Записки научных
семинаров ПОМИ
Том 552, 2026 г.

S. Elistratov, A. Podivilov, T. Iuzhakov, D. Vetrov

DIRECTIONAL NOISE AND IMPLICIT BIAS IN LION

Abstract. Lion is a novel optimization method that has outperformed traditional optimizers like Adam across a variety of tasks. Despite its empirical success, the reasons behind Lion's superiority remain unclear. In this paper, we investigate the mechanisms contributing to Lion's enhanced performance, focusing on the structured noise introduced by the use of the sign function in gradient updates. We characterize this noise by the angle of rotation between a vector and its signum. We inject this noise as a random fixed-angle rotation into normalized updates and analyze how the performance of this method compares to that of Lion. We also provide a theoretical analysis showing how this directional noise modifies the optimizer's implicit bias by shifting the induced sharpness/curvature penalty from the gradient direction toward the orthogonal component. We demonstrate that this method has stronger performance than Lion in our setting. This approach reveals a relationship between the learning rate and the noise specific to the Lion method, providing insights into its improved performance metrics. Additionally, we identify an effect we term "momentum tracing" in neural networks with normalization layers and ReLU activations, which can significantly destabilize the training process. Our analysis demonstrates that the rotation noise inherent in Lion mitigates the negative impact of "momentum tracing", leading to more stable learning. These findings offer theoretical justification for Lion's effectiveness and suggest avenues for developing more robust optimization algorithms.

§1. Introduction

Modern optimization methods leverage various properties of neural networks to enhance convergence toward solutions with superior performance. The loss landscape of neural networks possesses distinct characteristics that define criteria for effective optimization, and a substantial body of

---

* Equal contribution.

† Shared senior authorship.

Key words and phrases: optimization in deep learning, loss landscape.

work links these characteristics to generalization and implicit regularization. In this view, the optimizer is not merely a computational primitive: by shaping training dynamics, it influences which regions of parameter space are reached and which solutions ultimately generalize.

One of the earliest and most widely adopted optimizers specifically developed for neural networks is Adam [12]. This method utilizes first and second-order momentum estimates to stabilize the training process. Several studies have demonstrated how these mechanisms help the optimizer converge to regions that offer better generalization [18, 7, 4]. In addition to adaptive methods, fixed-step approaches that modify the update geometry have also been explored for neural networks. Norm (S)GD normalizes the gradient, optimizing in its direction at each iteration with a step size equal to the learning rate [14, 16, 23]. Another method, Sign (S)GD, applies the sign function to the update direction. This results in each parameter being adjusted by $\pm$ the learning rate, depending on the sign of the corresponding gradient component [15, 13, 2, 20]. However, these methods have shown inferior performance and have not gained widespread adoption, although they have been utilized in other learning paradigms.

A novel method, Lion [6], was developed through programmatic search within a symbolic representation of optimizer space. Lion integrates various mechanisms from earlier methods, such as the sign function, decoupled weight decay, momentum, and decoupling momentum tracking from the parameter update. The authors demonstrated that Lion outperforms Adam across a broad spectrum of CV and NLP tasks. However, due to Lion's novelty, there remains a limited systematic understanding of *why* its update rule can be advantageous and in which regimes its design choices improve stability and generalization.

The use of the sign function in Lion leads to optimization that does not strictly follow the true gradient direction. This can be interpreted as the introduction of a specific type of noise into the optimization process. While optimization with a fixed step size in the gradient direction, as in Norm SGD, may seem more logical, it has been observed to result in poorer model performance. This paper aims to explore why Lion exhibits superior performance and how the noise introduced by perturbing the gradient direction helps stabilize the learning process.

We characterize this noise via the angle between the gradient and its sign. Such noise can be synthetically introduced into the Norm SGD optimizer by applying random rotations with a fixed angle to the gradient

update as a form of regularization. Moreover, we theoretically justify how the noise can impact implicit bias properties of the optimizer. This characterization allows us to identify a relationship between the learning rate and the effective noise level specific to Lion, and to explain why Lion may achieve better performance despite using a direction that is only a coarse proxy for the gradient.

Furthermore, we identify an effect that we term *momentum tracing*, observed in both Lion and Norm SGD when they are applied to neural networks that include normalization layers and ReLU (or GELU) activations. This effect can lead to destabilization of the training process. Our study investigates how the structured noise introduced by Lion mitigates the negative impact of this effect, contributing to more stable learning.

An earlier version of this work appeared in the NeurIPS Workshop on Optimization for Machine Learning (OPT 2024)[1]. The present paper substantially extends that version with additional theoretical analysis.

§2. Background

Optimization in deep learning is complicated by highly nonconvex loss landscapes that exhibit scale symmetries and flat directions (especially in the presence of normalization), as well as by the use of stochastic gradients. As a result, the choice of optimizer and update geometry can significantly affect both optimization speed and the implicit bias toward solutions with better generalization. Classical momentum methods and adaptive-moment methods such as Adam [12] are widely used because they stabilize training through temporal averaging; in adaptive methods, this is complemented by coordinate-wise scaling. In contrast, a different line of work modifies the update geometry more directly, for instance by normalizing gradients or by applying quantization-like operators that alter the update direction. Lion optimizer. Lion [6] is a recent optimizer discovered by programmatic search in the symbolic space of update rules. It incorporates several mechanisms that improve optimization dynamics, including decoupled weight decay and a sign-based update direction, and it decouples momentum tracking from the parameter update.

