École Polytechnique Fédérale de Lausanne
Lausanne | Vaud | Switzerland

School of Computer and Communication Sciences

Comparison of Difference Methods for Zeroth Order Optimisation on Deep Learning Models

— CS-439 | Optimisation for Machine Learning | (Spring 2023)

By:
Edmund Hofflin, Eliott Van Dieren, Sabri El Amrani

Professorss:
Dr. Jaggi Martin
Dr. Flammarion Nicolas Henri Bernard

June 2023

1. Introduction

In many applications, one has to solve optimisation problems whose objective functions do not permit gradient computations [1]. As a result, gradient-free optimization methods that only evaluate the function are required to solve these black-box problems. Zeroth Order Stochastic Gradient Descent (ZO-SGD) is a class of optimisation algorithms that use difference methods to approximate the gradient. The performance of ZO-SGD depends on a variety of hyperparameters and the difference method used. We investigate these dependencies, also explored in [5] for black-box deep learning, on simple multilayer perceptrons, putting our focus on the effect of dimensionality on performance and computation time.

2. Algorithms

Stochastic Gradient Descent In many applications, objective functions have the form

$$f(x) = \frac{1}{n}\sum_{i=1}^{n} f_i(x), \tag{2.1}$$

where f_i is the loss function of the i-th observation, and n the number of observations. Stochastic Gradient Descent (SGD) is the preferred algorithm for optimising such functions, balancing low computational cost per step, requiring one ∇f_i to be computed, and a decent convergence rate of $\mathcal{O}\left(\frac{1}{\epsilon}\right)$ [2] (see Appendix A for the pseudo-code of the algorithm). However, SGD does require gradient computation, which is impossible if f or individual f_i are non-differentiable.

Zeroth Order Stochastic Gradient Descent Zeroth Order Stochastic Gradient Descent (ZO-SGD) is a class of optimisation methods that employ difference methods to approximate a stochastic gradient and then emulate SGD (see Appendix A for the pseudo-code of algorithms). There are two categories of difference methods, depending on the number of function evaluations needed: one-point and multi-point methods [4]

The one-point estimate can be defined as

$$\hat{\nabla} f(\mathbf{x}) = \frac{\phi(d)}{\mu} f(\mathbf{x} + \mu\mathbf{u})\mathbf{u}, \tag{2.2}$$

where $\mathbf{u} \sim P$ is a random direction vector which follows a given distribution P, ϕ denotes a dimension-linked factor related to the choice of P, and $\mu > 0$ is a perturbation radius. Typically, $P := \mathcal{N}(\mathbf{0}_d, \mathbf{I}_{d\times d})$ or a uniform distribution over the unit sphere such that $P := U(\mathcal{S}(0,1))_d$. Based on the choice of P, we let $\phi(d) = 1$ for $P = \mathcal{N}(\mathbf{0}_d, \mathbf{I}_{d\times d})$ and $\phi(d) = d$ for $P = U(\mathcal{S}(0,1))_d$.

The two-point estimate extends the single-point estimate by substracting the evaluation of the objective function at $\mathbf{x}$, as seen below:

$$\hat{\nabla} f(\mathbf{x}) := \frac{\phi(d)}{\mu}\left(f(\mathbf{x} + \mu\mathbf{u}) - f(\mathbf{x})\right)\mathbf{u} \tag{2.3}$$

Additionally, one can use the mini-batch technique, drawing $\{\mathbf{u}_i\}_{i=1,\ldots,b} \sim P$, and then define the following two-point estimate

$$\hat{\nabla} f(\mathbf{x}) := \frac{\phi(d)}{\mu}\sum_{i=1}^{b}\left(f(\mathbf{x} + \mu\mathbf{u}_i) - f(\mathbf{x})\right)\mathbf{u}_i \tag{2.4}$$

When the number of function evaluations reaches d, one can decide to use coordinate-wise gradient estimate instead of randomly drawing $\mathbf{u}_i$'s. This yields the coordinate-wise two-point estimate

$$\hat{\nabla} f(\mathbf{x}) := \frac{1}{\mu} \sum_{i=1}^{d} \left(f(\mathbf{x} + \mu \mathbf{e}_i) - f(\mathbf{x}) \right) \mathbf{e}_i, \tag{2.5}$$

where $\{\mathbf{e}_i\}_{i=1,\dots,d}$ is the canonical basis of $\mathbb{R}^d$.

