Monte Carlo Simulation
X-Page Essay

Monte Carlo Gradient Estimation

Shane Gladson

Department of Statistics, University of Chicago

1 Introduction

At the core of modern machine learning techniques is gradient estimation. Specifically, most machine learning models focus extensively on optimizing an objective function (referred to as the *cost*). We can see this in basic statistical models such as maximum likelihood estimation or least squares, however as model complexity increases we encounter a corresponding decrease in tractability. Analytic techniques such as the finite difference method exist when optimizing deterministic systems, however we must use probabilistic approaches for models in which the estimated value is stochastic in nature. In this paper, we will discuss approaches to using Monte Carlo methods for gradient estimation with applications to machine learning. This paper gives an overview of the topics covered in [5], and implements some of the techniques discussed.

We aim to compute the gradient of an objective function given as

$$\mathcal{F}(\boldsymbol{\theta}) := \int p(\boldsymbol{x}|\boldsymbol{\theta}) f(\boldsymbol{x}|\boldsymbol{\varphi}) d\boldsymbol{x} = \mathbb{E}_{p(\boldsymbol{x}|\boldsymbol{\theta})}[f(\boldsymbol{x}|\boldsymbol{\varphi})].$$

From this objective function, we assume f is a cost function with fixed *structural parameters* $\boldsymbol{\varphi}$, and p is a probability distribution with *unknown* parameters $\boldsymbol{\theta}$. Put plainly, we are attempting to optimize the expectation of the cost function f with respect to a distribution p with unknown parameters. A natural result using simple Monte Carlo gives the following estimator of the objective function (this will be used in the following section):

$$\widehat{\mathcal{F}}_N := \frac{1}{N}\sum_{i=1}^{N} f(\boldsymbol{x}_i)$$

where N is the sample size and $\boldsymbol{x}_i$ are independent draws from $p(\boldsymbol{x}|\boldsymbol{\theta})$. For reasons already covered in earlier sections of this course, we know that $\widehat{\mathcal{F}}_N$ is unbiased and consistent.

In the scenarios which we are exploring, the unknown parameters $\boldsymbol{\theta}$ are of interest, and so the gradient of this objective function is then written as

$$\nabla_{\boldsymbol{\theta}}\mathcal{F}(\boldsymbol{\theta}) = \nabla_{\boldsymbol{\theta}}\mathbb{E}_{p(\boldsymbol{x}|\boldsymbol{\theta})}[f(\boldsymbol{x}|\boldsymbol{\varphi})].$$

In machine learning, as well as in other areas of research, this gradient can quickly become impossible to compute analytically or intractable through numerical differentiation. Whether this is due to the cost function not being differentiable or the dimension being too large, there are huge benefits to the use of Monte Carlo methods for these applications.

In this paper, we will explore 3 different approaches to Monte Carlo gradient estimation: *score function estimators*, *pathwise gradient estimators*, and *measure valued gradients.*

2 Score Function

In this section, we will discuss the *score function.* This is simply defined as the gradient of the log-probability (measure). We use the score function, because the derivative of the log-measure has a nice property as can be seen below:

$$\nabla_{\boldsymbol{\theta}} \ln(p(\boldsymbol{x}|\boldsymbol{\theta})) = \frac{\nabla_{\boldsymbol{\theta}} p(\boldsymbol{x}|\boldsymbol{\theta})}{p(\boldsymbol{x}|\boldsymbol{\theta})}.$$

Assuming that $\nabla_{\boldsymbol{\theta}} \ln(p(\boldsymbol{x}|\boldsymbol{\theta}))$ is easier to compute either analytically or numerically, we can derive an estimator for $\nabla_{\boldsymbol{\theta}}\mathcal{F}(\boldsymbol{\theta})$ as follows:

$$\begin{aligned}
\nabla_{\boldsymbol{\theta}}\mathcal{F}(\boldsymbol{\theta})_{SG} &= \nabla_{\boldsymbol{\theta}} \int p(\boldsymbol{x}|\boldsymbol{\theta}) f(\boldsymbol{x}|\boldsymbol{\varphi}) d\boldsymbol{x} \\
&= \int \nabla_{\boldsymbol{\theta}} p(\boldsymbol{x}|\boldsymbol{\theta}) f(\boldsymbol{x}|\boldsymbol{\varphi}) d\boldsymbol{x} \\
&= \int p(\boldsymbol{x}|\boldsymbol{\theta}) f(\boldsymbol{x}|\boldsymbol{\varphi}) \nabla_{\boldsymbol{\theta}} \ln(p(\boldsymbol{x}|\boldsymbol{\theta})) d\boldsymbol{x} \\
&\approx \frac{1}{N}\sum_{i=1}^{N} f(\boldsymbol{x}_i) \nabla_{\boldsymbol{\theta}} \ln(p(\boldsymbol{x}_i|\boldsymbol{\theta})),
\end{aligned}$$

