Riemannian Lyapunov Framework: Optimization as Closed Loop Control on Riemannian Manifolds

Yixuan Wang[*]
University of Florida

Omkar Sudhir Patil
Louisiana State University

Warren E. Dixon
Virginia Tech

Abstract

We introduce the Riemannian Lyapunov Framework (RLF), a unified framework that represents first order optimization as a closed loop controlled dynamical system on a Riemannian parameter manifold. Within this framework, optimizers are specified by a quadruple (M, g, Φ, η) consisting of a Riemannian metric, an internal state transition map, a direction field generator, and a lifting parameter. Classical and modern methods including SGD, Adam, AdamW, Lion and Shampoo are recovered as specific instantiations of this quadruple. The central structural object of the framework is a time indexed family of target submanifolds in the extended state space of parameters, velocities, and internal memory, which we call the Normally Attracting Invariant Manifold (NAIM) family. These submanifolds organize training dynamics into two timescales: a fast contraction of the velocity state onto the current target submanifold, followed by a slow descent along it. Using Riemannian backstepping, we construct a strict Lyapunov function on the extended state space and prove Uniform Ultimate Boundedness (UUB) of the optimization trajectory under a descent alignment assumption and bounded stochastic disturbances. Under the additional Polyak-Lojasiewicz (PL) condition, the Lyapunov function converges linearly to a bounded neighborhood of the optimum, defined as the drift of the target direction field between consecutive iterates after parallel transport. To validate the framework's generativity, we derive three concrete instantiations of the quadruple, each exercising a distinct design axis, and show on geometric diagnostics, image classification benchmarks, and language modeling that their empirical behavior matches the theoretical predictions. The framework casts optimizer design as controller synthesis, complementing heuristic perspectives by providing a constructive template with provable guarantees.

1 Introduction

The evolution of deep learning optimization has produced a vast ecosystem of algorithms designed to navigate complex loss landscapes [1, 2, 3]. This lineage includes Stochastic Gradient Descent (SGD) [4, 5], momentum methods [6, 7], Nesterov's Accelerated Gradient (NAG) [8], and a diverse array of adaptive methods such as AdaGrad [9], RMSProp [10], Adam [11], AdamW [12], and Adafactor [13]. Recently, specialized techniques like Lion [14], Shampoo [15], and Sophia [16] have pushed the boundaries of efficiency in large scale training by utilizing sign based updates [17, 18] or second order information [19]. While these methods are empirically excellent and form the backbone of modern machine learning, they are largely treated as a collection of distinct heuristics. Theoretical analyses of these optimizers remain fragmented, often focusing on narrow properties or specific algorithmic artifacts [20, 21] while failing to provide a global, principled explanation for their shared success.

[*]Corresponding author: wang.yixuan@ufl.edu.

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

Previous attempts to unify accelerated methods often leverage continuous time limits that approximate discrete iterations by second order damped dynamics, providing interpretable surrogate models for rate and stability [22, 23]. Other works have explored unified frameworks through the lens of proximal operators [24] or mirror descent [25], yet these perspectives typically analyze algorithms through fixed time rescalings or simplified state representations, and therefore, do not directly synthesize feedback laws that explicitly regulate the evolution of auxiliary optimizer states. As a result, they do not provide a constructive mechanism level explanation for the observed separation between fast residual alignment and slower descent, especially in discrete time with time varying preconditioning and stochastic gradients. This lack of a cohesive geometric and control theoretic foundation limits our ability to systematically design new optimizers that are both stable and effective across different architectures.

In this paper, we answer the key open question: *Is there a unified geometric principle that governs the stability and efficacy of these diverse optimizers?* We introduce the Riemannian Lyapunov Framework (RLF), which provides a unified control perspective for understanding and designing machine learning optimizers. Our motivation is grounded in the observation of two timescale dynamics where the system state must track a specific relationship between its velocity and the gradient field. We formalize this using the concept of a Normally Attracting Invariant Manifold (NAIM), which serves as the geometric skeleton relating the update speed to the target direction field. Drawing inspiration from the idea of backstepping in nonlinear control theory [26, 27, 28], we construct a strict Lyapunov function that encodes both the objective value and the distance to the NAIM. This approach allows us to derive a controller that actively forces the system to track the manifold, transforming optimizer design from heuristic experimentation into a principled synthesis of Lyapunov based feedback laws. Our contributions lie on:

(i) Unified Riemannian Geometry: We demonstrate that diverse optimization components are equivalent to fundamental geometric objects where preconditioning corresponds to selecting a Riemannian metric and momentum represents an extended velocity state in the tangent bundle.

(ii) The NAIM Mechanism: We introduce the NAIM as the primary structure organizing training dynamics, proving that optimization consists of a fast process of backstepping the speed state to a target graph followed by a slow drift along that manifold.

(iii) Strict Lyapunov Design: We provide a systematic methodology for designing the RLO family of optimizers by synthesizing control laws that satisfy strict Lyapunov requirements using a Riemannian backstepping approach, ensuring robust convergence even under the disturbances typical of stochastic gradients.

This framework provides a blueprint for optimization algorithm design from the perspective of controller design and paves the way for principled innovations.

2 Related Work

Lyapunov methods provide a common basis for stability analysis and feedback design [27]. The analysis constructs a Lyapunov function V and studies its derivative along system trajectories to establish stability or convergence properties. Such functions also support analyses of accelerated and adaptive optimization dynamics [29, 30]. This perspective motivates constructing optimizer updates that jointly regulate objective descent and velocity tracking.

NAIM describe attraction toward an invariant manifold and evolution along it [31, 32]. Their transverse contraction dominates the tangent dynamics. Related analyses of fast and slow dynamics arise in singular perturbation theory and stochastic algorithms with two time scales [33, 34]. For optimizer design, this geometric perspective motivates separating the generation of a target direction from the dynamics that track it.

3 Geometric Setup and Problem Formulation

We study the minimization of a differentiable objective function $f\colon \mathcal{M} \to \mathbb{R}$ over a smooth n dimensional parameter manifold $\mathcal{M}$ endowed with a Riemannian metric g. The optimal value of f is denoted by f^*, which may be unknown. Discrete time is indexed by $k \in \mathbb{N}$.

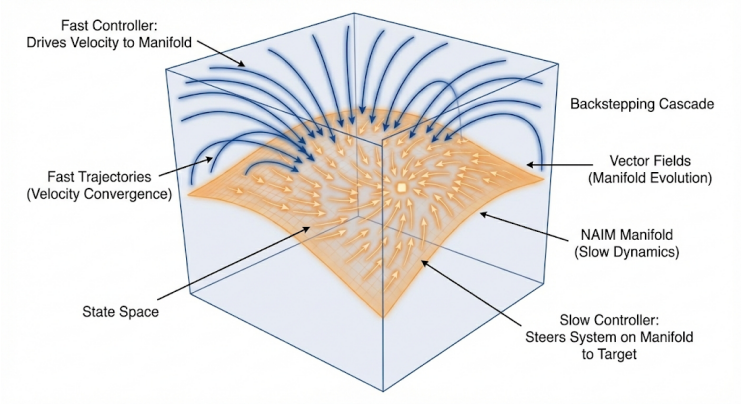


Figure 1: Geometric intuition of the NAIM Lyapunov framework. The orange surface represents the NAIM embedded in the extended state space θ, v. Blue streamlines illustrate the fast dynamics: the velocity state v rapidly contracts onto Λ at a rate governed by the lifting parameter η, regardless of initial conditions. Orange arrows show the slow dynamics: once the trajectory reaches Λ , the system evolves along the direction field Φ toward the target optimum at the center.

3.1 The Riemannian Configuration

For each $\theta \in \mathcal{M}$, let $T_\theta\mathcal{M}$ denote the tangent space at θ. The Riemannian metric assigns an inner product $g(\cdot,\cdot)$ on $T_\theta\mathcal{M}$, and the induced norm is

$$\|\xi\|_{g_\theta} \triangleq \sqrt{g_\theta(\xi,\xi)}, \quad \xi \in T_\theta\mathcal{M}.$$

The Riemannian gradient of f at θ, denoted $\operatorname{grad} f(\theta) \in T_\theta\mathcal{M}$, is defined by the identity

$$df(\theta)[\xi] = g_\theta(\operatorname{grad} f(\theta), \xi)$$

for all $\xi \in T_\theta\mathcal{M}$, where $df(\theta)[\xi]$ is the directional derivative of f at θ along ξ.

To perform iterative optimization, we require a mechanism to map tangent vectors back onto the manifold and to compare vectors across distinct tangent spaces. We employ a retraction R and a vector transport $\mathcal{T}$ as our primary computational operators.[2] A retraction is a smooth mapping $R\colon T\mathcal{M} \to \mathcal{M}$ such that for any θ, the restriction $R_\theta\colon T_\theta\mathcal{M} \to \mathcal{M}$ satisfies $R_\theta(0) = \theta$ and its local differential at the origin is the identity map. Furthermore, for any two points θ and $\varphi = R_\theta(\xi)$, the vector transport $\mathcal{T}_{\theta\to\varphi}\colon T_\theta\mathcal{M} \to T_\varphi\mathcal{M}$ provides a linear mapping that moves a tangent vector from the source space to the destination space along the retraction curve.

3.2 Optimization as an Extended Dynamical System

We conceptualize the optimizer not merely as a rule for updating parameters, but as a controlled dynamical system evolving in an extended state space. We define the system state at time k as the tuple $x_k = (\theta_k, v_k, y_k)$. Here, $\theta_k \in \mathcal{M}$ represents the current model parameters. The variable $v_k \in T_{\theta_k}\mathcal{M}$ represents the physical velocity or momentum of the system, residing in the tangent bundle. The variable $y_k \in \mathcal{Y}$ represents the internal memory state of the optimizer, such as the accumulated first or second moments of the gradients, where $\mathcal{Y}$ is a vector space appropriate for the selected statistics.

The evolution of this system is governed by an internal state transition map Ψ and a direction field generator Φ. The transition map $\Psi : \mathcal{Y} \times T\mathcal{M} \to \mathcal{Y}$ updates the memory state based on the current stochastic gradient. The direction field generator $\Phi : \mathcal{Y} \times T\mathcal{M} \to T\mathcal{M}$ constructs a target update direction $d_k = \Phi(y_k, \operatorname{grad} f(\theta_k))$ in the tangent space $T_{\theta_k}\mathcal{M}$. This map Φ encapsulates the structural logic of the optimizer, including operations such as normalization, clipping, or sign-based transformations. Consequently, a specific optimization algorithm is fully characterized by the tuple

[2]When $\mathcal{M} = \mathbb{R}^n$ with the Euclidean metric, one has $T_\theta\mathcal{M} = \mathbb{R}^n$, $R_\theta(\xi) = \theta + \xi$ and $\mathcal{T}_{\theta\to\varphi}(\xi) = \xi$. All definitions above reduce to their standard vector space counterparts.

$(\mathcal{M}, g, \Phi, \Psi)$ alongside a sequence of step sizes h_k and lifting parameters η_k. We rigorously demonstrate in Appendix B that adaptive preconditioning methods in Euclidean space are mathematically equivalent to the selection of a specific time-varying Riemannian metric g, thereby bringing adaptive optimizers within this geometric framework.

4 The NAIM-Lyapunov Framework

Building upon the geometric formulation of the optimizer as an extended dynamical system, we adopt a constructive control perspective. We begin with a generic open-loop mechanical system on the manifold and ask: *What control law is required to force the system to track the NAIM while minimizing the objective function?* The answer is obtained through Riemannian backstepping, a Lyapunov based synthesis within the proposed framework under a strict Lyapunov requirement. Fig 1 illustrates the geometric intuition underlying the framework.

4.1 Open-Loop Dynamics and the Geometric Objective

We model the optimization process as a mechanical system on the tangent bundle $T\mathcal{M}$. The open-loop dynamics in continuous time are governed by the covariant equations:

$$\dot{\theta} = -v, \quad \nabla_{\dot{\theta}} v = u,$$

where $\theta \in \mathcal{M}$ is the position, $v \in T_\theta \mathcal{M}$ is the velocity, ∇ is the Levi-Civita connection associated with metric g, and $u \in T_\theta \mathcal{M}$ is the control input (acceleration) we must design. Our geometric objective is twofold. First, we require the system to minimize the loss $f(\theta)$. Second, we require the physical velocity v to adhere to the direction field generator introduced in Section 3. This defines the NAIM graph

$$\Lambda(y) = \{(\theta, v) \in T_\theta \mathcal{M} \colon v = \Phi(y, \operatorname{grad} f)\}.$$

and the normal residual measuring the deviation from this graph is defined as: $z \triangleq v - \Phi(y, \operatorname{grad} f) \in T_\theta \mathcal{M}$. The control problem is to synthesize an input u that renders Λ attractive and invariant while ensuring $f(\theta)$ decreases.