3. Experiments

We now investigate the performance of ZO-SGD in an empirical setting, determining whether it is a viable class of optimisation algorithms. We conduct this investigation by assessing the impact of the difference method and hyperparameters μ and d on the ZO-SGD's proficiency to train multilayer perceptrons (MLPs). Our code can be found on Github.

3.1. Dataset Description

We performed our experiments on the Iris dataset, which is a small-size set well adapted for simple testing of multiclass classification. The dataset of 150 instances comprises 4 numeric features, is well-balanced and proved sufficiently complex to explore some interesting trends for the simple two-layer MLPs we studied. We also tested the bigger pendigits dataset (10 992 instances), but the computational cost of ZO-SGD made experiments impractical.

3.2. Methodology

We elected to train perceptrons with a single hidden layer, whose dimension we varied. We limited the tests to such simple models to ensure a direct relation between the dimensionality of the problem and the performance of ZO-SGD. We tested two standard activation functions, ReLU and sigmoid. However their results and trends were near identical, so we only present the ReLU results for clarity and to demonstrate ZO-SGD optimising non-differentiable functions. We used a learning rate of 0.01, 100 epochs, a batch-size of 32, and the cross-entropy loss. Additionally, all models were initialised with the same seed.

We varied the dimension of the hidden layer over $\{3, 4, 8, 16, 32, 64\}$, the perturbation radius hyper-parameter μ over $\left\{1, 10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}\right\}$, and the batch number b over $\left\{1, 10, 10^2, 10^3\right\}$. Finally, for all tests, we compared the multi-batch versions of one-point and two-point, and coordinate ZO-SGD against vanilla SGD. We will omit "multi-batch" for clarity.

3.3. Results

Batch requirement of one- and two-point methods The most immediate result was the high batch b requirement of both the one- and two-point estimation methods. The one-point method was never stable and did not consistently converge, even with $b = 1000$. The two-point method was somewhat stable, converging in roughly 50% of tests with $b = 1000$, but very infrequently with $b = 100$ and never with $b = 1$ or 10. These results align with the theory: one-point estimate suffers from extremely high variance, slowing or preventing convergence, hence multi-point estimates are almost exclusively used in practice [3]. Given this, we focus our attention hereafter on the two-point estimation method with $b = 1000$ and the coordinate estimation method.

General comparison of two-point and coordinate ZO-SGD and SGD Figure 1 compares these methods against SGD for $\mu = 0.01$ and $d = 4$. We observe that the SGD has a lower loss than its two other zeroth-order methods. The two-point and coordinate ZO-SGD have similar behaviours, and do not have a high loss reduction over the 100 epochs. Furthermore, the computational time in seconds per epoch was approximately 1, 0.1 and 0.001, for two-point, coordinate, and SGD, respectively. So not only did ZO-SGD converge more slowly than SGD, it required orders of magnitude more computation. While two-point and coordinate estimation variants of ZO-SGD are valid optimization methods, they are significantly slower than SGD, and should therefore be used when only necessary (i.e. when SGD is not available).

