
STRUCTZO: STRUCTURED ZEROth-ORDER OPTIMIZATION FOR MEMORY-EFFICIENT LLM FINE-TUNING

Xuanlin Yang
Peking University
2401112159@stu.pku.edu.cn

Jia Li
Beijing Normal University
jiali.gm@gmail.com

Zhouchen Lin
Peking University
ZLIN@pku.edu.cn

ABSTRACT

Fine-tuning large language models at scale is increasingly limited by the memory cost of back-propagation. Zeroth-Order (ZO) optimizers reduce this burden by relying only on forward function evaluations, but they still exhibit a substantial performance gap to first-order (FO) optimizers such as Adam. In this work, we argue that this gap does not arise solely from the absence of explicit gradients: FO fine-tuning also benefits from informative low-rank optimization geometry, large-scale signal aggregation, and well-conditioned update directions. Motivated by this perspective, we introduce **StructZO**, a **structured Zeroth-Order** optimization framework for memory-efficient LLM fine-tuning. StructZO performs large-query ZO estimation within an activation-induced informative subspace and applies spectral control to the resulting reduced-space update before mapping it back to the original parameters. This unified update procedure enables ZO optimization to aggregate useful directional signals in a compact, geometry-aware space while avoiding the activation and optimizer-state memory costs of backpropagation. We provide a theoretical analysis showing that, when the induced subspace captures concentrated optimization-relevant directions, StructZO reduces the effective ZO estimation dimension from mn to mr , with the remaining projection error governed by the spectral tail of the activation Gram matrix. Empirically, across OPT-series and Llama3.1-8B on diverse downstream fine-tuning tasks, StructZO improves over the strongest existing ZO baselines by up to 8.7% relative on average across model settings and up to 26.9% relative on individual tasks, while remaining within 1.5% of FO-Adam on average. These results suggest that in the evaluated fine-tuning settings, structured and spectrally controlled ZO updates can substantially narrow the gap between ZO and FO optimization.

1 Introduction

Fine-tuning large language models (LLMs) has become a central component in adapting large pretrained models to downstream tasks [1]. While first-order (FO) optimizers such as Adam [2] and AdamW [3] are commonly used in fine-tuning, they require full backpropagation through the model and therefore incur substantial memory overhead from storing intermediate activations and optimizer states. These limitations have driven increasing interest in parameter- and memory-efficient fine-tuning methods [4, 5, 6, 7]. Zeroth-Order (ZO) optimization [8], exemplified by MeZO [9], provides a promising alternative by replacing gradient computation with forward-pass-only perturbation estimates. As a result, MeZO reduces memory consumption close to the inference level, making full-parameter LLM fine-tuning feasible under tight memory budgets.

Despite this memory advantage, ZO fine-tuning still suffers from slow convergence and a persistent performance gap to first-order methods, largely due to noisy estimates in high-dimensional and ill-conditioned parameter spaces. Since MeZO, recent work has improved ZO fine-tuning from several perspectives, including low-rank structure or subspace [10, 11, 12], alternative sampling distributions [13, 14], momentum and variance-reduction techniques [15, 16], multi-sample estimators [17], and Hessian- or curvature-informed updates [18, 19]. These advances show that the naive full-space random estimator is far from optimal. However, most existing methods still focus primarily on improving the gradient estimation itself, while remaining limited by query efficiency, search-space quality, or update conditioning. In contrast, FO fine-tuning benefits from more than gradient access alone: it also exploits informative optimization geometry, aggregates rich derivative signals, and applies well-conditioned update transformations. This motivates the

following question: *Can we design a forward-only ZO optimizer that recovers these structural benefits of FO fine-tuning without incurring the memory cost of backpropagation?*

In this paper, we propose **StructZO**, a **structured Zeroth-Order** optimization framework that improves forward-only LLM fine-tuning by performing large-query directional signal aggregation inside activation-induced informative subspaces, thereby converting additional ZO queries into more useful low-dimensional update estimates without storing activations for backpropagation. Since these subspaces are intended to approximate the right-side geometry of the gradient, StructZO applies Muon-style spectral shaping directly in the reduced coordinates before mapping the update back to the original parameter space, producing well-conditioned directions that better reflect FO-style optimization behavior.

Existing works address these factors in isolation, while StructZO provides the first *coupled design*: the informative reduced space determines where ZO queries are spent, large-query estimation aggregates signals within this space, and spectral shaping is applied in the same coordinates before the update is lifted back. This coupling changes the role of multi-query ZO from noisy full-space gradient estimation to structured signal aggregation, making it possible to analyze the resulting variance–bias trade-off. Empirically, StructZO consistently improves over existing ZO baselines under comparable forward-query budgets and narrows the gap to FO-Adam in our evaluated fine-tuning settings.

Our contributions are summarized as follows:

- We introduce StructZO, a unified structured ZO framework that performs estimation and update conditioning in an informative reduced space, recovering several key optimization benefits of FO fine-tuning without backpropagation.
- We show that this structured update space makes large-query ZO estimation effective in practice: additional queries are aggregated along more useful low-dimensional directions, while the resampling trick avoids storing full perturbation directions.
- We provide an MSE analysis that characterizes the variance–bias trade-off of StructZO. The result shows that the ZO estimation dimension is reduced from mn to mr , while the remaining projection bias is controlled by the spectral tail of the activation Gram matrix.
- Experiments on OPT and Llama3.1 models show that StructZO consistently outperforms existing ZO baselines, achieves lower training loss, scales to larger models, and remains close to FO-Adam while preserving the memory advantage of forward-only optimization. For example, StructZO improves the average score on Llama3.1-8B from 76.3 to 82.9 over the strongest ZO baseline, and improves CB from 73.2 to 92.9.

2 Related Work

Fine-tuning LLMs. Fine-tuning large language models with first-order optimizers such as Adam and AdamW is effective but memory-intensive. This motivates parameter- and memory-efficient fine-tuning methods, including adapters [20], prefix tuning [21, 7], LoRA [4], and low-rank optimizer-memory reduction methods such as GaLore [6, 22]. Although these methods reduce the number of trainable parameters or optimizer states, they still rely on backpropagation and therefore retain nontrivial activation-memory cost.

Zeroth-Order Optimization. Zeroth-order optimization estimates update directions using only objective function values and has been widely studied in adversarial learning [23, 24], reinforcement learning [25], and automated ML [26]. Recently, MeZO [9] showed that ZO optimization can fine-tune LLMs with memory close to inference, making it attractive under limited hardware budgets. Recent works improve ZO fine-tuning through low-rank and structured perturbation [10, 11, 12, 13, 14], variance reduction and momentum techniques [15, 16, 17], or Hessian-informed preconditioning [18, 19]. Together, these methods demonstrate that ZO fine-tuning can be substantially improved by exploiting gradient structure, model architecture, and carefully designed perturbation distributions.

Subspace Optimization. Subspace-based optimization exploits the observation that useful update directions in large models often concentrate in a much lower-dimensional structure than the full parameter space [6, 22]. In the ZO setting, low-rank updates are frequently achieved through the use of randomly sampled projection matrices combined with lazy updates [27, 10, 12]. More closely related to our work, AGZO [11] takes an important step by constructing activation-guided subspaces for more informative ZO perturbations. Together, these works motivate a systematic study of the interaction between activation-induced spaces, large-query signal aggregation, and spectral update conditioning in ZO optimization.

3 Method

3.1 Motivation

First-order fine-tuning benefits from more than explicit gradient access. In practice, effective FO updates exploit informative low-dimensional geometry, aggregate derivative signals across many tokens and parameters, and apply well-conditioned update transformations. Standard ZO methods lack these properties: they perturb the full parameter space, rely on very limited directional information per step, and usually apply unconditioned updates. StructZO is designed to recover these structural benefits in a forward-only optimization pipeline by estimating updates in an informative reduced space and applying spectral shaping before lifting them back to the original parameters.

3.2 Preliminaries

Zeroth-Order (ZO) optimization refers to methods that update parameters without explicit gradient information. In this work, we focus on the *Random Gradient Estimator* (RGE), which samples a perturbation $\mathbf{z}$ matching the parameter shape and queries a “noisy function value oracle” to compute updates. RGE admits both forward difference (left) and central difference (right) variants:

$$\tilde{\nabla}_\mu f(\mathbf{x}) = \frac{1}{q} \sum_{i=1}^q \frac{f(\mathbf{x} + \mu \mathbf{z}_i) - f(\mathbf{x})}{\mu} \mathbf{z}_i, \quad \hat{\nabla}_\mu f(\mathbf{x}) = \frac{1}{q} \sum_{i=1}^q \frac{f(\mathbf{x} + \mu \mathbf{z}_i) - f(\mathbf{x} - \mu \mathbf{z}_i)}{2\mu} \mathbf{z}_i, \quad (1)$$

where $\mathbf{z}_i \sim N(\mathbf{0}, \mathbf{I})$. By Taylor expansion, when $\mu \rightarrow 0$, both tend to:

$$\hat{\nabla} f(\mathbf{x}) = \frac{1}{q} \sum_{i=1}^q \langle \nabla f(\mathbf{x}), \mathbf{z}_i \rangle \mathbf{z}_i, \quad (2)$$

which is an unbiased estimate of $\nabla f(\mathbf{x})$ with a variance scaling with the parameter dimension as $\Theta\left(\frac{d}{q}\right)$ and is utilized in *ZO-SGD* to update parameters like $\theta_{t+1} = \theta_t - \eta \hat{\nabla} f(\theta_t)$, where η is the learning rate. For LLM fine-tuning, d is extremely large, so applying large-scale multi-query ZO estimation directly in the full parameter space becomes computationally inefficient [17]. This motivates estimating ZO updates in a lower-dimensional but informative search space.

