f(\mathbf{z}) + c$ will give a different estimator
- ▶ **Random search in disguise!** [Rec18]

Relaxations

Idea: Relax the objective over discrete random samples \mathbf{z} into an objective over continuous random samples $\tilde{\mathbf{z}}$ during training and use the reparametrization trick:

$$\mathbb{E}_{q_{\phi}(\mathbf{z})} f(\mathbf{z}) \approx \mathbb{E}_{q_{\phi}(\tilde{\mathbf{z}})} f(\tilde{\mathbf{z}}) = \mathbb{E}_{p(\gamma)} f(\tilde{\mathbf{z}}(\gamma, \phi))$$

Keep the discrete testing phase model

Limitation: $f(\mathbf{z})$ has to be differentiable w.r.t. its input.

Idea: Relax the objective over discrete random samples \mathbf{z} into an objective over continuous random samples $\tilde{\mathbf{z}}$ during training and use the reparametrization trick:

$$\mathbb{E}_{q_{\phi}(\mathbf{z})} f(\mathbf{z}) \approx \mathbb{E}_{q_{\phi}(\tilde{\mathbf{z}})} f(\tilde{\mathbf{z}}) = \mathbb{E}_{p(\gamma)} f(\tilde{\mathbf{z}}(\gamma, \phi))$$

Keep the discrete testing phase model

Limitation: $f(\mathbf{z})$ has to be differentiable w.r.t. its input.

Idea: Relax the objective over discrete random samples \mathbf{z} into an objective over continuous random samples $\tilde{\mathbf{z}}$ during training and use the reparametrization trick:

$$\mathbb{E}_{q_{\phi}(\mathbf{z})} f(\mathbf{z}) \approx \mathbb{E}_{q_{\phi}(\tilde{\mathbf{z}})} f(\tilde{\mathbf{z}}) = \mathbb{E}_{p(\gamma)} f(\tilde{\mathbf{z}}(\gamma, \phi))$$

Keep the discrete testing phase model

Limitation: $f(\mathbf{z})$ has to be differentiable w.r.t. its input.

Idea: Relax the objective over discrete random samples \mathbf{z} into an objective over continuous random samples $\tilde{\mathbf{z}}$ during training and use the reparametrization trick:

$$\mathbb{E}_{q_{\phi}(\mathbf{z})} f(\mathbf{z}) \approx \mathbb{E}_{q_{\phi}(\tilde{\mathbf{z}})} f(\tilde{\mathbf{z}}) = \mathbb{E}_{p(\gamma)} f(\tilde{\mathbf{z}}(\gamma, \phi))$$

Keep the discrete testing phase model

Limitation: $f(\mathbf{z})$ has to be differentiable w.r.t. its input.

An old trick to sample Categorical random variables

$$z \sim \text{Categorical}(\pi_1, \dots, \pi_K),$$

Minimum of independent exponential distributions with carefully chosen probabilities has the same distribution:

$$z \stackrel{d}{=} \underset{k}{\operatorname{argmin}} \frac{\xi_k}{\pi_k}, \quad \xi_k \sim \text{Exp}(1)$$

Equivalently (applying $-\log$)

$$z \stackrel{d}{=} \underset{k}{\operatorname{argmax}} \left[\log \pi_k - \overbrace{\log \xi_k}^{\gamma_k} \right], \quad \xi_k \sim \text{Exp}(1)$$

Converts sampling K -ary discrete random variable into optimization of noise-perturbed logits, $\gamma := -\log \xi$ has standard Gumbel(0, 1) distribution.

- Gives a reparametrization, but not a continuous one!

An old trick to sample Categorical random variables

$$z \sim \text{Categorical}(\pi_1, \dots, \pi_K),$$

Minimum of independent exponential distributions with carefully chosen probabilities has the same distribution:

$$z \stackrel{d}{=} \underset{k}{\operatorname{argmin}} \frac{\xi_k}{\pi_k}, \quad \xi_k \sim \text{Exp}(1)$$

Equivalently (applying $-\log$)

$$z \stackrel{d}{=} \underset{k}{\operatorname{argmax}} \left[\log \pi_k - \overbrace{\log \xi_k}^{\gamma_k} \right], \quad \xi_k \sim \text{Exp}(1)$$

Converts sampling K -ary discrete random variable into optimization of noise-perturbed logits, $\gamma := -\log \xi$ has standard Gumbel(0, 1) distribution.

- Gives a reparametrization, but not a continuous one!

An old trick to sample Categorical random variables

$$z \sim \text{Categorical}(\pi_1, \dots, \pi_K),$$

Minimum of independent exponential distributions with carefully chosen probabilities has the same distribution:

$$z \stackrel{d}{=} \underset{k}{\operatorname{argmin}} \frac{\xi_k}{\pi_k}, \quad \xi_k \sim \text{Exp}(1)$$

Equivalently (applying $-\log$)

$$z \stackrel{d}{=} \underset{k}{\operatorname{argmax}} \left[\log \pi_k - \overbrace{\log \xi_k}^{\gamma_k} \right], \quad \xi_k \sim \text{Exp}(1)$$

Converts sampling K -ary discrete random variable into optimization of noise-perturbed logits, $\gamma := -\log \xi$ has standard Gumbel(0, 1) distribution.

- Gives a reparametrization, but not a continuous one!

An old trick to sample Categorical random variables

$$z \sim \text{Categorical}(\pi_1, \dots, \pi_K),$$

Minimum of independent exponential distributions with carefully chosen probabilities has the same distribution:

$$z \stackrel{d}{=} \underset{k}{\operatorname{argmin}} \frac{\xi_k}{\pi_k}, \quad \xi_k \sim \text{Exp}(1)$$

Equivalently (applying $-\log$)

$$z \stackrel{d}{=} \underset{k}{\operatorname{argmax}} \left[\log \pi_k - \overbrace{\log \xi_k}^{\gamma_k} \right], \quad \xi_k \sim \text{Exp}(1)$$

Converts sampling K -ary discrete random variable into optimization of noise-perturbed logits, $\gamma := -\log \xi$ has standard $\text{Gumbel}(0, 1)$ distribution.

