Leveraging Coordinate Momentum in SignSGD and Muon: Memory-Optimized Zero-Order LLM Fine-Tuning

Egor Petrov [*1] **Grigoriy Evseev** [*1] **Aleksey Antonov** [*1 2] **Andrey Veprikov** [1 3] **Pavel Plyusnin** [2]
Nikolay Bushkov [1 2] **Stanislav Moiseev** [2] **Aleksandr Beznosikov** [1 3 4]

Abstract

Fine-tuning Large Language Models (LLMs) is essential for adapting pre-trained models to downstream tasks. Yet traditional first-order optimizers such as Stochastic Gradient Descent (SGD) and Adam incur prohibitive memory and computational costs that scale poorly with model size. In this paper, we investigate zero-order (ZO) optimization methods as a memory- and compute-efficient alternative, particularly in the context of parameter-efficient fine-tuning techniques like LoRA. We propose `JAGUAR SignSGD`, a ZO momentum-based algorithm that extends ZO SignSGD, requiring the same number of parameters as the standard ZO SGD and only $\mathcal{O}(1)$ function evaluations per iteration. To the best of our knowledge, this is the first study to establish rigorous convergence guarantees for SignSGD in the stochastic ZO case. We further propose `JAGUAR Muon`, a novel ZO extension of the Muon optimizer that leverages the matrix structure of model parameters, and we provide its convergence rate under arbitrary stochastic noise. Through extensive experiments on challenging LLM fine-tuning benchmarks, we demonstrate that the proposed algorithms meet or exceed the convergence quality of standard first-order methods, achieving significant memory reduction. Our theoretical and empirical results establish new ZO optimization methods as a practical and theoretically grounded approach for resource-constrained LLM adaptation. Our code is available at `https://github.com/brain-mmo-lab/ZO_LLM`

[*]Equal contribution [1]Moscow Institute of Physics and Technology [2]T-Technologies [3]Institute for System Programming, RAS [4]Innopolis University. Correspondence to: Egor Petrov <petrov.egor.d@phystech.edu>.

Proceedings of the 42^{nd} International Conference on Machine Learning, Vancouver, Canada. PMLR 267, 2025.

1. Introduction

Fine-tuning pre-trained Large Language Models (LLMs) has become the standard technique in modern natural language processing (Howard & Ruder, 2018; Zhang et al., 2019; 2024a; Lester et al., 2021), enabling rapid adaptation to diverse downstream tasks with minimal labelled data (Raffel et al., 2020; Sanh et al., 2021; Zaken et al., 2021). These models, often trained on massive corpora, achieve state-of-the-art results when fine-tuned on specific applications, including question answering, summarization, and dialogue generation. The fine-tuning setup can be considered as a stochastic unconstrained optimization problem of the form

$$f^* := \min_{x \in \mathbb{R}^d} \left\{ f(x) := \mathbb{E}_{\xi \sim \mathcal{D}} \left[f(x, \xi) \right] \right\}, \tag{1}$$

where x are parameters of the fine-tuned LLM, $\mathcal{D}$ is the data distribution available for training, and $f(x, \xi)$ is the loss on data point ξ.

The de facto standard for solving (1) is the use of First-Order (FO) optimization methods. These approaches assume access to the stochastic gradient $\nabla f(x, \xi)$. Classical FO methods, such as Stochastic Gradient Descent (SGD) (Amari, 1993) and Adam (Kingma & Ba, 2014), remain the most widely used techniques for model adaptation due to their efficiency and compatibility with the backpropagation algorithm. Nevertheless, in contemporary fine-tuning tasks, alternative FO algorithms are often preferred.

A recent trend in optimization for LLMs is to represent optimization parameters in matrix form rather than as vectors (Bernstein & Newhouse, 2024b;a; Pethick et al., 2025). Algorithms such as Shampoo (Gupta et al., 2018) and SOAP (Vyas et al., 2024) have demonstrated superior performance on LLM training tasks compared to Adam and SGD (Dahl et al., 2023), which operate in an element-wise manner and do not utilize the underlying structure of the model parameters. Currently, the canonical matrix-based optimization algorithm is Muon (Jordan et al., 2024; Liu et al., 2025; Li & Hong, 2025), which integrates the principles of Shampoo and SOAP but does not employ any preconditioning matrices (Jordan et al., 2024). The central idea of this method is to project the gradient at each iteration onto the space of

semi-orthogonal matrices using the Newton–Schultz algorithm (Bernstein & Newhouse, 2024b).

However, as LLMs continue to scale, the backpropagation procedure, necessary for FO methods, becomes increasingly expensive in terms of memory consumption. For instance, the memory cost of computing gradients during the training of OPT-13B is reported to be more than an order of magnitude larger than that of inference (Zhu et al., 2023b). This imbalance poses a serious bottleneck for deploying LLM fine-tuning in resource-constrained environments such as edge devices (Zhu et al., 2023a; Gao et al., 2024), consumer-grade GPUs (Liao et al., 2024; Yin et al., 2023), or large-scale distributed settings (Han et al., 2015). To overcome these limitations, researchers are exploring various approaches to reduce the size of the required optimizer statistics. One such approach is the SignSGD algorithm, initially developed for distributed optimization (Yang et al., 2020), but which has also proven effective in LLM fine-tuning (Peng et al.), owing to its simplicity, memory efficiency, and surprising empirical effectiveness across a range of adaptation tasks (Jin et al., 2020; Mengoli et al., 2025). SignSGD was first rigorously analyzed in the FO setting by (Bernstein et al., 2018) and (Balles & Hennig, 2017). Minimal memory usage and straightforward hyperparameter tuning make SignSGD an attractive choice for memory-constrained fine-tuning of LLMs ($\sim 4/3\times$ memory usage compared to Adam). Beyond SignSGD, other FO methods also target memory reduction. AdaFactor (Shazeer & Stern, 2018) was among the first, lowering memory usage by storing a single value per block ($\sim 4/3\times$). Additional techniques include quantizing optimizer states to lower-precision formats (Dettmers et al., 2021; Li et al., 2023) ($\sim 4/3\times$ and $\sim 16/9\times$ respectively) and fusing the backward pass with optimizer updates (Lv et al., 2023) ($\sim 4/3\times$), further decreasing memory demands during training.

Nevertheless, the most memory-efficient methods are based on the Zero-Order (ZO) optimization technique, which avoids backpropagation entirely by estimating gradients using only forward passes. This flexibility allows us to treat the model as a black box, optimizing performance with minimal assumptions about its architecture or implementation details. Recent studies (Malladi et al., 2023a) have demonstrated the practical benefits of this approach: for example, the MeZO algorithm applies classical ZO SGD (Ghadimi & Lan, 2013) to fine-tune LLMs while maintaining four times lower memory requirements than traditional FO methods (Malladi et al., 2023b) ($\sim 10\times$ compared to Adam (Zhang et al., 2024b)). In ZO methods it is assumed that we only have access to the values of the stochastic function $f(x, \xi)$ from (1) (Flaxman et al., 2005; Ghadimi & Lan, 2013). Within LLMs pretraining or fine-tuning context, oracles are forward passes with small perturbations in parameters of the model. To estimate gradients, authors use finite differences:

$$\nabla f(x, \xi) \approx \frac{f(x + \tau e, \xi) - f(x - \tau e, \xi)}{2\tau} e, \qquad (2)$$

where $\tau > 0$ is a small number, frequently referred to as a smoothing parameter, and $e \in \mathbb{R}^d$ is some random vector (Nesterov & Spokoiny, 2017; Duchi et al., 2015; Malladi et al., 2023b; Zhang et al., 2024b). In the next section, we provide review about different ZO optimization methods, that somehow utilize formula (2).

2. Related Work and Our Contributions

ZO gradient estimators. The simplest zero-order gradient estimator employs the estimate (2) as the stochastic gradient. However, even this approach presents specific challenges, particularly regarding the selection of an appropriate distribution from which to sample the random vector e. The most commonly employed distributions include a uniform sampling over the unit sphere: $e \sim RS(1)^d_{\|\cdot\|}$ (Flaxman et al., 2005; Nesterov & Spokoiny, 2017), a Gaussian distribution with zero mean and identity covariance matrix: $e \sim \mathcal{N}(0, I)$ (Nesterov & Spokoiny, 2017; Ghadimi & Lan, 2013), and standard basis one-hot vectors (Duchi et al., 2015; Shamir, 2013). Also, some papers (Lian et al., 2016; Sahu et al., 2019; Akhtar & Rajawat, 2022) utilize the so-called full coordinate estimate, which approximates the gradient across all basis vectors. However, this approach requires $\mathcal{O}(d)$ calls to the zero-order oracle, making it impractical for large-scale fine-tuning tasks. Despite the prevalence of these approaches, alternative and more complicated sampling strategies have also been explored.

In (Roberts & Royer, 2023; Nozawa et al., 2025), the authors explore low-dimensional perturbations within random subspaces. The central concept of random subspace methods involves generating the perturbation vector e within a subspace spanned by a projection matrix $P \in \mathbb{R}^{d\times r}$ and a low-dimensional random vector $\tilde{e} \in \mathbb{R}^r$: $e = P\tilde{e}$. Typically, P and $\tilde{e}$ are sampled from a Gaussian distribution and $r \ll d$. The primary motivation for this method lies in the fact that gradients during the fine-tuning process exhibit a low-dimensional structure (Nozawa et al., 2025). In (Liu et al., 2024; Wang et al., 2024), the authors employ a masked random vector e, wherein at each iteration a random mask with r non-zero elements $m_r \in \{0, 1\}^d$ is generated and applied element-wise to a Gaussian vector e. This procedure accelerates the optimization step, as only the parameters corresponding to the active entries in m_r are updated, rather than the entire parameter set. In contrast, the authors of (Guo et al., 2024b) depart from random mask sampling at each iteration and instead select an optimal mask m_r prior to training, according to a specific criterion. Consequently, the update rule (2) modifies only the parameters selected

by the optimal mask during optimization. In our approach, we similarly do not utilize all coordinates of the random vector e in each estimation of (2), instead, we select a single coordinate at each step. However, unlike previous works (Liu et al., 2024; Wang et al., 2024; Guo et al., 2024b), we do not discard information from the remaining coordinates, but accumulate information from previous iterations. We employ the JAGUAR zero-order gradient estimation technique (Veprikov et al., 2024; Nazykov et al., 2024), which integrates the concept of sampling one-hot basis vectors with the utilization of a SAGA-like momentum update (Defazio et al., 2014). This approach facilitates convergence in the stochastic setting by leveraging memory from past iterations, while using the same amount of memory as standard zero-order methods like ZO SGD (MeZO) (Malladi et al., 2023b). In the original paper (Veprikov et al., 2024), the authors do not incorporate a momentum parameter, discarding coordinate information from previous iterations. In contrast, we introduce a momentum parameter, $0 \leq \beta \leq 1$ (see Algorithms 1 and 2), which controls the utilization of gradients from past iterations. We demonstrate that adding this momentum β allows the method to converge in the stochastic non-convex case (see Theorems 3.5 and A.1).

Momentum techniques. Numerous zero-order methods in the literature incorporate momentum techniques in various forms. However, these approaches typically introduce multiple additional variables of dimension d. Since zero-order methods are often chosen for fine-tuning tasks to save memory, the inclusion of such extra variables becomes a critical limitation in these settings. In (Huang et al., 2022), authors use variance reduction technique SPIDER (Fang et al., 2018), that uses approximately $5d$ parameters: $2d$ for ZO gradients, $2d$ for model parameters and $1d$ for momentum. In (Chen et al., 2019; Jiang et al., 2024), the authors employ the Adam optimization technique (Kingma & Ba, 2014), which is frequently used for stochastic non-convex optimization problems (Chen et al., 2019; et al., 2024). However, this technique incurs a significant memory overhead, requiring $4d$ parameters. The paper (Reddy & Vidyasagar, 2023) utilizes classical heavy-ball momentum within a zero-order framework, provided, only demonstrating almost sure convergence to a constant in the non-convex setting. In our work, we successfully incorporated a momentum technique using only $2d + 1$ parameters and proved the convergence rate within the standard stochastic non-convex setting (see Algorithm 1 and Theorem 3.5). It is worth noting that numerous other zero-order techniques exist in the literature to achieve convergence when the function f is convex (Gorbunov et al., 2022; Nesterov & Spokoiny, 2017; Duchi et al., 2015), satisfies conditions like PL (Reddy & Vidyasagar, 2023) or ABG (Rando et al., 2024), or in deterministic settings (Gorbunov et al., 2022). Since our focus is on fine-tuning problems, which fall under the stochastic non-convex case, we will not discuss these methods in detail.

