A Rule for Gradient Estimator Selection, with an Application to Variational Inference

Tomas Geffner
University of Massachusetts, Amherst

Justin Domke
University of Massachusetts, Amherst

Abstract

Stochastic gradient descent (SGD) is the workhorse of modern machine learning. Sometimes, there are many different potential gradient estimators that can be used. When so, choosing the one with the best tradeoff between cost and variance is important. This paper analyzes the convergence rates of SGD as a function of time, rather than iterations. This results in a simple rule to select the estimator that leads to the best optimization convergence guarantee. This choice is the same for different variants of SGD, and with different assumptions about the objective (e.g. convexity or smoothness). Inspired by this principle, we propose a technique to automatically select an estimator when a finite pool of estimators is given. Then, we extend to infinite pools of estimators, where each one is indexed by control variate weights. Empirically, automatically choosing an estimator performs comparably to the best estimator chosen with hindsight.

1 Introduction

Convergence guarantees for stochastic optimization algorithms depend on structural properties of the objective, such as convexity and smoothness, the variance of the gradient estimates, and the number of iterations performed. Naturally, lower variance and more iterations result in better guarantees (Bottou et al., 2018). Given a fixed total optimization time, fewer iterations are possible with a slower estimator. Therefore, an estimator with low variance and low cost is ideal.

This paper is inspired by stochastic gradient varia-

tional inference (SGVI), where there are multiple gradient estimators with varying costs and variances. Estimators may be obtained using the reparameterization trick (Kingma and Welling, 2013), the score function method (Williams, 1992), or other techniques (Titsias and Lázaro-Gredilla, 2015; Ruiz et al., 2016). Also, control variates can be added to the estimator to reduce variance (Miller et al., 2017; Mnih and Gregor, 2014; Paisley et al., 2012; Geffner and Domke, 2018).[1]

The cost and variance of an estimator significantly affects optimization convergence speed. Different estimators lead to different performances, and the optimal estimator is often situation-dependent. As an example, Fig. 1 shows the results of running SGVI on several models (described in Sec. 5.1). We compare three common gradient estimators:

- (Rep) Plain reparameterization estimator,
- (Miller) Estimator proposed by Miller et al. (2017); uses a Taylor expansion control variate,
- (STL) "Sticking-the-landing" estimator; avoids computing certain terms (Roeder et al., 2017).

(Detailed descriptions of the estimators are in Section 3.4.) Table 1 gives the time costs of each estimator. There is no universal best estimator. Different estimators perform better in different situations. An estimator's variance depends on the model, dataset, and current parameters, while the time cost also depends on the implementation and computational platform. Rather than rely on the user to navigate these tradeoffs, we propose that estimator selection could be done adaptively. This paper investigates how, given a pool of gradient estimators, *automatically* choose one to get the best convergence guarantee.

Section 2 introduces some background on SGVI and control variates. Section 3 studies cost-variance tradeoffs by analyzing the convergence rates of several variants of SGD. We do this with or without the structural

---

[1]In fact, it has been shown that the problem of selecting a gradient estimator for variational inference can be phrased as choosing a set of control variates to use along with their weights (Geffner and Domke, 2018).

Model	Estimator		
	Rep	Miller	STL
BNN-A	8.2	25	11
BNN-B	7.1	21	9.5
Log Reg (a1a)	6.9	120	12
Hier Poisson	2.6	7.9	3.8

Table 1: **Different estimators have different costs.** Time costs (in ms) for estimators in Fig. 1.

properties of *convexity*, *smoothness*, or *strong convexity*. We observe what we call the "G^2T principle". This states that, given a set of candidate estimators, the best convergence guarantee is obtained by the estimator for which G^2T is minimum, where $G^2 \geq \mathbb{E}\,\|g\|^2$ denotes a bound on an estimator's expected squared norm, and T its time cost. Put another way, estimators with lower G^2T lead to better convergence guarantees. This is true regardless of the objective's structural properties (Section 3.2).

In theory, this principle could be directly used if a pool of estimators with known values for G and T was given. In practice, these values are rarely known a priori. In Section 3.3 we propose an estimator selection algorithm that uses estimates of these quantities instead of the true values.

Finally, motivated by SGVI where control variates play an important role, we look at the problem of control variate selection. If used with the right weights, control variates lead to a reduction in the estimator's variance. However, they carry an extra computational cost, which may or may not be worth paying depending on the situation. In Section 4 we consider the case where there is an infinite pool of estimators, each indexed by a set of control variate weights. Selecting an estimator involves selecting what control variates to use along with their weights. We show that the problem of minimizing the estimated G^2T values over this infinite set of estimators can be reduced to a small mixed integer quadratically constrained program (MIQCP) and then solved with off the shelf methods. Section 5 gives an experimental evaluation.

1.1 Contributions

G^2T principle: Given a set of gradient estimators with different G^2 and T values, the best convergence guarantee is achieved by the estimator with minimum G^2T. The same estimator is optimal *regardless* of structural properties of the objective and with different variants of SGD (Section 3.2).

