2nd Symposium on Advances in Approximate Bayesian Inference, 2019 1–11

Automatically Trading off Time and Variance when Selecting Gradient Estimators

Tomas Geffner TGEFFNER@CS.UMASS.EDU and **Justin Domke** DOMKE@CS.UMASS.EDU
University of Massachusetts, Amherst

1. Introduction

In stochastic gradient variational inference (SGVI) there are multiple gradient estimators with varying costs and variances. Estimators may be obtained using the reparameterization trick (Kingma and Welling (2013); Rezende et al. (2014); Titsias and Lázaro-Gredilla (2014)), the score function method (Williams (1992)), or other techniques (Titsias and Lázaro-Gredilla (2015); Ruiz et al. (2016); Agakov and Barber (2004)). Also, many control variates can be added to an estimator to reduce variance (Miller et al. (2017); Grathwohl et al. (2018); Mnih and Gregor (2014); Paisley et al. (2012); Tucker et al. (2017); Geffner and Domke (2018)).

The cost and variance of an estimator significantly affects optimization convergence speed (Bottou et al. (2018)). The use of different estimators leads to different optimization performances, and the estimator with optimal cost-variance tradeoff is often situation-dependent (for an example see Fig. 1). In settings where multiple estimators with varying costs and variances are available, selecting the optimal one is important. Rather than rely on the user to manually select one, 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 for stochastic optimization.

We study cost-variance tradeoffs by analyzing the convergence rates of several variants of SGD. We express convergence rates in terms of time rather than iterations. This leads to what we call the "G^2T principle": A simple rule that predicts, given a pool of gradient estimators, which one results in the best convergence guarantees for optimization. We use the principle to propose two gradient estimator selection algorithms: One for the case in which a finite pool of estimators is available, and other when the pool contains an infinite number of estimators, each indexed by control variate weights (i.e. control variate selection).

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, and $T(g)$ to denote the computational cost of computing estimator $g(w, \xi)$, measured in seconds.

2. The G^{2}T Principle and Gradient Estimator Selection

2.1. G^{2}T 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. Convergence guarantees for several variants of SGD are shown in Table 1.

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

Table 1: η_k is the step size at iteration k. 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. Proofs in (Rakhlin et al. (2012); Nemirovski et al. (2009); Bottou et al. (2018)).

The crucial observation we make is that the right hand side of all guarantees in Table 1 (the upper bound) can be written as $\alpha\left(\frac{G^2}{K}\right)^p$, where α depends on the properties of the target (convexity, smoothness) and initialization, but *not on the gradient estimator used*. Given a total optimization time budget T_{opt}, an estimator g with time cost $T(g)$ can perform $K \approx T_{\text{opt}}/T(g)$ iterations. Using this value for K we get that all guarantees in Table 1 can be expressed as $\frac{\alpha}{T_{opt}^p}(G^2(g)T(g))^p$. This expression depends on the gradient estimator used only through the $G^2(g)T(g)$ factor. Lower G^2T values result in better guarantees. In other words, for algorithms in Table 1, estimators with lower G^2T lead to better convergence guarantees. We call this the "G^2T principle".[1]

2.2. G^{2}T for Gradient Estimator Selection

Given a pool of estimators with known G^2 and T, the one with minimum G^2T should be used. In practice, however, 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 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. We propose an approach that is justified under two assumptions: (i) Optimization starting from a point w_0 will only visit a restricted part of parameter space. It is sufficient to bound $\mathbb{E}\, ||g(w,\xi)||^2$ for the set of w that may actually be encountered. (ii) $\mathbb{E}\, \|g(w,\xi)\|^2$ tends to decrease slowly over time. When these are true, it makes sense to also form an estimate $\hat{G}^2(g)$ through an initial profiling phase, and to update these estimates a small number of times as optimization proceeds. The approach is summarized in Alg. 1.

2.3. G^{2}T for Control Variates Selection

It is possible to use multiple control variates to reduce a gradient estimator's variance. However, some control variates might be computationally expensive but only result in a

1. Table 2, with more algorithms (e.g. SGD+momentum) for which the G^2T principle holds, in appendix.

small reduction in variance. It may be better to remove them and accept a noisier but cheaper estimator. How can we select what control variates to use?