For clarity, when indexed by a discrete time step k we write $\Lambda_k = \Lambda(y_k)$ and $z_k = v_k - \Phi(y_k, \operatorname{grad} f(\theta_k))$. The full extended state (θ, v, y) lies in $T\mathcal{M} \times \mathcal{Y}$, and the NAIM is a y indexed family of submanifolds in this full space rather than a single static set.

4.2 Lyapunov-Based Control Synthesis

To synthesize the control law u, we postulate a strict Lyapunov function candidate V that encodes a weighted sum of the potential energy (loss) and the kinetic energy of the residual,

$$V(\theta, v, y) = f(\theta) - f^\star + \|z\|_{g_\theta}^2.$$

Taking the time derivative of V along the trajectories of the open-loop system yields

$$\dot{V} = \langle \operatorname{grad} f, \dot{\theta} \rangle_{g_\theta} + 2\langle z, \nabla_{\dot{\theta}} z \rangle_{g_\theta} = -\langle \operatorname{grad} f, v \rangle_{g_\theta} + 2\langle z, \nabla_{\dot{\theta}} z \rangle_{g_\theta}.$$

Substituting $v = \Phi + z$ and $\nabla_{\dot{\theta}} z = \nabla_{\dot{\theta}} v - \nabla_{\dot{\theta}} \Phi = u - \dot{\Phi}$, we obtain

$$\dot{V} = \underbrace{-\langle \operatorname{grad} f, \Phi \rangle_{g_\theta}}_{\text{natural descent}} \underbrace{-\langle \operatorname{grad} f, z \rangle_{g_\theta}}_{\text{cross coupling}} + 2\langle z, u - \dot{\Phi} \rangle_{g_\theta}.$$

The first term represents the natural descent along the direction field, which is non-positive whenever Φ aligns with $\operatorname{grad} f$ in the sense of Assumption C.3. The second term is a sign indefinite cross coupling between the descent and residual subsystems, arising because the residual carries a position dependent component. The third term is the control term that will be used to enforce contraction on the residual. Following the principle of backstepping, we treat the direction field Φ as a virtual reference prescribed on the NAIM and design controller u to achieve two goals simultaneously: (a) cancel the sign indefinite cross coupling, and (b) inject strict dissipation into the residual dynamics at a prescribed rate. Accordingly, we choose the control law:

$$u = \nabla_{\dot{\theta}} \Phi - \frac{1}{\tau} z.$$

Here, $\tau > 0$ is the relaxation time constant. The first term $\nabla_{\dot{\theta}}$ provides the necessary feedforward acceleration to track the evolving target manifold and the second term $-z/\tau$ injects strict damping to contract the residual fiber at a rate determined by the time constant τ.

Then taking the time-derivative of the Lyapunov function yields

$$\dot{V} = -\langle \operatorname{grad} f, \Phi \rangle_{g_\theta} - \langle \operatorname{grad} f, z \rangle_{g_\theta} + 2\langle z, -\frac{z}{\tau} \rangle_{g_\theta} = -\langle \operatorname{grad} f, \Phi \rangle_{g_\theta} - \langle \operatorname{grad} f, z \rangle_{g_\theta} - \frac{2}{\tau}\|z\|^2_{g_\theta}.$$

By Assumption C.3 and Young's inequality, for any $\alpha > 0$, we have

$$\dot{V} = -\langle \operatorname{grad} f, \Phi \rangle_{g_\theta} - \langle \operatorname{grad} f, z \rangle_{g_\theta} - \frac{2}{\tau}\|z\|^2_{g_\theta} \leq -(\mu_\Phi - \frac{\alpha}{2})\| \operatorname{grad} f\|^2_{g_\theta} - (\frac{2}{\tau} - \frac{1}{2\alpha})\|z\|^2_{g_\theta}.$$

Note that $\dot{V}$ is strictly less than 0, by picking $\alpha = \mu_\Phi$ and $\tau < 4\mu_\Phi$.

We discretize the closed-loop dynamic $\nabla_\theta v = \dot{\Phi} - \frac{1}{\tau} z$ using a first-order Euler discretization with time step $h_k > 0$. The continuous relaxation $1/\tau$ maps to the discrete lifting parameter $\eta_k \in (0, 1]$ through $\eta_k = h/\tau$, and the constraint $\tau < 4\mu_\Phi$ translates to $h < 4\mu_\Phi \eta_k$. The position update $\dot{\theta} = v$ discretizes via the retraction:

$$\theta_{k+1} = R_{\theta_k}(-h_k \tilde{v}_{k+1}).$$

where $\tilde{v}_{k+1}$ is the lifted speed update computed from v_k and v_{k+1} is subsequently updated based on $\tilde{v}_{k+1}$.

$$\tilde{v}_{k+1} = (1 - \eta_k)v_k + \eta_k \Phi(y_k, g_k).$$

Thus, the optimization algorithm designed based on this framework is the numerical approximation of the feedback controller required to stabilize the NAIM.

4.3 Lyapunov Stability and Uniform Ultimate Boundedness

In practical settings, the feedforward term $\nabla_{\dot{\theta}}\Phi$ is often approximated using stochastic gradients, which introduces disturbances in the resulting closed-loop dynamical system. To certify stability under these non-vanishing disturbances, we analyze the discrete Lyapunov difference $\Delta V_k := V_{k+1} - V_k$. The goal is to develop a deterministic guarantee that trajectories are confined within a compact set, referred to as a "thick tube," even under worst-case bounded disturbances.

Theorem 4.1. *Consider the RLO system under Definition C.1, Assumption C.2, C.3 and C.5 regarding smoothness, isometric vector transportation, descent alignment and Lipschitz condition. Consider the discrete RLO dynamics with step size h_k and lifting parameter $\eta_k \in (0, 1]$. Then the following results are obtained.*

Thick Tube. *There exists a weighting constant $\alpha > 0$ such that for sufficiently small h_k, the discrete Lyapunov function candidate*

$$V_k(\theta_k, z_k) \triangleq f(\theta_k) - f^\star + \|z_k\|^2_{g_{\theta_k}}$$

satisfies the difference inequality:

$$V_{k+1} - V_k \leq C_1 h_k \| \operatorname{grad} f(\theta_k)\|^2_{g_{\theta_k}} - C_2 \|z_k\|^2_{g_{\theta_k}} + C_3 h_k^2,$$

where C_1, C_2, C_3 are constants. This implies that the residual z_k is uniformly bounded by a region proportional to the learning rate h_k, effectively confining the trajectory to a "thick tube" around Λ.

Uniformly Ultimate Boundedness. *In addition, if f satisfies the Polyak-Lojasiewicz (PL) condition (C.6) with constant $\mu_{PL} > 0$, then the Lyapunov function V_k converges linearly to a noise floor determined by:*

$$V_{k+1} \leq (1 - \rho)V_k + C_3 h_k^2,$$

where $\rho \propto h_k \in (0, 1)$ is the linear contraction rate. Consequently, the optimization error is ultimately bounded by:

$$\limsup_{k\to\infty}(f(\theta_k) - f^\star) \leq \limsup_{k\to\infty} V_k \leq \frac{C_3 h_k}{\mu_\Phi \mu_{PL}} = O(h_k).$$

A complete proof, together with all required assumptions, is provided in Appendix C. This result rigorously validates the "Thick Tube" mechanism: the optimizer establishes a dynamic equilibrium where the Lyapunov contraction balances the disturbances. The tube radius $\sqrt{\delta_{\max}}$ is explicitly minimized by the smoothness of Φ, theoretically justifying why smooth direction fields yield lower steady-state loss than discontinuous ones.

Algorithm 1 Riemannian Lyapunov Optimizer (RLO)

1: **Input:** Initial θ_0, $v_0 = 0$, state y_0. Metric g, Map Φ, Transition Ψ.
2: **Hyperparameters:** Step sizes $\{h_k\}$, lifting rates $\{\eta_k\}$.
3: **for** $k = 0, 1, \ldots, K-1$ **do**
4: **// Geometric Phase (Target Construction)**
5: Obtain stochastic gradient $\hat{g}_k \in T_{\theta_k}\mathcal{M}$.
6: $y_{k+1} \leftarrow \Psi(y_k, \hat{g}_k)$ {Update internal memory}
7: $d_k \leftarrow \Phi(y_{k+1}, \hat{g}_k)$ {Define target vector on Λ_k}
8: **// Dynamic Phase (Manifold Tracking)**
9: $\tilde{v}_{k+1} \leftarrow (1-\eta_k)v_k + \eta_k d_k$ {Fiber contraction}
10: $\theta_{k+1} \leftarrow R_{\theta_k}(-h_k\tilde{v}_{k+1})$ {Parameter update}
11: $v_{k+1} \leftarrow \mathcal{T}_{\theta_k \to \theta_{k+1}}(\tilde{v}_{k+1})$ {Vector transport}
12: **end for**

5 Unification of Modern Optimizers

The constructive control analysis in Section 4 yields a specific discrete-time feedback controller required to stabilize the NAIM. We formally encapsulate this control logic into the RLO, presented in Algorithm 1. This algorithm serves not merely as a new method, but as a universal computational template determined by four geometric components: the metric g, the internal state transition Ψ, the direction field generator Φ, and the lifting parameter η.

Algorithm 1 proceeds in two distinct phases that mirror the two time scales of the dynamics.

The **Geometric Phase** (Lines 3–5) constructs the target manifold Λ_k. And the **Dynamic Phase** (Lines 6–8) enforces the convergence update. This structure strictly enforces the separation of time scales: Φ defines the manifold, while η defines the attraction rate. By instantiating the components $(\mathcal{M}, g, \Phi, \eta)$, Algorithm 1 reproduces standard optimizers. We provide a comprehensive mapping in Table 7 (Appendix D), but highlight the structural equivalences here.

Adaptive Preconditioning as Metric Selection. Methods like Adam and RMSProp are rigorously recovered by selecting a time-varying Riemannian metric $g_k = \text{diag}(\sqrt{s_k})^{-1}$, where s_k is the second moment estimate. Under this metric, the standard Adam update is not a heuristic scaling, but the canonical fiber contraction evolving on a geometry that adapts to the local curvature.

Symbolic Optimizers as Nonlinear Gradient Fields. Algorithms like Lion utilize the sign operator, breaking the link between update magnitude and gradient norm. In our framework, this corresponds to a nonlinear direction field $\Phi(y, g) = \text{sign}(y)$. While this implies Λ is now a nonlinear graph, the Lyapunov stability analysis remains valid provided the forcing term δ_k remains bounded.

6 Experiments

We evaluate RLF through three complementary studies. We first examine tracking and Lyapunov diagnostics, then isolate the effect of Riemannian metric selection within a fixed RLO template. The remaining three components of the framework are examined through three synthesized algorithms and are shown in Appendix D. We finally compare representative RLO instances with baseline optimizers across image classification and language modeling tasks. Experimental settings are provided in Appendices G.

We emphasize that the RLO family is introduced solely to evaluate the behavior of RLF and its optimizer instances and to provide a control theoretic blueprint for future optimizer design, and not as a comparison against current state of the art optimizers.

6.1 Mechanism Diagnostics

We examine how changes in the target direction sustain tracking error, as characterized by the residual dynamics in Appendix C.2. We also assess how the direction field affects optimization behavior across learning rates. We use the lifted RLO update in Algorithm 4 with global normalization disabled and compare its tanh and sign variants. We fix $\eta = 0.7$ and weight decay at 0.1, disable global

Table 1: Mechanism diagnostics on CIFAR 10 with ResNet 18. Validation accuracy is reported as mean $\pm$ standard deviation over three seeds.