Gradient estimator selection: Given a finite set of gradient estimators, we propose an algorithm to
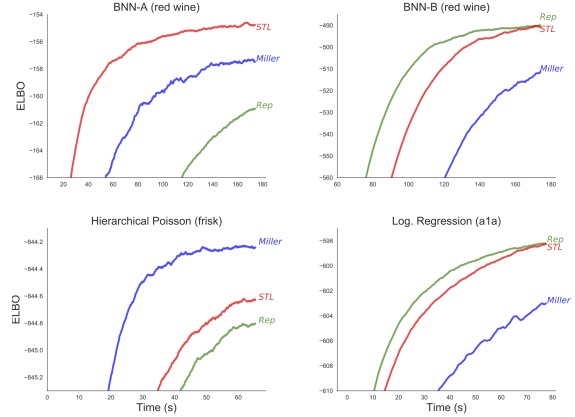

Figure 1: **Different estimators are better in different situations.** SGVI on four models. Each plot compares three different gradient estimators, described in the text. More details on the optimization algorithm and models are in Section 5.

automatically select one using empirical estimates of G^2 and T. (Algorithm 1, Section 3.3).

Control variate selection via MIQCP: Given an infinite set of estimators indexed by control variate weights, we propose an algorithm to automatically select a gradient estimator using empirical estimates of G^2 and T. This is based on a reduction to a small MIQCP, which can be solved efficiently in practice. (Algorithm 2, Section 4).

Empirical validation We apply the automatic estimator selection techniques to Bayesian logistic regression, hierarchical regression, and Bayesian neural network models. Empirically, automatic estimator selection performs similarly to choosing the best estimator with hindsight with finite pools of estimators (Section 3.4) and similarly with infinite pools of estimators (Section 5).

Notation: We use $g(w, \xi)$, where ξ is a random variable, to denote an unbiased estimator of target's gradient, $G^2(g)$ to denote a bound on g's expected squared norm, $\mathbb{E}_\xi\,\|g(w, \xi)\|^2 \leq G^2(g)\ \forall w$, and $T(g)$ to denote the computational cost of computing estimator $g(w, \xi)$, measured in seconds. We drop the dependencies of G^2 and T on g when clear from context.

2 Preliminaries

2.1 Stochastic Gradient Variational Inference

Given a model $p(x, z)$, where x is observed data and z are latent variables, the goal of variational inference (VI) is to find parameters w to approximate $p(z|x)$

with a simpler distribution $q_w(z)$. Since

$$\log p(x) = \underbrace{\mathop{\mathbb{E}}_{Z \sim q_w(z)} \left[\log \frac{p(x, Z)}{q_w(Z)} \right]}_{\text{ELBO}(w)} + \text{KL}(q_w(Z) || p(Z|x)),$$

minimizing the KL-divergence is equivalent to maximizing the $\text{ELBO}(w)$ (Evidence Lower BOund).

When p and q are complex, it is typically necessary to optimize the ELBO using stochastic methods (Mnih and Gregor, 2014; Paisley et al., 2012; Ranganath et al., 2014; Titsias and Lázaro-Gredilla, 2014; Mnih and Rezende, 2016). These require an unbiased estimator of $\nabla_w \text{ELBO}(w)$, which can be expressed as

$$\nabla_w \text{ELBO} = \nabla_w \mathop{\mathbb{E}}_{q_w(z)} \log p(x, Z) + \nabla_w \mathcal{H}(q_w) \quad (1)$$

$$= \nabla_w \mathop{\mathbb{E}}_{q_w(z)} \log p(x, Z)$$

$$- \left[\nabla_w \mathop{\mathbb{E}}_{q_w(z)} \log q_v(Z) + \nabla_w \mathop{\mathbb{E}}_{q_v(z)} \log q_w(Z) \right]_{v=w}. \quad (2)$$

Estimates for this gradient can be obtained through reparameterization (Kingma and Welling, 2013; Titsias and Lázaro-Gredilla, 2014; Rezende et al., 2014) and the score function method (Williams, 1992), among other techniques (Ruiz et al., 2016; Titsias and Lázaro-Gredilla, 2015; Schulman et al., 2015). All of these express the gradient as an expectation and approximate it using Monte Carlo sampling. For example, the reparameterization trick is based on a function $\mathcal{T}_w$ that transforms $\xi \sim q_0$ into $Z = \mathcal{T}_w(\xi) \sim q_w$. Then, it approximates terms like $\nabla_w \mathbb{E}_{q_w(z)} \log p(x, Z)$ with the unbiased estimator $\nabla_w \log p(x, \mathcal{T}_w(\xi))$.

2.2 Control Variates for Variational Inference

Control variates (Owen, 2013) are random variables with mean zero that can be used to reduce the variance of an estimator. They are widely used in SGVI (Grathwohl et al., 2018; Miller et al., 2017; Paisley et al., 2012; Ranganath et al., 2014; Tucker et al., 2017). Take an estimator, which is some function g_{base} such that $\mathbb{E}_\xi g_{\text{base}}(w, \xi) = \nabla_w \text{ELBO}(w)$. Take also the control variate c such that $\mathbb{E}_\xi c(w, \xi) = 0$. The final gradient estimator is $g(w, \xi) = g_{\text{base}}(w, \xi) + a\, c(w, \xi)$, where a is a scalar. When multiple control variates $c_1, ..., c_J$ are available, the final gradient estimator is

$$g(w, \xi) = g_{\text{base}}(w, \xi) + \sum_{i=1}^{J} a_i\, c_i(w, \xi). \quad (3)$$

Equivalently, we could write $g = g_{\text{base}} + Ca$, where $C(w, \xi)$ is a matrix with control variate c_i as its i-th column, and a is a vector containing the weights a_i.

As a concrete example, Miller et al. (2017) proposed a control variate to reduce the variance of the reparameterization estimator $g_{\text{base}}(w, \xi) = \nabla_w \log p(x, \mathcal{T}_w(\xi))$.