[1] `https://opt-ml.org/papers/2024/paper128.pdf`

Definition 1. *The update rule of Lion is*

$$\begin{cases} m_{t+1} = \beta_2 m_t + (1-\beta_2)\nabla L(w_t), \\ w_{t+1} = (1-\eta\lambda)\, w_t + \eta\, \operatorname{sign}\big(\beta_1 m_{t+1} + (1-\beta_1)\nabla L(w_t)\big), \end{cases} \tag{1}$$

where η is the learning rate, λ is the weight decay coefficient, m_t denotes the momentum, and w_t are the network parameters.

Empirically, the original Lion study reports improvements over Adam across a broad range of vision and language tasks [6]. The update is also computationally lightweight, since the final direction is a component-wise sign map and does not require second-moment accumulation.

Lion is naturally related to fixed-step methods that use a transformed gradient direction. In Sign (S)GD, the update direction is $\operatorname{sign}(\nabla L(w_t))$, which results in coordinate-wise steps of magnitude η and is often motivated by robustness and communication efficiency [15, 13, 2, 20]. In Norm (S)GD, the gradient is normalized and the iterate moves in the normalized gradient direction with step size η [14, 16, 23]. These methods explicitly decouple step length from gradient magnitude, but may also introduce a mismatch between the true gradient direction and the applied update, which can be understood as a form of structured perturbation to descent dynamics.

Recent follow-up work has begun to reduce the gap between Lion's empirical success and its theoretical understanding. The authors of the paper [8] analyze Lion through a constrained-optimization lens and derive explicit stochastic nonconvex convergence-rate guarantees with dimension dependence. The paper [11] extend the convergence analysis to both centralized and distributed regimes, and study variance-reduced Lion variants with improved rates; they additionally analyze communication-efficient distributed implementations that employ 1-bit (sign) compression. From a complementary dynamical-systems perspective, [5] develops a Lyapunov-function analysis for a continuous-time variant of Lion and introduces the Lion-$\mathcal{K}$ family, in which the non-differentiable $\operatorname{sign}(\cdot)$ operator is replaced by a subgradient mapping of a convex function; the paper interprets Lion as a principled method for a bound-constrained objective :

$$\min_w L(w) \quad \text{s.t.} \quad \|w\|_\infty \leqslant \frac{1}{\lambda}, \tag{2}$$

where λ is the weight decay hyperparameter. While this constrained interpretation provides an important structural explanation of Lion's dynamics, similar implicit or explicit constraint viewpoints can also arise for more classical optimizers under decoupled weight decay and related regularization mechanisms; by itself, it does not yet fully explain the consistently superior empirical performance reported for Lion across diverse deep-learning workloads.

On the algorithmic side, several variants replace the discrete sign step with smooth bounded nonlinearities to improve stability while preserving Lion's memory/computation profile. For example, RLion [19] proposes to substitute $\operatorname{sign}(\cdot)$ with an arctan-based bounded mapping and reports competitive performance on vision benchmarks. Finally, practical studies emphasize Lion's sensitivity to configuration and the need for careful, principled hyperparameter tuning when comparing to strong baselines such as AdamW, suggesting that part of Lion's apparent advantage can be contingent on matching training protocols and tuning budgets.

To summarize, Lion can be viewed as a member of the broader family of sign- and norm-based update rules, and recent work provides multiple complementary interpretations (constrained/composite optimization, Lyapunov analyses, and algorithmic smooth variants). In the remainder of this paper, we focus on a specific mechanism that is not fully isolated by these viewpoints: the direction perturbation induced by $\operatorname{sign}(\cdot)$ and its interaction with momentum and common architectural components (normalization layers and rectifying activations). This perspective motivates our angle-based characterization of the induced noise and the synthetic construction that injects comparable perturbations into Norm SGD, enabling controlled experiments and analysis.

§3. Rotation noise

Most loss function landscape (LFL) theories work with rotation-invariant constructions, such as the spectral norm of the Hessian. From the rotation-invariant perspective, the choice of basis for the sign operation is effectively random. The sign operation itself can be interpreted as a form of random rotational noise injection into the gradient direction. This interpretation seems reasonable since noise in loss function gradients plays a crucial role in neural network optimization [17]. We begin by analyzing this noise distribution theoretically.

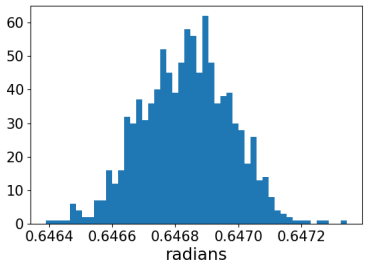

Figure 1. A histogram of the rotational noise sampled uniformly from the sphere $\mathbb{S}^{6000000}$.

Definition 2. *We define* ***Lion Noise Injection*** *(LNI) on a vector v as follows: sample an orthonormal basis B uniformly, express v in the coordinates of B, and then apply the sign function to these coordinates.*

Alternatively, we can describe LNI as follows: $\mathrm{LNI}(v) = w_B\sqrt{\dim(v)}$, where w_B is a normalized vector along the bisector of the hyperoctant to which v belongs, with respect to a sampled basis B. To analyze the noise distribution, we can reformulate the procedure: instead of sampling a basis, we can fix a basis and sample a vector v uniformly from the sphere — this gives an equivalent rotational noise distribution. This distribution is spherically symmetric, and in high dimensions the angle between (v) and $\mathrm{LNI}(v)$ concentrates around a constant; see Figure 1. Indeed, the limiting angle can be computed explicitly. Fix d and sample $x \sim \mathrm{Unif}(\mathbb{S}^{d-1})$. Write $x = g/\|g\|_2$ with $g_i \overset{iid}{\sim} \mathcal{N}(0,1)$. Then

$$\cos\theta = \langle x, \mathrm{sign}(x)/\sqrt{d}\rangle = \left\langle \frac{g}{\|g\|_2}, \frac{\mathrm{sign}(g)}{\sqrt{d}} \right\rangle = \frac{\sum_{i=1}^{d}|g_i|}{\sqrt{d}\,\|g\|_2} = \frac{\|g\|_1}{\sqrt{d}\,\|g\|_2}.$$

