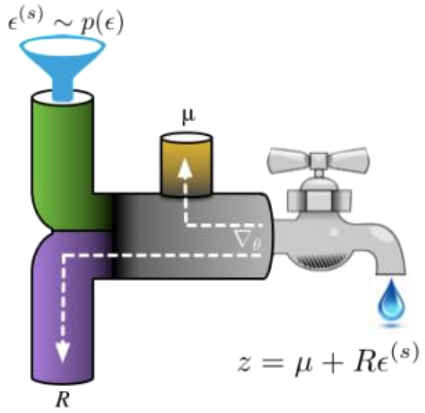
MC gradient estimators



http://blog.shakirm.com/2015/10/machine-learning-trick-of-the-day-4-reparameterisation-tricks/

Families of MC gradient estimators

- Score-function estimator
- Pathwise gradient estimators
- Measure-valued gradient estimators

Monte Carlo Gradient Estimation in Machine Learning

Score-function estimator

- Better known as the REINFORCE algorithm
- Exploiting the following property

$$\frac{d}{d\boldsymbol{x}}\log f(\boldsymbol{x}) = \frac{1}{f(\boldsymbol{x})}\cdot\frac{df(\boldsymbol{x})}{d\boldsymbol{x}}$$

- When our function $f(\boldsymbol{x})$ is a probability density

$$\nabla_\varphi \log p_\varphi(\boldsymbol{x}) = \frac{1}{p_\varphi(\boldsymbol{x})}\nabla_\varphi p_\varphi(\boldsymbol{x}) \Leftrightarrow \nabla_\varphi p_\varphi(\boldsymbol{x}) = p_\varphi(\boldsymbol{x})\nabla_\varphi \log p_\varphi(\boldsymbol{x})$$

  - $\nabla_\varphi \log p(x)$: score-function
- A neat trick to rewrite the gradient of a density as another density

Deriving the score-function estimator

- As a use case the following expectation from VAE: $\nabla_\varphi \mathbb{E}_{\boldsymbol{z}\sim q_\varphi(\boldsymbol{z}|\boldsymbol{x})}[\log p(\boldsymbol{x}|\boldsymbol{z})]$

$$\nabla_\varphi \mathbb{E}_{\boldsymbol{z}\sim q_\varphi(\boldsymbol{z}|\boldsymbol{x})}[\log p(\boldsymbol{x}|\boldsymbol{z})] =$$

$$= \nabla_\varphi \int_{\boldsymbol{z}} \log p(\boldsymbol{x}|\boldsymbol{z})\, q_\varphi(\boldsymbol{z}|\boldsymbol{x}) d\boldsymbol{z}$$

$$= \int_{\boldsymbol{z}} \log p(\boldsymbol{x}|\boldsymbol{z})\, \nabla_\varphi q_\varphi(\boldsymbol{z}|\boldsymbol{x})\, d\boldsymbol{z}$$

$$= \int_{\boldsymbol{z}} \log p(\boldsymbol{x}|\boldsymbol{z})\, q_\varphi(\boldsymbol{z}|\boldsymbol{x}) \nabla_\varphi \log q_\varphi(\boldsymbol{z}|\boldsymbol{x})\, d\boldsymbol{z}$$

$$= \mathbb{E}_{\boldsymbol{z}\sim q_\varphi(\boldsymbol{z}|\boldsymbol{x})}\left[\log p(\boldsymbol{x}|\boldsymbol{z})\, \nabla_\varphi \log q_\varphi(\boldsymbol{z}|\boldsymbol{x})\right]$$

$$\approx \frac{1}{n}\sum_i \log p\left(\boldsymbol{x}|\boldsymbol{z}^{(i)}\right) \nabla_\varphi \log q_\varphi\left(\boldsymbol{z}^{(i)}|\boldsymbol{x}\right), \boldsymbol{z}^{(i)} \sim q_\varphi(\boldsymbol{z}|\boldsymbol{x})$$

Score-function estimator properties

- Any function $f(\boldsymbol{x})$ amenable
  - Good for simulators or black box functions (RL)
- The $p_\varphi(\boldsymbol{x})$ must be differentiable w.r.t. to parameters φ
- It must be easy to sample from $p_\varphi(\boldsymbol{x})$
- Unbiased estimator
- High variance estimator
  - The gradient will deviate a lot, but in the limit of many samples is accurate
  - Increases with more dimensions
  - If you sample once, this can be a problem and slow down or stop learning
  - Variance reduction methods are usually needed