Given a second-order Taylor approximation $u(z) \approx \log p(x, z)$, they suggest the control variate $c(w, \xi) = \nabla_w \mathbb{E}_{q_w} u(Z) - \nabla_w u(\mathcal{T}_w(\xi))$. Since u is quadratic, this can be computed exactly when q_w is Gaussian. If the Taylor expansion is accurate, c will greatly reduce the estimator's variance.

Other control variates proposed can be built using upper/lower bounds of $\log p(x, z)$ (Paisley et al., 2012), baselines (Mnih and Gregor, 2014), or the difference between two unbiased estimates of a term (expectation of some function) (Geffner and Domke, 2018; Grathwohl et al., 2018; Tucker et al., 2017), among others.

3 The G²T principle

3.1 Stochastic Optimization Review

Many different first order stochastic optimization algorithms can be used to minimize an objective function. The simplest one, SGD, requires an initial guess w_0, a step-size η_k, and an unbiased estimator of the objective's gradient $g(w_k, \xi_k)$. It uses the update rule

$$w_{k+1} = w_k - \eta_k\, g(w_k, \xi_k). \quad (4)$$

Table 2 shows guarantees for different variants of SGD. These usually require two things: a bound on $\mathbb{E} \|g(w, \xi)\|^2$, and some structural assumption about the objective (the most common being convexity, λ-strong convexity, and L-smoothness).

It will be useful for the analysis in the following section to represent each of the guarantees in Table 2 as a function $\Theta(\lambda, L, C, K, G^2)$, where λ is the strong-convexity parameter (or 0), L is the smoothness parameter (or ∞), C is a boolean representing whether the problem is convex, K is the number of iterations, and G^2 represents the bound on the estimator's expected squared norm. For example, if the target is λ-strongly convex, then $\Theta(\lambda, L, C, K, G^2) = \frac{4}{\lambda^2} \frac{G^2}{K}$ (second row of Table 2). All bounds presented in Table 2 can be written as

$$\Theta(\lambda, L, C, K, G^2) = \alpha(\lambda, L, C) \left(\frac{G^2}{K} \right)^p, \quad (5)$$

where $p > 0$ and α depends on the properties of the target and initialization, but *not on the gradient estimator used.*

3.2 The G²T principle

Given a set of gradient estimators with varying costs and variances, our goal is to find the one that gives the best convergence guarantee for optimization algorithms. Given a total optimization time T_{opt}, an estimator g with cost $T(g)$ can perform $K \approx T_{\text{opt}}/T(g)$

Assumptions Needed			Algorithm	Guarantee
Convex	λ-strongly convex	L-smooth		
✓	✓	✓	SGD, $\eta_k = 1/(\lambda k)$ [1]	$\mathbb{E}\, F(w_K) - F(w^*) \leq \frac{2L}{\lambda^2}\frac{G^2}{K}$
✓	✓	–	SGD, $\eta_k = 1/(\lambda k)$ [1]	$\mathbb{E}\,\|w_K - w^*\|^2 \leq \frac{4}{\lambda^2}\frac{G^2}{K}$
✓	–	–	SGD, $\eta_k = \frac{D_w}{G\sqrt{K}}$ [2]	$\mathbb{E}\, F(\bar{w}) - F(w^*) \leq D_w \frac{G}{\sqrt{K}}$
–	–	✓	SGD, $\eta_k = \sqrt{\frac{2D_f}{LKG^2}}$ [3]	$\mathbb{E}\,\frac{1}{K}\sum_{i=1}^{K}\|\nabla F(w_i)\|^2 \leq \sqrt{LD_f}\frac{G}{\sqrt{K}}$
–	–	✓	SGD+Momentum (β) [4] $\eta_k = \sqrt{\frac{2D_f(1-\beta)^4}{(\beta^2+(1-\beta)^2)KLG^2}}$	$\mathbb{E}\,\frac{1}{K}\sum_{i=1}^{K}\|\nabla F(w_i)\|^2 \leq$ $\sqrt{\frac{8D_f L(\beta^2+(1-\beta)^2)}{(1-\beta)^2}}\frac{G}{\sqrt{K}}$
–	–	✓	SGD+Nesterov (β) [4] $\eta_k = \sqrt{\frac{2D_f(1-\beta)^4}{(\beta^4+(1-\beta)^2)KLG^2}}$	$\mathbb{E}\,\frac{1}{K}\sum_{i=1}^{K}\|\nabla F(w_i)\|^2 \leq$ $\sqrt{\frac{8D_f L(\beta^4+(1-\beta)^2)}{(1-\beta)^2}}\frac{G}{\sqrt{K}}$

Table 2: Convergence rates for stochastic optimization algorithms under different assumptions about the objective. Constants in the results are $D_f = F(w_0) - F(w^*)$ and $D_w = \|w_0 - w^*\|$. K is the number of optimization steps, and G^2 is such that $\mathbb{E}_\xi \|g(w,\xi)\|^2 \leq G^2 \ \forall w$. The learning rates shown are the optimal ones. [1]: Rakhlin et al. (2012), [2]: Nemirovski et al. (2009), [3]: Bottou et al. (2018), [4]: Yang et al. (2016).

iterations. Inserting this value for K in eq. (5) results in a convergence guarantee of

$$\Theta(\lambda, L, C, K, G^2(g)) = \frac{\alpha(\lambda,L,C)}{T_{\text{opt}}^p}\left(G^2(g)T(g)\right)^p. \quad (6)$$

In equation (6) the only dependence of $\Theta(\lambda, L, C, K, G^2)$ on the estimator used g comes through the $(G^2(g)T(g))^p$ factor. Thus, the best convergence guarantee is achieved by using the estimator with the smallest value of G^2T, *regardless* of the objective's properties (λ, L, C) and total optimization time (T_{opt}). We call this the "G^2T principle".

