Adaptive Vector Quantized Stochastic Gradient Descent

EE 364b Final Project

Samarth Kadaba
Department of Electrical Engineering
Stanford University
skadaba@stanford.edu

Yusef Qazi
Department of Electrical Engineering
Stanford University
yqazi27@stanford.edu

Abstract

Efficient schemes for online quantization can significantly reduce communication costs associated with large-scale, distributed training of parametric models. Numerous bit-wise schemes have been proposed; however, limited floating-point precision and insufficient compression can make these schemes sub-optimal. Here we consider vector-quantized stochastic gradient descent (vqSGD). Vector-quantization [1] is given by the selection of a point set from which to sample gradients per iteration. Thus, communication is dominated by the cost of sending an index from each local node to the central compute server. Deterministic point set constructions [1] may not be well suited to changing gradient and loss landscapes. We propose an adaptive methodology based on formulating the optimal point set as a convex set constructed form a horizon of observed gradients. We show our method performs comparably to full-precision gradient descent and far out-performs simple bit-wise quantization schemes.

1 Background

Motivation The recent surge in the volumes of trainable data for learning parametric models has motivated interest in large-scale distributed algorithms [2]. A broad class of gradient-based methods, which includes Stochastic Subgradient Descent (SGD), approximates an unbiased estimator of true gradients using local data and communicates the results to a central compute node where locally computed gradients are averaged and parameters are updated. This "Federated Learning" [3] setting is primarily bottle-necked by the communication costs of sharing locally computed gradients between multiple workers resulting in time-intensive processes [4]. Here, we consider communication costs in the number of bits needed to encode full-precision gradients of dimension d. To alleviate this, numerous quantization schemes have been developed for the post- and intra-training of models (including large neural networks). In general, the quantization problem, seeks an optimal mapping of continuous, floating-point gradients to discretized or compressed representations (i.e. in the number of required bits). We formulate this map as the solution to an optimization problem which seeks to minimize the relative error between quantized and full-precision gradients while achieving maximal compression (Figure 1).

2 Problem

2.1 Preliminaries

Notation We introduce some notation for clarity. We refer to a point set C as a set of M discrete points. g_i and $g \in \mathbb{R}^d$ refer to observed gradients of dimension d (i.e. at iteration i) and are approximated by $\hat{g}$. For each construction presented below we consider two matrices A which either are set of convex coefficients obeying the simplex constraint (2.3.1), such that $a_i \in \mathbb{R}$, or a PSD,

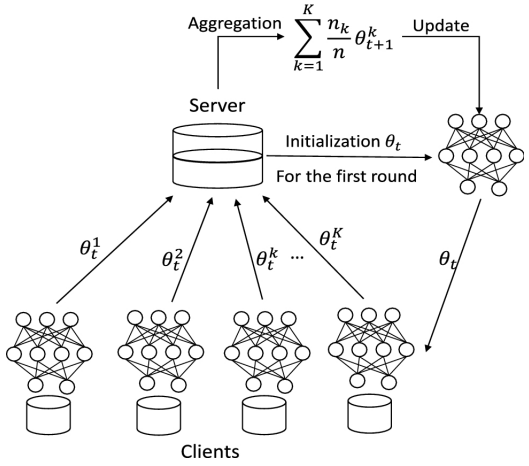

Figure 1: "Federated learning" architecture [3]. Note θ_t typically quantized to avoid expensive communication. Error compounds at server when quantization scheme is poor.

symmetric matrix $A \in \mathbb{S}_+^d$. We refer to a single discrete point from the set C as c_i. For all problem instances we consider constructions of C from a horizon of R gradients with C recomputed after Frq iterations (frequency of recomputation). In general, for a distributed learning setting, we consider the availability of n agents with problem data of varying dimension which we discuss below. We introduce additional notation as necessary.