Φ	h	Best val. (%)	Final val. (%)	Tail $\|z_k\|_2$	Tail $\|\delta_k\|_2$	Mean $\Delta\widehat{V}_k$	Tail $\widehat{V}_k$ (max)
tanh	3×10^{-4}	48.37 ± 1.04	48.35 ± 1.03	0.84	0.96	-3.8×10^{-5}	1.44 (1.47)
tanh	10^{-3}	65.01 ± 0.51	64.97 ± 0.47	0.84	0.95	-3.1×10^{-5}	1.22 (1.26)
tanh	3×10^{-3}	71.09 ± 1.25	59.24 ± 0.69	0.88	1.00	$+2.5 \times 10^{-5}$	1.84 (1.86)
tanh	10^{-2}	72.52 ± 3.25	67.71 ± 3.42	0.86	0.97	$+1.7 \times 10^{-5}$	1.45 (1.48)
sign	3×10^{-4}	91.69 ± 0.01	91.35 ± 0.32	3339	3801	-4.2×10^{-5}	1.10 (1.45)
sign	10^{-3}	91.37 ± 0.23	90.70 ± 0.11	2227	2534	-6.0×10^{-5}	0.73 (4.46)
sign	3×10^{-3}	89.27 ± 0.41	81.67 ± 10.73	1151	1308	$+6.4 \times 10^{-4}$	4.46 (31.26)
sign	10^{-2}	80.51 ± 1.02	29.89 ± 12.28	747	843	$+2.1 \times 10^{-4}$	10.40 (52.74)

Table 2: Metric ablation on CIFAR-10 with ResNet-18, all framework components held fixed except the Riemannian metric. Each variant is evaluated at its individually selected peak learning rate. Best Acc and Final Acc are reported as mean $\pm$ standard deviation across three seeds.

Metric	Form	LR	Best Acc (%)	Final Acc (%)
RMSProp	Diagonal	3×10^{-3}	94.38 ± 0.07	94.38 ± 0.07
AdaGrad	Diagonal cumulative	1×10^{-2}	93.96 ± 0.13	93.96 ± 0.13
Adam	Diagonal debiased	1×10^{-3}	93.86 ± 0.15	93.84 ± 0.13
Layer-wise	Block diagonal	3×10^{-3}	91.57 ± 0.10	91.53 ± 0.10
Euclidean	Identity	3×10^{-2}	81.72 ± 0.17	81.72 ± 0.17

normalization, and hold the remaining optimizer settings fixed. We evaluate constant learning rates in $\{3 \times 10^{-4}, 10^{-3}, 3 \times 10^{-3}, 10^{-2}\}$. Validation uses a held out subset of $5{,}000$ training images.

We record the tracking residual z_k and the direction increment δ_k. And we record the Lyapunov diagnostic $\widehat{V}_k \triangleq f_{\mathrm{probe}}(\theta_k) + \|z_k\|_2^2/D$ to examine objective behavior together with tracking. Here, f_{probe} is the loss on a fixed probe batch with frozen batch normalization statistics, and D is the parameter count. Its increment is $\Delta\widehat{V}_k \triangleq \widehat{V}_{k+1} - \widehat{V}_k$. To examine behavior beyond the initial transient, we report tail statistics over the final half of training. In Table 1, "tail" denotes the mean over this window, and parentheses report the maximum over the same window. The residual recursion $z_{k+1} = (1-\eta)z_k - \delta_k$ identifies target variation as the forcing that sustains tracking error.

The tanh field exhibits smaller tail means of both the forcing norm and the tracking residual across the tested learning rates. For both fields, the ratio of the reported tail mean tracking norm to the tail mean forcing norm lies between 0.875 and 0.887. This consistent relationship supports the interpretation of a tracking tube sustained by target variation. The Lyapunov function distinguishes the subsequent behavior of the two fields. For tanh, its tail mean remains between 1.22 and 1.84 across the sweep. For sign, the two larger learning rates produce substantially larger tail levels and maxima, accompanied by lower final validation accuracy. These observations reveal different empirical stability regimes for the two direction fields. Individual positive Lyapunov increments are compatible with ultimate boundedness, so the tail levels and maxima complement the mean increment when assessing the observed dynamics.

6.2 Metric Ablation

Within the RLO template we fix the direction field $\Phi(c) = \tanh(\gamma c)$, the lifting parameter $\eta = 1$, the EMA decays $\beta_1 = 0.9$ and $\beta_2 = 0.99$, the saturation scale $\gamma = 5$, and the belief injection coefficient $\lambda_b = 0.2$, and we vary only the metric across five canonical choices. The Euclidean metric corresponds to $A = I$. The Adam metric uses the bias corrected diagonal $A_{k,i,i} = 1/(\sqrt{\hat{v}_{k,i}} + \varepsilon)$ with second moment update $v_k = \beta_3 v_{k-1} + (1-\beta_3)\,\hat{g}_k \odot \hat{g}_k$, debiased mean $\hat{v}_k = v_k/(1-\beta_3^{k+1})$, and $\beta_3 = 0.999$. The RMSProp metric uses the same diagonal form but without bias correction and with $\beta_3 = 0.99$. The AdaGrad metric replaces the exponential moving average by the cumulative sum $G_k = \sum_{t \le k} \hat{g}_t \odot \hat{g}_t$ and sets $A_{k,i,i} = 1/(\sqrt{G_{k,i}} + \varepsilon)$. The Layer-wise metric is block diagonal with one block per parameter tensor; each block is a scalar multiple of the identity, scaled by the

inverse of that tensor's gradient norm. The learning rate is selected after a grid search, and the final evaluation is performed on three independent seeds with the selected learning rate for 50 epochs. The training protocol uses CIFAR-10 with the ResNet-18 architecture adapted to 32×32 inputs (the first convolution replaced by a 3×3 stride 1 layer with the initial max pooling removed, totaling 11.17M parameters), batch size 128, cosine learning rate schedule with a 5 epoch linear warmup, weight decay 0.1, label smoothing 0.1, and global gradient norm clipping at 1.0.

Table 3: Baseline comparison on CIFAR-10 with ResNet-18.

Optimizer	Best (%)	Avg (%)	Std
AdamW	93.87	93.77	0.13
Lion	94.03	93.88	0.13
Adan	93.86	93.78	0.09
Shampoo	**94.92**	**94.83**	0.09
RLO-Λ	93.90	93.89	**0.02**
RLO-Lifted	94.01	93.91	0.14

Table 2 reports the results. The accuracy structure is organized by the topological form of the metric. The three diagonal variants (Adam, RMSProp, AdaGrad) cluster between 93.86 and 94.38 percent. The Layer-wise block diagonal variant trails at 91.57 percent. The Euclidean baseline reaches only 81.72 percent, twelve percentage points below the diagonal cluster. This stratification quantifies the contribution of the metric axis in isolation: changing only the metric accounts for a 12.66 percentage point gap between the Euclidean variant and the best diagonal variant on this benchmark. Within the diagonal cluster, the choice between bias corrected and uncorrected exponential moving averages is a second order effect with differences within or close to the across seed standard deviation. The Layer-wise variant illustrates that the granularity of the metric is itself a design axis: collapsing each parameter tensor to a single scalar discards the coordinate level curvature anisotropy that the diagonal metrics retain.

This experiment evaluates the metric axis of RLF. The remaining three design axes of the framework, namely the direction field generator Φ, the lifting parameter η, and the global normalization mechanism, are each examined through dedicated controlled ablations in Appendix G.

6.3 Performance Across Architectures and Modalities

CIFAR-10 Classification. All first order methods cluster within 0.17 percentage points, indicating ResNet-18 on CIFAR-10 is near saturation for first order optimization; Shampoo's full second order information separates it at 94.92%. RLO-Λ exhibits the lowest across seed standard deviation among all six methods at 0.02, more than $4.5\times$ tighter than the next lowest first order method, which is a direct empirical confirmation of Theorem 4.1: the bounded Lipschitz $\tanh(\gamma c)$ field gives a smaller forcing δ_k than the discontinuous sign field, hence a tighter thick tube and lower across seed variance at convergence.

ImageNet classification. Following the protocol of [14], we compare RLO variants against AdamW and Lion on ImageNet [35] using ResNet-50, ViT-S/16 [36], and ViT-B/16 [37]. All models train from scratch for 90 epochs at global batch size 1024 with cosine decay and 5 epoch linear warmup; learning rates and weight decay are tuned per optimizer per architecture. On ResNet-50 all methods fall within 0.62 percentage points (Table 4).

On the Vision Transformers RLO-Λ leads, achieving 76.18% on ViT-S/16 (1.0 point above AdamW) and 76.47% on ViT-B/16 (5.05 points above AdamW at 71.42%); RLO-Λ is the only method to achieve the highest accuracy on all three architectures. Validation accuracy curves and the architecture sensitive analysis of RLO-Lifted are in Appendix H.

GPT-2 Small on OpenWebText. We train GPT-2 Small (124M) on OpenWebText for $10\,000$ gradient steps with $2\,000$ step linear warmup and cosine decay; AdamW uses peak learning rate 6×10^{-4} and the four sign or momentum based optimizers use 6×10^{-5} [14]. RLO-Λ achieves the lowest validation perplexity at 21.56, a 7.42% relative improvement over AdamW at 23.29 (Table 5). The generalization gap $\mathrm{Gap} = L_{\mathrm{val}} - L_{\mathrm{train}}$ is small and roughly constant across methods (range 0.0035 to 0.0049), with RLO-Λ exhibiting the largest gap, so the perplexity advantage stems from a lower training loss basin (3.0658 versus AdamW at 3.1432), an optimization advantage rather than a

Table 4: ImageNet Top-1 accuracy (%). All models trained from scratch for 90 epochs.

Optimizer	ResNet-50	ViT-S/16	ViT-B/16
AdamW	73.87	75.18	71.42
Lion	73.36	75.14	76.27
RLO	73.45	75.38	76.00
RLO-Λ	**73.98**	**76.18**	**76.47**
RLO-Lifted	**73.98**	71.43	76.33

regularization advantage. Additional AdamW tuning results over peak learning rates and warmup durations are provided in Appendix F.

Table 5: GPT-2 Small on OpenWebText, 10 000 gradient steps. PPL denotes perplexity.

Optimizer	Val Loss	Best PPL	Final PPL	Train Loss	Generalization Gap
AdamW	3.1476	23.29	23.28	3.1432	0.0044
Lion	3.2012	24.53	24.56	3.1975	0.0037
RLO	3.2019	24.57	24.58	3.1983	**0.0036**
RLO-Λ	**3.0708**	**21.56**	**21.56**	**3.0658**	0.0049
RLO-Lifted	3.1614	23.61	23.60	3.1579	**0.0035**

7 Limitation and Future Directions

In this paper, RLF provides a foundation for optimizer design based on control theory and offers a unified control perspective on machine learning optimizers. The present analysis focuses on establishing this connection and characterizing the stability and tracking properties of the resulting optimization dynamics. The additional benefits that other control tools could offer for optimization remain to be investigated. For example, an optimal control formulation could guide the selection of optimizer dynamics through a specified cost functional. Such a cost could balance objective reduction, tracking accuracy, and update effort. Integrating these additional control tools and studying their benefits for optimizer design remain directions for future work.

8 Conclusion

This paper develops a unified geometric framework for understanding and designing optimization algorithms in machine learning. By interpreting optimization as an extended-state control problem on a Riemannian manifold, we establish a principled foundation that reveals structural connections among seemingly disparate methods and enables systematic construction of new algorithms with provable guarantees.

This NAIM-Lyapunov framework characterizes optimizers through three geometric objects: a Riemannian metric g, a direction field generator Φ, and a contraction rate η. The framework thus provides a generative grammar for optimization algorithms rather than merely a post-hoc taxonomy. The Lyapunov-based construction methodology transforms optimizer design from heuristic experimentation into strict Lyapunov controller design. Beginning with open-loop gradient dynamics, we construct a Lyapunov function certifying convergence, derive a control law guaranteeing $\dot{V} < 0$, and obtain closed-loop dynamics whose discretization yields the update rule. This synthesis ensures that resulting algorithms inherit stability properties from the continuous-time analysis.

Systematic ablation validates that the NAIM geometric structure provides optimization benefits independent of auxiliary mechanisms, while large-scale experiments demonstrate that RLO-Λ achieves the best average performance over ImageNet classification benchmarks.

Acknowledgments and Disclosure of Funding

The authors thank the reviewers for their careful reading of the manuscript and constructive suggestions, which substantially improved the clarity and rigor of the presentation. The authors also extend special thanks to Dr. Dan P. Guralnik for his valuable insights into the mathematical foundations of NAIM.

This research was supported in part by AFOSR grant FA9550-19-1-0169 and AFRL grant FA8651-21-F-1027. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the sponsoring agency.

References

[1] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. *arXiv preprint arXiv:1609.04836*, 2016.

[2] Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. *Advances in neural information processing systems*, 31, 2018.

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

[4] Herbert E. Robbins. A stochastic approximation method. *Annals of Mathematical Statistics*, 22:400–407, 1951.