- Gives a reparametrization, but not a continuous one!

Approximate argmax with softmax (with temperature)

$$\text{softmax}_{\tau}(x)_j := \frac{\exp(x_j/\tau)}{\sum_{k=1}^K \exp(x_k/\tau)}$$

Temperature controls "sharpness" of the softmax:

- $\tau = 0$ recovers $\text{argmax} = \text{softmax}_0$
- $\tau = \infty$ leads to uniform distribution that ignores any disparities

Then assume the discrete \mathbf{z} is a one-hot vector and replace with a continuous relaxation $\tilde{\mathbf{z}}$

$$\tilde{\mathbf{z}}(\gamma, \pi) := \text{softmax}_{\tau}(\log \pi_1 + \gamma_1, \dots, \log \pi_K + \gamma_K)$$

Where each $\gamma_k \sim \text{Gumbel}(0, 1)$ is a standard Gumbel random variable, and can be generated from uniform noise u_k as

$$\gamma_k \stackrel{d}{=} -\log(-\log u_k), \quad u_k \sim \text{Uniform}(0, 1)$$

Approximate argmax with softmax (with temperature)

$$\text{softmax}_{\tau}(x)_j := \frac{\exp(x_j/\tau)}{\sum_{k=1}^K \exp(x_k/\tau)}$$

Temperature controls "sharpness" of the softmax:

- $\tau = 0$ recovers $\text{argmax} = \text{softmax}_0$
- $\tau = \infty$ leads to uniform distribution that ignores any disparities

Then assume the discrete \mathbf{z} is a one-hot vector and replace with a continuous relaxation $\tilde{\mathbf{z}}$

$$\tilde{\mathbf{z}}(\gamma, \pi) := \text{softmax}_{\tau}(\log \pi_1 + \gamma_1, \dots, \log \pi_K + \gamma_K)$$

Where each $\gamma_k \sim \text{Gumbel}(0, 1)$ is a standard Gumbel random variable, and can be generated from uniform noise u_k as

$$\gamma_k \stackrel{d}{=} -\log(-\log u_k), \quad u_k \sim \text{Uniform}(0, 1)$$

Approximate argmax with softmax (with temperature)

$$\text{softmax}_{\tau}(x)_j := \frac{\exp(x_j/\tau)}{\sum_{k=1}^K \exp(x_k/\tau)}$$

Temperature controls "sharpness" of the softmax:

- $\tau = 0$ recovers $\text{argmax} = \text{softmax}_0$
- $\tau = \infty$ leads to uniform distribution that ignores any disparities

Then assume the discrete \mathbf{z} is a one-hot vector and replace with a continuous relaxation $\tilde{\mathbf{z}}$

$$\tilde{\mathbf{z}}(\gamma, \pi) := \text{softmax}_{\tau}(\log \pi_1 + \gamma_1, \dots, \log \pi_K + \gamma_K)$$

Where each $\gamma_k \sim \text{Gumbel}(0, 1)$ is a standard Gumbel random variable, and can be generated from uniform noise u_k as

$$\gamma_k \stackrel{d}{=} -\log(-\log u_k), \quad u_k \sim \text{Uniform}(0, 1)$$

Now we can rewrite the expectation w.r.t. independent noise $\gamma_1, \dots, \gamma_K$

$$\mathcal{L}(\phi) = \mathbb{E}_{\gamma} f(\tilde{\mathbf{z}}(\gamma, \phi)), \quad \gamma_k \sim \text{Gumbel}(0, 1)$$

Gradient estimate is obtained simply by exchanging $\frac{\partial}{\partial \phi}$ and \mathbb{E} :

$$g^{\text{Rep}}(\gamma, \phi) = \frac{\partial}{\partial \phi} f(\tilde{\mathbf{z}}(\gamma, \pi(\phi)))$$

Similar to stochastic discrete nodes replaced by their expectation (softmax), but has **noise injected into log-probabilities**

- ▶ Noise helps exploration and regularizes
- ▶ Right kind of noise makes $\tilde{\mathbf{z}}$ similar to one-hot vectors
 - ▶ Reducing the train-test mismatch

Now we can rewrite the expectation w.r.t. independent noise $\gamma_1, \dots, \gamma_K$

$$\mathcal{L}(\phi) = \mathbb{E}_{\gamma} f(\tilde{\mathbf{z}}(\gamma, \phi)), \quad \gamma_k \sim \text{Gumbel}(0, 1)$$

Gradient estimate is obtained simply by exchanging $\frac{\partial}{\partial \phi}$ and \mathbb{E} :

$$g^{\text{Rep}}(\gamma, \phi) = \frac{\partial}{\partial \phi} f(\tilde{\mathbf{z}}(\gamma, \pi(\phi)))$$

Similar to stochastic discrete nodes replaced by their expectation (softmax), but has **noise injected into log-probabilities**

- ▶ Noise helps exploration and regularizes
- ▶ Right kind of noise makes $\tilde{\mathbf{z}}$ similar to one-hot vectors
 - ▶ Reducing the train-test mismatch

Now we can rewrite the expectation w.r.t. independent noise $\gamma_1, \dots, \gamma_K$

$$\mathcal{L}(\phi) = \mathbb{E}_{\gamma} f(\tilde{\mathbf{z}}(\gamma, \phi)), \quad \gamma_k \sim \text{Gumbel}(0, 1)$$

Gradient estimate is obtained simply by exchanging $\frac{\partial}{\partial \phi}$ and \mathbb{E} :

$$g^{\text{Rep}}(\gamma, \phi) = \frac{\partial}{\partial \phi} f(\tilde{\mathbf{z}}(\gamma, \pi(\phi)))$$

Similar to stochastic discrete nodes replaced by their expectation (softmax), but has **noise injected into log-probabilities**

- ▶ Noise helps exploration and regularizes
- ▶ Right kind of noise makes $\tilde{\mathbf{z}}$ similar to one-hot vectors
 - ▶ Reducing the train-test mismatch

In a special case of $K = 2$ we can write a bit more sample-efficient scheme:

$$\tilde{z} = \sigma_{\tau} \left(\log \frac{p}{1-p} + v \right), \quad v \sim \text{Logistic}(0, 1)$$

where $\text{Logistic}(0, 1)$ is the distribution of difference of two Gumbels:

$$v \stackrel{d}{=} \gamma_1 - \gamma_2$$

You can easily see this by yourself by simplifying the formula of the softmax over 2 classes.

