HELENE: <u>Hessian Layer-wise Clipping and</u> Gradi<u>ent Annealing for Accelerating Fine-</u> Tuning LLM with Zeroth-Order Optimization

Anonymous authors

Paper under double-blind review

ABSTRACT

Fine-tuning large language models (LLMs) poses significant memory challenges, as the back-propagation process demands extensive resources, especially with growing model sizes. Recent work, MeZO, addresses this issue using a zerothorder (ZO) optimization method, which reduces memory consumption by matching the usage to the inference phase. However, MeZO experiences slow convergence due to varying curvatures across model parameters. To overcome this limitation, we introduce HELENE, a novel scalable and memory-efficient optimizer that integrates annealed A-GNB gradients with a diagonal Hessian estimation and layerwise clipping, serving as a second-order pre-conditioner. This combination allows for faster and more stable convergence. Our theoretical analysis demonstrates that HELENE improves convergence rates, particularly for models with heterogeneous layer dimensions, by reducing the dependency on the total parameter space dimension. Instead, the method scales with the largest layer dimension, making it highly suitable for modern LLM architectures. Experimental results on RoBERTa-large and OPT-1.3B across multiple tasks show that HELENE achieves up to a $20 \times$ speedup compared to MeZO, with average accuracy improvements of 1.5%. Furthermore, HELENE remains compatible with both full parameter tuning and parameter-efficient fine-tuning (PEFT), outperforming several state-of-the-art optimizers. The codes will be released after reviewing.

031 032

033

006

008 009 010

011

013

014

015

016

017

018

019

021

023

025

026

027

028

029

1 INTRODUCTION

LLMs have demonstrated remarkable capabilities across various downstream tasks. Fine-tuning these
models has become the standard approach for improving task-specific performance, in which the firstorder optimizers like Stochastic Gradient Descent (SGD) (Robbins & Monro, 1951), Adam (Diederik,
2014) and AdamW (Hutter & Loshchilov, 2017) are widely used. While effective, however, these
methods demand significant memory resources primarily due to the backpropagation process, which
makes fine-tuning challenging, especially for large-scale models. To overcome this limitation, Malladi
et al. (2023) proposed a memory-efficient zeroth-order optimizer (MeZO) that estimates gradients
using only two forward passes per training step, contributing to considerable memory savings.

042 However, recent studies show that loss functions in deep learning often exhibit heterogeneous 043 curvatures across different model parameters and different model layers (Sagun et al., 2016; Ghorbani 044 et al., 2019; Zhang et al., 2022; Yao et al., 2020), which poses challenges to zeroth-order (ZO) optimization. This variation in curvature can overall hinder training efficiency and lead to the suboptimal solution. To address this issue, more advanced techniques are required, such as incorporating 046 second-order information to better account for curvature differences (Liu et al., 2023; Tran & 047 Cutkosky, 2022; Jahani et al., 2021). However, in ZO optimization, directly computing the Hessian 048 from first-order derivatives is nearly impossible, and partial Hessian evaluations are computationally intensive, leading to slower convergence. Moreover, we also observe that these methods like Naive Newton's method and Sophia (Liu et al., 2023) fail in fine-tuning LLMs in practice as illustrated in 051 Figure 1 and Figure 2. 052

To overcome the aforementioned challenges, we propose HELENE, a novel optimizer designed to estimate second-order curvature information efficiently in the context of ZO optimization. Originating

063 064

065

066

067

071

072 073

096

098

099

100

102

103

104

054





068 Figure 1: The motivating toy example. HE-069 LENE can maintain stable updates when facing curvature issues, while other second-order optimizers are severely affected by them.

Figure 2: Comparison of HELENE with Newton's method and Sophia. The performance of this training loss cross-checks with the toy sample in Figure 1.

074 from label-sampling-based Gaussian-Newton Garlett (GNB) Estimator (Schraudolph, 2002; Wei et al., 075 2020; Martens, 2020), in our HELENE algorithm, we introduce a label-sampling-free and efficient 076 Hessian estimator called the asymptotic Gauss-Newton-Bartlett estimator (A-GNB) Estimator, which 077 estimates the diagonal of the Hessian matrix. A-GNB is proven to asymptotically converge to the unbiased Gauss Newton matrix. Additionally, HELENE includes a layer-wise adaptive clipping mechanism that enables more precise curvature-aware updates, while magnitude-based clipping 079 helps prevent the overestimation of extreme values in the Hessian diagonal. Unlike existing methods 080 that rely on global clipping, which can distort gradient signals, HELENE preserves the integrity of 081 gradient information by applying clipping on a per-layer basis.

083 One of the key innovations of HELENE is its ability to adaptively clip Hessian updates according to 084 the curvature of each layer, which significantly enhances convergence rates. While the convergence 085 of state-of-the-art optimizers like MeZO-Sophia which may require $\mathcal{O}(d)$ steps; in contrast, the convergence of HELENE requires significantly less steps, which is $\mathcal{O}(\max_i d_i)$ based on the largest 086 layer dimension $\max_i d_i$ across all layers, making it more suitable for modern deep architectures. We 087 observed that zero-order methods require a stronger emphasis on momentum due to the increasing 088 noise in SPSA gradient estimation as optimization progresses, due to the greater difficulty of training 089 as the model parameters approach a local minimum, causing the noise magnitude from sampling 090 perturbations to exceed that of the true gradient signal. So contracting to most momentum design in 091 first-order methods, the proposed HELENE algorithm also integrates an novel annealing exponential 092 moving average (EMA) of the gradients, tailored for Zero-order methods, where the factor alpha dynamically reduce the weight of the gradient in the momentum update. 094

- Overall, our key contributions can be summarized as follows: 095
 - 1. HELENE integrates a novel asymptotic Gauss-Newton-Bartlett (A-GNB) estimator that efficiently estimates the diagonal of the Hessian matrix without the need for label sampling which may incur more noise in the Hessian estimation. This estimator asymptotically converges to the unbiased diagonal Gauss Newton matrix, improving the efficiency and precision of curvature-aware updates. In our proposed method, we also devise a new layerwise adaptive clipping mechanism by adjusting Hessian updates according to the curvature of each layer. HELENE integrates an new annealing exponential moving average (EMA) of the gradients, ensuring robustness in non-convex loss landscapes.
- 2. Our theoretical analysis demonstrates that HELENE achieves improved convergence rates 105 compared to existing methods, particularly for models with many layers. By reducing the convergence steps from $\mathcal{O}(d)$ to $\mathcal{O}(\max_i d_i)$, HELENE is provably more scalable for 107 modern deep learning architectures, especially LLM fine-tuning.

108
3. HELENE achieves up to 20× speedup compared to MeZO and improves performance by 1.5% on average. We conduct extensive experiments on RoBERTa-large and OPT-1.3B across various downstream tasks to verify HELENE's effectiveness. Furthermore, we demonstrate that HELENE not only remains compatible with full parameter tuning and PEFT, but also outperforms many of the latest optimizers across a range of tasks.

114 2 PRELIMINARIES

In this section, we briefly review essential background concepts related to zeroth-order optimization and diagonal Hessian approximation, which are fundamental to the design of our proposed method.

119 2.1 ZEROTH-ORDER GRADIENT ESTIMATORS AND MEZO

121 Zeroth-order (ZO) optimization has long been studied in the context of convex and non-convex 122 objectives. One of the typical ZO gradient estimators is the simultaneous perturbation stochastic 123 approximation (SPSA) (Spall, 1992; Maryak & Chin, 2001). Given a model with parameters $\theta \in \mathbb{R}^d$ 124 and loss function \mathcal{L} , SPSA estimates the gradient on a minibatch \mathcal{B} as:

$$g_{\epsilon}(\boldsymbol{\theta}) = \frac{\mathcal{L}(\boldsymbol{\theta} + \epsilon \boldsymbol{z}; \mathcal{B}) - \mathcal{L}(\boldsymbol{\theta} - \epsilon \boldsymbol{z}; \mathcal{B})}{2\epsilon} \boldsymbol{z} \approx \boldsymbol{z} \boldsymbol{z}^{\top} \nabla \mathcal{L}(\boldsymbol{\theta}; \mathcal{B})$$
(1)

where $m{z} \in \mathbb{R}^d$ with $m{z} \sim \mathcal{N}(m{0}, m{I}_d)$ and ϵ is the perturbation scale.

Building on the basic principles of ZO optimization, MeZO (Malladi et al., 2023) introduces a memory-efficient implementation of ZO-SGD. This approach reduces memory requirements, allowing optimization to proceed with the same memory usage as the inference phase of a model. The key innovation in MeZO lies in its use of a consistent random seed s to sample the random vector z, ensuring the same perturbation z at each step.

133 134

113

116

117

118

125 126 127

128

2.2 DIAGONAL HESSIAN APPROXIMATION

136 While zeroth-order methods like MeZO provide valuable tools for gradient estimation, optimization 137 can be significantly enhanced by incorporating second-order information, such as curvature. However, 138 directly computing and applying the full Hessian matrix is computationally expensive, particularly in 139 high-dimensional parameter spaces. Specifically, directly applying the Hessian pre-conditioner by calculating the inverse Hessian and multiplying it with the gradient vector at each iteration $H^{-1}g$ 140 is particularly computationally expensive. To address this challenge, inexact Newton methods have 141 been developed, where approximations of the Hessian are used instead of the full matrix (Dembo 142 et al., 1982; Bollapragada et al., 2019; Xu et al., 2020). 143

A simple yet effective alternative is to approximate the Hessian by its diagonal elements, which
 reduces computational complexity while retaining useful curvature information. In this approach, a
 general descent direction can be written as follows:

 $\Delta \boldsymbol{\theta} \approx \operatorname{diag}(\boldsymbol{H})^{-1}\boldsymbol{g},$

where diag(H) represents the diagonal elements of the Hessian matrix. This method enhances optimization by enabling efficient inverse Hessian application and supporting inexact Newton methods, providing improved convergence in complex problems.

3 Method

In this section, we formally present HELENE in Section 3.2, with pseudo-code provided in Algorithm 1. In Section 3.4, we introduce A-GNB, followed by a detailed discussion of layer-wise clipped diagonal Hessian in Section 3.5.

3.1 MOTIVATION

158 159

157

147

152

153

Highly variable curvature across different layers and parameters. Fine-tuning large language
 models (LLMs) has become essential for achieving state-of-the-art performance on various down stream tasks. Commonly employed first-order optimizers such as Stochastic Gradient Descent

162 (SGD)(Robbins & Monro, 1951), Adam(Diederik, 2014), and AdamW (Hutter & Loshchilov, 2017) 163 have proven effective in this regard. However, these methods require substantial memory, making 164 them difficult to apply to large models in memory-constrained environments. To mitigate this, 165 zeroth-order (ZO) optimizers, such as MeZO (Malladi et al., 2023), have been introduced, offer-166 ing memory-efficient solutions by approximating gradients through forward passes. Nevertheless, even with memory savings, existing ZO methods encounter significant challenges when dealing 167 with heterogeneous curvatures in LLMs, which can lead to inefficient convergence and sub-optimal 168 solutions. One key challenge is the inability of optimizers to adapt to the highly variable curvature across different layers and parameters in large models (Sagun et al., 2016; Ghorbani et al., 2019; 170 Zhang et al., 2022). While techniques that estimate second-order information—such as curvature-171 aware methods—have shown promise in improving optimization efficiency (Liu et al., 2023; Tran & 172 Cutkosky, 2022; Jahani et al., 2021), they are challenging to integrate into ZO optimizers due to the 173 noise from label-sampling and the difficulty of computing or approximating the Hessian efficiently in 174 high-dimensional spaces. 175

Limitation of EMA to balance short-term gradient noise and long-term convergence. A com-176 monly used technique to manage these curvature variations is the Exponential Moving Average 177 (EMA), which smooths the gradient updates over iterations. However, EMA alone can be insufficient 178 for highly non-convex loss landscapes, especially when it lacks mechanisms to adaptively adjust the 179 weights between the past momentum and the current gradient in the presence of strong noise in gradi-180 ent estimation. Without annealing, EMA risks accumulating excessive bias over time, particularly 181 when ZO gradient estimation is noisy, leading to suboptimal convergence. This issue is compounded 182 when the optimizer needs to balance short-term gradient noise and long-term convergence, calling for 183 more ZO-specific strategies to mitigate these effects.

184 Challenge in managing extreme curvature values using Universal clipping. Furthermore, clipping 185 the Hessian to manage extreme curvature values is another widely adopted strategy. Sophia (Liu et al., 2023), for example, performs global clipping with value 1 of Hessian-based updates to ensure 187 numerical stability, which essentially can slow down the convergence. While effective at curbing 188 extreme updates, applying a universal clipping threshold across all parameters is inherently suboptimal 189 for models with heterogeneous curvatures. A universal clip might suppress meaningful gradient information in some layers while insufficiently addressing extreme Hessian values in others, thus 190 limiting the optimizer's ability to adaptively handle the diverse learning dynamics across layers (Tran 191 & Cutkosky, 2022). This approach may result in slower convergence or failure to escape saddle points 192 and local maxima, where more flexible, curvature-aware updates are required (Yao et al., 2020). 193

194 To overcome these limitations, HELENE addresses both the limitation of EMA and the issue of global 195 Hessian clipping. We instantiate MeZO-Gradient Descent, MeZO-Adam, MeZO-Newton's method, 196 MeZO-Sophia, and HELENE on a simplified 2D problem to illustrate the advantages HELENE, as shown in Figure 1. A visual comparison of the methods reveals that while MeZO-Adam and 197 MeZO-Gradient Descent struggle to converge effectively, Newton's method and Sophia find it hard 198 to maintain stability when facing heterogeneous curvature, whereas HELENE succeeds. Refer to 199 Section 5 for a more comprehensive empirical analysis, including up to $20 \times$ faster convergence and 200 improved accuracy across various tasks and datasets. 201

201 202 203

204

3.2 HELENE: <u>He</u>ssian Layer-wise Clipping and Gradient An<u>ne</u>aling

In HELENE, we introduce an annealing mechanism to mitigate bias in SPSA-estimated gradients, combined with a clipped diagonal Hessian pre-conditioner that adjusts parameter update step sizes based on layer-wise curvature. First, the gradient is calculated using the SPSA, while the diagonal Hessian is efficiently estimated by the proposed new A-GNB method, to eliminate the noise incured in sampling labels from the model output used in GNB and Sophia. At each iteration, SPSA provides an estimate g_t using two forward passes with random perturbations, and A-GNB returns h_t , the diagonal Hessian of the mini-batch loss.