Optimal Quantization Vector quantization is given by a solution to the unbiased estimator condition in (1). We seek a finite point set C such that a quantization function Q_C yields an unbiased estimator of gradient $g \in \mathbb{R}^d$. Prior work has detailed randomized point sets following Gaussian and other deterministic constructions [1]. We propose an adaptive scheme which defines a point set from horizon R of observed gradients.

$$\min_C \mathbb{E} \left\| Q_C(g) - g \right\|_2^2 \tag{1}$$

2.2 Problem Data

Distributed Least Squares with Regularization We first consider the problem of minimizing (2). With $A_i \in \mathbb{R}^{m_i \times n}$, $b_i \in \mathbb{R}^{m_i}$ and n agents. Note $\sum_i m_i = m$. We evenly split up our large, random, Gaussian-distributed data matrices A and b amongst n agents, solving with local data at each iteration.

$$\min_x \quad \sum_{i=1}^m \left\| A_i x_i - b_i \right\|_2^2 + \lambda \left\| x_i \right\|_2^2 \tag{2}$$

2-Layer Neural Network Next, we consider the problem of minimizing (3), which is a neural network with a non-linear activation. We have $X_i \in \mathbb{R}^p$, $W_1 \in \mathbb{R}^{p \times r}$, $W_2 \in \mathbb{R}^{r \times s}$, and $Y_i \in \mathbb{R}^s$. We define an activation function $\mathcal{A} : \mathbb{R}^{m \times n} \to \mathbb{R}^{m \times n}$. Namely, we experiment $\mathcal{A}(M) = ReLU(M)$, the rectified linear unit, and $\mathcal{A}(M) = \sigma(M)$, the sigmoid function. We define X to contain m samples and evenly split up our large, random, Gaussian-distributed data matrices X and Y amongst n agents, solving with local data at each iteration.

$$\min_{W_1, W_2} \quad \sum_{i=1}^n \left\| W_2^T \mathcal{A}(W_1^T X_i) - Y_i \right\|_2^2 \tag{3}$$

2.3 Point Set Construction

Limitations of Point Sets The original implementation of vQSGD [1] enforced that the unit-norm and the $\mathbb{R}$-norm ball sandwich the set C. We ignore this constraint for simplicity although it is easy to show for Lipschitz-continuous functions, gradients will by default satisfy this bounded condition.

2.3.1 Convex Polytope confined to Simplex

We first discuss a construction which computes a subspace C for which full-precision gradients can be represented as a convex combination of the basis of C. Note that removing the convexity constraint, this formulation simplifies to a low-rank approximation of a set of observed, non-quantized gradients. Instead, representing gradients as specifically a convex combination in the range of C allows for the interpretation of optimal coefficients as a probability distribution over the basis vectors. Hence, sampling this distribution results in sending a simple integer, requiring $log(m)$ bits, to index C after it has been computed (where m is the dimension of subspace C). (2) gives the problem of choosing C over a horizon of observed gradients.

$$\begin{aligned} \min_{C,a_i} \quad & \textstyle\sum_{i=1}^{R} \lambda_i \left\| g_i - \hat{g}_i \right\|_2^2 \\ \text{s.t.} \quad & \hat{g}_i = \textstyle\sum_j a_{i,j} c_j, a_i \succeq 0, \sum_j \ a_{i,j} = 1 \end{aligned} \tag{4}$$

$$\begin{aligned} \min_{a_i} \quad & \left\| g_i - \hat{g}_i \right\|_2^2 \\ \text{s.t.} \quad & \hat{g}_i = \textstyle\sum_{i=1}^{M} a_i c_i, a_i \succeq 0, \sum_j \ a_{i,j} = 1 \end{aligned} \tag{5}$$

The above problem is non-convex due to the $a_{i,j}c_j$ term. Handling this induced complexity is discussed later. To enforce the simplex constraint over the coefficients a_i, we can efficiently project onto the simplex after each gradient step w.r.t a_i [5].