Impact of Dimensionality Figure 2 shows the results of our study of the effect of dimensionality (varied by changing the number of hidden neurons of our 2-layer network) on convergence and computation time. The left plot of Figure 2 shows that SGD and coordinate descent are quite robust methods: they remained stable for the range of dimensions tested. However, the loss of two-point estimates depends heavily on the number of hidden layers, suggesting a stability issue. This instability is not surprising when we know from [5] that the variance of mini-batch stochastic gradients for this method is

$$\mathbb{E}[\|\hat{\nabla} f(\mathbf{x}) - \nabla f(\mathbf{x})\|_2^2] = \|\nabla f(\mathbf{x})\|_2^2 \mathcal{O}(\frac{d}{b}) + \mathcal{O}\left(\frac{\mu^2 d^3}{\phi(d) b}\right) + \mathcal{O}\left(\frac{\mu^2 d}{\phi(d)}\right). \tag{3.1}$$

We can see in this equation that the variance scales with d^3, which could explain the instability with increasing dimensions. Coordinate ZO-SGD does not suffer from that problem: the same variance scales with $\mathcal{O}\left(d\mu^2\right)$ for this method.

Figure 2 also shows that both the computation time of the two-point estimate and coordinate descent scale linearly with the dimensionality. This is consistent with theory: we can see from equations (2.3) and (2.5) that the complexities of the gradient approximations are respectively of $\mathcal{O}(bd)$, as we sample u_i from $\mathbb{R}^d$ and $\mathcal{O}(d)$. SGD theoretically also has a complexity of $\mathcal{O}(d)$, but that was hardly visible in the experimental plot given its small slope.

Impact of μ Figure 3 shows the relationship between the perturbation radius μ and the performance of two-point and coordinate estimation methods for ZO-SGD, varying μ over $\{10^{-1}, 10^{-3}, 10^{-5}\}$. As evidenced by Equation (3.1), the variance of these difference methods scales with μ^2. However, in Figure 3, the coordinate method is seemingly invariant under the changes in μ, and the two-point method only alters its loss trajectory with minimal, if any, change in variance. We suspect that this minimal impact is due to the relative scale of μ: it is several orders of magnitude smaller than d, the most significant variance factor, and is potentially even small enough to be comparable to numerical errors. Therefore, at this scale we conclude that μ is a less significant hyperparameter than d for the performance of ZO-SGD methods.

4. Conclusion

In this project, we assessed and compared various zeroth-order methods for SGD. While those methods are valid optimization methods, they are extremely slow, and do not cope with dimensionality increases. Their usage should therefore be kept only for problems where the already existing first-order methods do not work, as a last resort. From our literature review, we noted that no empirical comparison of computation time with respect to the dimensionality was done. Hence, Figure 2 and its analysis is our main contribution to the subject.

Next steps of this work would be to assess the impact of the number of samples on computation time and loss for the ZO-SGD methods, and try to use those methods for other types of ML algorithms.

References

[1] Stéphane Alarie, Charles Audet, Aïmen E Gheribi, Michael Kokkolaras, and Sébastien Le Digabel. Two decades of blackbox optimization applications. *EURO Journal on Computational Optimization*, 9:100011, 2021. [Cited on page 1.]

[2] Léon Bottou, Frank E Curtis, and Jorge Nocedal. Optimization methods for large-scale machine learning. *SIAM review*, 60(2):223–311, 2018. [Cited on page 1.]

[3] Abraham D Flaxman, Adam Tauman Kalai, and H Brendan McMahan. Online convex optimization in the bandit setting: gradient descent without a gradient. *arXiv preprint cs/0408007*, 2004. [Cited on page 2.]

[4] Sijia Liu, Bhavya Kailkhura, Pin-Yu Chen, Paishun Ting, Shiyu Chang, and Lisa Amini. Zeroth-order stochastic variance reduction for nonconvex optimization, 2018. [Cited on page 1.]

[5] Sijia Liu, Pin-Yu Chen, Bhavya Kailkhura, Gaoyuan Zhang, Alfred O Hero III, and Pramod K Varshney. A primer on zeroth-order optimization in signal processing and machine learning: Principals, recent advances, and applications. *IEEE Signal Processing Magazine*, 37(5):43–54, 2020. [Cited on pages 1 and 3.]

A. Formal Algorithms

In this section, we present the algorithms GD and SGD to solve (2.1).

