# Muon 在大语言模型训练中的可扩展性研究

技术报告

**Jingyuan Liu**[1] **Jianlin Su**[1] **Xingcheng Yao**[2] **Zhejun Jiang**[1] **Guokun Lai**[1] **Yulun Du**[1] **Yidao Qin**[1] **Weixin Xu**[1] **Enzhe Lu**[1] **Junjie Yan**[1] **Yanru Chen**[1] **Huabin Zheng**[1] **Yibo Liu**[1] **Shaowei Liu**[1] **Bohong Yin**[1] **Weiran He**[1] **Han Zhu**[1] **Yuzhi Wang**[1] **Jianzhou Wang**[1] **Mengnan Dong**[1] **Zheng Zhang**[1] **Yongsheng Kang**[1] **Hao Zhang**[1] **Xinran Xu**[1] **Yutao Zhang**[1] **Yuxin Wu**[1] **Xinyu Zhou**[1] * **Zhilin Yang**[1]

[1] Moonshot AI [2] UCLA

## Abstract

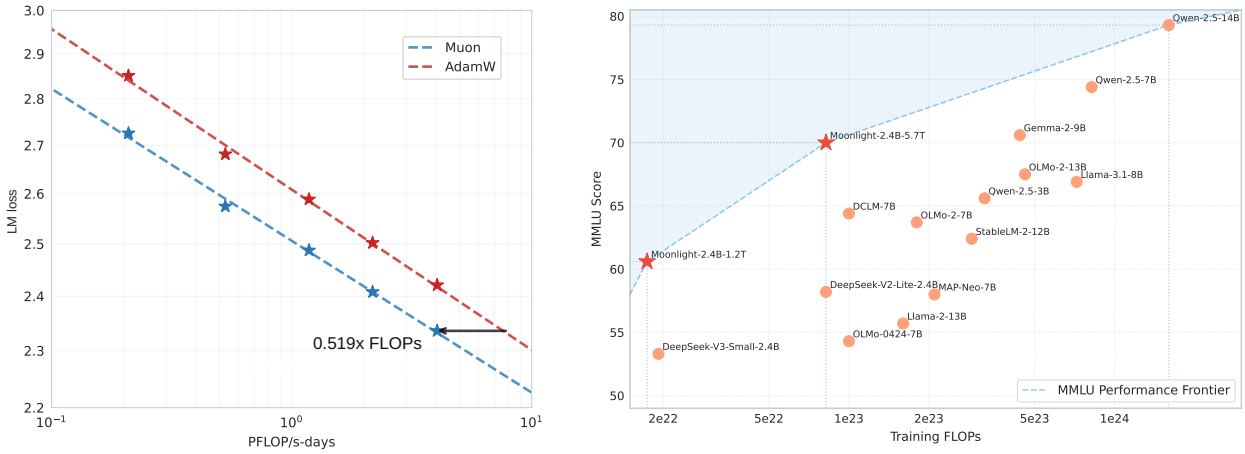
近期，基于矩阵正交化的 Muon 优化器 (K. Jordan et al. 2024) 在训练小规模语言模型方面展现出优异效果，但其在更大规模模型上的可扩展性尚未得到验证。我们识别出两项对 Muon 规模化扩展至关重要的技术：(1) 引入权重衰减（weight decay）和 (2) 精细调整逐参数更新尺度。这些技术使 Muon 能够在大规模训练中即插即用，无需进行超参数调优。扩展定律实验表明，在计算最优训练条件下，Muon 相比 AdamW 实现了约 $\sim 2\times$ 的计算效率提升。基于这些改进，我们推出了 Moonlight，一个使用 Muon 训练、具有 3B/16B 参数的混合专家（MoE）模型，训练数据量为 5.7T 个 token。我们的模型提升了当前的帕累托前沿，相比先前模型以更少的训练 FLOPs 实现了更优的性能。我们开源了分布式 Muon 实现，该实现具有内存最优和通信高效的特点。我们还发布了预训练模型、指令微调模型以及中间训练检查点，以支持未来研究。

## 1 引言

大语言模型（LLMs） (OpenAI et al. 2024; DeepSeek-AI et al. 2024; Grattafiori et al. 2024; Gemini Team et al. 2024) 的快速发展极大地推动了通用人工智能的进步。然而，由于扩展定律 (Kaplan et al. 2020; Hoffmann et al. 2022) 的存在，训练具有竞争力的大语言模型仍然是一个计算密集且资源需求巨大的过程。优化器在高效且有效地训练大语言模型方面发挥着关键作用，其中 Adam (Kingma et al. 2015) 及其变体 AdamW (Loshchilov et al. 2019) 是大多数大规模训练的标准选择。

近期优化算法的发展显示出超越 AdamW 提升训练效率的潜力 (Liu et al. 2024; K. Jordan et al. 2024; Yuan et al. 2024; Vyas et al. 2025; X.-L. Li 2018a; X.-L. Li 2018b; Pooladzandi et al. 2024; X. Li 2022; X.-L. Li 2024; Pethick et al. 2025)。其中，K. Jordan et al. 2024提出了 Muon，该优化器使用 Newton-Schulz 迭代对梯度动量进行正交化来更新矩阵参数。Muon 在小规模语言模型训练中的初步实

*通讯作者: zhouxinyu@moonshot.cn




(a) (b)

图 1: Muon 的规模化扩展。**(a)** Muon 与 Adam 的扩展定律实验对比。在计算最优训练条件下，Muon 相比 Adam 具有约 $\sim 2\times$ 的计算效率优势。**(b)** 我们使用 Muon 优化的 Moonlight 模型与其他可比模型的 MMLU 性能对比。Moonlight 在性能与训练 FLOPs 的帕累托前沿上取得了进展。

