Geometry-Aware Zeroth-Order Optimization for Fine-Tuning Quantized LLMs

Shaocong Ma[1], **Weidong Cai**[2], **Heng Huang**[1]
[1]Department of Computer Science, University of Maryland, College Park
[2]School of Computer Science, University of Sydney, Sydney, NSW, Australia
scma0908@gmail.com, tom.cai@sydney.edu.au, heng@umd.edu

Abstract

Zeroth-order optimization (ZOO) is a promising approach for memory-efficient fine-tuning of Large Language Models (LLMs). However, scaling ZOO to larger models on a single GPU necessitates aggressive quantization, which often destabilizes training. A primary cause of this instability is that existing methods treat strictly positive quantization scale parameters as Euclidean variables, potentially leading to boundary violations and optimization difficulties. To address this issue, we propose Hyper-Octant Zeroth-order Optimization (HoZO), a geometry-aware framework that formulates fine-tuning as Riemannian optimization on the positive orthant manifold. By constructing a novel geodesically complete metric and deriving the closed-form exponential map, HoZO inherently enforces positivity constraints without bias-inducing projections. On the theoretical side, we prove that HoZO achieves the optimal oracle complexity for zeroth-order methods. On the empirical side, HoZO outperforms baselines across six downstream tasks and three model sizes. Notably, it achieves a $34\times$ memory reduction compared to first-order fine-tuning, enabling the stable fine-tuning of a 4-bit quantized Llama-2-70b model on a single 48GB GPU.

1 Introduction

Zeroth-order optimization (ZOO) has demonstrated remarkable effectiveness in reducing the GPU memory footprint required for fine-tuning Large Language Models (LLMs) [1–14]. These gradient-free approaches enable the fine-tuning of an LLM on a single GPU with memory requirements comparable to inference-only workloads.

Despite these advancements, scaling ZOO methods to fine-tune larger models on a single GPU remains a significant challenge. For instance, a 70B parameter model typically exceeds the memory capacity of a single GPU, making standard ZOO infeasible. As a result, fine-tuning such models in resource-constrained settings requires aggressive quantization, which has motivated a growing body of literature [15, 8, 16–19].

However, existing ZOO methods primarily treat all trainable parameters as variables in an Euclidean space. This formulation is ill-suited for low-bit quantized models (*e.g.*, INT4), where the group-level scale parameters are strictly positive by definition [20–22]. As a result, standard Gaussian perturbations and gradient updates often violate these boundary constraints, necessitating additional projections. Such projections introduce bias into the gradient estimation and potentially degrade training efficiency.

To overcome this structural issue in fine-tuning highly quantized LLMs, we investigate the following core question:

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

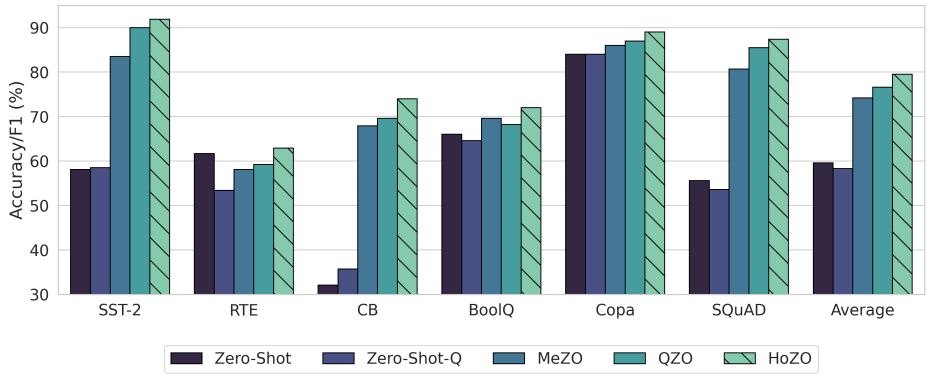


Figure 1: Performance comparison of the proposed HoZO method against baselines on the Llama-2-7b model across six downstream tasks (SST-2, RTE, CB, BoolQ, Copa, and SQuAD). HoZO consistently yields the highest Accuracy/F1 scores, outperforming all baselines across every individual task. The detailed numbers are presented in Table 2.

> ***Q:*** *Can we leverage the intrinsic geometric structure of scale parameters to formulate a projection-free ZOO method for quantized LLMs?*

To address this question, we propose **Hyper-Octant Zeroth-order Optimization (HoZO)**, a geometry-aware framework tailored for positive constraints. To mitigate the gradient estimation errors from discrete quantization noises, HoZO strategically targets the continuous scales parameters rather than the discrete weights.[1] Notably, to strictly enforce the strict-positivity constraint on these parameters, we model the scale parameter space $\mathbb{R}^d_{>0} := \{x \in \mathbb{R}^d : x_i > 0,\ \forall i\}$ as a positive orthant equipped with a specific geodesically complete metric g (defined in Section 4.1); this formulation naturally leads to an optimization problem over the Riemannian manifold. By optimizing along the geodesics of this Riemannian manifold $(\mathbb{R}^d_{>0}, g)$, HoZO naturally keeps the scale parameter dynamic within feasible regions, ensuring stable training without the need for manual clipping or boundary checks.

Our main **contributions** are summarized as follows:

1. On the theoretical side, we analytically derive the exponential map of $(\mathbb{R}^d_{>0}, g)$ in Proposition 4.1 and provide a convergence analysis in Proposition 4.3, showing that HoZO matches the optimal oracle complexity lower bound for zeroth-order methods.
2. Moreover, we design a memory-efficient implementation of HoZO by introducing a "reversed perturbation operator" (Proposition 4.4). This operator enables the MeZO-like in-place parameter recovery during training, leading to a memory footprint comparable to inference-only workloads.
3. Empirically, HoZO consistently outperforms the evaluated zeroth-order baselines across six representative downstream tasks and three model sizes (Section 5.1). As illustrated in Figure 1, HoZO achieves the best results on every individual task, with an average improvement of 19.9 ($59.6 \rightarrow 79.5$) in fine-tuning the Llama-2-7b model. Section 5.2 also demonstrates the stable training behavior under the extreme 2-bit quantization.
4. Furthermore, we show that HoZO enables fine-tuning of the 4-bit quantized Llama-2-70B model on a single 48GB GPU (Section 5.3), which achieves a $34\times$ memory reduction compared to standard fine-tuning and a $3\times$ memory reduction compared to standard zeroth-order approaches in the FP16 precision.

2 Related Work

A wide range of ZOO techniques have recently been proposed to address the memory-efficient LLM fine-tuning. We roughly summarize them as follows:

[1] This strategy has been widely applied in fine-tuning highly quantized models (*e.g.* Kim et al. [23], Shang et al. [18]) or model quantization (*e.g.* Xiao et al. [24], Lin et al. [22]).

ZOO for LLM Fine-Tuning MeZO [2] pioneered the application of ZOO to LLM fine-tuning, demonstrating significant reductions in memory requirements. Following this, Zhang et al. [5] established a comprehensive benchmark for evaluating ZO performance in this domain. Subsequent research has focused on structural efficiency: Liu et al. [25] introduced sparse MeZO to update only a subset of parameters, while Wang et al. [26] proposed a layer-wise sparse strategy to reduce computational overhead. Similarly, LOZO [9] incorporates low-rank gradient estimations to capture the inherent low-dimensional structure of LLM gradients. SubZero [11] restricts the optimization to a random low-dimensional subspace, improving convergence speed compared to full-parameter ZO methods. Other works aim to improve convergence by specific algorithmic and sampling design: Gautam et al. [6] proposed MeZO-SVRG to integrate variance reduction; Addax [12] combines first-order and ZO steps. ConMeZo [14] employs adaptive directional sampling via a momentum cone to mitigate the curse of dimensionality. Ma and Huang [27] proposes directionally aligned sampling and derives the minimum-variance condition for the two-point estimator. Beyond algorithmic design, research has expanded into systemic and hardware-aware scaling. DPZero [28] extends ZOO to differential privacy, with theoretical convergence guarantees provided by Ling et al. [7]. From a systems perspective, ZO2 [29, 30] utilizes adaptive CPU-offloading to unload non-essential components during the forward pass, enabling the fine-tuning of a 175B model on a single 24GB GPU. This framework was further extended to distributed environments by Wang et al. [31], showcasing the scalability of ZOO for massive-scale models. Additionally, HiZOO [32] estimates the diagonal Hessian to precondition zeroth-order updates, addressing heterogeneous curvature across parameter dimensions. AdaZeta [33] combines fast-forward Tensor-Train adapters with an adaptive query schedule to improve the accuracy and convergence of zeroth-order fine-tuning. RoZO [34] constrains zeroth-order perturbations to the tangent space of the LoRA low-rank manifold, whereas HoZO constructs a Riemannian geometry over positive quantization scales to preserve their positivity.

ZOO for Quantized LLMs Recent advancements in ZOO have pivoted toward the efficient fine-tuning of quantized LLMs. This line of research builds upon established first-order approaches [23, 35–38], adapting them to gradient-free regimes. QuZO [16] enables memory-efficient training in low-bit settings by utilizing a quantized zeroth-order estimator with optimized stochastic rounding. Similarly, Guo et al. [15, 17] propose a transferable static sparsity strategy, where only a minute subset of sensitive parameters is updated while the rest remain quantized. Feng et al. [8] introduces the quantized perturbation method. To bridge the precision gap inherent in quantized models, QZO [18] shifts the optimization target by perturbing continuous quantization scales instead of discrete weights. In a parallel vein, ZOQO [19] adapts sign-based gradient descent to the quantized domain via noise injection to ensure robust optimization.

3 Backgrounds

In this section, we provide the necessary background on quantized LLMs and the formulation of Riemannian ZOO.

3.1 Quantized LLMs

To load a LLM with billions of parameters on commodity hardware, aggressive quantization is often required [20]. Formally, a group of high-precision weight vector $\mathbf{w}$ can be approximated using a discrete integer vector $\mathbf{q}$ (named the *weight parameter*) and a continuous scale $x \in \mathbb{R}_+$ (named the *scale parameter*):

$$\mathbf{w} \approx \widehat{\mathbf{w}} := x \cdot \mathbf{q}, \qquad \mathbf{q} = \left\lfloor \frac{\mathbf{w}}{x} \right\rceil \tag{1}$$

where $\lfloor \cdot \rceil$ denotes rounding to the nearest integer, and $\widehat{\mathbf{w}}$ represents the de-quantized weight used for computation. As a rough rule of thumb, storing N parameters at b bits costs approximately $N \cdot b/8$ bytes; so, a 70B model would require about 140 GB in FP16 but only 35 GB in 4-bit. The following table (Table 1) summarizes these illustrative memory footprints.

The challenge appearing in fine-tuning a quantized LLM is the quantized weight $(x, \mathbf{q})$. The weight parameter $\mathbf{q}$ are discretized into low-bit integers and the scale parameter x are constrained to be strictly positive. The ZOO update rule in Equation (3) cannot be directly applied to the quantization parameter $\mathbf{q}$ due to their discrete nature. To circumvent this intractability, existing work [23, 24, 22, 18] typically

Table 1: Illustrative weight-memory footprint under different precisions for Llama-2 models [39].

Model	Params	Precision	Memory Cost
Llama-2-7B	6.7 B	4/16 bits	4/13 GB
Llama-2-13B	13 B	4/16 bits	7/26 GB
Llama-2-70B	69 B	4/16 bits	34/137 GB

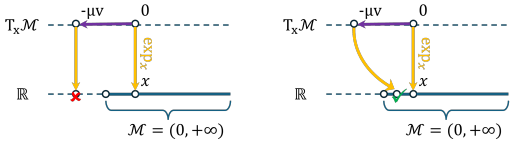


Figure 2: **(Left)** The standard Euclidean metric leads to geodesic incompleteness; applying the exponential map to a tangent vector $-\mu v \in T_x\mathcal{M}$ yields $\exp_x^{\mathrm{E}}(-\mu v) = x - \mu v$, which may fall outside $\mathcal{M} := (0, +\infty)$. **(Right)** Our HoZO formulation employs a geodesically complete Riemannian metric; its induced exponential map $\exp_x$ ensures that $\exp_x(-\mu v) \in \mathcal{M}$ for all μ.

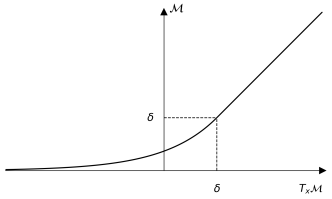


Figure 3: Illustration of the constructed exponential mapping, ensuring updates remain within the positive hyper-octant.

freezes the integer weights and optimizing the continuous scale parameter x instead, which naturally leads us to the Riemannian optimization re-formulation on the positive orthant $\mathbb{R}^d_{>0}$.