Algorithm 1: Gradient descent

```
Input: $\mathbf{w}_0 \in \mathbb{R}^p$ as starting point, stepsize $\eta$.
for $k$ = 0,1,2,... do
    Set $\mathbf{w}_{k+1} = \mathbf{w}_k - \eta \nabla P(\mathbf{w}_k)$
```

Algorithm 2: Stochastic gradient descent

```
Input: $\mathbf{w}_0 \in \mathbb{R}^p$ as starting point, stepsize $\eta$.
for $k$ = 0,1,2,... do
    Select $i \in \{1, ..., n\}$ uniformly at random.
    Set $\mathbf{w}_{k+1} = \mathbf{w}_k - \eta \nabla f_i(\mathbf{w}_k)$
```

Algorithm 3: Zeroth order stochastic gradient descent

```
Input: $\mathbf{x}_0 \in \mathbb{R}^p$ as starting point, stepsize $\eta$, and the gradient estimation function $\psi(\cdot, \phi, \mu)$.
for $k$ = 0,1,2,... do
    Select $i \in \{1, ..., n\}$ uniformly at random.
    Set $\hat{\nabla} f_i(\mathbf{x}_k) := \psi(f_i, \mathbf{x}_k, \phi, \mu)$
    Set $\mathbf{x}_{k+1} = \mathbf{x}_k - \eta \hat{\nabla} f_i(\mathbf{x}_k)$
```