We apply an exponential moving average (EMA) to both the gradient and diagonal Hessian estimates to reduce noise and improve stability. To further enhance convergence, we apply layer-wise magnitudebased clipping to the diagonal Hessian, ensuring extreme values do not disproportionately affect parameter updates. We provide our pseudo code in Algorithm 1 and each module description in the following section in details.

216 3.3 EMA OF DIAGONAL HESSIAN ESTIMATES217

When using a mini-batch to compute the local Hessian (curvature), the resulting estimates are often noisy. The Hessian diagonal can fluctuate significantly across different parameter dimensions of the problem. Inspired by the exponential moving average (EMA) of gradient moments in Adam, we apply a similar technique to reduce noise in the Hessian diagonal estimates over iterations. The updated Hessian diagonal is computed in the following:

$$\boldsymbol{h}_t = \beta_2 \boldsymbol{h}_{t-k} + (1 - \beta_2) \hat{\boldsymbol{h}}_t$$

where h_t represents the denoised Hessian diagonal at iteration t and \hat{h}_t is the current estimate of the diagonal at the k-th iteration.

Algorithm 1 HELENE with Layer-wise Clipping

228	1.	Input: Initial parameters $\mathbf{\theta}$ stap hydrat T learning rate schedule $[n]^T$ hyperparameters
229	1:	input: Initial parameters σ_1 , step budget 1, learning rate schedule $\{\eta_t\}_{t=1}$, hyperparameters $\{\lambda_t\}_{t=1}$, $\sigma_t \beta_t \beta_t \beta_t \beta_t$
230	2.	Set $m_0 = 0$ $h_0 = 0$
231	3:	for $t = 1$ to T do
232	4:	Estimate gradient g_t from $\nabla L_t(\theta_t)$ obtained from Eq. (1).
233	5:	$\alpha = \text{Anneal}(t)$
234	6:	$oldsymbol{m}_t = eta_1 oldsymbol{m}_{t-1} + lpha oldsymbol{g}_t$
235	7:	if $t \mod k = 1$ then
236	8:	Compute diagonal Hessian estimator $h_t = A$ -GNB (θ_t)
237	9:	$oldsymbol{h}_t = eta_2 oldsymbol{h}_{t-k} + (1-eta_2) \hat{oldsymbol{h}}_t$
238	10:	else
239	11:	$\boldsymbol{h}_t = \boldsymbol{h}_{t-1}$
240	12:	end if
241	13:	Apply weight decay: $\theta_t = \theta_t - \eta_t \epsilon \theta_t$
242	14:	For each layer <i>i</i> , update: $\theta_{t+1,i} = \theta_{t,i} - \eta_t \cdot \frac{m_{t,i}}{\gamma \cdot \max(h_{t,i},\lambda_i) + \epsilon}$
243	<u>15:</u>	end for
244	1:	Subroutine Anneal(t)
245 246		$\alpha \leftarrow \beta_1 + (1 - \beta_1) \cdot \exp\left(-t/T\right) $ (2)

247 248

249

223 224

225

226 227

3.3.1 ANNEALING MECHANISM

250 As illustrated in Figure 5, the native gradient EMA introduces bias, which adversely affects the training process and results in an increase in loss during the later stages. To mitigate these issues, we 251 introduce a gradient annealing mechanism to work in tandem with EMA. This adaptive adjustment 252 is crucial for ensuring that the model becomes less influenced by noisy or outdated gradients in 253 later stages. We observe that, unlike first-order methods such as SGD, Zero-order methods require 254 a stronger emphasis on momentum due to the increasing noise in SPSA gradient estimation as 255 optimization progresses, as illustrated in Appendix Figure 7. To address this, we introduce a novel 256 annealing strategy tailored for Zero-order methods, where the factor α , dynamically adjusts the 257 weight of the gradient in the momentum update. The increase in noise in SPSA is likely due to 258 the greater difficulty of training as the model parameters approach a local minimum, causing the 259 noise magnitude from sampling perturbations to exceed that of the true gradient signal. Notably, our 260 annealing approach is simple to implement, requiring the tuning of only a single hyperparameter.

261 At each iteration, the annealing mechanism computes α using an exponential decay schedule in Eq. 2, 262 where T is a predefined hyperparameter controlling the annealing rate. The increase in noise in SPSA 263 is likely due to the greater difficulty of training as the model parameters approach a local minimum, 264 causing the noise magnitude from sampling perturbations to exceed that of the true gradient signal. 265 To address this, as t increases, α gradually decreases to reduce the impact of gradient on the update, 266 mitigating the bias introduced by EMA. This ensures that, in the later stages of training, the model 267 focuses more on stable gradient estimates and less on noisy or rapidly changing updates via SPSA estimated gradient. The annealing mechanism is incorporated into the EMA update rule as line 6 in 268 Algorithm 1. Via dynamical α the annealing mechanism ensures that the optimizer can effectively 269 balance short-term noise with long-term convergence.

270 3.4 ASYMPTOTIC GAUSS-NEWTON-BARTLETT (A-GNB) ESTIMATOR 271

The original GNB (Martens, 2020) estimator relies on sampled labels \hat{y}_b drawn from the categorical distribution based on the model's output. However, this induces stochasticity due to label sampling, which could be problematic when label distributions are highly imbalanced, as is the case in large language model (LLM) training. We propose a new estimator, which we call the **Asymptotic Gauss-Newton-Bartlett (A-GNB) Estimator**, that replaces the sampled labels \hat{y}_b with the true labels y_b and asymptotically converges to the true diagonal of the Gauss-Newton matrix, which is a biased estimator for the diagonal of the Hessian as shown below:

$$\nabla_{\boldsymbol{\theta}}^2 L(\boldsymbol{\theta}) \approx J_{\boldsymbol{\theta}} f(\boldsymbol{\theta}, \boldsymbol{x}) \cdot S \cdot J_{\boldsymbol{\theta}} f(\boldsymbol{\theta}, \boldsymbol{x})^\top$$
(3)

where $J_{\theta}f(\theta, x)$ is the Jacobian of the model's output $f(\theta, x)$ with respect to the parameters θ , and $S = \frac{\partial^2 L(t,y)}{\partial t^2}$ is the second-order derivative of the loss w.r.t. the logits $t = f(\theta, x)$ and $y \sim p(y|x)$, which implies that $S = \mathbb{E}_{y \sim p(\theta, x)} \left[\frac{\partial^2 L(t,y)}{\partial t^2} \right]$ assuming that L is the Cross-Entropy loss. Consequently, the diagonal of the Gauss-Newton matrix for the mini-batch loss is estimated as:

$$diag(J_{\boldsymbol{\theta}}f(\boldsymbol{\theta},\boldsymbol{x})\cdot S\cdot J_{\boldsymbol{\theta}}f(\boldsymbol{\theta},\boldsymbol{x})^{\top}]) = \mathbb{E}_{y\sim p(y|\boldsymbol{x})} \left[diag(J_{\boldsymbol{\theta}}f(\boldsymbol{\theta},\boldsymbol{x})\frac{\partial L(t,y)}{\partial t}\frac{\partial L(t,y)}{\partial t}^{\top}J_{\boldsymbol{\theta}}f(\boldsymbol{\theta},\boldsymbol{x})^{\top}) \right]$$
$$= \mathbb{E}_{y\sim p(y|\boldsymbol{x})} \left[diag(\nabla_{\boldsymbol{\theta}}L(f(\boldsymbol{\theta},\boldsymbol{x}),y)\nabla_{\boldsymbol{\theta}}L(f(\boldsymbol{\theta},\boldsymbol{x}),y)^{\top}) \right]$$
$$\approx \frac{1}{B} \sum_{b=1}^{B} [(g(\boldsymbol{\theta},\boldsymbol{x}_{b},y_{b})] \odot [(g(\boldsymbol{\theta},\boldsymbol{x}_{b},y_{b})]$$

where $diag(\cdot)$ represents the diagonal elements of a matrix, and B denotes the batch size and g is the estimated gradient from Eq. (1). In contrast to GNB estimator, which includes sampling label \hat{y} from the logit probability output from the model, we replace it by y_b , the true label, thereby avoiding the need for post-output label sampling. By eliminating the stochasticity induced by sampled labels \hat{y} , we reduce the variance caused by sampling noise, and it is especially beneficial in imbalanced data scenarios, when samples from minor class is rarely selected unless sampling significantly many times.

The estimated gradient terms now correspond directly to the true labels, and their outer product sums up to the true Gauss-Newton approximation of the Hessian. As the batch size $B \to \infty$, the A-GNB estimator converges to the true Hessian's diagonal:

303 304 305

306

307

308

317

279 280

281

282

283 284

285

287

294

295

296

297

298

299

 $\lim_{B\to\infty}\frac{1}{B}\sum_{b=1}^{B}[g(\boldsymbol{\theta},\boldsymbol{x}_{b},y_{b})]\odot[g(\boldsymbol{\theta},\boldsymbol{x}_{b},y_{b})]=diag(J_{\boldsymbol{\theta}}f(\boldsymbol{\theta},\boldsymbol{x})\cdot S\cdot J_{\boldsymbol{\theta}}f(\boldsymbol{\theta},\boldsymbol{x})^{\top})$

Therefore, The A-GNB estimator asymptotically converges to the true diagonal of the Gauss-Newton matrix as *B* increases.

1:]	Parameters: θ	
2: 1	Draw a mini-batch of the input $\{x_b\}_{b=1}^B$	
3: 1	Estimate diagnal Hessian matrix by $h = \sum_{b=1}^{B} [g(\theta, x_b, y_b)] \odot [g(\theta, x_b, y_b)]$	
4: 1	return h	

3.5 LAYWERWISE CLIPPED DIAGONAL HESSIAN TO HELP NEWTON'S METHOD

As discussed in the motivating examples, fine-tuning LLMs and optimizing non-convex functions pose challenges for Newton's method, which uses the Hessian as a pre-conditioner. The method may converge to local maxima rather than local minima. Moreover, the inaccuracy of Hessian estimates and changes in the Hessian along the optimization trajectory can render second-order information unreliable. To address these issues, we draw inspiration from Sophia. While Sophia performs clipping on the Newton update $H^{-1}g$, we propose a more robust approach by applying layer-wise clipping directly to the Hessian matrix H using a universal clipping threshold for the update $H^{-1}g$ disregards the differences in layer-wise Hessian and gradient magnitudes, which are frequently observed during DNN training, and may distort valuable gradient information. Moreover, $H^{-1}g$ introduces excessive bias, potentially distorting useful gradient information, whereas clipping extreme Hessian values more effectively preserves essential second-order information.

In particular, we improve convergence rates by (1) considering only the positive entries of the diagonal Hessian and (2) introducing per-coordinate, layer-wise clipping of the Hessian values. This approach adapts the clipping threshold across layers to account for the diverse curvature across different parts of the model. Given a clipping threshold $\lambda_i > 0$ for layer *i*, the clipping function is defined as:

 $\operatorname{clip}(\boldsymbol{h}_i) = \max(\boldsymbol{h}_i, \lambda_i), \quad \lambda_i \in \mathbb{R},$

where all operations are applied element-wise for each layer. The update rule for layer i is then written as:

333

337

 $oldsymbol{ heta}_{t+1,i} = oldsymbol{ heta}_{t,i} - \eta \cdot rac{oldsymbol{m}_{t,i}}{\gamma \cdot \max(oldsymbol{h}_{t,i},\lambda_i) + \epsilon},$

where $\epsilon > 0$ is a small constant to avoid division by zero, and λ_i controls the fraction of clipped Hessian values per layer. By applying layer-wise clipping, we ensure that the optimizer is capable of adapting to the curvature of each layer individually, leading to improved stability and convergence rates across different parts of the model. We present the pseudo-code of our Hessian-clipped method in Algorithm 1.

For further information about the differences of HELENE with previous work, please reference the related work in Appendix A.

345 346

347

358

359

360 361

362

364 365

366

367 368

369 370

4 CONVERGENCE ANALYSIS

In this section, we provide a theoretical analysis of the convergence of our proposed method. The key improvement in our method comes from the use of layer-wise parameters λ_i , which reduces the dependency on the total dimension d and instead relies on the maximum layer dimension $\max_i d_i$.

The theoretical bound for the number of steps T in our method is given by the following theorem with two assumptions:

Assumption 1. Let $L : \mathbb{R}^d \to \mathbb{R}$ be a loss function. We assume L is twice continuously differentiable strictly convex, and has a unique minimizer denoted by θ^* . For each layer *i*, define μ_i as the minimum eigenvalue of the Hessian matrix of L concerning the parameters of that layer evaluated at its minimizer: $\mu_i = 1$ ($\nabla^2 L(\theta^*)$)

 $\mu_i \equiv \lambda_{\min}(\nabla^2_{\theta_i} L(\theta^*))$

where $\nabla^2_{\theta_i}$ denotes the Hessian with respect to the parameters of the *i*-th layer.