3.3 StructZO

The overall algorithm is summarized in Algorithm 1

3.3.1 Activation-Induced Informative Subspaces

Consider a Transformer linear layer $Y = AW^\top$, where $A \in \mathbb{R}^{B \times n}$ denotes the input activation matrix of the current mini-batch and $W \in \mathbb{R}^{m \times n}$ is the trainable weight matrix. From back-propagation, $\nabla_W L = \left(\frac{\partial L}{\partial Y}\right)^\top A := \Delta^\top A$ and subsequently:

$$(\nabla_W L)^\top \nabla_W L = A^\top \Delta \Delta^\top A. \quad (3)$$

When the output-side covariance $\Delta \Delta^\top$ is relatively isotropic or slowly varying, the right-side geometry of $\nabla_W L$ is largely governed by the activation Gram matrix $A^\top A$. This suggests that the dominant eigenspace of $A^\top A$ provides an informative coordinate system for ZO perturbations.

Instead of using isotropic random projections, StructZO therefore efficiently constructs an informative subspace directly from activations. Specifically, we sample a Gaussian sketch matrix $\Omega \in \mathbb{R}^{n \times (r+k)}$, where r is the target rank and k is an oversampling factor. We compute $Y = C\Omega = A^\top (A\Omega)$, followed by *QR* decomposition $Y = QR$. The informative subspace basis is then defined as $\mathbf{P} = Q_{[:,r]} \in \mathbb{R}^{n \times r}$. Importantly, this construction is performed entirely from forward activations and does not require storing activations for backward propagation. The resulting projector $\Pi = \mathbf{P}\mathbf{P}^\top$ captures dominant activation directions and defines a reduced optimization space.

3.3.2 Multi-Query Structured Zeroth-Order Estimation

After constructing the informative subspace $\mathbf{P}$, StructZO performs perturbations only inside the reduced coordinates. For each query, we sample $z_i \sim \mathcal{N}(0, I_{m \times r})$, and construct the lifted perturbation $\Delta_i = z_i \mathbf{P}^\top$. Consequently, the

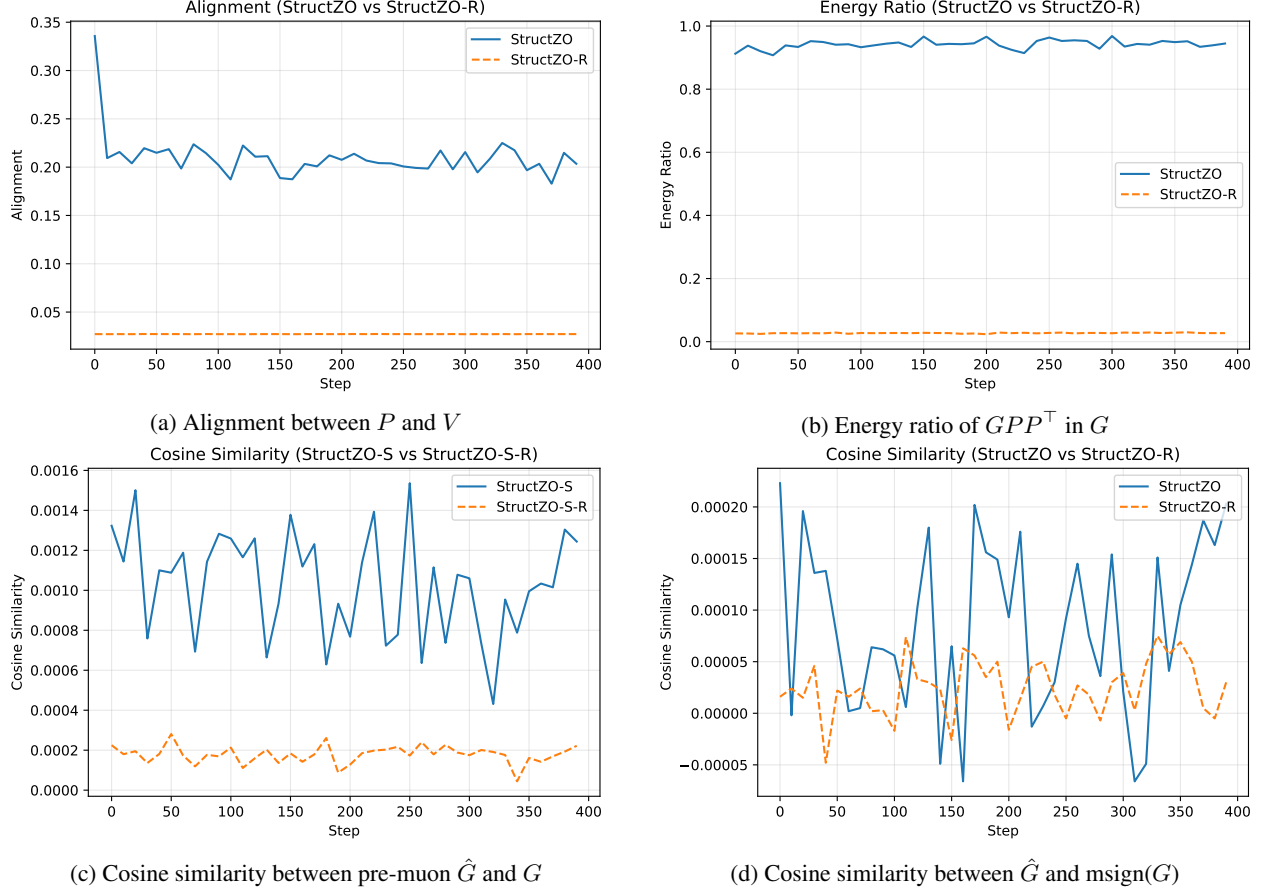


Figure 1: Empirical geometry diagnostics of StructZO on SST-2, OPT1.3B. (a) Subspace alignment $\frac{1}{r} \|P^\top V_r\|_F^2$ between the constructed basis P and the top- r right singular subspace V_r of the true gradient G . (b) Projected gradient energy $\|GPP^\top\|_F^2 / \|G\|_F^2$. (c) Cosine similarity between the pre-spectral estimator and the raw gradient G . (d) Cosine similarity between the final spectrally shaped update and $\text{msign}(G)$. The -R suffix indicates that a random subspace is used to replace the original informative one, and the -S suffix indicates that the Muon-style approach is not used.

aggregated gradient is:

$$\frac{1}{q} \sum_{i=1}^q \frac{f(W + \mu z_i P^\top) - f(W)}{\mu} z_i P^\top. \quad (4)$$

Compared with full-space Gaussian ZO, the perturbation dimension is reduced from mn to mr , significantly lowering estimator variance when $r \ll n$. Unlike random low-dimensional perturbations, the reduced coordinates here are constructed from the current forward activations and are therefore more aligned with optimization-relevant directions, achieving better query efficiency.

3.3.3 Spectral Shaping via Reduced-Space Muon

Although reduced-space perturbations substantially reduce estimation variance, the resulting updates may still suffer from anisotropic singular directions. To further stabilize optimization, we apply Muon-style spectral shaping inside the reduced coordinates and our final update direction is:

$$O = \text{msign}\left(\frac{1}{q} \sum_{i=1}^q \frac{L(W + \mu z_i P^\top) - L(W)}{\mu} z_i\right) P^\top, \quad (5)$$

where $\text{msign}(G) = UV^\top$ for the SVD $G = U\Sigma V^\top$.

Proposition 1. For a matrix G and its SVD decomposition $G = U\Sigma V^\top$, if $P = V$ then $\text{msign}(GP)P^\top = \text{msign}(G)$.

This operation equalizes active singular directions inside the informative subspace, producing well-conditioned updates analogous to spectrally normalized FO optimization. Moreover, according to Proposition 1, StructZO does not merely approximate the raw gradient G ; instead, it approximately reconstructs the spectrally conditioned update direction $\text{msign}(G)$, which is typically more stable and optimization-effective than directly estimating G itself.

3.3.4 Empirical Geometry Diagnostics

We further provide empirical diagnostics to validate whether the informative basis constructed from forward activations captures useful gradient geometry. In Figure 1, we compute true FO gradients only for analysis. Although the constructed basis P does not exactly match the top right singular subspace V_r of the true gradient, it shows substantially stronger alignment than a random basis. The projected energy ratio $\|GPP^\top\|_F^2/\|G\|_F^2$ is also much higher, indicating that the reduced space preserves a large fraction of gradient energy. Before spectral shaping, the structured estimator is closer to the raw gradient G than its random-space counterpart. After spectral shaping, the final StructZO update is more aligned with the spectrally conditioned direction $\text{msign}(G)$. These results support our design choice: the forward-induced basis provides a useful search space, while spectral shaping changes the optimization target from raw gradient approximation to a better-conditioned update direction.