B. Plots of Results

In this section, we present plots that evidence our results in Section 3

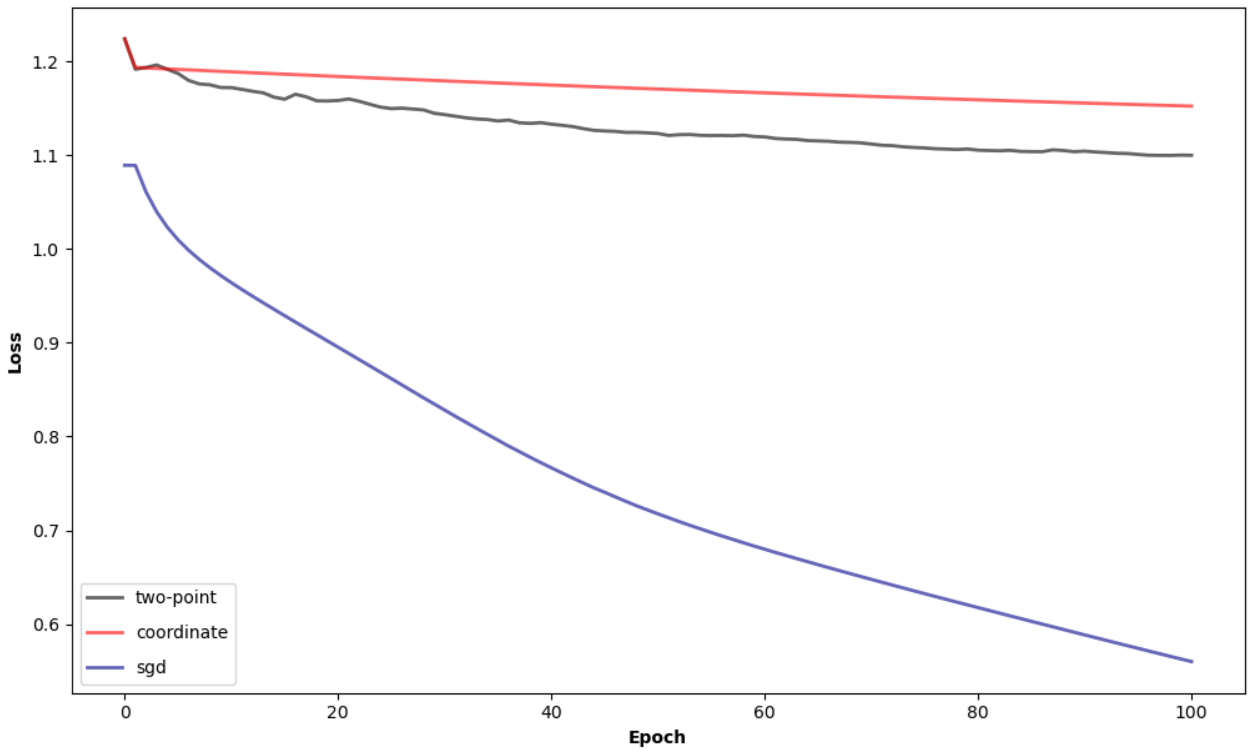

Figure 1: Comparison of loss over epochs of Two-Point and Coordinate Estimation variants of ZO-SGD and SGD with Hidden Dimension $d = 4$ and $\mu = 0.01$. SGD performs better regarding the two other methods over epochs.

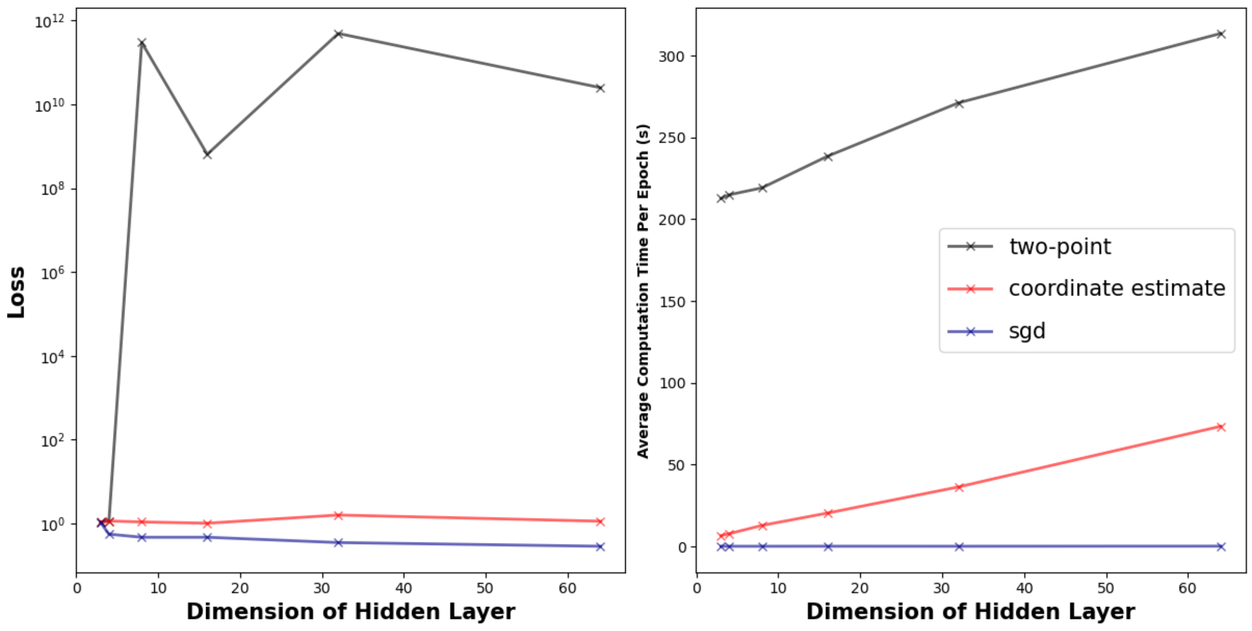

Figure 2: Impact of hidden layer dimension on the loss and computational time of two-point and coordinate estimation variants of ZO-SGD and SGD with $\mu = 0.01$. The left plot suggests that two-point estimates are quickly destabilised by high dimensionality, whereas coordinate descent and SGD look robust. The right plot on the other hand shows that the three algorithms scale linearly with dimensionality, but with very different coefficients: dimensionality is more problematic for two-point and coordinate estimates.