3.2 Riemannian ZOO

ZOO refers to a class of gradient-free optimization methods that rely solely on function evaluations to estimate gradients. This paradigm has recently attracted significant attention due to its memory efficiency in fine-tuning LLMs [2]. We consider the problem of minimizing the expected loss function $f(x)$ over a smooth Riemannian manifold $(\mathcal{M}, g)$:

$$\min_{x \in \mathcal{M}} f(x) := \mathbb{E}[f(x; \xi)], \tag{2}$$

where $f(\cdot; \xi) : \mathcal{M} \to \mathbb{R}$ is the individual loss associated with the data sample ξ, and $x \in \mathcal{M}$ represents the model parameters. In this work, we are particularly interested in the *positive orthant space* $\mathcal{M} = \mathbb{R}^d_{>0}$ with a geodesically complete Riemannian metric g.

In the Riemannian SGD algorithm [40], the t-th step parameter x_t is updated via the rule:

$$x_{t+1} = \exp_{x_t}\big(-\eta \nabla f(x_t; \xi_t)\big), \tag{3}$$

where $\exp_x : T_x\mathcal{M} \to \mathcal{M}$ denotes the exponential map that projects the point in the tangent space $T_x\mathcal{M}$ to the underlying manifold $\mathcal{M}$, ξ_t is the t-th step sampled data.

In the ZOO setting, the explicit gradient $\nabla f(x; \xi)$ is often unavailable; therefore, it is common to approximate it using a zeroth-order gradient estimator [41]:

$$\widehat{\nabla} f(x; \xi) = \frac{f(\exp_x(\mu v); \xi) - f(\exp_x(-\mu v); \xi)}{2\mu} v, \tag{4}$$

where $v \sim \sqrt{d}\, \mathcal{U}(\mathbb{S}_g^{d-1})$ is a scaled random perturbation vector drawn from the uniform distribution over the g-unit sphere $\mathbb{S}_g^{d-1}$ and $\mu > 0$ is a small smoothing parameter.

4 Methodology and Theoretical Analysis

In this section, we present the main methodology of the proposed HoZO method. We will formalize the geometry of the scale parameter space as a Riemannian manifold and derive the closed-form expression for the associated Riemannian zeroth-order gradient estimator in Section 4.1. We then provide the theoretical analysis which demonstrates that HoZO achieves the optimal oracle complexity in Section 4.2. Lastly, we discuss the practical implementation and considerations in Section 4.3.

4.1 Hyper-Octant Zeroth-order Optimization (HoZO)

The positive orthant $\mathbb{R}^d_{>0}$ is canonically a Riemannian submanifold of the Euclidean space $\mathbb{R}^d$ and therefore inherits the Euclidean metric g_E. However, the resulting Riemannian manifold $(\mathbb{R}^d_{>0}, g_E)$ is geodesically incomplete; that is, random smoothing perturbations $v \sim \mathcal{N}(0, I_d)$ or gradient updates with sufficiently large step sizes can easily drive some coordinates of the scale parameter x to non-positive values (*i.e.* $x_i \leq 0$), as illustrated in Figure 2. To address this geodesic incompleteness, we need to construct a Riemannian metric specifically tailored to the geometry of quantization scale parameters. Instead of relying on the standard Euclidean metric, we equip the manifold $\mathcal{M} = \mathbb{R}^d_{>0}$ with the the following Riemannian metric. For any point $x \in \mathcal{M}$ and tangent vectors $u, v \in T_x\mathcal{M}$, the metric g is defined as:

$$g_x(u, v) = u^\top G(x) v, \tag{5}$$

where $G(x) = \text{diag}\left(\max(1, \frac{\delta}{x_1})^2, \ldots, \max(1, \frac{\delta}{x_d})^2\right)$.

This metric g is continuous and piecewise smooth. Under this metric, the manifold $\mathcal{M} = \mathbb{R}^d_{>0}$ is geodesically complete and admits a closed-form exponential map, as characterized in the following proposition.

Proposition 4.1. *Let $\mathcal{M} = \mathbb{R}^d_{>0}$ be the Riemannian manifold with the metric g defined in Equation (5). Then $(\mathcal{M}, g)$ is geodesically complete. Moreover, restricted to the region where $x_i > \delta$ for all i, the exponential map $\exp_x : T_x\mathcal{M} \to \mathcal{M}$ is given by:*

$$[\exp_x(v)]_i = \begin{cases} x_i + v_i, & \text{if } x_i + v_i > \delta; \\ \delta e^{(x_i + v_i)/\delta - 1}, & \text{if } x_i + v_i \leq \delta. \end{cases} \tag{6}$$

Proof. For the sake of clarity, we present the exponential map restricted to the region where $x_i > \delta$ for all i. The complete formulation and the detailed proof are deferred to Section D.1. □

Remark 4.2. As illustrated in Figure 3, this exponential map behaves linearly at the i-th coordinate when $x_i + v_i$ exceeds the linear threshold δ. In contrast, when approaching the boundary, it warps the geometry to preserve positivity.

HoZO Gradient Estimation Leveraging this Riemannian structure, we formulate the HoZO gradient estimator by applying the standard zeroth-order estimator Equation (4), replacing the exponential map with Equation (6).

SGD Update Accordingly, the stochastic gradient update Equation (3) also follows this Riemannian structure:

$$x_{t+1} = \exp_{x_t}\left(-\eta \widehat{\nabla} f(x_t; \xi_t)\right), \tag{7}$$

where η is the learning rate.

4.2 Convergence Analysis and Oracle Complexity

In this section, we establish the convergence guarantee of our proposed HoZO method. Our analysis adopts the C^1-diffeomorphism between our constructed Riemannian manifold and the standard Euclidean space to build the equivalence between the Riemannian zeroth-order SGD dynamic and the standard zeroth-order SGD dynamic.

Proposition 4.3. *Let $\mathcal{M} = \mathbb{R}^d_{>0}$ be the Riemannian manifold with the Riemannian metric g defined in Equation (5) and $f(\cdot; \xi) : \mathcal{M} \to \mathbb{R}$ be the individual loss function associated with the stochastic data sample $\xi \sim \Xi$. Suppose that Assumptions D.2 and D.3 hold. Define the zeroth-order estimator as in Equation (4) with the exponential map defined in Equation (6). Let $\{x_t\}_{t=1}^T$ be the SGD dynamic generated by the update rule Equation (7) with requiring $\eta = \Theta(\frac{1}{\sqrt{Td}})$ and $\mu = \mathcal{O}(\frac{1}{d})$, then*

$$\min_{1 \leq t \leq T} \mathbb{E}\|\nabla f(x_t)\|^2 \leq \mathcal{O}(\sqrt{\frac{d}{T}}).$$

Moreover, to achieve the ϵ-stationary point, the required oracle complexity is $\mathcal{O}(d/\epsilon^4)$.

Algorithm 1 A Practical Implementation of Hyper-Octant Zeroth-order Optimization (HoZO)

Require: Scale parameters $x \in \mathbb{R}_+^d$, loss $f : \mathbb{R}_+^d \to \mathbb{R}$, learning rate η, optimization steps T, perturbation stepsize μ.

for $t = 1, \ldots, T$ **do**
  Sample a batch ξ and a random seed s
  `Perturbation`(x, μ, s) ▷ `Perturb parameters` $x \leftarrow \exp_x(\mu v)$ `in place`
  $\ell_+ \leftarrow f(x; \xi)$
  `Reversed_Perturbation`$(x, -\mu, s)$ ▷ `Reset parameters in place`
  `Perturbation`$(x, -\mu, s)$
  $\ell_- \leftarrow f(x; \xi)$
  `Reversed_Perturbation`(x, μ, s)

  `projected_grad` $\leftarrow (\ell_+ - \ell_-)/(2\mu)$
  Reset random number generator with seed s ▷ `For sampling the same` v
  for $x_i \in x$ **do**
    Sample a random direction v_i from the scaled sphere
    $x_i \leftarrow \exp_{x_i}(-\eta * \texttt{projected_grad} * v_i)$
  end for
end for

Subroutine 1: `Perturbation`

Require: $x \in \mathbb{R}_+^d$, stepsize $\widehat{\mu}$, random seed s

Reset random number generator with seed s
for $x_i \in x$ **do**
  Sample a random direction v_i from the scaled sphere
  $x_i \leftarrow \exp_{x_i}(\widehat{\mu} v_i)$
end for

Subroutine 2: `Reversed_Perturbation`

Require: $x \in \mathbb{R}_+^d$, stepsize $\widehat{\mu}$, random seed s

Reset random number generator with seed s
for $x_i \in x$ **do**
  Sample a random direction v_i from the scaled sphere
  $x_i \leftarrow \mathrm{rev}_{x_i}(\widehat{\mu} v_i)$
end for

Proof. The Riemannian metric defined in Equation (5) is continuous but only piecewise smooth; therefore, the convergence analysis for the Riemannian zeroth-order SGD algorithm cannot be directly applied. To address this issue, we employ a reparameterization trick. We construct a global C^1-diffeomorphism[2] in Equation (12) between the Riemannian manifold and the Euclidean space. Under this mapping, the Riemannian metric is the pullback of the Euclidean metric. Therefore, the convergence analysis of standard zeroth-order SGD algorithm in the Euclidean space directly implies convergence on our Riemannian manifold. A detailed proof is provided in Section D.2. □

Optimal Complexity We note that the achieved complexity matches the established lower bound for smooth non-convex optimization using zeroth-order methods in the Euclidean setting [42]. It is also consistent with the complexity found in recent Riemannian zeroth-order literature [41].

4.3 Practical Implementation and Considerations

The HoZO algorithm is presented in Algorithm 1. In this section, we additionally discuss the consideration is memory efficiency. Specifically, the implementation of MeZO [2] achieves memory efficiency by recovering original parameters via *in-place perturbations*, as illustrated below:

Example 1 (Reversibility in the Euclidean case). Let $x \in \mathbb{R}^d$ be the model parameter. Suppose that we take the following operations in sequence: *(i)* Forward perturbation: $x_{\texttt{tmp}} \leftarrow x + \mu v$. *(ii)* Backward perturbation: $x_{\texttt{tmp}} \leftarrow x_{\texttt{tmp}} - 2\mu v$. *(iii)* Recovery: $x' \leftarrow x_{\texttt{tmp}} + \mu v$. Then the final output x' is identical to x.

However, this property does not hold trivially in the Riemannian setting due to the nonlinearity of the exponential map $\exp_x$. To address this challenge and maintain the same memory efficiency as MeZO,

[2]We note that the C^1-diffeomorphism will induce a C^0-Riemannian metric instead of a C^1-Riemannian metric, which is aligned with the continuity of Equation (5).

Table 2: Performance comparison of HoZO against baseline methods on the Llama-2 model family (7B, 13B, and 70B) [39] across six downstream tasks. We report the classification accuracy and the F1 score on the test set. The "Average" column denotes the mean score across all six tasks, and the rightmost column highlights the absolute improvement (↑) or degradation (↓) relative to the FP16 Zero-Shot baseline. The highest score in each task is highlighted in **bold** and underlined.
⋆: Results are taken from Shang et al. [18].

Model	Method	Model Precision	SST-2	RTE	CB	BoolQ	Copa	SQuAD	Average	
Llama-2-7b	Zero-Shot	16 bits	58.1⋆	61.7⋆	32.1⋆	66.0⋆	84.0	55.6⋆	*59.6*	-
	Zero-Shot-Q	4 bits	58.5⋆	53.4⋆	35.7⋆	64.6⋆	84.0	53.6⋆	*58.3*	↓1.3
	MeZO	16 bits	83.5⋆	58.1⋆	67.9⋆	69.6⋆	86.0	80.7⋆	*74.2*	↑14.6
	QZO	4 bits	90.0⋆	59.2⋆	69.6⋆	68.2⋆	87.0	85.5⋆	*76.6*	↑17.0
	HoZO	4 bits	**91.9**	**62.9**	**74.0**	**72.0**	**89.0**	**87.4**	***79.5***	↑19.9
Llama-2-13b	Zero-Shot	16 bits	61.1	50.9	44.0	74.1	89.0	63.7	*63.8*	-
	Zero-Shot-Q	4 bits	60.0	47.3	47.0	74.7	87.0	63.1	*63.2*	↓0.6
	MeZO	16 bits	90.7	58.5	**77.0**	81.6	**92.0**	87.5	*81.2*	↑17.4
	QZO	4 bits	91.9	62.8	**77.0**	**82.4**	**92.0**	89.2	*82.5*	↑18.7
	HoZO	4 bits	**92.4**	**68.6**	75.0	81.6	**92.0**	**89.3**	***83.2***	↑19.4
Llama-2-70b	Zero-Shot-Q	4 bits	56.4	60.6	47.0	74.7	92.0	71.4	*67.0*	-
	QZO	4 bits	90.6	**80.8**	82.0	**83.8**	93.0	90.4	*86.8*	↑19.8
	HoZO	4 bits	**91.5**	79.1	**83.0**	**83.8**	**95.0**	**91.2**	***87.3***	↑20.3

we introduce a specific subroutine, the `Reversed_Perturbation` (Subroutine 2, Algorithm 1), defined as:

$$[\mathrm{rev}_y(v)]_i := \begin{cases} y_i - v_i & \text{if } y_i > \delta; \\ \delta(1 + \ln \frac{y_i}{\delta}) - v_i & \text{if } y_i \le \delta. \end{cases} \tag{8}$$

This operator guarantees exact recovery of the original parameters in exact arithmetic, as established below.