When an unbiased gradient estimator g_{base} and control variates $c_1, ...c_J$ are given, a gradient estimator can be expressed as

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

The available estimators are $\mathcal{G} = \{g_a : a \in \mathbb{R}^J\}$. The number of estimators $g_a \in \mathcal{G}$ is infinite, and Alg. 1 cannot be used (cannot measure $\hat{T}$ and $\hat{G}^2$ for each estimator g_a individually).

We show that, despite having an infinite number of estimators, when estimators are indexed by control variate weights finding the one with minimum $\hat{G}^2\hat{T}$ can be done efficiently. This is because two properties hold: (i) Estimates $\hat{T}(g_a)$ and $\hat{G}^2(g_a)$ 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 estimator and control variates); and (ii) The resulting (combinatorial) optimization problem $a^* = \arg\min_a \hat{G}^2(g_a)\hat{T}(g_a)$ can be reduced to a Mixed Integer Quadratically Constrained Program (MIQCP), which can be solved quickly in practice. The solution, a^*, indicates what control variates to use (those with $a_i^* \neq 0$) and their weights. The algorithm is summarized in Alg. 2. (More details in the appendix).

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

Require: Set of estimators, $\mathcal{G}$.
Require: 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**
      for each estimator g **do**
        $\hat{G}^2(g) = \frac{1}{M}\sum_{m=1}^{M} \|g(w_k, \xi_{km})\|^2$.
      end for
      $g \leftarrow \arg\min_{g \in \mathcal{G}} \hat{G}^2(g) \times \hat{T}(g)$.
    end if
    $w_{k+1} = w_k - \eta_k \; g(w_k, \xi_k)$
  end for

Algorithm 2 SGD with automatically selected control variates.

Require: Set of estimators, $\mathcal{G}$.
Require: Times to re-select estimator.
Require: Number of MC samples M.
  For g_{base} measure time t_0.
  For $i = 1, ..., J$, measure c_i time t_i.
  for $k = 1, 2, \cdots$ **do**
    if time to re-select estimator **then**
      $a = \arg\min_{a \in \mathbb{R}^J} \hat{G}^2(g_a) \times \hat{T}(g_a)$
      (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

3. Experiments and Results

This section presents an overview of the experiments. Full details are in the appendix. We tackle inference problems using SGVI. We consider three models: Logistic regression, a hierarchical regression model, and a Bayesian neural network. For the simple logistic regression model we use a Gaussian with a full rank covariance as variational distribution $q_w(z)$. For the other more complex models we use a factorized Gaussian. We use SGD with momentum to optimize, and five samples $z \sim q_w(z)$ to form Monte Carlo gradient estimates. For both Algs. 1 and 2 we update the estimator used (by minimizing $\hat{G}^2\hat{T}$) three times during training.

We first present an empirical validation for Alg. 1. We compare the results achieved by using three different gradient estimators: (Rep) the plain reparameterization estimator, (Miller) the estimator proposed by Miller et al. (Miller et al. (2017)), and (STL) the "sticking the landing" estimator (Roeder et al. (2017)). We also run Alg. 1 with the set of estimators $\mathcal{G} = \{\text{Rep}, \text{Miller}, \text{STL}\}$, which uses the estimator $g \in \mathcal{G}$ with minimum $\hat{G}^2\hat{T}$.

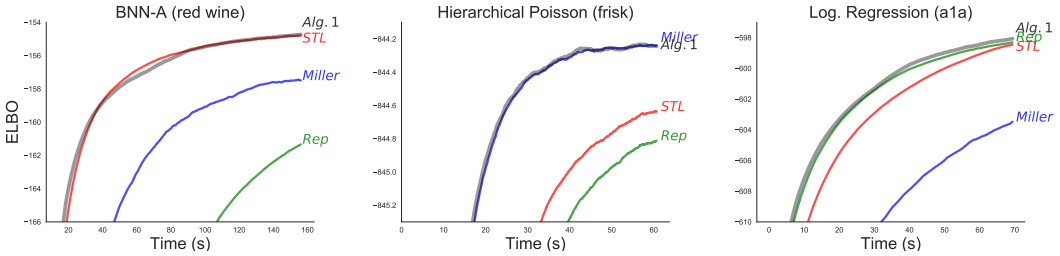

Figure 1: Algorithm 1 leads to results as good as the results obtained using the best estimator chosen retrospectively. (Higher ELBO is better.)