Assumption 2. Regarding the Hessian $\nabla^2 L(\theta)$ of the loss function, we assume:

• There exists a radius $R_i > 0$ such that for any $\theta_i, \theta'_i \in \mathbb{R}^d$ with $\|\theta_i - \theta'_i\|_2 \leq R_i$, the following inequality holds:

$$\left\|\nabla^{2} L(\boldsymbol{\theta}_{i}^{\prime} \mid \boldsymbol{\theta}_{-i})^{-1} \nabla^{2} L(\boldsymbol{\theta}_{i} \mid \boldsymbol{\theta}_{-i})\right\|_{2} \leq 2$$

where $\|\cdot\|_2$ represents the spectral norm.

Theorem 1. Under Assumptions 1 and 2, let $\eta = \frac{1}{2}$ and $\lambda_i = \frac{R_i}{2\sqrt{d_i}}$. The update reaches a loss at most ϵ in

$$T \le \max_{i} \left\{ d_{i} \cdot \left(L(\boldsymbol{\theta}_{0,i}) - \min L \right) + \ln \left(\frac{\mu_{i} R_{i}^{2}}{32d_{i} \epsilon} \right) \right\}.$$

steps, where *L* is the loss function, $\theta_{0,i}$ is the initial parameter vector for layer *i*, μ_i is the strong convexity constant for layer *i*, and R_i is the bound on the distance between $\theta_{0,i}$ and θ_i^* .

The best known theoretical bound for the number of steps T required by Sophia to reach a loss at most ϵ is given by Sophia in which $T_{\text{SOPHIA}} \sim \mathcal{O}(d)$, where d is the total dimension of the parameter space. This result implies that the convergence rate depends linearly on the total dimension d, which can lead to slow convergence for models with large parameter spaces. In contrast, our method introduces layerwise parameters $\rho_i = \frac{R_i}{2\sqrt{d_i}}$, where R_i is the bound on the distance between the initial parameters $\theta_{0,i}$ 378 and the optimal parameters θ_i^* for layer i, and d_i is the dimension of the parameter space for layer i. 379 This layer-wise setting significantly reduces the complexity to $T_{\text{SOPHIA}} \sim \mathcal{O}(\max_i d_i)$, which is the 380 maximum dimension across layers. Besides the lower runtime bound, our method allow each layer to 381 have its own parameter ρ_i , allowing the method to adapt to the specific geometry of each layer. Refer 382 to Appendix B.3 for the empirical study on the significant variance using unified parameter clipping across different layers. This flexibility leads to a more efficient optimization process, as each layer 383 is treated independently based on its characteristics. In large models where some layers have much 384 smaller dimensions than others, our method is able to achieve faster convergence by focusing on the 385 most difficult layer with the largest dimension, therefore making our method more scalable for deep 386 models with many layers. Detailed proof can be seen in the Appendix C. 387

388 389

390

5 EXPERIMENTS

Since the introduction of the Transformer (Vaswani, 2017), language models (LMs) have progressively
 developed through the use of different Transformer-based architectures. One of the iconic work is
 BERT (Devlin, 2018), which is based on the encoder architecture of Transformer and pre-trained
 with techniques like masked language modeling. As the field of natural language processing (NLP)
 develops, more powerful decoder-only LLMs also have shown their great potential.

396 Therefore, to rigorously evaluate the capability and universality of HELENE, we follow the ex-397 periments conducted in MeZO on both medium-sized masked LMs (RoBERTa-large (Liu, 2019), 398 350M) and auto-regressive LLMs (OPT-1.3B (Zhang et al., 2023)) under both few-shot and many-399 shot settings. Additionally, all optimization algorithms are evaluated with three tuning methods: 400 fine-tuning (FT) and two parameter-efficient fine-tuning (PEFT) methods, LoRA (Hu et al., 2021) 401 and prefix-tuning (Li & Liang, 2021). We also do experiments with zeroth-order (ZO) versions of some optimizers as well as ZO-SGD variants introduced in Zhang et al. (2024), and present them in 402 Section 5.3. 403

The experimental results show that across all settings, HELENE not only outperforms MeZO on most datasets by approximately 1.5% on average, but also makes the convergence process of gradient-free optimization more stable and faster, boosting to $20 \times$ times the original speed.

407 408 409

5.1 MASKED LANGUAGE MODELS

For masked LMs, we conduct experiments using RoBERTa-large on three types of NLP tasks, sentiment classification, natural language inference, and topic classification with k = 16 examples per class. We run HELENE for 5,000 steps and FT for 1,000 steps. The experimental results are listed in Table 1.

Task Type	SST-2	SST-5	SNLI	MNLI	RTE	TRE
	—— senti	ment ——	—— natura	l language inf	ference ——	— topi
Zero-shot	79.0	35.5	50.2	48.8	51.4	32.0
LP	76.0 (±2.8)	40.3 (±1.9)	66.0 (±2.7)	56.5 (±2.5)	59.4 (±5.3)	51.3 (±
FT	91.9 (±1.8)	46.7 (±1.9)	77.5 (±2.6)	70.0 (±2.3)	66.4 (±7.2)	85.0 (±
FT(LoRA)	91.4 (±1.7)	46.7 (±1.1)	74.9 (±4.3)	67.7 (±1.4)	66.1 (±3.5)	82.7 (±
FT(Prefix)	91.9 (±1.0)	47.7 (±1.1)	77.2 (±1.3)	66.5 (±2.5)	66.6 (±2.0)	85.7 (±
MeZO	90.5 (±1.2)	45.5 (±2.0)	68.5 (±3.9)	58.7 (±2.5)	64.0 (±3.3)	76.9 (±
MeZO (LoRA)	91.4 (±0.9)	43.0 (±1.6)	69.7 (±6.0)	64.0 (±2.5)	64.9 (±3.6)	73.1 (±
MeZO (Prefix)	90.8 (±1.7)	45.8 (±2.0)	71.6 (±2.5)	63.4 (±1.8)	65.4 (±3.9)	80.3 (±
HELENE	92.6 (±2.3)	46.7 (±0.8)	72.0 (±2.6)	58.9 (±1.1)	65.7 (±1.2)	78.1 (±
HELENE (LoRA)	90.6 (±0.3)	41.8 (±1.0)	68.5 (±2.0)	59.0 (±1.1)	66.8 (±3.2)	67.4 (±
HELENE (Prefix)	91.7 (±0.6)	46.0 (±0.7)	69.5 (±1.9)	64.6 (±2.1)	66.1 (±1.8)	77.4 (±

429

Table 1: Experiments on RoBERTa-large (350M parameters, k = 16). PEFT represents the LoRA and prefix and we report the best of them. All reported numbers are averaged accuracy (standard deviation) across 5 runs.

Task Task type	SST-2	RTE	CB — cla	BoolQ assificati	WSC on ——	WIC	COPA — multi	ReCoRD ple choice —	SQuAD – generation -
Zero-shot	53.4	53.1	37.5	45.7	44.2	57.0	75.0	70.3	27.1
ICL	80.3	53.1	48.2	58.5	44.2	50.6	69.0	71.0	59.0
LP	80.3	52.7	44.6	58.9	47.1	50.6	69.0	71	75.9
MeZO	89.6	55.8	77.0	59.6	55.0	58.0	74.0	60.0	62.2
MeZO (LoRA)	90.8	63.0	78.0	67.2	51.2	58.0	79.0	59.8	67.6
MeZO (prefix)	92.4	52.8	66.0	61.6	51.6	52.8	74.0	56.8	56.0
HELENE	91.2	64.4	87.0	60.8	55.4	58.4	69.0	55.6	63.8
HELENE (LoRA)	91.4	50.6	76	64.0	49.6	52.6	82.0	60.2	60.4
HELENE (prefix)	92.4	51.6	74.0	62.5	52	57.2	80.0	58.8	68.4
FT (12× memory)	90.8	73.4	77	70.2	53	60.2	81.0	59.6	70.9

Table 2: Experiments on OPT-1.3B (with 1000 examples). ICL: in-context learning; LP: linear probing; FT: full-parameter fine-tuning with Adam. We highlight the best results in bold to facilitate comparison.



Figure 3: Performance and convergence of MeZO and HELENE for fine-tuning, LoRA, and prefixtuning of OPT-1.3B on different datasets. HELENE achieves approximate $10 \times$ speedup and up to 15% accuracy improvement compared to MeZO.

HELENE largely outperforms zero-shot and linear probing. On all six datasets, HELENE can stably optimize the pre-trained LM and consistently perform better than zero-shot and linear probing.

HELENE delivers a 20× speed improvement over MeZO while also maintaining its performance. With the guidance of layer-wise clipped Hessian information, HELENE can reach conver-464 gence in about 5000 steps on average across the datasets, accelerating the optimization process by approximate $20 \times$ times than MeZO. Meanwhile, the results show that HELENE can still achieve performance on par with MeZO, with leading average accuracy of three tuning methods on the dataset SST-2, SST-5, MNLI and RTE.

469 5.2 AUTO-REGRESSIVE LLMS 470

444

445

446

451 452 453

454 455

456

457

458 459 460

461

462

463

465

466

467

468

471 LLMs such as GPT-3 (Brown et al., 2020), LLaMA (Touvron et al., 2023), and ChatGLM (Du et al., 2021) have become the predominant models in NLP, we include experiments with auto-regressive 472 LLM OPT-1.3B on three different task: text classification, multiple choice, and text generation. 473 We use various datasets from the SuperGLUE benchmark (Wang et al., 2019), which includes the 474 following datasets: SST-2, RTE, CB, WSC, WIC, COPA, and ReCoRD. Additionally, we also 475 experiment on BoolQ (Clark et al., 2019) and SQuAD (Rajpurkar, 2016). We run HELENE with 476 about 10,000 training steps for each dataset. The results are summarized in Table 2, from which we 477 can have the following observations. 478

HELENE has clear performance advantages compared with MeZO. Table 2 shows that HELENE 479 with its LoRA and prefix variants can consistently outperform MeZO. Specifically, the average 480 performances of HELENE with its LoRA and prefix variants remarkably exceed MeZO's by 5.3%, 481 2.1% and 1.3% on CB, SQuAD and COPA, respectively. 482

HELENE accelerates $10 \times$ times while remaining compatible with PEFT methods. In Figure 3, 483 we present results from four selected datasets across different tasks under three tuning methods. It 484 indicates that HELENE can consistently speed up the convergence by up to $10 \times$ times, and also 485 enhances the capability.

SST2	R	oberta-La	OPT-1.3		
5512	FT	LoRA	Prefix	FT	LoRA
O-SGD	91.4	91.2	89.6	91.1	93.6
Forward-Grad	90.1	89.7	89.5	90.3	90.3
ZO-SGD	89.4	90.8	90.0	90.8	90.1
ZO-SGD-MMT	89.6	90.9	90.1	85.2	91.3
ZO-SGD-Cons	89.6	91.6	90.1	88.3	90.5
ZO-SGD-Sign	52.5	90.2	53.6	87.2	91.5
ZO-Adam	89.8	89.5	90.2	84.4	92.3
HELENE	92.6	90.6	91.7	90.8	91.4



Table 3: Performance of LLM fine-tuning on SST2 over pre-trained Roberta-Large and OPT-1.3B. Best performance among ZO methods (including Forward-Grad) are in **bold**.



5.3 **EXPERIMENTS WITH OTHER ZO ALGORITHMS**

It is worth noting that the ZO optimization technique utilized in Malladi et al. (2023) is primarily the basic SGD version (ZO-SGD), and it is still not clear how effective HELENE is when comparing with other ZO optimization algorithms like ZO-SGD, ZO-SGD-MMT, ZO-SGD-Cons, ZO-SGD-Sign and ZO-Adam as introduced in Liu et al. (2020). Therefore, we reference the statistics of performances summarized in Zhang et al. (2024) and experiment under the same setting with them (Table 3), through which HELENE shows good functionality especially for FT and prefix-tuning.

We further implement the ZO versions of Adam, AdamW and Lion (Chen et al., 2024) and plot the results in Figure 4. The results indicate that HELENE helps the model converge faster as well as obtain lower validation loss value.

5.4 ABLATION STUDY

We conduct a comprehensive ablation study on the key techniques of HELENE in Appendix B, including in-depth analysis of the effects of magnitude clipping across different ranges. Additionally, we explore the factors resulting in Sophia's initial convergence and subsequent divergence.

CONCLUSION

In this paper, we present a novel optimizer, HELENE, which is designed to address the challenges of fine-tuning LLMs. HELENE integrates a new asymptotic Gauss-Newton-Bartlett (A-GNB) estimator for diagonal Hessian estimation, and a novel layer-wise clipping with the annealing module. The A-GNB estimator eliminates the need for label sampling, providing an unbiased Hessian approximation and improving the precision of curvature-aware updates. Furthermore, our layer-wise clipping mechanism provably ensures more adaptive Hessian updates based on the curvature of each layer, enhancing stability and scalability. Theoretical analysis shows that HELENE reduces convergence steps from $\mathcal{O}(d)$ to $\mathcal{O}(\max_i d_i)$, making it highly scalable for modern architectures with many layers. Experimental results on models like RoBERTa-large and OPT-1.3B demonstrate that HELENE achieves up to a $20 \times$ speedup compared to MeZO and improves performance by 1.5% on average. Compatible with both full parameter tuning and parameter-efficient fine-tuning, HELENE outperforms many state-of-the-art optimizers across diverse tasks and datasets.

