Control Variates in MCMC

Deepak Mahajan 170368
Aditya P Singh 16817052
Prateek Varshney 170494

1 Introduction

Control Variate is a common technique used in Monte Carlo methods to reduce variance of the estimators. These estimators are useful in optimization problems as well as in sampling problems. A common problem faced in both the fields is that computation of gradients over the whole dataset is usually expensive. One of the ways to tackle this is the use of noisy gradient estimators. In optimization literature, the use of stochastic gradients is very well known. The use of noisy estimators solves the problem of scalability but there is a trade off with the reliability on the output due to the high variance of the noisy gradient estimators. Use of control variates can, therefore, be potentially very helpful in reducing the variance in such cases [1].

Another field where the control variates are used is in the Monte Carlo estimators where the general aim is to estimate expectations of some function of the random variables with respect to the posterior distribution. In order to estimate the expectation of a function $g(\theta)$, the method involves generating the samples θ using MCMC algorithms and then averaging the value of the images of the samples which acts as an estimator of the expectation. Post-processing on the estimators are designed in order to improve the quality of estimators independent of the type of MCMC algorithm used to get the samples. Some of the well known post-processing methods include burn-in removal and thinning [2] . Burn-in removal involves removal of first few samples from the chain in order to improve the bias of the estimators. Thinning, on the other hand, involves retaining every k^{th} sample and discarding the rest in order to get samples that are not very correlated. We will discuss the use of control variates in MCMC algorithms as a post-processing step in order to reduce the estimator variance in section 2.2.

In this report, we will mainly focus on reproducing and extending two papers: 1. Wang et. al. (2013) [3] which discusses the theoretical aspects of control variates in optimization and empirically verifies the benefits, in 3.1, 2. Baker et. al. (2019) [4] which discusses control variates in the context of MCMC in section 2.2. Further, we will try to port their ideas to different settings such as MALA [5] and explore whether these ideas can be adopted and improved in these settings.

2 Background Theory

2.1 Control Variates in SGD

Consider a general optimization setting with a finite set of training points $\{\mathbf{x}_n\}_{n=1}^N$ with each $\mathbf{x}_n \in \mathbb{R}^D$. We want to maximise w.r.t $\mathbf{w} \in \mathbb{R}^D$ the objective:

$$\arg\max_{\mathbf{w}} \mathcal{L}(\mathbf{w}) = \arg\max_{\mathbf{w}} [\mathcal{R}(\mathbf{w}) + \frac{1}{N}\sum_{n=1}^{N} f(\mathbf{w}, \mathbf{x}_n)]$$

where $\mathcal{R}(\mathbf{w})$ is a regularization function. Unlike Gradient Based Methods, Stochastic Gradient (SG) methods use a noisy gradient estimated from random data samples:

$$g(\mathbf{w}, \mathbf{x}_n) = \nabla_{\mathbf{w}} \mathcal{R}(\mathbf{w}) + \nabla_{\mathbf{w}} f(\mathbf{w}, \mathbf{x}_n)$$
$$\mathbf{w}_{t+1} = \mathbf{w}_t + \rho_t g(\mathbf{w}, \mathbf{x}_n)$$

where $n \sim \text{Unif}(1, \ldots, N)$ at step t. For ease in notation, we use $g_n(\mathbf{w}) \coloneqq g(\mathbf{w}, \mathbf{x}_n)$. Note that to ensure the convergence of the stochastic gradient algorithm [6][7], we require:

$$\mathop{\mathbb{E}}_{n}[g_n(\mathbf{w})] = \nabla_{\mathbf{w}} \mathcal{L}(\mathbf{w})$$

However, if the variance of $g_n(\mathbf{w})$ is large, the algorithm can suffer from slow convergence. Introducing Control Variates aims to overcome this by constructing a new random vector which as the same expectation as the target expectation but has smaller variance. In our experiments, we use a random vector $h_n(\mathbf{w}) \in \mathbb{R}^D$ to reduce the variance of the stochastic gradient estimate as:

$$\tilde{g_n}(\mathbf{w}) = g_n(\mathbf{w}) - A^T(h_n(\mathbf{w}) - h(\mathbf{w}))$$

where A is a $D \times D$ matrix and $\mathop{\mathbb{E}}_{n}[h_n(\mathbf{w})] = h(\mathbf{w})$. Then clearly:

$$\begin{aligned}
\mathop{\mathbb{E}}_{n}[\tilde{g_n}(\mathbf{w})] &= \mathop{\mathbb{E}}_{n}[g_n(\mathbf{w})] - \mathop{\mathbb{E}}_{n}\left[A^T(h_n(\mathbf{w}) - h(\mathbf{w}))\right] \\
&= \mathop{\mathbb{E}}_{n}[g_n(\mathbf{w})] - A^T \mathop{\mathbb{E}}_{n}[(h_n(\mathbf{w}) - h(\mathbf{w}))] \\
&= \mathop{\mathbb{E}}_{n}[g_n(\mathbf{w})] - A^T(\mathop{\mathbb{E}}_{n}[h_n(\mathbf{w}] - h(\mathbf{w})) \\
&= \mathop{\mathbb{E}}_{n}[g_n(\mathbf{w})]
\end{aligned}$$