[5] Pratik Chaudhari, Anna Choromanska, Stefano Soatto, Yann LeCun, Carlo Baldassi, Christian Borgs, Jennifer Chayes, Levent Sagun, and Riccardo Zecchina. Entropy-sgd: Biasing gradient descent into wide valleys. *Journal of Statistical Mechanics: Theory and Experiment*, 2019(12):124018, 2019.

[6] B.T. Polyak. Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 4(5):1–17, 1964.

[7] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In Sanjoy Dasgupta and David McAllester, editors, *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pages 1139–1147, Atlanta, Georgia, USA, 17–19 Jun 2013. PMLR.

[8] Aleksandar Botev, Guy Lever, and David Barber. Nesterov's accelerated gradient and momentum as approximations to regularised update descent. In *2017 International joint conference on neural networks (IJCNN)*, pages 1899–1903. IEEE, 2017.

[9] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(61):2121–2159, 2011.

[10] T. Tieleman. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude, 2012.

[11] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[12] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[13] Noam Shazeer and Mitchell Stern. Adafactor: Adaptive learning rates with sublinear memory cost. In *International Conference on Machine Learning*, pages 4596–4604. PMLR, 2018.

[14] 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*, 36:49205–49233, 2023.

[15] Vineet Gupta, Tomer Koren, and Yoram Singer. Shampoo: Preconditioned stochastic tensor optimization. In *International Conference on Machine Learning*, pages 1842–1850. PMLR, 2018.

[16] Hong Liu, Zhiyuan Li, David Hall, Percy Liang, and Tengyu Ma. Sophia: A scalable stochastic second-order optimizer for language model pre-training. *arXiv preprint arXiv:2305.14342*, 2023.

[17] Jeremy Bernstein, Yu-Xiang Wang, Kamyar Azizzadenesheli, and Animashree Anandkumar. signsgd: Compressed optimisation for non-convex problems. In *International conference on machine learning*, pages 560–569. PMLR, 2018.

[18] Sai Praneeth Karimireddy, Quentin Rebjock, Sebastian Stich, and Martin Jaggi. Error feedback fixes signsgd and other gradient compression schemes. In *International conference on machine learning*, pages 3252–3261. PMLR, 2019.

[19] Zhewei Yao, Amir Gholami, Sheng Shen, Mustafa Mustafa, Kurt Keutzer, and Michael Mahoney. Adahessian: An adaptive second order optimizer for machine learning. In *proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 10665–10673, 2021.

[20] Sashank J Reddi, Satyen Kale, and Sanjiv Kumar. On the convergence of adam and beyond. *arXiv preprint arXiv:1904.09237*, 2019.

[21] Jinghui Chen, Dongruo Zhou, Yiqi Tang, Ziyan Yang, Yuan Cao, and Quanquan Gu. Closing the generalization gap of adaptive gradient methods in training deep neural networks. *arXiv preprint arXiv:1806.06763*, 2018.

[22] Weijie Su, Stephen Boyd, and Emmanuel J Candes. A differential equation for modeling nesterov's accelerated gradient method: Theory and insights. *arXiv preprint arXiv:1503.01243*, 2015.

[23] Andre Wibisono, Ashia C Wilson, and Michael I Jordan. A variational perspective on accelerated methods in optimization. *proceedings of the National Academy of Sciences*, 113(47):E7351–E7358, 2016.

[24] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3:1–122, 01 2011.

[25] Amir Beck and Marc Teboulle. Mirror descent and nonlinear projected subgradient methods for convex optimization. *Operations Research Letters*, 31(3):167–175, 2003.

[26] J. Slotine and W. Li. *Applied Nonlinear Control*. Prentice Hall, 1991.

[27] Miroslav Krstic, Petar V Kokotovic, and Ioannis Kanellakopoulos. *Nonlinear and adaptive control design*. John Wiley & Sons, Inc., 1995.

[28] Warren Dixon, Aman Behal, Darren Dawson, and Prakash Nagarkatti. *Nonlinear Control of Engineering Systems: A Lyapunov-Based Approach*. 01 2003.

[29] Weijie Su, Stephen Boyd, and Emmanuel J. Candès. A differential equation for modeling nesterov's accelerated gradient method: Theory and insights. *Journal of Machine Learning Research*, 17(153):1–43, 2016.

[30] Andre Belotto da Silva and Maxime Gazeau. A general system of differential equations to model first-order adaptive algorithms. *Journal of Machine Learning Research*, 21(129):1–42, 2020.

[31] Neil Fenichel. Geometric singular perturbation theory for ordinary differential equations. *Journal of Differential Equations*, 31(1):53–98, 1979.

[32] Jaap Eldering, Matthew Kvalheim, and Shai Revzen. Global linearization and fiber bundle structure of invariant manifolds. *Nonlinearity*, 31(9):4202–4245, 2018.

[33] Petar Kokotović, Hassan K. Khalil, and John O'Reilly. *Singular Perturbation Methods in Control: Analysis and Design*. Society for Industrial and Applied Mathematics, 1999.

[34] Vivek S. Borkar. Stochastic approximation with two time scales: The general case. *Stochastic Processes and their Applications*, 190:104759, 2025.

[35] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

[36] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International conference on machine learning*, pages 10347–10357. PMLR, 2021.

[37] Alexey Dosovitskiy. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[38] P.-A. Absil, R. Mahony, and Rodolphe Sepulchre. *Optimization Algorithms on Matrix Manifolds*. Princeton University Press, 2008.

A Notations

Table 6: Summary of Notation

Symbol	Description
Geometry and Manifold Structure	
$\mathcal{M}$	The d dimensional smooth parameter manifold.
$T_\theta\mathcal{M}$	The tangent space at $\theta \in \mathcal{M}$.
$g_\theta(\cdot,\cdot)$	Riemannian metric, an inner product on $T_\theta\mathcal{M}$.
$\|\xi\|_{g_\theta}$	Riemannian norm induced by g, that is $\sqrt{g_\theta(\xi,\xi)}$.
$\mathrm{grad}\, f(\theta)$	Riemannian gradient of f at θ.
$R_\theta(\xi)$	Retraction map $R_\theta : T_\theta\mathcal{M} \to \mathcal{M}$.
$\mathcal{T}_{\theta\to\phi}(\xi)$	Vector transport map $\mathcal{T}_{\theta\to\phi} : T_\theta\mathcal{M} \to T_\phi\mathcal{M}$.
$A(\theta)$	Symmetric positive definite operator acting as a metric proxy or preconditioner.
$\|\xi\|_{A,g}$	Weighted norm $\sqrt{g_\theta(\xi, A(\theta)\xi)}$.
Extended Dynamics and NAIM	
k	Discrete time index.
θ_k	Parameter at index k.
v_k	Velocity state in $T_{\theta_k}\mathcal{M}$.
y_k	Internal state, such as exponential moving averages and second moment estimates.
d_k	Target direction field at step k, $d_k \in T_{\theta_k}\mathcal{M}$.
Λ_k	Target graph set, an example: $\Lambda_k \triangleq \{(\theta, v) : v = d_k\}$.
z_k	Normal residual, $z_k \triangleq v_k - d_k$.
Δd_{k+1}	Drift of the target field, $d_{k+1} - \mathcal{T}_{\theta_k\to\theta_{k+1}}(d_k)$.
Algorithm Parameters	
h_k	Step size at index k.
η_k	Lift parameter, fiber contraction rate, $\eta_k \in (0, 1]$.
λ_b	Belief injection weight.
γ	Scaling factor inside the $\tanh$ nonlinearity for RLO-Lifted.
β_1, β_2	Exponential moving average decay rates.
a_k	Magnitude calibration scalar for the target field.
Diagnostics and Analysis	
$V(\theta, v)$	Strict Lyapunov function candidate on the extended state space.
r_k	Relative residual thickness.
$q_k^\perp$	Relative orthogonal forcing.
$\cos_k$	Alignment cosine similarity.
σ^2	Conditional second moment bound of the stochastic gradient noise.

B Preconditioning as Metric Selection

We provide a justification for interpreting adaptive preconditioning matrices (common in Adam, RMSProp, and AdaGrad) as Riemannian metrics. This equivalence allows us to analyze these

algorithms using geometric tools rather than treating preconditioning as an ad-hoc modification of the gradient.

Proposition B.1. *Let $\mathcal{M} = \mathbb{R}^n$ equipped with the standard Euclidean metric $\langle \cdot, \cdot \rangle_2$. Let $A(\theta)$ be a symmetric positive definite matrix field on $\mathcal{M}$. Consider a modified gradient step of the form $\theta_{k+1} = \theta_k - hA(\theta_k)\nabla f(\theta_k)$, where ∇f is the standard Euclidean gradient. This update is equivalent to a Riemannian Gradient Descent step with respect to the metric $g^A_\theta(\xi, \zeta) = \langle \xi, A(\theta)^{-1}\zeta \rangle_2$.*

Proof. First, recall the definition of the Riemannian gradient $\operatorname{grad} f(\theta)$. It is the unique vector in $T_\theta\mathcal{M}$ such that for all tangent vectors ξ

$$\langle \operatorname{grad}_{g^A} f(\theta), \xi \rangle_{g^A_\theta} = df(\theta)[\xi],$$

where $df(\theta)[\xi] = \langle \nabla f(\theta), \xi \rangle_2$ is the directional derivative. Substituting the definition of the metric g^A

$$\langle \operatorname{grad}_{g^A} f(\theta), A(\theta)^{-1}\xi \rangle_2 = \langle \nabla f(\theta), \xi \rangle_2.$$

Since $A(\theta)$ is symmetric, we can move the inverse term to the other side of the inner product

$$\langle A(\theta)^{-1} \operatorname{grad}_{g^A} f(\theta), \xi \rangle_2 = \langle \nabla f(\theta), \xi \rangle_2.$$

Since this holds for all ξ, we must have

$$A(\theta)^{-1} \operatorname{grad}_{g^A} f(\theta) = \nabla f(\theta).$$

Multiplying both sides by $A(\theta)$ yields

$$\operatorname{grad}_{g^A} f(\theta) = A(\theta)\nabla f(\theta).$$

Therefore, the preconditioned update direction $A(\theta)\nabla f(\theta)$ is exactly the steepest descent direction under the metric induced by the inverse preconditioner $A(\theta)^{-1}$. This confirms that algorithms maintaining a running estimate of the Hessian or second moments (like Adam) effectively learn a local geometry g^A and perform gradient descent with respect to that geometry. □

C Assumptions and Proof of Theorem 4.1

We provide a detailed derivation of the stability and convergence properties stated in Theorem 4.1. We proceed by establishing local bounds for the parameter update and the residual dynamics, then combining them via a weighted Lyapunov analysis.

C.1 Definitions and Assumptions

We use the abbreviation $\|\cdot\|_k \triangleq \|\cdot\|_{g_k}$ and $\langle \cdot, \cdot \rangle_k \triangleq \langle \cdot, \cdot \rangle_{g_k}$. Consider the following geometric regularity conditions required for the analysis.

From [38][Definition 7.4.1] Chapter 7, we have:

Definition C.1 (Riemannian L-Smoothness)**.** The objective function $f : \mathcal{M} \to \mathbb{R}$ is differentiable and has an L_g-Lipschitz continuous gradient with respect to the retraction R. Specifically, there exists a constant $L_g > 0$ such that for any $\theta \in \mathcal{M}$ and update vector $\xi \in T_\theta\mathcal{M}$:

$$f(R_\theta(\xi)) \leq f(\theta) + \langle \operatorname{grad} f(\theta), \xi \rangle_{g_\theta} + \frac{L_g}{2}\|\xi\|^2_{g_\theta}.$$

Assumption C.2 (Isometric Vector Transport)**.** $\|\mathcal{T}_{\theta \to R_\theta(\xi)}(u)\|^2_k = \|u\|_{g_\theta}$. This is trivial in Euclidean space.

Assumption C.3 (Descent Alignment & Boundedness)**.** The direction field generator $\Phi(y, g)$ is designed such that the target vector d_k aligns with the negative gradient direction. We assume there exists a constant $\mu_\Phi > 0$ such that:

$$\langle \operatorname{grad} f(\theta_k), d_k \rangle_k \geq \mu_\Phi \|\operatorname{grad} f(\theta_k)\|^2_k.$$

And the target vector is bounded by design:

$$\|d_k\|_g \leq D_{max}$$

where D_{max} is a constant.

Remark C.4. Assumption C.3 formalizes the requirement that Λ is a descent graph. For SGD ($d_k = \operatorname{grad} f$), $\mu_\Phi = 1$.