By the law of large numbers, $\frac{1}{d}\|g\|_1 \to \mathbb{E}|g_1| = \sqrt{2/\pi}$ and $\frac{1}{\sqrt{d}}\|g\|_2 \to \sqrt{\mathbb{E}g_1^2} = 1$, hence $\cos\theta \to \sqrt{2/\pi}$ and therefore

$$\theta \to \arccos\left(\sqrt{\tfrac{2}{\pi}}\right) \approx 0.6466 \text{ rad},$$

which matches the peak near 0.647 radians in Figure 1.

A significant distinction between this model and practice is that in practice, the distribution of gradient components tends to have heavy tails [21].

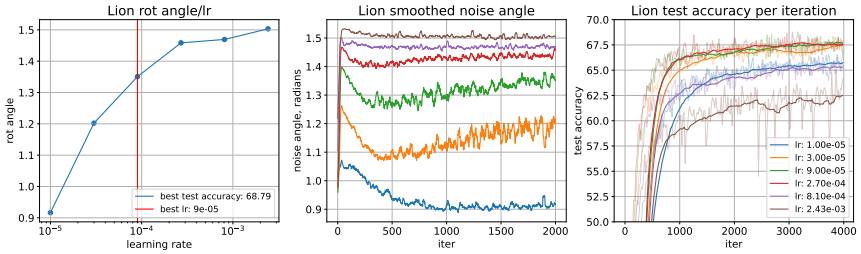

Figure 2. Multiple runs of Lion with different learning rates on ResNet9 trained for 200 epochs on CIFAR100. (Left:) Lion runs with larger learning rates correspond to larger angles. (Center:) The angle by which Lion's sign operation rotates the update vector. The observed angles exceed the theoretical estimate (which is based on the assumption that the gradients are uniformly sampled from the sphere).

This implies that gradient directions are concentrated near the hyperoctant's border rather than being uniformly distributed over the sphere.

As shown in Figure 2, Lion does exhibit the degenerate angle property, although the observed angle consistently exceeds the theoretical estimate. A heavy-tailed gradient implies that a small number of coordinates attain unusually large magnitudes, so L2 norm becomes dominated by these few large entries (quadratic amplification), while the L1 norm increases only linearly. Consequently, $\|\nabla L(w_t)\|_1/\|\nabla L(w_t)\|_2$ decreases as tail-heaviness (or coordinate sparsity/inequality) increases, which in turn reduces $\cos(\theta_t) = \|\nabla L(w_t)\|_1/(\sqrt{d}\,\|\nabla L(w_t)\|_2)$ and yields a larger typical angle. Moreover, the angle depends heavily on the learning rate. This suggests that the learning rate may influence the angle through its effect on tail behavior. Prior work has shown that SGD iterates in deep networks exhibit heavy-tailed behavior [21], and that in analytical models the heaviness of these tails increases with the learning rate [9], suggesting a principled link between step size and tail behavior.

Consider the following optimization scheme that mirrors Lion (see Definition 1).

Definition 3. *Define* ***Enim*** (*Evolved Noise Injection Momentum*) *as an algorithm that modifies Lion by replacing the sign operation with LNI. The*

update rule of Enim:

$$\begin{cases} m_{t+1} = \beta_2 m_t + (1-\beta_2)\nabla L(w_t) \\ w_{t+1} = (1-\eta\lambda)w_t + \eta\,\mathrm{LNI}(\beta_1 m_{t+1} + (1-\beta_1)\nabla L(w_t)) \end{cases} \tag{3}$$

where η is the learning rate, λ is the weight decay coefficient, m is momentum, w is the network weights.

To test whether this view of the sign operation is reasonable, we compare Lion directly to Enim. We calibrate Enim's noise injections to match Lion's empirically observed sign noise. More precisely, for each learning rate we first select an angle matching Lion's sign operation angle (observed at this learning rate). Then, at each iteration, we rotate the update vector by this angle in a random direction and normalize it to match Lion's step size. The right panel of Figure 3 shows that Enim with matched noise injections performs slightly better than Lion (see the right panel of Figure 2 for reference). Enim also appears less sensitive to the choice of the learning rate (Figure 3). One possible explanation lies in the temporal correlation of the directional error. In Lion, the update direction is a deterministic function of the momentum via the sign operator, so any angular bias relative to the true gradient is systematic and persists across steps; changing the learning rate therefore scales this persistent drift, making performance sensitive to its value. In Enim, the rotation is resampled independently at each step, so the directional error has near-zero mean and weak temporal correlation; varying the learning rate then primarily scales variance rather than coherent bias, yielding a broader range of stable learning rates.