To reduce the variance of the noisy gradient, the trace of the covariance matrix of $\tilde{g_n}(\mathbf{w})$ must be necessarily small, since:

$$\begin{aligned}
\text{Var}_n[\tilde{g_n}(\mathbf{w}))] &\coloneqq \text{Cov}_n[\tilde{g_n}(\mathbf{w}), g_n(\mathbf{w})] \\
&= \text{Var}_n[g_n(\mathbf{w})] - (\text{Cov}_n[h_n(\mathbf{w}), g_n(\mathbf{w})] + \text{Cov}_n[g_n(\mathbf{w}), h_n(\mathbf{w})])A + A^T \text{Var}_n[\tilde{h_n}(\mathbf{w})]A
\end{aligned}$$

Setting A to be the minimizer of $\text{Tr}(\text{Var}_n[\tilde{g_n}(\mathbf{w})])$, i.e.:

$$\begin{aligned}
A^* &= \text{Tr}(\text{Var}_n[\tilde{g_n}(\mathbf{w})]) \\
&= (\text{Var}_n[h_n(\mathbf{w}))]^{-1}\big(\text{Cov}_n[g_n(\mathbf{w}), h_n(\mathbf{w})] + \text{Cov}_n[h_n(\mathbf{w}), g_n(\mathbf{w})]\big)/2
\end{aligned}$$

which is a function of $\mathbf{w}$.

Note that: In a first-order stochastic oracle model, we usually assume a constant σ such that for any estimate $\mathbf{w}$ in its domain [8][9]:

$$\mathop{\mathbb{E}}_{n}\left[\left\|g_n(\mathbf{w}) - \mathop{\mathbb{E}}_{n}[g_n(\mathbf{w})]\right\|_2^2\right] = \text{Tr}(\text{Var}_n[g_n(\mathbf{w}))]) \leq \sigma^2$$

it can be shown that:

$$\mathop{\mathbb{E}}_{n}\left[\left\|\tilde{g}_n(\mathbf{w}) - \mathop{\mathbb{E}}_{n}[\tilde{g}_n(\mathbf{w})]\right\|_2^2\right] \leq \mathop{\mathbb{E}}_{n}\left[\left\|g_n(\mathbf{w}) - \mathop{\mathbb{E}}_{n}[g_n(\mathbf{w})]\right\|_2^2\right]$$

i.e., it is possible to find a constant $\tau \leq \sigma$ such that $\mathop{\mathbb{E}}_{n}\left[\left\|\tilde{g}_n(\mathbf{w}) - \mathop{\mathbb{E}}_{n}[\tilde{g}_n(\mathbf{w})]\right\|_2^2\right] \leq \tau^2 \; \forall \mathbf{w}$. Therefore, we can improve the optimal convergence rate of SG methods from $\mathcal{O}(\sigma/\sqrt{t})$ to $\mathcal{O}(\tau/\sqrt{t})$ for convex problems; and from $\mathcal{O}(\sigma^2/(\mu t))$ to $\mathcal{O}(\tau^2/(\mu t))$ for strongly convex problems [1].

Also note that: Estimating A can be computationally expensive due to the computation of $\text{Var}_n[\tilde{h_n}(\mathbf{w})]$. One can therefore use a simpler surrogate to A, by reducing A to a single real number a. In this case, the optimal a is then simply given by:

$$a^* = \frac{\text{Tr}(\text{Cov}_n[g_n(\mathbf{w}), h_n(\mathbf{w})])}{\text{Tr}(\text{Var}_n(h_n(\mathbf{w})))} \tag{1}$$

which can be approximated by the sample covariance and variance from mini-batch samples while running the stochastic gradient algorithm. If we cannot obtain mini-batch samples, then we can employ strategies like moving average across iterations.[10][11]

2.2 Control Variabtes in MCMC:

The general aim of MCMC is to estimate expectations of functions $g(\theta)$, with respect to the posterior distribution π. Given MCMC samples $\{\theta^{(i)}\}_{i=1}^{M}$ from π, we can estimate $\mathop{\mathbb{E}}_{\theta\sim\pi}[g(\theta)]$ with an unbiased estimator ($\hat{\mu}_g$):

$$\mathop{\mathbb{E}}_{\theta\sim\pi}[g(\theta)] \approx \frac{1}{M}\sum_{i=1}^{M} g(\theta^{(i)}) = \hat{\mu}_g$$

As before, we can then introduce a function $h(\theta)$ s.t. $\mathop{\mathbb{E}}_{\theta\sim}[h(\theta)] = 0$ as:

$$\begin{aligned} \tilde{g}(\theta) &= g(\theta) + h(\theta) \\ \implies \mathop{\mathbb{E}}_{\theta\sim\pi}[\tilde{g}(\theta)] &= \mathop{\mathbb{E}}_{\theta\sim\pi}[g(\theta) + h(\theta)] \\ &= \mathop{\mathbb{E}}_{\theta\sim\pi}[g(\theta)] + \mathop{\mathbb{E}}_{\theta\sim\pi}[h(\theta)] \\ &= \mathop{\mathbb{E}}_{\theta\sim\pi}[g(\theta)] \end{aligned}$$

Therefore if $h(\cdot)$ is chosen so that it is negatively correlated with $g(\theta)$, then the variance of $\tilde{g}(\theta)$ will be reduced considerably.