Algorithm 1 StructZO

Require: Parameters $\{\mathbf{W}_l\}_{l=1}^L$, loss L , learning rate η , max iterations T , smoothing coefficient $\mu > 0$, query number q , rank r , over-sample factor k

- 1: **for** $t = 0$ **to** T **do**
- 2: Sample base seed s and batch $\mathcal{B}_t$
- 3: $\ell_0 \leftarrow L(\{\mathbf{W}_l\}_{l=1}^L; \mathcal{B}_t)$ with activations
- 4: Obtain informative subspace $\{P_l\}_{l=1}^L \leftarrow \text{ExtractSubspace}(\{A_l\}_{l=1}^L, r, k)$ on-the-fly
- 5: **for** $i = 1$ **to** q **do**
- 6: $s_i \leftarrow s + i$
- 7: PerturbParameters($\{\mathbf{W}_l\}_{l=1}^L, \{P_l\}_{l=1}^L, \mu, s_i$)
- 8: $\ell_i \leftarrow L(\{\mathbf{W}_l\}_{l=1}^L; \mathcal{B}_t)$
- 9: PerturbParameters($\{\mathbf{W}_l\}_{l=1}^L, \{P_l\}_{l=1}^L, -\mu, s_i$)
- 10: **end for**
- 11: Initialize $\{g_l\}_{l=1}^L \leftarrow \mathbf{0}$
- 12: **for** $i = 1$ **to** q **do**
- 13: Resample $\{z_l\}_{l=1}^L \leftarrow \text{Generator}(s_i)$
- 14: $\{g_l\}_{l=1}^L \leftarrow \{g_l\}_{l=1}^L + \frac{\ell_i - \ell_0}{\mu} \{z_l\}_{l=1}^L$
- 15: **end for**
- 16: $\{\mathbf{W}_l\}_{l=1}^L \leftarrow \{\mathbf{W}_l - \eta \text{msign}(g_l) P_l^\top\}_{l=1}^L$
- 17: **end for**
- 18: **Function** ExtractSubspace(A, r, k):
- 19: Let $d = A.\text{Shape}[1]$, Sample Gaussian sketch matrix $\Omega \in \mathbb{R}^{d \times (r+k)}$
- 20: $Y = A^\top(A\Omega)$, Perform QR decomposition $Y = QR$
- 21: **Return** $P = Q[:, :r]$ $\triangleright$ First r columns of Q
- 22: **Function** PerturbParameters($\{\mathbf{W}_l\}_{l=1}^L, \{P_l\}_{l=1}^L, \mu, s$):
- 23: Reset random number generator seed s
- 24: **for** $l = 1$ **to** L **do**
- 25: Sample low-rank Perturbation z_l
- 26: $\mathbf{W}_l \leftarrow \mathbf{W}_l + \mu z_l P_l^\top$
- 27: **end for**

3.4 MSE Analysis of StructZO

We analyze the pre-spectral estimator to quantify the effect of informative reduced coordinates and multi-query aggregation. Consider one weight matrix $W \in \mathbb{R}^{n \times n}$ and stochastic loss $F_\xi(W)$, where ξ denotes mini-batch randomness. Let

$$f(W) = \mathbb{E}_\xi[F_\xi(W)], \quad G = \nabla f(W), \quad G_\xi = \nabla F_\xi(W).$$

For each mini-batch, StructZO constructs $P_\xi \in \mathbb{R}^{n \times r}$ from the activation Gram matrix $A_\xi^\top A_\xi$ using the randomized range finder described above, and defines $\Pi_\xi = P_\xi P_\xi^\top$. Unlike information-agnostic random subspace methods, here

Π_ξ is constructed from the same mini-batch activation A_ξ and is therefore statistically coupled with ξ . Thus, instead of enforcing $\mathbb{E}[\Pi_\xi] = cI_n$, we directly analyze the coupled expectation over (ξ, Ω) . Given q independent perturbations $z_i \sim \mathcal{N}(0, I_{m \times r})$, the pre-spectral estimator is

$$\hat{G} = \frac{1}{q} \sum_{i=1}^q \frac{F_\xi(W + \mu z_i P_\xi^\top) - F_\xi(W)}{\mu} z_i P_\xi^\top.$$

Theorem 2 (MSE bound for StructZO). *Under the assumptions 4, 5, 6 specified in Appendix A and standard smoothness conditions on F_ξ , the pre-Muon multi-query estimator satisfies*

$$\begin{aligned} \mathbb{E}_{\xi, \Omega, z} \|\hat{G} - G\|_F^2 &\leq \underbrace{\frac{mr + 1}{q} \mathbb{E}_\xi \|G_\xi\|_F^2}_{\text{query-induced variance}} + \underbrace{\beta_G \alpha_{r,k} \mathbb{E}_\xi \sum_{j>r} \lambda_j (A_\xi^\top A_\xi)}_{\text{subspace approximation bias}} \\ &\quad + \underbrace{\sigma^2}_{\text{stochastic noise}} + \underbrace{O(\mu^2)}_{\text{smoothing error}}. \end{aligned}$$

Remark 3 (Interpretation). *The bound contains two key terms. The first term is the multi-query ZO estimation variance inside the reduced space. Compared with full-space Gaussian ZO, its dimension factor changes from mn to mr , yielding a variance reduction controlled by r/n when $r \ll n$. The second term is the activation-subspace projection bias. It is controlled by the spectral tail of the activation Gram matrix $A_\xi^\top A_\xi$. Therefore, when the activation spectrum has low effective rank, the informative subspace can keep most useful gradient directions while avoiding full-dimensional zeroth-order perturbations.*

4 Experiments

Overview. We evaluate StructZO on autoregressive LLMs across classification, multiple-choice, and generation tasks from GLUE [28] and SuperGLUE [29], comparing with representative ZO baselines, including MeZO [9], LOZO [10], and HiZOO/HiZOO-L [18], as well as FO-Adam, on OPT-1.3B and Llama3.1-8B, and further test scalability on OPT-13B and OPT-30B. We report downstream performance, loss convergence, ablations, memory usage, and wall-clock time. Empirically, StructZO consistently outperforms existing ZO methods and substantially narrows the gap to FO-Adam. Its advantage comes from jointly leveraging informative subspaces, multi-query signal aggregation, and spectral shaping, which yields faster and deeper loss reduction while retaining the low-memory benefit of forward-only ZO optimization.

Setup. We implement all methods in the same training pipeline with identical data preprocessing, prompt templates, batching, and evaluation protocols. Following prior ZO fine-tuning work [9, 10, 18], each task uses 1,024 training examples, 512 development examples, and 1,024 evaluation examples; for small datasets such as CB and COPA, we reserve 100 examples for validation and use the rest for training. Unless otherwise specified, all ZO methods use batch size 16. Single-query ZO baselines are trained for **20K** steps, while StructZO uses $q = 100$ queries and is trained for **400** steps, giving a comparable forward-query scale but richer signal aggregation per update. FO-Adam is trained for 5 epochs with smaller task-dependent batch sizes when necessary. Hyperparameters are selected on the development set, and all experiments are conducted on a single NVIDIA H20 GPU with 96 GiB memory. Additional details are in Appendix B.

4.1 Main Results

Table 1 reports the main results on OPT-1.3B and Llama3.1-8B. StructZO consistently achieves the best performance among all ZO methods across classification, multiple-choice, and generation tasks, with up to 8.7% average improvement over the strongest ZO baseline and only about 1.5% average gap to FO-Adam. The gains are particularly pronounced on challenging tasks such as CB on Llama3.1-8B, where StructZO improves over the best ZO baseline by 26.9%, showing that the proposed structured update space can substantially narrow the ZO–FO performance gap. Figure 2 further compares training loss under the same forward-pass budget: StructZO not only reduces loss faster than MeZO, LOZO, and HiZOO/HiZOO-L, but also converges to a lower final loss, indicating stronger optimization effectiveness rather than merely better checkpoint selection. Finally, Table 2 extends the comparison to OPT-13B and OPT-30B, where StructZO remains consistently superior to existing ZO baselines. Overall, these results suggest that StructZO scales favorably with model size and provides a practical forward-only alternative to FO fine-tuning.

Table 1: Evaluation on OPT-1.3B and Llama3.1-8B using 1,024 training and 512 validation samples per task (for CB and COPA we reserve 100 validation samples and use the remainder for training). With the exception of the DROP task, where F1 is used, all other figures refer to accuracy (%). Each model block lists zeroth-order (ZO) methods first, followed by the first-order (FO) baseline separated by a rule. **Bold** entries indicate the best score among ZO methods; FO rows are provided for reference.