540 REFERENCES

550

571

578

579

580 581

582

583

586

587 588

589

590

- Rohan Anil, Vineet Gupta, Tomer Koren, Kevin Regan, and Yoram Singer. Scalable second order
 optimization for deep learning. *arXiv preprint arXiv:2002.09018*, 2020.
- Jimmy Ba, Roger B Grosse, and James Martens. Distributed second-order optimization using kronecker-factored approximations. In *ICLR (Poster)*, 2017.
- 547 S BECKER. Improving the convergence of backpropagation learning with second order method. In
 548 *Proceedings of the 1988 Connectionist Models Summer School, San Mateo, CA*. Morgan Kaufmann,
 549 1988.
- Raghu Bollapragada, Richard H Byrd, and Jorge Nocedal. Exact and inexact subsampled newton methods for optimization. *IMA Journal of Numerical Analysis*, 39(2):545–578, 2019.
- Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal,
 Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models
 are few-shot learners. In *Advances in neural information processing systems*, volume 33, pp. 1877–1901, 2020.
- C. G. BROYDEN. The Convergence of a Class of Double-rank Minimization Algorithms 1. General Considerations. *IMA Journal of Applied Mathematics*, 6(1):76–90, 03 1970. ISSN 0272-4960. doi: 10.1093/imamat/6.1.76. URL https://doi.org/10.1093/imamat/6.1.76.
- HanQin Cai, Yuchen Lou, Daniel McKenzie, and Wotao Yin. A zeroth-order block coordinate
 descent algorithm for huge-scale black-box optimization. In *International Conference on Machine Learning*, pp. 1193–1203. PMLR, 2021.
- Pin-Yu Chen, Huan Zhang, Yash Sharma, Jinfeng Yi, and Cho-Jui 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*, pp. 15–26, 2017.
- Xiangning Chen, Chen Liang, Da Huang, Esteban Real, Kaiyuan Wang, Hieu Pham, Xuanyi Dong, Thang Luong, Cho-Jui Hsieh, Yifeng Lu, et al. Symbolic discovery of optimization algorithms. *Advances in neural information processing systems*, 36, 2024.
- Xiangyi Chen, Steven Z Wu, and Mingyi Hong. Understanding gradient clipping in private sgd: A
 geometric perspective. *Advances in Neural Information Processing Systems*, 33:13773–13782, 2020.
- 575
 576
 576
 576
 577
 Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. Boolq: Exploring the surprising difficulty of natural yes/no questions. *arXiv preprint arXiv:1905.10044*, 2019.
 - Andrew R. Conn, Nicholas I. M. Gould, and Philippe L. Toint. *Trust-Region Methods*. Society for Industrial and Applied Mathematics, USA, 2000. ISBN 0898714605.
 - Ron S Dembo, Stanley C Eisenstat, and Trond Steihaug. Inexact newton methods. *SIAM Journal on Numerical analysis*, 19(2):400–408, 1982.
- Jacob Devlin. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.
 - P Kingma Diederik. Adam: A method for stochastic optimization. (No Title), 2014.
 - Zhengxiao Du, Yujie Qian, Xiao Liu, Ming Ding, Jiezhong Qiu, Zhilin Yang, and Jie Tang. Glm: General language model pretraining with autoregressive blank infilling. *arXiv preprint arXiv:2103.10360*, 2021.
- Tanmay Gautam, Youngsuk Park, Hao Zhou, Parameswaran Raman, and Wooseok Ha. Variance reduced zeroth-order methods for fine-tuning language models. *arXiv preprint arXiv:2404.08080*, 2024.

- 594 Thomas George, César Laurent, Xavier Bouthillier, Nicolas Ballas, and Pascal Vincent. Fast 595 approximate natural gradient descent in a kronecker factored eigenbasis. Advances in Neural 596 Information Processing Systems, 31, 2018. 597 Behrooz Ghorbani, Shankar Krishnan, and Ying Xiao. An investigation into neural net optimization 598 via hessian eigenvalue density. In International Conference on Machine Learning, pp. 2232–2241. PMLR, 2019. 600 601 Vineet Gupta, Tomer Koren, and Yoram Singer. Shampoo: Preconditioned stochastic tensor optimiza-602 tion. In International Conference on Machine Learning, pp. 1842–1850. PMLR, 2018. 603 Davood Hajinezhad and Michael M Zavlanos. Gradient-free multi-agent nonconvex nonsmooth 604 optimization. In 2018 IEEE Conference on Decision and Control (CDC), pp. 4939–4944. IEEE, 605 2018.606 607 Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, 608 and Weizhu Chen. Lora: Low-rank adaptation of large language models. arXiv preprint arXiv:2106.09685, 2021. 609 610 Frank Hutter and Ilya Loshchilov. Decoupled weight decay regularization. arXiv preprint arXiv: 611 1711.05101, 2017. 612 613 Majid Jahani, Sergey Rusakov, Zheng Shi, Peter Richtárik, Michael W Mahoney, and Martin Takáč. Doubly adaptive scaled algorithm for machine learning using second-order information. arXiv 614 preprint arXiv:2109.05198, 2021. 615 616 Shuoran Jiang, Qingcai Chen, Youcheng Pan, Yang Xiang, Yukang Lin, Xiangping Wu, Chuanyi Liu, 617 and Xiaobao Song. Zo-adamu optimizer: Adapting perturbation by the momentum and uncertainty 618 in zeroth-order optimization. In Proceedings of the AAAI Conference on Artificial Intelligence, 619 volume 38, pp. 18363-18371, 2024. 620 Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. arXiv 621 preprint arXiv:2101.00190, 2021. 622 623 Hong Liu, Zhiyuan Li, David Hall, Percy Liang, and Tengyu Ma. Sophia: A scalable stochastic 624 second-order optimizer for language model pre-training. arXiv preprint arXiv:2305.14342, 2023. 625 Mingrui Liu, Zhenxun Zhuang, Yunwen Lei, and Chunyang Liao. A communication-efficient 626 distributed gradient clipping algorithm for training deep neural networks. Advances in Neural 627 Information Processing Systems, 35:26204–26217, 2022. 628 629 Sijia Liu, Pin-Yu Chen, Xiangyi Chen, and Mingyi Hong. signsgd via zeroth-order oracle. In 630 International Conference on Learning Representations, 2019. 631 Sijia Liu, Pin-Yu Chen, Bhavya Kailkhura, Gaoyuan Zhang, Alfred O Hero III, and Pramod K 632 Varshney. A primer on zeroth-order optimization in signal processing and machine learning: 633 Principals, recent advances, and applications. IEEE Signal Processing Magazine, 37(5):43-54, 634 2020. 635 Yinhan Liu. Roberta: A robustly optimized bert pretraining approach. arXiv preprint 636 arXiv:1907.11692, 2019. 637 638 Sadhika Malladi, Tianyu Gao, Eshaan Nichani, Alex Damian, Jason D Lee, Danqi Chen, and Sanjeev 639 Arora. Fine-tuning language models with just forward passes. Advances in Neural Information 640 Processing Systems, 36:53038–53075, 2023. 641 James Martens. New insights and perspectives on the natural gradient method. Journal of Machine 642 Learning Research, 21(146):1-76, 2020. 643 644 James Martens and Roger Grosse. Optimizing neural networks with kronecker-factored approximate 645 curvature. In International conference on machine learning, pp. 2408–2417. PMLR, 2015. 646 James Martens, Jimmy Ba, and Matt Johnson. Kronecker-factored curvature approximations for
- ⁶⁴⁷ James Martens, Jimmy Ba, and Matt Johnson. Kronecker-factored curvature approximations for recurrent neural networks. In *International Conference on Learning Representations*, 2018.

648 649 650	James Martens et al. Deep learning via hessian-free optimization. In <i>Icml</i> , volume 27, pp. 735–742, 2010.
651 652 653	John L Maryak and Daniel C Chin. Global random optimization by simultaneous perturbation stochastic approximation. In <i>Proceedings of the 2001 American control conference.(Cat. No. 01CH37148)</i> , volume 2, pp. 756–762. IEEE, 2001.
654 655	Yurii Nesterov and B. T. Polyak. Cubic regularization of newton method and its global performance. <i>Math. Program.</i> , 108(1):177–205, aug 2006. ISSN 0025-5610.
657	R Pascanu. Revisiting natural gradient for deep networks. arXiv preprint arXiv:1301.3584, 2013.
658 659	P Rajpurkar. Squad: 100,000+ questions for machine comprehension of text. <i>arXiv preprint arXiv:1606.05250</i> , 2016.
661 662	Herbert Robbins and Sutton Monro. A stochastic approximation method. <i>The annals of mathematical statistics</i> , pp. 400–407, 1951.
663 664	Levent Sagun, Leon Bottou, and Yann LeCun. Eigenvalues of the hessian in deep learning: Singularity and beyond. <i>arXiv preprint arXiv:1611.07476</i> , 2016.
666 667	Tom Schaul, Sixin Zhang, and Yann LeCun. No more pesky learning rates. In <i>International conference on machine learning</i> , pp. 343–351. PMLR, 2013.
668 669	Nicol N Schraudolph. Fast curvature matrix-vector products for second-order gradient descent. <i>Neural computation</i> , 14(7):1723–1738, 2002.
670 671 672	James C Spall. Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. <i>IEEE transactions on automatic control</i> , 37(3):332–341, 1992.
673 674	Yujie Tang, Junshan Zhang, and Na Li. Distributed zero-order algorithms for nonconvex multiagent optimization. <i>IEEE Transactions on Control of Network Systems</i> , 8(1):269–281, 2020.
675 676 677 678	Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. <i>arXiv preprint arXiv:2307.09288</i> , 2023.
679 680	Hoang Tran and Ashok Cutkosky. Better sgd using second-order momentum. <i>Advances in Neural</i> <i>Information Processing Systems</i> , 35:3530–3541, 2022.
681 682	A Vaswani. Attention is all you need. Advances in Neural Information Processing Systems, 2017.
683 684 685	Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. Superglue: A stickier benchmark for general-purpose language understanding systems. <i>Advances in neural information processing systems</i> , 32, 2019.
686 687 688	Colin Wei, Sham Kakade, and Tengyu Ma. The implicit and explicit regularization effects of dropout. In <i>International conference on machine learning</i> , pp. 10181–10192. PMLR, 2020.
689 690	Peng Xu, Fred Roosta, and Michael W Mahoney. Newton-type methods for non-convex optimization under inexact hessian information. <i>Mathematical Programming</i> , 184(1):35–70, 2020.
691 692 693	Zhewei Yao, Amir Gholami, Kurt Keutzer, and Michael W Mahoney. Pyhessian: Neural networks through the lens of the hessian. In <i>2020 IEEE international conference on big data (Big data)</i> , pp. 581–590. IEEE, 2020.
695 696	Haishan Ye, Zhichao Huang, Cong Fang, Chris Junchi Li, and Tong Zhang. Hessian-aware zeroth- order optimization for black-box adversarial attack. <i>arXiv preprint arXiv:1812.11377</i> , 2018.
697 698 699	Jingzhao Zhang, Tianxing He, Suvrit Sra, and Ali Jadbabaie. Why gradient clipping accelerates training: A theoretical justification for adaptivity. <i>arXiv preprint arXiv:1905.11881</i> , 2019.
700 701	Lin Zhang, Shaohuai Shi, and Bo Li. Eva: Practical second-order optimization with kronecker- vectorized approximation. In <i>The Eleventh International Conference on Learning Representations</i> , 2022.

702 703 704	Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. Opt: Open pre-trained transformer language models, 2022. URL https://arxiv.org/abs/2205.01068, 3:19–0, 2023.
705 706 707	Yihua Zhang, Pingzhi Li, Junyuan Hong, Jiaxiang Li, Yimeng Zhang, Wenqing Zheng, Pin-Yu Chen, Jason D Lee, Wotao Yin, Mingyi Hong, et al. Revisiting zeroth-order optimization for
708	memory-efficient llm fine-tuning: A benchmark. arXiv preprint arXiv:2402.11592, 2024.
709	Yaniun Zhao, Sizhe Dang, Haishan Ye, Guang Dai, Yi Oian, and Ivor W Tsang. Second-order
710	fine-tuning without pain for llms: A hessian informed zeroth-order optimizer. arXiv preprint
711	arXiv:2402.15173, 2024.
712	
713	
714	
715	
/16	
710	
710	
720	
721	
722	
723	
724	
725	
726	
727	
728	
729	
730	
731	
732	
733	
734	
730	
737	
738	
739	
740	
741	
742	
743	
744	
745	
746	
747	
748	
749	
750	
751	
752	
754	
755	

756 A RELATED WORK

758 759

760

A.1 ZERO-ORDER OPTIMIZATION

761 Zeroth-order (ZO) optimization, which only relies on the forward passes of neural networks, offers 762 significant memory savings during the training process. Recently, MeZO (Malladi et al., 2023) adapted 763 the traditional zeroth-order SGD optimization method for fine-tuning LMs, achieving performance 764 comparable to full-parameter fine-tuning while significantly reducing memory usage. Thus, zeroth-765 order optimization is regarded as a promising approach for memory-efficient fine-tuning of LLMs. 766 Several studies have aimed to improve the MeZO algorithm. For instance, Gautam et al. (2024) 767 introduced a zeroth-order optimization algorithm that integrates both full-batch and mini-batch 768 information to produce asymptotically unbiased, low-variance gradient estimations. However, the convergence rate of their approach still leaves room for improvement. In pursuit of better gradient 769 estimation, Jiang et al. (2024) proposed an innovative perturbation sampling technique inspired by 770 the Adam optimizer. Other methods, such as SPSA (Spall, 1992; Maryak & Chin, 2001), have shown 771 to be effective in non-convex multi-agent optimization (Tang et al., 2020; Hajinezhad & Zavlanos, 772 2018) and in generating black-box adversarial examples (Chen et al., 2017; Cai et al., 2021; Liu et al., 773 2019; Ye et al., 2018).