We now present an empirical validation for Alg. 2 (control variate selection). We consider the same three models as above. 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. (2). 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 cost/variance tradeoffs. We thus compare against using each possible *fixed* subsets of control variates $S \subseteq \{c_1, c_2, c_3\}$, with the weights that minimize the estimator's variance (which can be estimated efficiently (Geffner and Domke (2018))).

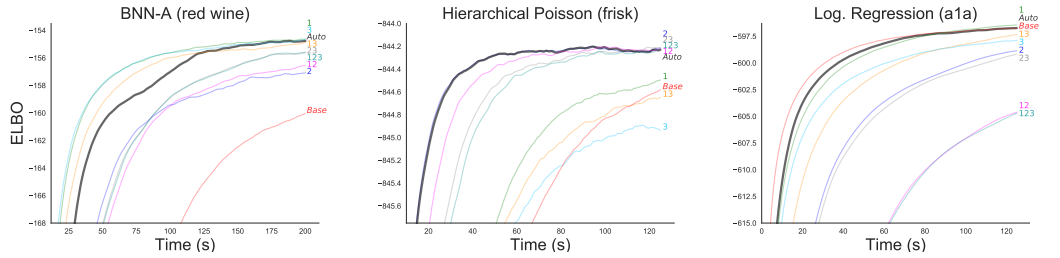

Figure 2: **Automatically selecting what control variates to use leads to ELBO values as high as the ones obtained with the best fixed set of control variates chosen with hindsight.** (Higher ELBO is better.) "ELBO vs Time" plots obtained using Alg. 2 to select what control variates to use and their weights (black line). We compare against using different fixed subsets of control variates with the weights that minimize the estimator's variance. 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.

References

Felix V Agakov and David Barber. An auxiliary variational method. In *International Conference on Neural Information Processing*, pages 561–566. Springer, 2004.

Léon Bottou, Frank E Curtis, and Jorge Nocedal. Optimization methods for large-scale machine learning. *Siam Review*, 60(2):223–311, 2018.

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

Andrew Gelman, Jeffrey Fagan, and Alex Kiss. 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*, 102(479):813–823, 2007.

Will Grathwohl, Dami Choi, Yuhuai Wu, Geoff Roeder, and David Duvenaud. Backpropagation through the void: Optimizing control variates for black-box gradient estimation. In *Proceedings of the International Conference on Learning Representations*, 2018.

LLC Gurobi Optimization. Gurobi optimizer reference manual, 2018. URL http://www.gurobi.com.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *Proceedings of the International Conference on Learning Representations*, 2013.

Andrew Miller, Nick Foti, Alexander D'Amour, and Ryan P Adams. Reducing reparameterization gradient variance. In *Advances in Neural Information Processing Systems*, pages 3708–3718, 2017.

Andriy Mnih and Karol Gregor. Neural variational inference and learning in belief networks. In *International Conference on Machine Learning*, 2014.

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

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

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

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

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

Francisco Ruiz, Titsias Michalis, and David Blei. The generalized reparameterization gradient. In *Advances in Neural Information Processing Systems*, pages 460–468, 2016.

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

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

George Tucker, Andriy Mnih, Chris J Maddison, John Lawson, and Jascha Sohl-Dickstein. Rebar: Low-variance, unbiased gradient estimates for discrete latent variable models. In *Advances in Neural Information Processing Systems*, pages 2627–2636, 2017.

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

Tianbao Yang, Qihang Lin, and Zhe Li. Unified convergence analysis of stochastic momentum methods for convex and non-convex optimization. *arXiv preprint arXiv:1604.03257*, 2016.

Appendix A. Appendix

A.1. Details on models used

Three different models were considered: a Bayesian neural network, a hierarchical Poisson model, and Bayesian logistic regression.