G²T principle. Given

- A target function characterized by (λ, L, C), which can be convex, strongly convex, strongly convex and smooth, or just smooth (non-convex);
- An optimization time budget T_{opt};
- A pool of gradient estimators.

Then, *for any* $(\lambda, L, C, T_{\text{opt}})$, provided the appropriate algorithm is used, the best convergence guarantee is obtained using the estimator g with minimum G^2T. That is, given two estimators g_1 and g_2, $K_1 = T_{\text{opt}}/T(g_1)$ and $K_2 = T_{\text{opt}}/T(g_2)$, we have

$$\Theta\left(\lambda, L, C, K_1, G^2(g_1)\right) < \Theta\left(\lambda, L, C, K_2, G^2(g_2)\right)$$

$$\iff G^2(g_1)T(g_1) < G^2(g_2)T(g_2),$$

regardless of the values of λ, L, C and T_{opt}.

3.3 G²T for Gradient Estimator Selection

This section presents a gradient estimator selection algorithm based on the G^2T principle. Given a pool of estimators with known G^2 and T, the one with minimum G^2T should be used. In practice, G^2 and T are typically not known. We propose to use estimates.

Assuming that the cost of an estimator $g(w,\xi)$ is independent of w, an estimate $\hat{T}(g)$ of $T(g)$ can be obtained for each $g \in \mathcal{G}$ through a single initial profiling phase.

Dealing with $G^2(g)$ is more involved. Convergence guarantees typically assume that $\mathbb{E}\,\|g(w,\xi)\|^2 \leq G^2(g)$ for *all* w. Often (e.g. when w is unbounded) this is not true for any finite $G^2(g)$. Even if a finite $G^2(g)$ exists, the bound may be conservative. In practice, optimization starting from w_0 will only visit a restricted part of the parameter space, $\mathcal{W}_0$. In that case, it is sufficient to bound $\mathbb{E}\,\|g(w,\xi)\|^2$ for this set of w that may actually be encountered. Since $\mathbb{E}\,\|g(w,\xi)\|^2$ tends to decrease as optimization proceeds, we propose to use $G^2(g, w_0) = \mathbb{E}\,\|g(w_0,\xi)\|^2$ as an effective upper bound for $\mathbb{E}\,\|g(w,\xi)\|^2$ when $w \in \mathcal{W}_0$. With this effective upper bound, we select an estimator by finding the one with minimum $G^2(g, w_0)T(g)$.

After k optimization steps, $G^2(g, w_k)$ will typically be less than $G^2(g, w_0)$. Starting from the current solution w_k, only a smaller part of the parameter space will be visited in the future, meaning a stronger bound for future gradients may be possible. We propose to use the new effective upper bound $G^2(g, w_k) = \mathbb{E}\,\|g(w_k,\xi)\|^2$, and to select the estimator g that now

minimizes $G^2(g, w_k)T(g)$. We perform this update only a few times throughout training [2]. Finally, since $G^2(g, w_k) = \mathbb{E} \|g(w_k, \xi)\|^2$ cannot be computed in closed form, we use a Monte Carlo estimate, $\hat{G}^2(g, w_k)$. The final algorithm is summarized in Algorithm 1.

Algorithm 1 SGD with minimum $\hat{G}^2\hat{T}$ estimator.

Require: Set of estimators, $\mathcal{G}$.
Require: Total opt time and times to re-select estimator.
Require: Number of MC samples M.
    For all $g \in \mathcal{G}$ measure time $\hat{T}(g)$.
    for $k = 1, 2, \cdots$ **do**
        if time to re-select estimator **then**
$$\forall g \in \mathcal{G}, \text{ set } \hat{G}^2(g, w_k) = \frac{1}{M} \sum_{m=1}^{M} \|g(w_k, \xi_{km})\|^2.$$
$$g \leftarrow \arg\min_{g \in \mathcal{G}} \hat{G}^2(g, w_k) \times \hat{T}(g).$$
        end if
        $w_{k+1} = w_k - \eta_k \ g(w_k, \xi_k)$
    end for

3.4 Empirical Validation of the G²T principle

This section presents an empirical validation of Algorithm 1. We consider the same setting as in Fig. 1. We ran SGVI on four different models: two variants of a Bayesian neural network, a hierarchical Poisson model, and Bayesian logistic regression. For the logistic regression model we used a Gaussian distribution with a full-rank covariance matrix as variational distribution, and for the other models we used a factorized Gaussian.[3] We use three gradient estimators:

- (Rep) Plain reparameterization estimator, which computes the entropy in closed form (eq. (1)),
- (Miller) Estimator proposed by Miller et al. (2017), which adds a Taylor expansion control variate to reduce (Rep)'s variance (see Section 2.2). As suggested by the authors. we use this control variate with a fixed weight of 1.
- (STL) "Sticking-the-landing" estimator (Roeder et al., 2017), which removes the last term of equation (2) (it is always zero) and estimates the remaining terms with reparameterization.

Additionally, we use Algorithm 1 with the set of estimators $\mathcal{G} = \{(\text{Rep}), (\text{Miller}), (\text{STL})\}$, which uses the estimator $g \in \mathcal{G}$ with minimum $\hat{G}^2\hat{T}$. We use $M = 400$

---

[2]We tried computing the optimal estimator just once by minimizing $G^2(g, w_0)T(g)$, and using that g for the full optimization. While this works, it often does not find the best estimator. This is likely because the initial solution w_0 is not so informative about the performance of gradient estimators closer to the solution.