Model Task type	Method	SST-2	RTE	CB	BoolQ	WiC	MultiRC	COPA	ReCoRD	DROP	AVG
		classification						_multiple choice_		generation	
OPT-1.3B	MeZO	88.8	66.4	67.8	63.4	57.9	56.2	73.0	70.7	22.4	63.0
	LOZO	89.4	61.3	67.8	60.8	55.1	58.6	73.0	71.2	21.9	62.1
	HiZOO	89.6	61.7	67.8	63.1	60.2	58.9	73.0	71.8	22.5	63.2
	StructZO	92.4	69.0	71.4	68.5	62.6	63.9	74.0	71.8	26.7	66.7
	FO-Adam	93.0	70.3	71.4	69.4	63.0	64.2	74.0	71.7	27.7	67.2
Llama3.1-8B	MeZO	90.2	70.7	69.6	79.1	58.7	78.1	90.0	84.9	62.2	75.9
	LOZO	91.3	70.7	73.2	80.2	58.7	73.8	91.0	84.9	62.7	76.3
	HiZOO-L	90.2	71.2	71.4	79.6	55.1	77.8	90.0	84.8	58.5	75.4
	StructZO	93.7	84.5	92.9	84.1	66.8	83.2	91.0	84.8	65.5	82.9
	FO-Adam	94.7	88.0	91.0	87.9	70.4	82.5	91.0	84.0	67.2	84.1

Model	Method	SST-2	RTE	BoolQ	WiC	AVG
OPT-13B	MeZO	91.3	67.5	68.0	59.5	71.6
	LOZO	91.7	70.4	71.9	60.8	73.7
	HiZOO-L	92.1	68.2	66.6	59.4	71.6
	StructZO	93.7	72.3	78.8	64.0	77.2
OPT-30B	MeZO	90.7	67.1	67.4	56.3	70.4
	LOZO	92.0	65.3	72.3	57.2	71.7
	HiZOO-L	90.3	69.3	68.4	58.6	71.7
	StructZO	93.2	79.5	78.7	60.4	78.0

Table 2: Full-parameter ZO fine-tuning results on OPT-13B and OPT-30B.

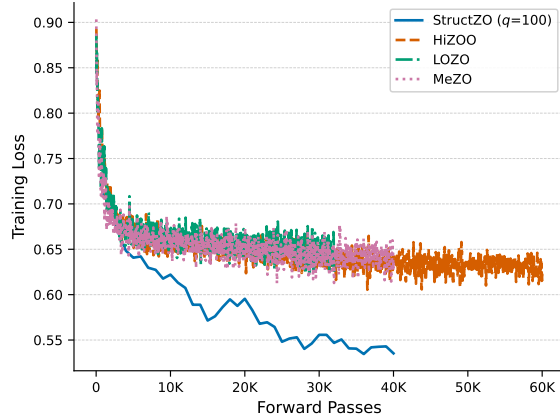
4.2 Ablation Study

Table 3 studies two ablated variants of StructZO: StructZO-R uses a random reduced space, while StructZO-S replaces spectral shaping with SGD-style updates. The full StructZO consistently outperforms both variants on OPT-1.3B and Llama3.1-8B. The clear drop from removing either part indicates that the effectiveness of StructZO comes from their joint interaction rather than from any single isolated component. Figure 3 further examines the effect of the query number q . Increasing q leads to faster and lower loss reduction, suggesting that once queries are collected in a more informative reduced space, larger query budgets can be effectively converted into more accurate update directions, similar in spirit to the signal aggregation in FO optimization. Notably, although the SGD-style variant can be closer to the raw gradient G shown in Figure 1, it still lags behind the full StructZO, suggesting that the spectrally conditioned direction, rather than raw gradient approximation alone, is crucial for effective optimization.

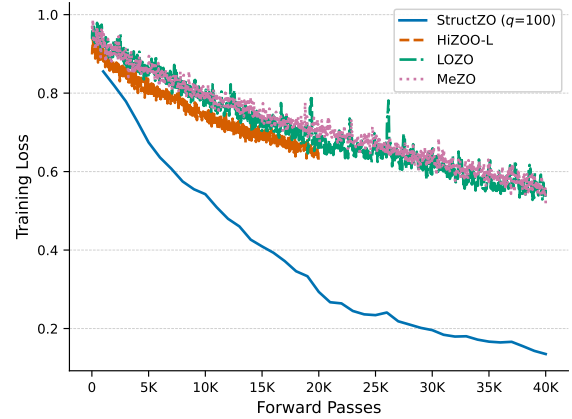
4.3 Memory and Wall-Clock Efficiency

Memory overhead. Figure 4 compares peak GPU memory usage. StructZO avoids backpropagation, backward activation storage, and Adam optimizer states, so its only major extra memory over MeZO is the informative basis $P \in \mathbb{R}^{n \times r}$ for each weight matrix $W \in \mathbb{R}^{m \times n}$, costing $O(rn)$ storage. With the resampling trick, perturbations and reduced-space estimates are regenerated from random seeds, avoiding storage of all q query directions. Therefore, StructZO adds only modest memory overhead compared with MeZO, avoids the per-parameter curvature states used by HiZOO-style methods, and still achieves up to $11.2\times$ memory reduction over FO-Adam.

Wall-clock time. Table 4 reports the wall-clock training time. Since StructZO uses multi-query estimation with $q = 100$, each optimization step is slower than single-query ZO baselines. However, each step also aggregates richer directional information in an informative reduced space, allowing StructZO to use far fewer optimization steps. Despite



(a) Training loss versus forward passes on BoolQ, OPT1.3B.



(b) Training loss versus forward passes on CB, Llama3.1-8B.

Figure 2: Training loss versus forward passes for (a) BoolQ on OPT1.3B and (b) CB on Llama3.1-8B, comparing MeZO, LOZO, HiZOO(-L) and StructZO.

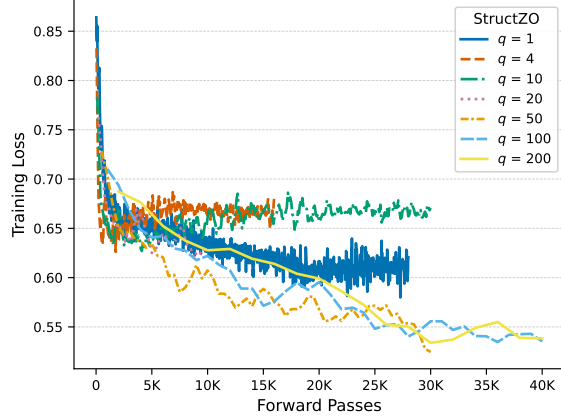
Table 3: Ablation study of StructZO on OPT-1.3B and Llama3.1-8B across classification, multiple-choice, and generation tasks. StructZO-R replaces the activation-induced subspace with a random subspace, while StructZO-S replaces Muon-style spectral shaping with SGD-style updates.

Model	Method	SST-2	RTE	CB	BoolQ	WiC	MultiRC	COPA	ReCoRD	DROP	AVG
Task type		classification						multiple choice		generation	
OPT-1.3B	StructZO-R	91.7	59.5	71.4	64.3	56.1	59.0	74.0	72.6	22.8	63.5
	StructZO-S	91.8	62.0	67.8	63.2	57.5	60.2	72.0	72.0	25.3	63.5
	StructZO	92.4	69.0	71.4	68.5	62.6	63.9	74.0	71.8	26.7	66.7
Llama3.1-8B	StructZO-R	88.3	66.7	73.2	77.7	62.5	77.1	90.0	86.5	62.4	76.0
	StructZO-S	92.8	76.1	75.0	84.0	61.5	73.0	90.0	86.4	57.3	77.3
	StructZO	93.7	84.5	92.9	84.1	66.8	83.2	91.0	84.8	65.5	82.9

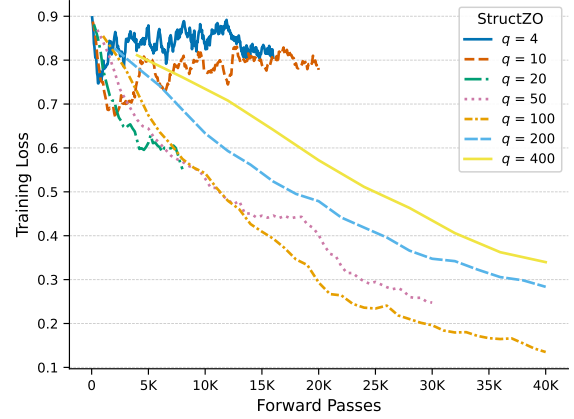
its higher per-step cost, StructZO achieves lower or comparable total training time than MeZO and LOZO, and is substantially faster than HiZOO/HiZOO-L. Overall, StructZO trades higher per-step cost for much fewer and more effective updates, resulting in a favorable end-to-end efficiency.

5 Conclusion

We introduced StructZO, a structured zeroth-order optimization framework for memory-efficient LLM fine-tuning. By aggregating large-query directional signals in an informative subspace and applying spectral shaping in reduced coordinates, StructZO improves the effectiveness of forward-only optimization. As a limitation, StructZO depends on the forward-induced subspace capturing optimization-relevant directions. When this structure is weak, the benefit may decrease, as suggested by the projection-bias term in our analysis. In addition, our experiments focus on GLUE/SuperGLUE-style limited-data fine-tuning, and extending the evaluation to longer-context and more complex generation tasks remains promising future work.



(a) Training loss versus forward passes on BoolQ, OPT1.3B.



(b) Training loss versus forward passes on CB, Llama3.1-8B.

Figure 3: Training loss versus forward passes for (a) BoolQ on OPT1.3B and (b) CB on Llama3.1-8B, comparing different query number q .