Bayesian logistic regression: We use the dataset a1a. The training set is given by $\mathcal{D} = \{x_i, y_i\}_{i=1}^N$, where y_i is binary. The model is specified by

$$\begin{aligned} w_0 &\sim \mathcal{N}(0, 1), \\ w &\sim \mathcal{N}(0, I), \\ p_i &= \left(1 + \exp(w_0 + w \cdot x_i)\right)^{-1}, \\ y_i &\sim \text{Bernoulli}(p_i). \end{aligned}$$

Hierarchical Poisson model: By Gelman et al. Gelman et al. (2007). The model measures the relative stop-and-frisk events in different precincts in New York city, for different ethnicities. The model is specified by

$$\begin{aligned} \mu &\sim \mathcal{N}(0, 10^2) \\ \log \sigma_\alpha &\sim \mathcal{N}(0, 10^2), \\ \log \sigma_\beta &\sim \mathcal{N}(0, 10^2), \\ \alpha_e &\sim \mathcal{N}(0, \sigma_\alpha^2), \\ \beta_p &\sim \mathcal{N}(0, \sigma_\beta^2), \\ \lambda_{ep} &= \exp(\mu + \alpha_e + \beta_p + \log N_{ep}), \\ Y_{ep} &\sim \text{Poisson}(\lambda_{ep}). \end{aligned}$$

In this case, e stands for ethnicity and p for precinct, Y_{ep} for the number of stops in precinct p within ethnicity group e (observed), and N_{ep} for the total number of arrests in precinct p within ethnicity group e (observed).

BNN: As done by Miller et al. (Miller et al. (2017)) we use a subset of 100 rows from the "Red-wine" dataset (regression). We implement a neural network with one hidden layer with 50 units and Relu activations. Let $\mathcal{D} = \{x_i, y_i\}_{i=1}^N$ be the training set. The model is specified by

$$\begin{aligned} \log \alpha &\sim \mathcal{N}(0, 10^2), \\ \log \tau &\sim \mathcal{N}(0, 10^2), \\ W &\sim \mathcal{N}(0, \alpha^2 I), && \text{(weights and biases)} \\ \hat{y}_i &= \text{FeedForward}(x_i, W), \\ y_i &\sim \mathcal{N}(\hat{y}_i, \tau^2). \end{aligned}$$

A.2. Details on the simulations

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.[2]

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. (Miller et al. (2017)) based on a second order Taylor expansion of $\log p(x, z)$.

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, $\mathbb{E}\,||g(w, \xi)||^2$ tends to change much more slowly, meaning our technique is more helpful.

The performance of all algorithms depends on the step-size. To give a fair comparison, Figs. 1 and 2 summarize by showing the results with the best step-size for each estimator. (12 stepsizes between 10^{-6} and 10^{-3} were considered.)

A.3. Details on G^2T for Control Variate selection

Estimators with control variates can be expressed as

$$\begin{aligned} g_a(w, \xi) &= g_{\text{base}}(w, \xi) + \sum_{i=1}^{J} a_i c_i(w, \xi) \\ &= g_{\text{base}}(w, \xi) + C(w, \xi)a. \end{aligned} \tag{2}$$

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 $g_a \in \mathcal{G}$,

$$\hat{T}(g_a) = \hat{T}(g_{\text{base}}) + \sum_{i=1}^{J} \hat{T}(c_i)\ \mathbf{1}[a_i \neq 0]. \tag{3}$$

2. 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.

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 $g_a \in \mathcal{G}$,

$$\hat{G}^2(g_a, w_k) = \frac{1}{M} \sum_{m=1}^{M} \|g_{\text{base}}(w_k, \xi_{km}) + C(w_k, \xi_{km})\, a\|^2. \tag{4}$$

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 (3) and (4) 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

$$a^*(w) = \arg\min_{a \in \mathbb{R}^J} \hat{G}^2(g_a, w) \times \hat{T}(g_a), \tag{5}$$

where $\hat{T}(g_a)$ and $\hat{G}^2(g_a, w)$ are as in equations (3) and (4). 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 (5) may be challenging. However, theorem 1 states that it can be reduced to a MIQCP, which can be solved fast using solvers such as Gurobi Gurobi Optimization (2018).

Theorem 1 *When different gradient estimators are indexed by a set of J control variate weights, the problem of finding $a^*(w)$ as in equation (5) can be reduced to solving a mixed integer quadratically constrained program with $2J + 2$ variables, one quadratic constraint, and one linear constraint.*

A.3.1. Mixed Integer Quadratic Program

A mixed integer quadratic program is an optimization problem in which the objective function and constraints are quadratic (or linear), and some (or all) variables are restricted to be integers:

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & \frac{1}{2} x^\top Q_0 x + r_0^\top x + u_0 \\ \text{s.t.} \quad & \frac{1}{2} x^\top Q_i x + r_i^\top x + u_i \geq 0 \quad i = 1, ..., m, \\ & Ax + b = 0, \end{aligned} \tag{6}$$

where $x \in \mathbb{R}^n$, $Q_0, ..., Q_m \in \mathbb{R}^{n \times n}$, and some components of x are restricted to be integers.