Proposition 4.4. *Let $\mathcal{M} = \mathbb{R}^d_{>0}$ be the Riemannian manifold with the metric g defined in Equation (5). Suppose that x satisfies that $x_i > \delta$ and $v \in T_x\mathcal{M}$ is a fixed vector. If $y = \exp_x(v)$ and $x' = \mathrm{rev}_y(v)$, then $x' = x$.*

Proof. The proof is provided in Section D.3. □

Proposition 4.4 establishes exact recovery in exact arithmetic. Our practical implementation performs perturbation normalization and logarithmic recovery calculations in FP32, then casts results back to the scale tensor's original dtype. For logarithm evaluation, scale values are floored at 10^{-7} to avoid $\log 0$. Floating-point rounding and this floor, when active, can introduce recovery errors; bitwise equality is therefore not guaranteed. With this formulation, HoZO matches the memory efficiency of QZO [18] while strictly satisfying the positivity constraint, which is empirically validated in Section 5.3. Furthermore, the additional perturbation step and nonlinearity incur negligible computational overhead, as demonstrated in Section 5.4.

5 Experiments

In this section, we empirically validate the performance of HoZO on fine-tuning quantized LLMs.

5.1 HoZO Scaling in the Model Size

In this section, we validate the scalability and performance of HoZO as the model size increases from 7B to 70B. We summarize our experimental setup and observations below.

Models and Datasets To evaluate the scalability of our approach, we employ the Llama-2 model family [39] across three scales: Llama-2-7B, Llama-2-13B, and Llama-2-70B. We conduct evaluations on six representative NLP tasks: SST-2 [44] for sentiment analysis; RTE [45, 46], CB [47], and BoolQ [48] for natural language inference; COPA [49] for causal reasoning; and SQuAD [50] for

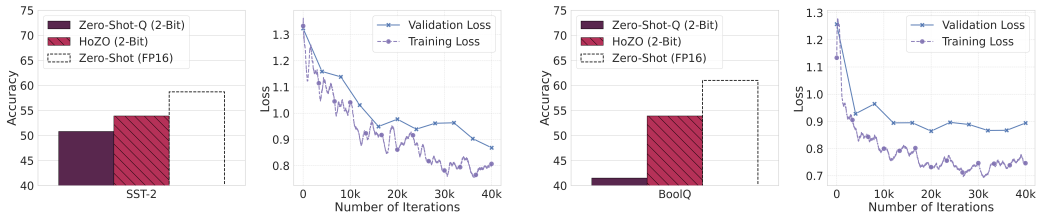


Figure 4: Experimental results for Llama-3-8b [43] on SST-2 (left pair) and BoolQ (right pair). The loss curves demonstrate that HoZO achieves stable training under extreme 2-bit quantization. In terms of accuracy, HoZO outperforms the Zero-Shot-Q (2-bit) baseline, although a gap remains compared to the unquantized (FP16) model. See Section C.2 for other tasks.

reading comprehension. Following prior work [2, 5, 18], we construct datasets by randomly sampling 1,000 training, 500 validation, and 1,000 testing examples; the exact same data splits are used for all experiments. We report accuracy for classification tasks and the F1 score for generation tasks. Throughout our experiments, we employ GPTQ [21] to quantize all models used in our experiments and follow the same setting as used in Shang et al. [18]. For more details and the hyperparameter setting, we refer to Section B.2.

Baseline Methods We compare our proposed HoZO method against the following baselines: (1) **Zero-Shot** and **Zero-Shot-Q**, the pre-trained FP16 and 4-bit quantized models without fine-tuning; (2) **MeZO** [2], optimization algorithm applied directly to the FP16 model; and (3) **QZO** [18], A state-of-the-art zeroth-order fine-tuning framework specifically designed for memory-efficient training of quantized models. We note that LoRA/QLoRA optimize low-rank adapters, QuZO/ZOQO update quantized weights, and QZO/HoZO optimize quantization scales. These methods use different parameterizations, with QZO serving as our most directly comparable baseline. Our empirical conclusions are limited to the evaluated methods and experimental settings.

Results Table 2 presents the experimental results across different model scales. We observe the following key findings: *(i) Scalability to 70B Models:* HoZO successfully enables the fine-tuning of the Llama-2-70B model on a single 48 GB GPU. Whereas standard Fine-Tuning and FP16 MeZO are infeasible due to GPU memory constraints, HoZO maintains stability and memory efficiency. *(ii) Consistent Performance Improvements:* HoZO consistently outperforms both zero-shot baselines and existing zeroth-order approaches across nearly all model sizes and tasks. On Llama-2-7B, it achieves the best performance on all individual tasks. *(iii) Bridging the Quantization Gap:* Despite operating on 4-bit quantized weights, HoZO matches or exceeds the performance of full-precision zeroth-order optimization methods. These findings demonstrate that the proposed method HoZO effectively mitigates the quantization gap while significantly reducing the computational and memory overhead required for training large-scale LLMs.

5.2 Stability of HoZO in the Extreme Quantization

To evaluate the stability of HoZO under extreme quantization, we apply INT2 quantization to the Llama-3-8b model on the SST-2 classification task. We compare the fine-tuned HoZO model against the direct post-training quantization baseline (Zero-Shot-Q) and the original FP16 baseline.

Results The results are presented in Figure 4. We observe two key findings: *(i) HoZO maintains training stability in 2-bit quantization:* As shown in Figure 4, both training and validation loss decrease consistently without divergence, demonstrating that HoZO can effectively optimize the landscape even in extreme 2-bit settings. *(ii) HoZO improves over direct quantization but remains below FP16 performance:* Compared to Zero-Shot-Q (2-bit) baseline, HoZO achieves consistently better performance; however, the 2-bit model does not fully recover the Zero-Shot (FP16) performance. This gap suggests that the information loss inherent in 2-bit quantization creates a capacity bottleneck that scale optimization alone cannot fully overcome.

5.3 Peak Memory Usage

This section evaluates the peak memory usage of the proposed HoZO method across varying quantization levels, comparing it against Zero-Shot-Q (4-bit), Zero-Shot (16-bit), standard Fine-Tuning (16-bit), and MeZO (16-bit). We evaluate Llama-2 models [39] of varying scales on NVIDIA RTX A6000 GPUs (48GB memory) using the SST-2 dataset [44]. For the multi-GPU setting, we employed the Fully Sharded Data Parallel (FSDP) implementation provided by PyTorch [51]. We report the peak GPU memory consumption in Figure 5 (see Section B.4 for profiling details).

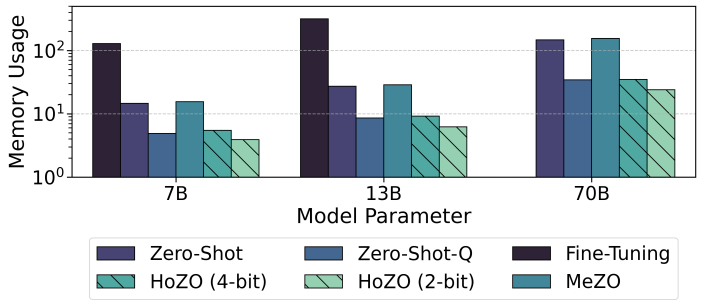


Figure 5: GPU peak memory consumption for Llama-2 model family [39] with different sizes (7B, 13B, and 70B) and methods on the SST-2 dataset [44]. HoZO achieves up to $34\times$ and $3\times$ memory reductions compared to standard fine-tuning and MeZO.

Results As illustrated in Figure 5 (refer to Section C.1 for detailed numbers), HoZO (4-bit) achieves the comparable memory efficiency as the Zero-Shot-Q method while offering up to $34\times$ *and* $3\times$ *memory reductions* compared to standard Fine-Tuning and MeZO in FP16 precision, respectively. Specifically, HoZO enables fine-tuning a 70B model on a single 48 GB A6000 GPU, whereas MeZO and standard fine-tuning exceed this memory capacity.

5.4 Convergence: Loss vs Wall-Clock Time

In this section, we evaluate the convergence rate and wall-clock time efficiency of the proposed HoZO method, comparing it against existing zeroth-order approaches including QZO (4-bit) and MeZO (16-bit). We evaluate the training dynamics of fine-tuning the Llama-2-13B model using the BoolQ dataset [48]. We report the training loss trajectories against both the number of iteration steps and total wall-clock time. The wall-clock time cost is evaluated on the NVIDIA RTX A6000 GPU platform to ensure fair comparison. To derive the time-based comparison, we calculate the averaged time cost per iteration for each method and scale the training loss curve accordingly (refer to Section C.3 for detailed numbers).

Results Despite higher per-step time cost caused by the additional perturbation operation and the non-linearity of perturbation operator, HoZO demonstrates superior convergence efficiency by employing the geometry-aware estimator that respects the intrinsic structure of the positive orthant space. As shown in Figure 6, HoZO reaches lower loss significantly faster than baselines, achieving *speedups of* $3\times$ *over MeZO and* $1.4\times$ *over QZO* (in wall-clock time).

5.5 Ablation Study: Necessity of Reversed Perturbation

Now we investigate the necessity of the reversed perturbation step in Subroutine 2 (Algorithm 1). In our proposed implementation, we recover the original parameter by adopting an explicitly reversed perturbation. To validate its effectiveness, we consider the traditional MeZO-like perturbation scheme (Example 1): *(i)* Forward perturbation: $x_{\texttt{tmp}} \leftarrow \exp_x(\mu v)$. *(ii)* Backward perturbation: $x_{\texttt{tmp}} \leftarrow \exp_{x_{\texttt{tmp}}}(-2\mu v)$. *(iii)* Recovery: $x' \leftarrow \exp_{x_{\texttt{tmp}}}(\mu v)$. In general, the final output x' is not equal to x.

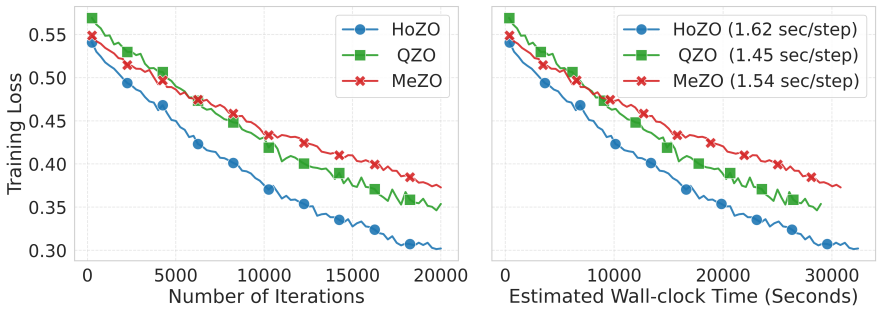


Figure 6: Comparison of the training loss of HoZO against QZO [18] and MeZO [2] fine-tuning on the BoolQ dataset [48]. **(Left)** Training loss versus number of iterations. **(Right)** Training loss versus wall-clock time. HoZO demonstrates superior convergence efficiency in terms of total training time.

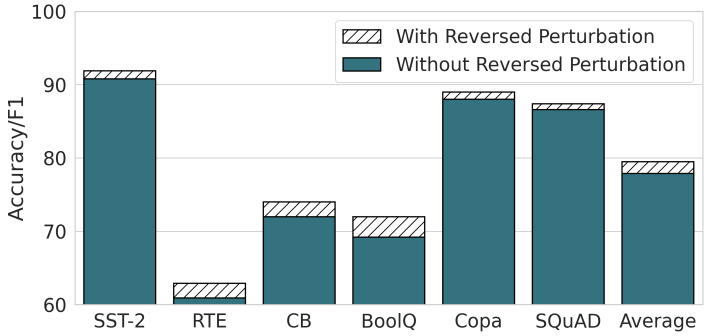


Figure 7: Impact of removing the reversed perturbation. This figure illustrates the performance drop in test set performance across the six downstream tasks used in Section 5.1 (Accuracy for Classification Tasks and F1 for SQuAD), relative to our proposed HoZO implementation.

Results We observe that this alternative approach leads to significant performance degradation compared to standard baselines, as illustrated in Figure 7. Consequently, the non-reversed operation appears to introduce bias. In contrast, our proposed method accurately recovers the original parameter, resulting in superior performance.

6 Conclusion

We have shown that HoZO can effectively fine-tune quantized LLMs across various tasks and scales, specifically enabling the training of 70B parameter models on a single 48GB GPU. Further experiments suggest that HoZO consistently outperforms the evaluated zeroth-order baselines in both accuracy and convergence speed by rigorously respecting the positive orthant geometry of quantization scales. Our theory illustrates that HoZO maintains the optimal sample complexity of zeroth-order methods which matches the theoretical lower bound in the Euclidean case. As a limitation, HoZO incurs a slightly higher computational cost per step due to the nonlinear exponential mapping, though we show that the significant reduction in total optimization steps leads to superior wall-clock efficiency compared to standard methods. We did not fully explore integrating HoZO with advanced adaptive optimizers like Adam for the PEFT module or combining it with other compression techniques, though we hope to investigate this in the future.