验展示了 promising 的结果。然而，如该博客 (K. Jordan et al. 2024) 所讨论的，几个关键挑战仍有待解决：(1) 如何有效地将基于矩阵正交化的优化器扩展到具有数十亿参数、使用数万亿 token 训练的更大模型，(2) 如何在分布式环境中计算近似正交化，以及 (3) 此类优化器是否能够泛化到包括预训练和 supervised finetuning（SFT）在内的不同训练阶段。

在本技术报告中，我们呈现了一项全面研究来解决这些挑战。我们的工作建立在 Muon 的基础上，同时系统地识别和解决了其在大规模训练场景中的局限性。我们的技术贡献包括：

- **Muon 有效扩展的分析**：通过大量分析，我们识别出权重衰减在 Muon 可扩展性中扮演的关键角色。此外，我们提出了对 Muon 逐参数更新规则的尺度调整。这些调整使 Muon 能够即插即用，无需超参数调优，同时显著提高了训练稳定性。

- **高效的分布式实现**：我们开发了具有 ZeRO-1 (Rajbhandari et al. 2020) 风格优化的分布式 Muon 版本，在保持算法数学特性的同时实现了最优内存效率和降低通信开销。

- **扩展定律验证**：我们进行了将 Muon 与强 AdamW 基线对比的扩展定律研究，展示了 Muon 的优越性能（见图 1a）。基于扩展定律结果，Muon 仅需约 52% 的训练 FLOPs 即可达到与 AdamW 训练模型相当的性能。

我们的全面实验表明，Muon 能够有效替代 AdamW 成为大规模大语言模型训练的事实标准优化器，在训练效率和模型性能方面都提供了显著改进。作为这项工作的一部分，我们发布了 Moonlight——一个使用 Muon 训练的 16B 参数 MoE 模型，以及我们的实现和中间训练检查点，以促进大语言模型可扩展优化技术的进一步研究。

## 2 方法

### 2.1 背景

**Muon 优化器** Muon (K. Jordan et al. 2024) 最近被提出用于优化可表示为矩阵的神经网络权重。在第 $t$ 次迭代时，给定当前权重 $\mathbf{W}_{t-1}$、动量 $\mu$、学习率 $\eta_t$ 和目标函数 $\mathcal{L}_t$，Muon 优化器的更新规则可表述如下：

$$\begin{aligned}\mathbf{M}_t &= \mu\mathbf{M}_{t-1} + \nabla\mathcal{L}_t(\mathbf{W}_{t-1}) \\ \mathbf{O}_t &= \text{Newton-Schulz}(\mathbf{M}_t)^1 \\ \mathbf{W}_t &= \mathbf{W}_{t-1} - \eta_t\mathbf{O}_t\end{aligned} \tag{1}$$

这里，$\mathbf{M}_t$ 是第 $t$ 次迭代的梯度动量，当 $t=0$ 时设为零矩阵。在公式 1中，采用 Newton-Schulz 迭代过程 (Bernstein et al. 2024) 来近似求解 $(\mathbf{M}_t\mathbf{M}_t^{\mathrm{T}})^{-1/2}\mathbf{M}_t$。设 $\mathbf{U}\,\mathbf{V}^{\mathrm{T}} = \mathbf{M}_t$ 为 $\mathbf{M}_t$ 的奇异值分解（SVD），我们有 $(\mathbf{M}_t\mathbf{M}_t^{\mathrm{T}})^{-1/2}\mathbf{M}_t = \mathbf{U}\mathbf{V}^{\mathbf{T}}$，它对 $\mathbf{M}_t$ 进行正交化。直观上，正交化可以确保更新矩阵是等距同构的，防止权重仅沿着少数主导方向学习 (K. Jordan et al. 2024)。

**用于矩阵正交化的 Newton-Schulz 迭代** 公式 1通过迭代过程计算。开始时，我们设 $\mathbf{X}_0 = \mathbf{M}_t/\|\mathbf{M}_t\|_{\mathrm{F}}$。然后，在每次迭代 $k$ 中，我们按如下方式从 $\mathbf{X}_{k-1}$ 更新 $\mathbf{X}_k$：

$$\mathbf{X}_k = a\mathbf{X}_{k-1} + b(\mathbf{X}_{k-1}\mathbf{X}_{k-1}^{\mathrm{T}})\mathbf{X}_{k-1} + c(\mathbf{X}_{k-1}\mathbf{X}_{k-1}^{\mathrm{T}})^2\mathbf{X}_{k-1} \tag{2}$$

其中 $\mathbf{X}_N$ 是经过 $N$ 次迭代步骤后该过程的结果。这里 $a$、$b$、$c$ 是系数。为了确保公式 2的正确收敛，我们需要调整系数使得多项式 $f(x) = ax + bx^3 + cx^5$ 在 1 附近有不动点。在K. Jordan et al. 2024的原始设计中，系数设置为 $a = 3.4445$、$b = -4.7750$、$c = 2.0315$，以使迭代过程对较小的初始奇异值收敛更快。在本工作中，我们遵循相同的系数设置。