2.3.2 Discretized Lowner-John Ellipsoids

$$\begin{aligned} \min_{A,b} \quad & \log\det(A^{-1}) \\ \text{s.t.} \quad & \left\| Ag_i - b \right\|_2 \leq 1, \quad i = 1, ..., R \end{aligned} \tag{6}$$

We next consider sampling from a Lowner-John ellipsoid of R observed gradients (6). We discretize the resulting ellipsoid to M points using (7) which is non-convex in general but can be given by an SDP relaxation (8) whose last inequality can be reduced to (9) [6]. We require that Y below is PSD.

$$\begin{aligned} \max \quad & t \\ \text{s.t.} \quad & \left\| c_i - c_j \right\|_2 \geq t, \quad \forall\, i \neq j, \\ & \left\| Ac_i - b \right\|_2 \leq 1 \quad \forall\, i \end{aligned} \tag{7}$$

$$\begin{aligned} \min_{X,Y} \quad & t + \textstyle\sum_{i=1}^{R} \left\| x_i - p_i \right\|_2 \\ \text{s.t.} \quad & l - t \leq e_{i,j}^T Y e_{i,j} \leq u + t, \quad \forall\, i \neq j, \\ & \left\| Ax_i - b \right\|_2 \leq 1 \quad \forall\, i, Y \succeq X^T X \end{aligned} \tag{8}$$

$$\begin{bmatrix} I & X \\ X^T & Y \end{bmatrix} \succeq 0 \tag{9}$$

Note in (8), l and u given lower and upper bounds, respectively, on the distance between "bin" points. X is a matrix where each column denoted by x_i are the representative points we seek. e_{ij} is a vector in $\mathbb{R}^d$ with 1 in the ith index and -1 in the jth index. To prevent the collapse of points x_i in X (i.e. converging to the same point), we introduce a euclidean distance term p_i which either pushes optimal discretized points to uniformly distributed points on the ellipse given by (6) or the most recently observed gradients g_i [7]. As a simple heuristic we take l to be a relaxation parameter multiplied by the minimum eigenvalue of A^* from (5) (i.e. $\lambda\sigma_{min}$) and u to be $\lambda\sigma_{max}$.

3 Solvers and Implementation Details

Quasi-linearization for Optimal Polytope Because (4) in general is non-convex, we require an approximate method for finding a solution. We form a Lagrangian from the given objective and constraints and quasi-linearize w.r.t. a_i and c_i. In other words, we take a gradient step w.r.t. to one variable while holding the other constant, given in A.4, and then take a second step w.r.t. to the previously frozen variable. This method works well as discussed later. We enforce simplex constraints on the a_i by implementing projected subgradient descent (i.e. we project the result of each step w.r.t. a_i onto the simplex using methods discussed above). 5 can be easily solved for using convex solvers.

Efficient Solutions for Lowner-John Ellipsoid Finding an optimal ellipsoid can be easily and efficiently achieved by enforcing constraints from each the observed gradients as detailed in (7). We invoke off the shelf solvers for finding this ellipsoid, limiting the number of maximum iterations and desired tolerance to speed up quantization. In general, we observe this to have no effect on the convergence of (7). The relaxed, SDP-form of (7) is given by (8) which can also be solved using standard methods. To limit distance of the SDP approximation to the original problem, we penalize large values of Y.

Correcting Descent Directions It is important to note that we incorporate full-precision gradients in our methods as needed. In particular, if we notice the objective value increasing for consecutive iterations, we determine that the descent method has deviated from a proper track and thus send a full-precision gradient from the agent to retain proper descent. This technique is used in other quantization schemes [2] and allows us to achieve some level of convergence regardless of quantization strength and efficiency.