where again N is the sample size and $\boldsymbol{x}_i$ are independent samples from $p(\boldsymbol{x}|\boldsymbol{\theta})$. One significant advantage of this method stems from the following result:

$$\mathbb{E}_{p(\boldsymbol{x}|\boldsymbol{\theta})}[\nabla_{\boldsymbol{\theta}} \ln(p(\boldsymbol{x}|\boldsymbol{\theta}))] = \int p(\boldsymbol{x}|\boldsymbol{\theta}) \frac{\nabla_{\boldsymbol{\theta}} p(\boldsymbol{x}|\boldsymbol{\theta})}{p(\boldsymbol{x}|\boldsymbol{\theta})} d\boldsymbol{x} = \nabla_{\boldsymbol{\theta}} \int p(\boldsymbol{x}|\boldsymbol{\theta}) d\boldsymbol{x} = \nabla_{\boldsymbol{\theta}} 1 = 0.$$

Using this result, we can rewrite the gradient by adding 0 in a special way to use control variates and reduce variance while maintaining an unbiased estimator as follows:

$$\nabla_{\boldsymbol{\theta}}\mathcal{F}(\boldsymbol{\theta})_{SG} = \mathbb{E}_{p(\boldsymbol{x}|\boldsymbol{\theta})}[(f(\boldsymbol{x}) - \beta) \nabla_{\boldsymbol{\theta}} \ln(p(\boldsymbol{x}|\boldsymbol{\theta}))].$$

Note that this result holds for all β. Although it is not guaranteed to minimize the variance, it is very common to choose β to be the average of the cost. This is largely due to ease of computation. An example is included in Figure 2. Using control variates with the average of the cost, the mean squared error for estimates of μ and σ^2 were reduced by 60% and 40% respectively.

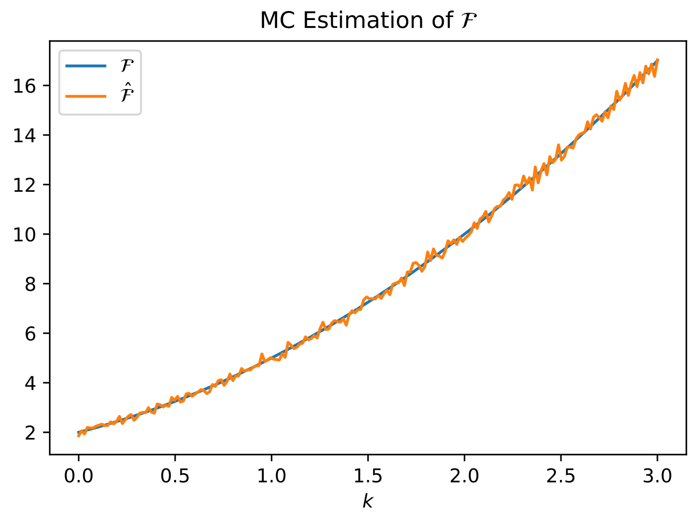

Figure 1 Classical Monte Carlo estimate of $\mathcal{F}$ where $p(\boldsymbol{x}|\boldsymbol{\theta}) \sim \mathcal{N}(1,1)$ and $f(\boldsymbol{x}|\boldsymbol{\varphi}) = (x+k)^2$, $N = 1000$.

Important conditions for this method to work as desired are that the measure p must be continuously differentiable with respect to $\boldsymbol{\theta}$ and $p(\boldsymbol{x}|\boldsymbol{\theta})f(\boldsymbol{x})$ must be integrable. In addition, we require that p is absolutely continuous to ensure that the gradient estimator is unbiased.

Using the formula for variance, we can determine the variance of the estimator as

$$\begin{aligned} V(\nabla_{\boldsymbol{\theta}}\widehat{\mathcal{F}(\boldsymbol{\theta})}_{SG}) &= \frac{1}{N}\mathbb{E}\Big[(f(\boldsymbol{x}_i)\nabla_{\boldsymbol{\theta}}\ln(p(\boldsymbol{x}_i|\boldsymbol{\theta})))^2\Big] - \frac{1}{N}\mathbb{E}[f(\boldsymbol{x}_i)\nabla_{\boldsymbol{\theta}}\ln(p(\boldsymbol{x}_i|\boldsymbol{\theta}))]^2 \\ &= \frac{1}{N}\mathbb{E}\Big[(f(\boldsymbol{x}_i)\nabla_{\boldsymbol{\theta}}\ln(p(\boldsymbol{x}_i|\boldsymbol{\theta})))^2\Big] - \frac{1}{N}(\nabla_{\boldsymbol{\theta}}\mathcal{F}(\boldsymbol{\theta}))^2. \end{aligned}$$

Main sources of high variance will be due to high dimensionality or high-variance cost functions. While not always possible, this emphasises the benefits of using low-variance cost functions when performing stochastic optimization.