**范数约束下的最速下降** Bernstein et al. 2024提出将深度学习中的优化过程视为范数约束下的最速下降。从这个角度来看，我们可以将 Muon 与 Adam (Kingma et al. 2015; Loshchilov et al. 2019) 之间的差异视为范数约束的差异。Adam 是在动态调整的 Max-of-Max 范数约束下的最速下降，而 Muon 提供的是一个位于某个大 $p$ 的 Schatten-$p$ 范数静态范围内的范数约束 (Franz 2024)。当公式 1被准确计算时，Muon 提供的范数约束将是谱范数。神经网络的权重用作输入空间或隐藏空间上的算子，这些通常是（局部）欧几里得的 (Cesista 2024)，因此权重的范数约束应该是诱导算子范数（或权重矩阵的谱范数）。在这个意义上，Muon 提供的范数约束比 Adam 提供的更合理。

### 2.2 扩展 Muon

**权重衰减** 虽然 Muon 在小规模上显著优于 AdamW（如K. Jordan et al. 2024所示），但我们发现当扩展到使用更多 token 训练更大模型时，性能提升会减弱。我们观察到权重和层输出的 RMS 持续增长到很大规模，超过了 bf16 的高精度范围，这可能会损害模型性能。为了解决这个问题，我们将标准的 AdamW（Loshchilov et al. 2019）权重衰减机制引入 Muon[2]。

[1] 在实践中，我们遵循 (K. Jordan et al. 2024) 使用 Nesterov 风格的动量，将 $\mu\mathbf{M}_t + \nabla\mathcal{L}_t(\mathbf{W}_{t-1})$ 而非 $\mathbf{M}_t$ 输入 Newton-Schulz 迭代。

[2] Muon 的原始实现省略了权重衰减。最近一项关于 Muon 的并行工作引入了权重衰减并展示了改进的性能。参见此提交和此讨论。

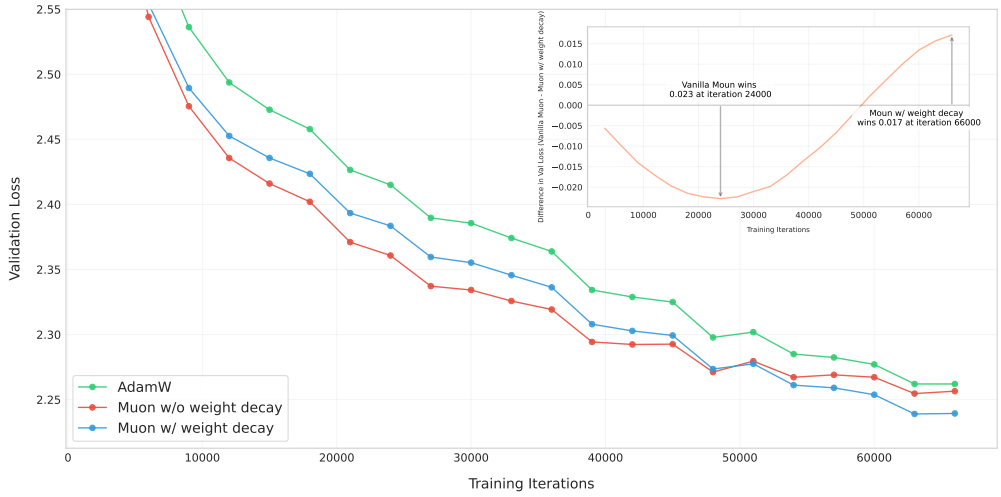



图 2: AdamW（绿色）、无权重衰减的 Muon（红色）和带权重衰减的 Muon（蓝色）的验证损失曲线。

$$\mathbf{W}_t = \mathbf{W}_{t-1} - \eta_t(\mathbf{O}_t + \lambda \mathbf{W}_{t-1}) \tag{3}$$

我们对 Muon 进行了有权重衰减和无权重衰减的实验，以了解其对大语言模型训练动态的影响。基于我们在第 3.2节中的扩展定律研究，我们使用 100B 个 token（$\sim 5\times$ 最优训练 token 量）训练了一个 800M 参数的模型。图 2展示了使用 AdamW、原始 Muon（无权重衰减）和带权重衰减的 Muon 训练的模型的验证损失曲线。虽然原始 Muon 最初收敛更快，但我们观察到一些模型权重随时间增长过大，可能限制了模型的长期性能。添加权重衰减解决了这个问题——结果表明，带权重衰减的 Muon 优于原始 Muon 和 AdamW，在过度训练状态下实现了更低的验证损失。因此，我们将更新规则调整为公式 3，其中 $\lambda$ 是权重衰减比率。

**一致的更新 RMS** Adam 和 AdamW（Kingma et al. 2015、Loshchilov et al. 2019）的一个重要特性是它们保持理论更新 RMS 约为 1[3]。然而，我们证明 Muon 的更新 RMS 根据参数的形状而变化，遵循以下引理：

*引理 1. 对于形状为 $[A, B]$ 的满秩矩阵参数，其理论 Muon 更新 RMS 为 $\sqrt{1/\max(A, B)}$。*

证明见附录 A。我们在训练期间监测了 Muon 的更新 RMS，发现它通常接近上述理论值。我们注意到这种不一致性在扩展模型大小时可能会产生问题：

- 当 $\max(A, B)$ 过大时，例如稠密 MLP 矩阵，更新变得过小，从而限制了模型的表示能力并导致次优性能；
- 当 $\max(A, B)$ 过小时，例如将 GQA（Shazeer 2019）或 MLA（DeepSeek-AI et al. 2024）中的每个 KV 头视为独立参数，更新变得过大，从而导致训练不稳定并同样导致次优性能。

[3]由于 Adam 的 $\beta_1 < \beta_2$ 和 $\epsilon > 0$，实际更新 RMS 通常小于 1。

为了保持不同形状矩阵之间一致的更新 RMS，我们建议将每个矩阵的 Muon 更新按其 $\sqrt{\max(A, B)}$ 进行缩放，以抵消引理 1的影响[4]。第 3.1节的实验表明这种策略对优化是有益的。