The natural question then becomes how different noise angles interact with Enim's performance and to what extent we can enhance its performance by varying the noise. See the left panel of Figure 3 (an analogous graph for noise is presented in the Appendix). As it turns out, the angles observed empirically for Lion yield near-optimal performance for Enim, at least in our small-scale CIFAR100 setting. One important observation is that the optimal noise angle for Enim increases when we increase the learning rate. This is counterintuitive from the perspective of LFL theories, as both angle growth and learning rate growth make positive contributions to the noisiness of the optimization process. One explanation for this phenomenon lies in several effects described below. The first, less dramatic effect, is that for small learning rates and large angles, Enim may not reach its best performance within the 200-epoch training budget. The second effect, which we term "momentum tracing", is discussed in the next section. For

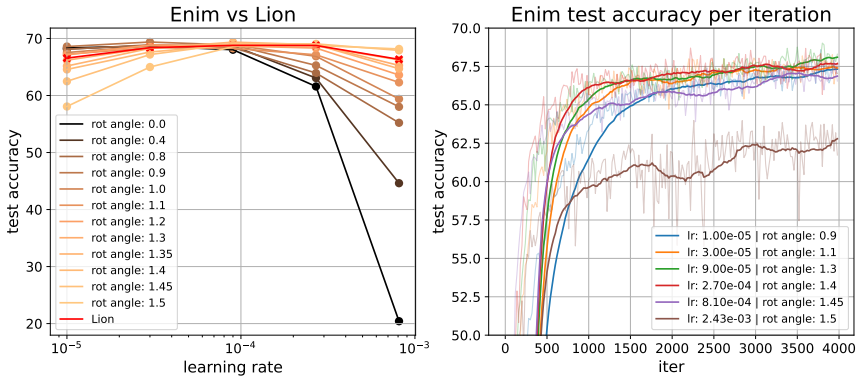

Figure 3. Enim's performance on ResNet9 trained for 200 epochs on CIFAR100, with noise angles matching those empirically observed for Lion's sign operation. The right panel (together with the right panel of Fig. 2) shows that Enim performs on par with or better than Lion, even without additional noise angle tuning.

Enim, we observe that large noise angles soften this effect, preventing the optimization process from destabilizing.

On the other hand, several intriguing properties regarding the implicit bias of the Enim and Lion optimizers can be formally derived. Let us consider the Taylor expansion of the loss function following a single update step of the simplified Lion optimizer, $w' = w - \eta \operatorname{sign}(g)$:

$$L(w') \approx L(w) - \eta g^\top \operatorname{sign}(g) + \frac{\eta^2}{2} \operatorname{sign}(g)^\top H \operatorname{sign}(g), \tag{4}$$

where $g = \nabla L(w)$ denotes the gradient and $H = \nabla^2 L(w)$ represents the Hessian matrix. Because of the sharpness term (the second-order component governed by the Hessian), gradient-based methods cannot guarantee a strictly monotonic decrease in the loss over a single step for an arbitrary objective function. In practice, however, these algorithms typically converge successfully – a phenomenon largely attributed to their implicit regularization properties. Standard gradient descent, for example, induces an implicit regularization penalty proportional to $\|\nabla L\|_2^2$, a fact formally established via backward error analysis [3, 22]. Over numerous optimization steps, this implicit bias steers the optimizer toward minima where the

second-order sharpness remains small, effectively preventing the loss from diverging. To elucidate how this stabilizing effect manifests within the Lion framework, we can reformulate the sign operator by decomposing the update step into what we term Lion Noise Injection (LNI) components. This yields a vector parallel to the true gradient g, coupled with an orthogonal rotation noise vector ξ:

$$\operatorname{sign}(g) = cg + \xi, \quad \text{where} \quad g^\top \xi = 0. \tag{5}$$

The scalar coefficient c emerges naturally when we take the inner product with g:

$$g^\top \operatorname{sign}(g) = c\|g\|_2^2 + g^\top \xi \implies \|g\|_1 = c\|g\|_2^2 \implies c = \frac{\|g\|_1}{\|g\|_2^2}. \tag{6}$$

Furthermore, since the squared ℓ_2-norm of any d-dimensional sign vector evaluates exactly to d, invoking the Pythagorean theorem yields:

$$d = \|\operatorname{sign}(g)\|_2^2 = \|cg\|_2^2 + \|\xi\|_2^2. \tag{7}$$

Recognizing that the angle θ between the raw gradient and its signum is formally defined by $\cos(\theta) = \frac{\|g\|_1}{\sqrt{d}\|g\|_2}$, we can rewrite the squared norm of the parallel component as follows:

$$\|cg\|_2^2 = c^2\|g\|_2^2 = \frac{\|g\|_1^2}{\|g\|_2^2} = d\left(\frac{\|g\|_1}{\sqrt{d}\|g\|_2}\right)^2 = d\cos^2(\theta). \tag{8}$$

Given that $d = d\cos^2(\theta) + \|\xi\|_2^2$, the squared magnitude of the orthogonal noise vector ξ inherently simplifies to:

$$\|\xi\|_2^2 = d - d\cos^2(\theta) = d\sin^2(\theta). \tag{9}$$

Now, let us examine the second-order sharpness penalty, $\mathcal{P}(w)$, originating from the Taylor expansion:

$$\mathcal{P}(w) = \frac{\eta^2}{2}(cg+\xi)^\top H(cg+\xi) = \frac{\eta^2}{2}\left(c^2 g^\top Hg + 2c\xi^\top Hg + \xi^\top H\xi\right). \tag{10}$$