Pathwise gradient estimator

- Also known as ‘reparameterization trick’
- Often the probability density can be rewritten as
  - a deterministic function of a simpler probability density
- Instead of sampling from a complex pdf → sample from the simpler one
  - then transform deterministically the sample

$$\hat{\boldsymbol{x}} \sim p_{\varphi}(\boldsymbol{x}) \Leftrightarrow \hat{\boldsymbol{x}} = g(\hat{\boldsymbol{\varepsilon}}, \varphi), \hat{\boldsymbol{\varepsilon}} \sim p(\boldsymbol{\varepsilon})$$

Pathwise gradient estimator

$$\hat{\boldsymbol{x}} \sim p_{\varphi}(\boldsymbol{x}) \Leftrightarrow \hat{\boldsymbol{x}} = g(\hat{\boldsymbol{\varepsilon}}, \varphi), \hat{\boldsymbol{\varepsilon}} \sim p(\boldsymbol{\varepsilon})$$

- Stochasticity flows through a simple probability density
  - And, complexity flows from the deterministic transformation
  - For NN it means backprop –for deterministic functions only- is possible
- At the heart of this method is the change of variables formula

$$p_{\varphi}(\boldsymbol{x}) = p(\boldsymbol{\varepsilon}) |\det \nabla_{\boldsymbol{\varepsilon}} g(\boldsymbol{\varepsilon}, \varphi)|^{-1}$$

  - We have seen normalizing flows using the same property

Deriving the pathwise gradient estimator

- As a use case the following expectation from VAE: $\nabla_\varphi \mathbb{E}_{\boldsymbol{z}\sim q_\varphi(\boldsymbol{z}|\boldsymbol{x})}[\log p(\boldsymbol{x}|\boldsymbol{z})]$
  - $\boldsymbol{z} = g(\boldsymbol{\varepsilon}, \varphi|\boldsymbol{x}) = \boldsymbol{\mu}_{\boldsymbol{x}} + \boldsymbol{\varepsilon}\cdot\boldsymbol{\sigma}_{\boldsymbol{x}}$, where $\varphi = (\boldsymbol{\mu}_{\boldsymbol{x}}, \boldsymbol{\sigma}_{\boldsymbol{x}}) \Rightarrow d\boldsymbol{z} = \boldsymbol{\sigma}_{\boldsymbol{x}} d\boldsymbol{\varepsilon}$
  - $\det \nabla_{\boldsymbol{\varepsilon}} g(\boldsymbol{\varepsilon}, \varphi|x) = \prod_i \sigma_{x,i}$

$$\nabla_\varphi \mathbb{E}_{\boldsymbol{z}\sim q_\varphi(\boldsymbol{z}|\boldsymbol{x})}[\log p(\boldsymbol{x}|\boldsymbol{z})] =$$

$$= \nabla_\varphi \int_{\boldsymbol{z}} q_\varphi(\boldsymbol{z}|\boldsymbol{x}) \log p(\boldsymbol{x}|\boldsymbol{z})\, d\boldsymbol{z}$$

$$= \nabla_\varphi \int_{\boldsymbol{z}} \frac{1}{\prod_i \sigma_{x,i}} p(\boldsymbol{\varepsilon}) \log p(\boldsymbol{x}|g(\boldsymbol{\varepsilon}, \varphi|\boldsymbol{x})) \prod_i \sigma_{x,i}\, d\boldsymbol{\varepsilon}$$

$$= \int_{\boldsymbol{\varepsilon}} p(\boldsymbol{\varepsilon})\, \nabla_\varphi \log p(\boldsymbol{x}|g(\boldsymbol{\varepsilon}, \varphi|\boldsymbol{x}))\, d\boldsymbol{\varepsilon}$$

$$= \mathbb{E}_{\boldsymbol{\varepsilon}\sim p(\boldsymbol{\varepsilon})}\left[\nabla_\varphi \log p(\boldsymbol{x}|g(\boldsymbol{\varepsilon}, \varphi|\boldsymbol{x}))\right]$$

$$\approx \frac{1}{n}\sum_i \nabla_\varphi \log p\left(\boldsymbol{x}\middle|g(\boldsymbol{\varepsilon}^{(i)}, \varphi|\boldsymbol{x})\right), \boldsymbol{\varepsilon}^{(i)}\sim p(\varepsilon)$$