**匹配 AdamW 的更新 RMS** Muon 设计用于更新基于矩阵的参数。在实践中，AdamW 与 Muon 配合使用以处理非矩阵参数，如 RMSNorm、LM 头和嵌入参数。我们希望优化器超参数（学习率 $\eta$、权重衰减 $\lambda$）在矩阵和非矩阵参数之间共享。

我们建议将 Muon 的更新 RMS 匹配到与 AdamW 相似的范围。根据经验观察，AdamW 的更新 RMS 通常在 0.2 到 0.4 之间。因此，我们通过以下调整将 Muon 的更新 RMS 缩放到该范围：

$$\mathbf{W}_t = \mathbf{W}_{t-1} - \eta_t(0.2 \cdot \mathbf{O}_t \cdot \sqrt{\max(A, B)} + \lambda \mathbf{W}_{t-1}) \tag{4}$$

我们通过实证结果验证了这种选择（详见附录 A）。此外，我们强调通过这种调整，Muon 可以直接**复用**为 AdamW 调优的学习率和权重衰减。

**其他超参数** Muon 包含另外两个可调超参数：Newton-Schulz 迭代步数和动量 $\mu$。我们经验性地观察到，当将 $N$ 设置为 10 时，迭代过程会比 $N = 5$ 产生更准确的正交化结果，但不会带来更好的性能。因此，为了效率起见，我们在本工作中设置 $N = 5$。我们没有看到调整动量带来一致的性能提升，因此我们选择 0.95，与K. Jordan et al. 2024相同。

### 2.3 分布式 Muon

**ZeRO-1 和 Megatron-LM** Rajbhandari et al. 2020引入了 ZeRO-1 技术，将昂贵的优化器状态（例如主权重、动量）分区到整个集群。Megatron-LM (Shoeybi et al. 2020) 将 ZeRO-1 集成到其原生并行设计中。基于 Megatron-LM 的复杂并行策略，例如张量并行（TP）、流水线并行（PP）、专家并行（EP）和数据并行（DP），ZeRO-1 的通信工作负载可以从在整个分布式世界范围内收集减少到仅在数据并行组内收集。

**方法** ZeRO-1 对 AdamW 高效，因为它以元素级方式计算更新。然而，Muon 需要完整的梯度矩阵来计算更新。因此，原始 ZeRO-1 不能直接应用于 Muon。我们提出了一种基于 ZeRO-1 的 Muon 新分布式解决方案，称为分布式 Muon。分布式 Muon 遵循 ZeRO-1 在 DP 上分区优化器状态，并相比原始 Zero-1 AdamW 优化器引入了两个额外操作：

1. `DP Gather`。对于本地 DP 分区的主权重（模型权重的 $1/DP$ 大小），此操作是将相应的分区梯度收集到完整的梯度矩阵中。
2. 计算完整更新。在上述收集之后，按照第 2.1节的描述在完整梯度矩阵上执行 Newton-Schulz 迭代步骤。注意，我们将丢弃完整更新矩阵的部分内容，因为执行更新时只需要与本地参数对应的分区。

分布式 Muon 的实现描述在算法 1中。分布式 Muon 引入的额外操作以蓝色标出。

[4]K. Jordan et al. 2024的原始实现按 $\sqrt{\max(1, A/B)}$ 缩放更新，如果所有矩阵具有相同的第二维度，则这与我们的建议（在全局尺度上）等价；Pethick et al. 2025和You 2025在与我们工作同时讨论了更新缩放因子的类似问题。

**Algorithm 1** 分布式 Muon

**Require:** 完整梯度 $\mathbf{G}$、DP 分区动量 $\mathbf{m}$、DP 分区参数 $\mathbf{p}$、动量 $\mu$。

// 在 DP 上对 $G$ 进行 reduce-scatter 以获得正确梯度
$\mathbf{g} = \text{reduce\_scatter}(\mathbf{G}, \text{dp\_group})$
// 使用本地分区动量 $\mathbf{m}$ 对 $\mathbf{g}$ 应用动量
$\mathbf{g}' = \text{update\_with\_momentum}(\mathbf{g}, \mathbf{m}, \mu)$
// DP Gather：在 DP 上收集 $\mathbf{g}'$ 到完整矩阵 $\mathbf{G}$
$\mathbf{G} = \text{gather}(\mathbf{g}', \text{dp\_group})$
// 计算 Muon 更新
$\mathbf{U} = \text{Newton-Schulz}(\mathbf{G})$
// 丢弃 $\mathbf{U}$ 的其余部分，仅保留本地分区 $\mathbf{u}$，然后应用更新规则
$\mathbf{p}' = \text{apply\_update}(\mathbf{p}, \mathbf{u})$
// All-gather 更新后的 $\mathbf{p}'$ 到 $\mathbf{P}$
$\mathbf{P} = \text{all\_gather}(\mathbf{p}', \text{dp\_group})$
// 返回更新 RMS 用于日志记录
**return** $\sqrt{\mathbf{u}^2.\texttt{mean}()}$