The computational cost of generating this estimator is on the order of $\mathcal{O}(N(D+L))$ where N is the sample size, D is the dimension of $\boldsymbol{\theta}$, and L is the computational cost of f. This is the same cost as the pathwise gradient estimator, which will be discussed next. The main draw of using this method is it's ease of computation and is generalized to any cost function, which is an advantage that not all estimators have as we will see later.

In practice, we can also use the score function $\nabla_{\boldsymbol{\theta}}\ln(p)$ to generate data to be used in machine learning models.

2.1 Example

Consider the objective function $\mathcal{F}$ where the cost function is $f(\boldsymbol{x}|\boldsymbol{\varphi}) = (x+k)^2$ and the density function $p(\boldsymbol{x}|\boldsymbol{\theta}) \sim \mathcal{N}(\mu, \sigma^2)$. Then we can compute the gradient $\nabla_{\boldsymbol{\theta}}\ln(p)$ as

$$\nabla_{\boldsymbol{\theta}}\mathcal{F}(\boldsymbol{\theta}) = \begin{pmatrix} \frac{\partial \mathcal{F}}{\partial \mu} \\ \frac{\partial \mathcal{F}}{\partial \sigma^2} \end{pmatrix} = \begin{pmatrix} \frac{x-\mu}{\sigma^2} \\ \frac{-1}{2\sigma^2} + \frac{(x-\mu)^2}{2\sigma^4} \end{pmatrix}.$$

With this result, we can sample $\boldsymbol{x}_i$ independently from $\mathcal{N}(\mu, \sigma^2)$ and use the score gradient estimator to get the plot of gradient estimates included in Figure 2. While the score gradient is simple to compute, we can very easily reduce variance using the previously discussed method, as shown by the green line in Figure 2.

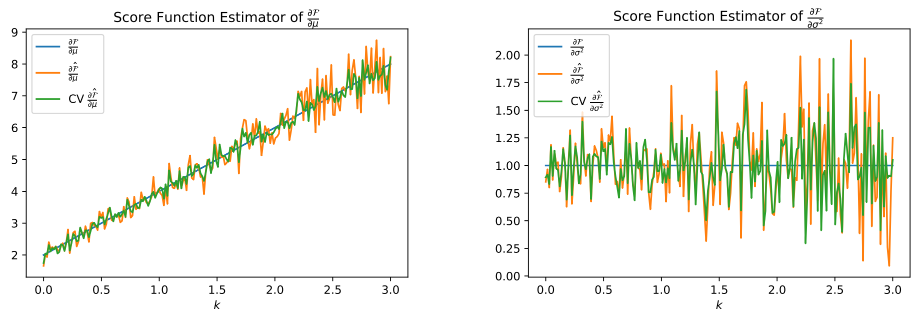

Figure 2 Score gradient estimate of $\mathcal{F}$ where $p(\boldsymbol{x}|\boldsymbol{\theta}) \sim \mathcal{N}(1,1)$ and $f(\boldsymbol{x}|\boldsymbol{\varphi}) = (x+k)^2$, $N = 1000$.

3 Pathwise Gradient Estimator

Pathwise gradient estimators are an interesting approach to computing the gradient. Similar to the intuition obtained from the first chapter of the textbook where we covered transformation methods, we can assume that all samples $\boldsymbol{x}_i$ from a probability distribution $p(\boldsymbol{x}|\boldsymbol{\theta})$ can be constructed as $\boldsymbol{x}_i = g(\boldsymbol{\varepsilon}_i|\boldsymbol{\theta})$ where $\boldsymbol{\varepsilon}_i$ is independently sampled from a simpler distribution $p_{base}(\boldsymbol{\varepsilon})$ and g maps $\boldsymbol{\varepsilon}_i$ to $\boldsymbol{x}_i$.