In a special case of $K = 2$ we can write a bit more sample-efficient scheme:

$$\tilde{z} = \sigma_{\tau} \left(\log \frac{p}{1-p} + v \right), \quad v \sim \text{Logistic}(0, 1)$$

where $\text{Logistic}(0, 1)$ is the distribution of difference of two Gumbels:

$$v \stackrel{d}{=} \gamma_1 - \gamma_2$$

You can easily see this by yourself by simplifying the formula of the softmax over 2 classes.

In a special case of $K = 2$ we can write a bit more sample-efficient scheme:

$$\tilde{z} = \sigma_{\tau} \left(\log \frac{p}{1-p} + v \right), \quad v \sim \text{Logistic}(0, 1)$$

where $\text{Logistic}(0, 1)$ is the distribution of difference of two Gumbels:

$$v \stackrel{d}{=} \gamma_1 - \gamma_2$$

You can easily see this by yourself by simplifying the formula of the softmax over 2 classes.

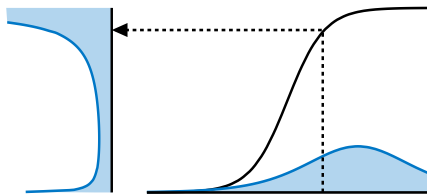
In a special case of $K = 2$ we can write a bit more sample-efficient scheme:

$$\tilde{z} = \sigma_{\tau} \left(\log \frac{p}{1-p} + v \right), \quad v \sim \text{Logistic}(0, 1)$$

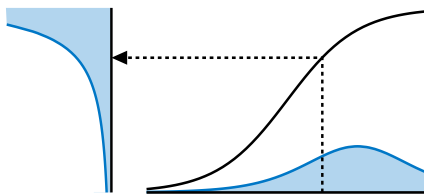
where $\text{Logistic}(0, 1)$ is the distribution of difference of two Gumbels:

$$v \stackrel{d}{=} \gamma_1 - \gamma_2$$

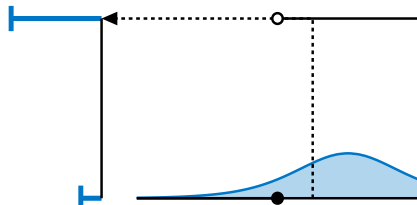
You can easily see this by yourself by simplifying the formula of the softmax over 2 classes.



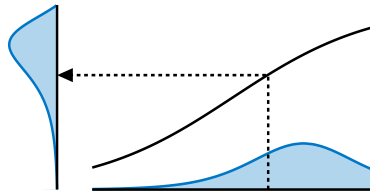
$\tau = 1/2$



$\tau = 1$

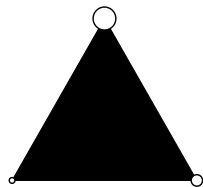


$\tau = 0$

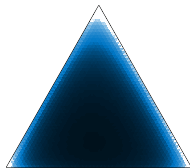


$\tau = 2$

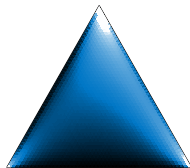
For K -ary categorical r.v. it has been shown that for $\tau \leq \frac{1}{K-1}$ there are no modes in the interior of the probability simplex



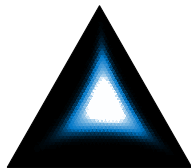
$\tau = 0$



$\tau = 1/2$



$\tau = 1$

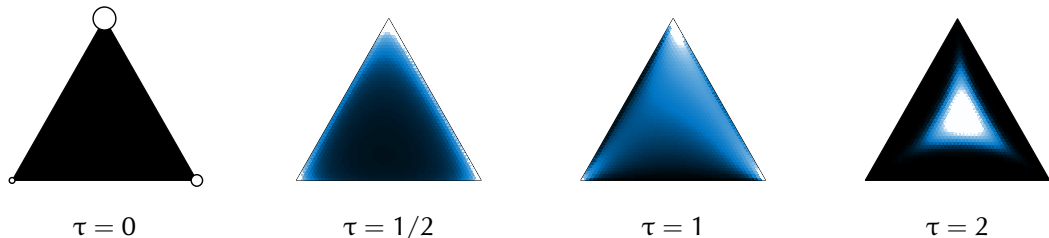


$\tau = 2$

How to choose a specific temperature?

- ▶ Small temperature leads to high variances, but resembles discrete case well
- ▶ Large temperatures have lower variance, but deviates away from the discrete case
- ▶ In practice grid search over a couple possible values

For K -ary categorical r.v. it has been shown that for $\tau \leq \frac{1}{K-1}$ there are no modes in the interior of the probability simplex



How to choose a specific temperature?

- ▶ Small temperature leads to high variances, but resembles discrete case well
- ▶ Large temperatures have lower variance, but deviates away from the discrete case
- ▶ In practice grid search over a couple possible values

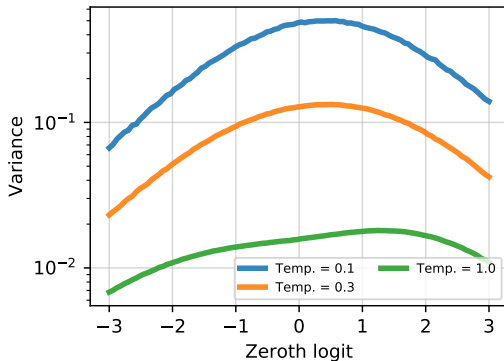
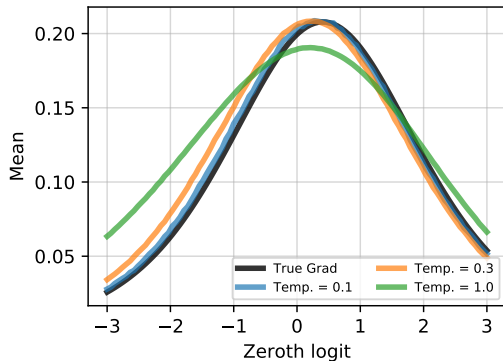
Consider 4-way categorical r.v. $z \sim q(z) = \text{Categorical}(z|\pi_1, \pi_2, \pi_3, \pi_4)$ where $\pi_j = \text{softmax}(\theta)_j$ and $\theta \in \mathbb{R}^4$ is a parameter vector. We seek to estimate the gradient of the following objective:

$$\mathbb{E}_{q(z)} \cos(z) \rightarrow \min_{\theta}$$

It's relaxed version is

$$\mathbb{E}_{p(\gamma_{1:4})} \cos(\text{softmax}_{\tau}(\theta + \gamma)^T[0, 1, 2, 3]) \rightarrow \min_{\theta}$$