**分析** 我们将分布式 Muon 与经典的基于 ZeRO-1 的分布式 AdamW（为简化称为分布式 AdamW）在几个方面进行了比较：

- 内存使用。Muon 仅使用一个动量缓冲区，而 AdamW 使用两个动量缓冲区。因此，Muon 优化器使用的额外内存是分布式 AdamW 的一半。
- 通信开销。对于每个设备，额外的 DP 收集仅需要本地 DP 分区参数 $\mathbf{p}$。因此，通信成本小于 $\mathbf{G}$ 的 reduce-scatter 或 $\mathbf{P}$ 的 all-gather。此外，Muon 仅在 bf16 中需要 Newton-Schulz 迭代步骤，因此与 fp32 相比进一步将通信开销降低到 50%。总体而言，分布式 Muon 的通信工作量是分布式 AdamW 的 $(1, 1.25]$ 倍。上界计算为分布式 Muon 的通信量是 4（fp32 $\mathbf{G}$ reduce-scatter）+ 2（bf16 Muon gather）+ 4（fp32 $\mathbf{P}$ all-gather），而分布式 AdamW 是 4 + 4。在实践中，由于我们通常使用多个 DP 进行训练，经验上的额外成本通常更接近下界 1[5]。
- 延迟。分布式 Muon 比分布式 AdamW 具有更大的端到端延迟，因为它引入了额外的通信并需要运行 Newton-Schulz 迭代步骤。然而，这不是一个重大问题，因为 (a) 只需要约 5 个 Newton-Schulz 迭代步骤即可获得良好结果（在第 2.2节讨论），以及 (b) 优化器引起的端到端延迟与模型的前向-后向传递时间相比可以忽略不计（例如通常为 1% 到 3%）。此外，几种工程技术，如重叠收集和计算，以及重叠优化器 reduce-scatter 与参数收集，可以进一步降低延迟。

在我们的分布式集群中训练大规模模型时，分布式 Muon 与其 AdamW 对应版本相比没有明显的延迟开销。我们将很快发布一个为开源 Megatron-LM (Shoeybi et al. 2020) 项目实现分布式 Muon 的 pull request。

[5]如果启用 TP，分布式 Muon 需要在 TP 组上进行额外的 bf16 TP 收集。

表 1: 控制 Muon 在不同模型参数间的更新 RMS

| 方法 | 训练损失 | 验证损失 | 查询权重 RMS | MLP 权重 RMS |
|---|---|---|---|---|
| 基线 | 2.734 | 2.812 | 3.586e-2 | 2.52e-2 |
| 更新范数 | **2.72** | **2.789** | 4.918e-2 | 5.01e-2 |
| 调整学习率 | 2.721 | **2.789** | 3.496e-2 | 4.89e-2 |

## 3 实验

### 3.1 一致的更新 RMS

如第 2.2节所讨论的，我们的目标是匹配所有矩阵参数的更新 RMS，并使其与 AdamW 的更新 RMS 相匹配。我们尝试了两种方法来控制 Muon 在参数间的更新 RMS，并将其与仅保持与 AdamW 一致 RMS 的基线进行比较：

1. 基线。我们将更新矩阵乘以 $0.2 \cdot \sqrt{H}$（$H$ 是模型隐藏层大小）以保持与 AdamW 一致的更新 RMS。注意，对于大多数矩阵，$\max(A, B)$ 等于 $H$。

$$\mathbf{W}_t = \mathbf{W}_{t-1} - \eta_t(0.2 \cdot \mathbf{O}_t \cdot \sqrt{H} + \lambda \mathbf{W}_{t-1}) \tag{5}$$

2. 更新范数。我们可以直接归一化通过 Newton-Schulz 迭代计算的更新，使其 RMS 严格变为 0.2；

$$\mathbf{W}_t = \mathbf{W}_{t-1} - \eta_t(0.2 \cdot \mathbf{O}_t / \operatorname{RMS}(\mathbf{O}_t) + \lambda \mathbf{W}_{t-1}) \tag{6}$$

3. 调整学习率。对于每个更新矩阵，我们可以根据其形状按因子 $0.2 \cdot \sqrt{\max(A, B)}$ 缩放其学习率。

$$\mathbf{W}_t = \mathbf{W}_{t-1} - \eta_t(0.2 \cdot \mathbf{O}_t \cdot \sqrt{\max(A, B)} + \lambda \mathbf{W}_{t-1}) \tag{7}$$

**分析** 我们设计了实验来说明 Muon 更新 RMS 在训练早期阶段的影响，因为我们观察到在更大规模训练模型时，意外行为很快就会出现。我们使用如 3.2节所述的小规模 800M 模型进行实验。当矩阵维度之间的差异增大时，不一致更新 RMS 的问题更加明显。为了突出问题以供进一步研究，我们通过将 Swiglu MLP 替换为标准 2 层 MLP，将其矩阵参数的形状从 $[H, 2.6H]$ 改为 $[H, 4H]$，对模型架构进行了轻微修改。我们评估了模型的损失并监测了其部分参数的 RMS，特别是注意力查询（形状 $[H, H]$）和 MLP（形状 $[H, 4H]$）。我们在 20B token 训练计划中的 4B token 后评估了模型。从表 1中，我们观察到几个有趣的发现：

1. 更新范数和调整学习率都取得了比基线更好的性能；
2. 对于形状为 $[H, 4H]$ 的 MLP 权重矩阵，更新范数和调整学习率获得的权重 RMS 大约是基线的两倍。这是合理的，因为 $\sqrt{\max(H, 4H)}/\sqrt{H} = 2$，所以更新范数和调整学习率的更新 RMS 大约是基线的两倍；
3. 对于形状为 $[H, H]$ 的注意力查询权重矩阵，更新范数仍然对更新进行归一化，而调整学习率不这样做，因为 $\sqrt{\max(H, H)}/\sqrt{H} = 1$。因此，调整学习率产生的权重 RMS 与基线相似，但更新范数具有与其 MLP 相似的最大权重 RMS。