The most natural method would be to sample $\boldsymbol{\varepsilon}_i$ from the Uniform$(0,1)$ distribution and to use the inverse transformation method or the Knothe-Rosenblatt rearrangement, although this method is also valid for transforming centered and scaled distributions to their desired mean and variance. We assume that g is invertible with respect to $\boldsymbol{\varepsilon}$. Then the pathwise gradient estimator is derived as follows:

$$\begin{aligned}
\nabla_{\boldsymbol{\theta}}\mathcal{F}(\boldsymbol{\theta}) &= \nabla_{\boldsymbol{\theta}} \int p(\boldsymbol{x}|\boldsymbol{\theta}) f(\boldsymbol{x}) d\boldsymbol{x} \\
&= \nabla_{\boldsymbol{\theta}} \int p_{base}(\boldsymbol{\varepsilon}) f(g(\boldsymbol{\varepsilon}|\boldsymbol{\theta})) d\boldsymbol{\varepsilon} \\
&= \int p_{base}(\boldsymbol{\varepsilon}) \nabla_{\boldsymbol{\theta}} f(g(\boldsymbol{\varepsilon}|\boldsymbol{\theta})) d\boldsymbol{\varepsilon} \\
&= \mathbb{E}_{p_{base}(\boldsymbol{\varepsilon})}[\nabla_{\boldsymbol{\theta}} f(g(\boldsymbol{\varepsilon}|\boldsymbol{\theta}))] \\
&\approx \frac{1}{N} \sum_{n=1}^{N} \nabla_{\boldsymbol{\theta}} f(g(\boldsymbol{\varepsilon}_i|\boldsymbol{\theta})),
\end{aligned}$$

where N is the sample size and $\boldsymbol{\varepsilon}_i$ is independently sampled from $p_{base}(\boldsymbol{\varepsilon})$.

This estimator, $\widehat{\nabla_{\boldsymbol{\theta}}\mathcal{F}(\boldsymbol{\theta})}_{PG}$, is limited to distributions that have a differentiable path g. For this reason, we cannot use this method with distributions that would require rejection sampling as it does not produce a differentiable path.

It is simple to see that this estimator is unbiased, however one should note that low variance is far from guaranteed. The estimator makes extensive use of the cost function f, and so highly variable cost functions could quickly lead to gradient estimates with large variance. On the other hand, the variance of the estimator does not depend on the dimension of $\boldsymbol{\theta}$, and so this method may be desirable for estimating high-dimensional gradients.

As for computational considerations, this method has cost on the order of $\mathcal{O}(N(D+L))$ where N is the sample size, D is the dimension of $\boldsymbol{\theta}$, and L is the computational cost of f. This is the same as for the score function, however it is popular to implement this method with only a single sample ($N = 1$) due to rapid convergence of the estimator. While we require that the cost function f is differentiable, one advantage is that only the path function g is needed while the density $p(\boldsymbol{x}|\boldsymbol{\theta})$ does not need to be known.

3.1 Example

Again, consider the objective function $\mathcal{F}$ where the cost function is defined by $f(\boldsymbol{x}|\boldsymbol{\varphi}) = (x+k)^2$ and density function $p(\boldsymbol{x}|\boldsymbol{\theta}) \sim \mathcal{N}(\mu, \sigma^2)$. We are interested in computing the gradient for μ and σ across values of k. We can consider the path $g(\boldsymbol{\varepsilon}|\boldsymbol{\theta}) = \sigma\boldsymbol{\varepsilon} + \mu$ where $\boldsymbol{\varepsilon} \sim \mathcal{N}(0, 1)$. From this, we can compute the pathwise gradient as

$$\nabla_{\boldsymbol{\theta}}\mathcal{F}(\boldsymbol{\theta}) = \begin{pmatrix} \frac{\partial\mathcal{F}}{\partial\mu} \\ \frac{\partial\mathcal{F}}{\partial\sigma^2} \end{pmatrix} = \begin{pmatrix} 2(\sigma\boldsymbol{\varepsilon}+\mu+k) \\ \frac{\boldsymbol{\varepsilon}}{\sigma}(\sigma\boldsymbol{\varepsilon}+\mu+k) \end{pmatrix}.$$

Using the pathwise gradient estimator defined above, we can sample $\boldsymbol{\varepsilon}_i$ independently from the standard normal distribution to get the plot of gradient estimates included in Figure 3.

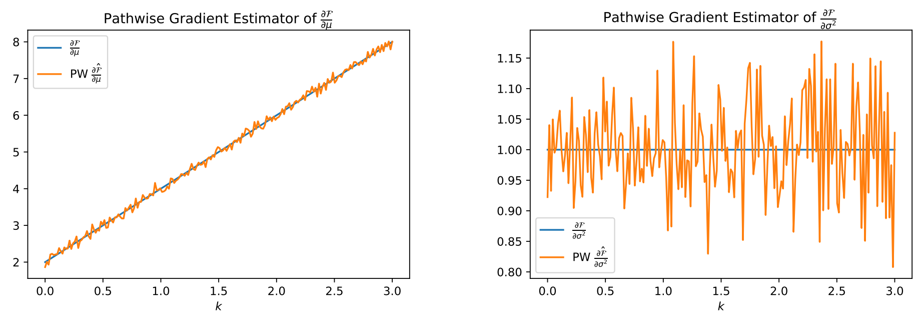

Figure 3 Pathwise gradient estimate of $\frac{\partial\mathcal{F}}{\partial\mu}$ where $p(\boldsymbol{x}|\boldsymbol{\theta}) \sim \mathcal{N}(1, 1)$ and $f(\boldsymbol{x}|\boldsymbol{\varphi}) = (x+k)^2$, $N = 1000$.