This penalty fundamentally depends on the specific realization of the rotation noise vector ξ. Within the Enim optimizer, ξ acts as isotropic random rotation noise sampled from the $(d-1)$-dimensional orthogonal subspace. Taking the expectation over this noise distribution, the linear cross-term naturally vanishes due to the zero-mean property of ξ (i.e., $\mathbb{E}[\xi] = 0$):

$$\mathbb{E}[2c\xi^\top Hg] = 0. \tag{11}$$

To evaluate the quadratic noise component $\mathbb{E}[\xi^\top H\xi]$, we apply the lemma proven in Appendix C:

$$\mathbb{E}\left[\xi^\top H\xi\right] = \frac{\|\xi\|_2^2}{d-1}\mathrm{Tr}\left(P^\perp HP^\perp\right) = \frac{\|\xi\|_2^2}{d-1}\left(\mathrm{Tr}(H) - \frac{g^\top Hg}{\|g\|_2^2}\right). \tag{12}$$

To clarify the geometric intuition underpinning this regularization, we define $h_g = \frac{g^\top Hg}{\|g\|_2^2}$ as the *directional sharpness* (capturing the curvature strictly along the gradient's trajectory), and $\bar{h}_\perp = \frac{\mathrm{Tr}(H)-h_g}{d-1}$ as the *average orthogonal sharpness* (representing the mean curvature across all $d-1$ dimensions perpendicular to the gradient). By substituting these definitions alongside our previously derived norm identities, $\|cg\|_2^2 = d\cos^2(\theta)$ and $\|\xi\|_2^2 = d\sin^2(\theta)$, the expected sharpness penalty for Enim elegantly reduces to:

$$\mathcal{P}_{\mathrm{Enim}}(w) = \frac{\eta^2 d}{2}\Big[\cos^2(\theta)h_g + \sin^2(\theta)\bar{h}_\perp\Big]. \tag{13}$$

This formulation explicitly demonstrates that the rotation noise angle θ operates as a continuous mixing weight. Whereas standard normalized gradient descent ($\theta = 0$) strictly penalizes local directional sharpness (h_g), the injection of rotation noise ($\theta > 0$) shifts a portion of the implicit penalty toward $\bar{h}_\perp$. By penalizing $\bar{h}_\perp$, the algorithm effectively restrains the trace of the Hessian, actively steering the optimization trajectory toward flatter minima in all directions, which are widely associated with superior generalization capabilities.

Since in the random-basis (LNI) view the sign-induced residual $\xi_t \perp g_t$ is approximately isotropic in the subspace orthogonal to g_t (i.e., $\mathbb{E}[\xi_t\xi_t^\top \mid g_t] \propto I - \frac{g_t g_t^\top}{\|g_t\|_2^2}$), Lion can be modeled as Enim with a time-varying θ_t, and the same expected curvature-mixing implicit regularizer follows. Empirically, Enim is on par with Lion when θ is calibrated to match Lion's observed angle, supporting the view that Lion's gains are largely explained by this directional-noise mechanism. This automatic adaptation of θ is induced by the geometric constraints of the update itself: the $\mathrm{sign}(\cdot)$ operator restricts the attainable update directions to the vertices of a discrete hypercube. This biases the trajectory toward a specific class of minima. As training enters these regimes, the effective angle between the true gradient and its signum changes accordingly, manifesting as an adaptive level of directional noise.

Moreover, the empirical relationship between the learning rate and the rotation angle (Figure 2) strongly aligns with this theoretical result. A

larger learning rate drastically amplifies the sharpness penalty via its quadratic scaling. Simultaneously, large learning rates in neural networks are known to drive trajectories into regimes characterized by heavy-tailed gradients [9]. As established, our heavy-tail proxy metric — the ℓ_1/ℓ_2 gradient norm ratio — is directly related to the rotation angle. Heavy-tailed gradients typically arise in landscape regimes where a small number of top Hessian eigenvalues dominate the spectrum. In such regions, the sharpness strictly along the gradient direction is vastly larger than the average orthogonal sharpness ($h_g \gg \bar{h}_\perp$). If θ were small, the massive h_g term would cause the quadratically scaled penalty to explode, leading to divergence. To maintain stability, the optimization dynamics are implicitly constrained to sparse gradient regimes that geometrically yield a larger rotation angle θ. This shift inherently shrinks $\cos^2(\theta)$ and grows $\sin^2(\theta)$, effectively dampening the impact of the dominant directional sharpness h_g and safely dispersing the regularization burden over the surrounding loss landscape corresponding to $\bar{h}_\perp$. Overall, the theoretical role of rotation noise bridges the geometry of the sign operator with the known properties of heavy-tailed gradient dynamics.

§4. Momentum tracing

Most modern neural networks incorporate normalization layers, such as Batch Normalization [10] and Layer Normalization [1], followed by ReLU or similar activations. When the preactivation feature map components are negative, ReLU outputs zero, resulting in no backpropagation signal through those components. Consequently, the corresponding components of the bias term in the normalization layer receive zero gradient.

After the gradient becomes zero, these components continue to update in the direction of momentum for an extended period until the gradient becomes non-zero again or the momentum value diminishes to numerical precision limits. If the momentum components associated with the bias term of the normalization layers are negative, a positive feedback loop can occur. In this scenario, the bias components shift further into negative values, reducing the preactivation values even more, which in turn maintains zero gradients.