Note: SGMCMC methods produce unbiased estimates of the log-posterior gradient, and so it follows that these gradient estimates can be applied as ZV control variates.

Mira et al. (2013) [12] proposed $h(\theta)$ to be

$$h(\theta) = \nabla Q(\theta) + \nabla Q(\theta)\cdot\mathbf{z}$$

where $Q(\theta)$ is a polynomial of θ, $\mathbf{z} = f(\theta)/2$ and ∇ refers to the Laplace operator $\frac{\partial^2}{\partial\theta_1^2},\ldots,\frac{\partial^2}{\partial\theta_d^2}$. Therefore, to get the best variance reduction, we have to optimize over the coefficients of the $Q(.)$. In practice, first or second degree polynomials $Q(\theta)$ often provide good variance reduction (Mira et al., 2013). In our experiments, we employ first degree polynomials, so $Q(\theta) = \mathbf{a}^T\theta$.

Since SGLD also calculates an unbiased estimate of $\nabla f(\theta)$, we replace $h(\theta)$ with the unbiased estimate:

$$\tilde{h}(\theta) = \Delta Q(\theta) + \nabla Q(\theta)\cdot\hat{\mathbf{z}}$$

where $\hat{\mathbf{z}} = \nabla\hat{f}(\theta)/2$. Note that $\hat{h}(\theta)$ is a valid control variate [13]. Using this and $Q(\theta)$ to be a linear polynomial, we have our SGLD-ZV estimate as:

$$\tilde{g}(\theta) = g(\theta) + \mathbf{a}^T\hat{\mathbf{z}}$$

where, the optimal value of $\mathbf{a}$ is given by:

$$\hat{\mathbf{a}} = \text{Var}^{-1}(\hat{\mathbf{z}})\text{Cov}(\hat{\mathbf{z}}, g(\theta))$$

.

Since SGLD already calculates all the necessary terms, the post-processing step can simply be applied once when the SGLD algorithm has finished, provided the full output plus gradient estimates are stored. Note that the efficiency of ZV control variates in reducing the variance of our MCMC sample is directly affected by using an estimate of the gradient rather than the truth.

Assumption. *Var*$[\psi(\theta)] < \infty$ *and Var*$[\hat{\psi(\theta)}] < \infty$. $\mathop{\mathbb{E}}_{\theta|x}\left[\|\nabla f_i(\theta)\|^2\right]$ *is bounded by some constant* $\sigma\ \ \forall i \in \{0,\ldots,N\} \in \mathbb{R}^D$.

Theorem. *Under the above assumption, define the optimal variance reduction for ZV control variates using the full gradient estimate to be* R*, and the optimal variance reduction using SGLD gradient estimates to be* $\hat{R}$*. Then we have that:*

$$\hat{R} \geq \frac{R}{1 + [\sigma(N+1)]^{-1}\mathop{\mathbb{E}}_{\theta|\mathbf{x}}\left[\mathop{\mathbb{E}}_{S}\left[\|\xi(\theta)\|^2\right]\right]}$$

An important consequence of the above thoerem is that if we use the standard SGLD gradient estimate, then the denominator of right hand side is $\mathcal{O}(n/N)$, so our variance reduction diminishes as N gets large. However, if we use the SGLD-CV estimate instead then under standard asymptotics, the denominator of is $\mathcal{O}(n)$, so the variance reduction does not diminish with increasing dataset size. Therefore for best results, especially for large N, we use the ZV post-processing step after running the SGLD-CV algorithm.

Note: There are some storage constraints for SGLD-ZV. This algorithm requires storing the full MCMC chain, as well as the gradient estimates at each iteration. So the storage cost is twice the storage cost of a standard SGMCMC run. However, in some high dimensional cases, the required SGMCMC test statistic is estimated on the fly using the most recent output of the chain and thus reducing the storage costs.

Note: Wile the original paper employed SGLD, in our experiments we will be using the SGD algorithm as the above properties hold for SGD as well.

3 Experiments

3.1 SG with Variance Reduction in Logistic Regression

Given a set of training examples $\{\mathbf{x}_n, y_n\}_{n=1}^N$, where $y_n \in \{1, -1\}$ indicates class labels, the likelihood of y_n is:

$$p(y_n|\mathbf{x}_n, \mathbf{w}) = \sigma(y_n \mathbf{w}^T \mathbf{x}_n)$$

where $\sigma(z) = 1/(1+\exp(z))$ is the logistic function [14]. The averaged log likelihood of the training data is then given by:

$$l(\mathbf{w}) = \frac{1}{N} \sum_{n=1}^{N} \left[y_n \mathbf{w}^T \mathbf{x}_n - \log(1 + \exp(y_n \mathbf{w}^T \mathbf{x}_n)) \right]$$