Acknowledgements

This work was partially supported by NSF IIS 2347592, 2348169, DBI 2405416, CCF 2348306, CNS 2347617, RISE 2536663.

References

[1] Tianxiang Sun, Zhengfu He, Hong Qian, Yunhua Zhou, Xuan-Jing Huang, and Xipeng Qiu. Bbtv2: Towards a gradient-free future with large language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 3916–3930, 2022.

[2] Sadhika Malladi, Tianyu Gao, Eshaan Nichani, Alex Damian, Jason D Lee, Danqi Chen, and Sanjeev Arora. Fine-tuning language models with just forward passes. *Advances in Neural Information Processing Systems*, 36:53038–53075, 2023.

[3] Xinyu Tang, Ashwinee Panda, Milad Nasr, Saeed Mahloujifar, and Prateek Mittal. Private fine-tuning of large language models with zeroth-order optimization. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856. URL https://openreview.net/forum?id=3Y3o0yFZfu.

[4] Feihu Jin, Yifan Liu, and Ying Tan. Derivative-free optimization for low-rank adaptation in large language models. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2024.

[5] Yihua Zhang, Pingzhi Li, Junyuan Hong, Jiaxiang Li, Yimeng Zhang, Wenqing Zheng, Pin-Yu Chen, Jason D Lee, Wotao Yin, Mingyi Hong, et al. Revisiting zeroth-order optimization for memory-efficient llm fine-tuning: A benchmark. In *International Conference on Machine Learning*, pages 59173–59190. PMLR, 2024.

[6] Tanmay Gautam, Youngsuk Park, Hao Zhou, Parameswaran Raman, and Wooseok Ha. Variance-reduced zeroth-order methods for fine-tuning language models. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=VHO4nE7v41.

[7] Zhenqing Ling, Daoyuan Chen, Liuyi Yao, Yaliang Li, and Ying Shen. On the convergence of zeroth-order federated tuning for large language models. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1827–1838, 2024.

[8] Chen Feng, Shaojie Zhuo, Xiaopeng Zhang, Ramchalam K Ramakrishnan, Zhaocong Yuan, and Andrew Z Li. Stepping forward on the last mile. *Advances in Neural Information Processing Systems*, 37:94851–94870, 2024.

[9] Yiming Chen, Yuan Zhang, Liyuan Cao, Kun Yuan, and Zaiwen Wen. Enhancing zeroth-order fine-tuning for language models with low-rank structures. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=9BiVepgmWW.

[10] Yifan Yang, Kai Zhen, Ershad Banijamali, Athanasios Mouchtaris, and Zheng Zhang. Adazeta: Adaptive zeroth-order tensor-train adaption for memory-efficient large language models fine-tuning. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 977–995, 2024.

[11] Ziming Yu, Pan Zhou, Sike Wang, Jia Li, Mi Tian, and Hua Huang. Zeroth-order fine-tuning of llms in random subspaces. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4475–4485, 2025.

[12] Zeman Li, Xinwei Zhang, Peilin Zhong, Yuan Deng, Meisam Razaviyayn, and Vahab Mirrokni. Addax: Utilizing zeroth-order gradients to improve memory efficiency and performance of SGD for fine-tuning language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=QhxjQOMdDF.

[13] Xiaoyu Zhang, Yong Lin, Meixia Miao, Jian Lou, Jin Li, and Xiaofeng Chen. Zeroth-order federated private tuning for pretrained large language models. In *Australasian Conference on Information Security and Privacy*, pages 285–306. Springer, 2025. URL https://link.springer.com/chapter/10.1007/978-981-96-9101-2_15.

[14] Lejs Deen Behric, Liang Zhang, Bingcong Li, and Kiran Koshy Thekumparampil. ConmeZO: Adaptive directional sampling for gradient-free finetuning of language models. In *ES-FoMo III: 3rd Workshop on Efficient Systems for Foundation Models*, 2025. URL https://openreview.net/forum?id=qUIMnrYEPU.

[15] Wentao Guo, Jikai Long, Yimeng Zeng, Zirui Liu, Xinyu Yang, Yide Ran, Jacob R Gardner, Osbert Bastani, Christopher De Sa, Xiaodong Yu, et al. Zeroth-order fine-tuning of llms with extreme sparsity. In *2nd Workshop on Advancing Neural Network Training: Computational Efficiency, Scalability, and Resource Optimization (WANT@ ICML 2024)*, 2024.

[16] Jiajun Zhou, Yifan Yang, Kai Zhen, Ziyue Liu, Yequan Zhao, Ershad Banijamali, Athanasios Mouchtaris, Ngai Wong, and Zheng Zhang. Quzo: Quantized zeroth-order fine-tuning for large language models. *arXiv preprint arXiv:2502.12346*, 2025.

[17] Wentao Guo, Jikai Long, Yimeng Zeng, Zirui Liu, Xinyu Yang, Yide Ran, Jacob R. Gardner, Osbert Bastani, Christopher De Sa, Xiaodong Yu, Beidi Chen, and Zhaozhuo Xu. Zeroth-order fine-tuning of LLMs with transferable static sparsity. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=myYzr50xBh.

[18] Sifeng Shang, Jiayi Zhou, Chenyu Lin, Minxian Li, and Kaiyang Zhou. Fine-tuning quantized neural networks with zeroth-order optimization. *arXiv preprint arXiv:2505.13430*, 2025.

[19] Noga Bar and Raja Giryes. Zoqo: Zero-order quantized optimization. In *ICASSP 2025-2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2025.

[20] Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2704–2713, 2018.

[21] Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. OPTQ: Accurate quantization for generative pre-trained transformers. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=tcbBPnfwxS.

[22] Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. Awq: Activation-aware weight quantization for on-device llm compression and acceleration. *Proceedings of machine learning and systems*, 6:87–100, 2024.

[23] Jeonghoon Kim, Jung Hyun Lee, Sungdong Kim, Joonsuk Park, Kang Min Yoo, Se Jung Kwon, and Dongsoo Lee. Memory-efficient fine-tuning of compressed large language models via sub-4-bit integer quantization. *Advances in Neural Information Processing Systems*, 36: 36187–36207, 2023.

[24] Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. Smoothquant: Accurate and efficient post-training quantization for large language models. In *International conference on machine learning*, pages 38087–38099. PMLR, 2023.

[25] Yong Liu, Zirui Zhu, Chaoyu Gong, Minhao Cheng, Cho-Jui Hsieh, and Yang You. Sparse mezo: Less parameters for better performance in zeroth-order llm fine-tuning. *arXiv preprint arXiv:2402.15751*, 2024.

[26] Fei Wang, Li Shen, Liang Ding, Chao Xue, Ye Liu, and Changxing Ding. Simultaneous computation and memory efficient zeroth-order optimizer for fine-tuning large language models. *arXiv preprint arXiv:2410.09823*, 2024.

[27] Shaocong Ma and Heng Huang. Revisiting zeroth-order optimization: Minimum-variance two-point estimators and directionally aligned perturbations. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=ywFOSIT9ik.

[28] Liang Zhang, Bingcong Li, Kiran Koshy Thekumparampil, Sewoong Oh, and Niao He. DPZero: Private fine-tuning of language models without backpropagation. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=QJkG8Mln72.

[29] Liangyu Wang, Jie Ren, Hang Xu, Junxiao Wang, Huanyi Xie, David E. Keyes, and Di Wang. Scalable zeroth-order fine-tuning for extremely large language models with limited GPU memory. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=s0p9xpORgP.

[30] Liangyu Wang, Jie Ren, Hang Xu, Junxiao Wang, Huanyi Xie, David E Keyes, and Di Wang. Zo2: Scalable zeroth-order fine-tuning for extremely large language models with limited gpu memory. *arXiv preprint arXiv:2503.12668*, 2025.

[31] Liangyu Wang, Huanyi Xie, and Di Wang. Distzo2: High-throughput and memory-efficient zeroth-order fine-tuning llms with distributed parallel computing. *arXiv preprint arXiv:2507.03211*, 2025. URL https://arxiv.org/abs/2507.03211.

[32] Yanjun Zhao, Sizhe Dang, Haishan Ye, Guang Dai, Yi Qian, and Ivor Tsang. Second-order fine-tuning without pain for LLMs: A hessian informed zeroth-order optimizer. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=bEqI61iBue.

[33] Yifan Yang, Kai Zhen, Ershad Banijamali, Athanasios Mouchtaris, and Zheng Zhang. AdaZeta: Adaptive zeroth-order tensor-train adaption for memory-efficient large language models fine-tuning. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 977–995, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.56. URL https://aclanthology.org/2024.emnlp-main.56/.

[34] Zichen Song and Weijia Li. RoZO: Geometry-aware zeroth-order fine-tuning on low-rank adapters for black-box large language models. In Vera Demberg, Kentaro Inui, and Lluís Marquez, editors, *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1773–1783, Rabat, Morocco, March 2026. Association for Computational Linguistics. ISBN 979-8-89176-380-7. doi: 10.18653/v1/2026.eacl-long.80. URL https://aclanthology.org/2026.eacl-long.80/.

[35] Haocheng Xi, Changhao Li, Jianfei Chen, and Jun Zhu. Training transformers with 4-bit integers. *Advances in Neural Information Processing Systems*, 36:49146–49168, 2023.

[36] Changhun Lee, Jungyu Jin, Taesu Kim, Hyungjun Kim, and Eunhyeok Park. Owq: Outlier-aware weight quantization for efficient fine-tuning and inference of large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 13355–13364, 2024.

[37] Changhun Lee, Jun-gyu Jin, YoungHyun Cho, and Eunhyeok Park. Qeft: Quantization for efficient fine-tuning of llms. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 13823–13837, 2024.

[38] Jahid Hasan. Optimizing large language models through quantization: A comparative analysis of ptq and qat techniques. *arXiv preprint arXiv:2411.06084*, 2024.

[39] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

[40] Silvere Bonnabel. Stochastic gradient descent on riemannian manifolds. *IEEE Transactions on Automatic Control*, 58(9):2217–2229, 2013.

[41] Jiaxiang Li, Krishnakumar Balasubramanian, and Shiqian Ma. Stochastic zeroth-order riemannian derivative estimation and optimization. *Mathematics of Operations Research*, 48(2):1183–1211, 2023.

[42] John C Duchi, Michael I Jordan, Martin J Wainwright, and Andre Wibisono. Optimal rates for zero-order convex optimization: The power of two function evaluations. *IEEE Transactions on Information Theory*, 61(5):2788–2806, 2015.

[43] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

[44] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642, 2013.

[45] Ido Dagan, Oren Glickman, and Bernardo Magnini. The pascal recognising textual entailment challenge. In *Machine learning challenges workshop*, pages 177–190. Springer, 2005.

[46] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP workshop BlackboxNLP: Analyzing and interpreting neural networks for NLP*, pages 353–355, 2018.

[47] Marie-Catherine De Marneffe, Mandy Simons, and Judith Tonhauser. The commitmentbank: Investigating projection in naturally occurring discourse. In *proceedings of Sinn und Bedeutung*, volume 23, pages 107–124, 2019.

[48] Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. Boolq: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2924–2936, 2019.

[49] Melissa Roemmele, Cosmin Adrian Bejan, and Andrew S Gordon. Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In *AAAI spring symposium: logical formalizations of commonsense reasoning*, pages 90–95, 2011.

[50] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, 2016.

[51] Yanli Zhao, Andrew Gu, Rohan Varma, Liang Luo, Chien-Chin Huang, Min Xu, Less Wright, Hamid Shojanazeri, Myle Ott, Sam Shleifer, et al. Pytorch fsdp: experiences on scaling fully sharded data parallel. *arXiv preprint arXiv:2304.11277*, 2023.

[52] Hongyi Zhang and Suvrit Sra. First-order methods for geodesically convex optimization. In *Conference on learning theory*, pages 1617–1638. PMLR, 2016.

[53] Jiaxiang Li, Krishnakumar Balasubramanian, and Shiqian Ma. Zeroth-order riemannian averaging stochastic approximation algorithms. *arXiv preprint arXiv:2309.14506*, 2023.

[54] Vyacheslav Kungurtsev, Francesco Rinaldi, and Damiano Zeffiro. Retraction-based direct search methods for derivative free riemannian optimization. *Journal of Optimization Theory and Applications*, 2023.

[55] Teng-Teng Yao, Zhi Zhao, Zheng-Jian Bai, and Xiao-Qing Jin. A riemannian derivative-free polak–ribière–polyak method for tangent vector field. *Numerical Algorithms*, 86(1):325–355, 2021.

[56] Shaocong Ma and Heng Huang. Riemannian zeroth-order gradient estimation with structure-preserving metrics for geodesically incomplete manifolds. *arXiv preprint arXiv:2601.08039*, 2025. URL http://arxiv.org/abs/2601.08039.

[57] Yurii Nesterov and Vladimir Spokoiny. Random gradient-free minimization of convex functions. *Foundations of Computational Mathematics*, 17:527–566, 2017.