4 Results

4.1 Communication Cost

Parameters Impacting Quantization Efficiency In general we observe that increasing the dimensionality of C results in less quantization. We measure the ratio from Table 1. using (10). We consider the number of "bits" needed to communicate C and related coefficients in our calculation. ϕ is a parameter that takes on either values of M or 1 based on whether coefficients (simplex method) or an index (ellipsoid method) is sent at each iteration. N is the number of iterations for a given solution instance.

$$E = \frac{N * (\frac{M*d}{Frq} + \phi)}{N * M * d} \tag{10}$$

Less obviously, we observe that the ellipsoid method performs significantly better in terms of quantization efficiency. This is likely due to the more efficient scheme of sending an index versus a set of coefficients which scale with the dimensionality of C. Neural networks perform comparably, likely due to their large weight matrices. We observe in general that the number of agents does not largely impact quantization efficiency.

Method	M **= 5**	M **= 10**	M **= 20**	Frq **= 10**	Frq **= 50**	Frq **= 100**
Simplex (DRR)	0.25	0.40	0.85	1.24	0.41	0.33
Ellipsoid (DRR)	0.12	0.23	0.25	2.03	0.28	0.30
Simplex (NN - ReLU)	0.47	0.65	1.20	1.28	0.72	0.60
Simplex (NN - Sigmoid)	0.47	0.64	1.12	1.35	0.68	0.62

Table 1: Number of floating point values required for solution instances with varying parameters. These values are proxies for bit communication requirements with each value ≈ 64 bits.

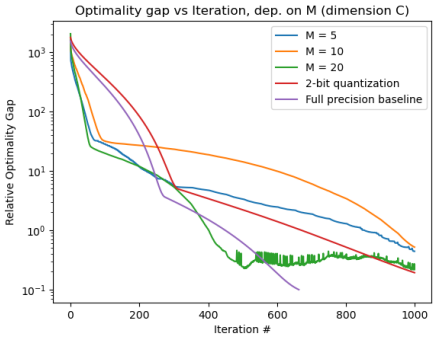

Figure 2: Simplex - Increasing dimensionality of point set C computed from (4) and (5) shows better convergence with a value of $Frq = 20$ converging to optimality gap 10^{-1}. As solutions progress, gradient directions generally become noisier.

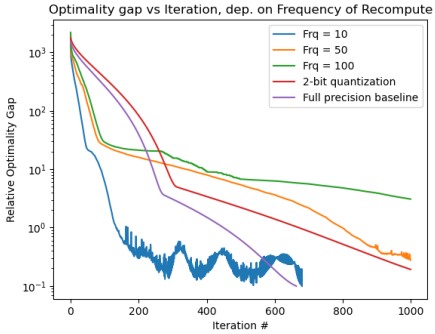

Figure 3: Simplex - We observe that updating the the point set C to infrequently results in sub-optimality. In this case, updating C every $Frq = 10$ iterations yielded the best convergence although decreases quantization efficiency.

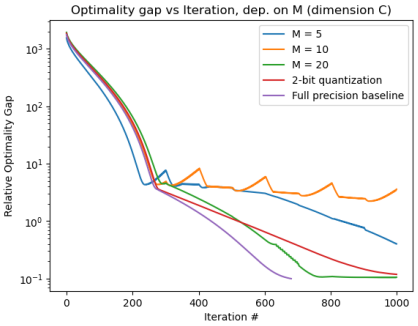

Figure 4: Ellipsoid - Increasing dimensionality of point set C computed from (6-8) shows better convergence with a value of $Frq = 20$ converging to optimality gap 10^{-1}. As solutions progress, gradient directions generally become noisier.

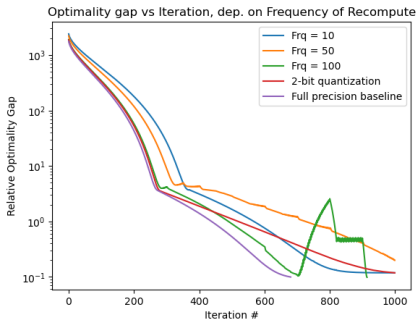

Figure 5: With same problem data as shown in Figure 4. We observe little impact of the frequency of updating point set C on convergence using ellipsoids. This suggests that quantization limits are greater with ellipsoidal approximations.