Now we can use the reparametrization trick to estimate the gradient w.r.t. θ



- ▶ Gumbel-Softmax relaxes discrete random variables into continuous, enabling the reparametrization trick
- ▶ Relaxations change the objective, yet no theory on how good the relaxation is
 - ▶ Relaxation introduces bias
- ▶ Temperature τ is a hyperparameter that needs to be tuned

There exist other relaxations, however they are

- ▶ Less mathematically elegant
- ▶ Do not seem to work better empirically
- ▶ Sometimes heuristic in nature

- ▶ Gumbel-Softmax relaxes discrete random variables into continuous, enabling the reparametrization trick
- ▶ Relaxations change the objective, yet no theory on how good the relaxation is
 - ▶ Relaxation introduces bias
- ▶ Temperature τ is a hyperparameter that needs to be tuned

There exist other relaxations, however they are

- ▶ Less mathematically elegant
- ▶ Do not seem to work better empirically
- ▶ Sometimes heuristic in nature

- ▶ Gumbel-Softmax relaxes discrete random variables into continuous, enabling the reparametrization trick
- ▶ Relaxations change the objective, yet no theory on how good the relaxation is
 - ▶ Relaxation introduces bias
- ▶ Temperature τ is a hyperparameter that needs to be tuned

There exist other relaxations, however they are

- ▶ Less mathematically elegant
- ▶ Do not seem to work better empirically
- ▶ Sometimes heuristic in nature

- ▶ Gumbel-Softmax relaxes discrete random variables into continuous, enabling the reparametrization trick
- ▶ Relaxations change the objective, yet no theory on how good the relaxation is
 - ▶ Relaxation introduces bias
- ▶ Temperature τ is a hyperparameter that needs to be tuned

There exist other relaxations, however they are

- ▶ Less mathematically elegant
- ▶ Do not seem to work better empirically
- ▶ Sometimes heuristic in nature

- ▶ Gumbel-Softmax relaxes discrete random variables into continuous, enabling the reparametrization trick
- ▶ Relaxations change the objective, yet no theory on how good the relaxation is
 - ▶ Relaxation introduces bias
- ▶ Temperature τ is a hyperparameter that needs to be tuned

There exist other relaxations, however they are

- ▶ Less mathematically elegant
- ▶ Do not seem to work better empirically
- ▶ Sometimes heuristic in nature

Variance Reduction

Consider some $b(\mathbf{z})$ with **tractable** expectation $\mu := \mathbb{E}_{q(\mathbf{z})} b(\mathbf{z})$. Then

$$\mathbb{E}_{q(\mathbf{z})} f(\mathbf{z}) = \mathbb{E}_{q(\mathbf{z})} [(f(\mathbf{z}) - b(\mathbf{z})) + \mu]$$

■ might be a lower-variance estimate if $f(\mathbf{z})$ and $b(\mathbf{z})$ are positively correlated:

$$\text{Var} [\text{■}] = \text{Var} [f(\mathbf{z}) - b(\mathbf{z})] = \text{Var} [f(\mathbf{z})] + \text{Var} [b(\mathbf{z})] - 2\text{Cov} [f(\mathbf{z}), b(\mathbf{z})]$$

- ▶ Unbiased estimator
- ▶ $b(\mathbf{z})$ is called *Control Variate*
- ▶ Convenient if $b(\mathbf{z})$ is zero-mean
- ▶ We can choose any $b(\mathbf{z})$ we want
- ▶ Can take several samples M to reduce the variance further

Intuitively, we extract some tractable part $b(\mathbf{z})$ of the $f(\mathbf{z})$ and estimate the rest with Monte Carlo.

Consider some $b(\mathbf{z})$ with **tractable** expectation $\mu := \mathbb{E}_{q(\mathbf{z})} b(\mathbf{z})$. Then

$$\mathbb{E}_{q(\mathbf{z})} f(\mathbf{z}) = \mathbb{E}_{q(\mathbf{z})} [(f(\mathbf{z}) - b(\mathbf{z})) + \mu]$$

■ might be a lower-variance estimate if $f(\mathbf{z})$ and $b(\mathbf{z})$ are positively correlated:

$$\text{Var} [\text{■}] = \text{Var} [f(\mathbf{z}) - b(\mathbf{z})] = \text{Var} [f(\mathbf{z})] + \text{Var} [b(\mathbf{z})] - 2\text{Cov} [f(\mathbf{z}), b(\mathbf{z})]$$

- ▶ Unbiased estimator
- ▶ $b(\mathbf{z})$ is called *Control Variate*
- ▶ Convenient if $b(\mathbf{z})$ is zero-mean
- ▶ We can choose any $b(\mathbf{z})$ we want
- ▶ Can take several samples M to reduce the variance further

Intuitively, we extract some tractable part $b(\mathbf{z})$ of the $f(\mathbf{z})$ and estimate the rest with Monte Carlo.