By the moment when the gradient signal suddenly reappears (e.g., due to skip-connections), the weight values contributing to the corresponding activation may have shifted into suboptimal regions, causing abrupt

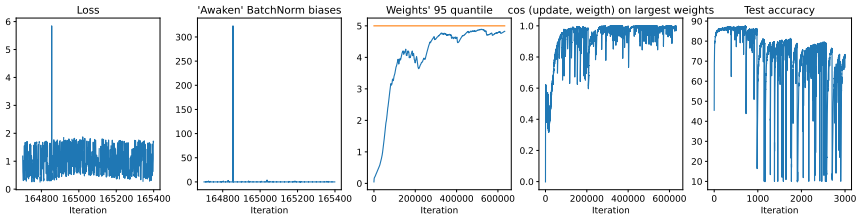

Figure 4. A display of Lion divergence with ResNet18 on CIFAR-10 . (Center left:) A number of BN biases with a near-zero momentum that receive a non-zero gradient on the current iteration. (Center:) Blue — a 95 quantile of networks' parameters values. Under Lion-weight decay dynamics weights converge to $\pm 5 = \pm 1/\lambda$ when their gradients are identical to zero. (Center right:) Cosine similarity between 5% largest weights and their updates converges to 1 as the weights converge to $1/\lambda$.

changes that destabilize the training process. We term this effect "momentum tracing". With both convolutional and linear layers multiple parameters lose the gradient signal when the corresponding BatchNorm activation outputs zero.

This effect can occur with any optimization method. However, in methods like Adam, the magnitude of the update for a component decreases rapidly at a geometric progression rate of approximately $\beta_1/\sqrt{\beta_2}$, where commonly used values are $\beta_1 = 0.9$ and $\beta_2 = 0.999$. This rapid decay prevents the optimizer from pushing the weights too far into suboptimal regions. In contrast, methods like Norm Gradient Descent (Norm GD) with momentum and decoupled momentum mechanisms similar to Lion exhibit a slower decay of the momentum component, typically with $\beta_2 = 0.99$. In Lion, the momentum decay rate is similar to that of Norm GD, but due to the use of the signum function, the method can take larger update steps for a component, which would be expected to lead to even greater destabilization. However, the noise introduced by the method reduces the frequency of zero gradients.

A distinctive aspect of Lion here is that decoupled weight decay combined with a sign update induces a natural *saturation scale* for coordinates with vanishing gradients. For a coordinate with $g_t = \nabla L(w_t) = 0$, any

steady state $w_{t+1} = w_t$ must satisfy

$$0 = w_{t+1} - w_t = -\eta\lambda w_t + \eta \operatorname{sign}(m_{t+1}) \quad \Longrightarrow \quad w_t \in \left\{ \pm \frac{1}{\lambda} \right\}.$$

Hence, if a parameter spends several iterations with (approximately) zero gradient while $\operatorname{sign}(m_t)$ remains persistent, Lion drives it toward $\pm 1/\lambda$. This is reflected in Figure 4 (center), where large-magnitude parameters cluster near $\pm 5 = \pm 1/\lambda$, and in Figure 4 (right), where the cosine similarity between the largest weights and their updates approaches 1 as $|w_t| \to 1/\lambda$.

In experiments with Enim, momentum tracing is most noticeable when the noise is set to zero and with large learning rates. The larger the learning rate, the more significant the steps the optimizer takes in the direction of momentum when the gradient has diminished. Introducing noise by rotating the update direction by a random fixed angle can result in gradients becoming non-zero earlier, preventing the optimizer from taking excessive steps in the momentum direction.

§5. Conclusion

In this work, we analyzed the characteristics of the specific sign noise introduced by the Lion optimizer. We demonstrated that Lion adapts the rotation noise in accordance with the learning rate to stabilize the training process, which can significantly impact model performance. Additionally, we introduced a novel optimization method that employs random rotations of the gradient update at fixed angles, serving as a tool for analyzing Lion's behavior. Finally, our theoretical analysis shows that the rotation angle provides a controllable degree of freedom that modulates the optimizer's implicit bias by tuning the balance between directional sharpness and average sharpness across directions.

Future work. We plan to delve deeper into the mechanisms of Lion superior performance related to noise injection and per-element fixed step size. In addition, we will conduct a more systematic study of the loss-landscape geometry around solutions favored by Lion and assess how these empirical signatures align with our theoretical characterization of implicit bias.

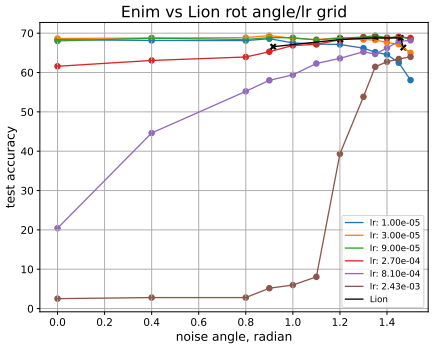

Figure 5. For larger learning rates, reducing the noise level leads to a sharp deterioration in performance. Enim also appears less sensitive to the learning rate than Lion.

Appendix A. Rotation angle dependence

Specified effects were better visible for methods with momentum and affine transform in normalization layers. Figure 5 explores the joint dependence of performance on learning rate and rotation angle. We observe that for larger learning rates, reducing the noise level leads to a rapid degradation in performance. Moreover, the empirically observed Lion angles lie close to the ridge of near-optimal performance.

In contrast to Lion, Enim exhibits a broader plateau of stable learning rates, which is consistent with its weaker temporal correlation in directional error.

Appendix B. Momentum tracing

The momentum tracing effect is amplified in architectures with normalization layers and ReLU activations. Figure 6 shows that for large learning rates and small rotation angles, the number of zero gradient components increases substantially. In these coordinates, the update direction can remain aligned with the accumulated momentum over many steps.