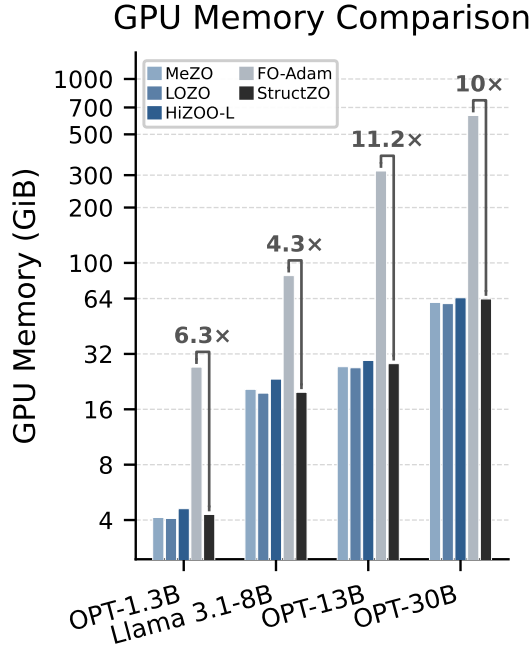


Figure 4: Memory overhead for different models under various ZO algorithms and FO-Adam. Details are shown in Appendix B.

Task	Method	Time / step (s)	Training steps	Total time (h)
SST-2	MeZO	0.658	20K	3.656
	LOZO	0.658	20K	3.654
	HiZOO-L	1.435	20K	7.970
	StructZO	26.58	400	2.954
RTE	MeZO	1.762	20K	9.79
	LOZO	1.762	20K	9.79
	HiZOO-L	3.653	20K	20.30
	StructZO	82.53	400	9.17
BoolQ	MeZO	2.698	20K	14.99
	LOZO	2.698	20K	14.99
	HiZOO-L	5.507	20K	30.60
	StructZO	132.87	400	14.76

Table 4: Wall-clock time for SST-2, RTE and BoolQ on Llama3.1-8B. We measured the time overhead for the first 100 steps and the first 10 steps for both ZO baselines and StructZO respectively, and thereby estimated their per-step and total execution times.

References

- [1] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, “Exploring the limits of transfer learning with a unified text-to-text transformer,” *Journal of Machine Learning Research*, 2020.
- [2] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in *International Conference on Learning Representations*, 2015.
- [3] I. Loshchilov and F. Hutter, “Decoupled weight decay regularization,” in *International Conference on Learning Representations*, 2019.

- [4] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, “LoRA: Low-rank adaptation of large language models,” in *International Conference on Learning Representations*, 2022.
- [5] N. Ding, Y. Qin, G. Yang, F. Wei, Z. Yang, Y. Su, S. Hu, Y. Chen, C.-M. Chan, W. Chen *et al.*, “Parameter-efficient fine-tuning of large-scale pre-trained language models,” *Nature Machine Intelligence*, 2023.
- [6] J. Zhao, Z. Zhang, B. Chen, Z. Wang, A. Anandkumar, and Y. Tian, “GaLore: Memory-efficient LLM training by gradient low-rank projection,” in *International Conference on Machine Learning*, 2024.
- [7] X. L. Li and P. Liang, “Prefix-tuning: Optimizing continuous prompts for generation,” in *Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, 2021.
- [8] Y. Nesterov and V. Spokoiny, “Random gradient-free minimization of convex functions,” *Foundations of Computational Mathematics*, 2017.
- [9] S. Malladi, T. Gao, E. Nichani, A. Damian, J. D. Lee, D. Chen, and S. Arora, “Fine-tuning language models with just forward passes,” *Advances in Neural Information Processing Systems*, 2023.
- [10] Y. Chen, Y. Zhang, L. Cao, K. Yuan, and Z. Wen, “Enhancing zeroth-order fine-tuning for language models with low-rank structures,” in *International Conference on Learning Representations*, 2025.
- [11] W. Lin, Y. Jiang, Q. Song, Q. Xiang, and H. Xu, “AGZO: activation-guided zeroth-order optimization for LLM fine-tuning,” *CoRR*, vol. abs/2601.17261, 2026.
- [12] Y. Lang, C. Wang, Y. Zhang, M. Hong, Z. Zhang, W. Yin, and S. Liu, “Powering up zeroth-order training via subspace gradient orthogonalization,” *CoRR*, vol. abs/2602.17155, 2026.
- [13] S. Ma and H. Huang, “Revisiting zeroth-order optimization: Minimum-variance two-point estimators and directionally aligned perturbations,” in *International Conference on Learning Representations*, 2025.
- [14] S. Dang, Y. Guo, Y. Zhao, H. Ye, X. Zheng, G. Dai, and I. Tsang, “FZOO: Fast zeroth-order optimizer for fine-tuning large language models towards adam-scale speed,” *arXiv preprint arXiv:2506.09034*, 2025.
- [15] P. Yue, X. Yang, M. Xiao, and Z. Lin, “PseuZO: Pseudo-zeroth-order algorithm for training deep neural networks,” in *Conference on Neural Information Processing Systems*, 2025.
- [16] T. Gautam, Y. Park, H. Zhou, P. Raman, and W. Ha, “Variance-reduced zeroth-order methods for fine-tuning language models,” in *International Conference on Machine Learning*, 2024.
- [17] Y. Peng, Y. Liu, F. Wen, and X. Chen, “MUZO: leveraging multiple queries and momentum for zeroth-order fine-tuning of large language models,” in *Conference on Empirical Methods in Natural Language Processing*, 2025.
- [18] Y. Zhao, S. Dang, H. Ye, G. Dai, Y. Qian, and I. W. Tsang, “Second-order fine-tuning without pain for LLMs: A Hessian informed zeroth-order optimizer,” in *International Conference on Learning Representations*, 2025.
- [19] H. Zhao, J. Li, Y. Pan, S. Liang, X. Yang, F. Dou, T. Liu, and J. Lu, “Helene: Hessian layer-wise clipping and gradient annealing for accelerating fine-tuning LLM with zeroth-order optimization,” in *Conference on Empirical Methods in Natural Language Processing*, 2025.
- [20] N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. De Laroussilhe, A. Gesmundo, M. Attariyan, and S. Gelly, “Parameter-efficient transfer learning for nlp,” in *International Conference on Machine Learning*, 2019.
- [21] B. Lester, R. Al-Rfou, and N. Constant, “The power of scale for parameter-efficient prompt tuning,” in *Conference on Empirical Methods in Natural Language Processing*, 2021.
- [22] Y. He, P. Li, Y. Hu, C. Chen, and K. Yuan, “Subspace optimization for large language models with convergence guarantees,” in *International Conference on Machine Learning*, 2025.
- [23] P.-Y. Chen, H. Zhang, Y. Sharma, J. Yi, and C.-J. Hsieh, “Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models,” in *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, 2017.
- [24] A. Ilyas, L. Engstrom, A. Athalye, and J. Lin, “Black-box adversarial attacks with limited queries and information,” in *International Conference on Machine Learning*, 2018.
- [25] A. Vemula, W. Sun, and J. Bagnell, “Contrasting exploration in parameter and action space: A zeroth-order optimization perspective,” in *International Conference on Artificial Intelligence and Statistics*, 2019.
- [26] X. Wang, W. Guo, J. Su, X. Yang, and J. Yan, “Zarts: On zero-order optimization for neural architecture search,” *Advances in Neural Information Processing Systems*, 2022.

- [27] Z. Yu, P. Zhou, S. Wang, J. Li, M. Tian, and H. Huang, “Zeroth-order fine-tuning of llms in random subspaces,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2025, pp. 4475–4485.
- [28] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. 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*, 2018.
- [29] A. Wang, Y. Pruksachatkun, N. Nangia, A. Singh, J. Michael, F. Hill, O. Levy, and S. Bowman, “SuperGLUE: A stickier benchmark for general-purpose language understanding systems,” *Advances in Neural Information Processing Systems*, 2019.
- [30] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Y. Ng, and C. Potts, “Recursive deep models for semantic compositionality over a sentiment treebank,” in *Conference on Empirical Methods in Natural Language Processing*, 2013.
- [31] I. Dagan, O. Glickman, and B. Magnini, “The pascal recognising textual entailment challenge,” in *Machine Learning Challenges Workshop*, 2005.
- [32] M.-C. De Marneffe, M. Simons, and J. Tonhauser, “The commitmentbank: Investigating projection in naturally occurring discourse,” in *proceedings of Sinn und Bedeutung*, 2019.
- [33] C. Clark, K. Lee, M. Chang, T. Kwiatkowski, M. Collins, and K. Toutanova, “BoolQ: Exploring the surprising difficulty of natural yes/no questions,” in *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2019.
- [34] M. T. Pilehvar and J. Camacho-Collados, “WiC: the word-in-context dataset for evaluating context-sensitive meaning representations,” in *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2019.
- [35] D. Khashabi, S. Chaturvedi, M. Roth, S. Upadhyay, and D. Roth, “Looking beyond the surface: A challenge set for reading comprehension over multiple sentences,” in *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2018.
- [36] M. Roemmele, C. A. Bejan, and A. S. Gordon, “Choice of plausible alternatives: An evaluation of commonsense causal reasoning,” in *AAAI Spring Symposium: Logical Formalizations of Commonsense Reasoning*, 2011.
- [37] S. Zhang, X. Liu, J. Liu, J. Gao, K. Duh, and B. Van Durme, “Record: Bridging the gap between human and machine commonsense reading comprehension,” *ArXiv preprint arXiv:1810.12885*, 2018.
- [38] D. Dua, Y. Wang, P. Dasigi, G. Stanovsky, S. Singh, and M. Gardner, “DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs,” *arXiv preprint arXiv:1903.00161*, 2019.