- 774 775
- 776 777

778

A.2 SECOND-ORDER INFORMATION FOR FINE-TUNING LLMS

779 Classic second-order optimization algorithms pre-condition the gradient with curvature information (BROYDEN, 1970; Nesterov & Polyak, 2006; Conn et al., 2000). Over the years, people have 781 developed numerous ways to adapt these methods to deep learning. To the best of our knowledge, 782 BECKER (1988) was the first to use diagonal Hessian as the pre-conditioner. Martens et al. (2010) 783 approximated the Hessian with conjugate gradient. Schaul et al. (2013) automatically tuned the 784 learning rate of SGD by considering diagonal Hessian. Pascanu (2013) considered Gaussian Newton's 785 approximation of Hessian and Fisher information matrix. Martens & Grosse (2015) and follow-up works (Ba et al., 2017; George et al., 2018; Martens et al., 2018; Zhang et al., 2022) proposed to 786 approximate the Hessian based on the structure of neural networks. Despite these progress on deep 787 learning applications, for decoder-only LLMs, Adam still appears to be the most popular optimizer. 788 The authors of this paper suspect that many previous second-order optimizers face the challenge that 789 the computational / memory overhead due to frequent Hessian computation hinders improvements in 790 wall-clock time (Martens & Grosse, 2015; Gupta et al., 2018). Some of them also depend on specific 791 model architecture or hardware structures, e.g., Anil et al. (2020) offloads hessian computation to 792 CPUs, and George et al. (2018) needs ResNet and very large batch size to approximate the Fisher 793 information matrix. To the best of our knowledge, there was no previous report that second-order 794 optimizers can achieve a speed-up on LLMs in total compute.

There is also a concurrent work HiZOO (Zhao et al., 2024) that utilizes Hessian information to enhance zeroth-order optimization for fine-tuning LLMs. A major focus of HiZOO is to introduce one more forward pass to handle heterogeneous curvatures across parameter dimensions. However, our work focus on incorporating layer-wise clipping to exclude extreme Hessian values and Exponential Moving Average (EMA) to improve generalization.

- 800 801
- 802

A.3 GRADIENT CLIPPING

803 804

Global gradient clipping has been a widely adopted practice in fine-tuning LLMs (Chen et al., 2020;
Zhang et al., 2019; Liu et al., 2022). This technique stabilizes training by mitigating the impact of
rare examples and large gradient noise. In addition to gradient clipping, HELENE is the first method
to clip the Hessian matrix in second-order optimization techniques. This approach addresses the issue
of the Hessian matrix fluctuating along the optimization trajectory and reduces the errors in Hessian
approximations.



(a) Ablation study for key components of HELENE. The effectiveness of annealing and the clipped diagonal Hessian are demonstrated progressively from high to low.

(b) The image on the right is a zoomed-in view of the left one, showing that our proposed HELENE is twice as fast as the compared methods.

Figure 5: Comparison of tuning processes and ablation studies with different optimization algorithms.

B ABLATION STUDY

B.1 EVALUATING THE IMPACT OF KEY COMPONENTS ON CONVERGENCE AND STABILITY

Figure 5 illustrates the effectiveness of each component in our algorithm. Adding momentum to MeZO alone doesn't improve performance. Introducing bias in the gradient boosts convergence speed, but causes loss to increase later in training due to biased gradient estimates. To counter this, we added an annealing term to make the gradient asymptotically unbiased, which stabilizes the loss. Inspired by Sophia, we introduced the clipped Hessian to address heterogeneous curvatures, further improving convergence speed. Our ablation study validates both the motivation and effectiveness of these components.

823

824

825

826

827 828 829

830 831

832 833

835

836

837

838

B.2 MAGNITUDE CLIPPING

Figure 6 addresses the robustness of clipping in our optimizer. Our empirical study is as follows:
First, we explored the impact of lower bounds ranging from 1 to 3, all of which demonstrated stability.
As a hyperparameter, this lower bound shows consistent robustness. However, when the lower bound
was set to 0.9, the model performance dropped by 10 points, leading us to believe that problematic
Hessian values are concentrated below 1, while values above 1 are less critical. Second, we argue
that layer-wise clipping based on magnitude is reasonable in a zeroth-order setting, as performing
percentage-based clipping for each layer would be time-consuming.

850 851

852

B.3 INVESTIGATION INTO THE CONVERGENCE INSTABILITY OF SOPHIA

We study the reasons for Sophia's failure in the Figure 1 by counting the number of clip triggers. We computed the loss between timesteps 400 and 800, with a mean value of 0.57. The average loss between timesteps 1400 and 1800 was 0.65. We then analyzed the number of times the Sophia clipping mechanism was triggered within these two time intervals. Our analysis covered the Q, K, We natrices, fully connected layers, and bias layers. We found that the frequency of clipping in the interval where the mean loss was 0.65 was 1.18 to 1.22 times higher than in the interval where the mean loss was 0.57.

Based on these experimental observations, we conclude that Sophia's clipping mechanism tends to be over-triggered in complex data scenarios, particularly when faced with heterogeneous curvature. This over-triggering can result in non-convergence, aligning with our intuition. In the zeroth-order setting, gradients are estimated using SPSA, and excessive clipping of the $\frac{g}{H}$ terms can lead to instability and failure of the model to converge.



911 912

C DETAILED CONVERGENCE ANALYSIS

913 914

915 **Lemma 1** (Divergence to Infinity). Under Assumption 1, for each layer *i* in a neural network model, 916 assume the function $L : \mathbb{R}^{d_i} \to \mathbb{R}$ is strictly convex, twice continuously differentiable, and has a 917 unique minimizer denoted by θ_i^* . For any parameter vector θ_i of layer *i* such that $\|\theta_i - \theta_i^*\|_2 \ge 1$, 918 the function $L(\theta_i)$ diverges to infinity as $\|\theta_i\|_2 \to \infty$. *Proof.* By the strict convexity of L, for any θ_i such that $\|\theta_i - \theta_i^*\|_2 \ge 1$, we have:

$$\frac{L(\boldsymbol{\theta}_i) - L(\boldsymbol{\theta}_i^*)}{\|\boldsymbol{\theta}_i - \boldsymbol{\theta}_i^*\|_2} \ge \min_{\|\boldsymbol{\phi}\|_2 = 1} L(\boldsymbol{\theta}_i^* + \boldsymbol{\phi}) - L(\boldsymbol{\theta}_i^*), \tag{4}$$

where ϕ is a unit vector. For the convenience, here $L(\theta_i)$ denotes $L(\theta_i|\theta_{-i})$ where θ_{-i} denotes the parameters in the whole model except θ_i , and $L(\theta_i^*)$ denotes $L(\theta_i^*|\theta_{-i}^*)$. Define Δ_i as:

$$\Delta_i = \min_{\|\phi\|_2=1} L(\boldsymbol{\theta}_i^* + \phi) - L(\boldsymbol{\theta}_i^*), \tag{5}$$

a positive constant due to the strict convexity of L indicating the minimal rate of increase of L around θ_i^* .

Thus, the inequality can be written as:

$$L(\boldsymbol{\theta}_i) \ge \|\boldsymbol{\theta}_i - \boldsymbol{\theta}_i^*\|_2 \Delta_i + L(\boldsymbol{\theta}_i^*).$$
(6)

This implies that as $\|\boldsymbol{\theta}_i\|_2 \to \infty$, which necessarily implies $\|\boldsymbol{\theta}_i - \boldsymbol{\theta}_i^*\|_2 \to \infty$, the loss $L(\boldsymbol{\theta}_i)$ also diverges to infinity, since the term $\|\boldsymbol{\theta}_i - \boldsymbol{\theta}_i^*\|_2 \Delta_i$ dominates and increases without bound.

Note: We do not require the Hessian of the loss function, $\nabla^2 L(\theta_i)$, to be Lipschitz continuous; Assumption 2 only requires that the Hessian is continuous in a multiplicative sense within a neighbor-hood of constant radius.

Lemma 2 (Parameter Bound). Let $L : \mathbb{R}^d \to \mathbb{R}$ be a loss function for a neural network composed of multiple layers, each with parameters θ_i , and L is twice continuously differentiable and strictly convex with respect to each layer's parameters at a global minimizer θ^* . Assume each layer i satisfies the following condition:

$$L(\boldsymbol{\theta}_i) - \min L \le \frac{\mu_i R_i^2}{4},$$

where μ_i is the minimum eigenvalue of the Hessian of L at the minimizer for parameters of layer i, and R_i is a predefined radius. Then, it holds that

$$\|\boldsymbol{\theta}_i - \boldsymbol{\theta}_i^*\|_2 \leq R_i.$$

Proof. Suppose, by way of contradiction, there exists a θ_i such that $L(\theta_i) \leq \frac{\mu_i R_i^2}{4}$, but $\|\theta_i - \theta_i^*\|_2 > 0$ R_i . Define θ'_i as:

$$\boldsymbol{\theta}_{i}^{\prime} = \boldsymbol{\theta}_{i}^{*} + \sqrt{2(L(\boldsymbol{\theta}_{i}) - \min L)} \frac{\boldsymbol{\theta}_{i} - \boldsymbol{\theta}_{i}^{*}}{\mu_{i} \|\boldsymbol{\theta}_{i} - \boldsymbol{\theta}_{i}^{*}\|_{2}}.$$
(7)

Since θ'_i is a point between θ_i and θ^*_i , and due to the strict convexity of L, we have $L(\theta'_i) < L(\theta_i)$ by convexity. Considering the Taylor expansion of L at θ_i^* along the direction towards θ_i' , we have:

$$f(t) = L(\theta_i^* + t(\theta_i' - \theta_i^*)), \quad f(1) = L(\theta_i'), \quad f(0) = L(\theta_i^*), \quad f'(0) = 0,$$
(8)

$$f''(t) = (\boldsymbol{\theta}'_i - \boldsymbol{\theta}^*_i)^T \nabla^2 L(t\boldsymbol{\theta}'_i + (1-t)\boldsymbol{\theta}^*_i)(\boldsymbol{\theta}'_i - \boldsymbol{\theta}^*_i), \tag{9}$$

Given $f''(t) \ge \frac{\mu_i}{2} \|\boldsymbol{\theta}'_i - \boldsymbol{\theta}^*_i\|_2^2$ from Assumption 2 and the convexity, the Taylor expansion yields:

$$f(1) \ge f(0) + f'(0) + \frac{1}{2}f''(t) = L(\boldsymbol{\theta}_i^*) + \frac{\mu_i}{2} \|\boldsymbol{\theta}_i' - \boldsymbol{\theta}_i^*\|_2^2,$$

thus,

$$L(\boldsymbol{\theta}_i') \geq L(\boldsymbol{\theta}_i^*) + \frac{\mu_i}{2} \|\boldsymbol{\theta}_i' - \boldsymbol{\theta}_i^*\|_2^2$$

which contradicts the assumption that $L(\theta'_i) < L(\theta_i)$. Therefore, the original assumption that $\|\boldsymbol{\theta}_i - \boldsymbol{\theta}_i^*\|_2 > R_i$ must be false, concluding that $\|\boldsymbol{\theta}_i - \boldsymbol{\theta}_i^*\|_2 \leq R_i$.

Lemma 3 (Gradient Norm Bound). For any θ_i in layer *i* of a neural network, satisfying $\|\nabla L(\boldsymbol{\theta}_i)\|_2 \leq \frac{\mu_i R_i}{2}$, where μ_i is the minimum eigenvalue of the Hessian of L at the minimizer for layer i parameters, it holds that $\|\boldsymbol{\theta}_i - \boldsymbol{\theta}_i^*\|_2 \leq R_i$.

973 *Proof.* Assume, by way of contradiction, there exists a θ_i with $\|\nabla L(\theta_i)\|_2 \le \frac{\mu_i R_i}{2}$ and $\|\theta_i - \theta_i^*\|_2 > R_i$. Define a function f(t) by:

$$f(t) = \nabla L(\boldsymbol{\theta}_i^* + t \cdot (\boldsymbol{\theta}_i - \boldsymbol{\theta}_i^*)) \cdot \frac{\boldsymbol{\theta}_i - \boldsymbol{\theta}_i^*}{\|\boldsymbol{\theta}_i - \boldsymbol{\theta}_i^*\|_2},$$
(10)

where $f(0) = \nabla L(\boldsymbol{\theta}_i^*)$ due to $\boldsymbol{\theta}_i^*$ being a minimizer, and $f(R_i) = \nabla L(\boldsymbol{\theta}_i)$.

Due to the strict convexity of L, f(t) is a strictly monotone increasing function. The derivative with respect to t, must satisfy:

$$f'(t) = \frac{d}{dt} \left(\nabla L(\boldsymbol{\theta}_i^* + t \cdot (\boldsymbol{\theta}_i - \boldsymbol{\theta}_i^*)) \cdot \frac{\boldsymbol{\theta}_i - \boldsymbol{\theta}_i^*}{\|\boldsymbol{\theta}_i - \boldsymbol{\theta}_i^*\|_2} \right) \ge \frac{\|\nabla L(\boldsymbol{\theta}_i)\|_2}{2}, \tag{11}$$

by Assumption 2 and the fact that the gradient norm does not increase more than twice in any direction within the ball of radius R_i .

The fundamental theorem of calculus and the above inequality imply:

$$f(R_i) = f(0) + \int_0^{R_i} f'(t)dt \ge \int_0^{R_i} \frac{\|\nabla L(\boldsymbol{\theta}_i)\|_2}{2} dt = \frac{R_i \|\nabla L(\boldsymbol{\theta}_i)\|_2}{2},$$
(12)

However, $f(R_i) = \|\nabla L(\boldsymbol{\theta}_i)\|_2$ and this leads to

$$\|\nabla L(\boldsymbol{\theta}_i)\|_2 \ge \frac{R_i \|\nabla L(\boldsymbol{\theta}_i)\|_2}{2},\tag{13}$$

a contradiction unless $\|\boldsymbol{\theta}_i - \boldsymbol{\theta}_i^*\|_2 \leq R_i$.