Matrix ZO optimization. In the context of zero-order optimization, transitioning to matrix-valued parameters necessitates replacing the random vector $e \in \mathbb{R}^d$ in zero-order gradient approximation (2) with a random matrix $E \in \mathbb{R}^{m \times n}$, and correspondingly, projecting this matrix E onto a semi-orthogonal space, as is done in the Muon algorithm (Jordan et al., 2024). Since the random matrix E is typically drawn from a known distribution, it is possible to directly sample orthogonal matrices when computing the gradient estimator (2). A similar approach has previously appeared in the zero-order optimization literature (Chen et al., 2024); however, that work did not consider the Muon algorithm, but rather focused on sampling two Gaussian matrices $V \in \mathbb{R}^{m \times r}$ and $U \in \mathbb{R}^{n \times r}$ of rank $r \ll \min\{m, n\}$. This approach does not correspond to the decomposition of the random matrix E, as E is almost surely of full rank. Additionally, alternative techniques for sampling low-rank matrices have been proposed in the literature. For instance, in (Yu et al., 2024), a method analogous to the sampling of low-rank vectors described in (Roberts & Royer, 2023; Nozawa et al., 2025) is utilized. In our work, we extend our memory-efficient momentum method to the ZO version of the matrix-based Muon algorithm (Jordan et al., 2024) (see Algorithm 2 and Theorem A.1), keeping the $2d+1$ parameter efficiency while also broadening our analysis to more modern algorithms that leverage the matrix structure of parameters.

We present a summary of relevant results from the existing zero-order literature in Table 1.

Table 1: Summary of relevant results from the existing zero-order literature.

	Method	Parameter Count	Convergence Rate Stochastic Non-convex Case	Momentum	Fine-tuning (LLM) Setup
Vector Parameters $x \in \mathbb{R}^d$	ZO-SGD (Ghadimi & Lan, 2013)	$2 \cdot \mathbf{d}$	✓	✗	✗
	ZO-PSGD (Ghadimi et al., 2016)	$2 \cdot \mathbf{d}$	✓	✗	✗
	ZO-SCD (Lian et al., 2016) [1]	$2 \cdot \mathbf{d}$	✓	✗	✗[2]
	ZO-SPIDER (Fang et al., 2018)	$5 \cdot \mathbf{d}$	✓	✓	✗
	ZO-AdaMM (Chen et al., 2019)	$4 \cdot \mathbf{d}$	✓	✓	✗
	ZO-SignSGD (Liu et al., 2019a)	$2 \cdot \mathbf{d}$	✗✓[3]	✗	✗[4]
	Acc-ZOM (Huang et al., 2022)	$5 \cdot \mathbf{d}$	✓	✓	✗
	DSFBSD (Roberts & Royer, 2023)	$(1+\mathbf{r}) \cdot \mathbf{d}$ [5]	✗	✗	✗
	MeZO (Malladi et al., 2023b)	$2 \cdot \mathbf{d}$	✗	✗	✓
	ZO-ProxSTORM (Qian & Zhao, 2023)	$5 \cdot \mathbf{d}$	✓	✓	✗
	HB ZO-SGD (Reddy & Vidyasagar, 2023)	$3 \cdot \mathbf{d}$	✗[6]	✓	✗
	Sparse ZO-SGD (Guo et al., 2024a)	$(2+\mathbf{r}) \cdot \mathbf{d}$ [5]	✗	✗	✓
	Sparse MeZO (Liu et al., 2024)	$3 \cdot \mathbf{d}$	✗	✗	✓
	LeZO (Wang et al., 2024)	$2 \cdot \mathbf{d}$	✗	✗	✓
	ZO-AdaMU (Jiang et al., 2024)	$4 \cdot \mathbf{d}$	✓	✓	✓
	ZO-SGD-Cons (Kim et al., 2025)	$2 \cdot \mathbf{d}$	✗	✗	✓
	SGFM (Nozawa et al., 2025)	$(2+\mathbf{r}) \cdot \mathbf{d}$ [5]	✗	✗	✗
	CompSGD (Kornilov et al., 2025)	$2 \cdot \mathbf{d}$	✗✓[3]	✗	✓
	JAGUAR SignSGD **Algorithm 1**	$2 \cdot \mathbf{d} + 1$	✓	✓	✓
Matrix Parameters $X \in \mathbb{R}^{m \times n}$	ZO-RMS (Maass et al., 2021) [7]	$2 \cdot \mathbf{mn}$	✗✓[8]	✗	✗
	MeZO (Malladi et al., 2023b)	$2 \cdot \mathbf{mn}$	✗	✗	✓
	LOZO (Chen et al., 2024)	$(\mathbf{m}+\mathbf{n})\mathbf{r} + 2 \cdot \mathbf{mn}$ [5]	✓	✗	✓
	SubZero (Yu et al., 2024) [8]	$(\mathbf{m}+\mathbf{n}+\mathbf{r})\mathbf{r} + 2 \cdot \mathbf{mn}$ [5]	✗	✗	✓
	JAGUAR Muon **Algorithm 2**	$2 \cdot \mathbf{mn} + 1$	✓	✓	✓

[1] Uses a full coordinate ZO estimator. [2] Considers asynchronous algorithms. [3] Convergence only to a neighborhood of the solution. [4] Addresses adversarial attacks in deep learning. [5] $r \ll d, m, n$ is a small number. [6] Only asymptotic convergence to a constant. [7] Assumes that parameters are symmetric matrices. [8] Assumes sparsity of parameters.

2.1. Our Contributions

While zero-order optimization methods have recently attracted attention for LLM fine-tuning, previous work has primarily focused on basic algorithms. In this paper, we

broaden the scope of zero-order optimization by introducing advanced momentum techniques, specifically adapting the JAGUAR approach (Veprikov et al., 2024) to the SignSGD algorithm in the zero-order setting (see Algorithms 1). We consider this algorithm because SignSGD has demonstrated state-of-the-art performance in LLM fine-tuning tasks, outperforming even AdamW (Peng et al.). Our key contributions are as follows:

- We provide the first convergence analysis in the stochastic non-convex setting for zero-order SignSGD with momentum (Algorithm 1 and Theorem 3.5), requiring only $2d+1$ parameters and $\mathcal{O}(1)$ ZO oracle calls per iteration.
- We extend our memory-efficient momentum method to the Muon algorithm (Algorithm 2 in Appendix A), introducing the first zero-order variant of Muon that preserves memory efficiency. We also establish its convergence rate in the stochastic non-convex setting (Theorem A.1).
- We empirically evaluate the proposed zero-order methods on challenging LLM fine-tuning benchmarks, demonstrating their effectiveness and practical relevance.

3. Main results

3.1. Preliminaries

Notations. We denote the ℓ_1 and ℓ_2 (Euclidean) norms of a vector $x \in \mathbb{R}^d$ as $\|x\|_1 := \sum_{i=1}^d |x_i|$ and $\|x\|_2^2 := \sum_{i=1}^d x_i^2$, respectively. For clarity, matrix-valued variables are denoted by capital letters. For matrices $X \in \mathbb{R}^{m\times n}$, we use the Schatten 1-norm ($\mathcal{S}_1$) and Schatten 2-norm ($\mathcal{S}_2$, Frobenius): $\|X\|_{\mathcal{S}_1} := \sum_{i=1}^d |(\Sigma_X)_{i,i}|$ and $\|X\|_{\mathcal{S}_2}^2 := \sum_{i=1}^d (\Sigma_X)_{i,i}^2 = \sum_{i=1}^m \sum_{j=1}^n X_{i,j}^2 =: \|X\|_F^2$, where $X = U_X \Sigma_X V_X^T$ is the reduced Singular Value Decomposition (SVD) of X. The standard dot product between two vectors $x, y \in \mathbb{R}^d$ is defined as $\langle x, y\rangle := x^T y$. For matrices $X, Y \in \mathbb{R}^{m\times n}$, we define the inner product as $\langle X, Y\rangle := \mathrm{tr}(X^T Y)$.

We now provide several assumptions that are necessary for the analysis.

Assumption 3.1 (Smoothness)**.** The functions $f(x,\xi)$ are $L(\xi)$-smooth on the $\mathbb{R}^d$ with respect to the Euclidean norm $\|\cdot\|$, i.e., for all $x, y \in \mathbb{R}^d$ it holds that $\|\nabla f(x,\xi) - \nabla f(y,\xi)\|_2 \le L(\xi)\|x-y\|_2$. We also assume that exists constant $L^2 := \mathbb{E}\left[L(\xi)^2\right]$.

Assumption 3.2 (Bounded variance of the gradient)**.** The variance of the $\nabla f(x,\xi)$ is bounded with respect to the Euclidean norm, i.e., there exists $\sigma > 0$, such that for all $x \in \mathbb{R}^d$ it holds that $\mathbb{E}\left[\|\nabla f(x,\xi) - \nabla f(x)\|_2^2\right] \le \sigma^2$.

We assume access only to a zero-order oracle, which returns a noisy evaluation of the function $f(x,\xi)$. Therefore, we are limited to using this noisy value $\hat{f}(x,\xi)$ in the estimation of the ZO gradient (2). This noise may originate not only from inherent randomness (stochastic noise), but also from systematic effects (deterministic noise), such as computer rounding errors. Therefore, we make a common assumption about the function $\hat{f}(x,\xi)$ returned by the oracle (Dvurechensky et al., 2021; Veprikov et al., 2024).

Assumption 3.3 (Bounded oracle noise)**.** The noise in the oracle is bounded with respect to the Euclidean norm, i.e., there exists $\Delta > 0$, such that for all $x \in \mathbb{R}^d$ it holds that $\mathbb{E}\left[\left|\hat{f}(x,\xi) - f(x,\xi)\right|^2\right] \le \Delta^2$.

Assumptions 3.1 and 3.2 are standard in the theoretical analysis of stochastic non-convex zero-order optimization problems (Reddy & Vidyasagar, 2023; Guo et al., 2024b; Liu et al., 2024; Wang et al., 2024). In contrast, Assumption 3.3 is frequently omitted in the existing literature, as it is commonly presumed that $\Delta = 0$, implying access to an ideal zero-order oracle. However, this assumption does not hold in practice, as numerical errors such as machine precision inevitably introduce a non-zero perturbation. Consequently, while Δ is typically small, it is never zero, which does not allow us to restore a true gradient along the direction e in the estimation (2) if we set $\tau \to 0$.

3.2. Zero-Order Momentum SignSGD with JAGUAR Gradient Approximation

In this section, we introduce zero-order SignSGD algorithm with JAGUAR gradient approximation (Veprikov et al., 2024; Nazykov et al., 2024) and momentum of the form:

Algorithm 1 Zero-Order Momentum SignSGD with JAGUAR (`JAGUAR SignSGD`)

Parameters: stepsize γ, momentum β, gradient approximation parameter τ, number of iterations T.
Initialization: choose $x^0 \in \mathbb{R}^d$ and $m^{-1} = \mathbf{0} \in \mathbb{R}^d$.
for $t = 0, 1, 2, \ldots, T$ **do**
  Sample $i_t \sim$ Uniform$(\overline{1,d})$
  Set one-hot vector e^t with 1 in the i_t coordinate
  Sample stochastic variable $\xi^t \sim \mathcal{D}$
  Compute $\widetilde{\nabla}_{i_t} f(X^t, \xi^t) := \frac{f_+ - f_-}{2\tau} \in \mathbb{R}$, where $f_+ = \hat{f}(X^t + \tau E^t, \xi^t)$, $f_- = \hat{f}(X^t - \tau E^t, \xi^t)$
  Set $m_{i_t}^t = \beta m_{i_t}^{t-1} + (1-\beta)\widetilde{\nabla}_{i_t} f(x^t, \xi^t)$ and $m_{i\neq i_t}^t = m_{i\neq i_t}^{t-1}$ for all $i \in \overline{1,d}$
  Set $x^{t+1} = x^t - \gamma \cdot \mathrm{sign}(m^t)$
end for
Return: $x^{N(T)}$, where $N(T) \sim$ Uniform$(\overline{1,T})$.