3.2 Hybrid Gradients

One interesting result we can glean from the previous two sections is that it is possible to use both the score function and pathwise estimator to compute the gradient. Specifically in cases where the base distribution p_{base} cannot be independently determined, we can instead write it as a function of the parameters $\boldsymbol{\theta}$. This gives the following form:

$$g(\boldsymbol{\varepsilon}|\boldsymbol{\theta}) \sim p(\boldsymbol{x}|\boldsymbol{\theta}), \text{ where } \boldsymbol{\varepsilon} \sim p_{base}(\boldsymbol{\varepsilon}|\boldsymbol{\theta}).$$

Then we can use the product rule when computing the gradient to get the following estimator:

$$\begin{aligned}
\nabla_{\boldsymbol{\theta}}\mathcal{F}(\boldsymbol{\theta}) &= \nabla_{\boldsymbol{\theta}} \int p(\boldsymbol{x}|\boldsymbol{\theta}) f(\boldsymbol{x}|\boldsymbol{\varphi}) d\boldsymbol{x} \\
&= \nabla_{\boldsymbol{\theta}} \int p_{base}(\boldsymbol{\varepsilon}|\boldsymbol{\theta}) f(g(\boldsymbol{\varepsilon}|\boldsymbol{\theta})|\boldsymbol{\varphi}) d\boldsymbol{\varepsilon} \\
&= \int p_{base}(\boldsymbol{\varepsilon}|\boldsymbol{\theta}) \nabla_{\boldsymbol{\theta}} f(g(\boldsymbol{\varepsilon}|\boldsymbol{\theta})|\boldsymbol{\varphi}) d\boldsymbol{\varepsilon} + \int f(g(\boldsymbol{\varepsilon}|\boldsymbol{\theta})|\boldsymbol{\varphi}) \nabla_{\boldsymbol{\theta}} p_{base}(\boldsymbol{\varepsilon}|\boldsymbol{\theta}) d\boldsymbol{\varepsilon} \\
&= \mathbb{E}_{p_{base}(\boldsymbol{\varepsilon}|\boldsymbol{\theta})}[\nabla_{\boldsymbol{\theta}} f(g(\boldsymbol{\varepsilon}|\boldsymbol{\theta})|\boldsymbol{\varphi})] + \mathbb{E}_{p_{base}(\boldsymbol{\varepsilon}|\boldsymbol{\theta})}[f(g(\boldsymbol{\varepsilon}|\boldsymbol{\theta})|\boldsymbol{\varphi}) \nabla_{\boldsymbol{\theta}} \ln(p_{base}(\boldsymbol{\varepsilon}|\boldsymbol{\theta}))] \\
&= \nabla_{\boldsymbol{\theta}}\mathcal{F}(\boldsymbol{\theta})_{PG} + \nabla_{\boldsymbol{\theta}}\mathcal{F}(\boldsymbol{\theta})_{SG}.
\end{aligned}$$

Although it should be avoided if possible, this form allows us to generalize the transformation method when sampling $\boldsymbol{\varepsilon}$ from a simpler distribution.

4 Measure-Valued Gradients

Also known as the weak-derivative method, measure-valued gradients are not commonly used in machine learning applications however the mathematics behind the derivation is quite elegant. Specifically, note that while the gradient $\nabla_{\boldsymbol{\theta}_i} p(\boldsymbol{x}|\boldsymbol{\theta})$ is *not* a density, the Hahn decomposition theorem states that we can construct this gradient as a linear combination of two signed measures as follows:

$$\nabla_{\boldsymbol{\theta}_i} p(\boldsymbol{x}|\boldsymbol{\theta}) = c_i^+ p^+(\boldsymbol{x}|\boldsymbol{\theta}) + c_i^- p^-(\boldsymbol{x}|\boldsymbol{\theta})$$

where $p^+(\boldsymbol{x}|\boldsymbol{\theta})$ and $p^-(\boldsymbol{x}|\boldsymbol{\theta})$ are densities corresponding to the "positive" and "negative" measure respectively, and c_i^+ and c_i^- are constants. One nice result is that $c_i^+ = -c_i^-$. In the below proof, we take advantage of the result shown in Section 2 that $\nabla_{\boldsymbol{\theta}} \int p(\boldsymbol{x}|\boldsymbol{\theta}) = \mathbf{0}$ to get

$$0 = \int \nabla_{\boldsymbol{\theta}_i} p(\boldsymbol{x}|\boldsymbol{\theta}) d\boldsymbol{x} = c_i^+ \int p^+(\boldsymbol{x}|\boldsymbol{\theta}) d\boldsymbol{x} + c_i^- \int p^-(\boldsymbol{x}|\boldsymbol{\theta}) d\boldsymbol{x} = c_i^+ + c_i^-.$$