A Proof

Proof of Proposition 1. Recall that the matrix sign operation used in Muon can be written as the polar factor

$$\text{msign}(X) = X(X^\top X)^{-1/2},$$

which is equivalent to $\text{msign}(X) = U_X V_X^\top$ for the SVD $X = U_X \Sigma_X V_X^\top$. Let the SVD of G be

$$G = U \Sigma V^\top.$$

When $P = V$, we have

$$GP = GV = U \Sigma V^\top V = U \Sigma.$$

Therefore,

$$\begin{aligned} \text{msign}(GP) &= GP((GP)^\top (GP))^{-1/2} \\ &= U \Sigma((U \Sigma)^\top (U \Sigma))^{-1/2} \\ &= U \Sigma(\Sigma^\top \Sigma)^{-1/2}. \end{aligned}$$

Since Σ is diagonal with nonnegative singular values, we have

$$\Sigma(\Sigma^\top \Sigma)^{-1/2} = I$$

on the singular subspace. Hence

$$\text{msign}(GP) = U.$$

Multiplying by $P^\top = V^\top$ gives

$$\text{msign}(GP)P^\top = UV^\top = \text{msign}(G).$$

This completes the proof. $\square$

Interpretation as approximate low-rank spectral shaping. Proposition 1 gives an exact identity when the reduced basis matches the right singular space of the gradient. In practice, the activation-induced basis only approximates this space. We therefore do not claim exact recovery of the full-space Muon update; instead, the empirical diagnostics in Figure 1 show that the constructed basis captures substantially more gradient-relevant right-side structure than a random basis, which explains why applying spectral shaping in the reduced coordinates can be effective.

For Theorem 2, we use the following assumptions.

Assumption 4 (Bounded stochastic gradient noise). *There exists $\sigma^2 > 0$ such that*

$$\mathbb{E}_\xi \|G_\xi - G\|_F^2 \leq \sigma^2.$$

Assumption 5 (Projected alignment assumption). *Let $C_\xi = A_\xi^\top A_\xi$. There exists $\beta_G > 0$ such that*

$$\text{tr}(G^\top G(I - \Pi_\xi)) \leq \beta_G \text{tr}(C_\xi(I - \Pi_\xi))$$

for the activation-induced projector Π_ξ .

Assumption 6 (Activation Gram sketchability). *There exists $\alpha_{r,k} \geq 1$ such that the randomized QR subspace satisfies*

$$\mathbb{E}_\Omega [\text{tr}(C_\xi(I - \Pi_\xi)) \mid \xi] \leq \alpha_{r,k} \sum_{j>r} \lambda_j(C_\xi),$$

where $\lambda_1(C_\xi) \geq \dots \geq \lambda_n(C_\xi) \geq 0$ are the eigenvalues of C_ξ .

A.1 A Coupled MSE Decomposition

We first prove a decomposition for the pre-Muon estimator. Throughout this section, we fix one weight matrix $W \in \mathbb{R}^{m \times n}$. Let $F_\xi(W)$ be the stochastic loss, and define

$$G_\xi = \nabla F_\xi(W), \quad G = \nabla f(W) = \mathbb{E}_\xi[G_\xi].$$

Let $P_\xi = P(A_\xi, \Omega) \in \mathbb{R}^{n \times r}$ be the activation-induced subspace obtained from $A_\xi^\top A_\xi \Omega$ by QR decomposition, and let

$$\Pi_\xi = P_\xi P_\xi^\top.$$

We assume $P_\xi^\top P_\xi = I_r$, so that Π_ξ is an orthogonal projector.

Proposition 7 (Coupled MSE decomposition). *Let $z_1, \dots, z_q \stackrel{\text{i.i.d.}}{\sim} \mathcal{N}(0, I_{mr})$ be independent of (ξ, Ω) , and define*

$$\widehat{G} = \frac{1}{q} \sum_{i=1}^q \frac{F_\xi(W + \mu z_i P_\xi^\top) - F_\xi(W)}{\mu} z_i P_\xi^\top.$$

Under standard smoothness conditions on F_ξ ,

$$\begin{aligned} \mathbb{E}_{\xi, \Omega, z} \|\widehat{G} - G\|_F^2 &= \frac{mr+1}{q} \mathbb{E}_{\xi, \Omega} \text{tr}(G_\xi^\top G_\xi \Pi_\xi) \\ &\quad + \mathbb{E}_{\xi, \Omega} \text{tr}((G_\xi - G)^\top (G_\xi - G) \Pi_\xi) \\ &\quad + \mathbb{E}_{\xi, \Omega} \text{tr}(G^\top G (I - \Pi_\xi)) + O(\mu^2). \end{aligned}$$

Proof. We decompose the finite-difference estimator into its first-order part and a smoothing remainder. Define

$$\widehat{G}^{(1)} = \frac{1}{q} \sum_{i=1}^q \langle G_\xi, z_i P_\xi^\top \rangle z_i P_\xi^\top.$$

Since

$$\langle G_\xi, z_i P_\xi^\top \rangle = \langle G_\xi P_\xi, z_i \rangle,$$

we can write

$$\widehat{G}^{(1)} = \frac{1}{q} \sum_{i=1}^q \langle G_\xi P_\xi, z_i \rangle z_i P_\xi^\top.$$

The difference between $\widehat{G}$ and $\widehat{G}^{(1)}$ is the finite-difference smoothing error. Under standard smoothness conditions, its contribution to the squared error is $O(\mu^2)$.

We first analyze $\widehat{G}^{(1)}$. Conditioning on (ξ, Ω) , both G_ξ and P_ξ are fixed. Let

$$H_\xi = G_\xi P_\xi \in \mathbb{R}^{m \times r}.$$

Then

$$\widehat{G}^{(1)} = \left(\frac{1}{q} \sum_{i=1}^q \langle H_\xi, z_i \rangle z_i \right) P_\xi^\top.$$

Because $z_i \sim \mathcal{N}(0, I_{mr})$, we have

$$\mathbb{E}_z[\langle H_\xi, z_i \rangle z_i \mid \xi, \Omega] = H_\xi.$$

Therefore,

$$\mathbb{E}_z[\widehat{G}^{(1)} \mid \xi, \Omega] = H_\xi P_\xi^\top = G_\xi P_\xi P_\xi^\top = G_\xi \Pi_\xi.$$

Next, we compute the conditional variance. Let

$$h_\xi = \text{vec}(H_\xi), \quad u_i = \text{vec}(z_i).$$

Then $u_i \sim \mathcal{N}(0, I_{mr})$ and

$$\text{vec}(\langle H_\xi, z_i \rangle z_i) = (u_i^\top h_\xi) u_i.$$

Using the Gaussian fourth-moment identity,

$$\mathbb{E}[(u^\top h)^2 u u^\top] = \|h\|_2^2 I + 2h h^\top,$$

we obtain

$$\text{Cov}((u^\top h) u) = \|h\|_2^2 I + h h^\top.$$

Hence

$$\mathbb{E}_z \left[\left\| \frac{1}{q} \sum_{i=1}^q \langle H_\xi, z_i \rangle z_i - H_\xi \right\|_F^2 \mid \xi, \Omega \right] = \frac{mr+1}{q} \|H_\xi\|_F^2.$$

Since $P_\xi^\top P_\xi = I_r$, right multiplication by $P_\xi^\top$ preserves the Frobenius norm of matrices in $\mathbb{R}^{m \times r}$:

$$\|M P_\xi^\top\|_F^2 = \text{tr}(M P_\xi^\top P_\xi M^\top) = \|M\|_F^2.$$

Thus,

$$\mathbb{E}_z \left[\|\widehat{G}^{(1)} - G_\xi \Pi_\xi\|_F^2 \middle| \xi, \Omega \right] = \frac{mr+1}{q} \|G_\xi P_\xi\|_F^2.$$

Equivalently,

$$\|G_\xi P_\xi\|_F^2 = \text{tr}(G_\xi^\top G_\xi \Pi_\xi).$$

Now decompose the error to the full gradient G :

$$\widehat{G}^{(1)} - G = (\widehat{G}^{(1)} - G_\xi \Pi_\xi) + (G_\xi \Pi_\xi - G).$$

The cross term vanishes after conditioning on (ξ, Ω) because

$$\mathbb{E}_z[\widehat{G}^{(1)} - G_\xi \Pi_\xi \mid \xi, \Omega] = 0.$$

Therefore,

$$\mathbb{E}_z \left[\|\widehat{G}^{(1)} - G\|_F^2 \middle| \xi, \Omega \right] = \frac{mr+1}{q} \text{tr}(G_\xi^\top G_\xi \Pi_\xi) + \|G_\xi \Pi_\xi - G\|_F^2.$$

It remains to expand the second term. Let

$$E_\xi = G_\xi - G.$$

Then

$$G_\xi \Pi_\xi - G = (G + E_\xi) \Pi_\xi - G = E_\xi \Pi_\xi - G(I - \Pi_\xi).$$

Because Π_ξ is an orthogonal projector,

$$\Pi_\xi(I - \Pi_\xi) = 0.$$

Thus the two components are orthogonal in Frobenius inner product:

$$\langle E_\xi \Pi_\xi, G(I - \Pi_\xi) \rangle_F = \text{tr}(\Pi_\xi E_\xi^\top G(I - \Pi_\xi)) = 0.$$

Consequently,

$$\|G_\xi \Pi_\xi - G\|_F^2 = \|E_\xi \Pi_\xi\|_F^2 + \|G(I - \Pi_\xi)\|_F^2.$$

Equivalently,

$$\|E_\xi \Pi_\xi\|_F^2 = \text{tr}(E_\xi^\top E_\xi \Pi_\xi),$$

and

$$\|G(I - \Pi_\xi)\|_F^2 = \text{tr}(G^\top G(I - \Pi_\xi)).$$

Taking expectation over (ξ, Ω) and adding the smoothing remainder $O(\mu^2)$ completes the proof. $\square$

A.2 Proof of Theorem 2

Proof. Starting from Proposition ??, we bound the three leading terms separately.