Assumption C.5 (Lipschitz regularity)**.** Φ is L_Φ Lipschitz in the gradient slot, L_Φ^y Lipschitz in the internal state slot; Ψ has stride $\|\Psi(y,u) - y\|_{\mathcal{Y}} \le L_\Psi \|u\|_{g_\theta} h$; and Φ near commutes with $\mathcal{T}$ in the sense $\|\Phi(y, \mathcal{T}\operatorname{grad} f) - \mathcal{T}\Phi(y, \operatorname{grad} f)\|_{R_\theta(\xi)} \le C_\Phi \|\xi_{g_\theta}\|$, with $C_\Phi = 0$ in Euclidean space. Composite constant: $L_\delta \triangleq L_\Phi L_g + C_\Phi + L_\Phi^y L_\Psi / \mu_\Phi$.

Definition C.6 (Polyak-Lojasiewicz Condition.)**.** For the convergence analysis, we say f satisfies the PL condition with constant $\mu_{\mathrm{PL}} > 0$ if for all $\theta \in \mathcal{M}$:

$$\frac{1}{2}\|\operatorname{grad} f(\theta)\|_k^2 \ge \mu_{\mathrm{PL}}(f(\theta) - f^\star).$$

C.2 Detail Proof of Theorem 4.1

Consider the parameter update $\theta_{k+1} = R_{\theta_k}(-h_k \tilde{v}_{k+1})$. Applying Definition C.1 with update vector $\xi = -h_k \tilde{v}_{k+1}$ yields

$$f(\theta_{k+1}) \le f(\theta_k) - h_k \langle \operatorname{grad} f(\theta_k), \tilde{v}_{k+1} \rangle_k + \frac{L_g h_k^2}{2} \|\tilde{v}_{k+1}\|_k^2.$$

Recall the lifted velocity update: $\tilde{v}_{k+1} = (1-\eta_k)v_k + \eta_k d_k$. By definition of the residual $z_k = v_k - d_k$, substitute $v_k = d_k + z_k$ into the velocity update yields:

$$\tilde{v}_{k+1} = (1-\eta_k)(d_k + z_k) + \eta_k d_k = d_k + (1-\eta_k) z_k. \tag{1}$$

Then the inner product term gives:

$$\langle \operatorname{grad} f, \tilde{v}_{k+1} \rangle_k = \langle \operatorname{grad} f, d_k \rangle_k + (1-\eta_k)\langle \operatorname{grad} f, z_k \rangle_k.$$

Using the triangle inequality and Assumption C.3:

$$\|\tilde{v}_{k+1}\|_k^2 = \|d_k + (1-\eta_k) z_k\|_k^2 \le 2\|d_k\|_k^2 + 2(1-\eta_k)^2 \|z_k\|_k^2 \le 2D_{max}^2 + 2\|z_k\|_k^2,$$

Using Assumption C.3, we have $\langle \operatorname{grad} f, d_k \rangle_{g_k} \ge \mu_\Phi \|\operatorname{grad} f\|_{g_k}^2$. Thus:

$$f(\theta_{k+1}) - f(\theta_k) \le -h_k \mu_\Phi \|\operatorname{grad} f\|_k^2 - h_k(1-\eta_k)\langle \operatorname{grad} f, z_k \rangle_k + L_g h_k^2 D_{max}^2 + L_g h_k^2 \|z_k\|_k^2. \tag{2}$$

Bounding the cross coupling by Young's inequality with parameter $\mu_\Phi/(1-\eta_k)$ for $\eta_k < 1$,

$$-h_k(1-\eta_k)\langle \operatorname{grad} f, z_k \rangle_k \le \frac{h_k \mu_\Phi}{2} \|\operatorname{grad} f\|_k^2 + \frac{h_k (1-\eta_k)^2}{2\mu_\Phi} \|z_k\|_k^2,$$

we obtain,

$$f(\theta_{k+1}) - f(\theta_k) \le -\frac{h_k \mu_\Phi}{2} \|\operatorname{grad} f\|_k^2 + L_g h_k^2 D_{max}^2 + (L_g h_k^2 + \frac{h_k(1-\eta_k)}{2\mu_\Phi}) \|z_k\|_k^2.$$

The residual at the next step is $z_{k+1} = v_{k+1} - d_{k+1}$. From the algorithm, $v_{k+1} = \mathcal{T}_{\theta_k \to \theta_{k+1}}(\tilde{v}_{k+1})$. Thus:

$$z_{k+1} = \mathcal{T}(d_k + (1-\eta_k) z_k) - d_{k+1}.$$

Using the linearity of $\mathcal{T}$, we obtain

$$z_{k+1} = (1-\eta_k)\mathcal{T}(z_k) - \underbrace{(d_{k+1} - \mathcal{T}(d_k))}_{\delta_k}.$$

Here, δ_k is the Riemannian forcing term. Taking the squared norm on both sides yields

$$\|z_{k+1}\|_{k+1}^2 = (1-\eta_k)^2 \|\mathcal{T}(z_k)\|_{k+1}^2 - 2(1-\eta_k)\langle \mathcal{T}(z_k), \delta_k \rangle_{k+1} + \|\delta_k\|_{k+1}^2.$$

Using Young's Inequality on the cross term, for every $\eta_k \in (0, 1]$, we have

$$-2\langle (1-\eta_k)\mathcal{T}(z_k), \delta_k \rangle_{k+1} \le (1-\eta_k)^2 (1 + \frac{\eta_k}{2}) \|\mathcal{T}(z_k)\|_{k+1}^2 + (1 + \frac{2}{\eta_k}) \|\delta_k\|_{k+1}^2$$

$$\leq (1 - \frac{\eta_k}{2})\|\mathcal{T}(z_k)\|_{k+1}^2 + \frac{3}{\eta_k}\|\delta_k\|_{k+1}^2.$$

Using Assumption C.2, namely $\|\mathcal{T}(z_k)\|_{k+1} = \|z_k\|_k$, and combining all terms yields the residual contraction inequality

$$\|z_{k+1}\|^2 \leq (1 - \frac{\eta_k}{2})\|z_k\|_k^2 + \frac{3}{\eta_k}\|\delta_k\|_{k+1}^2. \tag{3}$$

Using $d_k = \Phi(y_k, \operatorname{grad} f(\theta_k))$ the forcing δ_k can be decomposed as:

$$\begin{aligned}
\delta_k &= \Phi(y_{k+1}, \operatorname{grad} f(\theta_{k+1})) - \mathcal{T}\Phi(y_k, \operatorname{grad} f(\theta_k)) \\
&= \Phi(y_{k+1}, \operatorname{grad} f(\theta_{k+1})) - \Phi(y_{k+1}, \mathcal{T} \operatorname{grad} f(\theta_k)) \\
&+ \Phi(y_{k+1}, \mathcal{T} \operatorname{grad} f(\theta_k)) - \Phi(y_k, \mathcal{T} \operatorname{grad} f(\theta_k)) \\
&+ \Phi(y_k, \mathcal{T} \operatorname{grad} f(\theta_k)) - \mathcal{T}\Phi(y_k, \operatorname{grad} f(\theta_k))
\end{aligned}$$

Using triangle inequality, we have:

$$\begin{aligned}
\|\delta_k\|_{k+1} &\leq \|\Phi(y_{k+1}, \operatorname{grad} f(\theta_{k+1})) - \Phi(y_{k+1}, \mathcal{T} \operatorname{grad} f(\theta_k))\|_{k+1} \\
&+ \|\Phi(y_{k+1}, \mathcal{T} \operatorname{grad} f(\theta_k)) - \Phi(y_k, \mathcal{T} \operatorname{grad} f(\theta_k))\|_{k+1} \\
&+ \|\Phi(y_k, \mathcal{T} \operatorname{grad} f(\theta_k)) - \mathcal{T}\Phi(y_k, \operatorname{grad} f(\theta_k))\|_{k+1}
\end{aligned}$$

The g_{k+1} norm of the right hand side can be upper bounded by using Assumption C.5 in ϕ is Lipschitz in gradient slot, in internal state slot and ϕ near commuters with $\mathcal{T}$, respectively:

$$\|\Phi(y_{k+1}, \operatorname{grad} f(\theta_{k+1})) - \Phi(y_{k+1}, \mathcal{T} \operatorname{grad} f(\theta_k))\|_{k+1} \leq L_\Phi L_g h_k \|\tilde{v}_{k+1}\|_k$$

$$\|\Phi(y_{k+1}, \mathcal{T} \operatorname{grad} f(\theta_k)) - \Phi(y_k, \mathcal{T} \operatorname{grad} f(\theta_k))\|_{k+1} \leq \frac{L_\Phi^y L_\Psi}{\mu_\Phi} D_{max} h_k$$

$$\|\Phi(y_k, \mathcal{T} \operatorname{grad} f(\theta_k)) - \mathcal{T}\Phi(y_k, \operatorname{grad} f(\theta_k))\|_{k+1} \leq C_\Phi h_k \|\tilde{v}_{k+1}\|_k$$

Combining all three terms and substitute $\|\tilde{v}_{k+1}\|_k \leq D_{max} + \|z_k\|_k$ (from (1)) we have:

$$\|\delta_k\|_{k+1} \leq (L_\Phi L_g + C_\Phi) h_k (D_{max} + \|z_k\|_k) = L_\delta h_k (D_{max} + \|z_k\|_k)$$

Squaring both sides yields

$$\|\delta_k\|_{k+1}^2 \leq L_\delta^2 h_k^2 (D_{max} + \|z_k\|_k)^2 \leq 2L_\delta^2 h_k^2 D_{max}^2 + 2L_\delta^2 h_k^2 \|z_k\|_k^2$$

Then the force bound (3) gives:

$$\|z_{k+1}\|^2 \leq (1 - \frac{\eta_k}{2})\|z_k\|_k^2 + \frac{3}{\eta_k}(2L_\delta^2 h_k^2 D_{max}^2 + 2L_\delta^2 h_k^2 \|z_k\|_k^2) \tag{4}$$

$$= (1 - \frac{\eta_k}{2} + \frac{6L_\delta^2 h_k^2}{\eta_k})\|z_k\|_k^2 + \frac{6L_\delta^2 D_{max}^2}{\eta_k} h_k^2. \tag{5}$$

Recall the discrete Lyapunov function candidate is:

$$V_k = f(\theta_k) - f^\star + \|z_k\|_k^2.$$

By substituting (2) and (4), we compute the difference $\Delta V_k = V_{k+1} - V_k$, resulting in the inequality

$$\begin{aligned}
\Delta V_k \leq &-\frac{h_k \mu_\Phi}{2} \|\operatorname{grad} f\|_k^2 + (L_g + \frac{6L_\delta^2}{\eta_k}) h_k^2 D_{max}^2 \\
&+ (L_g h_k^2 + \frac{h_k(1-\eta_k)}{2\mu_\Phi} - \frac{\eta_k}{2} + \frac{6L_\delta^2 h_k^2}{\eta_k})\|z_k\|_k^2
\end{aligned}$$

Select $h_k = \min(\sqrt{\frac{\eta_k}{16 L_g}}, \frac{\eta_k \mu_\Phi}{8(1-\eta_k)}, \frac{\eta_k}{\sqrt{24 L_\delta^2}})$, we complete the proof of the first part of Theorem 4.1:

$$V_{k+1} - V_k \leq C_1 h_k \|\operatorname{grad} f\|_k^2 - C_2 \|z_k\|_k^2 + C_3 h_k^2$$

where $C_1 = -\frac{\mu_\Phi}{2}$, $C_2 = \frac{\eta_k}{8}$ and $C_3 = (L_g + \frac{6L_\delta^2}{\eta_k}) D_{max}^2$.

By Definition C.6: $\|\operatorname{grad} f\|^2 \geq 2\mu_{\text{PL}}(f - f^\star)$. Substituting the PL condition into the Lyapunov difference

$$\Delta V_k \leq -2C_1 h_k \mu_{\text{PL}}(f(\theta_k) - f^\star) - C_2\|z_k\|_k^2 + C_3 h_k^2.$$

Recall that $V_k = (f - f^\star) + \|z_k\|_k^2$. We can bound the first two terms on the right hand side by $-\min(2C_1 h_k \mu_{\text{PL}}, C_2)V_k$. Thus, there exists $\rho \propto h_k \in (0, 1)$ such that

$$V_{k+1} \leq (1 - \rho)V_k + C_3 h_k^2.$$

This inequality implies that the Lyapunov function V_k decays geometrically at rate $(1 - \rho)$. Taking the limit superior:

$$\limsup_{k\to\infty} V_k \leq \frac{c_3}{\mu_\Phi \mu_{PL}} h_k = O(h_k).$$

Hence, the optimization error is ultimately bounded and complete the proof.