Therefore we can write the gradient of p as

$$\nabla_{\boldsymbol{\theta}_i} p(\boldsymbol{x}|\boldsymbol{\theta}) = c_i \Big(p^+(\boldsymbol{x}|\boldsymbol{\theta}) - p^-(\boldsymbol{x}|\boldsymbol{\theta}) \Big).$$

Using this result, we can derive the measure-valued gradient estimator as

$$
\begin{aligned}
\nabla_{\boldsymbol{\theta}_i}\mathcal{F}(\boldsymbol{\theta}) &= \int \nabla_{\boldsymbol{\theta}_i} p(\boldsymbol{x}|\boldsymbol{\theta}) f(\boldsymbol{x}) d\boldsymbol{x} \\
&= c_i\Big(\int p^{+}(\boldsymbol{x}|\boldsymbol{\theta}) f(\boldsymbol{x}) d\boldsymbol{x} - \int p^{-}(\boldsymbol{x}|\boldsymbol{\theta}) f(\boldsymbol{x}) d\boldsymbol{x}\Big) \\
&= c_i\Big(\mathbb{E}_{p^{+}(\boldsymbol{x}|\boldsymbol{\theta})}[f(\boldsymbol{x})] - \mathbb{E}_{p^{-}(\boldsymbol{x}|\boldsymbol{\theta})}[f(\boldsymbol{x})]\Big) \\
&\approx c_i\left(\frac{1}{N}\sum_{i=1}^{N} f(\boldsymbol{x}'_i) - \frac{1}{N}\sum_{i=1}^{N} f(\boldsymbol{x}''_i)\right),
\end{aligned}
$$

Where N is the sample size, $\boldsymbol{x}'_i$ is independently drawn from p^+, and $\boldsymbol{x}''_i$ is independently drawn from p^-. To be precise, the tuple (c_i, p^+, p^-) is known as the measure-valued derivative, or weak derivative. One of the benefits of this method is that it does not assume any f is differentiable. While closed-form solutions exist for the measure-valued derivative under common distributions of p, this method can quickly become intractable for more complex distributions which explains the minimal use in modern-day machine learning techniques.

Although it will not be shown in this paper, it is simple to see that this method gives an unbiased estimator of the gradient. As for the variance of the estimator, we have the following:

$$
V\Big(\widehat{\nabla_{\boldsymbol{\theta}_i}\mathcal{F}(\boldsymbol{\theta})}_{MVG}\Big) = \frac{1}{N}V_{p^+}(f(\boldsymbol{x})) + \frac{1}{N}V_{p^-}(f(\boldsymbol{x})) - \frac{2}{N}Cov_{p^+,p^-}(f(\boldsymbol{x}'), f(\boldsymbol{x})).
$$

One common strategy to reduce the variance is to use common random numbers and applying the inverse transform method for each distribution. Doing so increases the covariance, thereby reducing the overall variance of the gradient estimate.

While a benefit of measure-valued gradients is that they do not assume differentiability of the cost function f, they *can* be more expensive to compute when compared to the other two methods covered in this paper. Note that the gradient is computed for each parameter in $\boldsymbol{\theta}$, and that the cost function is evaluated twice in each computation as opposed to once in the other methods. This leads to the computational cost being on the order of $\mathcal{O}(2NDL)$ where N is the Monte Carlo sample size, D is the dimension of the parameter vector $\boldsymbol{\theta}$, and L is the computational cost of evaluating f. For machine learning techniques with complex cost functions, this can quickly become intractable.

Note that this weak derivative is not a true gradient. It can be applied to discrete probability distributions which is a key benefit, however all implementation requires analytical computation of the positive and negative measures.

4.1 Example

Here, we will use measure-valued gradients to estimate the gradient for the same objective function in the previous two examples: the density function $p(\boldsymbol{x}|\boldsymbol{\theta}) \sim \mathcal{N}(\mu, \sigma^2)$ and cost function $f(\boldsymbol{x}|\boldsymbol{\varphi}) = (x + k)^2$. Then the constant, positive measure, and negative measure for estimating μ are

$$(c_\mu, p^+, p^-) = \left(\frac{1}{\sigma\sqrt{2\pi}}, \theta + \sigma\mathcal{W}\left(2, \frac{1}{2}\right), \theta - \sigma\mathcal{W}\left(2, \frac{1}{2}\right)\right),$$

where $\mathcal{W}(2, \frac{1}{2})$ is the Weibull distribution with parameters $\alpha = 2$ and $\beta = \frac{1}{2}$. This particular derivation is explained in more depth in [1]. Using this, we can compute the gradient of $\mathcal{F}$ with respect to μ for the $\mathcal{N}(1, 1)$ distribution to get the result in Figure 4.