Therefore, the original assumption that $\|\theta_i - \theta_i^*\|_2 > R_i$ must be false, proving the lemma.

Lemma 4 (Stability of Gradient Flow). Suppose the gradient $\nabla L(\boldsymbol{\theta}_i(t))$ and the Hessian $\nabla^2 L(\boldsymbol{\theta}_i(t))$ of the loss function L satisfy the conditions for all $t \in [0, 1]$ that ensure stability and convergence to a minimizer $\boldsymbol{\theta}_i^*$. Assume the differential equation

$$\frac{d\boldsymbol{\theta}_i(t)}{dt} = -\left(\nabla^2 L(\boldsymbol{\theta}_i(t))\right)^{-1} \nabla L(\boldsymbol{\theta}_i(t)), \quad \boldsymbol{\theta}_i(0) = \boldsymbol{\theta}_i, \quad \boldsymbol{\theta}_i(1) = \boldsymbol{\theta}_i^*, \tag{14}$$

has at least one solution on the interval [0, 1] and satisfies $\nabla L(\boldsymbol{\theta}_i(t)) = (1-t)\nabla L(\boldsymbol{\theta}_i)$ for all $t \in [0, 1]$.

Proof. We demonstrate this by showing that the given ordinary differential equation (ODE) is 1007 well-posed under the assumptions. The initial value problem

$$\frac{d\boldsymbol{\theta}_i(t)}{dt} = -\left(\nabla^2 L(\boldsymbol{\theta}_i(t))\right)^{-1} \nabla L(\boldsymbol{\theta}_i(t)),\tag{15}$$

can be solved over the interval [0, 1] due to the continuity and positive definiteness of $\nabla^2 L$, which ensures the existence and uniqueness of the solution by the Picard-Lindelöf theorem.

1013 Define $T_{\max,i}$ as the largest positive number such that the solution exists on $[0, T_{\max,i}]$. We claim 1014 $T_{\max,i} \ge 1$, based on the behavior of the gradient along the solution path. Considering:

$$\frac{d}{dt}\nabla L(\boldsymbol{\theta}_i(t)) = \nabla^2 L(\boldsymbol{\theta}_i(t)) \frac{d\boldsymbol{\theta}_i(t)}{dt} = -\nabla L(\boldsymbol{\theta}_i(t)),$$
(16)

which implies that $\nabla L(\boldsymbol{\theta}_i(t)) = e^{-t} \nabla L(\boldsymbol{\theta}_i)$. Since $\nabla L(\boldsymbol{\theta}_i(t)) = (1-t) \nabla L(\boldsymbol{\theta}_i)$ for $t \in [0, 1]$, the condition aligns perfectly.

Finally, since $\theta_i(1)$ has zero gradient by the construction of the ODE, $\theta_i(1)$ must be θ_i^* . This completes the proof.

Lemma 5 (Quadratic Form Integration). Assume the gradient norm $\|\nabla L(\theta_i)\|_2$ and the Hessian $\nabla^2 L(\theta_i)$ satisfy certain conditions over the interval [0, 1]. Suppose either

1.
$$L(\theta_i) - \min L \le \frac{\mu_i R_i^2}{16}$$
, or

1026 2.
$$\|\nabla L(\boldsymbol{\theta}_i)\|_2 \leq \frac{\mu_i R_i}{4}$$

where μ_i is the minimum eigenvalue of the Hessian at the minimizer for the parameters of layer *i*, then it holds that

$$\left\|\nabla L(\boldsymbol{\theta}_i)^T (\nabla^2 L(\boldsymbol{\theta}_i))^{-1} \nabla L(\boldsymbol{\theta}_i)\right\| \le 4(L(\boldsymbol{\theta}_i) - \min L).$$
(17)

1032 *Proof.* Let $\{\boldsymbol{\theta}_i(t)\}_{t=0}^1$ be the solution of the following differential equation:

$$\frac{d\boldsymbol{\theta}_i(t)}{dt} = -(\nabla^2 L(\boldsymbol{\theta}_i(t)))^{-1} \nabla L(\boldsymbol{\theta}_i(t)), \quad \boldsymbol{\theta}_i(0) = \boldsymbol{\theta}_i, \quad \boldsymbol{\theta}_i(1) = \boldsymbol{\theta}_i^*.$$
(18)

From Lemma 4, adapted for each layer, we have $\nabla L(\boldsymbol{\theta}_i(t)) = (1-t)\nabla L(\boldsymbol{\theta}_i)$ for all $t \in [0,1]$. Assume $\|\boldsymbol{\theta}_i(t) - \boldsymbol{\theta}_i^*\| \le R_i/2$ by Lemmas 2 and 3.

1039 By Assumption 2, for each layer, this implies:

$$(\nabla^2 L(\boldsymbol{\theta}_i(t)))^{-1} \ge \frac{1}{2} (\nabla^2 L(\boldsymbol{\theta}_i))^{-1}$$
 (19)

1042 for all $t \in [0, 1]$.

1030 1031

1034 1035

1040

1041

1044 Integrating the quadratic form along the path, we have:

$$L(\boldsymbol{\theta}_i) - \min L = L(\boldsymbol{\theta}_i(0)) - L(\boldsymbol{\theta}_i(1))$$

=
$$\int_0^1 (1-t)^2 (\nabla L(\boldsymbol{\theta}_i))^T (\nabla^2 L(\boldsymbol{\theta}_i(t)))^{-1} \nabla L(\boldsymbol{\theta}_i) dt.$$
 (20)

Substituting the inequality from equation 19, we simplify:

$$\begin{array}{ll}
1051 & \frac{1}{2} \int_{0}^{1} (1-t)^{2} dt (\nabla L(\boldsymbol{\theta}_{i}))^{T} (\nabla^{2} L(\boldsymbol{\theta}_{i}))^{-1} \nabla L(\boldsymbol{\theta}_{i}) \\
1053 & = \frac{1}{6} (\nabla L(\boldsymbol{\theta}_{i}))^{T} (\nabla^{2} L(\boldsymbol{\theta}_{i}))^{-1} \nabla L(\boldsymbol{\theta}_{i}).
\end{array} \tag{21}$$

1055 1056 1057
This integration shows that $\|\nabla L(\boldsymbol{\theta}_i)^T (\nabla^2 L(\boldsymbol{\theta}_i))^{-1} \nabla L(\boldsymbol{\theta}_i)\| \le 4(L(\boldsymbol{\theta}_i) - \min L)$, completing the proof.

Lemma 6 (Gradient and Loss Bound). Assuming the gradient norm $\|\nabla L(\theta_i)\|_2$ and the conditions on the loss function L are such that either

1.
$$L(\boldsymbol{\theta}_i) - \min L \leq \frac{\mu_i R_i^2}{4}, or$$

2.
$$\|\nabla L(\boldsymbol{\theta}_i)\|_2 \leq \frac{R_i \mu_i}{2}$$
,

it holds that

$$L(\boldsymbol{\theta}_i) - \min L \le \frac{1}{\mu_i} \|\nabla L(\boldsymbol{\theta}_i)\|^2.$$
(22)

1068 *Proof.* The proof follows a reasoning similar to that of Lemma 5 but adapted for each layer. Given 1069 the conditions on $L(\theta_i) - \min L$ or the norm of the gradient $\|\nabla L(\theta_i)\|_2$, we utilize the connection 1070 between the gradient norm and the difference in loss to bound $L(\theta_i) - \min L$.

From the gradient norm bound $\|\nabla L(\theta_i)\|_2$ and the positive definiteness and continuity of $\nabla^2 L$, the loss function exhibits quadratic behavior near the minimizer. This is characterized by the Taylor expansion:

$$L(\boldsymbol{\theta}_i) \approx L(\boldsymbol{\theta}_i^*) + \frac{1}{2} (\boldsymbol{\theta}_i - \boldsymbol{\theta}_i^*)^T \nabla^2 L(\boldsymbol{\theta}_i^*) (\boldsymbol{\theta}_i - \boldsymbol{\theta}_i^*),$$
(23)

1076 where θ_i^* is the minimizer of L.

Using the bound $\|\nabla L(\boldsymbol{\theta}_i)\|_2 \leq \frac{R_i \mu_i}{2}$, the Taylor series expansion around $\boldsymbol{\theta}_i^*$ implies:

$$L(\boldsymbol{\theta}_i) - \min L \le \frac{1}{2\mu_i} \|\nabla L(\boldsymbol{\theta}_i)\|^2,$$
(24)

1066 1067

1065

satisfying the condition given by Lemma 6.

This completes the proof by relating the behavior of the loss function's gradient at θ_i to its minimum value, leveraging the quadratic approximation provided by the Hessian at the minimizer. \square

Lemma 7 (Norm Bound on Inverse Hessian and Gradient Product). Assuming the gradient $\nabla L(\theta_i)$ and the Hessian $\nabla^2 L(\boldsymbol{\theta}_i)$ satisfy certain conditions such that either

1. $L(\theta_i) - \min L \leq \frac{\mu_i R_i^2}{16}, or$

2.
$$\|\nabla L(\boldsymbol{\theta}_i)\|_2 \leq \frac{R_i \mu_i}{4}$$
,

it holds that

$$\|\nabla^2 L(\boldsymbol{\theta}_i)^{-1} \nabla L(\boldsymbol{\theta}_i)\|_2 \le \frac{8(L(\boldsymbol{\theta}_i) - \min L)}{\mu_i}.$$
(25)

Proof. We derive this by using the properties of the Hessian and the gradient for the loss function L specific to layer *i*. From Lemma 2, we have:

$$\|\boldsymbol{\theta}_i - \boldsymbol{\theta}_i^*\|_2 \leq R_i$$

Given that $\nabla^2 L(\boldsymbol{\theta}_i^*) \geq \frac{\mu_i}{2} I$, and from Lemma 5 adapted for layer *i*, it holds that:

$$4(L(\boldsymbol{\theta}_i) - \min L) \geq \|\nabla L(\boldsymbol{\theta}_i)^T (\nabla^2 L(\boldsymbol{\theta}_i))^{-1} \nabla L(\boldsymbol{\theta}_i)\|.$$

Expanding and manipulating the inequality, we derive:

$$\|\nabla L(\boldsymbol{\theta}_i)\|^T (\nabla^2 L(\boldsymbol{\theta}_i))^{-1} \|\nabla L(\boldsymbol{\theta}_i)\| = \|\nabla^2 L(\boldsymbol{\theta}_i)^{-1} \nabla L(\boldsymbol{\theta}_i)\|^2$$

Given $\nabla^2 L(\theta_i)^{-1} \leq \frac{2}{\mu_i} I$, we can substitute this into our calculation to find:

$$\|\nabla^2 L(\boldsymbol{\theta}_i)^{-1} \nabla L(\boldsymbol{\theta}_i)\|^2 \le \frac{2}{\mu_i} \|\nabla L(\boldsymbol{\theta}_i)\|^2 \le \frac{4(L(\boldsymbol{\theta}_i) - \min L)}{\mu_i}$$

and finally,

$$\|\nabla^2 L(\boldsymbol{\theta}_i)^{-1} \nabla L(\boldsymbol{\theta}_i)\|_2 \le \frac{8(L(\boldsymbol{\theta}_i) - \min L)}{\mu_i}$$

completing the proof.

Lemma 8. For any $\theta_i \in \mathbb{R}^{d_i}$, where θ_i represents the parameters for the *i*-th layer, and satisfying that

$$\|\nabla^2 L(\boldsymbol{\theta}_i)^{-1} \nabla L(\boldsymbol{\theta}_i)\|_2 \leq \frac{R}{2},$$

it holds that

$$L(\boldsymbol{\theta}_i) - \min L \leq \nabla L(\boldsymbol{\theta}_i)^T (\nabla^2 L(\boldsymbol{\theta}_i))^{-1} \nabla L(\boldsymbol{\theta}_i) \leq 4(L(\boldsymbol{\theta}_i) - \min L).$$

Proof. Let $\{\theta_i(t)\}_{t=0}^1$ be the solution of the following differential equation:

$$\frac{d\boldsymbol{\theta}_i(t)}{dt} = -(\nabla^2 L(\boldsymbol{\theta}_i(t)))^{-1} \nabla L(\boldsymbol{\theta}_i(t)),$$

1122
1123 where
$$\theta_i(0) = \theta_i$$
 and $\theta_i(1) = \theta_i^*$.