The gradient approximation employed in Algorithm 1 deviates from that of the original JAGUAR method, as we

introduce a momentum variable β. The estimator from the original work can be recovered by setting $\beta = 0$.

We now present a lemma characterizing the closeness between the momentum variable m^t from line 8 of Algorithm 1 and the true gradient $\nabla f(x^t)$.

Lemma 3.4. *Consider m^t from line 8 of Algorithm 1. Under Assumptions 3.1, 3.2, 3.3 it holds that:*

$$\mathbb{E}\left[\left\|m^t - \nabla f(x^t)\right\|_2^2\right] = \mathcal{O}\Bigg[\frac{d^3L^2\gamma^2}{(1-\beta)^2} + (1-\beta)d\sigma^2 + dL^2\tau^2 + \frac{2d\Delta^2}{\tau^2} + \left(\frac{1-\beta}{d}\right)^t \left\|\nabla f(x^0)\right\|_2^2\Bigg].$$

Discussion. This lemma closely parallels Lemma 1 from (Veprikov et al., 2024), with the key distinction that our analysis incorporates the momentum parameter β, which was not present in (Veprikov et al., 2024). The introduction of momentum is essential for proving convergence of algorithms such as SignSGD (Algorithm 1) and Muon (see Algorithm 2 in the next section) in the stochastic zero-order setting (Sun et al., 2023), as it enables more careful handling of variance σ in the gradient estimates (2). Another important difference from prior works is that result from Lemma 3.4 does not involve the term $\|\nabla f(x^t)\|_2^2$, which typically appears in analyses where the zero-order gradient estimator (2) is constructed using random uniform or Gaussian vectors e (Cai et al., 2021; Kozak et al., 2021; Gorbunov et al., 2022; Qian & Zhao, 2023). With the presence of the term $\|\nabla f(x^t)\|_2^2$, it is not possible to achieve convergence guarantees for SignSGD (Algorithm 1) and Muon (Algorithm 2 in Appendix A) even with momentum in the stochastic zero-order setting. It is worth noting that a similar result can be obtained when using a full coordinate estimator (Lian et al., 2016). However, this approach requires $\mathcal{O}(d)$ calls to the zero-order oracle per iteration, which can be computationally expensive. In contrast, the JAGUAR method achieves the same result with only $\mathcal{O}(1)$ oracle calls and with the same number of parameters, offering significant improvements in efficiency. This makes our approach particularly attractive for large-scale optimization tasks, where reducing oracle complexity is critical.

With the help of Lemma 3.4, we provide convergence analysis of `JAGUAR SignSGD` (Algorithm 1).

Theorem 3.5. *Consider Assumptions 3.1, 3.2 and 3.3. Then `JAGUAR SignSGD` (Algorithm 1) has the following convergence rate:*

$$\mathbb{E}\left[\left\|\nabla f\left(x^{N(T)}\right)\right\|_1\right] = \mathcal{O}\Bigg[\frac{\delta_0}{\gamma T} + \frac{d\left\|\nabla f(x^0)\right\|_2}{T\sqrt{1-\beta}} + \frac{d^2L\gamma}{1-\beta} + \sqrt{1-\beta}d\sigma + dL\tau + \frac{d\Delta}{\tau}\Bigg],$$

where we used a notation $\delta_0 := f(x^0) - f^*$.

Corollary 3.6. *Consider the conditions of Theorem 3.5. In order to achieve the ε-approximate solution (in terms of $\mathbb{E}\left[\left\|\nabla f(x^{N(T)})\right\|_1\right] \leq \varepsilon$), Algorithm 1 needs T iterations (ZO oracle calls), with:*

$$\gamma = \sqrt{\frac{\delta_0(1-\beta)}{d^2LT}}, \beta = 1 - \min\left\{1; \sqrt{\frac{L\delta_0}{T\sigma^2}}\right\},$$
$$\tau = (\Delta/L)^{1/2}, \varepsilon \geq d\sqrt{\Delta L}:$$
$$T = \mathcal{O}\left[\frac{\delta_0 L d^2}{\varepsilon^2} + \frac{\delta_0 L d^2}{\varepsilon^2}\cdot\left(\frac{d\sigma}{\varepsilon}\right)^2\right].$$

Discussion. The convergence rate established in Theorem 3.5 is similar to what is known for first-order methods (Bernstein et al., 2018; Jin et al., 2020; Safaryan & Richtárik, 2021; Kornilov et al., 2025), however our bounds include an additional factor of d, which is typical for all coordinate-based methods (Nesterov, 2012; Richtárik & Takáč, 2016), not just zero-order ones. This dependence on the dimension arises because coordinate methods process one direction at a time, accumulating complexity proportional to d. It is also important to note that without momentum ($\beta = 0$), the algorithm can only guarantee convergence to a neighbourhood of the optimum of size proportional to σ, as shown in previous works on zero-order SignSGD (Liu et al., 2019a; Kornilov et al., 2025). Let us also point out that we cannot choose an arbitrary ε in Corollary 3.6, since there exists an irreducible (Dvurechensky et al., 2021; Veprikov et al., 2024) error Δ in the zero-order oracle (see Assumption 3.3). However, since Δ is very small, we can still achieve an acceptable accuracy ε. In our analysis, we use the ℓ_1-norm of the gradient as the convergence criterion, while the standard in non-convex optimization is the ℓ_2-norm (Euclidean) (Ghadimi & Lan, 2013; 2016). By setting $\varepsilon_{\ell_1} = \sqrt{d}\cdot\varepsilon_{\ell_2}$, we can rescale our result of Corollary 3.6 as

$$T_{\text{Euclidean}} = \mathcal{O}\Big[\frac{\delta_0 L d}{\varepsilon^2} + \frac{\delta_0 L d}{\varepsilon^2}\cdot\Big(\frac{\sqrt{d}\sigma}{\varepsilon}\Big)^2\Big].$$

This substitution allows us to obtain improved results in terms of the dependence on d.

4. Experiments

In this section, we present a comprehensive empirical evaluation to validate the theoretical contributions of our proposed ZO optimization methods for fine-tuning large language models. Our study aims to assess both the accuracy and memory efficiency of these methods, comparing them against established ZO and FO baselines. We build upon the experimental framework proposed in (Zhang et al., 2024b), extending it to incorporate our novel algorithms: `JAGUAR SignSGD` (Algorithm 1) and `JAGUAR Muon` (Algorithm

2 in Appendix A). The primary objective is to achieve competitive test accuracy on downstream tasks while maintaining memory efficiency comparable to the baseline methods. Additionally, we introduce `ZO-Muon` (Algorithm 3 in Appendix B), a direct zero-order adaptation of Muon (Jordan et al., 2024), utilizing the standard Gaussian zero-order gradient estimation (2).

4.1. Experimental Setup

Fine-Tuning Task and Schemes. Fine-tuning LLMs is a pivotal process in adapting pre-trained models to downstream tasks, enabling high performance with limited task-specific data. To explore the efficacy of our ZO methods, we focus on the SST2 dataset (Socher et al., 2013), a widely-used benchmark for binary sentiment classification (Zhang et al., 2024b; Chen et al., 2024; Malladi et al., 2023b). Additionally, we measure performance of Llama2-7B (Touvron et al., 2023) and OPT-13B (Zhang et al., 2022) on WinoGrande (Sakaguchi et al., 2021) and COPA (Roemmele et al., 2011) datasets (see Appendix C). We consider two fine-tuning schemes:

- **Full Fine-Tuning (FT):** Updates all parameters of the pre-trained model.
- **Low-Rank Adaptation (LoRA):** Introduces a small set of trainable parameters while keeping the original model parameters frozen (Hu et al., 2021).

Models. We conduct experiments using four prominent LLMs: OPT-1.3B (Zhang et al., 2022), a 1.3 billion parameter model from the OPT family; RoBERTa-Large (Liu et al., 2019b), a 355 million parameter model known for its robust performance in natural language processing tasks; Llama 2 (Touvron et al., 2023) and OPT-13B (Zhang et al., 2022), state-of-the-art open-source models widely used for research and applications. These models represent a range of sizes and architectures, allowing us to assess the scalability and generality of our methods.

Methods. We evaluate the following ZO optimization methods proposed in this work:

- **`JAGUAR SignSGD`:** Combines the JAGUAR gradient approximation with SignSGD and momentum for efficient updates (Algorithm 1).
- **`JAGUAR Muon`:** Integrates JAGUAR with the Muon optimizer, incorporating momentum and orthogonalization (Algorithm 2 in Appendix A).
- **`ZO-Muon`:** A novel ZO adaptation of the Muon optimizer, leveraging matrix-based optimization principles (Algorithm 3 in Appendix B).

Comparison procedure. For comparison, we include baseline methods from (Zhang et al., 2024b): ZO-SGD (Ghadimi & Lan, 2013), Acc-ZOM (Huang et al., 2022), ZO-SGD-Cons (Kim et al., 2025), ZO-SignSGD (Liu et al., 2019a), ZO-AdaMM (Chen et al., 2019), Forward-Grad (Baydin et al., 2022), and the FO method FO-SGD (Amari, 1993). The results for which are given in the benchmark paper. Additionally, we compare our methods with LeZO (Wang et al., 2024), which employs a comparable layer-wise selection mechanism similar to `JAGUAR SignSGD` coordinate-wise updates. We perform experiments for our methods in accordance with similar experiments from (Zhang et al., 2024b). See other experiments with large-size models and technical details in Appendix C.

4.2. Results

Table 2: Test accuracy on SST2 for OPT-1.3B and RoBERTa-Large with FT and LoRA. Best performance among ZO methods is in **bold**. Blue indicates outperformance of all baseline ZO methods, red indicates matching or exceeding FO-SGD.

Method	OPT-1.3B		RoBERTa-Large	
	FT	LoRA	FT	LoRA
FO-SGD	91.1	93.6	91.4	91.2
Forward-Grad	90.3	90.3	90.1	89.7
ZO-SGD	90.8	90.1	89.4	90.8
Acc-ZOM	85.2	91.3	89.6	90.9
ZO-SGD-Cons	88.3	90.5	89.6	91.6
ZO-SignSGD	87.2	91.5	52.5	90.2
ZO-AdaMM	84.4	92.3	89.8	89.5
LeZO	85.1	92.3	90.4	91.8
`JAGUAR SignSGD`	**94.0 ± 0.1**	92.5 ± 0.5	**92.2 ± 0.2**	**92.2 ± 0.4**
`ZO-Muon`	86.5 ± 0.1	93.5 ± 0.1	72.0 ± 0.1	86.0 ± 0.2
`JAGUAR Muon`	84.0 ± 0.1	**94.0 ± 0.1**	85.0 ± 0.1	**92.2 ± 0.2**

Discussion. Based on the results presented in the Table 2, proposed methods (Algorithms 1 and 2) that leverage the JAGUAR approximation of gradient outperform comparable approaches utilizing standard random vector sampling e in equation (2) or vanilla momentum techniques originally designed for FO algorithms. `JAGUAR SignSGD` and `JAGUAR Muon` achieve superior performance, demonstrating the effectiveness and robustness of our method compared to existing baselines.

4.3. Memory Efficiency

Table 3: GPU allocated memory (GB) for OPT-1.3B (half-precision, F16) on SST2 with FT and LoRA.