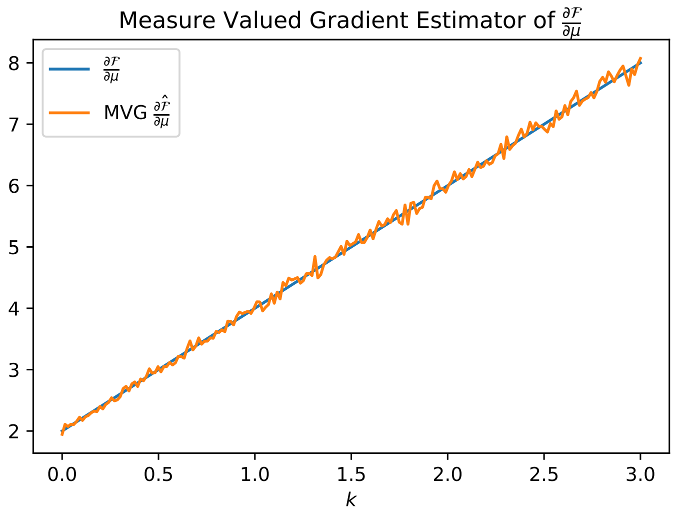

Figure 4 Measure-valued gradient estimate of $\frac{\partial\mathcal{F}}{\partial\mu}$ where $p(\boldsymbol{x}|\boldsymbol{\theta}) \sim \mathcal{N}(1, 1)$ and $f(\boldsymbol{x}|\boldsymbol{\varphi}) = (x + k)^2$, $N = 1000$.

5 Discussion and Applications

This paper covered three different approaches to Monte Carlo gradient estimation. This topic is of interest due to its prevalence in machine learning techniques, where we aim to minimize or maximize an objective function with respect to a distribution of unknown parameters and a given cost function. The score function method is popular due to ease of implementation and the minimal assumptions (specifically the freedom of the cost function f). Pathwise gradient methods are also commonly used in modern applications due to ease of sampling from simpler distributions, however it assumes both the cost function and path function are differentiable, potentially complicating computations. Finally, measure-valued gradients offer strong analytic properties pertaining to deeper results in measure theory, however difficulty of implementation and availability of simpler alternatives means that this approach has seen limited use in application.

While some variance reduction techniques have been discussed in this paper, there are many more approaches that exist to improve each of these methods. Unfortunately, they are beyond the scope of this paper, however the reader can refer to [5] for more detail and further references. From the simulated examples concerning gradient computation on a random variable distributed as $\mathcal{N}(1, 1)$ with cost function $f(\boldsymbol{x}|\boldsymbol{\varphi}) = (x + k)^2$, the mean square error (MSE) was computed for each of the estimates compared to the analytically computed derivative on $k \in [0, 3]$ and is displayed in Table 5.1:

Method	$\frac{\partial \mathcal{F}}{\partial \mu}$ MSE	$\frac{\partial \mathcal{F}}{\partial \sigma^2}$ MSE
Score Function Gradient	0.1659	0.1206
Score Function (Control Variates)	0.0503	0.0693
Pathwise Gradient	0.0084	0.0053
Measure-Valued Gradient	0.0066	N/A

Table 5.1 Sample MSE for $p(\boldsymbol{x}|\boldsymbol{\theta}) \sim \mathcal{N}(1, 1)$, $f(\boldsymbol{x}|\boldsymbol{\varphi}) = (x + k)^2$, $N = 1000$, and $k \in [0, 3]$.

5.1 Variational Inference

As has been recently discussed in this course, variational inference is an alternative to traditional Monte Carlo methods when approximating Bayesian posteriors. Specifically, it views computing the posterior as an optimization problem by maximizing the Evidence Lower Bound (ELBO) with respect to the KL divergence [3, 2]. In this case, the score function gradient estimator will have the form

$$\nabla_{\boldsymbol{\theta}} \text{ELBO} = \mathbb{E}_{q_{\boldsymbol{\theta}}(z)}\left[\nabla_{\boldsymbol{\theta}} \frac{\ln(p(x, z))}{\ln(q_{\boldsymbol{\theta}}(z))} + \frac{\ln(p(x, z))}{\ln(q_{\boldsymbol{\theta}}(z))} \nabla_{\boldsymbol{\theta}} \ln(q_{\boldsymbol{\theta}}(z))\right].$$

However, as we have discussed previously, this estimator suffers from relatively high variance. In a limited number of cases, it is more optimal to use pathwise gradient estimators if such a path function exists and satisfies the necessary assumptions. More examples and implementations are included in [3].