For the ZO query variance term, since $0 \preceq \Pi_\xi \preceq I_n$,

$$\text{tr}(G_\xi^\top G_\xi \Pi_\xi) = \|G_\xi P_\xi\|_F^2 \leq \|G_\xi\|_F^2.$$

Therefore,

$$\frac{mr+1}{q} \mathbb{E}_{\xi, \Omega} \text{tr}(G_\xi^\top G_\xi \Pi_\xi) \leq \frac{mr+1}{q} \mathbb{E}_\xi \|G_\xi\|_F^2.$$

For the stochastic gradient noise term, using again $0 \preceq \Pi_\xi \preceq I_n$, we have

$$\text{tr}((G_\xi - G)^\top (G_\xi - G) \Pi_\xi) = \|(G_\xi - G) P_\xi\|_F^2 \leq \|G_\xi - G\|_F^2.$$

By the bounded stochastic gradient noise assumption,

$$\mathbb{E}_{\xi, \Omega} \text{tr}((G_\xi - G)^\top (G_\xi - G) \Pi_\xi) \leq \sigma^2.$$

For the projection bias term, the activation-gradient alignment assumption gives

$$\text{tr}(G^\top G(I - \Pi_\xi)) \leq \beta_G \text{tr}(C_\xi(I - \Pi_\xi)).$$

Taking conditional expectation over Ω and using activation Gram sketchability,

$$\mathbb{E}_\Omega [\text{tr}(G^\top G(I - \Pi_\xi)) \mid \xi] \leq \beta_G \alpha_{r,k} \sum_{j>r} \lambda_j(C_\xi).$$

Taking expectation over ξ yields

$$\mathbb{E}_{\xi,\Omega} \text{tr}(G^\top G(I - \Pi_\xi)) \leq \beta_G \alpha_{r,k} \mathbb{E}_\xi \sum_{j>r} \lambda_j(C_\xi).$$

Since $C_\xi = A_\xi^\top A_\xi$, combining the above inequalities gives

$$\begin{aligned} \mathbb{E}_{\xi,\Omega,z} \|\hat{G} - G\|_F^2 &\leq \frac{mr+1}{q} \mathbb{E}_\xi \|G_\xi\|_F^2 + \beta_G \alpha_{r,k} \mathbb{E}_\xi \sum_{j>r} \lambda_j(A_\xi^\top A_\xi) \\ &\quad + \sigma^2 + O(\mu^2). \end{aligned}$$

This completes the proof. $\square$

On Projected alignment assumption. For a linear layer, the stochastic gradient often admits the form

$$G_\xi = \Delta_\xi^\top A_\xi,$$

where A_ξ is the input activation and Δ_ξ is the output-side error signal. Hence

$$G_\xi^\top G_\xi = A_\xi^\top \Delta_\xi \Delta_\xi^\top A_\xi.$$

If the output-side error satisfies $\Delta_\xi \Delta_\xi^\top \preceq \beta_\xi I$, then

$$G_\xi^\top G_\xi \preceq \beta_\xi A_\xi^\top A_\xi.$$

This shows that the right-side gradient energy is controlled by the activation Gram matrix. Therefore, directions with small activation Gram energy are unlikely to carry large stable gradient energy, which motivates the activation-gradient alignment assumption.

On activation Gram sketchability. The construction

$$Y_\xi = A_\xi^\top (A_\xi \Omega) = C_\xi \Omega$$

is a randomized range finder applied to the positive semidefinite activation Gram matrix $C_\xi = A_\xi^\top A_\xi$. The QR step extracts an orthonormal basis for the sketched range. The sketchability assumption states that this randomized subspace captures the dominant eigenspace of C_ξ up to a multiplicative factor $\alpha_{r,k}$. The right-hand side

$$\sum_{j>r} \lambda_j(C_\xi)$$

is the optimal rank- r residual trace by the Ky Fan principle. Thus, when the activation Gram matrix has fast spectral decay, the residual projection error is small.

Remark 8 (Comparison with full-space and random-subspace ZO). *For full-space Gaussian ZO with the same query number q , the leading variance term scales as*

$$\frac{mn+1}{q} \mathbb{E}_\xi \|G_\xi\|_F^2.$$

In contrast, our activation-induced estimator has

$$\frac{mr+1}{q} \mathbb{E}_{\xi,\Omega} \text{tr}(G_\xi^\top G_\xi \Pi_\xi) \leq \frac{mr+1}{q} \mathbb{E}_\xi \|G_\xi\|_F^2.$$

Thus, the ZO search dimension is reduced from mn to mr .

For an isotropic random rank- r projector Π_{rand} satisfying $\mathbb{E}[\Pi_{\text{rand}}] = \frac{r}{n} I_n$, the projection bias is

$$\mathbb{E}_{\Pi_{\text{rand}}} \|G(I - \Pi_{\text{rand}})\|_F^2 = \left(1 - \frac{r}{n}\right) \|G\|_F^2.$$

When $r \ll n$, this random projection discards most of the full-gradient energy. By contrast, our activation-induced projector has projection bias controlled by

$$\mathbb{E}_\xi \sum_{j>r} \lambda_j(A_\xi^\top A_\xi),$$

which can be much smaller when the activation Gram matrix has low effective rank.

B Experimental details

Table 5: The hyperparameter grids for OPT-1.3B and Llama3.1-8B; the weight decay is set to zero.

Experiment	Hyperparameters	Values
MeZO	Batch size	16
	Learning rate	$\{1e-6, 1e-7\}$
	μ	$1e-3$
HiZOO	Batch size	16
	Learning rate	$\{1e-6, 1e-7\}$
	μ	$1e-3$
	α_{HiZOO}	$\{1e-7, 1e-8\}$
HiZOO-L	Batch size	16
	Learning rate	$\{1e-6, 1e-7\}$
	μ	$1e-3$
	α_{HiZOO}	$\{1e-9, 1e-10\}$
LOZO	Batch size	16
	Learning rate	$\{1e-6, 1e-7\}$
	μ	$1e-3$
	r	$\{1, 2\}$
	Step interval	$\{50, 100\}$
StructZO	Batch size	16
	Learning rate	$\{1e-2, 1e-3\}$
	μ	$1e-3$
	k	10
	r	64
StructZO-S	Batch size	16
	Learning rate	$\{1e-5, 1e-6\}$
	μ	$1e-3$
	k	10
	r	64
StructZO-R	Batch size	16
	Learning rate	$\{1e-2, 1e-3\}$
	μ	$1e-3$
	k	10
	r	64
FO-Adam (OPT-1.3B)	Batch size	4,8
	Learning rate	$\{1e-4, 5e-5, 1e-5\}$
FO-Adam (Llama3.1-8B)	Batch size	1,2
	Learning rate	$\{1e-4, 5e-5, 1e-5\}$

Datasets. We evaluate on classification, multiple-choice and generative tasks drawn from GLUE and SuperGLUE [28, 29]. The individual tasks are SST-2 [30], RTE [31], CB [32], BoolQ [33], WiC [34], MultiRC [35], COPA [36], ReCoRD [37] and DROP [38]. For each task, we randomly select 1024 training samples and 512 validation samples. For small datasets with very limited examples (CB and COPA) we reserve 100 examples for validation and use the remainder

Table 6: Peak GPU memory usage (GiB) on MultiRC with batch size 1 and an average sequence length of approximately 400 tokens. HiZOO-L denotes the low-rank version of HiZOO. FO-Adam results for OPT-13B and OPT-30B are taken from MeZO [9].

Model	MeZO	LOZO	HiZOO-L	StructZO	FO-Adam
OPT1.3B	4.13	4.08	4.61	4.29	27.14
Llama3.1-8B	20.55	19.59	23.35	20.8	85.29
OPT13B	27.27	26.91	29.49	28.33	316
OPT30B	60.98	60.08	64.64	63.67	633

for training. We also adopt the same prompt design approach as MeZO [9], following GPT-3 and PromptSource with minor modifications. All experiments are conducted on a single NVIDIA H20 96GiB GPU.