Method	FT Memory	LoRA Memory
FO-SGD	12.246	5.855
ZO-SGD	4.171	4.125
ZO-AdaMM	13.046	6.132
`JAGUAR SignSGD`	4.172	4.128
`ZO-Muon`	4.177	4.130
`JAGUAR Muon`	4.179	4.132

Acknowledgements

The work on the final version was supported by the Ministry of Economic Development of the Russian Federation (code 25-139-66879-1-0003).

References

Akhtar, Z. and Rajawat, K. Zeroth and first order stochastic frank-wolfe algorithms for constrained optimization. *IEEE Transactions on Signal Processing*, 70:2119–2135, 2022.

Amari, S.-i. Backpropagation and stochastic gradient descent method. *Neurocomputing*, 5(4):185–196, 1993.

Balles, L. and Hennig, P. Dissecting adam: The sign, magnitude and variance of stochastic gradients. In *International Conference on Machine Learning (ICML)*, 2017.

Baydin, A. G., Pearlmutter, B. A., Syme, D., Wood, F., and Torr, P. Gradients without backpropagation, 2022. URL https://arxiv.org/abs/2202.08587.

Bernstein, J. and Newhouse, L. Modular duality in deep learning. *arXiv preprint arXiv:2410.21265*, 2024a.

Bernstein, J. and Newhouse, L. Old optimizer, new norm: An anthology. *arXiv preprint arXiv:2409.20325*, 2024b.

Bernstein, J., Zhao, J., Azizzadenesheli, K., and Anandkumar, A. signsgd with majority vote is communication efficient and fault tolerant. *arXiv preprint arXiv:1810.05291*, 2018.

Cai, H., Lou, Y., McKenzie, D., and Yin, W. A zeroth-order block coordinate descent algorithm for huge-scale black-box optimization. In *ICML*, pp. 1182–1191, 2021.

Chen, X., Liu, S., Xu, K., Li, X., Lin, X., Hong, M., and Cox, D. Zo-adamm: Zeroth-order adaptive momentum method for black-box optimization. *Advances in neural information processing systems*, 32, 2019.

Chen, Y., Zhang, Y., Cao, L., Yuan, K., and Wen, Z. Enhancing zeroth-order fine-tuning for language models with low-rank structures. *ArXiv*, abs/2410.07698, 2024.

Dahl, G. E., Schneider, F., Nado, Z., Agarwal, N., Sastry, C. S., Hennig, P., Medapati, S., Eschenhagen, R., Kasimbeg, P., Suo, D., et al. Benchmarking neural network training algorithms. *arXiv preprint arXiv:2306.07179*, 2023.

Defazio, A., Bach, F., and Lacoste-Julien, S. Saga: A fast incremental gradient method with support for non-strongly convex composite objectives. *Advances in neural information processing systems*, 27, 2014.

Dettmers, T., Lewis, M., Shleifer, S., and Zettlemoyer, L. 8-bit optimizers via block-wise quantization. *arXiv preprint arXiv:2110.02861*, 2021.

Duchi, J. C., Jordan, M. I., Wainwright, M. J., and Wibisono, A. Optimal rates for zero-order convex optimization: The power of two function evaluations. *IEEE Transactions on Information Theory*, 61(5):2788–2806, 2015. doi: 10.1109/TIT.2015.2413811.

Dvurechensky, P., Gorbunov, E., and Gasnikov, A. An accelerated directional derivative method for smooth stochastic convex optimization. *European Journal of Operational Research*, 290(2):601–621, 2021.

et al., A. A. Mezo-a^{3}dam: Memory-efficient zeroth-order adam with adaptivity adjustments. *OpenReview, ICLR 2025*, 2024. URL https://openreview.net/forum?id=OBIuFjZzmp. Under review.

Fang, C., Li, C. J., Lin, Z., and Zhang, T. Spider: Near-optimal non-convex optimization via stochastic path-integrated differential estimator. *Advances in neural information processing systems*, 31, 2018.

Flaxman, A. D., Kalai, A. T., and McMahan, H. B. Online convex optimization in the bandit setting: gradient descent without a gradient. In *Proceedings of the Sixteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pp. 385–394, 2005.

Gao, L., Ziashahabi, A., Niu, Y., Avestimehr, S., and Annavaram, M. Enabling efficient on-device fine-tuning of llms using only inference engines. *arXiv preprint arXiv:2409.15520*, 2024.

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

Ghadimi, S. and Lan, G. Accelerated gradient methods for nonconvex optimization. *arXiv preprint arXiv:1608.06860*, 2016.

Ghadimi, S., Lan, G., and Zhang, H. Mini-batch stochastic approximation methods for nonconvex stochastic composite optimization. *Mathematical Programming*, 155(1): 267–305, 2016.

Gorbunov, E., Dvurechensky, P., and Gasnikov, A. An accelerated method for derivative-free smooth stochastic convex optimization. *SIAM Journal on Optimization*, 32 (2):1210–1238, 2022.

Guo, W., Long, J., Zeng, Y., Liu, Z., Yang, X., Ran, Y., Gardner, J. R., Bastani, O., De Sa, C., Yu, X., et al. Zeroth-order fine-tuning of llms with extreme sparsity. *arXiv preprint arXiv:2406.02913*, 2024a.

Guo, W., Long, J., Zeng, Y., Liu, Z., Yang, X., Ran, Y., Gardner, J. R., Bastani, O., Sa, C. D., Yu, X., Chen, B., and Xu, Z. Zeroth-order fine-tuning of llms with extreme sparsity, 2024b. URL https://arxiv.org/abs/2406.02913.

Gupta, V., Koren, T., and Singer, Y. Shampoo: Preconditioned stochastic tensor optimization, 2018. URL https://arxiv.org/abs/1802.09568.

Han, S., Mao, H., and Dally, W. J. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint*, abs/1510.00149, 2015.

Howard, J. and Ruder, S. Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*, 2018.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. LoRA: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

Huang, F., Gao, S., Pei, J., and Huang, H. Accelerated zeroth-order and first-order momentum methods from mini to minimax optimization. *Journal of Machine Learning Research*, 23(36):1–70, 2022.

Jiang, S., Chen, Q., Pan, Y., Xiang, Y., Lin, Y., Wu, X., Liu, C., and Song, X. Zo-adamu optimizer: Adapting perturbation by the momentum and uncertainty in zeroth-order optimization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 18363–18371, 2024.

Jin, R., Huang, Y., He, X., Dai, H., and Wu, T. Stochastic-sign sgd for federated learning with theoretical guarantees. *arXiv preprint arXiv:2002.10940*, 2020.

Jordan, K., Jin, Y., Boza, V., You, J., Cesista, F., Newhouse, L., and Bernstein, J. Muon: An optimizer for hidden layers in neural networks, 2024. URL https://kellerjordan.github.io/posts/muon/.

Kim, B., McKenzie, D., Cai, H., and Yin, W. Curvature-aware derivative-free optimization. *Journal of Scientific Computing*, 103(2), March 2025. ISSN 1573-7691. doi: 10.1007/s10915-025-02855-8. URL http://dx.doi.org/10.1007/s10915-025-02855-8.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Kornilov, N., Zmushko, P., Semenov, A., Gasnikov, A., and Beznosikov, A. Sign operator for coping with heavy-tailed noise: High probability convergence bounds with extensions to distributed optimization and comparison oracle. *arXiv preprint arXiv:2502.07923*, 2025.

Kovalev, D. Understanding gradient orthogonalization for deep learning via non-euclidean trust-region optimization. *arXiv preprint arXiv:2503.12645*, 2025.

Kozak, D., Molinari, C., Rosasco, L., Tenorio, L., and Villa, S. Zeroth order optimization with orthogonal random directions. *arXiv preprint*, abs/2107.03941, 2021.

Lester, B., Al-Rfou, R., and Constant, N. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*, 2021.

Li, B., Chen, J., and Zhu, J. Memory efficient optimizers with 4-bit states. *Advances in Neural Information Processing Systems*, 36:15136–15171, 2023.

Li, J. and Hong, M. A note on the convergence of muon and further, 2025. URL https://arxiv.org/abs/2502.02900.

Lian, X., Zhang, H., Hsieh, C.-J., Huang, Y., and Liu, J. A comprehensive linear speedup analysis for asynchronous stochastic parallel optimization from zeroth-order to first-order. *Advances in neural information processing systems*, 29, 2016.

Liao, C., Sun, M., Yang, Z., Xie, J., Chen, K., Yuan, B., Wu, F., and Wang, Z. Lohan: Low-cost high-performance framework to fine-tune 100b model on a consumer gpu. *arXiv preprint arXiv:2403.06504*, 2024.

Liu, J., Su, J., Yao, X., Jiang, Z., Lai, G., Du, Y., Qin, Y., Xu, W., Lu, E., Yan, J., Chen, Y., Zheng, H., Liu, Y., Liu, S., Yin, B., He, W., Zhu, H., Wang, Y., Wang, J., Dong, M., Zhang, Z., Kang, Y., Zhang, H., Xu, X., Zhang, Y., Wu, Y., Zhou, X., and Yang, Z. Muon is scalable for llm training, 2025. URL https://arxiv.org/abs/2502.16982.

Liu, S., Chen, P.-Y., Chen, X., and Hong, M. signsgd via zeroth-order oracle. In *International conference on learning representations*, 2019a.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. Roberta: A robustly optimized bert pretraining approach, 2019b. URL https://arxiv.org/abs/1907.11692.

Liu, Y., Zhu, Z., Gong, C., Cheng, M., Hsieh, C.-J., and You, Y. Sparse mezo: Less parameters for better performance in zeroth-order llm fine-tuning. *arXiv preprint arXiv:2402.15751*, 2024.

Lv, K., Yang, Y., Liu, T., Gao, Q., Guo, Q., and Qiu, X. Full parameter fine-tuning for large language models with limited resources. *arXiv preprint arXiv:2306.09782*, 2023.

Maass, A. I., Manzie, C., Shames, I., and Nakada, H. Zeroth-order optimization on subsets of symmetric matrices with application to mpc tuning. *IEEE Transactions on Control Systems Technology*, 30(4):1654–1667, 2021.

Malladi, B., Aghazadeh, A., Tang, H., et al. Mezo: Memory-efficient zeroth-order optimization of large language models. *arXiv preprint*, abs/2305.18660, 2023a.

Malladi, S., Gao, T., Nichani, E., Damian, A., Lee, J. D., Chen, D., and Arora, S. Fine-tuning language models with just forward passes. *Advances in Neural Information Processing Systems*, 36:53038–53075, 2023b.

Mengoli, E., Moll, L., Strozzi, V., and El-Mhamdi, E.-M. On the byzantine fault tolerance of signsgd with majority vote. *arXiv preprint arXiv:2502.19170*, 2025.

Nazykov, R., Shestakov, A., Solodkin, V., Beznosikov, A., Gidel, G., and Gasnikov, A. Stochastic frank-wolfe: Unified analysis and zoo of special cases. In *International Conference on Artificial Intelligence and Statistics*, pp. 4870–4878. PMLR, 2024.

Nesterov, Y. Efficiency of coordinate descent methods on huge-scale optimization problems. *SIAM Journal on Optimization*, 22(2):341–362, 2012.

Nesterov, Y. and Spokoiny, V. Random gradient-free minimization of convex functions. *Foundations of Computational Mathematics*, 17(2):527–566, 2017. doi: 10.1007/s10208-015-9296-2.

Nozawa, R., Poirion, P.-L., and Takeda, A. Zeroth-order random subspace algorithm for non-smooth convex optimization. *Journal of Optimization Theory and Applications*, 204(3):53, 2025.

Peng, H., Qin, S., Jiang, F., Yu, Y., Wang, H., and Li, G. Softsignsgd (s3): An enhanced optimize for practical dnn training and loss spikes minimization beyond adam.

Pethick, T., Xie, W., Antonakopoulos, K., Zhu, Z., Silveti-Falls, A., and Cevher, V. Training deep learning models with norm-constrained lmos. *arXiv preprint arXiv:2502.07529*, 2025.