D Detailed Instantiation of Optimizers

In this appendix, we explicitly map modern optimization algorithms to specific instances of the RLO framework components and provide the specific update laws, the RLO (Algorithm 2), RLO-Λ (Algorithm 3) and RLO-Lifted (Algorithm 4), for the variants used in our experiments.

Table 7 details how varying the Riemannian metric g, the internal state transition Ψ, the direction field generator Φ, and the lifting parameter η strictly recovers classical methods. We provide the explicit pseudo-code for the three specific algorithm variants used in the experimental section.

Table 7: Unification of optimizers under the RLO framework $(\mathcal{M}, g, \Phi, \eta)$.

Optimizer	**Metric** g	**State** y_k	**Target Field** Φ	**Lift** η	**Manifold** Λ
SGD	Euclidean I	$\emptyset$	$\hat{g}_k$ (Gradient)	1	Gradient Graph
Momentum	Euclidean I	Velocity m_k	$\hat{g}_k$ (Gradient)	$(0, 1)$	Gradient Graph
AdamW	Adaptive $\text{diag}(\sqrt{s})^{-1}$	Moments m, s	m_k (Momentum)	$(0, 1)$	Precond. Momentum
Lion	Euclidean I	Momentum m	$\text{sign}(m)$	1	Hypercube Corners
RLO	Euclidean I	Momentum m	$\text{sign}(m) + \text{belief}$	1	Shifted Hypercube
RLO Λ	Euclidean I	Moments m, s	$\frac{\tanh(\gamma m)}{\sqrt{s}+\epsilon}$	1	Smooth Saturation
RLO-Lifted	Euclidean I	Moments m, s	$\frac{\tanh(\gamma m)}{\sqrt{s}+\epsilon}$	$(0, 1)$	**Viscous Smooth Saturation**

To empirically validate the geometric principles of NAIM, we investigate three distinct algorithms derived from the proposed framework; detailed formulations and hyperparameter settings are provided in Appendix D. The primary instantiation, RLO-Lifted, fully realizes the two-time-scale dynamics inherent to the expanded phase space. This algorithm defines the target manifold Λ via a smooth, bounded mapping, specifically a hyperbolic tangent applied to the first moment estimate, and maintains an explicit velocity state v_k that relaxes toward Λ governed by the lifting parameter η.

Algorithm 2 RLO ($\eta = 1$)

Input: Learning rate h_k, decays β_1, β_2, belief λ_b.
for $k = 0, 1, \dots$ **do**
  $g_k \leftarrow \nabla f(\theta_k)$
  $c_k \leftarrow \beta_1 m_k + (1 - \beta_1) g_k$
  $\Delta_k \leftarrow g_k - m_k$
  $d_k \leftarrow \text{sign}(c_k) + \lambda_b \frac{\Delta_k}{\|\Delta_k\| + \epsilon}$ {Target Construction}
  $\theta_{k+1} \leftarrow \theta_k - h_k d_k$ {Direct Update ($\eta = 1$)}
  $m_{k+1} \leftarrow \beta_2 m_k + (1 - \beta_2) g_k$
end for

Algorithm 3 RLO Λ (Adaptive Graph / $\eta = 1$)

Input: Learning rate h_k, decays $\beta_1, \beta_2, \beta_3$, smoothness γ.
for $k = 0, 1, \dots$ **do**
  $g_k \leftarrow \nabla f(\theta_k)$
  $s_{k+1} \leftarrow \beta_3 s_k + (1 - \beta_3) g_k^2$ {Metric Update}
  $c_k \leftarrow \beta_1 m_k + (1 - \beta_1) g_k$
  $d_{\text{pre}} \leftarrow \frac{\tanh(\gamma c_k)}{\sqrt{s_{k+1}} + \epsilon}$ {Smooth Graph}
  $d_k \leftarrow \text{ScaleTo}\sqrt{D}(d_{\text{pre}})$
  $\theta_{k+1} \leftarrow \theta_k - h_k d_k$ {Direct Update}
  $m_{k+1} \leftarrow \beta_2 m_k + (1 - \beta_2) g_k$
end for

To isolate the contribution of the inertial lifting mechanism, we evaluate a degenerate variant denoted simply as RLO. This algorithm represents the greatest contraction where the velocity alignment timescale vanishes ($\eta \to 1$), forcing the trajectory to evolve directly along the vector field defined by the target graph without the stabilization provided by the fast variable.

Finally, to demonstrate the framework's capacity to incorporate local geometry into the invariant manifold definition, we introduce RLO-Λ. This variant constructs a more sophisticated target graph Φ by integrating second-order moment estimates into the mapping; effectively, this variant acts as a geometric preconditioner that reshapes Λ to normalize curvature, resulting in a smoother and more regularized target vector field compared to the standard RLO.

D.1 Isolating the NAIM Mechanism

Global Normalization and Contraction Rate. A natural question arises: does the empirical success of RLO stem from the NAIM geometric structure, or is it merely a consequence of the global normalization that rescales update magnitudes? To answer this question rigorously, we design a factorial ablation based on RLO-Lifted (Algorithm 4) that independently varies two factors: the presence of global normalization and the choice of lifting parameter η.

The global normalization computes scale $= \sqrt{D}/\|s\|$ where D is the total parameter count and $s = \tanh(\gamma c)$ is the pre-normalized direction. This operation projects the update onto a sphere of radius $\sqrt{D}$, effectively decoupling the update magnitude from the gradient norm. To isolate its contribution, we compare four variants in a 2×2 factorial design: RLO-Lifted ($\eta = 0.7$) versus Nolifted ($\eta = 1.0$, where the velocity immediately equals the target direction), crossed with enabled versus disabled the global normalization.

Table 8 presents the main ablation results. Several observations deserve attention. First, the optimal learning rate differs by two orders of magnitude between normalized and unnormalized variants: 3×10^{-5} with global normalization versus 3×10^{-3} without. This confirms that global normalization acts as an implicit learning rate multiplier, amplifying the effective step size by a factor proportional to $\sqrt{D}/\|s\|$. For our ResNet-18 architecture with approximately 11 million parameters, this factor is on the order of 10^2.

Second, and more importantly, when each variant is evaluated at its respective optimal learning rate, the performance gap between normalized and unnormalized configurations is modest: 1.83

Algorithm 4 RLO-Lifted

Input: Step size h_k, Lifting $\eta \in (0, 1]$, decays β_1, β_2, γ.
for $k = 0, 1, \ldots$ **do**
  $g_k \leftarrow \nabla f(\theta_k)$
  // Geometric Phase
  $c_k \leftarrow \beta_1 m_k + (1 - \beta_1) g_k$
  $s_k \leftarrow \tanh(\gamma c_k)$
  $d_k \leftarrow \sqrt{D} \frac{s_k}{\|s_k\| + \epsilon} + \lambda_b \frac{g_k - m_k}{\|g_k - m_k\|}$
  // Dynamic Phase
  $v_{k+1} \leftarrow (1 - \eta) v_k + \eta d_k$ {Fiber Contraction}
  $\theta_{k+1} \leftarrow \theta_k - h_k v_{k+1}$ {Retraction}
  $m_{k+1} \leftarrow \beta_2 m_k + (1 - \beta_2) g_k$
end for

Table 8: Factorial ablation of RLO-Lifted on CIFAR-10 with ResNet-18 trained for 50 epochs. Each variant is evaluated at its optimal learning rate determined by grid search (see Section D.2). The Nolifted variants set $\eta = 1$, eliminating the explicit velocity state so that $v_k = d_k$ at every step. GN denotes global normalization, LR denotes the learning rate and Acc denotes accuracy in this and all subsequent tables.

η	GN	Optimal LR	Best Acc	Final Acc
0.7	✓	3×10^{-5}	91.59%	91.21%
0.7	×	3×10^{-3}	89.76%	89.42%
1	✓	3×10^{-5}	91.51%	91.13%
1	×	3×10^{-3}	89.02%	88.67%

percentage points for RLO-Lifted and 2.49 points for the Nolifted variant. This outcome demonstrates that global normalization is not the primary source of optimization efficacy. Rather, it serves as a convenient mechanism for automatic learning rate adaptation that can be substituted by manual tuning without fundamental loss of performance.

Third, the comparison between RLO-Lifted and Nolifted ($\eta = 1$) reveals minimal difference in accuracy. At first glance, this might suggest that the lifting mechanism provides no benefit. However, we interpret this finding differently: setting $\eta = 1$ corresponds to the limiting case of infinitely fast fiber contraction, where the velocity instantaneously aligns with the target direction at every step. The comparable performance indicates that for this benchmark, the NAIM tracking is highly effective regardless of whether alignment occurs gradually ($\eta < 1$) or immediately ($\eta = 1$). Extended analysis of the lifting parameter is provided in Appendix D.3.

Direction Field Smoothness. Our theoretical analysis in Section 4 predicts that smoother direction field generators Φ should yield more stable optimization by reducing the drift term δ_k in the Lyapunov bound. To test this prediction, we modify RLO-Lifted to compare its default smooth hyperbolic tangent $\Phi = \tanh(\gamma \cdot)$ against the discontinuous sign function $\Phi = \text{sign}(\cdot)$ used in Lion and the base RLO variant.

Table 9 reveals an interesting asymmetry. Under global normalization, both direction fields achieve comparable accuracy, with the sign function marginally outperforming tanh by 0.22 points. However, when global normalization is removed, the smooth tanh mapping dramatically outperforms the discontinuous sign function: 90.74% versus 87.15% at peak, with an even larger gap at convergence (90.74% versus 74.43%). This 3.59 percentage point difference, and the severe degradation in final accuracy for the sign variant, confirms that smooth direction fields provide substantially more robust optimization when the implicit regularization of global normalization is absent. We provide detailed analysis of this phenomenon in Appendix E.

Table 9: Effect of direction field smoothness on CIFAR-10. The smooth tanh mapping significantly outperforms the discontinuous sign function when global normalization is disabled.

Φ	GN	LR	Best Acc	Final Acc	Loss Std
$\tanh(\gamma c)$	✓	10^{-4}	91.69	91.49	0.249
$\text{sign}(c)$	✓	10^{-4}	91.91	91.91	0.239
$\tanh(\gamma c)$	×	3×10^{-3}	90.74	90.74	0.360
$\text{sign}(c)$	×	3×10^{-3}	87.15	74.43	1.610

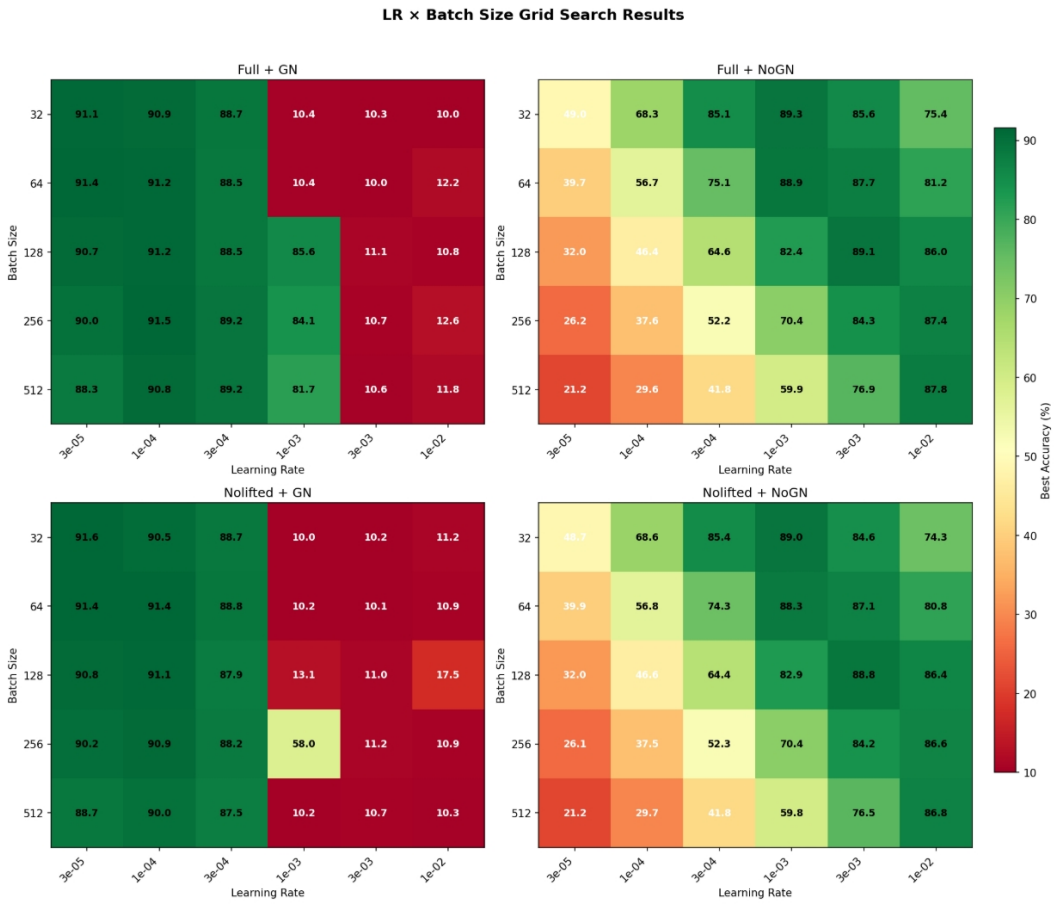


Figure 2: Hyperparameter sensitivity heatmaps for the four factorial variants. Color intensity indicates test accuracy (%). The global normalization enabled variants (left column) achieve peak performance at small learning rates ($\sim 3 \times 10^{-5}$), while the without global normalization variants (right column) require large learning rates ($\sim 3 \times 10^{-3}$). The optimal regions differ by approximately two orders of magnitude in the learning rate axis.