Consider some $b(\mathbf{z})$ with **tractable** expectation $\mu := \mathbb{E}_{q(\mathbf{z})} b(\mathbf{z})$. Then

$$\mathbb{E}_{q(\mathbf{z})} f(\mathbf{z}) = \mathbb{E}_{q(\mathbf{z})} [(f(\mathbf{z}) - b(\mathbf{z})) + \mu]$$

■ might be a lower-variance estimate if $f(\mathbf{z})$ and $b(\mathbf{z})$ are positively correlated:

$$\text{Var} [\text{■}] = \text{Var} [f(\mathbf{z}) - b(\mathbf{z})] = \text{Var} [f(\mathbf{z})] + \text{Var} [b(\mathbf{z})] - 2\text{Cov} [f(\mathbf{z}), b(\mathbf{z})]$$

- ▶ Unbiased estimator
- ▶ $b(\mathbf{z})$ is called *Control Variate*
- ▶ Convenient if $b(\mathbf{z})$ is zero-mean
- ▶ We can choose any $b(\mathbf{z})$ we want
- ▶ Can take several samples M to reduce the variance further

Intuitively, we extract some tractable part $b(\mathbf{z})$ of the $f(\mathbf{z})$ and estimate the rest with Monte Carlo.

Consider some $b(\mathbf{z})$ with **tractable** expectation $\mu := \mathbb{E}_{q(\mathbf{z})} b(\mathbf{z})$. Then

$$\mathbb{E}_{q(\mathbf{z})} f(\mathbf{z}) = \mathbb{E}_{q(\mathbf{z})} [(f(\mathbf{z}) - b(\mathbf{z})) + \mu]$$

■ might be a lower-variance estimate if $f(\mathbf{z})$ and $b(\mathbf{z})$ are positively correlated:

$$\text{Var} [\text{■}] = \text{Var} [f(\mathbf{z}) - b(\mathbf{z})] = \text{Var} [f(\mathbf{z})] + \text{Var} [b(\mathbf{z})] - 2\text{Cov} [f(\mathbf{z}), b(\mathbf{z})]$$

- ▶ Unbiased estimator
- ▶ $b(\mathbf{z})$ is called *Control Variate*
- ▶ Convenient if $b(\mathbf{z})$ is zero-mean
- ▶ We can choose any $b(\mathbf{z})$ we want
- ▶ Can take several samples M to reduce the variance further

Intuitively, we extract some tractable part $b(\mathbf{z})$ of the $f(\mathbf{z})$ and estimate the rest with Monte Carlo.

$$\mathbb{E}_{q_{\phi}(\mathbf{z}_{1:M})} \left[\frac{1}{M} \sum_{m=1}^M (f(\mathbf{z}_m) - b(\mathbf{z}_m)) \right] + \mu(\phi)$$

It's REINFORCE gradient is

$$g_b^{\text{REINFORCE}}(\mathbf{z}, \phi) = \frac{1}{M} \sum_{m=1}^M (f(\mathbf{z}_m) - b(\mathbf{z}_m)) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}_m) + \frac{\partial}{\partial \phi} \mu(\phi)$$

- ▶ $b(\mathbf{z})$ is typically called *baseline*
- ▶ Essentially a control variate of the form $b(\mathbf{z}) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z})$
 - ▶ Other CVs are possible, but this is convenient as it approximates the function itself
- ▶ Unbiased estimate of the true gradient since $\mathbb{E}_{q_{\phi}(\mathbf{z})} b(\mathbf{z}) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}) = \frac{\partial}{\partial \phi} \mu(\phi)$
- ▶ For right $b(\mathbf{z})$ might have much lower variance

$$\mathbb{E}_{q_{\phi}(\mathbf{z}_{1:M})} \left[\frac{1}{M} \sum_{m=1}^M (f(\mathbf{z}_m) - b(\mathbf{z}_m)) \right] + \mu(\phi)$$

It's REINFORCE gradient is

$$g_b^{\text{REINFORCE}}(\mathbf{z}, \phi) = \frac{1}{M} \sum_{m=1}^M (f(\mathbf{z}_m) - b(\mathbf{z}_m)) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}_m) + \frac{\partial}{\partial \phi} \mu(\phi)$$

- ▶ $b(\mathbf{z})$ is typically called *baseline*
- ▶ Essentially a control variate of the form $b(\mathbf{z}) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z})$
 - ▶ Other CVs are possible, but this is convenient as it approximates the function itself
- ▶ Unbiased estimate of the true gradient since $\mathbb{E}_{q_{\phi}(\mathbf{z})} b(\mathbf{z}) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}) = \frac{\partial}{\partial \phi} \mu(\phi)$
- ▶ For right $b(\mathbf{z})$ might have much lower variance

$$\mathbb{E}_{q_{\phi}(\mathbf{z}_{1:M})} \left[\frac{1}{M} \sum_{m=1}^M (f(\mathbf{z}_m) - b(\mathbf{z}_m)) \right] + \mu(\phi)$$

It's REINFORCE gradient is

$$g_b^{\text{REINFORCE}}(\mathbf{z}, \phi) = \frac{1}{M} \sum_{m=1}^M (f(\mathbf{z}_m) - b(\mathbf{z}_m)) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}_m) + \frac{\partial}{\partial \phi} \mu(\phi)$$

- ▶ $b(\mathbf{z})$ is typically called *baseline*
- ▶ Essentially a control variate of the form $b(\mathbf{z}) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z})$
 - ▶ Other CVs are possible, but this is convenient as it approximates the function itself
- ▶ Unbiased estimate of the true gradient since $\mathbb{E}_{q_{\phi}(\mathbf{z})} b(\mathbf{z}) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}) = \frac{\partial}{\partial \phi} \mu(\phi)$
- ▶ For right $b(\mathbf{z})$ might have much lower variance

$$\mathbb{E}_{q_{\phi}(\mathbf{z}_{1:M})} \left[\frac{1}{M} \sum_{m=1}^M (f(\mathbf{z}_m) - b(\mathbf{z}_m)) \right] + \mu(\phi)$$

It's REINFORCE gradient is

$$g_b^{\text{REINFORCE}}(\mathbf{z}, \phi) = \frac{1}{M} \sum_{m=1}^M (f(\mathbf{z}_m) - b(\mathbf{z}_m)) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}_m) + \frac{\partial}{\partial \phi} \mu(\phi)$$