表 2: 扩展定律模型和超参数

| # 参数（不含嵌入层） | 头数 | 层数 | 隐藏层大小 | Token 数 | 学习率 | 批次大小 * |
|---|---|---|---|---|---|---|
| 399M | 12 | 12 | 1536 | 8.92B | 9.503e-4 | 96 |
| 545M | 14 | 14 | 1792 | 14.04B | 9.143e-4 | 128 |
| 822M | 16 | 16 | 2048 | 20.76B | 8.825e-4 | 160 |
| 1.1B | 18 | 18 | 2304 | 28.54B | 8.561e-4 | 192 |
| 1.5B | 20 | 20 | 2560 | 38.91B | 8.305e-4 | 256 |

* 以 8K 上下文长度中的样本数为单位。

基于这些发现，我们选择调整学习率方法用于未来实验，因为它的成本更低。

### 3.2 Muon 的扩展定律

为了与 AdamW 进行公平比较，我们在一系列 Llama (Grattafiori et al. 2024) 架构的稠密模型上进行了扩展定律实验。在优化器研究中，建立强基线至关重要。因此，我们按照计算最优训练设置 (Kaplan et al. 2020) 对 AdamW 的超参数进行了网格搜索（网格搜索实验见附录 B）。模型架构和超参数的详细信息见表 2。对于 Muon，如第 2.2节所讨论的，由于我们匹配了 Muon 与 AdamW 的更新 RMS，我们直接复用了对 AdamW 基线最优的超参数。

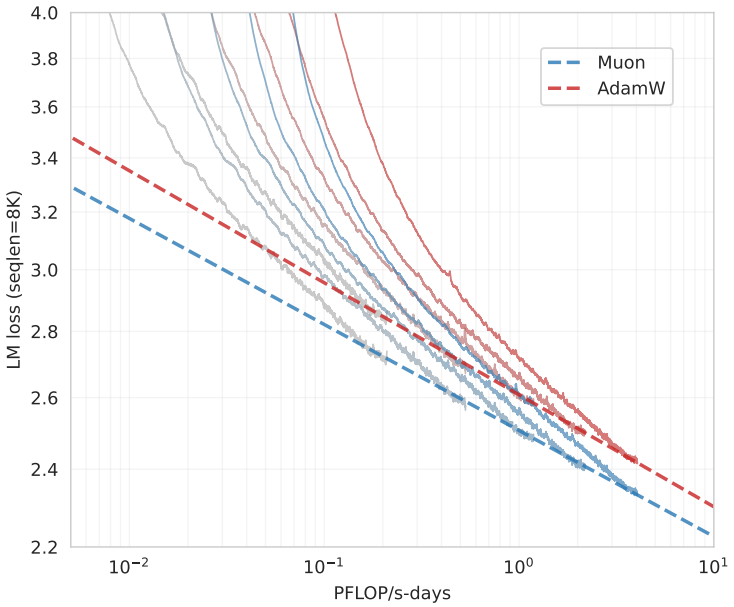



图 3: Muon 和 AdamW 优化器的拟合扩展定律曲线。

拟合的扩展定律曲线见图 3，拟合方程详见表 3。如图 1a所示，在计算最优设置下，Muon 仅需约 52% 的训练 FLOPs 即可匹配 AdamW 的性能。

表 3: 扩展定律曲线的拟合参数

| | Muon | AdamW |
|---|---|---|
| LM 损失（序列长度 =8K） | $2.506 \times C^{-0.052}$ | $2.608 \times C^{-0.054}$ |

### 3.3 使用 Muon 进行预训练

**模型架构** 为了评估 Muon 与当代模型架构的对比，我们使用 deepseek-v3-small 架构 (DeepSeek-AI et al. 2024) 从头开始预训练，因为它展示了强大的性能，且原始结果可作为比较的参考。我们的预训练模型有 22.4 亿激活参数和 152.9 亿总参数（包含嵌入层时为 30 亿激活参数和 160 亿总参数）。对架构的轻微修改详见附录 C。

**预训练数据** 我们的预训练数据详情见K. Team 2025。预训练期间的最大上下文长度为 8K。

**预训练** 模型分几个阶段训练。我们在阶段 1 和 2 使用 1e-3 的 auxfree 偏置更新率，在阶段 3 使用 0.0 的 auxfree 偏置更新率。所有阶段的权重衰减设置为 0.1。更多细节和模型训练讨论见附录 D。

1. `0` 到 `33B token`：在此阶段，学习率在 2k 步内线性增加到 4.2e-4。批次大小保持在 2048 个样本；
2. `33B` 到 `5.2T token`：在此阶段，学习率以余弦方式从 4.2e-4 衰减到 4.2e-5。我们将批次大小保持在 2048 直到 200B token，然后加倍到 4096；
3. `5.2T` 到 `5.7T token`：在此阶段（也称为冷却阶段），学习率在 100 步内增加到 1e-4，然后在 500B token 内线性衰减到 0，我们保持恒定的 4096 批次大小。在此阶段，我们使用最高质量的数据，专注于数学、代码和推理。

**评估基准** 我们的评估涵盖四类主要基准，每类旨在评估模型的不同能力：

- **英语语言理解与推理**：MMLU（5-shot）(Hendrycks, Burns, Basart, et al. 2021)、MMLU-pro（5-shot）(Wang et al. 2024)、BBH（3-shot）(Suzgun et al. 2022)、TriviaQA（5-shot）(Joshi et al. 2017)
- **代码生成**：HumanEval（pass@1）(M. Chen et al. 2021)、MBPP（pass@1）(Austin et al. 2021)
- **数学推理**：GSM8K（4-shot）(**cobbe2021trainingverifiersolvemath**)、MATH (Hendrycks, Burns, Kadavath, et al. 2021)、CMATH (Wei et al. 2023)
- **中文语言理解与推理**：C-Eval（5-shot）(Y. Huang et al. 2023)、CMMLU（5-shot）(H. Li et al. 2024)