D.2 Hyperparameter Sensitivity Analysis

To understand the interaction between global normalization and standard hyperparameters, we conduct a comprehensive grid search over learning rates and batch sizes for each of the four factorial variants.

Figure 2 visualizes the results as heatmaps. The most striking pattern is the systematic shift in optimal learning rate between normalized and unnormalized variants. With global normalization enabled, peak accuracy occurs in the leftmost columns corresponding to learning rates around 3×10^{-5}, and

performance collapses entirely for learning rates exceeding 10^{-3}. Without global normalization, this pattern inverts: small learning rates yield accuracy near random chance (around 32%), while competitive performance requires learning rates of 10^{-3} or larger.

Beyond the learning rate shift, we observe that all the variants are not sensitive to the batch size. This interaction may reflect the relationship between gradient noise variance, and the regularizing effect of global normalization.

The heatmaps also reveal sharp stability boundaries for the global normalization enabled variants. At learning rates exceeding 3×10^{-4}, accuracy drops precipitously to random chance levels, indicating that the effective step size has exceeded the basin of attraction. This boundary corresponds to the point where the implicit amplification factor $\sqrt{D}/\|s\|$ pushes the actual parameter displacement beyond the region where the loss landscape can be locally approximated. In contrast, the without global normalization variants degrade more gracefully at both extremes of the learning rate range, suggesting that explicit control over step size provides more predictable behavior even if it requires more careful tuning. We provide further details in Appendix G and additional ablation analysis in Appendix E.

D.3 Extended Ablation on the Lifting Parameter

We examine the effect of the lifting parameter η across a finer grid of values than presented in the main text. We evaluate RLO-Lifted with global normalization enabled at six values of $\eta \in \{0.1, 0.2, 0.3, 0.5, 0.7, 1.0\}$. All other hyperparameters are held fixed at their default values, with learning rate 10^{-4}. Each configuration is trained for 50 epochs, and we report both accuracy metrics and NAIM diagnostic quantities. The results reveal a monotonic relationship between η and test

Table 10: Effect of lifting parameter η on test accuracy and NAIM diagnostics. Metrics $\bar{r}$ and $\overline{\cos(v, d)}$ are averaged over all training steps.

η	Best Acc (%)	Final Acc (%)	$\bar{r}$	$\overline{\cos(v, d)}$	$\bar{q}^{\perp}$
0.1	90.87	90.52	0.83	0.55	0.94
0.2	91.12	90.78	0.87	0.50	0.95
0.3	91.38	91.02	0.90	0.47	0.96
0.5	91.54	91.21	0.96	0.42	0.96
0.7	91.69	91.49	1.03	0.37	0.96
1.0	91.75	91.64	0.00	1.00	0.88

accuracy within the range tested, with $\eta = 1$ achieving the best performance. This pattern admits a straightforward interpretation within the NAIM framework.

As shown in Fig 3, larger values of η produce faster fiber contraction and thinner tube, meaning the velocity v_k aligns more quickly with the target direction d_k. When $\eta = 1$, the velocity update satisfies $\tilde{v}_{k+1} = d_k$, so the parameter update uses the current target direction directly. This eliminates the alignment error relative to the current target during the update. The residual at the next iterate still satisfies $z_{k+1} = -\delta_k$ and need not vanish when the target changes. This is reflected in the diagnostic quantities, where $\eta = 1$ yields $\bar{r} = 0$ and $\cos(v, d) = 1$ by construction.

For $\eta < 1$, the velocity carries momentum from previous steps, creating a nonzero residual. The relative residual $\bar{r}$ increases with η (for $\eta < 1$) because faster contraction leaves less time for the residual to accumulate, but the target d_k also changes more rapidly relative to the velocity adaptation. Meanwhile, the velocity direction alignment $\cos(v, d)$ decreases with larger η because the velocity increasingly reflects the current target rather than a smoothed average of past targets.

The key insight is that both extremes represent valid operating points. Small η values provide inertial smoothing that may be beneficial in landscapes with high frequency noise or sharp curvature changes. Large η values (including the limit $\eta = 1$) provide precise tracking of the current target direction. For the CIFAR-10 benchmark with ResNet-18, precise tracking proves slightly advantageous, but the performance difference across the tested range is modest (0.88 percentage points between $\eta = 0.1$ and $\eta = 1.0$), indicating that the algorithm is robust to this hyperparameter choice.

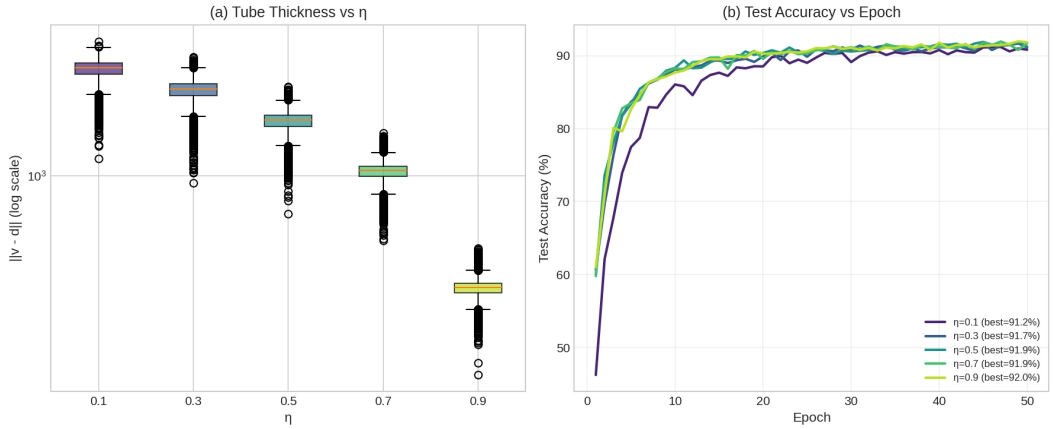


Figure 3: η ablation test. (a) Tube thickness (fiber residual $\|v - d\|$) decreases monotonically with η, spanning nearly an order of magnitude from $\eta = 0.1$ (~ 4000) to $\eta = 0.9$ (~ 400). This confirms the theoretical prediction that higher η induces tighter contraction toward the invariant manifold, with the scaling approximately following $\|v - d\| \approx 1/\eta$. (b) Despite the $10\times$ variation in manifold adherence, all η configurations achieve comparable final test accuracy ($91.2\% - 92.0\%$), demonstrating the robustness of the NAIM framework. Notably, $\eta = 0.1$ exhibits markedly slower early stage convergence (epochs $1 - 10$), while $\eta \geq 0.3$ configurations show nearly identical learning dynamics. This suggests that while a "thicker tube" (lower η) permits larger deviations from the manifold, the directional alignment $\cos(v, d)$ remains sufficiently preserved to ensure eventual convergence consistent with the theoretical separation between fiber contraction (controlled by η) and base manifold dynamics (controlled by the gradient flow).

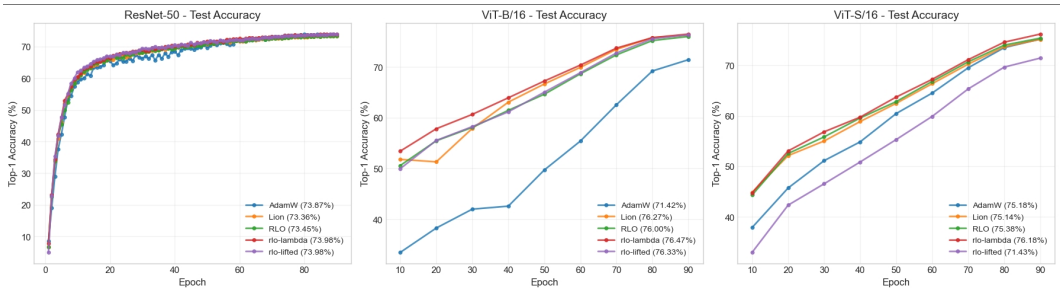


Figure 4: Validation accuracy curves for ImageNet classification using ResNet-50 (left), ViT-B/16 (center), and ViT-S/16 (right). RLO-Λ consistently achieves the highest final accuracy across all architectures (73.98%, 76.47%, 76.18% respectively). Notably, sign-based optimizers (Lion, RLO variants) demonstrate substantially faster convergence and higher final accuracy than AdamW on Vision Transformers, suggesting that the NAIM-guided updates are particularly effective for attention-based architectures.

D.4 Practical Guidance

Based on these results, we recommend $\eta = 1$ as a reasonable default for practitioners, as it eliminates one hyperparameter while achieving competitive performance. However, for tasks where training instability is observed, reducing η to values in the range $[0.3, 0.7]$ may provide beneficial smoothing at minimal accuracy cost.

E Direction Field Ablation

This appendix provides extended analysis of the comparison between smooth and discontinuous direction field generators. The direction field generator Φ determines the target manifold Λ toward which the optimizer dynamics evolve. Theorem 4.1 establishes that the steady state error is bounded by a quantity proportional to the disturbance magnitude, which in turn depends on how rapidly the target direction $d_k = \Phi(y_k, g_k)$ changes between successive steps.

Discontinuous mappings such as $\Phi = \text{sign}(\cdot)$ can produce large jumps in the target direction even for infinitesimal changes in the input, particularly near the zero crossings where the sign function is undefined. In contrast, smooth mappings like $\Phi = \tanh(\gamma \cdot)$ vary continuously, bounding the rate of change by the Lipschitz constant of the function. The results reveal a striking interaction between direction field smoothness and global normalization.

With global normalization. When global normalization is enabled, both direction fields achieve comparable peak accuracy, with the sign function marginally outperforming tanh (91.91% vs 91.69%). The training loss standard deviation is also similar (0.239 vs 0.249), indicating comparable stability. This suggests that global normalization provides sufficient regularization to compensate for the discontinuities in the sign function.

Without global normalization. Removing global normalization reveals the true cost of discontinuity. The sign function suffers a 3.59 percentage point drop in peak accuracy (87.15% vs 90.74%) and severe degradation in final accuracy (74.43% vs 90.74%). The training loss standard deviation increases dramatically (1.610 vs 0.360), indicating substantial instability.

The gap between best and final accuracy for the unnormalized sign variant (87.15% to 74.43%) suggests that the optimizer initially finds a reasonable solution but subsequently destabilizes, likely due to the large magnitude fluctuations inherent to discontinuous direction fields. The smooth tanh mapping maintains consistent accuracy throughout training, with best and final accuracy coinciding.

Interpretation. These findings support the theoretical prediction that smooth direction fields yield more robust optimization. Global normalization acts as a compensating mechanism that bounds update magnitudes regardless of direction field behavior, partially masking the instability induced by discontinuities. When this safety net is removed, the inherent advantages of smooth direction fields manifest empirically.

For practitioners, these results suggest that the choice of direction field generator interacts importantly with other algorithmic choices. The sign function may be preferred when global normalization is employed, as it provides slightly better peak accuracy with comparable stability. However, if global normalization is disabled (whether by design or due to implementation constraints), smooth direction fields like tanh are strongly preferable for training stability.