[58] Saeed Ghadimi and Guanghui Lan. Stochastic first-and zeroth-order methods for nonconvex stochastic programming. *SIAM journal on optimization*, 23(4):2341–2368, 2013.

[59] Kaiyi Ji, Zhe Wang, Yi Zhou, and Yingbin Liang. Improved zeroth-order variance reduced algorithms and analysis for nonconvex optimization. In *International conference on machine learning*, pages 3100–3109. PMLR, 2019.

[60] Albert S Berahas, Liyuan Cao, Krzysztof Choromanski, and Katya Scheinberg. A theoretical and empirical comparison of gradient approximations in derivative-free optimization. *Foundations of Computational Mathematics*, 22(2):507–560, 2022.

[61] Guanting Chen. Unbiased gradient simulation for zeroth-order optimization. In *2020 Winter Simulation Conference (WSC)*, pages 2947–2959. IEEE, 2020.

[62] Shaocong Ma and Heng Huang. On the optimal construction of unbiased gradient estimators for zeroth-order optimization. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=rVT1GK60Nt.

[63] Damek Davis, Dmitriy Drusvyatskiy, Yin Tat Lee, Swati Padmanabhan, and Guanghao Ye. A gradient sampling method with complexity guarantees for lipschitz functions in high and low dimensions. *Advances in Neural Information Processing Systems*, 35:6692–6703, 2022.

[64] Jingzhao Zhang, Hongzhou Lin, Stefanie Jegelka, Suvrit Sra, and Ali Jadbabaie. Complexity of finding stationary points of nonconvex nonsmooth functions. In *International Conference on Machine Learning*, pages 11173–11182. PMLR, 2020.

[65] Tianyi Lin, Zeyu Zheng, and Michael Jordan. Gradient-free methods for deterministic and stochastic nonsmooth nonconvex optimization. *Advances in Neural Information Processing Systems*, 35:26160–26175, 2022.

[66] Marco Rando, Cesare Molinari, Lorenzo Rosasco, and Silvia Villa. An optimal structured zeroth-order algorithm for non-smooth optimization. *Advances in Neural Information Processing Systems*, 36, 2024.

[67] Guy Kornowski and Ohad Shamir. An algorithm with optimal dimension-dependence for zero-order nonsmooth nonconvex stochastic optimization. *Journal of Machine Learning Research*, 25(122):1–14, 2024.

[68] Michael P Connolly, Nicholas J Higham, and Theo Mary. Stochastic rounding and its probabilistic backward error analysis. *SIAM Journal on Scientific Computing*, 43(1):A566–A585, 2021.

Appendix

Table of Contents

A Other Related Work **16**
- A.1 Related Theoretical Results . . . 16
- A.2 Connections to Existing Results . . . 17

B Experimental Setups **17**
- B.1 Hardware and System Environment . . . 17
- B.2 More Detailed Experimental Setups and Hyperparameters in Section 5.1 . . . 18
- B.3 The 2-Bit Extreme Quantization . . . 18
- B.4 Memory Profiling . . . 18
- B.5 Loss vs Wall-Clock Time . . . 18

C More Experimental Results **19**
- C.1 Memory Profiling . . . 19
- C.2 The 2-Bit Extreme Quantization . . . 19
- C.3 Loss vs Wall-Clock Time . . . 19

D Theoretical Results **19**
- D.1 Proof of Proposition 4.1 . . . 20
- D.2 Proof of Proposition 4.3 . . . 22
  - D.2.1 Assumptions and Justifications . . . 23
  - D.2.2 Supporting Lemmas . . . 23
  - D.2.3 Main Proof of Proposition 4.3 . . . 25
- D.3 Proof of Proposition 4.4 . . . 26

E Other Experiments **27**
- E.1 Results across Random Seeds . . . 27
- E.2 Comparison with 70B Zero-Shot Performance . . . 27
- E.3 Evaluation on Additional Models . . . 28
- E.4 Comparisons with Additional Baselines . . . 28

F Limitations **28**

A Other Related Work

A.1 Related Theoretical Results

Our theoretical analysis builds upon the existing literature on ZOO, which we summarize below.

Riemannian ZOO Riemannian first-order optimization extends gradient-based methods to Riemannian manifolds. For geodesically convex functions, Riemannian gradient descent matches the convergence guarantees of Euclidean gradient descent. Furthermore, Zhang and Sra [52] established global complexity bounds on Hadamard manifolds with curvature-dependent rates, while stochastic Riemannian gradient descent converges almost surely under standard assumptions [40]. When explicit gradients are unavailable, zeroth-order methods are required to estimate descent directions via sampling. To this end, Li et al. [53, 41] applied Gaussian smoothing in tangent spaces using exponential maps to construct unbiased gradient estimators, establishing variance bounds that depend on curvature and dimension. Other derivative-free approaches including retraction-based direct search methods [54] and Ribiére-Polyak conjugate gradient method [55] also are shown to have convergence

guarantees. Ma and Huang [56] recently proposes the intrinsic convergence analysis of Riemannian zeroth-order SGD without treating the Riemannian metric g as the pullback of an Euclidean metric g_E, which further builds the convergence guarantee of Riemannian zeroth-order SGD for geodesically incomplete Riemannian manifolds.

Theoretical Analysis of ZOO The convergence properties of ZOO have been extensively characterized across various settings. For smooth convex objectives, Duchi et al. [42] and Nesterov and Spokoiny [57] established optimal convergence bounds for general ZOO and Gaussian smoothing, respectively. In the nonconvex regime, Ghadimi and Lan [58] analyzed Gaussian smoothing complexity, while Ji et al. [59] achieved improved query complexity through variance-reduced algorithms like ZO-SPIDER-Coord. Theoretical foundations for gradient estimators, including finite differences and uniform sphere smoothing, were further unified by Berahas et al. [60], with Ma and Huang [27] recently proposing minimum-variance two-point estimators. Some unbiased variants of ZOO [61, 62] have also been analyzed. Research has also extended to non-smooth settings: while Davis et al. [63], Zhang et al. [64], and Lin et al. [65] derived complexity bounds for Lipschitz functions (noting a $\sqrt{d}$ dependency), Rando et al. [66] and Kornowski and Shamir [67] demonstrated that, utilizing specific techniques, the non-smooth case is not inherently more challenging than the smooth case.

A.2 Connections to Existing Results

In this section, we discuss how our work is connected to other existing literature in fine-tuning quantized LLMs.

QZO [18] QZO also focuses on perturbing scales to avoid discrete weight issues and shows strong empirical performance compared to MeZO (as shown in Table 2). Consequently, QZO is the method most closely related to our approach; therefore, we select it as our primary baseline in Table 2. The primary distinction between HoZO and QZO lies in their zeroth-order gradient estimation strategies. QZO estimates gradients in Euclidean space, relying on *(i)* Directional Derivative Clipping (DDC) to bound the projected finite difference $\frac{f(x+\mu v)-f(x-\mu v)}{2\mu}$, and *(ii)* hard clipping on gradient updates to enforce the positivity of scale parameters. In contrast, HoZO unifies these constraints naturally through the Riemannian structure of the positive orthant $\mathbb{R}^d_{>0}$.

QuZO [16] QuZO adapts the stochastic rounding technique [68] to update the integer parameter $\mathbf{q}$, rather than the positive scale parameter x. Unlike QZO and our proposed HoZO, which target scale parameters, QuZO focuses on the quantized weights. This difference suggests that QuZO and HoZO are complementary approaches, as they optimize distinct sets of parameters.

ZOQO [19] ZOQO estimates the gradient of the integer weight $\mathbf{q}$ by perturbing the de-quantized value $x \cdot \mathbf{q}$. To address the challenges associated with differentiating discrete parameters, the method relies on the gradient's sign rather than its magnitude. Consequently, HoZO is complementary to ZOQO, as the two methods optimize distinct sets of parameters.

ZO2 [29, 30] ZO2 provides another alternative approach to scale ZOO methods to fine-tune an extremely large language model; *e.g.* ZO2 enables the fine-tuning of OPT-175B model in a single 18 GB GPU. Its main methodology relies on the CPU off-loading technique; as a result, it is orthogonal to our proposed HoZO method.

B Experimental Setups

B.1 Hardware and System Environment

Our main experiments were conducted on a high-performance computing (HPC) cluster running Linux 4.18.0, consisting of multiple nodes equipped with multiple NVIDIA RTX A5000, RTX A6000, and RTX 6000 Ada Generation GPUs. The implementation was developed in Python 3.12.7. Due to resource allocation policies, simultaneous access to multiple GPUs and control over the specific GPU type were restricted. Consequently, most experiments were performed using a single GPU, and the exact GPU model could vary across runs. We observed no significant impact of these hardware

variations on the final performance. The full source code and a complete list of dependencies (see "requirements.txt") are provided in the supplementary materials.

B.2 More Detailed Experimental Setups and Hyperparameters in Section 5.1

In Section 5.1, we trained all methods across all tasks and models for 20,000 iterations using a stochastic batch size of 16, logging validation loss every 4,000 steps. The random seed is fixed for all experiments. Throughout the training process, we checkpointed the model achieving the lowest validation loss.

Upon completion, we evaluated this best-performing model using test and validation sets containing 1,000 and 500 examples, respectively. We also note that CB [47] and COPA [49] have less than 1,000 training samples; we only use 100 examples as the validation set.

For HoZO, the linear threshold was fixed at $\delta = 10^{-6}$. Although we evaluated smaller values (e.g., $\delta = 10^{-8}$) in preliminary experiments, we observed no significant impact on performance. For QZO [18], we set the directional derivative clipping parameter to 100, adopting the optimal value suggested by the authors. We selected the remaining hyperparameters, such as the smoothing parameter and learning rate, via a grid search based on validation accuracy. The search ranges are detailed in Table 3. Typically, a larger learning rate η will result in better performance, while in some cases, a too large learning rate will result in instable training and halving this learning rate will usually fix this issue. We also observe that the value of smoothing factor μ does not significantly affect the performance, which has also been noticed by Malladi et al. [2] (in their Appendix E.3).

Table 3: The hyperparameter grid used in our experiments. For simplicity, we only use a constant learning rate schedule.

Method	Hyperparameters	Values
All Methods	Smoothing factor μ	$\{10^{-3}, 10^{-4}\}$
	Learning rate η	$\{2 \times 10^{-7}, 10^{-7}, 5 \times 10^{-8}\}$
QZO	Directional Derivative Clipping	$\{100\}$
HoZO	Linear threshold δ	$\{10^{-6}\}$

B.3 The 2-Bit Extreme Quantization

In the 2-bit quantization experiment, we follow the same setting as Section B.2, except for replacing the model to the Llama-3-8b model [43] and increasing the maximum steps from 20,000 to 40,000. Extending the visualization used in Figure 4, Section C.2 lists the detailed numerical results and visualizes additional losses.

B.4 Memory Profiling

For memory profiling, we utilize two open-source frameworks: QZO [18] and ZO-Bench [5]. To ensure a direct comparison, we disabled advanced memory-saving techniques, such as gradient checkpointing. We fixed the batch size at 16 across all optimization algorithms.

For the multi-GPU setting, we employed the Fully Sharded Data Parallel (FSDP) implementation provided by PyTorch [51]. We monitored usage via the "GPU Memory Allocated (%)" metric in wandb, recording the peak memory usage observed during the training or evaluation phases.

B.5 Loss vs Wall-Clock Time

To evaluate wall-clock efficiency, we conducted full fine-tuning sessions of the Llama-2-13B model [39] on the BoolQ dataset [48]. All methods were executed on a single NVIDIA RTX A6000 GPU using identical hyperparameters. We recorded the duration of each training step and reported the average time cost per iteration to estimate the wall-clock time.

C More Experimental Results

C.1 Memory Profiling

Table 4 lists the detailed numerical results for the memory profiling shown in Figure 5. The experimental setup for these measurements is described in Section B.4.

Table 4: GPU memory consumption for Llama-2 model family [39] with different sizes (7B, 13B, and 70B) and methods on the SST-2 dataset [44].
† : It indicates that the result is projected without testing.

Method	**Model Precision**	**Model Name**		
		Llama-2-7B	Llama-2-13B	Llama-2-70B
Zero-Shot	16 bits	14.6	27.3	147
Zero-Shot-Q	4 bits	4.91	8.59	34.4
Fine-Tuning	16 bits	129	316†	-
MeZO	16 bits	15.5	28.7	155
HoZO (4-bit)	4-bits	5.50	9.21	34.9
HoZO (2-bit)	2-bits	3.91	6.24	24.1

C.2 The 2-Bit Extreme Quantization

Table 5 lists the detailed numerical results for the time cost shown in Figure 4's bar plots. We also visualize the training and validation loss over other tasks (RTE, CB, Copa, and SQuAD) in Figure 8. The experimental setup for these measurements is described in Section B.3. As we have discussed in Section 5.2, the performance of HoZO (2-bit) typically significantly improves Zero-Shot (2-bit), and the training and validation loss can decrease as expected; however, it doesn't exceed the full-precision performance of Zero-Shot (FP16) due to the limited representation ability of extreme quantization.