4.2 Primary Problem Optimality

Problem Instances We analyze the optimality of distributed ridge regression and neural networks, comparing different quantization schemes. As a baseline, we compute full-precision gradients and send them in full back from the n agents back to the main server. We also utilize a secondary baseline of a 2-bit quantization in which each element of a gradient is converted to its sign, 1 or -1. For distributed ridge regression, we explore the quasi-linearization quantization technique, which we will refer to as the simplex method. We also use the ellipsoid quantization method and compare their performances. We initialize random data $A \in \mathbb{R}^{10000\times 50}$ and $b \in \mathbb{R}^{10000}$. An optimal value is calculated using the Moore-Penrose pseudo-inverse of A. We use the simplex method and initialize a two-layer neural network with data $X \in \mathbb{R}^{10000\times 2}$ and $Y \in \mathbb{R}^{10000\times 4}$. For this problem, we generate X randomly and then use random proxy weights to generate Y. This ensures that there is a set of weights that fully optimizes the neural network.

Varying Point Set Dimensionality We next consider varying the dimensionality (M) of the point set C. We find that, in general, increasing M decreases terminal optimality gap. This is an intuitive result since a higher dimensionality produces a quantized gradient that is closer to the true gradient. Our method (Figure 2) approaches full-precision trajectories, interestingly converging faster (albeit to to a slightly higher value than optimal). The ellipsoid method (Figure 4) performs comparably,

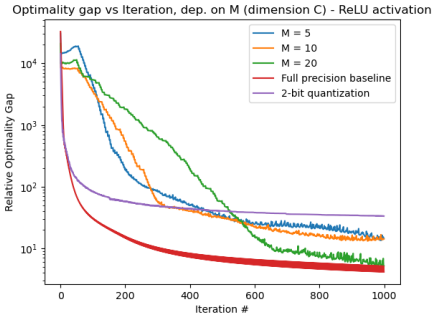

Figure 6: With ReLU, compression ratio was much as $\sim$0.7 was achieved. More optimal results were found with varying Fr and M together.

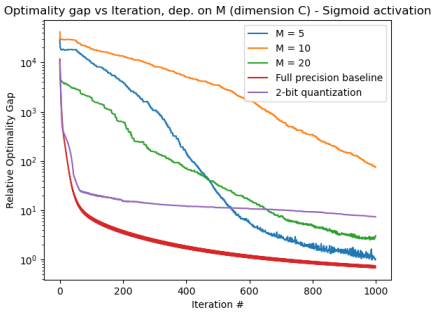

Figure 7: With Sigmoid (a non-convex activation), our method further outperforms 2-bit quantization, achieving a compression ratio of $\sim$0.5.

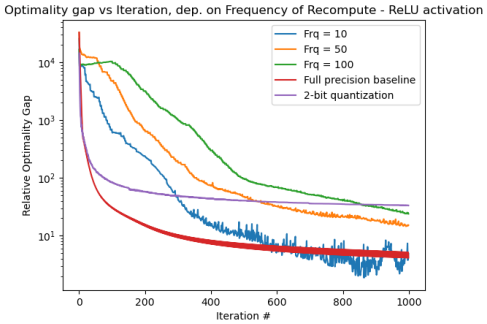

Figure 8: ReLU - We see high frequency recomputation aproaches full-precision baseline. In the limit, all methods outperform 2-bit quanization.

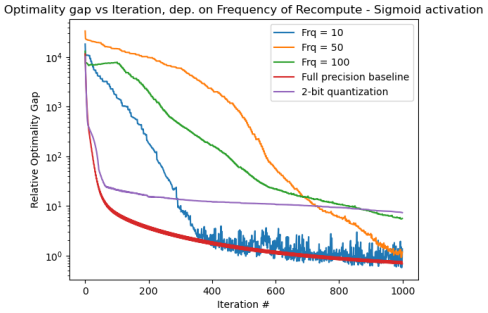

Figure 9: Compared with Figure 8, our methods perform comparably better (to baseline) on Sigmoid, a non-convex activation. Sampling can help mitigate effects of noisy gradients.

demonstrating slightly slower convergence than simplex, although achieving a lower final optimality gap.