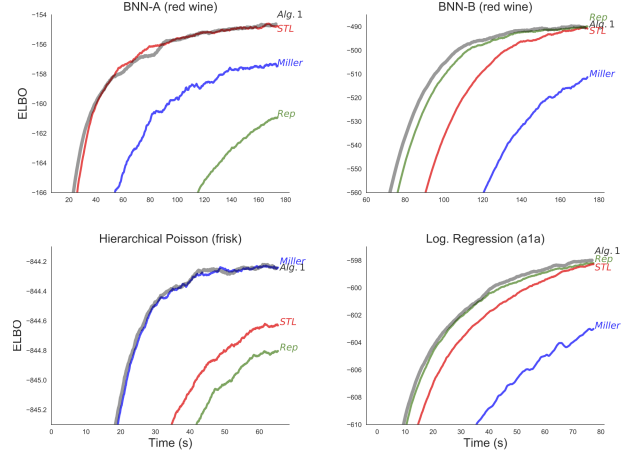[3]More details on the models and optimization parameters in Section 5.1

Figure 2: **Algorithm 1 leads to results comparable to the results obtained using the best estimator chosen retrospectively.** SGVI on four models. Each plot compares Algorithm 1 with three different gradient estimators, as described in the text.

and we re-select the optimal estimator three times throughout training, initially, after 10% of training is done, and after 50% of training is done. Results are shown in Fig. 2. For all models the use of Algorithm 1 leads to results that are at least as good as the results obtained using the best estimator chosen with hindsight. The estimators selected by Algorithm 1 for each model are shown in the appendix.

4 G²T with Control Variates

It is possible to use multiple control variates to reduce a gradient estimator's variance (Geffner and Domke, 2018). However, some control variates might be computationally expensive but only result in a small reduction in variance. It may be better to remove them and accept a noisier but cheaper estimator. We should select what control variates to use adaptively.

Estimators with control variates can be expressed as

$$g_a(w, \xi) = g_{\text{base}}(w, \xi) + \sum_{i=1}^{J} a_i c_i(w, \xi)$$
$$= g_{\text{base}}(w, \xi) + C(w, \xi)a. \quad (7)$$

The set of available estimators is $\mathcal{G} = \{g_a : a \in \mathbb{R}^J\}$. In this case, the number of estimators $g_a \in \mathcal{G}$ is infinite. Therefore, Algorithm 1 cannot be used (since we cannot measure $\hat{T}$ and $\hat{G}^2$ for each estimator).

This section shows how to find the minimum $\hat{G}^2\hat{T}$ estimator when different estimators are indexed by control variate weights. This is possible because two properties hold: (i) $\hat{T}(g_a)$ and $\hat{G}^2(g_a, w)$ can be efficiently

obtained for all estimators $g_a \in \mathcal{G}$ through the use of shared statistics (a finite number of evaluations of the base gradients and control variates); and (ii) Finding g_a that minimizes $\hat{G}^2(g_a, w)\hat{T}(g_a)$ can be reduced to a Mixed Integer Quadratically Constrained Program (MIQCP), which can be solved quickly in practice.

An expression for $\hat{T}(g_a)$ can be obtained by noticing that computing g_a only requires computing the base gradient and the control variates with *non-zero* weights. Then, for all $g_a \in \mathcal{G}$,

$$\hat{T}(g_a) = \hat{T}(g_{\text{base}}) + \sum_{i=1}^{J} \hat{T}(c_i) \, \mathbf{1}[a_i \neq 0]. \quad (8)$$

Thus, we can compute $\hat{T}(g_a)$ for all $g_a \in \mathcal{G}$ only by profiling the base gradient and each control variate individually.

Similarly, $\hat{G}^2(g_a, w)$ is determined by the same set of base gradient and control variate evaluations, regardless of the value of a. Suppose that, at iteration k, we sample $\xi_{k1}, ..., \xi_{kM}$. Then, for all $g_a \in \mathcal{G}$,

$$\hat{G}^2(g_a, w_k) = \frac{1}{M} \sum_{m=1}^{M} \|g_{\text{base}}(w_k, \xi_{km}) + C(w_k, \xi_{km}) \, a\|^2. \quad (9)$$

Thus, we can compute $\hat{G}^2(g_a, w_k)$ for all $g_a \in \mathcal{G}$ using only M evaluations of the base gradient g_{base} and each control variate c_i.

Equations (8) and (9) characterize the (estimated) cost and variance of the gradient estimator with weights a. We find the weights that result in the optimal cost-variance tradeoff by solving

$$a^*(w) = \arg\min_{a \in \mathbb{R}^J} \hat{G}^2(g_a, w) \times \hat{T}(g_a), \quad (10)$$

where $\hat{T}(g_a)$ and $\hat{G}^2(g_a, w)$ are as in equations (8) and (9). The solution $a^*(w)$ indicates what control variates to use (those with $a_i^* \neq 0$), and their weights.

Solving the (combinatorial) minimization problem in equation (10) may be challenging. However, theorem 4.1 states that it can be reduced to a MIQCP, which can be solved fast using solvers such as Gurobi (Gurobi Optimization, 2018).