An SG algorithm employs the following noisy gradient:

$$g_n(\mathbf{w}) = y_n \mathbf{w} \sigma(-y_n \mathbf{w}^T \mathbf{x}_n)$$

We construct our control variate $h_n(\mathbf{w})$ for this logistic regression case as follows. We start with the first-order Taylor expansion around z for the sigmoid function:

$$\sigma(z) \approx \sigma(\hat{z})(1 + \sigma(-\hat{z})(z - \hat{z}))$$

Using this approximation for $\sigma(-y_n \mathbf{w}^T \mathbf{x}_n)$ in the previous equation, we can obtain our control variate. Note that for logistic regression, since data samples within each class are more likely to be similar, we can consider the two classes separately. Consider the positive data samples $\{\mathbf{x}_n^{(1)}, y_n = 1\}_{n=1}^{N^{(1)}}$ first. Let $z^{(1)} = -\mathbf{w}^T \mathbf{x}_n^{(1)}$, and define control variate $h_n^{(1)}(\mathbf{w})$ for $y_n = 1$ as:

$$\begin{aligned} h_n^{(1)}(\mathbf{w}) &:= \mathbf{x}_n^{(1)} \sigma(\hat{z}^{(1)})(1 + \sigma(-\hat{z}^{(1)})(z^{(1)} - \hat{z}^{(1)})) \\ &= \mathbf{x}_n^{(1)} \sigma(\hat{z}^{(1)})(1 + \sigma(-\hat{z}^{(1)})(-\mathbf{w}^T \mathbf{x}_n^{(1)} - \hat{z}^{(1)})) \end{aligned}$$

Its expectation given $y_n = 1$ can be computed in closed-form as:

$$\mathop{\mathbb{E}}_{n} \left[h_n^{(1)}(\mathbf{w}) | y_n = 1 \right] = \sigma(\hat{z}^{(1)}) \left(\bar{\mathbf{x}}^{(1)} \left(1 - \sigma(-\hat{z}^{(1)}) \hat{z^{(1)}} \right) - \sigma(-\hat{z}^{(1)}) \left(\text{Var}^{(1)}[\mathbf{x}_n^{(1)}] + \bar{\mathbf{x}}^{(1)} (\bar{\mathbf{x}}^{(1)})^T \right) \mathbf{w} \right)$$

where $\bar{\mathbf{x}}^{(1)}$ and $\text{Var}^{(1)}[\mathbf{x}_n]$ are the mean and variance of the input features for the positive examples.

For our experiments, we choose $\hat{z} = \mathbf{w}^T \bar{\mathbf{x}}^{(1)}$, which is the center of the positive examples. We can similarly derive the control variate $h_n^{(-1)}(\mathbf{w})$ for negative examples:

$$h_n^{(-1)}(\mathbf{w}) := -\mathbf{x}_n^{(-1)} \sigma(\hat{z}^{(-1)})(1 + \sigma(-\hat{z}^{(-1)})(\mathbf{w}^T \mathbf{x}_n^{(-1)} - \hat{z}^{(-1)}))$$

$$\mathop{\mathbb{E}}_{n} \left[h_n^{(-1)}(\mathbf{w}) | y_n = -1 \right] = -\sigma(\hat{z}^{(-1)}) \left(\bar{\mathbf{x}}^{(-1)} \left(1 - \sigma(-\hat{z}^{(-1)}) \hat{z}^{(-1)} \right) + \sigma(-\hat{z}^{(-1)}) \left(\text{Var}^{(-1)}[\mathbf{x}_n^{(-1)}] + \bar{\mathbf{x}}^{(-1)} (\bar{\mathbf{x}}^{(-1)})^T \right) \mathbf{w} \right)$$

where the superscript (-1) denotes negative samples (i.e. $y_n = -1$), $\hat{z}^{(-1)} = \mathbf{w}^T\bar{\mathbf{x}}^{(-1)}$, $\bar{\mathbf{x}}^{(-1)}$ and $\text{Var}^{(-1)}[\mathbf{x}_n^{(-1)}]$ are the mean and variance of the input features for the negative examples.

Given the random sample regardless its label, the expectation of the control variate is computed as:

$$\mathop{\mathbb{E}}_{n}[h_n(\mathbf{w})] = \frac{N^{(1)}}{N}\mathop{\mathbb{E}}_{n}\left[h_n^{(1)}(\mathbf{w})|y_n = 1\right] + \frac{N^{(-1)}}{N}\mathop{\mathbb{E}}_{n}\left[h_n^{(-1)}(\mathbf{w})|y_n = -1\right]$$

where $N^{(1)}$ and $N^{(-1)}$ are the number of positive and negative examples and $\frac{N^{(1)}}{N}$ is the probability of choosing a positive example from the training set (similarly for the negative samples). With Taylor approximation, we would expect our control variate is highly correlated with the noisy gradient.