Mnih et. al. [4] discussed methods to reduce the variance of the gradient estimator for this problem (the algorithm for the implementation is included as supplementary material). Through their research, they aimed to maximize the likelihood function given in the form

$$\mathbb{E}_{Q(h_{1:K}|x)}\left[\ln\left(\frac{1}{K}\sum_{k=1}^{K} f(x, h_k)\right)\right]$$

which can be interpreted as a lower bound on the log-likelihood of a model given input data h_i for $i = 1, \cdots K$. To reduce variance, they introduced the concept of per-sample learning signals (likely inspired by leave-one-out cross validation). By assuming independence of the input data, they can use the law of total expectation to condition each observation on all remaining observations as follows:

$$\mathbb{E}_{Q(h_{1:K}|x)}\left[\widehat{L}(h_{1:K})\nabla_{\boldsymbol{\theta}} \ln(Q(h_j|x))\right] = \mathbb{E}_{Q(h_{-j}|x)}\left[\mathbb{E}_{Q(h_j|x)}\left[\widehat{L}(h_{1:K})\nabla_{\boldsymbol{\theta}} \ln(Q(h_j|x))|h_{-j}\right]\right],$$

where $\widehat{L}$ is a lower bound on the log-likelihood. By computing the "local-learning signal" for observation h, they developed the Variational Inference for Monte Carlo Objectives (VIMCO) estimator as

$$\nabla_{\boldsymbol{\theta}}\mathcal{L}_K(x) := \sum_{j=1}^{K} \widehat{L}(h_j|h_{-j})\nabla_{\boldsymbol{\theta}} \ln(Q(h_j|x)) + \sum_{j=1}^{K} \tilde{w}_j \nabla_{\boldsymbol{\theta}} \ln(f(x, h_j))$$

where $\tilde{w}_j$ are normalized weights defined as

$$\tilde{w}_j = \frac{f(x, h_j)}{\sum_{k=1}^{K} f(x, h_k)}.$$

A more detailed derivation is included in [4], with pseudocode as well. This method has been shown to have better convergence rates than previously discussed Monte Carlo variational inference gradient estimators, although it should be noted that this is still an area of active research.

5.2 Future Work

The subject of Monte Carlo gradient estimators is an active area of research. In particular, novel methods of variance reduction are investigated in order to improve machine learning techniques. While not covered in this paper, it is also possible to compute higher order derivatives of the objective function $\mathcal{F}$, although this estimation quickly becomes intractable and the variance bounds are undesirable.

It should be noted that choice of cost function can play a significant role in the convergence properties of each estimator. For this particular report, only a degree 2 polynomial was used in the cost function, and it can be seen that variance of the estimator increases with k. However exponential costs can have different properties as their parameters change, which could be explored in future investigations.

On a personal level, I am interested in implementing gradient estimation in an applied machine learning problem to better understand the benefits and drawbacks of methods such as the score function and pathwise gradients. I believe that this will enable me to better grasp the inner workings of stochastic optimization, and give me intuition when reading future papers regarding Monte Carlo techniques.

References

[1] B. Heidergott, F. J. Vázquez-Abad, and W. Volk-Makarewicz. Sensitivity estimation for gaussian systems. *European Journal of Operational Research*, 187(1):193–207, 2008. ISSN 0377-2217. doi: https://doi.org/10.1016/j.ejor.2007.04.004. URL https://www.sciencedirect.com/science/article/pii/S0377221707003177.

[2] M. D. Hoffman, D. M. Blei, C. Wang, and J. Paisley. Stochastic variational inference. *Journal of Machine Learning Research*, 14(40):1303–1347, 2013. URL http://jmlr.org/papers/v14/hoffman13a.html.

[3] M. Jankowiak and F. Obermeyer. Pathwise derivatives beyond the reparameterization trick. In J. Dy and A. Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 2235–2244. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/jankowiak18a.html.

[4] A. Mnih and D. Rezende. Variational inference for monte carlo objectives. In M. F. Balcan and K. Q. Weinberger, editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 2188–2196, New York, New York, USA, 20–22 Jun 2016. PMLR. URL https://proceedings.mlr.press/v48/mnihb16.html.

[5] S. Mohamed, M. Rosca, M. Figurnov, and A. Mnih. Monte carlo gradient estimation in machine learning. *Journal of Machine Learning Research*, 21(132):1–62, 2020. URL http://jmlr.org/papers/v21/19-346.html.