We now prove the theorem 1.

Proof

Given

$$\bar{T}(g_a) = \bar{T}(g_{\text{base}}) + \sum_{i=1}^{J} \bar{T}(c_i)\ \mathbf{1}[a_i \neq 0] \tag{7}$$

and

$$\bar{G}^2(g_a, w) = \frac{1}{M}\sum_{m=1}^{M} \|g_{\text{base}}(w, \xi_m) + C(w, \xi_m)a\|^2, \tag{8}$$

we want to find

$$a^*(w) = \arg\min_{a\in\mathbb{R}^J} \bar{G}^2(g_a, w) \times \bar{T}(g_a). \tag{9}$$

To simplify notation, we use $\bar{G}^2 = \bar{G}^2(g_a, w)$, $g_{bm} = g_{\text{base}}(w, \xi_m)$ and $C_m = C(w, \xi_m)$. Expanding the squared norm in eq. 8 we get

$$\begin{aligned}
\bar{G}^2 &= \frac{1}{M}\sum_{m=1}^{M} \|g_{bm} + C_m a\|^2 \\
&= \frac{1}{M}\sum_{m=1}^{M} \left(g_{bm}^\top g_{bm} + 2g_{bm}^\top C_m a + a^\top C_m^\top C_m a\right) \\
&= \underbrace{\frac{1}{M}\sum_{m=1}^{M} \|g_{bm}\|^2}_{u_1} + \underbrace{\left(\frac{2}{M}\sum_{m=1}^{M} g_{bm}^\top C_m\right)}_{r_1} a \qquad (10) \\
&\quad + \frac{1}{2}a^\top \underbrace{\left(\frac{2}{M}\sum_{m=1}^{M} C_m^\top C_m\right)}_{Q_1} a \\
&= u_1 + r_1^\top a + \frac{1}{2}a^\top Q_1 a. \qquad (11)
\end{aligned}$$

On the other hand, equation 7 can be expressed as

$$\bar{T}(g_a) = t_0 + t^\top b, \text{ s.t. } b_i = \mathbf{1}[a_i \neq 0], \tag{12}$$

where $t_0 = \bar{T}(g_{\text{base}})$, and $t_i = \bar{T}(c_i)$. Using equations 11 and 12, the minimization problem from equation 9 can be expressed as

$$\begin{aligned}
(a^*, b^*) = \arg\min_{a\in\mathbb{R}^J, b\in\{0,1\}^J} &\left(\frac{1}{2}a^\top Q_1 a + r_1^\top a + u_1\right) \times \left(t_0 + t^\top b\right), \\
\text{s.t } b_i &= \mathbf{1}[a_i \neq 0].
\end{aligned} \tag{13}$$

Introducing two extra varaibles, V_G and V_T, we can express the minimization problem in eq. 13 as

$$\begin{aligned}
(a^*, b^*, V_G^*, V_T^*) = \arg\min_{a\in\mathbb{R}^J, b\in\{0,1\}^J, V_G\in\mathbb{R}, V_T\in\mathbb{R}} & V_G \times V_T, \\
\text{s.t } V_G &\geq \frac{1}{2}a^\top Q_1 a + r_1^\top a + u_1 \\
V_T &= t_0 + t^\top b \\
b_i &= \mathbf{1}[a_i \neq 0].
\end{aligned} \tag{14}$$

The final minimization problem shown in equation 14 has the form of a general MIQCP, shown in equation 6, with the exception of the last constraint $b_i = \mathbf{1}[a_i \neq 0]$. Despite not being in the original definition of a MIQCP, several solver accept constraints of this type (Gurobi Gurobi Optimization (2018), the solver used in our simulation, does).

■

A.4. More SGD convergence rates

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

Table 2: η_k is the step size at iteration k. 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. Proofs in (Rakhlin et al. (2012); Nemirovski et al. (2009); Bottou et al. (2018); Yang et al. (2016)).