Note that the expressions obtained above are for the completely stochastic regime. The analogous expressions for the batch case can be similarly obtained as :

$$g_B(\mathbf{w}) = \frac{1}{B}\sum_{b=1}^{B} y_b\mathbf{w}\sigma(-y_b\mathbf{w}^T\mathbf{x}_b)$$

$$h_B(\mathbf{w}) \coloneqq \frac{1}{B}\sum_{b=1}^{B} y_b\mathbf{x}_b\sigma(\hat{z}_B)(1+\sigma(\hat{z}_B)(-y_b\mathbf{w}^T\mathbf{x}_b - \hat{z}_B))$$

$$\mathop{\mathbb{E}}_{b}[h_b(\mathbf{w})] = \frac{B^{(1)}}{B}\mathop{\mathbb{E}}_{b}\left[h_b^{(1)}(\mathbf{w})|y_b = 1\right] + \frac{B^{(-1)}}{B}\mathop{\mathbb{E}}_{b}\left[h_b^{(-1)}(\mathbf{w})|y_b = -1\right]$$

where

$$\mathop{\mathbb{E}}_{b}\left[h_b^{(1)}(\mathbf{w})|y_b = 1\right] = \sigma(\hat{z}_B^{(1)})\left(\bar{\mathbf{x}}_B^{(1)}\left(1-\sigma(-\hat{z}_B^{(1)})\hat{z}_B^{(1)}\right) - \sigma(-\hat{z}_B^{(1)})\left(\text{Var}^{(1)}[\mathbf{x}_b^{(1)}] + \bar{\mathbf{x}}_B^{(1)}(\bar{\mathbf{x}}_B^{(1)})^T\right)\mathbf{w}\right)$$

$$\mathop{\mathbb{E}}_{b}\left[h_b^{(-1)}(\mathbf{w})|y_b = -1\right] = -\sigma(\hat{z}_B^{(-1)})\left(\bar{\mathbf{x}}_B^{(-1)}\left(1-\sigma(-\hat{z}_B^{(-1)})\hat{z}_B^{(-1)}\right) + \sigma(-\hat{z}_B^{(-1)})\left(\text{Var}^{(-1)}[\mathbf{x}_b^{(-1)}] + \bar{\mathbf{x}}_B^{(-1)}(\bar{\mathbf{x}}_B^{(-1)})^T\right)\mathbf{w}\right)$$

$$\hat{z}_B = \frac{B^{(1)}\hat{z}_B^{(1)} + B^{(-1)}\hat{z}_B^{(-1)}}{B}$$

$$\hat{z}_B^{(1)} = \mathbf{w}^T\bar{\mathbf{x}}_B^{(1)}$$

$$\hat{z}_B^{(-1)} = \mathbf{w}^T\bar{\mathbf{x}}_B^{(-1)}$$

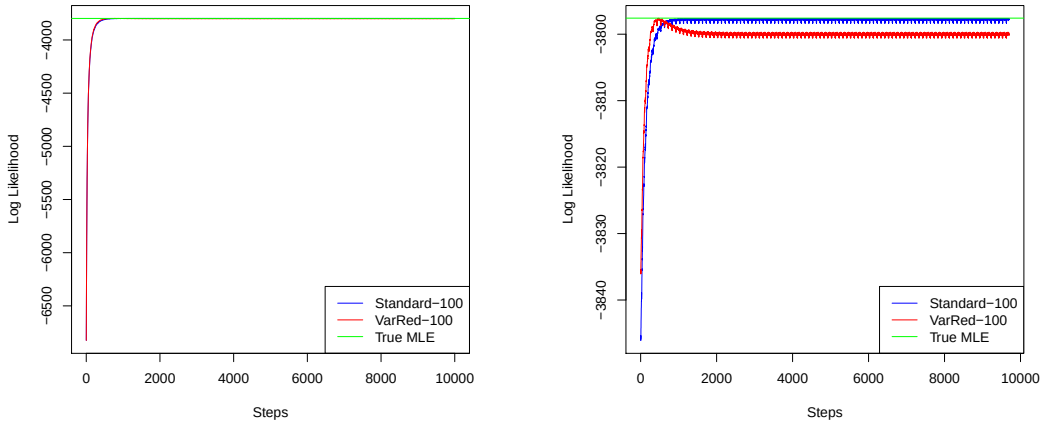
$$\bar{\mathbf{x}}_B^{(1)} = \frac{1}{B}\sum_{b=1}^{B} x_b^{(1)}$$

$$\bar{\mathbf{x}}_B^{(-1)} = \frac{1}{B}\sum_{b=1}^{B} x_b^{(-1)}$$

where 'b' is an iterator over the current batch $\{\mathbf{x}_b, y_b\}_{b=1}^{B}$, B is the batch size, $B^{(1)}$ and $B^{(-1)}$ are the number of positive and negative examples in the current batch, $\text{Var}^{(1)}[\mathbf{x}_b^{(1)}]$ and $\text{Var}^{(-1)}[\mathbf{x}_b^{(-1)}]$ variances of the input features for the positive and negative examples respectively in the current batch.