Qian, Y. and Zhao, Y. Zeroth-order proximal stochastic recursive momentum algorithm for nonconvex nonsmooth optimization. In *2023 International Conference on New Trends in Computational Intelligence (NTCI)*, volume 1, pp. 419–423. IEEE, 2023.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21 (140):1–67, 2020.

Rando, M., Molinari, C., Villa, S., and Rosasco, L. Stochastic zeroth order descent with structured directions. *Computational Optimization and Applications*, pp. 1–37, 2024.

Reddy, T. U. K. and Vidyasagar, M. Convergence of momentum-based heavy ball method with batch updating and/or approximate gradients. In *2023 Ninth Indian Control Conference (ICC)*, pp. 182–187. IEEE, 2023.

Richtárik, P. and Takáč, M. Distributed coordinate descent method for learning with big data. *Journal of Machine Learning Research*, 17(75):1–25, 2016.

Roberts, L. and Royer, C. W. Direct search based on probabilistic descent in reduced spaces. *SIAM Journal on Optimization*, 33(4):3057–3082, 2023.

Roemmele, M., Bejan, C. A., and Gordon, A. S. Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In *AAAI spring symposium: logical formalizations of commonsense reasoning*, pp. 90–95, 2011.

Safaryan, M. and Richtárik, P. Stochastic sign descent methods: New algorithms and better theory. In *International Conference on Machine Learning*, pp. 9224–9234. PMLR, 2021.

Sahu, A. K., Zaheer, M., and Kar, S. Towards gradient free and projection free stochastic optimization. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pp. 3468–3477. PMLR, 2019.

Sakaguchi, K., Bras, R. L., Bhagavatula, C., and Choi, Y. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021.

Sanh, V., Webson, A., Raffel, C., Bach, S. H., Sutawika, L., Alyafeai, Z., Chaffin, A., Stiegler, A., Scao, T. L., Raja, A., et al. Multitask prompted training enables zero-shot task generalization. *arXiv preprint arXiv:2110.08207*, 2021.

Shamir, O. On the complexity of bandit and derivative-free stochastic convex optimization. In *ICML*, pp. 1001–1009, 2013.

Shazeer, N. and Stern, M. Adafactor: Adaptive learning rates with sublinear memory cost. In *International Conference on Machine Learning*, pp. 4596–4604. PMLR, 2018.

Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A. Y., and Potts, C. Recursive deep models for semantic compositionality over a sentiment treebank. *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 1631–1642, 2013.

Sun, T., Wang, Q., Li, D., and Wang, B. Momentum ensures convergence of signsgd under weaker assumptions. In *International Conference on Machine Learning*, pp. 33077–33099. PMLR, 2023.

Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Veprikov, A., Bogdanov, A., Minashkin, V., and Beznosikov, A. New aspects of black box conditional gradient: Variance reduction and one point feedback. *Chaos, Solitons & Fractals*, 189:115654, 2024.

Vyas, N., Morwani, D., Zhao, R., Kwun, M., Shapira, I., Brandfonbrener, D., Janson, L., and Kakade, S. Soap: Improving and stabilizing shampoo using adam. *arXiv preprint arXiv:2409.11321*, 2024.

Wang, F., Shen, L., Ding, L., Xue, C., Liu, Y., and Ding, C. Simultaneous computation and memory efficient zeroth-order optimizer for fine-tuning large language models. *arXiv preprint arXiv:2410.09823*, 2024.

Yang, H., Zhang, X., Fang, M., and Liu, J. Adaptive multi-hierarchical signsgd for communication-efficient distributed optimization. In *2020 IEEE 21st International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, pp. 1–5. IEEE, 2020.

Yin, J., Dong, J., Wang, Y., De Sa, C., and Kuleshov, V. Modulora: finetuning 2-bit llms on consumer gpus by integrating with modular quantizers. *arXiv preprint arXiv:2309.16119*, 2023.

Yu, Z., Zhou, P., Wang, S., Li, J., and Huang, H. Zeroth-order fine-tuning of llms in random subspaces, 2024. URL https://arxiv.org/abs/2410.08989.

Zaken, E. B., Ravfogel, S., and Goldberg, Y. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. *arXiv preprint arXiv:2106.10199*, 2021.

Zhang, H., Li, Z., Wang, P., and Zhao, H. Selective prefix tuning for pre-trained language models. In *Findings of the Association for Computational Linguistics ACL 2024*, pp. 2806–2813, 2024a.

Zhang, S., Roller, S., Goyal, N., Artetxe, M., Chen, M., Dewan, S., Diab, M., Li, X., Lin, X. V., Mihaylov, T., et al. Opt: Open pre-trained transformer language models, 2022. URL https://arxiv.org/abs/2205.01068.

Zhang, Y., Sun, S., Galley, M., Chen, Y.-C., Brockett, C., Gao, X., Gao, J., Liu, J., and Dolan, B. Dialogpt: Large-scale generative pre-training for conversational response generation. *arXiv preprint arXiv:1911.00536*, 2019.

Zhang, Y., Li, P., Hong, J., Li, J., Zhang, Y., Zheng, W., Chen, P.-Y., Lee, J. D., Yin, W., Hong, M., Wang, Z., Liu, S., and Chen, T. Revisiting zeroth-order optimization for memory-efficient llm fine-tuning: A benchmark, 2024b. URL https://arxiv.org/abs/2402.11592.

Zhu, L., Hu, L., Lin, J., Chen, W.-M., Wang, W.-C., Gan, C., and Han, S. Pockengine: Sparse and efficient fine-tuning in a pocket. In *Proceedings of the 56th Annual IEEE/ACM International Symposium on Microarchitecture*, pp. 1381–1394, 2023a.

Zhu, Y., Zhang, Y., Zhang, Z., et al. Efficient fine-tuning of language models via zeroth-order optimization. *arXiv preprint*, abs/2305.14395, 2023b.

A. Zero-Order Muon with JAGUAR Gradient Approximation

In this section, we address the matrix optimization setting, where the optimization variables X_t are elements of the matrix space $\mathbb{R}^{m\times n}$, rather than the standard vector space $\mathbb{R}^d$. Such a formulation allows for a more direct representation of model parameters, helping to better capture their underlying structure (Bernstein & Newhouse, 2024b; Pethick et al., 2025). For the first time in the literature, we introduce a zero-order version of the Muon (Jordan et al., 2024) algorithm (Algorithm 2), broadening the applicability to matrix-structured optimization tasks where only function evaluations are available.

Algorithm 2 Zero-Order Muon with JAGUAR (`JAGUAR Muon`)

1: **Parameters:** stepsize (learning rate) γ, momentum β, gradient approximation parameter τ, number of Newton-Schulz steps ns_steps, number of iterations T.
2: **Initialization:** choose $X^0 \in \mathbb{R}^{m\times n}$ and $M^{-1} = \mathbf{0} \in \mathbb{R}^{m\times n}$.
3: **for** $t = 0, 1, 2, \ldots, T$ **do**
4: Sample $i_t \sim$ Uniform$(\overline{1,m})$ and $j_t \sim$ Uniform$(\overline{1,n})$
5: Set one-hot matrix E^t with 1 in the (i_t, j_t) coordinate
6: Sample stochastic variable $\xi^t \sim \mathcal{D}$
7: Compute $\widetilde{\nabla}_{i_t} f(X^t, \xi^t) := \frac{\hat{f}(X^t+\tau E^t,\xi^t)-\hat{f}(X^t-\tau E^t,\xi^t)}{2\tau} \in \mathbb{R}$
8: Set $M^t_{i_t,j_t} = \beta M^{t-1}_{i_t,j_t} + (1-\beta)\widetilde{\nabla}_{i_t} f(x^t, \xi^t)$ and $M^t_{i\neq i_t, j\neq j_t} = M^{t-1}_{i\neq i_t, j\neq j_t}$
9: Set $X^{t+1} = X^t - \gamma \cdot$ `Newton_Schulz`$(M^t, K =$ ns_steps$)$
10: **end for**
11: **Return:** $X^{N(T)}$, where $N(T) \sim$ Uniform$(\overline{1,T})$.

1: **Subroutine** `Newton_Schulz`$(A \in \mathbb{R}^{m\times n}, K = 10)$ (Bernstein & Newhouse, 2024b):
2: Set $A^0 = A/\|A\|_F$
3: **for** $k = 0, 1, 2, \ldots, K$ **do**
4: $A^{k+1} = 3/2 \cdot A^k - 1/2 \cdot A^k (A^k)^T A^k$
5: **end for**
6: **Return:** $A^K \approx U_A \cdot V_A^T$.

Algorithm 2 is similar to the first-order Muon algorithm (Jordan et al., 2024), the only difference is that we use zero-order gradient approximation JAGUAR (Veprikov et al., 2024) in line 8.

Let us note that when extending to matrix-valued parameters, it is necessary to slightly modify Assumptions 3.1 and 3.2: all occurrences of the ℓ_2 norm $\|\cdot\|_2$ should be replaced with the Frobenius norm $\|\cdot\|_F$. This modification is justified, as the following property holds for all matrices $A \in \mathbb{R}^{m\times n}$: $\|A\|_F = \|\overline{\text{vec}}(A)\|_2$. We now provide the convergence analysis of `JAGUAR Muon` (Algorithm 2).

Theorem A.1. *Consider Assumptions 3.1, 3.2 (with Frobenius norm) and 3.3. Then* `JAGUAR Muon` *(Algorithm 2) has the following convergence rate:*

$$\mathbb{E}\left[\left\|\nabla f\left(X^{N(T)}\right)\right\|_{\mathcal{S}_1}\right] = \mathcal{O}\Bigg[\frac{\delta_0}{\gamma T} + \frac{m^{1/2}n\left\|\nabla f(X^0)\right\|_2}{T\sqrt{1-\beta}} + \frac{m^{3/2}n^2\gamma}{1-\beta} + \sqrt{1-\beta}m^{1/2}n\sigma + m^{1/2}nL\tau + \frac{m^{1/2}n\Delta}{\tau}\Bigg],$$

where we used a notation $\delta_0 := f(x^0) - f^*$. We also assume that $n \leq m$.

Corollary A.2. *Consider the conditions of Theorem A.1. In order to achieve the ε-approximate solution (in terms of $\mathbb{E}[\|\nabla f(X^{N(T)})\|_{\mathcal{S}_1}] \leq \varepsilon$), Algorithm 2 needs T iterations (ZO calls), for:*

Arbitrary tuning: $\gamma = \gamma_0 \cdot T^{-3/4}(mn)^{-1}, \beta = 1 - T^{-1/2}, \tau = (\Delta/L)^{1/2}, \varepsilon \geq m^{1/2}n\sqrt{\Delta L}$:

$$T = \mathcal{O}\left[\left(\frac{mn\delta_0/\gamma_0 + m^{1/2}n\left\|\nabla f(X^0)\right\|_2 + m^{1/2}nL\gamma_0 + m^{1/2}n\sigma}{\varepsilon}\right)^4\right].$$

Optimal tuning: $\gamma = \sqrt{\frac{\delta_0(1-\beta)}{m^{3/2}n^2LT}}, \beta = 1 - \min\left\{1; \sqrt{\frac{L\delta_0}{T\sigma^2}}\right\}, \tau = (\Delta/L)^{1/2}, \varepsilon \geq m^{1/2}n\sqrt{\Delta L}$:

$$T = \mathcal{O}\left[\frac{\delta_0 L m^{3/2} n^2}{\varepsilon^2} + \frac{\delta_0 L m^{3/2} n^2}{\varepsilon^2} \cdot \left(\frac{m^{3/2} n^2 \sigma}{\varepsilon}\right)^2\right].$$

Discussion. The convergence rate established in Theorem A.1 is consistent with the first-order case (Li & Hong, 2025; Kovalev, 2025). However, there remain zero-order terms depending on τ and Δ, as for Algorithm 1 (see Theorem 3.5 and Discussion part after it). From a proof perspective, Theorems 3.5 and A.1 are very similar, since the orthogonalization operation (`Newton_Schulz`) in Algorithm 2 can be interpreted as taking the sign of the gradient matrix eigenvalues. Accordingly, both the form and the convergence rate criterion are analogous (the ℓ_1 norm for Algorithm 1 and the $\mathcal{S}_1$ norm for Algorithm 2). Nevertheless, the convergence rates of the two algorithms differ slightly. We examine the two boundary cases in the following remark.