Table 5: Performance of Llama-3-8b [43] in different tasks.

Model	**Task**	**Method**		
		Zero-Shot (FP16)	Zero-Shot (2-bit)	**HoZO (2-bit)**
Llama-3-8b	SST-2	58.7	50.8	53.9
	RTE	48.0	45.8	52.0
	CB	57.0	41.0	57.0
	BoolQ	61.0	41.5	53.9
	Copa	88.0	53.0	57.0
	SQuAD	64.7	1.8	5.7

C.3 Loss vs Wall-Clock Time

Table 6 lists the detailed numerical results for the time cost shown in Figure 6. The experimental setup for these measurements is described in Section B.5. We report wall-clock time costs only for experiments conducted on NVIDIA RTX A6000 GPU platform to ensure fair comparison. Results involving heterogeneous hardware configurations are excluded.

D Theoretical Results

In this section, we present the proofs for our main theoretical results.

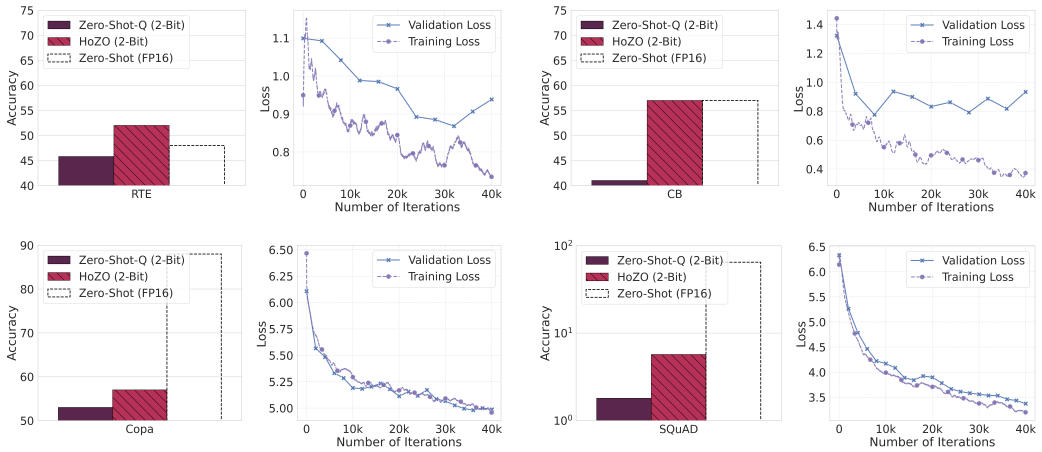


Figure 8: Experimental results for Llama-3-8b [43] on RTE (top left), CB (top right), Copa (bottom left), and SQuAD (bottom right). The detailed numbers of the bar plots are presented in Table 5.

Table 6: Comparison of averaged step time cost per iteration (in seconds).

Model	**Task**	**Method**		
		MeZO (FP16)	QZO (4-bit)	**HoZO (4-bit)**
Llama-2-13B	BoolQ	1.54	1.45	1.62
Llama-2-70B	SST-2	-	1.26	1.26
Llama-2-70B	Copa	-	0.61	0.63

D.1 Proof of Proposition 4.1

We recall that the Riemannian metric defined in Equation (5) is given as

$$g_x(u, v) = u^\top G(x) v, \tag{9}$$

where

$$G(x) = \operatorname{diag}\left(\max(1, \frac{\delta}{x_1})^2, \ldots, \max(1, \frac{\delta}{x_d})^2\right).$$

Since $G(x)$ is diagonal, the metric decouples across coordinates. Consequently, the geodesics (and thus the exponential map) can be computed element-wise.

We restate Proposition 4.1 with its full version below:

Proposition D.1. *Let $\mathcal{M} = \mathbb{R}^d_{>0}$ be the Riemannian manifold equipped with the metric g defined in Equation (5). Then the Riemannian manifold $(\mathcal{M}, g)$ is geodesically complete. Moreover, let x_i be the i-th coordinate of $x \in \mathcal{M}$ and v_i be the i-th coordinate of $v \in T_x\mathcal{M}$. Then the exponential map associated with g is given as:*

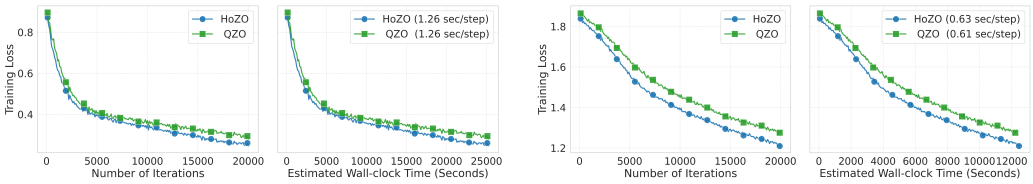
(a) If $x_i \geq \delta$:

$$[\exp_x(v)]_i = \begin{cases} x_i + v_i & \text{if } x_i + v_i \geq \delta, \\ \delta \exp\left(\frac{x_i + v_i - \delta}{\delta}\right) & \text{if } x_i + v_i < \delta. \end{cases} \tag{10}$$

(b) If $0 < x_i < \delta$:

$$[\exp_x(v)]_i = \begin{cases} x_i \exp\left(\frac{v_i}{x_i}\right) & \text{if } x_i \exp(v_i/x_i) \leq \delta, \\ \delta + \delta\frac{v_i}{x_i} - \delta \ln\left(\frac{\delta}{x_i}\right) & \text{if } x_i \exp(v_i/x_i) > \delta. \end{cases} \tag{11}$$

Proof. To prove the completeness, it suffices to evaluate the exponential function and verify that its domain is the whole tangent space $T_x\mathcal{M}$ for all $x \in \mathcal{M}$. Since the metric tensor $G(x)$ is diagonal,



(a) SST-2 (b) Copa

Figure 9: Comparison of the training losses of fine-tuning the Llama-2-70b model over SST-2 and Copa datasets.

the geodesic equations for each coordinate i are independent. Let $\gamma(t)$ be a geodesic curve such that $\gamma(0) = x$ and $\dot{\gamma}(0) = v$. We analyze the i-th component $\gamma_i(t)$ based on the value of $G_{ii}(x)$. We distinguish two regimes:

(a) **Case 1:** $x_i \geq \delta$**.**

In this case, $G_{ii}(x) = 1$. The geodesic equation corresponds to zero acceleration, $\ddot{\gamma}_i(t) = 0$. The solution is linear: $\gamma_i(t) = \gamma_i(0) + t\dot{\gamma}_i(0)$ (for $x_i \geq \delta$). We now derive the explicit formulas for $\gamma_i(1) = [\exp_x(v)]_i$ by considering whether the geodesic crosses the threshold δ within the time interval $t \in [0, 1]$.

- If $x_i + v_i \geq \delta$, the path never enters the hyperbolic region (*i.e.* $x_i < \delta$). The standard Euclidean update applies:
$$[\exp_x(v)]_i = x_i + v_i.$$
We obtain the first formula in Equation (10).
- If $x_i + v_i < \delta$, the path crosses δ at time $t_c = (\delta - x_i)/v_i \in [0, 1]$. At the crossing, continuity of velocity implies the particle enters the hyperbolic region ($x_i < \delta$) with velocity v_i[3]. For $t > t_c$, the path follows the hyperbolic law starting from δ:
$$\gamma_i(t) = \delta \exp\left(\frac{v_i}{\delta}(t - t_c)\right).$$
We evaluate it at $t = 1$:
$$\gamma_i(1) = \delta \exp\left(\frac{v_i}{\delta}\left(1 - \frac{\delta - x_i}{v_i}\right)\right) = \delta \exp\left(\frac{v_i - \delta + x_i}{\delta}\right).$$
It results in the second formula in Equation (10).

(b) **Case 2:** $x_i < \delta$**.**

In this case, $G_{ii}(x) = (\delta/x_i)^2$. The geodesic equation is derived from the Lagrangian $L = (\frac{\delta}{x})^2\dot{x}^2$. The Euler-Lagrange equation yields $\ddot{\gamma}_i - \frac{1}{\gamma_i}\dot{\gamma}_i^2 = 0$, which has the solution $\gamma_i(t) = \gamma_i(0)\exp(\frac{\dot{\gamma}_i(0)}{\gamma_i(0)}t)$.

- If $x_i \exp(v_i/x_i) \leq \delta$, the path stays within the hyperbolic region. The solution is
$$[\exp_x(v)]_i = x_i \exp(v_i/x_i).$$
It yields to the first formula in Equation (11).
- If $x_i \exp(v_i/x_i) > \delta$, the path crosses δ at time t_c satisfying $x_i \exp(\frac{v_i}{x_i}t_c) = \delta$, which implies $t_c = \frac{x_i}{v_i}\ln(\frac{\delta}{x_i})$. The velocity at the crossing point t_c (approaching from below) is $\dot{\gamma}_i(t_c) = \gamma_i(t_c)\frac{v_i}{x_i} = \delta\frac{v_i}{x_i}$. Upon entering the Euclidean region, the velocity remains constant. For $t > t_c$:
$$\gamma_i(t) = \delta + \dot{\gamma}_i(t_c)(t - t_c) = \delta + \frac{\delta v_i}{x_i}(t - t_c).$$

[3]Although the metric tensor $G(x)$ is not differentiable here, it remains continuous. By the conservation of the geodesic speed $\|\dot{\gamma}\|_g$, we have: $\lim_{t\to t_c^-} G_{ii}(\gamma(t))\dot{\gamma}_i(t)^2 = \lim_{t\to t_c^+} G_{ii}(\gamma(t))\dot{\gamma}_i(t)^2$. Since $G_{ii}(\delta) = 1$ from both sides due to the continuity, it implies $\lim_{t\to t_c^-}\dot{\gamma}_i(t) = \lim_{t\to t_c^+}\dot{\gamma}_i(t)$, verifying that the velocity is continuous at this point.

We evaluate it at $t = 1$:

$$\gamma_i(1) = \delta + \frac{\delta v_i}{x_i}\left(1 - \frac{x_i}{v_i}\ln\frac{\delta}{x_i}\right) = \delta + \frac{\delta v_i}{x_i} - \delta\ln\left(\frac{\delta}{x_i}\right).$$

It yields to the second formula in Equation (11).

Combining these cases, we obtain the explicit expression for the exponential map. It is straightforward to verify that in both regions, the components of $\exp_x(v)$ are strictly positive. This implies that $\exp_x(v) \in \mathcal{M}$ for all $x \in \mathcal{M}$ and all $v \in T_x\mathcal{M}$. Since the exponential map is defined on the entire tangent space, the Riemannian manifold $(\mathcal{M}, g)$ is geodesically complete. □

D.2 Proof of Proposition 4.3

Recall that for any point $x \in \mathcal{M} := \mathbb{R}^d_{>0}$ and tangent vectors $u, v \in T_x\mathcal{M}$, the metric g is defined as:

$$g_x(u, v) = u^\top G(x) v,$$

where $G(x) = \operatorname{diag}\left(\max\left(1, \frac{\delta}{x_1}\right)^2, \ldots, \max\left(1, \frac{\delta}{x_d}\right)^2\right)$.We consider the transformation $\Phi : \mathcal{M} \to \mathbb{R}^d$ defined element-wise. For each coordinate $i \in \{1, \ldots, d\}$, the mapping is given by:

$$[\Phi(x)]_i = \begin{cases} x_i, & \text{if } x_i > \delta; \\ \delta\left(\ln\left(\frac{x_i}{\delta}\right) + 1\right), & \text{if } x_i \le \delta. \end{cases} \tag{12}$$

Observe that the map $\Phi : (\mathcal{M}, g) \to (\mathbb{R}^d, g_E)$ is a global isometry (as a C^0 Riemannian manifold); that is, it defines a C^1-diffeomorphism mapping the positive orthant $\mathbb{R}^d{>}\,0$ to the entire Euclidean space $\mathbb{R}^d$, preserving the inner product structure (shown in Lemma D.4). Consequently, we can employ the re-parameterization trick to convert the optimization problem over the Riemannian manifold $(\mathcal{M}, g)$ into an equivalent problem over Euclidean space.

Equivalence of Objective Functions Specifically, under this re-parameterization, the Riemannian SGD update can be explicitly mapped to the Euclidean SGD update. We consider the following Riemannian optimization problem:

$$\min_{x \in \mathcal{M}} \mathbb{E}_{\xi \sim \Xi}[f(x; \xi)]. \tag{13}$$

Let $h := f \circ \Phi^{-1} : \mathbb{R}^d \to \mathbb{R}$ be the re-parameterized objective function. By substituting $y = \Phi(x)$, the problem becomes:

$$\min_{y \in \mathbb{R}^d} \mathbb{E}_{\xi \sim \Xi}[h(y; \xi)]. \tag{14}$$

Thus, solving Equation (13) is equivalent to solving Equation (14). As this result is simply the change of variable, we omit the formal proof for this statement.