Under decoupled weight decay and sign-based updates (Lion), this can drive parameters toward the saturation scale $\pm 1/\lambda$ and reinforce zero-gradient regimes due to activation-induced gradient suppression. Introducing rotation noise reduces the persistence of such coordinates, weakening temporal bias accumulation and improving stability.

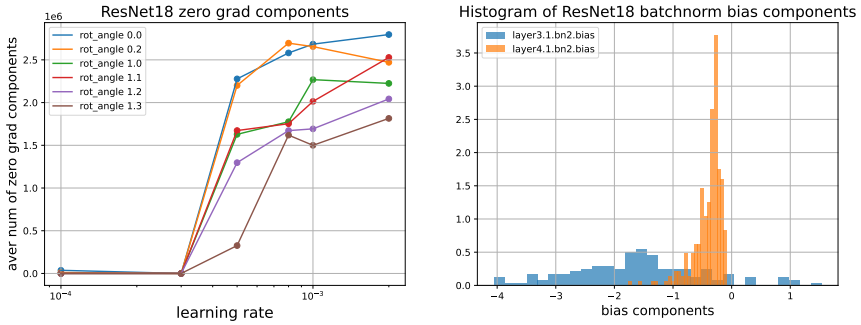

Figure 6. (Left:) Relationship between learning rate, noise and average number of zero components in gradient. The more zero components gradient have, the more the method can be destabilized due to momentum tracing. (Right:) Distribution of bias term in batch normalization layer. Most components are negative, which could lead to positive feedback loop.

Appendix C. Lemma

Lemma 1 (Expected Curvature of Orthogonal Noise)**.** *Let $g \in \mathbb{R}^d \setminus \{0\}$ be a fixed gradient vector and $H \in \mathbb{R}^{d\times d}$ be a symmetric Hessian matrix. Let $\xi \in \mathbb{R}^d$ be a random noise vector with a fixed squared norm $\|\xi\|_2^2$, isotropically distributed in the $(d-1)$-dimensional subspace orthogonal to g (i.e., $g^\top \xi = 0$). Then:*

$$\mathbb{E}\big[\xi^\top H \xi\big] = \frac{\|\xi\|_2^2}{d-1}\mathrm{Tr}\big(P^\perp H P^\perp\big) = \frac{\|\xi\|_2^2}{d-1}\left(\mathrm{Tr}(H) - \frac{g^\top H g}{\|g\|_2^2}\right)$$

where $P^\perp = I - \frac{gg^\top}{\|g\|_2^2}$ is the orthogonal projection matrix onto the subspace perpendicular to g.

Proof. Because ξ is zero-mean and isotropic within the $(d-1)$-dimensional subspace defined by the projection matrix $P^\perp$, its covariance matrix is strictly proportional to $P^\perp$:

$$\mathbb{E}[\xi\xi^\top] = cP^\perp$$

To find the scalar c, we take the trace of both sides. For the left side, using the cyclic property ($\mathrm{Tr}(AB) = \mathrm{Tr}(BA)$):

$$\mathrm{Tr}(\mathbb{E}[\xi\xi^\top]) = \mathbb{E}[\mathrm{Tr}(\xi\xi^\top)] = \mathbb{E}[\mathrm{Tr}(\xi^\top\xi)] = \|\xi\|_2^2$$

For the right side, the trace of a projection matrix equals the dimension of its image subspace:

$$\mathrm{Tr}(cP^\perp) = c \cdot \mathrm{Tr}\left(I - \frac{gg^\top}{\|g\|_2^2}\right) = c\left(d - \frac{\mathrm{Tr}(g^\top g)}{\|g\|_2^2}\right) = c(d-1)$$

Equating these gives $c = \frac{\|\xi\|_2^2}{d-1}$. Thus, the exact covariance matrix is $\mathbb{E}[\xi\xi^\top] = \frac{\|\xi\|_2^2}{d-1}P^\perp$.

Since $\xi^\top H\xi$ is a 1×1 scalar, it equals its own trace. Using the linearity of expectation and the cyclic property of the trace:

$$\mathbb{E}\big[\xi^\top H\xi\big] = \mathbb{E}[\mathrm{Tr}(\xi^\top H\xi)] = \mathbb{E}[\mathrm{Tr}(H\xi\xi^\top)] = \mathrm{Tr}\big(H\mathbb{E}[\xi\xi^\top]\big)$$

Substituting the covariance matrix from Step 1:

$$\mathbb{E}\big[\xi^\top H\xi\big] = \mathrm{Tr}\left(H\left[\frac{\|\xi\|_2^2}{d-1}P^\perp\right]\right) = \frac{\|\xi\|_2^2}{d-1}\mathrm{Tr}(HP^\perp)$$

Because $P^\perp$ is symmetric and idempotent ($(P^\perp)^2 = P^\perp$), applying the cyclic property yields $\mathrm{Tr}(HP^\perp) = \mathrm{Tr}(HP^\perp P^\perp) = \mathrm{Tr}(P^\perp HP^\perp)$, which proves the first equality.