Remark A.3. For optimal tuning from Corollary A.2 we can specify the number of iterations of Algorithm 2 to achieve the ε-approximate solution in terms of the total number of parameters $d = m \cdot n$ in the two boundary cases:

- If $n \ll m \approx d$:
$$T_{n \ll m \approx d} = \mathcal{O}\left[\frac{\delta_0 L d^{3/2}}{\varepsilon^2} + \frac{\delta_0 L d^{3/2}}{\varepsilon^2} \cdot \left(\frac{d^{3/2}\sigma}{\varepsilon}\right)^2\right].$$
- If $n \approx m \approx \sqrt{d}$:
$$T_{n \approx m \approx \sqrt{d}} = \mathcal{O}\left[\frac{\delta_0 L d^{7/4}}{\varepsilon^2} + \frac{\delta_0 L d^{7/4}}{\varepsilon^2} \cdot \left(\frac{d^{7/4}\sigma}{\varepsilon}\right)^2\right].$$

Accordingly, comparing these convergence rates with that obtained in Corollary A.2, we observe an improvement by factors of $d^{1/2}$ and $d^{1/4}$, respectively.

B. Classical ZO Muon

Using gradient estimate in the form (2), we adapt the Muon algorithm (Jordan et al., 2024) into zero-order form:

Algorithm 3 Zero-Order Muon (`ZO-Muon`)

Parameters: stepsize (learning rate) γ, gradient approximation parameter τ, number of iterations T.
Initialization: choose $X^0 \in \mathbb{R}^{m \times n}$
for $t = 0, 1, 2, \ldots, T$ **do**
  Sample $E^t \in \mathbb{R}^{m \times n}$ from $\mathcal{N}(0, 1)$
  Compute $G^t = \frac{\hat{f}(X^t + \tau E^t) - \hat{f}(X^t - \tau E^t)}{2\tau} E^t$
  Set $X^{t+1} = X^t - \gamma \cdot$ `Newton_Schulz`(G^t)
end for
Return: X^T

C. Additional Experiments and Fine-Tuning Setup

C.1. Additional experiments

To justify reliability of the proposed in this paper methods we conduct additional experiments with large-size models: Llama2-7B (Touvron et al., 2023) and OPT-13B (Zhang et al., 2022) on WinoGrande (Sakaguchi et al., 2021) and COPA (Roemmele et al., 2011) tasks. Within this series of evaluations we implement learning scheduler - cosine scheduler for Llama2-7B and polynomial decay scheduler for OPT-13B. We repeat the evaluation results from (Zhang et al., 2024b) as baselines in Table 4. However, in the mentioned work authors do not report memory efficiency which is a necessary indicator in parameter-efficient fine-tuning competition.

Table 4: Test accuracy on COPA and WinoGrande for OPT-13B and Llama2-7B with LoRA. Best performance among ZO methods is in **bold**. Blue indicates outperformance of all baseline ZO methods, red indicates matching or exceeding FO-SGD.

Method	OPT-13B	LLaMA2-7B
COPA		
FO-SGD	88	85
Forward-Grad	89	82
ZO-SGD	87	86
ZO-SGD-CONS	88	85
`JAGUAR SignSGD`	**89 ± 0.3**	**88 ± 0.2**
`ZO-Muon`	87 ± 0.2	85 ± 0.2
`JAGUAR Muon`	87 ± 0.2	**88 ± 0.1**
WinoGrande		
FO-SGD	66.9	66.9
Forward-Grad	62.9	64.3
ZO-SGD	62.6	64.3
ZO-SGD-CONS	63.3	64.6
`JAGUAR SignSGD`	**63.7± 0.1**	**64.9± 0.1**
`ZO-Muon`	61.9 ± 0.3	61.6 ± 0.2
`JAGUAR Muon`	62.3 ± 0.2	62.8 ± 0.2

Discussion. This series of experiments on large-scale models further reinforces that the proposed methods consistently outperform the baselines.The ZO-Adam method was not considered in our experiments due to its prohibitively high memory requirements. The results demonstrate the effectiveness and scalability of our approach, confirming its advantages even in challenging, high-capacity settings.

C.2. Evaluation Procedure

Schedulers. We conducted experiments with different scheduling types. Therefore, results for `Jaguar Muon` (Algorithm 2) and `Muon` (Algorithm 3) from Tables 2 and 4 are obtained using polynomial scheduling technique with F32-precision.

Hyperparameter Tuning. To ensure optimal performance, we conducted a grid search over key hyperparameters for each method:

- Momentum parameter: $\beta \in \{10^{-3}, 10^{-2}, 10^{-1}, 8 \cdot 10^{-1}\}$,
- Learning rate: $\gamma \in [10^{-6}, 10^{-1}]$,
- Smoothing parameter: $\tau \in \{10^{-1}, 10^{-2}, 10^{-3}\}$.

Additional fixed parameters include an epsilon of 10^{-3} for numerical stability. The best-performing hyperparameters for each algorithm are detailed on our github `https://github.com/brain-mmo-lab/ZO_LLM`.

Evaluation Metrics. We assess performance using:

- **Test Accuracy:** Measured as the percentage of correct predictions on the test set, reflecting model effectiveness.
- **GPU allocated memory:** Quantified in gigabytes (GB) during training, indicating memory efficiency.

Implementation Details. Experiments were conducted with three independent runs per configuration, each with a randomly selected seed fixed at the start to ensure reproducibility. We report the mean and standard deviation of test accuracy. Following (Malladi et al., 2023b), we employed half-precision (F16) training for ZO methods and mixed-precision (FP16) training for FO methods to optimize memory usage. We use LoRA (Hu et al., 2021) fine-tuning strategy with $r = 16$.

Training was performed on a single NVIDIA A100 GPU and a single NVIDIA H100 GPU, with memory profiling conducted using standard PyTorch utilities.

We report the GPU memory allocation for our methods and baselines in Table 5.

Table 5: GPU allocated memory (GB) for OPT-13B and LLaMA2-7B on WinoGrande and COPA with LoRA

Method	Llama-7B	OPT-13B
COPA		
ZO-SGD	13.219	24.710
ZO-Adam	27.971	38.612
JAGUAR SignSGD	13.219	24.712
ZO-Muon	15.021	25.740
JAGUAR Muon	16.032	25.880
WinoGrande		
ZO-SGD	14.670	26.407
ZO-Adam	29.440	39.872
JAGUAR SignSGD	14.672	26.408
ZO-Muon	16.992	27.416
JAGUAR Muon	17.992	27.440

C.3. Experimental Methodology

Our experimental procedure was designed to rigorously evaluate the proposed methods under controlled conditions. We consider diffiren datasets (SST2, COPA, WinoGrande), models (OPT-1.3B, RoBERTa-Large, Llama2 7B, OPT-13B), fine-tuning schemes (FT, LoRA), and ZO and FO optimization methods (see Tables 2 and 4). We executed the following steps:

1. **Initialization:** Loaded the pre-trained model and initialized trainable parameters (all for FT, LoRA-specific for LoRA).
2. **Hyperparameter Selection:** Performed a preliminary parameter search to identify the best hyperparameters per method, iterating over the specified ranges and selecting based on validation accuracy.
3. **Evaluation:** Computed test accuracy on the dataset test set after each run, averaging results across three runs with different seeds.
4. **Memory Profiling:** Recorded GPU allocated memory during training, ensuring consistency by maintaining identical hardware settings.

This methodology ensures a fair comparison across methods, capturing both performance and resource utilization comprehensively.

D. Proofs for ZO Momentum SignSGD with JAGUAR (Algorithm 1)

D.1. Proof of Lemma 3.4

Proof. We start with applying one step recursion to the momentum form the Algorithm 1:

$$\begin{aligned}
\mathbb{E}\left[\left\|m^t - \nabla f(x^t)\right\|_2^2\right] &= \mathbb{E}\Big[\Big\|m^{t-1} - (1-\beta)\left\langle m^{t-1}, e^t\right\rangle e^t \\
&\qquad + (1-\beta)\widetilde{\nabla}_{i_t} f(x^t,\xi^t) - \nabla f(x^t)\Big\|_2^2\Big] \\
&= \mathbb{E}\Big[\Big\| \left\{I - (1-\beta)e^t(e^t)^T\right\} \underbrace{\left\{m^{t-1} - \nabla f(x^{t-1})\right\}}_{=:a^t} \\
&\qquad + (1-\beta)e^t(e^t)^T \underbrace{\left\{\widetilde{\nabla} f(x^t,\xi^t) - \nabla f(x^t)\right\}}_{=:b^t} \\
&\qquad - \left\{I - (1-\beta)e^t(e^t)^T\right\} \underbrace{\left\{\nabla f(x^t) - \nabla f(x^{t-1})\right\}}_{=:c^t}\Big\|_2^2\Big],
\end{aligned} \tag{3}$$

where we used a notation $\widetilde{\nabla} f(x,\xi) := \sum_{i=1}^d \frac{\hat{f}(x+\tau e^i,\xi) - \hat{f}(x-\tau e^i,\xi)}{2\tau} e^i$, and e^i is the one-hot vector with 1 in the i-th coordinate. In equation (3) we also used the classical notation of the identity matrix $I \in \mathbb{R}^{d\times d}$.

Now using axillary notations a^t, b^t, c^t from equation (3), we divide it into six parts:

$$\begin{aligned}
\mathbb{E}\left[\left\|a^{t+1}\right\|_2^2\right] &= \underbrace{\mathbb{E}\left[\left\|\left\{I - (1-\beta)e^t(e^t)^T\right\} a^t\right\|_2^2\right]}_{①} \\
&\quad + \underbrace{\mathbb{E}\left[\left\|(1-\beta)e^t(e^t)^T b^t\right\|_2^2\right]}_{②} \\
&\quad + \underbrace{\mathbb{E}\left[\left\|\left\{I - (1-\beta)e^t(e^t)^T\right\} c^t\right\|_2^2\right]}_{③} \\
&\quad + \underbrace{\mathbb{E}\left[2\left\langle\left\{I - (1-\beta)e^t(e^t)^T\right\} a^t, (1-\beta)e^t(e^t)^T b^t\right\rangle\right]}_{④} \\
&\quad + \underbrace{\mathbb{E}\left[2\left\langle\left\{I - (1-\beta)e^t(e^t)^T\right\} a^t, \left\{I - (1-\beta)e^t(e^t)^T\right\} c^t\right\rangle\right]}_{⑤} \\
&\quad + \underbrace{\mathbb{E}\left[2\left\langle(1-\beta)e^t(e^t)^T b^t, \left\{I - (1-\beta)e^t(e^t)^T\right\} c^t\right\rangle\right]}_{⑥}.
\end{aligned} \tag{4}$$

Consider ①. Since i_t from Algorithm 1 is generated independent and uniform and $\{m^{s-1}, x^s\}_{s=0}^t$ do not depend on i_t, we can apply tower property:

$$\begin{aligned}
① &= \mathbb{E}\left[\left\|\left\{I - (1-\beta)e^t(e^t)^T\right\} a^t\right\|_2^2\right] \\
&= \mathbb{E}\left[(a^t)^T \left\{I - (1-\beta)e^t(e^t)^T\right\}^T \left\{I - (1-\beta)e^t(e^t)^T\right\} a^t\right] \\
&= \mathbb{E}\left[(a^t)^T \left\{I - (1-\beta)(2-(1-\beta))e^t(e^t)^T\right\} a^t\right] \\
&= \mathbb{E}\left[(a^t)^T \cdot \mathbb{E}_{i_t\sim U[1;d]}\left[I - (1-\beta^2)e^t(e^t)^T\right] \cdot a^t\right] \\
&= \mathbb{E}\left[(a^t)^T \cdot \left(1 - \frac{1-\beta^2}{d}\right) I \cdot a^t\right] = \left(1 - \frac{1-\beta^2}{d}\right)\mathbb{E}\left[\left\|a^t\right\|_2^2\right].
\end{aligned} \tag{5}$$

Here we used the fact that $\left(e^t(e^t)^T\right)^T e^t(e^t)^T = e^t(e^t)^T$ and $\mathbb{E}_{i_t\sim U[1;d]}\left[e^t(e^t)^T\right] = \frac{1}{d}I$.