We evaluate our algorithm on stochastic gradient (SG) for logistic regression. For the standard SG algorithm, we also evaluated the version with averaged output (ASG), although we did not find it outperforms the standard SG algorithm much. We generated a random dataset (N = 10,000 and p = 5) to test our algorithm. We experimented with constant learning rates and explored $\rho_t \in \{0.1, 0.01, 0.001\}$. In our experiments, we did not notice any significant improvement in the quality of samples.

Results: The best result of variance reduction is obtained when $\rho_t = 0.1$ while the best case for the standard SG algorithm occurs when $\rho_t = 0.001$. In our experiments, we did not find any significant advantage of using these control variates. In fact, for our small datasets, the convergence of the VarRed algorithm was slower. Also, the original algorithm is extremely slow due to the calculation of generalised inverses in each step of the algorithm and this also leads to numerical instability. We substituted this with the expression in equation 1 to circumvent these issues.



(a) Plot in original scale

(b) Plot when zoomed in

Figure 1: Log likelihood comparison of the Standard and Variance Reduction algorithm

3.2 ZV-postprocessing in SG for Logistic Regression

Consider the logistic regression model as described in section 3.1. We employed a simple Stochastic Gradient Ascent algorithm to compute the regression coefficients β for this model and save all the iterates along with their mini-batch gradients. Following the discussion in section 2.2, we have the following algorithm for computing new estimates $\hat{\beta}$ from β.

Algorithm 1: Post-processing ZV

Input : The β estimates along with their mini-batch gradients, $\left\{\theta_k, \nabla \hat{f}(\theta_k)\right\}_{k=1}^{K}$

Set $\mathbf{z}_k = \frac{1}{2}\nabla \hat{f}(\theta_k)$;
Estimate $V_{\mathbf{z}} = \text{Var}[\mathbf{z}]$ and $C_{\mathbf{z}} = \text{Cov}(\theta, \mathbf{z})$;
$\hat{\mathbf{a}_j} = [V_{\mathbf{z}}]^{-1} C_{\mathbf{z}}$;
for *k = 1 to K* **do**
    $\hat{\theta}_k = \theta_k + \hat{\mathbf{a}}^T \mathbf{z}$;
end

We evaluated our algorithm on stochastic gradient descent (SGD) for logistic regression. We generated a random dataset (N = 1e6, p = 30) to test our algorithm. We pursued the version with averaged output (ASG) but the postprocessing did not perform well on those outputs and in fact, was considerably worse with an apparent bias in the final estimate. Therefore, we only tested the algorithm on standard SG where the results were promising.

Results: The results were overwhelmingly good, a sharp reduction in variance was observed which also lead to a better estimation of the mean β. The variability of the postprocessed estimates were about 43 times better and these estimates were also closer to the true values. We show the first the comparison for the first few coordinates in figure 2 and table 1.

True Value	Original Estimate	Postprocessed Estimate	SD Original Estimates (σ_o)	SD Postprocessed Estimates (σ_p)	Ratio ($\frac{\sigma_o}{\sigma_p}$)
0.087199	0.10916	0.087146	0.0713	0.00121	58.919
0.455167	0.46037	0.454854	0.0744	0.00101	73.755
-0.922060	-0.95603	-0.920953	0.0637	0.00129	49.538