Pathwise gradient estimator properties

- Only differentiable cost functions
  - Otherwise we cannot compute the $\nabla_\varphi f(\boldsymbol{x}, g(\boldsymbol{\varepsilon}, \varphi))$
  - Unlike score-function estimators that work with any cost function
- No need to know the pdf explicitly
  - Only the deterministic transformation and the base sampling distribution
- Low variance in general
  - Lower than the score-function estimator
  - Example: if you compare the VAE score-function and pathwise gradients , the score-function has an extra multiplicative term

$$\frac{1}{n}\sum_i \log p\left(\boldsymbol{x}|\boldsymbol{z}^{(i)}\right) \nabla_\varphi \log q_\varphi\left(\boldsymbol{z}^{(i)}|\boldsymbol{x}\right) \qquad \frac{1}{n}\sum_i \nabla_\varphi \log p\left(\boldsymbol{x}|g(\boldsymbol{\varepsilon}^{(i)}, \varphi)\right)$$

- Very efficient (why proposed in VAE)
  - Even a single sample suffices no matter dimensionality

Qualitative comparison between estimators (1)

- Pathwise gradients have consistently lower variance

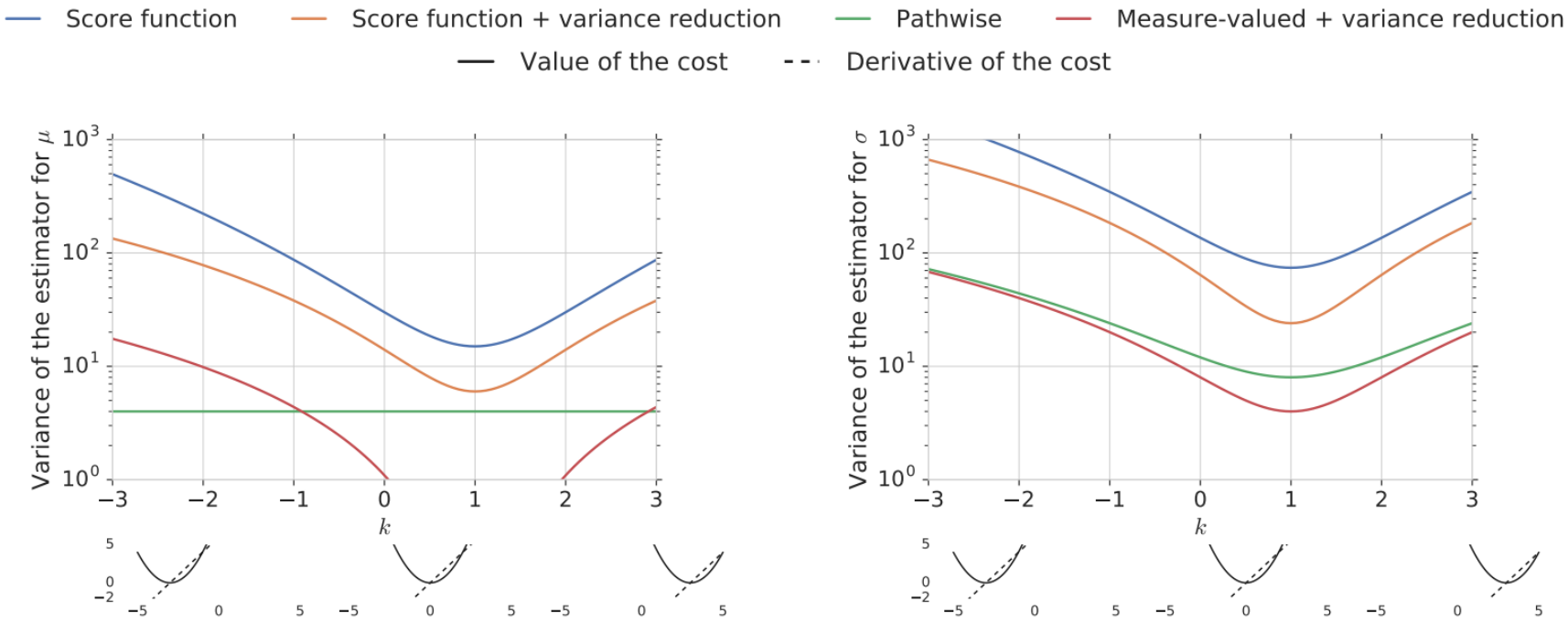