- ▶ $b(\mathbf{z})$ is typically called *baseline*
- ▶ Essentially a control variate of the form $b(\mathbf{z}) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z})$
 - ▶ Other CVs are possible, but this is convenient as it approximates the function itself
- ▶ Unbiased estimate of the true gradient since $\mathbb{E}_{q_{\phi}(\mathbf{z})} b(\mathbf{z}) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}) = \frac{\partial}{\partial \phi} \mu(\phi)$
- ▶ For right $b(\mathbf{z})$ might have much lower variance

$$\mathbb{E}_{q_{\phi}(\mathbf{z}_{1:M})} \left[\frac{1}{M} \sum_{m=1}^M (f(\mathbf{z}_m) - b(\mathbf{z}_m)) \right] + \mu(\phi)$$

It's REINFORCE gradient is

$$g_b^{\text{REINFORCE}}(\mathbf{z}, \phi) = \frac{1}{M} \sum_{m=1}^M (f(\mathbf{z}_m) - b(\mathbf{z}_m)) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}_m) + \frac{\partial}{\partial \phi} \mu(\phi)$$

- ▶ $b(\mathbf{z})$ is typically called *baseline*
- ▶ Essentially a control variate of the form $b(\mathbf{z}) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z})$
 - ▶ Other CVs are possible, but this is convenient as it approximates the function itself
- ▶ Unbiased estimate of the true gradient since $\mathbb{E}_{q_{\phi}(\mathbf{z})} b(\mathbf{z}) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}) = \frac{\partial}{\partial \phi} \mu(\phi)$
- ▶ For right $b(\mathbf{z})$ might have much lower variance

- ▶ **Constant baseline** $b(\mathbf{z}) = c$

$$(f(\mathbf{z}) - c) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}) + \underbrace{\frac{\partial}{\partial \phi} \mathbb{E}_{q_{\phi}(\mathbf{z})} c}_{=0}$$

Just centers the learning signal, the optimal c is

$$c = \frac{\sum_{d=1}^D \text{Cov} \left[f(\mathbf{z}) \frac{\partial}{\partial \phi_d} \log q_{\phi}(\mathbf{z}), \frac{\partial}{\partial \phi_d} \log q_{\phi}(\mathbf{z}) \right]}{\sum_{d=1}^D \text{Var} \left[\frac{\partial}{\partial \phi_d} \log q_{\phi}(\mathbf{z}) \right]}$$

Can be estimated using moving averages, but beware the variance!

- ▶ If some contextual observation is available (like x in VAE), the optimal constant baseline now depends on x . **NVIL** [MG14] proposes to learn the baseline network by minimizing the expected MSE:

$$\mathbb{E}_{p(x)} \mathbb{E}_{q_{\phi}(z|x)} (f(\mathbf{z}) - b(x))^2 \rightarrow \min_{b(x)}$$

- ▶ **Constant baseline** $b(\mathbf{z}) = c$

$$(f(\mathbf{z}) - c) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}) + \underbrace{\frac{\partial}{\partial \phi} \mathbb{E}_{q_{\phi}(\mathbf{z})} c}_{=0}$$

Just centers the learning signal, the optimal c is

$$c = \frac{\sum_{d=1}^D \text{Cov} \left[f(\mathbf{z}) \frac{\partial}{\partial \phi_d} \log q_{\phi}(\mathbf{z}), \frac{\partial}{\partial \phi_d} \log q_{\phi}(\mathbf{z}) \right]}{\sum_{d=1}^D \text{Var} \left[\frac{\partial}{\partial \phi_d} \log q_{\phi}(\mathbf{z}) \right]}$$

Can be estimated using moving averages, but beware the variance!

- ▶ If some contextual observation is available (like x in VAE), the optimal constant baseline now depends on x . **NVIL** [MG14] proposes to learn the baseline network by minimizing the expected MSE:

$$\mathbb{E}_{p(x)} \mathbb{E}_{q_{\phi}(z|x)} (f(\mathbf{z}) - b(x))^2 \rightarrow \min_{b(x)}$$

- ▶ **Constant baseline** $b(\mathbf{z}) = c$

$$(f(\mathbf{z}) - c) \frac{\partial}{\partial \phi} \log q_{\phi}(\mathbf{z}) + \underbrace{\frac{\partial}{\partial \phi} \mathbb{E}_{q_{\phi}(\mathbf{z})} c}_{=0}$$

Just centers the learning signal, the optimal c is

$$c = \frac{\sum_{d=1}^D \text{Cov} \left[f(\mathbf{z}) \frac{\partial}{\partial \phi_d} \log q_{\phi}(\mathbf{z}), \frac{\partial}{\partial \phi_d} \log q_{\phi}(\mathbf{z}) \right]}{\sum_{d=1}^D \text{Var} \left[\frac{\partial}{\partial \phi_d} \log q_{\phi}(\mathbf{z}) \right]}$$

Can be estimated using moving averages, but beware the variance!

- ▶ If some contextual observation is available (like x in VAE), the optimal constant baseline now depends on x . **NVIL** [MG14] proposes to learn the baseline network by minimizing the expected MSE:

$$\mathbb{E}_{p(x)} \mathbb{E}_{q_{\phi}(z|x)} (f(\mathbf{z}) - b(x))^2 \rightarrow \min_{b(x)}$$

Let $b(\mathbf{z})$ be a first-order Taylor expansion of $f(\mathbf{z})$ at some point μ :

$$b(\mathbf{z}) = f(\mu) + \frac{\partial f}{\partial \mathbf{z}}(\mu)^T (\mathbf{z} - \mu)$$

We'll take $\mu := \mu(\phi) = \mathbb{E}_{q_\phi(\mathbf{z})} \mathbf{z}$. This leads to

$$g^{\mu\text{-prop}}(\mathbf{z}, \phi) = (f(\mathbf{z}) - b(\mathbf{z})) \frac{\partial}{\partial \phi} \log q_\phi(\mathbf{z}) + \frac{\partial f(\mu(\phi))}{\partial \phi}$$

- ▶ Backpropagates through the mean, and then fine-tunes inaccuracies with REINFORCE
- ▶ One could use 2nd order Taylor expansion, but that is more computationally expensive

Let $b(\mathbf{z})$ be a first-order Taylor expansion of $f(\mathbf{z})$ at some point μ :

$$b(\mathbf{z}) = f(\mu) + \frac{\partial f}{\partial \mathbf{z}}(\mu)^T (\mathbf{z} - \mu)$$