We claim that for all $t \in [0, 1]$, $\|\boldsymbol{\theta}_i(t) - \boldsymbol{\theta}_i\|_2 \leq R_i$. If not, let T be the smallest positive number such that $\|\boldsymbol{\theta}_i(T) - \boldsymbol{\theta}_i\|_2 = R_i$. Such T exists because $\|\boldsymbol{\theta}_i(t) - \boldsymbol{\theta}_i\|_2$ is continuous in t and $\|\boldsymbol{\theta}_i(0) - \boldsymbol{\theta}_i\|_2 = 0.$

We can now bound the distance:

$$R_i = \|\boldsymbol{\theta}_i(T) - \boldsymbol{\theta}_i(0)\|_2 \le \int_0^T \left\|\frac{d\boldsymbol{\theta}_i(t)}{dt}\right\|_2 dt.$$

Substituting the derivative expression for $\theta_i(t)$, we get:

1133
$$= \int_0^T \left\| (\nabla^2 L(\boldsymbol{\theta}_i(t)))^{-1} \nabla L(\boldsymbol{\theta}_i(t)) \right\|_2 dt$$

 $\leq \int_0^T \|\nabla^2 L(\boldsymbol{\theta}_i(t))^{-1}\|_2 \|\nabla L(\boldsymbol{\theta}_i(t))\|_2 dt.$ From Assumption 2, we know that $\nabla^2 L(\boldsymbol{\theta}_i)^{-1} < 2(\nabla^2 L(\boldsymbol{\theta}_i(t)))^{-1}.$ Thus, we can bound this integral: $\leq 2 \int_{-\infty}^{T} \|\nabla^2 L(\boldsymbol{\theta}_i(t))^{-1} \nabla L(\boldsymbol{\theta}_i(t))\|_2 dt$ $\leq 2T \|\nabla^2 L(\boldsymbol{\theta}_i(t))^{-1} \nabla L(\boldsymbol{\theta}_i(t))\|_2$ Using the assumption that $\|\nabla^2 L(\boldsymbol{\theta}_i)^{-1} \nabla L(\boldsymbol{\theta}_i)\|_2 \leq \frac{R_i}{2}$, we get: $\leq 2T \frac{R_i}{2} = R_i T,$ which implies that T = 1. Therefore, for all $t \in [0, 1]$, $\|\boldsymbol{\theta}_i(t) - \boldsymbol{\theta}_i\|_2 \leq R_i$. By Assumption 2, we also have: $2(\nabla^2 L(\boldsymbol{\theta}_i))^{-1} \preceq (\nabla^2 L(\boldsymbol{\theta}_i(t)))^{-1} \preceq \frac{1}{2} (\nabla^2 L(\boldsymbol{\theta}_i))^{-1}.$ Now, we compute the difference in the loss function: $L(\boldsymbol{\theta}_i) - \min L = L(\boldsymbol{\theta}_i(0)) - L(\boldsymbol{\theta}_i(1)) = \int_0^1 \nabla L(\boldsymbol{\theta}_i(t))^T (\nabla^2 L(\boldsymbol{\theta}_i(t)))^{-1} \nabla L(\boldsymbol{\theta}_i(t)) dt.$ $= \int_{0}^{1} (1-t) \nabla L(\boldsymbol{\theta}_{i})^{T} (\nabla^{2} L(\boldsymbol{\theta}_{i}))^{-1} \nabla L(\boldsymbol{\theta}_{i}) dt.$ Thus: $L(\boldsymbol{\theta}_i) - \min L \leq \frac{1}{2} \nabla L(\boldsymbol{\theta}_i)^T (\nabla^2 L(\boldsymbol{\theta}_i))^{-1} \nabla L(\boldsymbol{\theta}_i).$ Finally, using the fact that $\int_0^1 (1-t) dt = \frac{1}{2}$, we complete the proof, showing that: $L(\boldsymbol{\theta}_i) - \min L < \nabla L(\boldsymbol{\theta}_i)^T (\nabla^2 L(\boldsymbol{\theta}_i))^{-1} \nabla L(\boldsymbol{\theta}_i) \le 4(L(\boldsymbol{\theta}_i) - \min L).$ **Lemma 9.** If $\lambda_i \leq \frac{R_i}{2\sqrt{d_i}}$, then for any $\Delta \leq \frac{R_i\mu}{10}$ and any $\theta_i \in \mathbb{R}_{\exists}^{d_i}$ satisfying $\sum_{i=1}^{n_{i}} \min\left\{\boldsymbol{p}_{i}^{T} \nabla L(\boldsymbol{\theta}_{i}) \sigma_{i}^{-1} \boldsymbol{p}_{i}^{T} \nabla L(\boldsymbol{\theta}_{i})\right\} \leq \Delta,$ where $\nabla^2 L(\boldsymbol{\theta}_i) = V_i \Sigma_i V_i^T$ is the eigen-decomposition of $\nabla^2 L(\boldsymbol{\theta}_i)$, \boldsymbol{p}_i is the *i*-th row of V_i , and $\Sigma_i = diag(\sigma_1, \ldots, \sigma_{d_i})$, it holds that $L(\boldsymbol{\theta}_i) - \min L \leq \frac{25\Delta^2}{\lambda^2 \mu}.$ *Proof.* Let $\{\boldsymbol{\theta}_i(t)\}_{t=0}^1$ be the solution to the ODE $\frac{d\boldsymbol{\theta}_i(t)}{dt} = -(\nabla^2 L(\boldsymbol{\theta}_i(t)))^{-1} \nabla L(\boldsymbol{\theta}_i(t)),$ starting from $\theta_i(0) = \theta_i$ and assume $\theta_i(1) = \theta_i^*$ as derived in previous lemmas. By Lemma 2, $\|\boldsymbol{\theta}_i(t) - \boldsymbol{\theta}_i^*\|_2 \leq R_i$ for all $t \in [0, 1]$. Define $I_0 \subseteq [d_i]$ as the indices where clipping does not occur. We have: $\sum_{i \in I} \sigma_i^{-1} \left| \boldsymbol{p}_i^T \nabla L(\boldsymbol{\theta}_i) \right|^2 \leq \Delta.$

Using Assumption 2, the Hessian continuity within a local radius implies:

1190
$$\sum_{i \in I_0} \left| \boldsymbol{p}_i^T \nabla L(\boldsymbol{\theta}_i(t)) \right|^2 \leq \Delta.$$
1191

For the newly restricted convex function L_0 on $\mathbb{R}_{\exists}^{I_0}$, which is L restricted to the subspace of $\mathbb{R}_{\exists}^{d_i}$ spanned by vectors corresponding to I_0 , by Lemma 1 and assuming L_0 is strictly convex, we apply Lemmas 6 and 8 by restricting to I_0 :

 $\|\nabla L_0(\boldsymbol{\theta}_i) + V_{I_0}^T \boldsymbol{\theta}_i^*\|_2^2 = \|\nabla L_0(\boldsymbol{\theta}_i)\|_2^2 \le \mu^{-1} \|\nabla L_0(\boldsymbol{\theta}_i)\|_2^2 \le \frac{25\Delta^2}{\lambda_i^2 \mu}.$

1200 Integrating the differential for L_0 , we can show:

$$L(\boldsymbol{\theta}_i) - \min L \leq \int_0^1 \nabla L(\boldsymbol{\theta}_i(t))^T (\nabla^2 L(\boldsymbol{\theta}_i(t)))^{-1} \nabla L(\boldsymbol{\theta}_i(t)) dt \leq \frac{25\Delta^2}{\lambda_i^2 \mu}.$$

1204 This completes the proof.

Lemma 10 (Descent Lemma). For any $\eta > 0$ and per-layer $\lambda_i > 0$ with $\eta \lambda_i \leq \frac{R_i}{\sqrt{d_i}}$, define

$$L(\boldsymbol{\theta}_{i}^{+}) - L(\boldsymbol{\theta}_{i}) \leq -(\eta - \eta^{2}\beta_{i}\lambda_{i})\sum_{j=1}^{d_{i}} \min\left\{\lambda_{i}, \frac{1}{\sigma_{i,j}}|\mathbf{v}_{i,j}^{T}\nabla L(\boldsymbol{\theta}_{i})|^{2} + C(\delta_{g}^{2} + \delta_{H}^{2})\right\}.$$
 (26)

 $\boldsymbol{\theta}_{i}^{+} = \boldsymbol{\theta}_{i} - \eta clip\left((\hat{\boldsymbol{g}}_{i}\hat{\boldsymbol{g}}_{i})^{-1}\hat{\boldsymbol{g}}_{i}, \lambda_{i}\right),$

where \hat{g}_i is the estimated gradient using a zero-order finite difference method with noise ϵ , such that:

$$\hat{\boldsymbol{g}}_i = \nabla L(\boldsymbol{\theta}_i) + \epsilon.$$

The theoretical bound for the descent is given by: 1218

$$L(\boldsymbol{\theta}_{i}^{+}) - L(\boldsymbol{\theta}_{i}) \leq -(\eta - \eta^{2}\beta_{i}\lambda_{i})\sum_{j=1}^{d_{i}} \min\left\{\lambda_{i}, \frac{1}{\sigma_{i,j}}|\mathbf{v}_{i,j}^{T}\nabla L(\boldsymbol{\theta}_{i})|^{2} + C(\delta_{g}^{2} + \delta_{H}^{2})\right\}$$

$$\leq -(\eta - \eta^2) \sum_{i=1}^{a} \min\left\{\lambda_i \, |\hat{\boldsymbol{g}}_i|, \, (\hat{\boldsymbol{g}}_i \hat{\boldsymbol{g}}_i)^{-1} |\hat{\boldsymbol{g}}_i|^2\right\} + O(h) + O(1/\sqrt{m}),$$

where h is the step size of the finite difference and m is the number of perturbations performed for
finite difference estimation.

Proof. Step 1. Derivation of the upper bound for $\|\hat{g}_i - \nabla L(\theta_i)\|$. To derive a theoretical bound for $\|\hat{g}_i - \nabla L(\theta_i)\|$, where \hat{g}_i is the gradient estimated using our proposed zero-order method, and $\nabla L(\theta_i)$ is the true gradient, we need to quantify the error due to using finite perturbations to approximate the gradient. Let's denote this error by ϵ , such that:

$$\epsilon_i = \hat{\boldsymbol{g}}_i - \nabla L(\boldsymbol{\theta}_i)$$

1235 Specifically, the gradient estimate for dimension *i* is obtained by:

$$\hat{\boldsymbol{g}}_i = rac{1}{m}\sum_{k=1}^m rac{L(oldsymbol{ heta}+holdsymbol{u}_k)-L(oldsymbol{ heta})}{h}oldsymbol{u}_k^{(i)},$$

where m is the number of perturbations, h is the step size for finite differences, and $u_k^{(i)}$ represents the *i*-th component of the random vector u_k . The true gradient, on the other hand, is:

1242

1243 1244 1245

1251 1252 1253

1257 1258 1259

$$abla L(oldsymbol{ heta}_i) = \lim_{h o 0} rac{L(oldsymbol{ heta} + holdsymbol{u}_k) - L(oldsymbol{ heta})}{h} oldsymbol{u}_k^{(i)},$$

The error between the estimated gradient \hat{g}_i and the true gradient $\nabla L(\theta_i)$ arises from two main sources. To derive a theoretical bound for the estimation error, $\|\hat{g}_i - \nabla L(\theta_i)\|$, we consider both sources of error.

1249 1. Finite Difference Approximation Error. By Taylor expansion, for a small step size h, we have: 1250

$$L(\boldsymbol{\theta} + h\boldsymbol{u}_k) = L(\boldsymbol{\theta}) + h\nabla L(\boldsymbol{\theta})^T \boldsymbol{u}_k + \frac{h^2}{2}\boldsymbol{u}_k^T H(\boldsymbol{\theta})\boldsymbol{u}_k + O(h^3),$$

where $H(\theta)$ is the Hessian of L at θ . - Thus, the error due to finite differences is of order O(h). Specifically, the bias in the gradient estimate is proportional to:

$$\operatorname{Bias} = O\left(\frac{h}{2} \|H(\boldsymbol{\theta})\|\right)$$

12602. Monte Carlo Sampling Error. The gradient estimate involves averaging over m samples of random1261perturbations. By the Central Limit Theorem, the variance of the gradient estimate decreases with the1262number of samples m. Specifically:

Variance =
$$O\left(\frac{\sigma^2}{m}\right)$$

1267 where σ^2 is the variance of the directional derivative $\nabla L(\boldsymbol{\theta})^T \boldsymbol{u}_k$. 1268

1269The total error can be expressed as a combination of the bias and variance components. Using a norm
(e.g., Euclidean norm) to quantify the error, we have:

 $\|\hat{\boldsymbol{g}}_i - \nabla L(\boldsymbol{\theta}_i)\| \leq O(h\|H(\boldsymbol{\theta})\|) + O\left(\frac{\sigma}{\sqrt{m}}\right).$

 $\|\hat{\boldsymbol{g}}_i - \nabla L(\boldsymbol{\theta}_i)\| = O\left(h\|H(\boldsymbol{\theta})\| + \frac{\sigma}{\sqrt{m}}\right).$

1271 1272

1273 1274

Thus, the theoretical bound on the error is:

1276 1277

1275

1278 1279

1280

1285

1286 1287 Step 2. Derivation of the upper bound for $\|\hat{g}_i^2 - \text{diag}(\nabla^2 L(\theta_i))\|$. Let's denote $\text{diag}(\nabla^2 L(\theta_i))$ as the diagonal of the true Hessian, and $\hat{H}_i = \hat{g}_i^2$ as the diagonal Hessian estimated from the zero-order

the diagonal of the true Hessian, and $H_i = \hat{g}_i^2$ as the diagonal Hessian estimated from the signalient estimate, where each diagonal element is given by $\hat{g}_i \hat{g}_i$. The diagonal diagonal element is given by $\hat{g}_i \hat{g}_i$.

To derive the theoretical bound for $||H_i - H_i||$, we consider:

 $\|\hat{H}_i - H_i\| = \|\hat{\boldsymbol{g}}_i^2 - \operatorname{diag}(\nabla^2 L(\boldsymbol{\theta}_i))\|.$

1288 Let's rewrite \hat{g}_i as:

1289 1290 1291

 $\hat{\boldsymbol{g}}_i = \nabla L(\boldsymbol{\theta}_i) + \epsilon_i,$

1292 where ϵ_i represents the noise introduced due to the limited number of perturbations.