Hyperparameters. Unless otherwise noted, each Zeroth-Order (ZO) algorithm is trained for 20K optimization steps and FO-Adam is trained for 5 epochs. For tokenization, we set a uniform maximum length $\text{MAX_LEN} = 2048$ which can cover the vast majority of language sequences across all tasks. For ZO methods we use a batch size of 16, and for FO-Adam we use the largest batch size that fits the GPU under identical system settings. To ensure a fair comparison, hyperparameters such as learning rate η is selected by a small grid search and perturbation length μ is set $\mu = 1e - 3$ as default for each Zeroth-Order algorithm. Specifically, for HiZOO and HiZOO-L, we sweep Hessian smoothing coefficients and decay factors, and for LOZO we sweep the low-rank dimension r and perturbation interval. For StructZO, the default oversampling factor used is $k = 10$, and the low-rank subspace rank is $r = 64$. Final hyperparameter settings are listed in Table 5.

Detailed memory comparison. Table 6 reports the detailed peak GPU memory usage on MultiRC with batch size 1 and an average sequence length of about 400 tokens. StructZO maintains memory usage close to inference-level ZO baselines across all tested model scales. Compared with MeZO and LOZO, StructZO introduces only a small additional memory cost, e.g., 4.29 GiB versus 4.13 GiB on OPT-1.3B, 20.80 GiB versus 20.55 GiB on Llama3.1-8B, and 63.67 GiB versus 60.98 GiB on OPT-30B. This modest overhead mainly comes from storing the activation-induced basis P , while the query perturbations are regenerated through the resampling trick rather than stored explicitly. In contrast, FO-Adam requires substantially larger memory due to backpropagation activations and optimizer states. StructZO reduces peak memory from 27.14 GiB to 4.29 GiB on OPT-1.3B, from 85.29 GiB to 20.80 GiB on Llama3.1-8B, from 316 GiB to 28.33 GiB on OPT-13B, and from 633 GiB to 63.67 GiB on OPT-30B. These results show that StructZO preserves the memory advantage of forward-only ZO optimization while scaling to substantially larger models than standard FO fine-tuning under the same memory budget.

Wall-clock time. Table 7 provides a detailed wall-clock comparison on OPT-1.3B and Llama3.1-8B across SST-2, RTE, and BoolQ. For single-query ZO baselines, we measure the average time over the first 100 optimization steps and estimate the total training time using 20K steps. For StructZO, we measure the average time over the first 10 optimization steps and estimate the total training time using 400 steps, following our main experimental setting with $q = 100$. As expected, StructZO has a substantially larger per-step cost because each update performs large-query estimation and constructs structured perturbations in the activation-induced subspace. However, this cost should be interpreted together with the number of optimization steps: each StructZO step aggregates much richer directional information, allowing the method to use $50\times$ fewer update steps than the single-query baselines. As a result, StructZO achieves comparable or lower end-to-end training time than MeZO and LOZO in most settings, and is consistently much faster than HiZOO/HiZOO-L. For example, on Llama3.1-8B, StructZO reduces the estimated total time from 3.656h to 2.954h on SST-2, from 9.79h to 9.17h on RTE, and from 14.99h to 14.76h on BoolQ compared with MeZO. These results show that although StructZO trades higher per-step latency for more informative updates, the resulting reduction in required optimization steps leads to favorable overall wall-clock efficiency.

Time breakdown of StructZO. To further understand where the computational cost of StructZO comes from, Table 8 reports the per-step wall-clock time breakdown into subspace construction, ZO forward evaluation, and ZO update. Across both OPT-1.3B and Llama3.1-8B, the dominant cost consistently comes from the ZO forward evaluations, which account for 83.91%–93.32% of the total step time. In contrast, constructing the activation-induced basis P only contributes about 4.95%–6.37% of the step time, showing that the randomized subspace extraction introduces a

Table 7: Detailed wall-clock comparison on OPT-1.3B and Llama3.1-8B. For MeZO, LOZO, and HiZOO/HiZOO-L, we report the average time per optimization step measured over the first 100 steps and estimate the total training time using 20K steps. For StructZO, we report the average time per optimization step measured over the first 10 steps and estimate the total training time using 400 steps, matching the main experimental setting with $q = 100$.

Model	Task	Method	Time / step (s)	Training steps	Total time (h)
OPT1.3B	SST-2	MeZO	0.146	20K	0.811
		LOZO	0.145	20K	0.804
		HiZOO	0.290	20K	1.613
		StructZO	6.890	400	0.766
	RTE	MeZO	0.358	20K	1.99
		LOZO	0.360	20K	2.00
		HiZOO	0.717	20K	3.98
		StructZO	17.59	400	1.96
	BoolQ	MeZO	0.558	20K	3.102
		LOZO	0.554	20K	3.080
		HiZOO	1.112	20K	6.178
		StructZO	28.13	400	3.125
Llama3.1-8B	SST-2	MeZO	0.658	20K	3.656
		LOZO	0.658	20K	3.654
		HiZOO-L	1.435	20K	7.970
		StructZO	26.58	400	2.954
	RTE	MeZO	1.762	20K	9.79
		LOZO	1.762	20K	9.79
		HiZOO-L	3.653	20K	20.30
		StructZO	82.53	400	9.17
	BoolQ	MeZO	2.698	20K	14.99
		LOZO	2.698	20K	14.99
		HiZOO-L	5.507	20K	30.60
		StructZO	132.87	400	14.76

relatively small overhead. The final ZO update, including reduced-space aggregation and spectral shaping, takes an even smaller fraction, especially on Llama3.1-8B where it is below 2.3% for all three tasks. These results indicate that the main computational cost of StructZO is not the structured subspace construction or the spectral update itself, but rather the large-query forward evaluations. This also suggests that the proposed structure adds only modest algorithmic overhead beyond the necessary forward-query cost of multi-query ZO estimation.

Effect of query number. Table 10 reports the downstream performance of StructZO under different query numbers q . The query number controls how many directional evaluations are aggregated within the activation-induced reduced space at each optimization step. When q is small, the ZO estimator remains relatively noisy even though the search space is structured, leading to weaker performance. For example, on OPT-1.3B BoolQ, increasing q from 4 to 100 improves accuracy from 63.5 to 68.5; on Llama3.1-8B CB, the improvement is even more pronounced, from 64.3 to 92.9. This trend is consistent with the training-loss curves in Figure 3, where larger query budgets lead to faster and deeper loss reduction under the same forward-pass view. However, increasing q beyond a certain point does not necessarily further improve downstream performance under a fixed computational budget. Overall, $q = 100$ provides a strong balance between estimator quality, update frequency, and end-task performance, and we therefore use it as the default setting.

Table 8: Component-wise wall-clock time breakdown of StructZO. The ZO forward evaluations dominate the per-step cost, while constructing the activation-induced subspace P and applying the final ZO update introduce only modest overhead.

Model	Task	Time / Subspace construction P (%)	Time / ZO forward (%)	Time / ZO update (%)
OPT1.3B	SST-2	5.43	83.91	10.66
	RTE	6.37	89.55	4.08
	BoolQ	6.25	91.15	2.60
Llama3.1-8B	SST-2	4.95	92.77	2.28
	RTE	6.03	93.26	0.71
	BoolQ	6.20	93.32	0.48

Table 9: Performance of the reduced-space rank r in StructZO.

Model	Task	$r = 16$	$r = 32$	$r = 64$	$r = 128$	$r = 256$
OPT1.3B	RTE	63.2	67.8	69.0	66.1	68.9
Llama3.1-8B	SST-2	94.0	93.8	93.7	93.8	92.8

Effect of subspace rank. Table 9 studies the effect of the reduced-space rank r in StructZO. The rank r is an important hyperparameter because it directly controls the dimension and expressiveness of the activation-induced search space. When r is too small, the reduced space may fail to preserve enough optimization-relevant directions, leading to a clear performance drop. For example, on OPT-1.3B RTE, increasing r from 16 to 64 improves accuracy from 63.2 to 69.0, showing that a sufficiently expressive subspace is important for StructZO. However, once the rank is large enough, further increasing r does not necessarily improve performance: using $r = 128$ or $r = 256$ gives comparable or slightly worse results on OPT-1.3B RTE, and the performance on Llama3.1-8B SST-2 remains relatively stable for $r \in \{16, 32, 64, 128\}$ with a mild drop at $r = 256$. This suggests that r plays a meaningful role in balancing subspace expressiveness and ZO estimation efficiency. The default choice $r = 64$ provides a strong trade-off: it is large enough to capture useful activation-induced directions, while avoiding the unnecessary increase in effective ZO dimension caused by overly large ranks.

Table 10: Performance of StructZO with different query numbers q on OPT-1.3B BoolQ and Llama3.1-8B CB. All results use the same StructZO pipeline while varying the number of ZO queries aggregated at each optimization step.

Model	Task	$q = 4$	$q = 10$	$q = 20$	$q = 50$	$q = 100$	$q = 200$
OPT1.3B	BoolQ	63.5	62.4	65.3	66.9	68.5	66.1
Llama3.1-8B	CB	64.3	66.1	69.7	80.4	92.9	78.6