Equivalence of Gradient Estimation We also observe that the zeroth-order gradient estimation in solving Equation (14) is equivalent to the Riemannian zeroth-order gradient estimation in solving Equation (13). To formally state this equivalence, we show that the ZOO gradient estimation can be exactly transformed into the Euclidean form. Recall the zeroth-order gradient estimator on the manifold (Equation (4)):

$$\widehat{\nabla} f(x; \xi) = \frac{f(\exp_x(\mu v); \xi) - f(\exp_x(-\mu v); \xi)}{2\mu} v,$$

where $v \sim \sqrt{d}\,\mathcal{U}(\mathbb{S}_g^{d-1})$ is a scaled random perturbation vector drawn from the uniform distribution over the g-unit sphere $\mathbb{S}_g^{d-1}$, and $\mu > 0$ is a small smoothing parameter. Its corresponding Euclidean form is given by:

$$\widehat{\nabla} h(y; \xi) = \frac{h(y + \mu u; \xi) - h(y - \mu u; \xi)}{2\mu} u, \tag{15}$$

where $y = \Phi(x)$ is the re-parameterized variable and $u := (d\Phi)_x v$ corresponds to the pushforward of the perturbation direction v. Then we have the following equivalence:

$$\widehat{\nabla} h(y;\xi) = (d\Phi)_x \left(\widehat{\nabla} f(x;\xi) \right). \tag{16}$$

The formal proof is presented in Lemma D.5. Consequently, Equation (15) is exactly the standard zeroth-order gradient estimator applied to the transformed function $h(y)$ in Euclidean space up to a linear transformation determined by the isometric transformation Φ.

Equivalence of Optimization Trajectories Lastly, we show that solving Equation (13) using Equation (4) as the gradient estimation is equivalent to solving Equation (14) using Equation (15); that is, their optimization trajectories are exactly same by employing the isometric mapping Φ. Due to the isometric property of Φ, the mapping preserves the geometry of the tangent space. Specifically, the relationship $u = (d\Phi)_x v$ implies that if v is drawn from a scaled uniform distribution on the Riemannian unit sphere $\mathbb{S}_g^{d-1}$, then u follows the same scaled uniform distribution on the standard Euclidean sphere $\mathbb{S}^{d-1}$ (i.e., $u \sim \sqrt{d}\,\mathcal{U}(\mathbb{S}^{d-1})$).

The Riemannian zeroth-order SGD update step using the exponential map is given by:

$$\begin{aligned}
x_{k+1} &= \exp_{x_k}\left(-\eta \widehat{\nabla} f(x_k;\xi_k)\right) \\
\Phi(x_{k+1}) &\overset{(i)}{=} \Phi(\exp_{x_k}\left(-\eta \widehat{\nabla} f(x_k;\xi_k)\right)) \\
&\overset{(ii)}{=} \Phi(x_k) - \eta (d\Phi)_x \widehat{\nabla} f(x_k;\xi_k) \\
&\overset{(iii)}{=} \Phi(x_k) - \eta (d\Phi)_x (d\Phi)_x^{-1} \widehat{\nabla} h(y_k;\xi_k) \\
&= \Phi(x_k) - \eta \widehat{\nabla} h(y_k;\xi_k)
\end{aligned}$$

where (i) applies Φ on both sides, (ii) applies the property that $\Phi(\exp_x(v)) = \Phi(x) + (d\Phi)_x v$, and (iii) applies the equivalence Equation (16). Consequently, to derive the convergence properties of the HoZO algorithm, it suffices to analyze the equivalent optimization problem in the Euclidean space. This reduction constitutes the main proof strategy employed in Proposition 4.3.

D.2.1 Assumptions and Justifications

As the Riemannian metric g is not second-order continuously differentiable, the L-smoothness condition may not be directly applied to the function on the Riemannian manifold $(\mathbb{R}^d_{>0}, g)$. Therefore, we directly assume the L-smoothness on the reparameterized function $h := f \circ \Phi^{-1}$. We note that when the Riemannian metric is smooth, this definition is equivalent to the L-smoothness of the function $f : \mathcal{M} \to \mathbb{R}$.

Assumption D.2 (L-Smoothness)**.** Let $f(\cdot;\xi) : \mathcal{M} \to \mathbb{R}$ be the individual loss function. The reparameterized function $h = f(\,\cdot\,;\xi) \circ \Phi^{-1} : \mathbb{R}^d \to \mathbb{R}$ is L-smooth with respect to the Euclidean norm. That is, for all $u \in \mathbb{R}^d$, there exists $L \geq 0$ such that

$$u^\top \nabla^2 h(y) u \leq L \|u\|_{g_E}^2$$

holds almost surely with respect to $y \in \mathbb{R}^d$.

We note that here we explicitly assume the L-smoothness over the region where the Hessian is well-defined. For the specific hyperplane determined by the C^1-function Φ^{-1}, it has measure 0 with the Lebesgue measure, which is included in this almost-sure statement.

Assumption D.3 (Lower boundedness)**.** Let $f(\cdot;\xi) : \mathcal{M} \to \mathbb{R}$ be the individual loss function. The infimum $f_\xi^* := \inf_x f(x;\xi)$ exists almost surely with $\xi \sim \Xi$.

D.2.2 Supporting Lemmas

Lemma D.4. *The map $\Phi : (\mathcal{M}, g) \to (\mathbb{R}^d, g_E)$ is a C^1-diffeomorphism mapping the positive orthant $\mathbb{R}^d_{>0}$ to the entire Euclidean space $\mathbb{R}^d$, and it preserves the inner product structure:*

$$\langle u, v \rangle_{g_x} = \langle d\Phi(x)[u], d\Phi(x)[v] \rangle_{g_E}, \quad \forall x \in \mathcal{M},\ u, v \in T_x\mathcal{M},$$

where $d\Phi(x)$ denotes the differential of Φ at x.

Proof. Since the map Φ and the metric g are defined element-wise (diagonally), it suffices to prove the properties for the one-dimensional scalar function $\phi : (0, \infty) \to \mathbb{R}$ defined by:

$$\phi(t) = \begin{cases} t, & t > \delta \\ \delta(\ln(t/\delta) + 1), & 0 < t \leq \delta \end{cases}$$

and the corresponding metric component $g(t) = \max(1, \delta/t)^2$.

1. To prove it is a C^1-diffeomorphism, we need to prove:

   - **It is a bijection:** First, we check the range. For $t \in (\delta, \infty)$, $\phi(t) = t$, mapping (δ, ∞) to (δ, ∞). For $t \in (0, \delta]$, $\phi(t)$ is a strictly increasing logarithmic function. As $t \to 0^+$, $\phi(t) \to -\infty$; at $t = \delta$, $\phi(\delta) = \delta$. Thus, it maps $(0, \delta]$ bijectively to $(-\infty, \delta]$. Combining these, the image of Φ is $(-\infty, \delta] \cup (\delta, \infty) = \mathbb{R}$. Since $\phi'(t) > 0$ for all $t > 0$ (as shown below), the map is strictly monotonic and thus injective. Hence, Φ is a bijection from $\mathbb{R}^d_{>0}$ to $\mathbb{R}^d$.
   - **It is continuously differentiable (C^1):** Clearly, $\phi(t)$ is smooth strictly inside the intervals $(0, \delta)$ and (δ, ∞). We only need to verify continuity and differentiability at the junction $t = \delta$.
     - *Continuity of Φ:* $\lim_{t \to \delta^+} \phi(t) = \delta$ and $\lim_{t \to \delta^-} \phi(t) = \delta(\ln(1) + 1) = \delta$. The function is continuous.
     - *Continuity of Derivative:* For $t > \delta$, $\phi'(t) = 1$. Thus $\lim_{t \to \delta^+} \phi'(t) = 1$. For $t < \delta$, $\phi'(t) = \frac{d}{dt}[\delta \ln t - \delta \ln \delta + \delta] = \frac{\delta}{t}$. Thus $\lim_{t \to \delta^-} \phi'(t) = \frac{\delta}{\delta} = 1$.

     Since the left and right derivatives match, Φ is C^1 on the entire domain.
   - **Its inverse is continuously differentiable:** We can derive the inverse map $\Phi^{-1} : \mathbb{R}^d \to \mathbb{R}^d_{>0}$ explicitly. Let $y = \phi(t)$. If $y > \delta$, then $t = y$. If $y \leq \delta$, then $y/\delta - 1 = \ln(t/\delta) \implies t = \delta \exp(y/\delta - 1)$. Thus, the inverse function $\psi(y) = \phi^{-1}(y)$ is:

     $$\psi(y) = \begin{cases} y, & y > \delta \\ \delta e^{\frac{y}{\delta} - 1}, & y \leq \delta \end{cases}$$

     Checking the derivative $\psi'(y)$: For $y > \delta$, $\psi'(y) = 1$. For $y \leq \delta$, $\psi'(y) = \delta \cdot \frac{1}{\delta} e^{\frac{y}{\delta} - 1} = e^{\frac{y}{\delta} - 1}$. At the junction $y = \delta$, the left derivative is $e^{\delta/\delta - 1} = e^0 = 1$, which matches the right derivative. Thus Φ^{-1} is also C^1.

2. To prove it preserves the inner product structure, we show that Φ is an isometry. This is equivalent to showing that the pullback of the Euclidean metric g_E under Φ equals the metric g.

   Let $J(x) = d\Phi(x)$ be the Jacobian matrix of Φ at x. Since Φ operates element-wise, $J(x)$ is a diagonal matrix with entries $J_{ii} = \phi'(x_i)$. The condition for isometry is:

   $$\langle d\Phi(x)[u], d\Phi(x)[v] \rangle_{g_E} = (Ju)^\top (Jv) = u^\top (J^\top J) v = u^\top G(x) v.$$

   Therefore, we must satisfy $J(x)^2 = G(x)$, or equivalently for each component: $(\phi'(x_i))^2 = G_{ii}(x_i)$.

   We examine the two cases for an arbitrary coordinate x_i:

   - **Case 1:** $x_i > \delta$**.** The metric term is $G_{ii} = \max(1, \delta/x_i)^2$. Since $x_i > \delta$, $\delta/x_i < 1$, so $G_{ii} = 1^2 = 1$. The derivative is $\phi'(x_i) = 1$. Thus, $(\phi'(x_i))^2 = 1 = G_{ii}$.
   - **Case 2:** $x_i \leq \delta$**.** The metric term is $G_{ii} = \max(1, \delta/x_i)^2$. Since $x_i \leq \delta$, $\delta/x_i \geq 1$, so $G_{ii} = (\delta/x_i)^2$. The derivative is $\phi'(x_i) = \frac{\delta}{x_i}$. Thus, $(\phi'(x_i))^2 = (\frac{\delta}{x_i})^2 = G_{ii}$.

   Since the squared Jacobian eigenvalues match the metric eigenvalues everywhere, the inner product is preserved:

   $$\langle u, v \rangle_{g_x} = \langle d\Phi(x)[u], d\Phi(x)[v] \rangle_{g_E}.$$

Then the proof is completed. □

Lemma D.5 (Equivalence of gradient estimation)**.** *Let $x \in \mathcal{M}$ and $y = \Phi(x) \in \mathbb{R}^d$. Let $v \in T_x\mathcal{M}$ be a perturbation vector and $u = (d\Phi)_x v \in \mathbb{R}^d$ be its pushforward. Let $\widehat{\nabla} f(x;\xi)$ and $\widehat{\nabla} h(y;\xi)$ be the gradient estimators defined in Equation (4) and Equation (15), respectively. Then, the Euclidean estimator is the pushforward of the Riemannian estimator:*

$$\widehat{\nabla} h(y;\xi) = (d\Phi)_x \left(\widehat{\nabla} f(x;\xi)\right). \tag{17}$$

Proof. By the definition of the function h, we have $f(x) = h(\Phi(x))$. Since Φ is an isometry mapping the Riemannian manifold $(\mathcal{M}, g)$ to the Euclidean space $(\mathbb{R}^d, g_E)$ (Lemma D.4), it maps geodesics in $\mathcal{M}$ to straight lines in $\mathbb{R}^d$. Specifically, for any $x \in \mathcal{M}$ and $v \in T_x\mathcal{M}$, the exponential map satisfies:

$$\Phi(\exp_x(\mu v)) = \Phi(x) + \mu(d\Phi)_x v = y + \mu u.$$

Substituting this into the Riemannian gradient estimator $\widehat{\nabla} f(x;\xi)$:

$$\begin{aligned}
\widehat{\nabla} f(x;\xi) &= \frac{f(\exp_x(\mu v);\xi) - f(\exp_x(-\mu v);\xi)}{2\mu} v \\
&= \frac{h(\Phi(\exp_x(\mu v));\xi) - h(\Phi(\exp_x(-\mu v));\xi)}{2\mu} v \\
&= \frac{h(y+\mu u;\xi) - h(y-\mu u;\xi)}{2\mu} v.
\end{aligned}$$

Now, applying the pushforward $(d\Phi)_x$ to both sides, and noting that $(d\Phi)_x$ is linear and $(d\Phi)_x v = u$:

$$\begin{aligned}
(d\Phi)_x \left(\widehat{\nabla} f(x;\xi)\right) &= \frac{h(y+\mu u;\xi) - h(y-\mu u;\xi)}{2\mu} (d\Phi)_x v \\
&= \frac{h(y+\mu u;\xi) - h(y-\mu u;\xi)}{2\mu} u \\
&= \widehat{\nabla} h(y;\xi).
\end{aligned}$$

This completes the proof. □

D.2.3 Main Proof of Proposition 4.3

We re-state the full result as follows:

Proposition D.6. *Let $\mathcal{M} = \mathbb{R}^d_{>0}$ be the Riemannian manifold with the Riemannian metric g defined in Equation (5) and $f(\cdot;\xi) : \mathcal{M} \to \mathbb{R}$ be the individual loss function associated with the stochastic data sample $\xi \sim \Xi$. Suppose that Assumptions D.2 and D.3 hold. Define the zeroth-order estimator as in Equation (4) with the exponential map defined in Equation (6). Let $\{x_t\}_{t=1}^T$ be the SGD dynamic generated by the update rule Equation (7) with requiring*

$$\eta \le \min\left\{\frac{1}{2L}, \frac{1}{L\sqrt{10Td}}\right\},$$

then

$$\min_{1\le t\le T} \mathbb{E}\|\nabla f(x_t)\|_g^2 \le \frac{1}{\eta T}\left(f(x_1) - f^*\right) + 2\eta\left[4L^2(f^* - \mathbb{E}_\xi f_\xi^*)d + \mu^2 L^3 d^2\right].$$

where $f^* := \min_x f(x)$, and $f_\xi^* := \min_x f(x;\xi)$.