Varying Frequency of Point Set Recomputation We separately adjust the frequency of how often the point set C is recomputed in terms of number of iterations. We observe that increasing the frequency (reducing the value of Frq) performs better than more infrequent updates. Updating the point set C more often implicitly weights recent gradients more as thus quantization results in outputs that are likely more representative of the next observed gradient. Figure 3 shows the interesting result that frequent recomputation yields faster, but nosier, convergence than full-precision baselines. Practically, this is of little benefit since quantization efficiency is poor with this high frequency of recomputation ($Frq = 10$). The ellipsoid method shows more stable convergence in general for decreasing values of Frq (Figure 5).

Parallels for Neural Network Figures 6-9 indicate that in the case of neural networks, appropriate dimensionality of C and frequency of recomputation Frq yields identical results to full-precision baselines and far outperforms 2-bit quantization. This holds true for non-convex activation functions such as sigmoid activation and indicates that our quantization scheme can be effective for training deeper networks with more complex non-linear activation functions.

We also analyze convergence of subproblems (computing C) in sections A.2 and A.3. We show generally that convergence of computing C is correlated with primary problem optimality (discussed above).

5 Contributions and Future Work

In this project, we explore applications of convex optimization in quantizing gradients for distributed optimization and nonlinear, non-differentiable neural network problems. We show convergence to acceptable optimality gaps for all methods while achieving positive compression. See code in section A.1.

Quantization via Point Sets In general, defining a point set from a horizon of previously observed gradients provides a reasonable approximation for future gradients in both convex (distributed least squares) and non-convex (neural network) settings. Similarly, minimum volume ellipsoids are efficient sets from which we can sample gradients effectively while reducing computational overhead compared with optimizing directly for euclidean distance. In many metrics such such as iterations for convergence and final optimality gap, sampling via Lowner-John ellipsoids outperforms simple polytope construction. Adaptive schemes which update point sets are necessary for non-stagnant progress towards optimal. By saving on communication costs, we sacrifice efficiency of convergence and final optimality gap when defining the dimensionality of point sets from which we sample gradients. A marginal communication benefit is observed in smaller problem instances and only over a large number of iterations do we achieve meaningful compression.

Efficient Updates In the future, we would like to extend these quantization schemes at a lower level in hardware. Particularly, to validate communication speedup from reduced bit transmission, we plan to experiment with synchronization and multiprocessing/multi-threading modules to observe compute time in a variety of problem instances. We also aim to improve the sub-problem computational time for the point set C. We will explore foundations for efficient updates of polyhedral convex sets and Lowner-John ellipsoids after observing a new gradient g_i. These efficient updates allow for expanding horizons (i.e. increasing R). There also simpler quantization techniques that can serve as heuristics to own methods. Convex optimization formulations of finding point set C allows for certificates of global optimality. These simpler point set constructs, such as based on Singular Value Decomposition (SVD) or Principal Component Analysis (PCA), may yield as good or better results.

References

[1] Venkata Gandikota, Daniel Kane, Raj Kumar Maity, and Arya Mazumdar. vqsgd: Vector quantized stochastic gradient descent. 2020.

[2] Chung-Yi Lin, Victoria Kostina, and Babak Hassibi. Differentially quantized gradient methods. volume 68, pages 6078–6097, 2022.

[3] Xin Yao, Tianchi Huang, Rui-Xiao Zhang, Ruiyu Li, and Lifeng Sun. Federated learning with unbiased gradient aggregation and controllable meta updating. volume abs/1910.08234, 2019.

[4] Guangfeng Yan, Shao-Lun Huang, Tian Lan, and Linqi Song. Dq-sgd: Dynamic quantization in sgd for communication-efficient distributed learning. 2021.