Similarly to equation (5), we can estimate ② and ③:

$$② = \mathbb{E}\left[\left\|(1-\beta)e^t(e^t)^T b^t\right\|_2^2\right] = \frac{(1-\beta)^2}{d}\mathbb{E}\left[\left\|b^t\right\|^2\right],$$
$$③ = \mathbb{E}\left[\left\|\left\{I-(1-\beta)e^t(e^t)^T\right\}c^t\right\|_2^2\right] = \left(1-\frac{1-\beta^2}{d}\right)\mathbb{E}\left[\left\|c^t\right\|^2\right].$$

Since $b^t = \widetilde{\nabla} f(x^t,\xi^t) - \nabla f(x^t)$, we can use Lemma 4 from (Veprikov et al., 2024) with $\sigma_f = 0, \sigma_\nabla = \sigma$ and obtain the result of the form:

$$② \leq \frac{(1-\beta)^2}{d}\cdot\left(dL^2\tau^2 + 2d\sigma^2 + \frac{2d\Delta^2}{\tau^2}\right), \tag{6}$$

where L, σ and Δ come from Assumptions 3.1, 3.2 and 3.3.

Since $c^t = \nabla f(x^t) - \nabla f(x^{t-1})$, we can use Assumption 3.1 and obtain:

$$③ \leq \left(1-\frac{1-\beta^2}{d}\right)L^2\left\|x^t - x^{t-1}\right\|_2^2 = \left(1-\frac{1-\beta^2}{d}\right)L^2\left\|\text{sign}(m^t)\right\|_2^2$$
$$= \left(1-\frac{1-\beta^2}{d}\right)dL^2\gamma^2 \leq dL^2\gamma^2. \tag{7}$$

Consider ④. Let us move all matrixes to the left side of the dot product:

$$④ = \mathbb{E}\left[2\left\langle(1-\beta)\left\{I-(1-\beta)e^t(e^t)^T\right\}e^t(e^t)^T\cdot a^t, b^t\right\rangle\right]$$
$$= \mathbb{E}\left[2\left\langle(1-\beta)\beta e^t(e^t)^T\cdot a^t, b^t\right\rangle\right].$$

Now we use tower property for i_t as we did for ①, ②, ③ and use the definitions of a^t and b^t:

$$④ = \frac{(1-\beta)\beta}{d}\cdot\mathbb{E}\left[2\left\langle a^t, b^t\right\rangle\right]$$
$$= \frac{(1-\beta)\beta}{d}\cdot\mathbb{E}\left[2\left\langle m^{t-1}-\nabla f(x^{t-1}), \widetilde{\nabla} f(x^t,\xi^t) - \nabla f(x^t)\right\rangle\right].$$

We now again use tower property, but with stochastic variable ξ^t. Since $\{m^{s-1}, x^s\}_{s=0}^t$ do not depend on ξ^t, we can obtain that:

$$④ = \frac{(1-\beta)\beta}{d}\cdot\mathbb{E}\left[2\left\langle m^{t-1}-\nabla f(x^{t-1}), \mathbb{E}_{\xi^t}\left[\widetilde{\nabla} f(x^t,\xi^t)\right] - \nabla f(x^t)\right\rangle\right]$$
$$\leq \frac{(1-\beta)\beta}{2d}\cdot\mathbb{E}\left[\left\|m^{t-1}-\nabla f(x^{t-1})\right\|_2^2\right] \tag{8}$$
$$+ \frac{2(1-\beta)\beta}{d}\cdot\mathbb{E}\left[\left\|\mathbb{E}_{\xi^t}\left[\widetilde{\nabla} f(x^t,\xi^t)\right] - \nabla f(x^t)\right\|_2^2\right].$$

In (8) we use Fenchel-Young inequality. For estimating $\|\mathbb{E}_{\xi^t}[\widetilde{\nabla} f(x^t,\xi^t)] - \nabla f(x^t)\|_2^2$ we again can use Lemma 4 from (Veprikov et al., 2024) but now with $\sigma_\nabla = \sigma_f = 0$ since we have no randomness in $\mathbb{E}_{\xi^t}\left[\widetilde{\nabla} f(x^t,\xi^t)\right]$. Therefore ④ is bounded as:

$$④ \leq \frac{(1-\beta)\beta}{2d}\cdot\mathbb{E}\left[\left\|a^t\right\|_2^2\right] + \frac{2(1-\beta)\beta}{d}\cdot\left(dL^2\tau^2 + \frac{2d\Delta^2}{\tau^2}\right). \tag{9}$$

Consider ⑤. Similar to ④ we can obtain:

$$⑤ = \mathbb{E}\left[2\left\langle\left\{I-(1-\beta)e^t(e^t)^T\right\}a^t, \left\{I-(1-\beta)e^t(e^t)^T\right\}c^t\right\rangle\right]$$
$$= \mathbb{E}\left[2\left\langle\left\{I-(1-\beta^2)e^t(e^t)^T\right\}a^t, c^t\right\rangle\right]$$

$$
\begin{aligned}
&= \left(1 - \frac{1-\beta^2}{d}\right) \cdot \mathbb{E}\left[2\left\langle a^t, c^t\right\rangle\right] \\
&\leq \left(1 - \frac{1-\beta^2}{d}\right) \cdot \frac{1-\beta}{2d} \cdot \mathbb{E}\left[\left\|a^t\right\|_2^2\right] + \left(1 - \frac{1-\beta^2}{d}\right) \cdot \frac{2d}{1-\beta} \cdot \mathbb{E}\left[\left\|c^t\right\|_2^2\right] \\
&\leq \frac{1-\beta}{2d} \cdot \mathbb{E}\left[\left\|a^t\right\|_2^2\right] + \frac{2d}{1-\beta} \cdot dL^2\gamma^2. \qquad (10)
\end{aligned}
$$

Finally, we estimate ⑥ in the same way:

$$
\begin{aligned}
⑥ &= \mathbb{E}\left[2\left\langle (1-\beta)e^t(e^t)^T b^t, \left\{I - (1-\beta)e^t(e^t)^T\right\} c^t\right\rangle\right] \\
&= \mathbb{E}\left[2\left\langle (1-\beta)\beta e^t(e^t)^T b^t, c^t\right\rangle\right] \\
&= \frac{(1-\beta)\beta}{d} \cdot \mathbb{E}\left[2\left\langle b^t, c^t\right\rangle\right] \\
&\leq \frac{(1-\beta)\beta}{d} \cdot \mathbb{E}\left[\left\|\mathbb{E}_{\xi^t}\left[\widetilde{\nabla} f(x^t, \xi^t)\right] - \nabla f(x^t)\right\|_2^2\right] + \frac{(1-\beta)\beta}{d} \cdot \mathbb{E}\left[\left\|c^t\right\|_2^2\right] \\
&\leq \frac{(1-\beta)\beta}{d} \cdot \left(dL^2\tau^2 + \frac{2d\Delta^2}{\tau^2}\right) + \frac{(1-\beta)\beta}{d} \cdot dL^2\gamma^2. \qquad (11)
\end{aligned}
$$

We made it! Now let us combine equations (5), (6), (7), (9), (10) and (11) to bound $\mathbb{E}[\|a^{t+1}\|_2^2]$ from equation (4):

$$
\begin{aligned}
\mathbb{E}\left[\left\|a^{t+1}\right\|_2^2\right] \leq &\left(1 - \frac{1-\beta}{d}\left[\underbrace{1+\beta}_{(5)} - \underbrace{\frac{\beta}{2}}_{(9)} - \underbrace{\frac{1}{2}}_{(10)}\right]\right) \cdot \mathbb{E}\left[\left\|a^t\right\|_2^2\right] \\
&+ \frac{1-\beta}{d}\left(\underbrace{1-\beta}_{(6)} + \underbrace{2\beta}_{(9)} + \underbrace{\beta}_{(11)}\right) \cdot \left(dL^2\tau^2 + \frac{2d\Delta^2}{\tau^2}\right) + \underbrace{\frac{(1-\beta)^2}{d}}_{(6)} \cdot 2d\sigma^2 \\
&+ \left(\underbrace{1}_{(7)} + \underbrace{\frac{2d}{1-\beta}}_{(10)} + \underbrace{\frac{(1-\beta)\beta}{d}}_{(11)}\right) \cdot dL^2\gamma^2 \\
\leq &\left(1 - \frac{1-\beta^2}{2d}\right) \cdot \mathbb{E}\left[\left\|a^t\right\|_2^2\right] \\
&+ 3\frac{1-\beta}{d} \cdot \left(dL^2\tau^2 + \frac{2d\Delta^2}{\tau^2}\right) + 2\frac{(1-\beta)^2}{d} \cdot d\sigma^2 + \frac{4d}{1-\beta} \cdot dL^2\gamma^2.
\end{aligned}
$$

By unrolling the recursion in the last inequality we obtain:

$$
\begin{aligned}
\mathbb{E}\left[\left\|m^t - \nabla f(x^t)\right\|_2^2\right] &\leq 8\frac{d^2}{(1-\beta)(1-\beta^2)} \cdot dL^2\gamma^2 + 4\frac{(1-\beta)^2}{1-\beta^2} \cdot d\sigma^2 \\
&\quad + 6\frac{1-\beta}{1-\beta^2} \cdot \left(dL^2\tau^2 + \frac{2d\Delta^2}{\tau^2}\right) + \left(\frac{1-\beta^2}{2d}\right)^t \left\|\nabla f(x^0)\right\|_2^2 \\
&= \mathcal{O}\Bigg[\frac{d^3}{(1-\beta)^2}L^2\gamma^2 + (1-\beta)d\sigma^2 + dL^2\tau^2 + \frac{2d\Delta^2}{\tau^2} \\
&\qquad + \left(1 - \frac{1-\beta}{2d}\right)^t \left\|\nabla f(x^0)\right\|_2^2\Bigg].
\end{aligned}
$$

This finishes the proof. □

D.2. Proof of Theorem 3.5

Proof. We start from using Lemma 1 from (Sun et al., 2023). For the points x^t, generated by Algorithm 1 it holds that:

$$f(x^{t+1}) - f(x^t) \leq -\gamma \left\|\nabla f(x^t)\right\|_1 + 2\sqrt{d}\gamma \left\|m^t - \nabla f(x^t)\right\|_2 + \frac{dL\gamma^2}{2}. \tag{12}$$

Now we take mathematical expectation of the both sides of the inequality (12) and use the results from Lemma 3.4:

$$\begin{aligned}\mathbb{E}\left[f(x^{t+1})\right] - \mathbb{E}\left[f(x^t)\right] &\leq -\gamma\mathbb{E}\left[\left\|\nabla f(x^t)\right\|_1\right] + 2\sqrt{d}\gamma\mathbb{E}\left[\left\|m^t - \nabla f(x^t)\right\|_2\right] + \frac{dL\gamma^2}{2} \\ &= -\gamma\mathbb{E}\left[\left\|\nabla f(x^t)\right\|_1\right] + \mathcal{O}\Bigg[\frac{d^2}{1-\beta}\cdot L\gamma^2 + \sqrt{1-\beta}d\gamma\sigma + d\gamma L\tau \\ &\quad + \frac{d\gamma\Delta}{\tau} + \sqrt{d}\gamma\left(1 - \frac{1-\beta}{d}\right)^{t/2}\left\|\nabla f(x^0)\right\|_2\Bigg] + \frac{dL\gamma^2}{2}.\end{aligned}$$