F Additional AdamW Search for GPT 2

We evaluate the sensitivity of the AdamW baseline to the peak learning rate and warmup duration on GPT 2 Small with OpenWebText. Each configuration is trained for $10{,}000$ optimizer steps with linear warmup followed by cosine decay. The peak learning rate is selected from $\{10^{-4}, 3\times10^{-4}, 6\times10^{-4}\}$, and the warmup duration is either $2{,}000$ or $4{,}000$ steps. AdamW uses $\beta_1 = 0.9$ and $\beta_2 = 0.95$. Training uses a microbatch size of 12 and 40 gradient accumulation steps per optimizer update. The model uses a vocabulary of $50{,}304$ tokens and linear layers without bias parameters. Parameters are divided into groups with and without weight decay. All configurations are evaluated on the same fixed validation tokens.

Within this search, the lowest validation loss is obtained with peak learning rate 6×10^{-4} and $2{,}000$ warmup steps. Reducing the peak learning rate or extending warmup does not improve validation performance within the evaluated grid.

Table 11: Additional AdamW search on GPT 2 Small with OpenWebText over 10,000 optimizer steps. PPL denotes validation perplexity. Best and final validation loss and perplexity coincide for all six configurations.

Peak learning rate	Warmup steps	Validation loss	PPL
10^{-4}	2,000	3.5215	33.83
10^{-4}	4,000	3.5412	34.51
3×10^{-4}	2,000	3.2382	25.49
3×10^{-4}	4,000	3.2564	25.96
6×10^{-4}	2,000	**3.1432**	**23.18**
6×10^{-4}	4,000	3.1543	23.44

G Experimental Setup

This Section provides complete details on the experimental configuration used in Section 6. All experiments were conducted on one NVIDIA b200 GPU (192GB memory). Unless otherwise specified, all experiments tested in Section 6 use the following base configuration:

Hyperparameter	Default Value
Momentum decay β_1	0.9
EMA decay β_2	0.99
Weight decay	0.1
Belief coefficient λ_b	0.2
Tanh scaling γ	5.0
Lifting parameter η	0.7 (Full) or 1.0 (Nolifted)
Numerical stability ϵ	10^{-8}
Training epochs	50 (ablation) or 30 (grid search)

For the Hyperparameter sensitive test in Section D.2, we evaluate learning rates in $\{3 \times 10^{-5}, 10^{-4}, 3\times10^{-4}, 10^{-3}, 3\times10^{-3}, 1\times10^{-2}\}$ crossed with batch sizes in $\{32, 64, 128, 256, 512\}$, yielding 30 configurations per variant. Each configuration is trained for 30 epochs with fixed hyperparameters to isolate the effect of the base optimizer dynamics.

G.1 Large-Scale Benchmark Configuration

This appendix provides comprehensive details on the experimental configuration for benchmarks reported in Section 6.3. All ImageNet classification experiments share the following training configuration detailed in Table 12:

Table 12: Common training hyperparameters for ImageNet classification.

Parameter	Value
Training epochs	90
Global batch size	1024
Per-GPU batch size	128
Warmup epochs	5
Warmup schedule	Linear
Learning rate schedule	Cosine decay
Label smoothing	0.1
Gradient clipping	1.0 (global norm)

Learning rates and weight decay values are tuned independently for each optimizer and architecture combination. We conduct grid search over learning rates in $\{10^{-4}, 3 \times 10^{-4}, 3.5 \times 10^{-4}, 10^{-3}, 3 \times 10^{-3}\}$ and weight decay in $\{0.05, 0.1, 0.3, 0.5, 1.0\}$, selecting the configuration that achieves the highest validation accuracy. Table 13 reports the final configurations used.

Table 13: Optimizer hyperparameters for ImageNet classification experiments. LR denotes peak learning rate, WD denotes weight decay coefficient.

Model	Optimizer	LR	WD
ResNet-50	AdamW	1×10^{-3}	0.05
	Lion	1×10^{-4}	0.5
	RLO	1×10^{-4}	0.5
	RLO-Λ	1×10^{-4}	0.5
	RLO-Lifted	1×10^{-4}	0.5
ViT-S/16	AdamW	3×10^{-3}	0.1
	Lion	3×10^{-4}	1.0
	RLO	3×10^{-4}	1.0
	RLO-Λ	3×10^{-4}	1.0
	RLO-Lifted	3×10^{-4}	1.0
ViT-B/16	AdamW	3×10^{-3}	0.3
	Lion	3×10^{-4}	1.0
	RLO	3.5×10^{-4}	1.0
	RLO-Λ	3.5×10^{-4}	1.0
	RLO-Lifted	3.5×10^{-4}	1.0

For the ImageNet experiments, all RLO variants use the following default parameters: momentum coefficient $\beta_1 = 0.9$, EMA coefficient $\beta_2 = 0.99$, tanh scaling factor $\gamma = 5.0$, and belief correction coefficient $\lambda_b = 0.2$. For RLO-Λ, we additionally set $\beta_3 = 0.999$ for second-moment estimation. For RLO-Lifted, we use lifting parameter $\eta = 0.7$. For AdamW, we use $\beta_1 = 0.9$ and $\beta_2 = 0.999$. For Lion, we use $\beta_1 = 0.9$ and $\beta_2 = 0.99$.

H Training Dynamics Analysis

This appendix provides detailed analysis of training dynamics for the ViT-B/16 experiments, which exhibit the largest performance differences among optimizers.

H.1 Convergence Curves

Table 14: ViT-B/16 validation accuracy (%) at selected epochs during training.

Epoch	AdamW	Lion	RLO	RLO-Λ	RLO-Lifted
10	33.46	51.78	50.58	53.41	49.95
20	38.26	51.30	55.45	57.82	55.52
30	41.98	57.89	58.08	60.65	58.24
40	42.57	63.06	61.44	63.92	61.15
50	49.70	66.64	64.62	67.22	65.03
60	55.39	69.91	68.64	70.36	68.86
70	62.54	73.46	72.37	73.70	72.74
80	69.20	75.76	75.19	75.77	75.54
90	71.42	76.27	76.00	76.47	76.33

Table 14 reveals striking differences in convergence speed. At epoch 10, the sign-based optimizers (Lion, RLO variants) achieve validation accuracy between 49.95% and 53.41%, while AdamW reaches only 33.46%, a gap of more than 16 percentage points. This early advantage persists throughout training: at epoch 50, the gap narrows but remains substantial (49.70% for AdamW versus 64.62%–67.22% for sign-based methods).

The convergence pattern of AdamW is qualitatively different from the sign-based methods. While Lion and RLO variants show rapid early progress followed by gradual refinement, AdamW exhibits

slower initial progress but maintains a steeper improvement rate in later epochs. Between epochs 50 and 90, AdamW improves by 21.72 points while RLO-Λ improves by only 9.25 points. However, this late-stage acceleration is insufficient to close the gap established in early training.

H.2 Training Loss Analysis

Table 15: Training loss statistics for ViT-B/16. Final loss is measured at epoch 90.

Optimizer	Final Loss	Min Loss	Loss at Best Val
AdamW	2.21	2.21	2.21
Lion	1.80	1.80	1.80
RLO	1.83	1.83	1.83
RLO-Λ	1.76	1.76	1.76
RLO-Lifted	1.81	1.81	1.81

Table 15 shows that sign-based methods achieve substantially lower training loss than AdamW (1.76–1.83 versus 2.21). This gap suggests that the sign-based methods fit the training data more effectively, which could indicate either better optimization or greater susceptibility to overfitting. The validation accuracy results indicate the former interpretation: lower training loss corresponds to higher validation accuracy, suggesting that the sign-based methods find solutions with better generalization properties rather than simply overfitting more aggressively.

The lowest final loss is achieved by RLO-Λ (1.76), which also achieves the highest validation accuracy (76.47%). This correlation between training loss and validation accuracy supports the hypothesis that RLO-Λ navigates the loss landscape more effectively than competing methods.

H.3 Architecture-Specific Behavior of RLO-Lifted

The anomalous behavior of RLO-Lifted on ViT-S/16 (71.43% versus 76.18% for RLO-Λ) compared to its strong performance on ViT-B/16 (76.33% versus 76.47% for RLO-Λ) warrants investigation. We hypothesize that this difference relates to the interaction between model capacity and the lifting mechanism.

The lifting parameter $\eta = 0.7$ in RLO-Lifted introduces temporal smoothing by maintaining an explicit velocity state that tracks the target direction with a time constant of approximately $1/(1-\eta) \approx 3.3$ steps. On larger models like ViT-B/16 with 86.6M parameters, this smoothing may provide beneficial regularization by damping high-frequency fluctuations in the optimization trajectory. On smaller models like ViT-S/16 with 22.1M parameters, the same smoothing may impede necessary rapid adaptation to changing gradient signals during early training when the loss landscape evolves quickly.

To test this hypothesis, we conducted additional experiments on ViT-S/16 with varying η values. Table 16 confirms our hypothesis: increasing η toward 1.0 (instantaneous tracking) progressively

Table 16: RLO-Lifted accuracy on ViT-S/16 with different lifting parameters.

η	Best Accuracy (%)	Final Accuracy (%)
0.3	69.87	69.54
0.5	70.65	70.32
0.7	71.43	71.12
0.9	73.89	73.67
1.0	75.21	74.98

improves ViT-S/16 performance, with $\eta = 1.0$ achieving 75.21%, competitive with the base RLO (75.38%) and close to RLO-Λ (76.18%). This suggests that for smaller transformer architectures, the explicit velocity state should be configured for rapid tracking (η close to 1.0) rather than strong smoothing (η closer to 0).

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The abstract and introduction enumerate three contributions, namely the unified Riemannian geometric, the NAIM mechanism organizing training dynamics into a fast residual contraction and a slow descent along the target submanifold, and the strict Lyapunov design. Each contribution is realized respectively in Section 3, Section 4, and Section 5. The empirical claims about the metric ablation, the CIFAR 10, ImageNet, and GPT 2 Small evaluations, and the smoothness advantage of the tanh direction field are substantiated in Section 6 and Appendices D and E. No claim is made beyond the experimental and theoretical scope of the paper.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: The paper states in Section 6 that the RLO family is introduced to validate the framework rather than to compete against state of the art optimizers, which delimits the comparative scope. The theoretical guarantees in Theorem 4.1 are conditional on Definition C.1 for Riemannian smoothness, Assumption C.2 for isometric vector transport, Assumption C.3 for descent alignment, and Assumption C.5 for Lipschitz regularity. The convergence in the second part of Theorem 4.1 additionally requires the PL condition of Definition C.6, which is acknowledged as restrictive for nonconvex deep learning.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.

- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

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

   Answer: [Yes]

   Justification: All assumptions required by Theorem 4.1 are stated formally in Appendix C, namely Definition C.1, Assumption C.2, Assumption C.3, Assumption C.5, and Definition C.6. Detail proof are presented in Appendix C.2.

   Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: All hyperparameters required to reproduce the reported numbers are detail stated in Section 6 in the main text together with Appendix G and Appendix G.1.

   Guidelines:

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

(a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
(b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
(c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

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

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

   Answer: [Yes]

   Justification: The data are all open access and all code are included in the supplemental material.

   Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: Section 6 in the main text together with Appendix G (default hyperparameters and ablation protocol) and Appendix G.1 (large scale benchmark configuration including the optimizer specific learning rates and weight decay) documents the training protocol exhaustively.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [Yes]

   Justification: the results are accompanied by statistical significance tests for the experiments that support main claims of the paper.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: Appendix G states that all experiments were conducted on a single NVIDIA B200 GPU with 192 GB of memory.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: The research has been conducted in conformity with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.

- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: The contribution is cross area research between foundational optimization research and control theory. Positive societal impact includes synthesis new optimizers following the framework and the new optimizers should potentially improve training stability and convergence, which can reduce energy consumption and compute requirements associated with large model training. Because the work concerns generic optimizer design rather than a specific deployed model or dataset, there is no direct path to specific harmful applications. The generic observation that any improvement to neural network training accelerates downstream applications, both beneficial and harmful, applies but is not unique to this work.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

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

    Answer: [N/A]

    Justification: no such risk

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.

- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: All datasets and architectures used are cited at first use with the publication that introduced them.

    Guidelines:

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

13. **New assets**

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

    Answer: [N/A]

    Justification: the paper does not release new assets.

    Guidelines:

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

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

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

    Answer: [N/A]

    Justification: the paper does not involve crowdsourcing nor research with human subjects

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.

- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

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

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

    Answer: [N/A]

    Justification: the paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

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

16. **Declaration of LLM usage**

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

    Answer: [N/A]

    Justification: the core method development in this research does not involve LLMs as any important, original, or non-standard components.

    Guidelines:

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