Table 1: Comparison between the estimates with and without post-processing

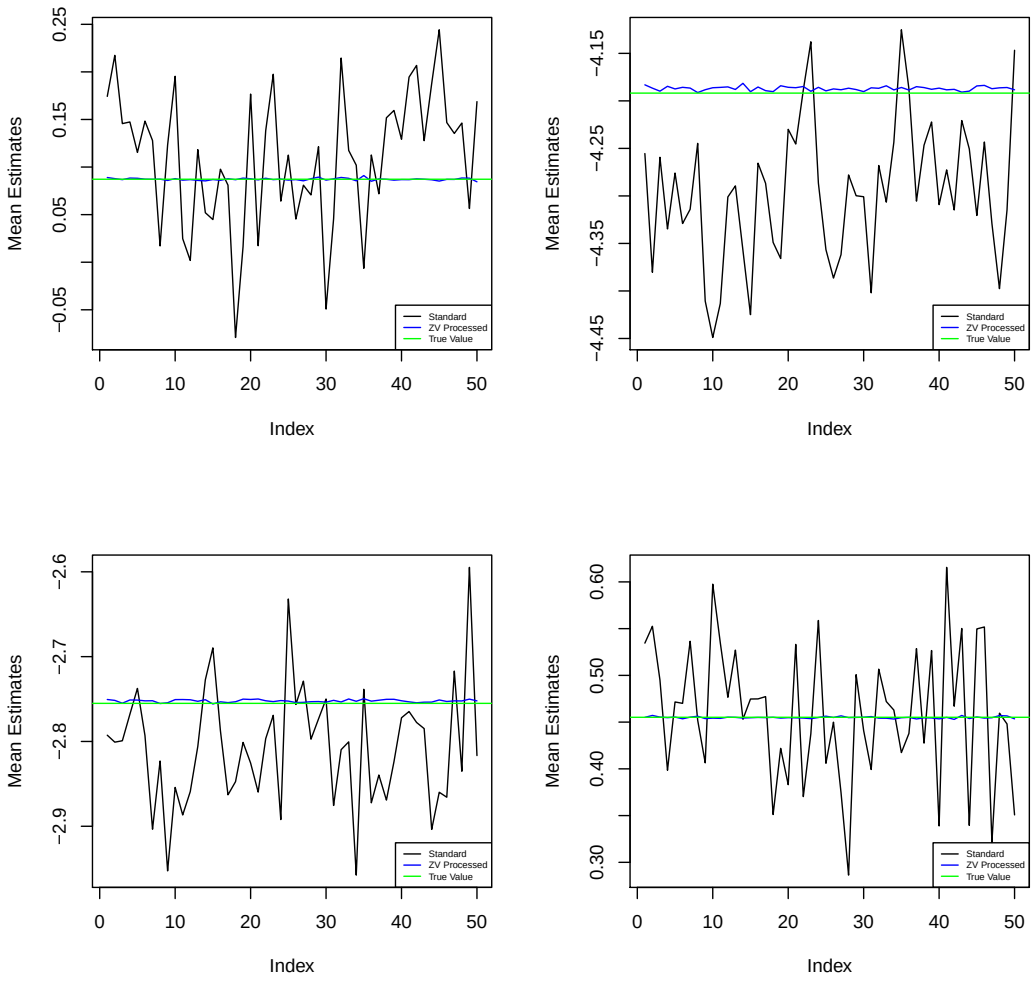


Figure 2: Comparison of standard SG estimates vs ZV processed estimates

3.3 ZV-postprocessing in MALA

We then tried to port the idea of ZV postprocessing to the MALA algorithm. We again considered the logistic regression regime as before and tried to estimate the mean β.

We generated a random dataset (N = 100, p = 5) to test our algorithm. The results in this case were not as promising as the SGD-ZV. There was little to no improvement in the quality of samples and there was a slight incrase in the variance of the samples on average. (Fig. 3)

Remark. *Note that we had to choose a considerably smaller dataset as MALA is quite slow in comparison to stochastic gradient algorithms.*

4 Analysis

4.1 Experiment 1: SGD vs SGD-CV

The results of our experiments were contradictory to the claims of the original paper. While the authors claimed that that their method "consistently" performed better than the standard stochastic variational inference, we found that the introducing covariates led to a faster convergence rate to the