Consequently, after summing all T steps, we obtain:

$$\begin{aligned}\gamma\sum_{t=0}^{T}\mathbb{E}\left[\left\|\nabla f(x^t)\right\|_1\right] &= \mathcal{O}\Bigg[f(x^0) - f(x^T) + T\cdot\left(\frac{d^2}{1-\beta}\cdot L\gamma^2 + \sqrt{1-\beta}d\gamma\sigma + d\gamma L\tau\right) \\ &\quad + T\cdot\frac{d\gamma\Delta}{\tau} + \sqrt{d}\gamma\sum_{t=0}^{T}\left(1 - \frac{1-\beta}{d}\right)^{t/2}\left\|\nabla f(x^0)\right\|_2\Bigg].\end{aligned} \tag{13}$$

Now, we divide equation (13) by γT from both sides and obtain:

$$\frac{1}{T}\sum_{t=0}^{T}\mathbb{E}\left[\left\|\nabla f(x^t)\right\|_1\right] = \mathcal{O}\left[\frac{\delta_0}{\gamma T} + \frac{d\left\|\nabla f(x^0)\right\|_2}{T\sqrt{1-\beta}} + \frac{d^2L\gamma}{1-\beta} + \sqrt{1-\beta}d\sigma + dL\tau + \frac{d\Delta}{\tau}\right],$$

where we used a notation $\delta_0 := f(x^0) - f^*$. This finishes the proof. □

E. Proofs for ZO Muon with JAGUAR (Algorithm 2)

E.1. Technical Lemmas

Lemma E.1. *Consider two arbitrary matrixes A, B of the same shape and their SVD decomposition: $A = U_A\Sigma_A V_A^T$, $B = U_B\Sigma_B V_B^T$. Define r_A and r_B as ranks of A and B, then it holds that*

$$\left|\left\langle A, U_A V_A^T - U_B V_B^T\right\rangle\right| \leq 2\left\|A - B\right\|_{\mathcal{S}_1} \leq 2\sqrt{\operatorname{rank}(A-B)}\left\|A - B\right\|_F.$$

Proof. We first provide an axillary notation:

$$\delta := \left\langle A, U_A V_A^T - U_B V_B^T\right\rangle.$$

Because U_A and V_A have orthonormal columns:

$$\langle A,\, U_A V_A^\top\rangle = \operatorname{tr}\big(V_A\Sigma_A U_A^\top U_A V_A^\top\big) = \operatorname{tr}(\Sigma_A) = \|A\|_{\mathcal{S}_1}.$$

Hence

$$\delta = \|A\|_{\mathcal{S}_1} - \langle A,\, U_B V_B^\top\rangle.$$

Insert B and regroup:

$$\delta = \|A\|_{\mathcal{S}_1} - \big(\langle B,\, U_B V_B^\top\rangle + \langle A - B,\, U_B V_B^\top\rangle\big) = \|A\|_{\mathcal{S}_1} - \|B\|_{\mathcal{S}_1} - \langle A - B,\, U_B V_B^\top\rangle.$$

The first difference is controlled by the triangle inequality for the nuclear norm:

$$\left|\|A\|_{\mathcal{S}_1} - \|B\|_{\mathcal{S}_1}\right| \leq \|A - B\|_{\mathcal{S}_1}.$$

For the second term, Hölder's inequality with $\|U_B V_B^\top\|_2 = 1$ gives

$$\left|\langle A - B,\, U_B V_B^\top \rangle\right| \leq \|A - B\|_{\mathcal{S}_1}.$$

Therefore

$$|\delta| \leq \|A - B\|_{\mathcal{S}_1} + \|A - B\|_{\mathcal{S}_1} = 2\,\|A - B\|_{\mathcal{S}_1}.$$

Using the connection between the Frobenius ($\mathcal{S}_2$) by nuclear ($\mathcal{S}_1$) norms we obtain that:

$$|\delta| = \left\langle A, U_A V_A^T - U_B V_B^T \right\rangle \leq 2\,\|A - B\|_{\mathcal{S}_1} \leq 2\sqrt{\operatorname{rank}(A - B)}\,\|A - B\|_F.$$

The factor 2 in the nuclear norm bound is sharp, as equality holds for $B = -A$. This finishes the proof. □

We now provide lemma similar to the step Lemma 1 from (Sun et al., 2023), but in the matrix case.

Lemma E.2 (Step lemma for Muon with momentum)**.** *Let f be an L-smooth function (Assumption 3.1), and let $X^\dagger, M \in \mathbb{R}^{m\times n}$ with $m \geq n$ be an arbitrary matrixes. We define*

$$X^\ddagger := X^\dagger - \gamma \cdot U_M V_M^T,$$

where $\gamma > 0$ and $U_M V_M^T$ comes from SVD decomposition of M: $M = U_M \Sigma_M V_M^T$. Then, it holds that:

$$f\left(X^\ddagger\right) - f\left(X^\dagger\right) \leq -\gamma \left\|\nabla f\left(X^\dagger\right)\right\|_{\mathcal{S}_1} + 2\sqrt{n}\gamma \left\|\nabla f\left(X^\dagger\right) - M\right\|_F + \frac{Ln\gamma^2}{2}.$$

Proof. The L-smoothness of the gradient (Assumption 3.1) gives us

$$\begin{aligned}
f\left(X^\ddagger\right) - f\left(X^\dagger\right) &\leq \left\langle \nabla f\left(X^\dagger\right), X^\ddagger - X^\dagger \right\rangle + \frac{L}{2}\left\|X^\ddagger - X^\dagger\right\|_F^2 \\
&= -\gamma \left\langle \nabla f\left(X^\dagger\right), U_M V_M^T \right\rangle + \frac{Ln\gamma^2}{2} \\
&= -\gamma \left\langle \nabla f\left(X^\dagger\right), U_\nabla V_\nabla^T \right\rangle + \gamma \left\langle \nabla f\left(X^\dagger\right), U_\nabla V_\nabla^T - U_M V_M^T \right\rangle + \frac{Ln\gamma^2}{2},
\end{aligned}$$

where $U_\nabla V_\nabla^T$ comes from SVD decomposition of $\nabla f\left(X^\dagger\right)$: $\nabla f\left(X^\dagger\right) = U_\nabla \Sigma_\nabla V_\nabla^T$. Therefore the first dot product takes form:

$$-\gamma \left\langle \nabla f\left(X^\dagger\right), U_\nabla V_\nabla^T \right\rangle = -\gamma \mathrm{tr}\left(V_\nabla \Sigma_\nabla U_\nabla^T U_\nabla V_\nabla^T\right) = -\gamma \mathrm{tr}\left(\Sigma_\nabla\right) = -\gamma \left\|\nabla f\left(X^\dagger\right)\right\|_{\mathcal{S}_1}.$$

Now we utilize Lemma E.1 with $A = \nabla f\left(X^\dagger\right)$ and $B = M$:

$$\begin{aligned}
f\left(X^\ddagger\right) - f\left(X^\dagger\right) &\leq -\gamma \left\|\nabla f\left(X^\dagger\right)\right\|_{\mathcal{S}_1} + 2\gamma \left\|\nabla f\left(X^\dagger\right) - M\right\|_{\mathcal{S}_1} + \frac{Ln\gamma^2}{2} \\
&\leq -\gamma \left\|\nabla f\left(X^\dagger\right)\right\|_{\mathcal{S}_1} + 2\sqrt{n}\gamma \left\|\nabla f\left(X^\dagger\right) - M\right\|_F + \frac{Ln\gamma^2}{2}.
\end{aligned}$$

This finishes the proof. □

E.2. Proof of Theorem A.1

Proof. We start from using Lemma E.2. For the points X^t, generated by Algorithm 2 it holds that:

$$f\left(X^{t+1}\right) - f\left(X^t\right) \leq -\gamma \left\|\nabla f\left(X^t\right)\right\|_{\mathcal{S}_1} + 2\sqrt{n}\gamma \left\|\nabla f\left(X^t\right) - M^t\right\|_F + \frac{Ln\gamma^2}{2}. \tag{14}$$

Now we take mathematical expectation of the both sides if (14) and bound the term $\mathbb{E}[\|\nabla f\left(X^t\right) - M^t\|_F]$ we again use Lemma 3.4 with $x^t = \overline{\text{vec}}(X^t)$ and $m^t = \overline{\text{vec}}(M^t)$. The result of Lemma 3.4 holds true with $d = m \cdot n$, since $\|A\|_F = \|\overline{\text{vec}}(A)\|_2$. Therefore (14) takes form:

$$\begin{aligned}
\mathbb{E}\left[f(X^{t+1})\right] - \mathbb{E}\left[f(X^t)\right] &\leq -\gamma\mathbb{E}\left[\left\|\nabla f(X^t)\right\|_{\mathcal{S}_1}\right] + 2\sqrt{n}\gamma\mathbb{E}\left[\left\|M^t - \nabla f(X^t)\right\|_2\right] + \frac{nL\gamma^2}{2} \\
&= -\gamma\mathbb{E}\left[\left\|\nabla f(X^t)\right\|_{\mathcal{S}_1}\right] + n^{1/2}\mathcal{O}\Bigg[\frac{(mn)^{3/2}}{1-\beta}\cdot L\gamma^2 \\
&\quad + \sqrt{1-\beta}(mn)^{1/2}\gamma\sigma + (mn)^{1/2}\gamma L\tau + \frac{(mn)^{1/2}\gamma\Delta}{\tau} \\
&\quad + n^{1/2}\gamma\left(1 - \frac{1-\beta}{mn}\right)^{t/2}\left\|\nabla f(X^0)\right\|_2\Bigg] + \frac{nL\gamma^2}{2}. \\
&= -\gamma\mathbb{E}\left[\left\|\nabla f(X^t)\right\|_{\mathcal{S}_1}\right] + \mathcal{O}\Bigg[\frac{m^{3/2}n^2}{1-\beta}\cdot L\gamma^2 \\
&\quad + \sqrt{1-\beta}m^{1/2}n\gamma\sigma + m^{1/2}n\gamma L\tau + \frac{m^{1/2}n\gamma\Delta}{\tau} \\
&\quad + n^{1/2}\gamma\left(1 - \frac{1-\beta}{mn}\right)^{t/2}\left\|\nabla f(X^0)\right\|_2\Bigg].
\end{aligned}$$

Consequently, after summing all T steps, we obtain:

$$\begin{aligned}
\gamma\sum_{t=0}^{T}\mathbb{E}\left[\left\|\nabla f(X^t)\right\|_{\mathcal{S}_1}\right] = \mathcal{O}\Bigg[&f(X^0) - f(X^T) \\
&+ T\cdot\left(\frac{m^{3/2}n^2}{1-\beta}\cdot L\gamma^2 + \sqrt{1-\beta}m^{1/2}n\gamma\sigma\right) \\
&+ T\cdot\left(m^{1/2}n\gamma L\tau + \frac{m^{1/2}n\gamma\Delta}{\tau}\right) \\
&+ n^{1/2}\gamma\sum_{t=0}^{T}\left(1 - \frac{1-\beta}{mn}\right)^{t/2}\left\|\nabla f(X^0)\right\|_2\Bigg].
\end{aligned} \tag{15}$$

Now, we divide equation (15) by γT from both sides and obtain:

$$\begin{aligned}
\frac{1}{T}\sum_{t=0}^{T}\mathbb{E}\left[\left\|\nabla f(X^t)\right\|_{\mathcal{S}_1}\right] = \mathcal{O}\Bigg[&\frac{\delta_0}{\gamma T} + \frac{m^{1/2}n\left\|\nabla f(x^0)\right\|_2}{T\sqrt{1-\beta}} + \frac{m^{3/2}n^2\gamma}{1-\beta} + \sqrt{1-\beta}m^{1/2}n\sigma \\
&+ m^{1/2}nL\tau + \frac{m^{1/2}n\Delta}{\tau}\Bigg],
\end{aligned}$$

where we used a notation $\delta_0 := f(x^0) - f^*$. This finishes the proof. □