[5] Weiran Wang and Miguel Á. Carreira-Perpiñán. Projection onto the probability simplex: An efficient algorithm with a simple proof, and an application. 2013.

[6] Stephen Boyd, Laurent El Ghaoui, Eric Feron, and Venkataramanan Balakrishnan. *Linear Matrix Inequalities in System and Control Theory*. Society for Industrial and Applied Mathematics, 1994.

[7] Mervin E. Muller. A note on a method for generating points uniformly on n-dimensional spheres. *Commun. ACM*, 2:19–20, 1959.

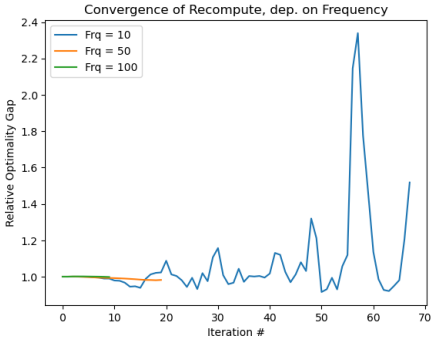

Figure 10: Using the simplex method we observe noisy recomputation of C as Frq decreases. Intuitively, a larger horizon of gradients implicitly adds affine constraints which over-determine the problem. Since gradient norms may not smoothly increase, a sliding window approach yields noisy computation.

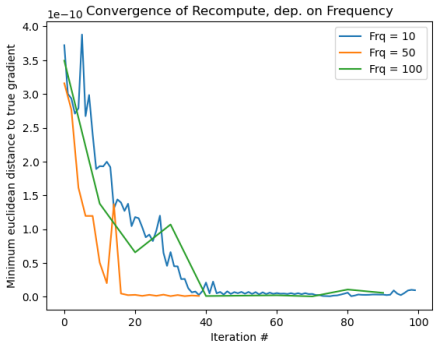

Figure 11: The ellipsoid method shows optimal convergence of computing C for intermediate values of Frq. In this case we observe and empirical trade-off between the number of constraints and the minimum volume ellipsoid containing those observed gradients.

A Appendix

A.1 Code

We implement our algorithms for Distributed Ridge Regression and 2-Layer Neural Networks. All code and instructions for execution are attached at this link.

A.2 Subproblem Optimality

Convergence of Point Set Computation We also analyze the optimality of computing the point set C itself with the quasi-linearization and ellipsoid methods. As discussed above, both of these problems are non-convex but can be relaxed into convex adjacents and solved for. In Figures 10 and 11, we vary the frequency of the recomputation and notice that regardless of the frequency, we eventually reach close to optimality (zero cost of computation), indicating that we are able to accurately represent all observed gradients. This occurs most frequently for low frequencies of recomputation except in the case of the ellipsoid method whose recomputation seems robust to frequency parameter Frq.

Parallels for Neural Networks In neural networks, however, we do not see this same convergence and instead observance noisy recomputation. A similar trend is observed in that higher frequencies of recomputation yields nosier optimal values when computing C however in general we observe that noise persists in the layers for a range of values of Frq. Interestingly, we note that noise in Figure 13 is primarily localized to the first layer. Since back-propogated gradients with respect to the second layer are more normalized in value (as a product of applying sigmoid activation), we rationalize that they are better captured by a polytope. This implicates that smooth activations can be utilized to better bound convex sets containing gradients for highly parametric models.

Subproblem Optimality as Explanatory Variable Analyzing the optimality of set C recomputation informs whether our method is truly sampling unbiased estimates of the gradient or making random predictions which arbitrarily are descent directions. The results presented above directly corroborate those in Section 4.2 since we observe that optimality gap of computing C is closely related to primary optimality convergence.

A.3 Sphere-packing for Ellipsoid Method

Figure 14 provides graphical intuition for the ellipsoid method. Intuitively, we seek a discretization whose points are maximally spanning within the minimal volume ellipsoid of a horizon of gradients.