**性能** 我们将使用 Muon 训练的模型命名为 “Moonlight”。我们将 Moonlight 与不同规模的公开模型进行了比较。我们首先评估了在 1.2T token 时的 Moonlight，并将其与以下具有相同架构且使用相当数量 token 训练的模型进行比较：

- `Deepseek-v3-Small`（DeepSeek-AI et al. 2024）是一个 24 亿/160 亿参数的 MoE 模型，使用 1.33T token 训练；

- `Moonlight-A` 遵循与 Moonlight 相同的训练设置，但使用 AdamW 优化器。

对于 Moonlight 和 Moonlight-A，我们使用了总共 5.7T 预训练中的 1.2T token 中间检查点，此时学习率尚未衰减到最小，模型尚未经过冷却阶段。

表 4: 不同模型在约 1.2T token 时的比较。

| | 基准（指标） | DSV3-Small | Moonlight-A@1.2T | Moonlight@1.2T |
|---|---|---|---|---|
| | 激活参数 † | 2.24B | 2.24B | 2.24B |
| | 总参数 † | 15.29B | 15.29B | 15.29B |
| | 训练 Token | 1.33T | 1.2T | 1.2T |
| | 优化器 | AdamW | AdamW | Muon |
| 英语 | MMLU | 53.3 | 60.2 | **60.4** |
| | MMLU-pro | - | 26.8 | **28.1** |
| | BBH | 41.4 | **45.3** | 43.2 |
| | TriviaQA | - | 57.4 | **58.1** |
| 代码 | HumanEval | 26.8 | 29.3 | **37.2** |
| | MBPP | 36.8 | 49.2 | **52.9** |
| 数学 | GSM8K | 31.4 | 43.8 | **45.0** |
| | MATH | 10.7 | 16.1 | **19.8** |
| | CMath | - | 57.8 | **60.2** |
| 中文 | C-Eval | - | 57.2 | **59.9** |
| | CMMLU | - | 58.2 | **58.8** |

† 报告的参数计数不包含嵌入层参数。

如表 4所示，我们使用 AdamW 训练的基线模型 Moonlight-A 与类似的公开模型相比表现出强劲性能。Moonlight 显著优于 Moonlight-A，证明了 Muon 的扩展有效性。我们观察到 Muon 在数学和代码相关任务上表现尤为出色，我们鼓励研究界进一步调查这一现象。在 Moonlight 完全训练到 5.7T token 后，我们将其与类似规模的公开模型进行比较，结果如表 5所示：

- `LLAMA3-3B` 来自Grattafiori et al. 2024，是一个 30 亿参数的稠密模型，使用 9T token 训练。
- `Qwen2.5-3B` 来自Yang et al. 2024，是一个 30 亿参数的稠密模型，使用 18T token 训练。
- `Deepseek-v2-Lite` 来自DeepSeek-AI 2024，是一个 24 亿/160 亿参数的 MOE 模型，使用 5.7T token 训练。

如表 5所示，Moonlight 优于使用相似架构且使用等量 token 训练的模型。即使与使用更大规模数据集训练的稠密模型相比，Moonlight 仍保持有竞争力的性能。详细比较见附录 E。Moonlight 的性能进一步与其他知名语言模型在 MMLU 和 GSM8k 上进行比较，如图 1b和附录 E图 8所示[6]。值得注意的是，Moonlight 位于模型性能与训练预算的帕累托前沿上，在各种规模上优于许多其他模型。

[6]基线模型的性能指标和计算需求（FLOPs）来自 (OLMo et al. 2024)。

表 5: 不同模型在各种基准上的比较。

| | 基准（指标） | Llama3.2-3B | Qwen2.5-3B | DSV2-Lite | Moonlight |
|---|---|---|---|---|---|
| | 激活参数 † | 2.81B | 2.77B | 2.24B | 2.24B |
| | 总参数 † | 2.81B | 2.77B | 15.29B | 15.29B |
| | 训练 Token | 9T | 18T | 5.7T | 5.7T |
| | 优化器 | AdamW | 未知 | AdamW | Muon |
| 英语 | MMLU | 54.7 | 65.6 | 58.3 | **70.0** |
| | MMLU-pro | 25.0 | 34.6 | 25.5 | **42.4** |
| | BBH | 46.8 | 56.3 | 44.1 | **65.2** |
| | TriviaQA‡ | 59.6 | 51.1 | 65.1 | **66.3** |
| 代码 | HumanEval | 28.0 | 42.1 | 29.9 | **48.1** |
| | MBPP | 48.7 | 57.1 | 43.2 | **63.8** |
| 数学 | GSM8K | 34.0 | **79.1** | 41.1 | 77.4 |
| | MATH | 8.5 | 42.6 | 17.1 | **45.3** |
| | CMath | - | 80.0 | 58.4 | **81.1** |
| 中文 | C-Eval | - | 75.0 | 60.3 | **77.2** |
| | CMMLU | - | 75.0 | 64.3 | **78.2** |