Figure 2: Variance of the stochastic estimates of $\nabla_\theta \mathbb{E}_{\mathcal{N}(x|\mu,\sigma^2)}\left[(x-k)^2\right]$ for $\mu = \sigma = 1$ as a function of k for three different classes of gradient estimators. Left: $\theta = \mu$; right: $\theta = \sigma$. The graphs in the bottom row show the function (solid) and its gradient (dashed) for $k \in \{-3, 0, 3\}$.

Monte Carlo Gradient Estimation in Machine Learning

Qualitative comparison between estimators (2)

- For complex functions the pathwise gradient might have higher variance

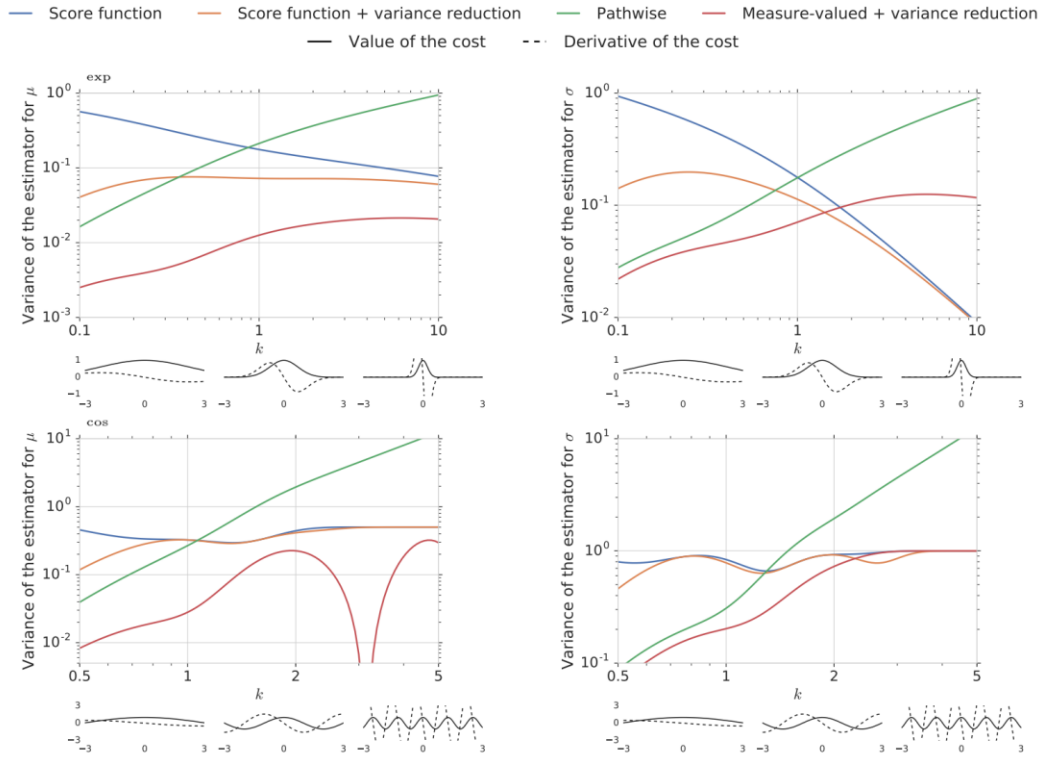


Figure 3: Variance of the stochastic estimates of $\nabla_\theta \mathbb{E}_{\mathcal{N}(x|\mu,\sigma^2)}[f(x;k)]$ for $\mu = \sigma = 1$ as a function of k. Top: $f(x;k) = \exp(-kx^2)$, bottom: $f(x;k) = \cos kx$. Left: $\theta = \mu$; right: $\theta = \sigma$. The graphs in the bottom row show the function (solid) and its gradient (dashed): for $k \in \{0.1, 1, 10\}$ for the exponential function, and $k \in \{0.5, 1.58, 5.\}$ for the cosine function.

Monte Carlo Gradient Estimation in Machine Learning