Theorem 4.1. *When different gradient estimators are indexed by a set of J control variate weights, the problem of finding $a^*(w)$ as in equation (10) can be reduced to solving a mixed integer quadratically constrained program with $2J + 2$ variables, one quadratic constraint, and one linear constraint.*

The proof idea is as follows: If one expands the norm, equation (9) can be written as a quadratic function $\hat{G}^2(g_a, w_k) = \frac{1}{2}a^\top Q a + r^\top a + u$, where Q, r and u

depend on the M evaluations of the base gradient g_{base} and control variates C. Using this, the minimization problem from equation (10) can be expressed as

$$a^* = \arg\min_{a \in \mathbb{R}^J, b \in \{0,1\}^J} \left(\frac{1}{2}a^\top Q a + r^\top a + u\right) \times \left(t_0 + t^\top b\right)$$

$$\text{s.t } b_i = \mathbf{1}[a_i \neq 0], \quad (11)$$

where $t_0 = \hat{T}(g_{\text{base}})$, and $t_i = \hat{T}(c_i)$. Going from eq. (11) to a MIQCP is straightforward. Available solvers can solve the final MIQCP fast. The full proof and resulting MIQCP problem are shown in the appendix.

The final algorithm is summarized in Algorithm 2.

Algorithm 2 SGD with minimum $\hat{G}^2\hat{T}$ estimator; estimators indexed by control variates weights.

Require: Set of available gradient estimators $\mathcal{G}$.
Require: Total opt time and times to re-select estimator.
Require: Number of MC samples M.
  For g_{base} measure time t_0.
  For all $i = 1, ..., J$, measure c_i time t_i.
  for $k = 1, 2, \cdots$ **do**
    if time to re-select estimator **then**
      $a = \text{argmin}_a \hat{G}^2(g_a, w_k)\hat{T}(g_a)$ (solve MIQCP).
    end if
    $g_a = g_{\text{base}}(w, \xi_k) + \sum_{i:a_i \neq 0} a_i c_i(w, \xi_k)$
    $w_{k+1} = w_k - \eta_k \, g_a(w_k, \xi_k)$
  end for

---

5 Experiments and Results

This section presents results that empirically validate Alg. 2 for control variate selection. We tackle variational inference problems on several probabilistic models using SGVI. We perform stochastic gradient momentum while using Algorithm 2 to select the gradient estimator at three different times throughout optimization. The set of candidate estimators is $\mathcal{G}_{\text{Auto}} = \{g_a | a \in \mathbb{R}^3\}$, where g_a is as defined in eq. (7), the base estimator is plain reparameterization, and there are three candidate control variates (c_1, c_2, c_3).

The goal is to check if Alg. 2 successfully navigates time / variance tradeoffs. We thus compare against using *fixed* subsets of control variates with the weights that minimize the estimator's variance (which can be estimated efficiently (Geffner and Domke, 2018)). We consider each possible subset of control variates $S \subseteq \{c_1, c_2, c_3\}$. This is essentially the same as Algorithm 2, but with the set of estimators $\mathcal{G}_S$, which contains all estimators that use the fixed subset of control variates S. Since all estimators in $\mathcal{G}_S$ use the same control variates, they all have the same cost. Therefore, finding the estimator with optimal cost variance tradeoff (minimum $\hat{G}^2\hat{T}$) reduces to finding the estimator with minimum $\hat{G}$.

5.1 Experimental details

Tasks and datasets: We use SGVI on several probabilistic models: (i) Bayesian logistic regression with the datasets Mushrooms and a1a, with a standard normal prior on the weights; (ii) a hierarchical Poisson model with the Frisk dataset (Gelman et al., 2007); (iii) a Bayesian neural network with the Red-wine dataset, with a scaled standard normal prior on the weights and biases; and (iv) a Bayesian neural network with the Red-wine dataset, with a hierarchical distribution as prior on the weights and biases. (The likelihoods for (iii) and (iv) are the same, they only differ in the prior used. We denote them by BNN-A and BNN-B, respectively.) Details for all models are given in the appendix.

Variational distribution $q_w(z)$: For the simpler logistic regression models we use a Gaussian distribution with a full-rank covariance matrix as variational distribution. For the other more complex models we use a factorized Gaussian (diagonal covariance matrix).

Base gradient estimator: As base gradient estimator, g_{base}, we use what seems to be the most common estimator. We compute the entropy term $\nabla_w \mathbb{E}_{q_w}[\log q_w(Z)]$ in closed form, and estimate the term $\nabla_w \mathbb{E}_{q_w}[\log p(x, Z)]$ with reparameterization.[4]

Control variates used:

c_1: Difference between the entropy term computed exactly and estimated using reparameterization: $c(w, \xi) = \nabla_w \log q_w(\mathcal{T}_w(\xi)) - \nabla_w \mathbb{E}_{q_w} \log q_w(Z)$.

c_2: Control variate by Miller et al. (2017) based on a second order Taylor expansion of $\log p(x, z)$ (Sec 2.2).

c_3: Difference between the prior term computed exactly and estimated using reparameterization: $c(w, \xi) = \nabla_w \log p(\mathcal{T}_w(\xi)) - \nabla_w \mathbb{E}_{q_w} \log p(Z)$.

Algorithmic details: For Alg. 2 we use $M = 400$ to estimate $\hat{G}^2$ (except for Logistic regression, where we use $M = 200$). We re-select the optimal estimator three times during training, initially, after 10% of training is done, and after 50% of training is done.

Optimization details: We use SGD with momentum ($\beta = 0.9$) with 5 samples $z \sim q_w(z)$ to form the Monte Carlo gradient estimates. For all models we find an initial set of parameters by optimizing with the base gradient for 300 steps and a fixed learning rate of 10^{-5}. This initialization was helpful in practice because w tends to change rapidly at the beginning of optimization. After this brief initialization, $G^2(g, w)$ tends to

---

[4]Using $\mathcal{T}_w(\xi) = \mu + D^{1/2}\xi$, where $\xi \sim \mathcal{N}(0, I)$, μ is the mean of q_w and $D^{1/2}$ is the Cholesky factorization of the covariance of q_w.

change much more slowly, meaning our technique is more helpful.