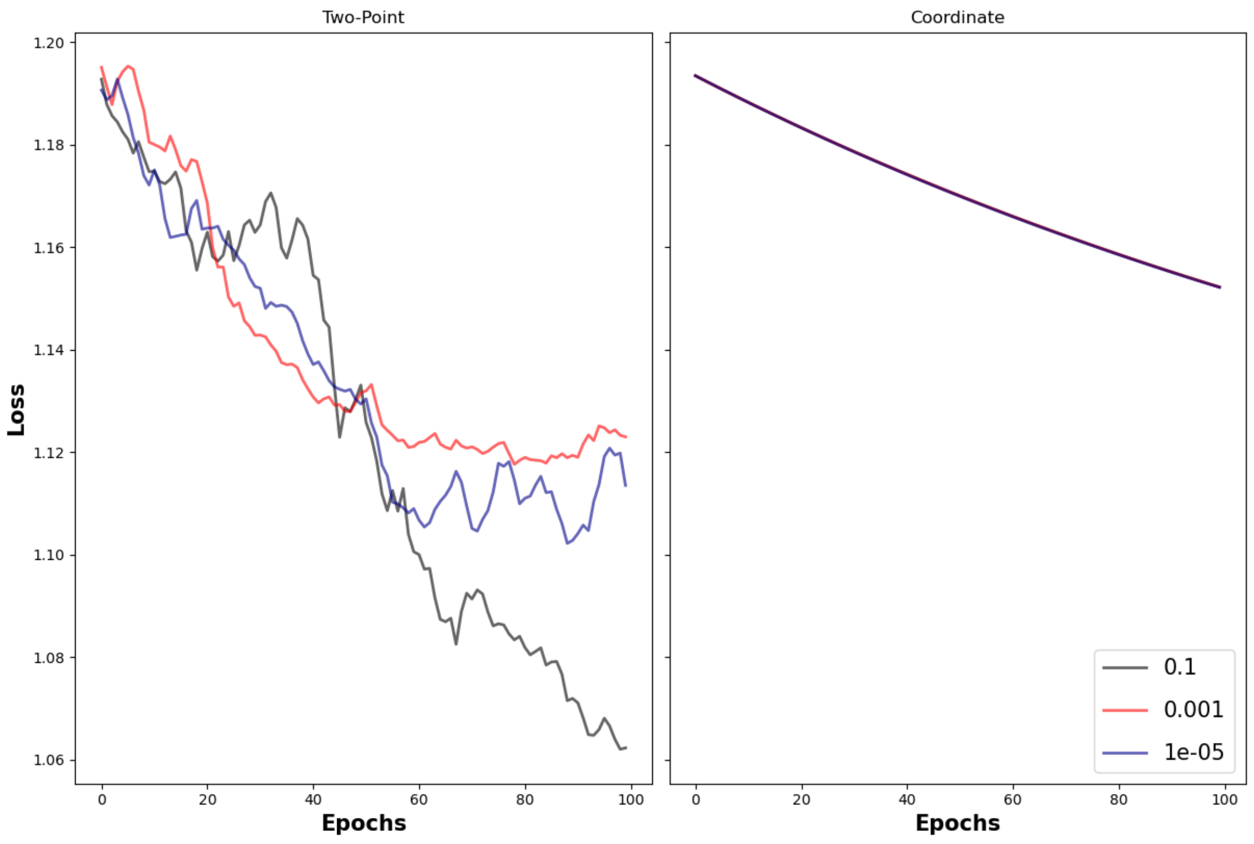

Figure 3: Relationship between the perturbation radius μ and the performance of two-point and coordinate estimation methods for ZO-SGD, varying μ over $\{10^{-1}, 10^{-3}, 10^{-5}\}$. The left plot shows the loss trajectory of the two-point method, showing small but relatively insignificant changes. The right plot shows the loss trajectory of the coordinate method, showing no noticeable changes. Overall, at these scales, the impact of μ is minimal.