We'll take $\mu := \mu(\phi) = \mathbb{E}_{q_\phi(\mathbf{z})} \mathbf{z}$. This leads to

$$g^{\mu\text{-prop}}(\mathbf{z}, \phi) = (f(\mathbf{z}) - b(\mathbf{z})) \frac{\partial}{\partial \phi} \log q_\phi(\mathbf{z}) + \frac{\partial f(\mu(\phi))}{\partial \phi}$$

- ▶ Backpropagates through the mean, and then fine-tunes inaccuracies with REINFORCE
- ▶ One could use 2nd order Taylor expansion, but that is more computationally expensive

Let $b(\mathbf{z})$ be a first-order Taylor expansion of $f(\mathbf{z})$ at some point μ :

$$b(\mathbf{z}) = f(\mu) + \frac{\partial f}{\partial \mathbf{z}}(\mu)^T (\mathbf{z} - \mu)$$

We'll take $\mu := \mu(\phi) = \mathbb{E}_{q_\phi(\mathbf{z})} \mathbf{z}$. This leads to

$$g^{\mu\text{-prop}}(\mathbf{z}, \phi) = (f(\mathbf{z}) - b(\mathbf{z})) \frac{\partial}{\partial \phi} \log q_\phi(\mathbf{z}) + \frac{\partial f(\mu(\phi))}{\partial \phi}$$

- ▶ Backpropagates through the mean, and then fine-tunes inaccuracies with REINFORCE
- ▶ One could use 2nd order Taylor expansion, but that is more computationally expensive

Let $b(\mathbf{z})$ be a first-order Taylor expansion of $f(\mathbf{z})$ at some point μ :

$$b(\mathbf{z}) = f(\mu) + \frac{\partial f}{\partial \mathbf{z}}(\mu)^T (\mathbf{z} - \mu)$$

We'll take $\mu := \mu(\phi) = \mathbb{E}_{q_\phi(\mathbf{z})} \mathbf{z}$. This leads to

$$g^{\mu\text{-prop}}(\mathbf{z}, \phi) = (f(\mathbf{z}) - b(\mathbf{z})) \frac{\partial}{\partial \phi} \log q_\phi(\mathbf{z}) + \frac{\partial f(\mu(\phi))}{\partial \phi}$$

- ▶ Backpropagates through the mean, and then fine-tunes inaccuracies with REINFORCE
- ▶ One could use 2nd order Taylor expansion, but that is more computationally expensive

Typically we seek low-variance estimators. Why not minimize the variance w.r.t. a baseline in the first place?

$$\text{Var}[g(\mathbf{z}, \phi)] = \mathbb{E} g(\mathbf{z}, \phi)^2 - (\mathbb{E} g(\mathbf{z}, \phi))^2$$

- ▶ In general minimizing variance leads to increase in bias
 - ▶ Estimators with control variates are **unbiased for any baseline**
- ▶ Use Stochastic Optimization to minimize the second moment of the gradient.

For example, for **NVIL** a better objective would be

$$\mathbb{E}_{p(x)} \mathbb{E}_{q_{\phi}(z|x)} \left((f(\mathbf{z}) - b(x)) \frac{\partial}{\partial \phi} \log q_{\phi}(z|x) \right)^2 \rightarrow \min_{b(x)}$$

Typically we seek low-variance estimators. Why not minimize the variance w.r.t. a baseline in the first place?

$$\text{Var}[g(\mathbf{z}, \phi)] = \mathbb{E} g(\mathbf{z}, \phi)^2 - (\mathbb{E} g(\mathbf{z}, \phi))^2$$

- ▶ In general minimizing variance leads to increase in bias
 - ▶ Estimators with control variates are **unbiased for any baseline**
- ▶ Use Stochastic Optimization to minimize the second moment of the gradient.

For example, for **NVIL** a better objective would be

$$\mathbb{E}_{p(x)} \mathbb{E}_{q_{\phi}(z|x)} \left((f(\mathbf{z}) - b(x)) \frac{\partial}{\partial \phi} \log q_{\phi}(z|x) \right)^2 \rightarrow \min_{b(x)}$$

Typically we seek low-variance estimators. Why not minimize the variance w.r.t. a baseline in the first place?

$$\text{Var}[g(\mathbf{z}, \phi)] = \mathbb{E} g(\mathbf{z}, \phi)^2 - (\mathbb{E} g(\mathbf{z}, \phi))^2$$

- ▶ In general minimizing variance leads to increase in bias
 - ▶ Estimators with control variates are **unbiased for any baseline**
- ▶ Use Stochastic Optimization to minimize the second moment of the gradient.

For example, for **NVIL** a better objective would be

$$\mathbb{E}_{p(x)} \mathbb{E}_{q_{\phi}(z|x)} \left((f(\mathbf{z}) - b(x)) \frac{\partial}{\partial \phi} \log q_{\phi}(z|x) \right)^2 \rightarrow \min_{b(x)}$$

Typically we seek low-variance estimators. Why not minimize the variance w.r.t. a baseline in the first place?

$$\text{Var}[g(\mathbf{z}, \phi)] = \mathbb{E} g(\mathbf{z}, \phi)^2 - (\mathbb{E} g(\mathbf{z}, \phi))^2$$

- ▶ In general minimizing variance leads to increase in bias
 - ▶ Estimators with control variates are **unbiased for any baseline**
- ▶ Use Stochastic Optimization to minimize the second moment of the gradient.