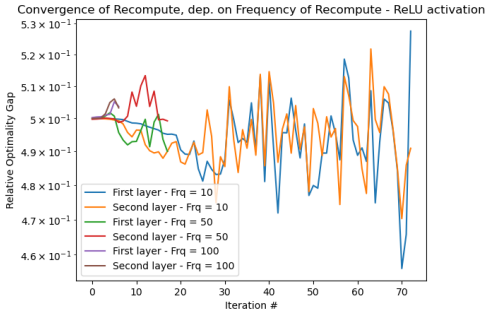

Figure 12: Neural networks with ReLU activation show similarly intuitive results with increasing noise of recomputation as Frq decreases. Noise between the first and second layers look typically identical.

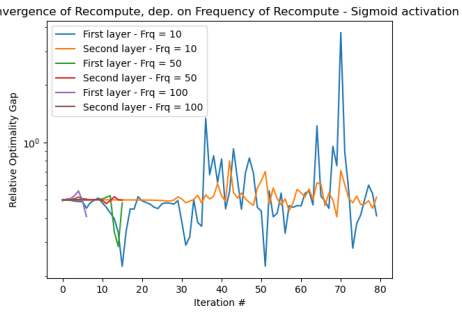

Figure 13: Sigmoid activation show slightly varying patterns with increasing noise related to decreasing values of Frq but also noticeably greater variation in the first layer compared with the second.

This is to ensure that the complete variance of observed gradients is captured by optimally sampling the ellipsoid. This NP-hard sphere-packing problem is approached by rewarding discrete points picked close to the boundary while maximizing their inter-distance. When deciding how to localize "bin" points, we weight distances to most recently observed gradients the most to preserve the unbiasedness of our estimator.

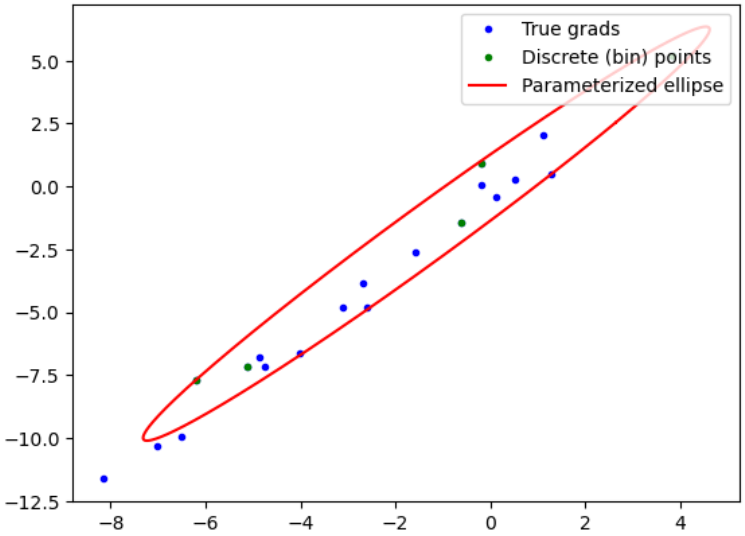

Figure 14: Bins (in green) are chosen to maximize pairwise distance with regularization terms to drive points towards boundaries. Although not plotted here, as we progress in SGD, ellipsoids of a horizon of gradients shrink reflecting lower margin of error.

A.4 Gradient Derivations for Simplex-Bounded Polytope Construction

Below are derivations of gradients w.r.t C and coefficients a_i for the method described in Section 2.3.1. Note, we treat C as a matrix of arbitrary size and thus can take meaningful derivatives of it.

$$\frac{\partial f_i}{\partial C} = \lambda_i (C a_i - g_i) a_i^T \tag{11}$$

$$\frac{\partial a_i}{\partial C} = \lambda_i (C^T C a_i - C^T g_i) \tag{12}$$