5.2 Results

Fig. 3 shows the optimization results ("ELBO vs. Time" plots). All simulations were performed independently 20 times, and the average result is shown. The performance of all algorithms depends on the stepsize. To give a fair comparison, Fig. 3 summarizes by showing the results with the best step-sizes.[5] It can be observed that automatically selecting what control variates to use and their weights (using Alg. 2) leads to good results for all models, as good as the ones obtained using the best fixed set of control variates chosen retrospectively. It can also be observed that results depend heavily on the model considered. For instance, for some models using the base gradient without any control variates appears to be the best choice (BNN-B), while for others it is suboptimal (BNN-A).

Raw results for individual step-sizes are shown in the appendix. The G^2T principle is not ideal when the step-sizes are chosen poorly. For example, if the step-size is chosen very small, convergence may be "iteration limited" rather than "variance limited", so that reducing variance of the gradient estimator does not speed up convergence. This is all consistent with our theory – the convergence rates in Section 3 are based on well-chosen step-sizes.

Fig. 4 shows a summary of the results. For each model, it shows the final training ELBO achieved by using each possible fixed subset of control variates and the automatically chosen ones. It can be observed that there is no fixed subset of control variates that performs well across all models. However, using the automatically chosen control variates does.

Figures 3 and 4 indicate that Alg. 2 selects a "good" set of control variates (and weights). What control variates are being chosen? This is shown in Table 3. It can be observed that the algorithm makes different choices for different models, and that the control variates chosen depend on the stage of the training process (initially, or later stages). As shown in Fig. 3, these choices lead to good results.

6 Conclusion

Results show that there is no single optimal estimator. The best estimator depends on several factors, such as model, dataset and implementation. Estimator selection must be done adaptively. We presented the "G^2T principle" and used it to derive two gradient estimator selection algorithms. We showed empirically that

---

[5]12 stepsizes between 10^{-6} and 10^{-3} were considered.
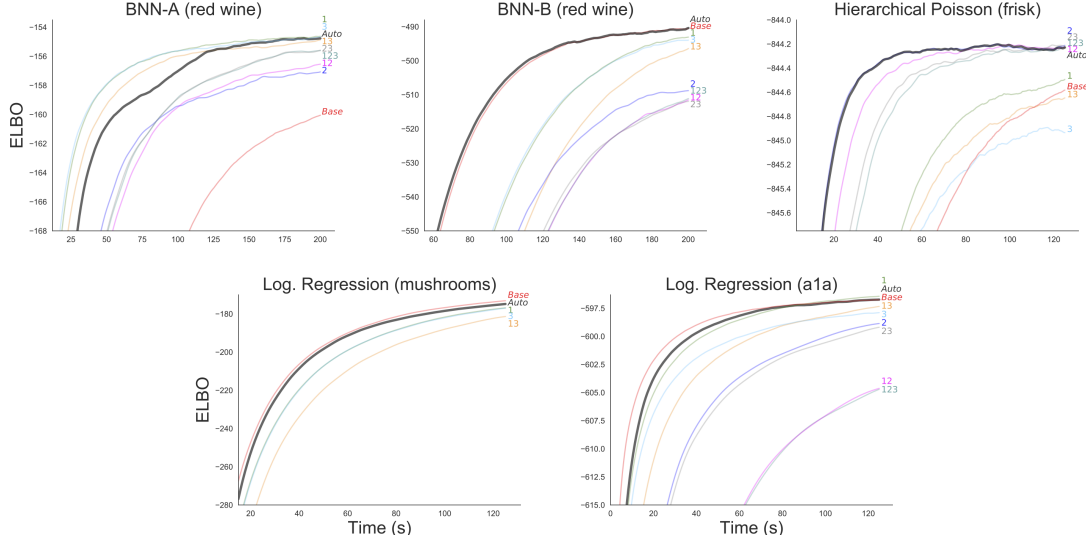
Figure 3: **Automatically selecting what control variates to use performs comparably to the best fixed set of control variates chosen with hindsight.** (Higher ELBO is better.) "ELBO vs Time" plots obtained using Alg. 2 to select what control variates to use and their weights (plotted in black). We compare against using different fixed subsets of control variates with the weights that minimize the estimator's variance. Five models considered, described in the text. Lines are identified as follows: "Auto" stands for using Alg. 2 to select what control variates to use and their weights, "Base" stands for optimizing using the base gradient alone, "1" stands for using the fixed set of control variates $\{c_1\}$ with the minimum variance weights, "12" stands for using the fixed set of control variates $\{c_1, c_2\}$, and so on. Overheads incurred by Algorithm 2 (profiling phase and MIQCP solving) are included in the times. (Note: For the model logistic regression (mushrooms) computing c_2 is slow. When c_2 is used, optimization does not converge in the amount of time provided.)
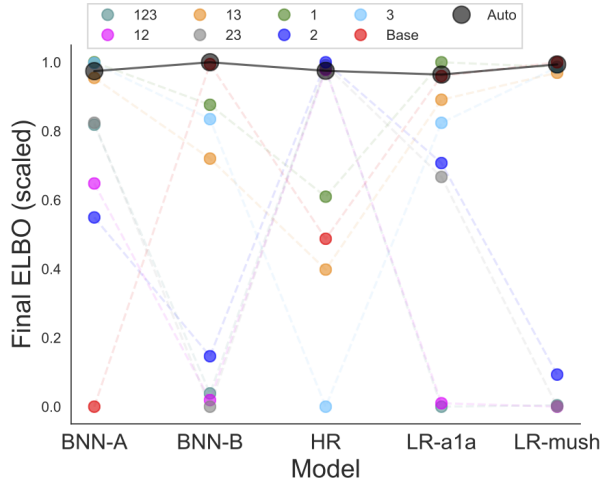

Figure 4: **The automatically chosen control variates perform well in all models.** (Higher ELBO is better.) For each model, we show the ELBO at the final time in Fig. 3, with the lowest ELBO scaled to zero (bottom) and the highest scaled to one (top).