Substitute the explicit definition of $P^\perp$ into $\mathrm{Tr}(HP^\perp)$:

$$\mathrm{Tr}(HP^\perp) = \mathrm{Tr}\left(H\left(I - \frac{gg^\top}{\|g\|_2^2}\right)\right) = \mathrm{Tr}(H) - \mathrm{Tr}\left(\frac{Hgg^\top}{\|g\|_2^2}\right)$$

Applying the cyclic property to the second term ($\mathrm{Tr}(Hgg^\top) = \mathrm{Tr}(g^\top Hg)$) yields:

$$\mathrm{Tr}\left(\frac{Hgg^\top}{\|g\|_2^2}\right) = \frac{g^\top Hg}{\|g\|_2^2} \implies \mathrm{Tr}(HP^\perp) = \mathrm{Tr}(H) - \frac{g^\top Hg}{\|g\|_2^2}$$

Multiplying this expanded trace by the covariance scalar $\frac{\|\xi\|_2^2}{d-1}$ completes the proof. □

References

1. J. L. Ba, J. R. Kiros, and G. E. Hinton, *Layer Normalization*, ArXiv preprint arXiv:1607.06450 (2016).

2. L. Balles, F. Pedregosa, and N. Le Roux, *The Geometry of Sign Gradient Descent*, ArXiv preprint arXiv:2002.08056 (2020).
3. D. G. T. Barrett and B. Dherin, *Implicit Gradient Regularization*, ArXiv preprint arXiv:2009.11162 (2020).
4. M. D. Cattaneo, J. M. Klusowski, and B. Shigida, *On the Implicit Bias of Adam*, ArXiv preprint arXiv:2309.00079 (2023).
5. L. Chen, B. Liu, K. Liang, and Q. Liu, *Lion Secretly Solves Constrained Optimization: As Lyapunov Predicts*, ArXiv preprint arXiv:2310.05898 (2023).
6. X. Chen, C. Liang, D. Huang, E. Real, K. Wang, H. Pham, X. Dong, T. Luong, C.-J. Hsieh, Y. Lu, *et al.*, *Symbolic Discovery of Optimization Algorithms*, in: Advances in Neural Information Processing Systems **36**, 2024.
7. J. M. Cohen, B. Ghorbani, S. Krishnan, N. Agarwal, S. Medapati, M. Badura, D. Suo, D. Cardoze, Z. Nado, G. E. Dahl, *et al.*, *Adaptive Gradient Methods at the Edge of Stability*, ArXiv preprint arXiv:2207.14484 (2022).
8. Y. Dong, H. Li, and Z. Lin, *Convergence Rate Analysis of Lion*, ArXiv preprint arXiv:2411.07724 (2024).
9. M. Gurbuzbalaban, U. Simsekli, and L. Zhu, *The Heavy-Tail Phenomenon in SGD*, in: International Conference on Machine Learning, PMLR, 2021, pp. 3964–3975.
10. S. Ioffe and C. Szegedy, *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*, ArXiv preprint arXiv:1502.03167 (2015).
11. W. Jiang and L. Zhang, *Convergence Analysis of the Lion Optimizer in Centralized and Distributed Settings*, ArXiv preprint arXiv:2508.12327 (2025).
12. D. P. Kingma, *Adam: A Method for Stochastic Optimization*, ArXiv preprint arXiv:1412.6980 (2014).
13. X. Li, K.-Y. Lin, L. Li, Y. Hong, and J. Chen, *On Faster Convergence of Scaled Sign Gradient Descent*, IEEE Transactions on Industrial Informatics **20** (2023), no. 2, 1732–1741.
14. D. P. Mandic, *A Generalized Normalized Gradient Descent Algorithm*, IEEE Signal Processing Letters **11** (2004), no. 2, 115–118.
15. E. Moulay, V. Léchappé, and F. Plestan, *Properties of the Sign Gradient Descent Algorithms*, Information Sciences **492** (2019), 29–39.

16. R. Murray, B. Swenson, and S. Kar, *Revisiting Normalized Gradient Descent: Fast Evasion of Saddle Points*, IEEE Transactions on Automatic Control **64** (2019), no. 11, 4818–4824.
17. A. Neelakantan, L. Vilnis, Q. V. Le, I. Sutskever, L. Kaiser, K. Kurach, and J. Martens, *Adding Gradient Noise Improves Learning for Very Deep Networks*, ArXiv preprint arXiv:1511.06807 (2015).
18. S. J. Reddi, S. Kale, and S. Kumar, *On the Convergence of Adam and Beyond*, ArXiv preprint arXiv:1904.09237 (2019).
19. J. Rong, C. Ma, Q. Zhang, and Y. Cao, *RLion: A Refined Lion Optimizer for Deep Learning* (2024).
20. M. Safaryan and P. Richtárik, *Stochastic Sign Descent Methods: New Algorithms and Better Theory*, in: International Conference on Machine Learning, PMLR, 2021, pp. 9224–9234.
21. U. Simsekli, L. Sagun, and M. Gurbuzbalaban, *A Tail-Index Analysis of Stochastic Gradient Noise in Deep Neural Networks*, in: International Conference on Machine Learning, PMLR, 2019, pp. 5827–5837.
22. S. L. Smith, B. Dherin, D. G. T. Barrett, and S. De, *On the Origin of Implicit Regularization in Stochastic Gradient Descent*, ArXiv preprint arXiv:2101.12176 (2021).
23. S.-Y. Zhao, Y.-P. Xie, and W.-J. Li, *On the Convergence and Improvement of Stochastic Normalized Gradient Descent*, Science China Information Sciences **64** (2021), 1–13.

Lomonosov Moscow State University,
Moscow, Russia
E-mail: semenelist@gmail.com

St.Petersburg State University,
St. Petersburg, Russia
E-mail: andrey.podivilov@gmail.com

Constructor University,
Bremen, Germany;
HSE University, Moscow, Russia
E-mail: tiuzhakov@constructor.university

Constructor University,
Bremen, Germany
E-mail: dvetrov@constructor.university

Поступило 4 февраля 2026 г.