The estimated diagonal Hessian element for each component i can be written as:

1295

 $\hat{H}_i^{(i)} = (\nabla L(\boldsymbol{\theta}_i) + \epsilon_i)^2.$

Expanding this expression gives:

$$\hat{H}_{i}^{(i)} = (\nabla L(\boldsymbol{\theta}_{i}))^{2} + 2\nabla L(\boldsymbol{\theta}_{i})\epsilon_{i} + \epsilon_{i}^{2}$$

13001301 The true diagonal Hessian element is:

$$H_i^{(i)} = \operatorname{diag}(\nabla^2 L(\boldsymbol{\theta}_i))^{(i)}.$$

1305 Thus, the error for each component can be expressed as:

$$\hat{H}_i^{(i)} - H_i^{(i)} = (\nabla L(\boldsymbol{\theta}_i))^2 + 2\nabla L(\boldsymbol{\theta}_i)\epsilon_i + \epsilon_i^2 - H_i^{(i)}.$$

1309 To find the bound for the error, we need to bound the terms involving ϵ_i :

1310 1. Term 1: $2\nabla L(\boldsymbol{\theta}_i)\epsilon_i$

This term represents the interaction between the true gradient and the noise. Since $\|\epsilon_i\| \leq O\left(h\|H(\boldsymbol{\theta}_i)\| + \frac{\sigma}{\sqrt{m}}\right)$, we can bound this term as:

$$|2\nabla L(\boldsymbol{\theta}_i)\epsilon_i| \leq 2 \|\nabla L(\boldsymbol{\theta}_i)\|O\left(h\|H(\boldsymbol{\theta}_i)\| + \frac{\sigma}{\sqrt{m}}\right).$$

1318 2. Term 2: ϵ_i^2

1320 The noise squared term can be bounded by:

$$\epsilon_i^2 \le O\left(h^2 \|H(\boldsymbol{\theta}_i)\|^2 + \frac{\sigma^2}{m}\right).$$

Combining these results, we have:

$$\|\hat{H}_i - H_i\| = O\left(\left(\nabla L(\boldsymbol{\theta}_i)\right)^2 - H_i + 2\|\nabla L(\boldsymbol{\theta}_i)\| \left(h\|H(\boldsymbol{\theta}_i)\| + \frac{\sigma}{\sqrt{m}}\right) + \left(h^2\|H(\boldsymbol{\theta}_i)\|^2 + \frac{\sigma^2}{m}\right)\right).$$

Thus, the error bound for the diagonal Hessian estimation is:

$$\|\hat{H}_i - H_i\| = O\left(h^2 \|H(\boldsymbol{\theta}_i)\|^2 + \frac{\sigma^2}{m} + 2h \|\nabla L(\boldsymbol{\theta}_i)\| \|H(\boldsymbol{\theta}_i)\| + \frac{2\|\nabla L(\boldsymbol{\theta}_i)\|\sigma}{\sqrt{m}}\right).$$

Step 3. Combination of the bounds. Let $u_i = \operatorname{clip} \left((\hat{g}_i \hat{g}_i)^{-1} \hat{g}_i, \lambda_i \right)$. By the definition of the clipping operation:

$$||u_i||_{\infty} \le \lambda_i.$$

1341 Thus:

 $||\boldsymbol{\theta}_i^+ - \boldsymbol{\theta}_i|| = \eta ||u_i|| \le \eta \lambda_i \sqrt{d_i}.$

1346 Define the function:

 $f(t) = L(\boldsymbol{\theta}_i + (1-t)u_i).$

By Assumption 4.2, we know that:

$$f''(t) \le 2f''(0) \quad \text{for all } t \in [0,1],$$

1353 and hence:

$$f(1) = f(0) + f'(0) + \int_0^1 \int_0^s f''(s) \, ds \, dt \le f(0) + f'(0) + f''(0).$$

1358 The zero-order estimate introduces noise ϵ in the estimated gradient:

 $\hat{\boldsymbol{g}}_i = \nabla L(\boldsymbol{\theta}_i) + \epsilon.$

1362 Thus:

$$f'(0) = -\eta \sum_{i=1}^{d} \min \left\{ \lambda_i |\hat{g}_i|, \, (\hat{g}_i \hat{g}_i)^{-1} |\hat{g}_i|^2 \right\}.$$

1368 Using the bound for the error $||\epsilon|| \le O\left(h ||H(\theta_i)|| + \frac{\sigma}{\sqrt{m}}\right)$, the noise affects the effective descent 1369 rate. Therefore, the new bound for f'(0) is:

$$f'(0) \approx -\eta \sum_{i=1}^{d} \min\left\{\lambda_i \left(|\nabla L(\boldsymbol{\theta}_i)| + |\boldsymbol{\epsilon}|\right), \left(\hat{\boldsymbol{g}}_i \hat{\boldsymbol{g}}_i\right)^{-1} \left(|\nabla L(\boldsymbol{\theta}_i)| + |\boldsymbol{\epsilon}|\right)^2\right\}.$$

The Hessian is estimated using \hat{g}_i^2 . The noise in the diagonal Hessian estimate affects the curvature. Therefore, for the second derivative, we have:

$$f''(0) \le \eta^2 \sum_{i=1}^d \min\left\{\lambda_i |\hat{g}_i|, \, (\hat{g}_i \hat{g}_i)^{-1} |\hat{g}_i|^2\right\}.$$

1380 The noise in the Hessian (δ_H) affects the estimation, and thus the bound is affected as follows:

$$f''(0) \le \eta^2 \sum_{i=1}^d \min\left\{\lambda_i \left(|\nabla L(\boldsymbol{\theta}_i) + \epsilon|\right), \, (\hat{\boldsymbol{g}}_i \hat{\boldsymbol{g}}_i)^{-1} (|\nabla L(\boldsymbol{\theta}_i) + \epsilon|)^2\right\}.$$

Combining these results, the descent bound is affected by both the gradient and Hessian noise. We obtain:

$$L(\boldsymbol{\theta}_{i}^{+}) - L(\boldsymbol{\theta}_{i}) \leq -(\eta - \eta^{2}) \sum_{i=1}^{d} \min\left\{\lambda_{i} |\hat{\boldsymbol{g}}_{i}|, \, (\hat{\boldsymbol{g}}_{i} \hat{\boldsymbol{g}}_{i})^{-1} |\hat{\boldsymbol{g}}_{i}|^{2}\right\} + C(\delta_{g}^{2} + \delta_{H}^{2}),$$

where C is a constant that depends on the properties of the function L. δ_g represents the bound on the gradient estimation noise $\delta_g = O\left(h ||H(\boldsymbol{\theta}_i)|| + \frac{\sigma}{\sqrt{m}}\right)$, and δ_H represents the bound on the Hessian estimation noise:

$$\delta_H = O\left(h^2 ||H(\boldsymbol{\theta}_i)||^2 + \frac{\sigma^2}{m} + 2h ||\nabla L(\boldsymbol{\theta}_i)|| ||H(\boldsymbol{\theta}_i)|| + \frac{2||\nabla L(\boldsymbol{\theta}_i)||\sigma}{\sqrt{m}}\right).$$

1399 Lemma 11 (Convergence Lemma). For any $\lambda_i \leq \frac{R_i}{\sqrt{d_i}}$ and some $T_i \in \mathbb{N}$, if $L(\boldsymbol{\theta}_{T_i,i}) - \min L \leq \frac{\mu_i^2}{8}$, 1400 then for all $t \geq T_i$,

1.
$$\boldsymbol{\theta}_{t+1,i} = \boldsymbol{\theta}_{t,i} - \eta (\nabla_{\boldsymbol{\theta}_i}^2 L(\boldsymbol{\theta}_{t,i}))^{-1} \nabla L(\boldsymbol{\theta}_{t,i}),$$

2. $L(\boldsymbol{\theta}_{t,i}) - \min L \leq (1 - \eta (1 - \eta))^{t - T_i} (L(\boldsymbol{\theta}_{T_i,i}) - \min L).$

Proof. By Lemma 10, we have for all $t \ge T$, $(\theta_{t,i}) - \min L \le L(\theta_{T,i}) - \min L \le \frac{\mu^2}{8}$. Therefore, by Lemma 7, we have that $\|\nabla^2 L(\theta_{t,i}) - \nabla L(\theta_{t,i})\|_2 \le \lambda_i$ for all $t \ge T$, which implies clipping will not happen.

For the second claim, by Lemmas 5 and 10, we have that

1409 1410

1411 1412

1415

1421

1422

$$L(\boldsymbol{\theta}_{t+1,i}) - L(\boldsymbol{\theta}_{t,i}) \le -(\eta - \eta^2) \sum_{i=1}^{a} \sigma_i^{-1} |\mathbf{v}_i^T \nabla L(\boldsymbol{\theta}_{t,i})|^2,$$
(27)

where v_i is the *i*-th row of matrix V from the eigen-decomposition of $\nabla^2 L(\theta_i)$. By further simplification,

$$-(\eta - \eta^2)\nabla L(\boldsymbol{\theta}_{t,i})^T (\nabla^2 L(\boldsymbol{\theta}_{t,i}))^{-1} \nabla L(\boldsymbol{\theta}_{t,i}) \leq -\eta(1-\eta)(L(\boldsymbol{\theta}_{t,i}) - \min L),$$

thus, we conclude that the loss decreases at least geometrically by the factor $(1 - \eta(1 - \eta))$ each step after time *T*, thereby proving the convergence rate.

Theorem 2. Under Assumptions 1 and 2, let $\eta = \frac{1}{2}$ and $\lambda_i = \frac{R_i}{2\sqrt{d_i}}$. The update reaches a loss at most ϵ in

 $T \le \max_{i} \left\{ d_i \cdot \left(L(\boldsymbol{\theta}_{0,i}) - \min L \right) + \ln \left(\frac{\mu_i R_i^2}{32d_i \epsilon} \right) \right\}.$ (28)

1423 steps, where L is the loss function, $\theta_{0,i}$ is the initial parameter vector for layer *i*.

1425 *Proof.* Phase 1: Initial Rapid Decrease.

1427 By Lemma 10 (Descent Lemma), we have a guarantee on the descent rate per step for each layer *i*:

$$L(\boldsymbol{\theta}_{t+1,i}) - L(\boldsymbol{\theta}_{t,i}) \leq -(\eta - \eta^2) \sum_{j=1}^{d_i} \min\left\{\lambda_i; \frac{1}{\sigma_{i,j}} \left| v_{i,j}^T \nabla L(\boldsymbol{\theta}_{t,i}) \right|^2 \right\},$$

1429 1430 1431

1437 1438 1439

1444

1449 1450 1451

1453 1454 1455

1428

where $\sigma_{i,j}$ is the *j*-th eigenvalue corresponding to the *i*-th layer, and $v_{i,j}$ is the corresponding eigenvector.

1434 Applying this result, we estimate a decrease in the loss function per layer under the condition that the 1435 gradient norm for layer *i* is significantly larger than the error threshold ϵ . This phase continues until 1436 the loss reduction per step for each layer falls below a certain threshold, say when:

$$L(\boldsymbol{\theta}_{t,i}) - \min L \le \frac{\mu_i^2}{8}.$$

1440 Phase 2: Exponential Decay.

1441 1442 Once the loss for each layer is sufficiently reduced, Lemma 11 guides the convergence from this 1443 point:

$$L(\boldsymbol{\theta}_{t,i}) - \min L \le (1 - \eta(1 - \eta))^{t - T_i} (L(\boldsymbol{\theta}_{T_i,i}) - \min L),$$

indicating an exponential decay in error for each layer. The factor $(1 - \eta(1 - \eta))$ represents the contraction per step, dependent on the learning rate η .

1447 To calculate the total number of steps T_i for each layer, consider that: 1448

$$\frac{\mu_i^2}{8} \approx \epsilon \Rightarrow T_i \approx \frac{\ln\left(\frac{L(\boldsymbol{\theta}_{0,i}) - \min L}{\epsilon}\right)}{-\ln(1 - \eta(1 - \eta))}$$

1452 Simplifying the expression for $\eta = \frac{1}{2}$, we get:

$$T_i \approx 2 \ln \left(\frac{L(\boldsymbol{\theta}_{0,i}) - \min L}{\epsilon} \right)$$

1456 since $\ln(1 - \eta(1 - \eta)) \approx -\eta(1 - \eta)$ for small η . 1457

Combining Phases 1 and 2.

For each layer, combining the estimates from Phase 1 and Phase 2, the total number of steps T_i needed to reach a loss of ϵ for layer *i* is given by:

 $T_i \leq d_i \cdot (L(\boldsymbol{\theta}_{0,i}) - \min L) + \ln\left(\frac{\mu_i R_i^2}{32d_i \epsilon}\right),$

1464 Finally, to ensure convergence across all layers, we take the maximum over all layers:

$$T \le \max_{i} \left\{ d_{i} \cdot \left(L(\boldsymbol{\theta}_{0,i}) - \min L \right) + \ln \left(\frac{\mu_{i} R_{i}^{2}}{32 d_{i} \epsilon} \right) \right\}.$$

This completes the proof by integrating the rapid initial decrease and the subsequent exponential decay for each layer.

1473 This reflects an improved convergence rate due to the use of different λ_i values for different layers, 1474 reducing the dependency on the total dimension d into the dimension $\max_i d_i$.

1476 C.1 LIMITATIONS

Like other second-order optimizers, HELENE stores the history of gradients and Hessian values, with memory usage proportional to the size of the parameters. Therefore, both theoretically and practically, HELENE requires only three times the memory of MeZO. For example, in OPT-1.3b, MeZO/zero-shot requires 4GB, ICL needs 6GB, Prefix Fine-Tuning uses 19GB, and full-parameter fine-tuning consumes 27GB, while HELENE needs just 14GB. Despite this, HELENE achieves up to 20 times faster convergence and delivers the best overall performance.