Proof. As established in the beginning of Section D.2, the trajectory $\{x_t\}$ on $\mathcal{M}$ generated by the Riemannian Zeroth-Order SGD (Equation (7)) is equivalent to the trajectory $\{y_t\}$ in $\mathbb{R}^d$ via $y_t = \Phi(x_t)$, where $\{y_t\}$ is generated by the standard Euclidean Zeroth-Order SGD on the function $h = f \circ \Phi^{-1}$.

As the lower boundedness of f (Assumption D.3) implies the lower boundedness of h, and the L-smoothness of h is made in Assumption D.2, we can directly apply Theorem 3.1 from Ma and Huang [27] here. For the learning rate

$$\eta \le \min\left\{\frac{1}{2L}, \frac{1}{L\sqrt{10Td}}\right\},$$

the Euclidean convergence is given by:

$$\min_{1\le t\le T}\mathbb{E}\|\nabla h(y_t)\|_2^2 \le \frac{1}{\eta T}\left(h(y_1)-h^*\right)+2\eta\left[4L^2(h^*-\mathbb{E}_\xi h_\xi^*)d+\mu^2L^3d^2\right],$$
$$\overset{(i)}{=}\frac{1}{\eta T}\left(f(x_1)-f^*\right)+2\eta\left[4L^2(f^*-\mathbb{E}_\xi f_\xi^*)d+\mu^2L^3d^2\right],$$

where (i) applies

$$h^* := \min_y h(y) = \min_y f(\Phi^{-1}(y)) = \min_x f(x) =: f^*,$$

and

$$h_\xi^* := \min_y h(y;\xi) = \min_y f(\Phi^{-1}(y);\xi) = \min_x f(x;\xi) =: f_\xi^*.$$

Finally, we map this result back to the manifold. By the isometric property of Φ, the norms of the gradients are preserved:

$$\|\nabla f(x_t)\|_g = \|(d\Phi)_{x_t}^{-1}\nabla h(y_t)\|_g = \|\nabla h(y_t)\|_{g_E}.$$

Substituting this relation into the Euclidean bound yields:

$$\min_{1\le t\le T}\mathbb{E}\|\nabla f(x_t)\|_g^2 \le \frac{1}{\eta T}\left(f(x_1)-f^*\right)+2\eta\left[4L^2(f^*-\mathbb{E}_\xi f_\xi^*)d+\mu^2L^3d^2\right].$$

Then the complexity of achieving $\min_{1\le t\le T}\mathbb{E}\|\nabla f(x_t)\|_g \le \epsilon$ is $\mathcal{O}(d/\epsilon^4)$. □

D.3 Proof of Proposition 4.4

We recall that the reversed map defined in Equation (8) is given as

$$[\mathrm{rev}_y(v)]_i := \begin{cases} y_i - v_i, & \text{if } y_i > \delta; \\ \delta\left(1+\ln\left(\frac{y_i}{\delta}\right)\right) - v_i, & \text{if } y_i \le \delta. \end{cases} \tag{18}$$

We recap Proposition 4.4 below:

Proposition D.7. *Let $\mathcal{M}=\mathbb{R}^d_{>0}$ be the Riemannian manifold with the metric g defined in Equation (5). Suppose that x satisfies that $x_i > \delta$ and $v \in T_x\mathcal{M}$ is a fixed vector. If $y = \exp_x(v)$ and $x' = \mathrm{rev}_y(v)$, then $x' = x$.*

Proof. We provide the proof by analyzing the element-wise operations for the i-th coordinate. Let x_i be the i-th component of the parameter, v_i be the perturbation, and $y_i = [\exp_x(v)]_i$. Recall the definition of the forward exponential map from Equation (6) on the region $\{x \in \mathcal{M} : x_i > \delta\ \forall i\}$:

$$y_i = \begin{cases} x_i + v_i, & \text{if } x_i + v_i > \delta; \\ \delta\exp\left(\frac{x_i+v_i}{\delta} - 1\right), & \text{if } x_i + v_i \le \delta. \end{cases}$$

We analyze the two cases based on the value of the intermediate term $x_i + v_i$.

- If $x_i + v_i > \delta$, In this regime, the forward map gives $y_i = x_i + v_i$. Since $x_i + v_i > \delta$, it follows that $y_i > \delta$. Therefore, when applying the reversed map $\mathrm{rev}_y(v)$, the condition $y_i > \delta$ is satisfied, leading to the first branch of Equation (8):

  $$\begin{aligned}[\mathrm{rev}_y(v)]_i &= y_i - v_i \\ &= (x_i + v_i) - v_i \\ &= x_i.\end{aligned}$$

  Thus, exact recovery is achieved in the linear regime.

- If $x_i + v_i \le \delta$, In this regime, the forward map is given by:

  $$y_i = \delta\exp\left(\frac{x_i+v_i}{\delta} - 1\right).$$

Since the exponential function maps to positive values and the argument is non-positive (because $(x_i + v_i)/\delta \leq 1 \implies (x_i + v_i)/\delta - 1 \leq 0$), we observe that $0 < y_i \leq \delta$ (specifically, $y_i \leq \delta e^0 = \delta$).

Moreover, as $y_i \leq \delta$, the reversed map triggers the second branch of Equation (8). We substitute y_i into the formula:

$$[\mathrm{rev}_y(v)]_i = \delta \left(1 + \ln\left(\frac{y_i}{\delta}\right)\right) - v_i.$$

Substituting the expression for y_i:

$$\begin{aligned}[\mathrm{rev}_y(v)]_i &= \delta \left(1 + \ln\left(\frac{\delta \exp\left(\frac{x_i+v_i}{\delta} - 1\right)}{\delta}\right)\right) - v_i \\ &= \delta \left(1 + \ln\left(\exp\left(\frac{x_i + v_i}{\delta} - 1\right)\right)\right) - v_i.\end{aligned}$$

Using the identity $\ln(\exp(z)) = z$, we simplify the term inside the parenthesis:

$$\begin{aligned}[\mathrm{rev}_y(v)]_i &= \delta \left(1 + \left(\frac{x_i + v_i}{\delta} - 1\right)\right) - v_i \\ &= \delta \left(\frac{x_i + v_i}{\delta}\right) - v_i \\ &= (x_i + v_i) - v_i \\ &= x_i.\end{aligned}$$

In both cases, $[\mathrm{rev}_y(v)]_i = x_i$, which confirms that $\mathrm{rev}_y(v)$ is the exact inverse of $\exp_x(v)$ on the region $\{x \in \mathcal{M} : x_i > \delta \ \forall i\}$. It completes the proof. □

E Other Experiments

In this section, we additionally include more experiments to further support our statements.

E.1 Results across Random Seeds

We include 3 additional seeds to validate the stability of HoZO for Llama-2-13B fine-tuning, which yields a six-task average of 82.9 ± 0.28 versus fixed-seed QZO at 82.5.

Table 7: Results across Random Seeds

Method	SST-2	RTE	CB	BoolQ	Copa	SQuAD	AVG
MeZO	90.7	58.5	77.0	81.6	92.0	87.5	81.2
QZO	91.9	62.8	77.0	82.4	92.0	89.2	82.5
HoZO (Avg)	92.3	68.2	74.8	81.5	92.3	88.6	82.9 ± 0.28
HoZO (Seed 1)	92.3	69.8	75.0	81.0	93.0	88.2	83.2
HoZO (Seed 2)	93.2	67.2	74.0	81.4	92.0	87.7	82.6
HoZO (Seed 3)	91.1	67.1	75.0	82.0	92.0	89.2	82.7
HoZO (Seed 42)	92.4	68.6	75.0	81.6	92.0	89.3	83.2

Moreover, we note that the training and evaluation are taken over a sub-sampled dataset instead of the whole dataset, but we hope highlight that using sub-sampled training and evaluation sets is a common practice in the zeroth-order fine-tuning literature including [2, 5, 18].

E.2 Comparison with 70B Zero-Shot Performance

We have included the following additional comparison with Zero-Shot for Llama-2-70B models.

Table 8: Comparison with 70B Zero-Shot Performance

Method	Model Precision	SST-2	RTE	CB	BoolQ	Copa	SQuAD	Average	
Zero-Shot-Q	4 bits	56.4	60.6	47.0	74.7	92.0	71.4	67.0	–
Zero-Shot	16 bits	62.1	61.7	53.0	71.9	90.0	76.2	69.2	↑ 2.2
QZO	4 bits	90.6	80.8	82.0	83.8	93.0	90.4	86.8	↑ 19.8
HoZO	4 bits	91.5	79.1	83.0	83.8	95.0	91.2	87.3	↑ 20.3

E.3 Evaluation on Additional Models

We also included matched INT4 experiments on Llama-3-8B and the non-Llama OPT-6.7B.

Table 9: Evaluation on Additional Models

Method	Model	SST-2	RTE	CB	Avg
QZO	Llama-3-8B (INT4)	91.6	73.9	69.0	78.2
HoZO	Llama-3-8B (INT4)	**93.0**	**75.1**	**70.0**	**79.4**
QZO	OPT-6.7B (INT4)	86.7	61.0	**67.0**	71.6
HoZO	OPT-6.7B (INT4)	**87.7**	**63.1**	66.0	**72.3**

E.4 Comparisons with Additional Baselines

We have included the following comparison on Llama-2-7B. We emphasize that these methods optimize different parameterizations: LoRA and QLoRA update adapters, QuZO and ZOQO update discrete quantized weights, while QZO and HoZO update quantization scales.

Table 10: Comparisons with Additional Baselines

Method	TYPE	SST-2	RTE	CB	BoolQ	Copa	SQuAD	Avg
LoRA	First-order	94.6	62.1	71.0	71.8	84.0	82.5	77.7
QLoRA	First-order	95.1	62.3	68.0	72.0	85.0	83.5	77.6
QuZO (4-bit)	Zeroth-order	92.5	60.3	61.0	65.5	83.0	82.8	74.2
ZOQO (4-bit)	Zeroth-order	88.7	59.2	58.0	64.7	84.0	82.5	72.9
QZO (4-bit)	Zeroth-order	90.0	59.2	69.6	68.2	87.0	85.5	76.6
HoZO (4-bit)	Zeroth-order	91.9	62.9	74.0	72.0	89.0	87.4	79.5

F Limitations

HoZO incurs additional per-step computation due to its nonlinear perturbation operator and reversed-perturbation procedure. For Llama-2-13B on BoolQ, the average iteration time is 1.62 s, compared with 1.45 s for QZO and 1.54 s for MeZO. Although faster convergence compensates for this overhead in the reported experiment, the resulting wall-clock advantage remains to be evaluated more broadly across tasks, model sizes, and hardware configurations. Furthermore, we have not systematically explored integrating HoZO with adaptive optimizers, such as Adam for PEFT modules, or combining it with other model compression techniques. The effectiveness and computational trade-offs of these combinations remain open questions for future work.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We have included the main claims in the abstract and introduction accurately reflect the paper's contributions and scope.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We have discussed the limitations of this work.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: We have fully stated the assumptions and proof in the appendix.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We have fully disclosed all details needed to reproduce the experimental results.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We will release codes and we have attached all codes to the supplementary materials.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: We have specified all the training and test details.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: Fine-tune a 70B model is extremely expensive. We have set a standard seeded training pipeline, which allows us to fairly compare different methods.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We have specified the compute resources in the appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: We will follow the code of ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification: There is no societal impact of the work performed.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: Not applicable to our work.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [N/A]

    Justification: Not applicable to our work.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: Not applicable to our work.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: Not applicable to our work.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: Not applicable to our work.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: Not applicable to our work.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.