Model	Time of choice		
	$T = 0$	$T = T_{opt}/10$	$T = T_{opt}/2$
Log Reg (a1a)	$\{c_1, c_3\}$	$\{c_1\}$	$\{c_1\}$
Log Reg (mush)	$\emptyset$	$\emptyset$	$\{c_1\}$
Hier Poisson	$\{c_2\}$	$\{c_2\}$	$\{c_2\}$
BNN-A	$\{c_3\}$	$\{c_1\}$	$\{c_1\}$
BNN-B	$\emptyset$	$\emptyset$	$\emptyset$

Table 3: **Algorithm 2 selects different control variates for different models.** Control variates chosen by Algorithm 2. Each column shows the set of control variates selected at some selection time. (In different runs different choices may ocurr. We show the "most popular" choice across runs.)

both these algorithms work well in practice on inference problems tackled using SGVI.

Despite the empirical success of the proposed algorithms, several open questions remain: Can convergence guarantees use "local" G^2 values, as opposed to (looser) global bounds? Should algorithms adapt (e.g. by changing the step size) when the underlying gradient estimator changes? We plan to address these questions in future work.

References

Bottou, L., Curtis, F. E., and Nocedal, J. (2018). Optimization methods for large-scale machine learning. *Siam Review*.

Geffner, T. and Domke, J. (2018). Using large ensembles of control variates for variational inference. In *Advances in Neural Information Processing Systems*, pages 9960–9970.

Gelman, A., Fagan, J., and Kiss, A. (2007). An analysis of the new york city police department's "stop-and-frisk" policy in the context of claims of racial bias. *Journal of the American Statistical Association*.

Grathwohl, W., Choi, D., Wu, Y., Roeder, G., and Duvenaud, D. (2018). Backpropagation through the void: Optimizing control variates for black-box gradient estimation. In *Proceedings of the International Conference on Learning Representations*.

Gurobi Optimization, L. (2018). Gurobi optimizer reference manual.

Kingma, D. P. and Welling, M. (2013). Auto-encoding variational bayes. In *Proceedings of the International Conference on Learning Representations*.

Miller, A., Foti, N., D'Amour, A., and Adams, R. P. (2017). Reducing reparameterization gradient variance. In *Advances in Neural Information Processing Systems*, pages 3708–3718.

Mnih, A. and Gregor, K. (2014). Neural variational inference and learning in belief networks. In *International Conference on Machine Learning*.

Mnih, A. and Rezende, D. (2016). Variational inference for monte carlo objectives. In *International Conference on Machine Learning*, pages 2188–2196.

Nemirovski, A., Juditsky, A., Lan, G., and Shapiro, A. (2009). Robust stochastic approximation approach to stochastic programming. *SIAM Journal on optimization*, 19(4):1574–1609.

Owen, A. B. (2013). *Monte Carlo theory, methods and examples*.

Paisley, J., Blei, D., and Jordan, M. (2012). Variational bayesian inference with stochastic search. In *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, pages 1363–1370.

Rakhlin, A., Shamir, O., and Sridharan, K. (2012). Making gradient descent optimal for strongly convex stochastic optimization. In *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, pages 1571–1578.

Ranganath, R., Gerrish, S., and Blei, D. (2014). Black box variational inference. In *Artificial Intelligence and Statistics*, pages 814–822.

Rezende, D. J., Mohamed, S., and Wierstra, D. (2014). Stochastic backpropagation and approximate inference in deep generative models. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1278–1286.

Roeder, G., Wu, Y., and Duvenaud, D. K. (2017). Sticking the landing: Simple, lower-variance gradient estimators for variational inference. In *Advances in Neural Information Processing Systems*, pages 6925–6934.

Ruiz, F., Michalis, T., and Blei, D. (2016). The generalized reparameterization gradient. In *Advances in Neural Information Processing Systems*, pages 460–468.

Schulman, J., Heess, N., Weber, T., and Abbeel, P. (2015). Gradient estimation using stochastic computation graphs. In *Advances in Neural Information Processing Systems*, pages 3528–3536.

Titsias, M. and Lázaro-Gredilla, M. (2014). Doubly stochastic variational bayes for non-conjugate inference. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1971–1979.

Titsias, M. and Lázaro-Gredilla, M. (2015). Local expectation gradients for black box variational inference. In *Advances in neural information processing systems*, pages 2638–2646.

Tucker, G., Mnih, A., Maddison, C. J., Lawson, J., and Sohl-Dickstein, J. (2017). Rebar: Low-variance, unbiased gradient estimates for discrete latent variable models. In *Advances in Neural Information Processing Systems*, pages 2627–2636.

Williams, R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256.

Yang, T., Lin, Q., and Li, Z. (2016). Unified convergence analysis of stochastic momentum methods for convex and non-convex optimization. *arXiv preprint arXiv:1604.03257*.