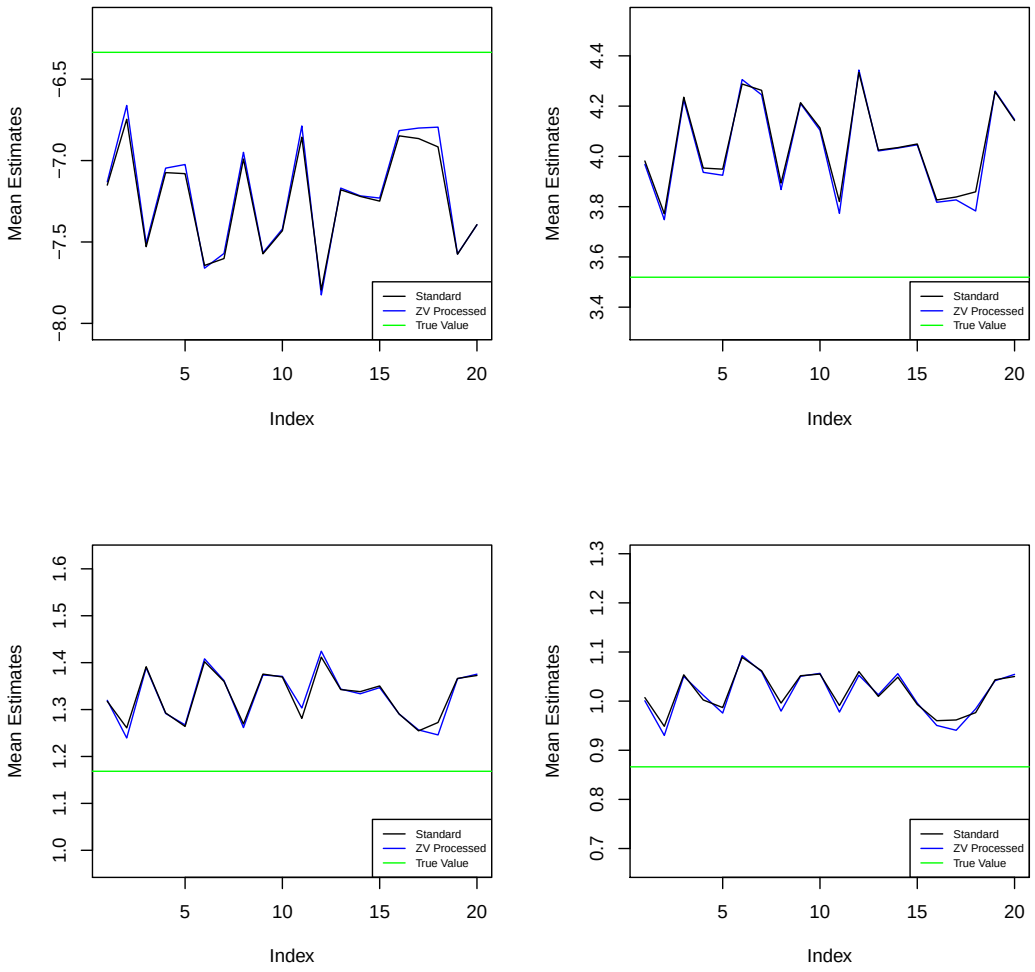


Figure 3: Comparison of standard MALA estimates vs ZV processed estimates

maxima but the weight vector jumped out of the maxima and stabilised in a local mode as indicated in the figure 1 as compared against the standard Stochastic Gradient Method. We found this to be the case even when running on batch size equal to the entire dataset. One possible explanation for the same can be that the original paper had experimented on a very large "real" dataset while our experiments were run on synthesised dataset. This can be corrected by fine tuning and employing an adaptive learning rate as a function of the log likelihood or methods like early stopping (e.g., restoring the current value if the next few iterations lead to continuous decrements in log likelihood). Since the authors do not mention any such additional information about their implementation, we cannot directly claim that SGD-CV is better than CV on the original experiment settings.

Interestingly, we did observe that introducing CV enabled us to use a larger learning rate as compared to the standard algorithm, without sacrificing much performance.

4.2 Experiment 2: SGD vs SGD-ZV Post Processing

The Zero Variates post processing for the SG works as intended. We observed massive reductions in variance in our experiments. However, the postprocessing does not work with all kinds of SG algorithms. We tried to run the postprocess on the averaged output version (ASG) and there were little to no improvements in the quality of samples and in fact, it introduced some bias into the samples.

4.3 Experiment 3: MALA vs MALA-ZV Post Processing

We tried to implement the idea of ZV-postprocessing to other MCMC algorithms, in particular, MALA. We chose MALA because it requires gradients and hence, those could be directly used for our purposes. However, the postprocessed outputs did not show much promise. They did not seem to introduce any bias to the samples but the variability of the post-processed samples were higher. We suspect that this is due to the use of unbiased estimates of the gradient used in the algorithm or due to the fact that a lot of samples used in calculating the mean of β are wasteful and thus, we could not converge close enough to the true mean in 1,00,000 samples we generated.

5 Conclusion

In this report, we have provided empirical evidence for the variance reduction of ZV post-processing control variates. As we need not change the stochastic gradient algorithms already in place to implement this algorithm, this method can be pushed into production without bearing any significant costs, with a promise of superior results. We also saw that this method cannot be used in a MALA setting as it is, but exploring certain variations of this method, to work in these settings, can be a possible line of research.

Further, we saw that SG-CV algorithm does not consistently perform well under all circumstances and some research has to be done into what kind of conditions are necessary to affirm its superiority over the standard SG algorithm keeping in mind that it is already quite slow in comparison to the standard algorithm.

References

[1] Chong Wang, Xi Chen, Alexander Smola, and Eric P Xing. Variance reduction for stochastic gradient optimization. 2013.

[2] Leah F South, Marina Riabiz, Onur Teymur, Chris Oates, et al. Post-processing of mcmc. *arXiv preprint arXiv:2103.16048*, 2021.

[3] Chong Wang, Xi Chen, Alexander Smola, and Eric P Xing. Variance reduction for stochastic gradient optimization, Jun 2018.

[4] Jack Baker, Paul Fearnhead, Emily Fox, and Christopher Nemeth. Control variates for stochastic gradient mcmc. *Statistics and Computing*, 29, 05 2019.

[5] Mark Girolami and Ben Calderhead. Riemann manifold langevin and hamiltonian monte carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 73:123 – 214, 03 2011.

[6] James C. Spall. *Introduction to Stochastic Search and Optimization*. John Wiley Sons, Inc., USA, 1 edition, 2003.

[7] Arkadi Nemirovski, Anatoli Juditsky, Guanghui Lan, and And Shapiro. Robust stochastic approximation approach to stochastic programming. *Society for Industrial and Applied Mathematics*, 19:1574–1609, 01 2009.

[8] Guanghui Lan. An optimal method for stochastic composite optimization. *Mathematical Programming*, 133:1–33, 06 2012.

[9] Xi Chen, Qihang Lin, and Javier Pena. Optimal regularized dual averaging methods for stochastic optimization. 09 2018.

[10] Tom Schaul, Sixin Zhang, and Yann LeCun. No more pesky learning rates, 2013.

[11] Rajesh Ranganath, Chong Wang, Blei David, and Eric Xing. An adaptive learning rate for stochastic variational inference. In Sanjoy Dasgupta and David McAllester, editors, *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pages 298–306, Atlanta, Georgia, USA, 17–19 Jun 2013. PMLR.

[12] Antonietta Mira, Reza Solgi, and Daniele Imparato. Zero variance markov chain monte carlo for bayesian estimators. *Statistics and Computing*, 23(5):653–662, Jul 2012.

[13] Nial Friel, Antonietta Mira, and Chris. J. Oates. Exploiting multi-core architectures for reduced-variance estimation with intractable likelihoods, 2015.

[14] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2006.