For example, for **NVIL** a better objective would be

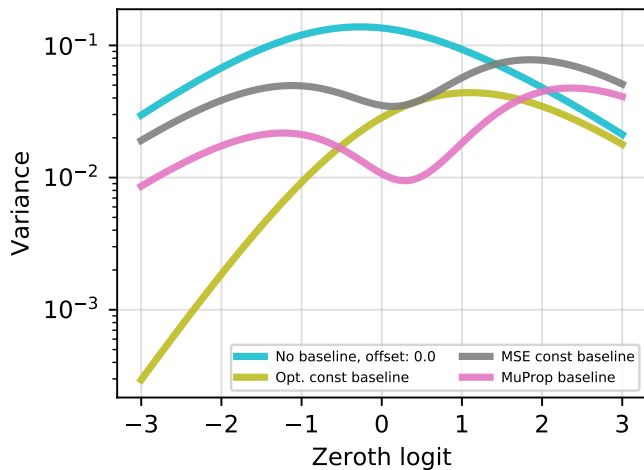
$$\mathbb{E}_{p(x)} \mathbb{E}_{q_{\phi}(z|x)} \left((f(\mathbf{z}) - b(x)) \frac{\partial}{\partial \phi} \log q_{\phi}(z|x) \right)^2 \rightarrow \min_{b(x)}$$

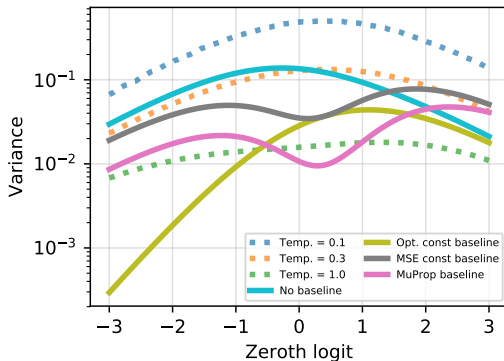
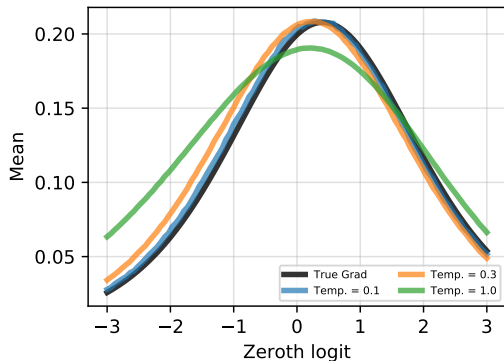
- ▶ REBAR [TMM⁺17] uses Gumbel-relaxed f as a baseline

$$g^{\text{REBAR}}(\mathbf{z}, \phi) := (f(\mathbf{z}) - \eta f(\tilde{\mathbf{z}}_\phi | \mathbf{z})) \frac{\partial}{\partial \phi} \log q_\phi(\mathbf{z}) + \eta \frac{\partial}{\partial \phi} (f(\tilde{\mathbf{z}}_\phi) - f(\tilde{\mathbf{z}}_\phi | \mathbf{z}))$$

- ▶ Efficiently reparametrizable
 - ▶ Hyperparameters τ and η can be learned using the variance minimization principle
 - ▶ Backpropagates through Gumbel-relaxed objective, but has additional corrections for the introduced bias
 - ▶ **The baseline's expectation is intractable, but reparametrizable**
- ▶ RELAX [GCW⁺18] learns the baseline $b(z)$ using variance minimization:

$$g^{\text{RELAX}}(\mathbf{z}, \phi) := (f(\mathbf{z}) - b(\tilde{\mathbf{z}}_\phi | \mathbf{z})) \frac{\partial}{\partial \phi} \log q_\phi(\mathbf{z}) + \frac{\partial}{\partial \phi} (b(\tilde{\mathbf{z}}_\phi) - b(\tilde{\mathbf{z}}_\phi | \mathbf{z}))$$





Conclusion

Relaxation-based methods

- ▶ Straightforward to implement
- ▶ Work well in practice
- ▶ Have hyperparameters to tune
- ▶ Have biased gradients aka introduce train-test mismatch

Variance Reduction methods

- ▶ Cumbersome
- ▶ Not clear if their results are worth added complexity
- ▶ Always unbiased
- ▶ Allow you to tune baseline to minimize variance
- ▶ **Random search on steroids**

Still ongoing research topic, many other approaches not covered

Relaxation-based methods

- ▶ Straightforward to implement
- ▶ Work well in practice
- ▶ Have hyperparameters to tune
- ▶ Have biased gradients aka introduce train-test mismatch

Variance Reduction methods

- ▶ Cumbersome
- ▶ Not clear if their results are worth added complexity
- ▶ Always unbiased
- ▶ Allow you to tune baseline to minimize variance
- ▶ **Random search on steroids**

Still ongoing research topic, many other approaches not covered

For more estimators for both discrete and continuous cases see my blog:

- ▶ [http://artem.sobolev.name/tags/stochastic computation graphs series.html](http://artem.sobolev.name/tags/stochastic%20computation%20graphs%20series.html)

For more in-depth treatment of the continuous case see "Monte Carlo Gradient Estimation in Machine Learning" by S. Mohamed, M. Rosca, M. Figurnov, A. Mnih:

- ▶ <https://arxiv.org/abs/1906.10652>

-  Will Grathwohl, Dami Choi, Yuhuai Wu, Geoff Roeder, and David Duvenaud, *Backpropagation through the void: Optimizing control variates for black-box gradient estimation*, International Conference on Learning Representations, 2018.
-  Shixiang Gu, Sergey Levine, Ilya Sutskever, and Andriy Mnih, *Muprop: Unbiased backpropagation for stochastic neural networks*, International Conference on Learning Representations, 2016.
-  Eric Jang, Shixiang Gu, and Ben Poole, *Categorical reparameterization with gumbel-softmax*, International Conference on Learning Representations, 2017.
-  Andriy Mnih and Karol Gregor, *Neural variational inference and learning in belief networks*, Proceedings of the 31st International Conference on Machine Learning (ICML), 2014.

-  Chris J. Maddison, Andriy Mnih, and Yee Whye Teh, *The concrete distribution: A continuous relaxation of discrete random variables*, International Conference on Learning Representations, 2017.
-  Ben Recht, *arg min blog: The policy of truth*, <http://www.argmin.net/2018/02/20/reinforce/>, 2018.
-  George Tucker, Andriy Mnih, Chris J Maddison, John Lawson, and Jascha Sohl-Dickstein, *Rebar: Low-variance, unbiased gradient estimates for discrete latent variable models*, Advances in Neural Information Processing Systems 30 (I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds.), Curran Associates, Inc., 2017, pp. 2627–2636.