† 报告的参数计数不包含嵌入层参数。‡ 我们使用完整的 TriviaQA 测试集测试所有列出的模型。

### 3.4 奇异谱动态

为了验证 Muon 能够在更多样化的方向上优化权重矩阵的直觉，我们对使用 Muon 和 AdamW 训练的权重矩阵进行了谱分析。对于具有奇异值 $\sigma = (\sigma_1, \sigma_2, \cdots, \sigma_n)$ 的权重矩阵，我们按如下方式计算该矩阵的 SVD 熵 (Alter et al. 2000; Roy et al. 2007)：

$$H(\sigma) = -\frac{1}{\log n} \sum_{i=1}^{n} \frac{\sigma_i^2}{\sum_{j=1}^{n} \sigma_j^2} \log \frac{\sigma_i^2}{\sum_{j=1}^{n} \sigma_j^2}$$

如图 4所示，我们可视化了在 1.2T token 预训练期间不同训练检查点处权重矩阵的平均 SVD 熵。我们可以看到，在所有训练检查点和所有权重矩阵组中，Muon 的 SVD 熵高于 AdamW，这验证了 Muon 能够为权重矩阵提供更多样化谱更新的直觉。这种差异在专家选择的路由器权重中更为显著，表明混合专家模型可以从 Muon 中获益更多。

此外，我们可视化了在 1.2T token 训练检查点处每个权重矩阵的奇异值分布，如附录 F所示。我们发现，对于超过 90% 的权重矩阵，Muon 优化时的 SVD 熵高于 AdamW，为 Muon 在探索多样化优化方向方面的优越能力提供了强有力的实证证据。

### 3.5 使用 Muon 进行监督微调（SFT）

在本节中，我们在大语言模型训练的标准 SFT 阶段对 Muon 优化器进行了消融研究。我们的发现表明，Muon 带来的益处持续存在于 SFT 阶段。具体来说，同时使用 Muon 进行预训练和微调的模型在消融研究中表现最佳。然而，我们也观察到，当 SFT 优化器与预训练优化器不同时，使用 Muon 进行 SFT 相比 AdamW 并未显示出显著优势。这表明仍有相当大的探索空间，我们将其留待未来工作。

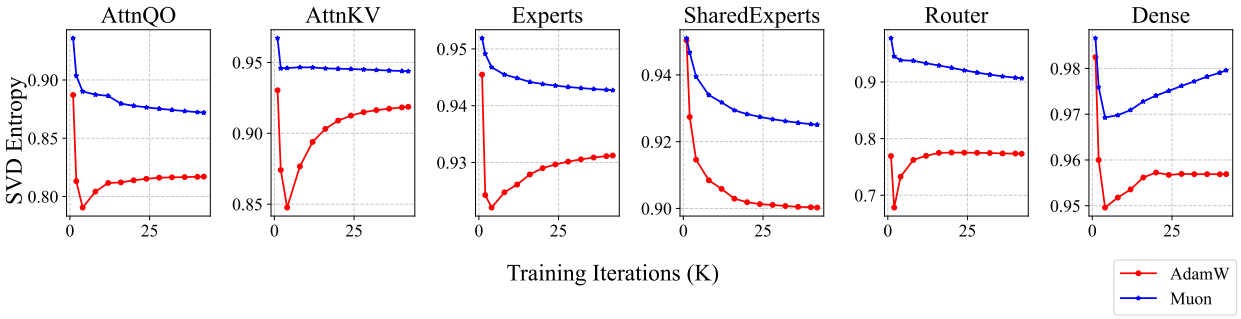



图 4: 不同训练迭代中权重矩阵的 SVD 熵。我们将权重矩阵分为 6 个不同组：1）AttnQO 表示注意力层中与查询和输出投影相关的权重矩阵；2）AttnKV 表示注意力层中与键和值投影相关的权重矩阵；3）Experts 表示专家模型中的权重矩阵；4）SharedExperts 表示共享专家模型中的权重矩阵；5）Router 表示路由器中的权重矩阵；6）Dense 表示第一层稠密层中的权重矩阵。SVD 熵计算为每层各组中权重矩阵的宏平均。对于专家模型中的权重，为效率起见，我们仅计算不同层中 64 个专家中的 3 个。

#### 3.5.1 预训练与 SFT 优化器可互换性的消融研究

为了进一步研究 Muon 的潜力，我们使用 Muon 和 AdamW 优化器对 Moonlight@1.2T 和 Moonlight-A@1.2T 进行了微调。这些模型在开源 tulu-3-sft-mixture 数据集（Lambert et al. 2024）上微调了两个 epoch，该数据集包含 4k 序列长度数据。学习率遵循线性衰减计划，从 $5 \times 10^{-5}$ 开始逐渐降低到 0。结果如表 6所示，突出了 Moonlight@1.2T 相比 Moonlight-A@1.2T 的优越性能。

表 6: 检验预训练与 SFT 阶段之间优化器可互换性的影响。

| **基准（指标）** | **# Shots** | **Moonlight-1.2T** | | | |
|---|---|---|---|---|---|
| 预训练优化器 | - | Muon | AdamW | Muon | AdamW |
| SFT 优化器 | - | Muon | Muon | AdamW | AdamW |
| MMLU (EM) | 0-shot (CoT) | **55.7** | 55.3 | 50.2 | 52.0 |
| HumanEval (Pass@1) | 0-shot | **57.3** | 53.7 | 52.4 | 53.1 |
| MBPP (Pass@1) | 0-shot | **55.6** | 55.5 | 55.2 | 55.2 |
| GSM8K (EM) | 5-shot | **68.0** | 62.1 | 64.9 | 64.6 |

#### 3.5.2 在公开预训练模型上使用 Muon 进行 SFT

我们进一步将 Muon 应用于公开预训练模型的监督微调（SFT），具体是 Qwen2.5-7B 基础模型（Yang et al. 2024），使用开源 tulu-3-sft-mixture 数据集（Lambert et al. 2024）。数据集以 8k 序列长度打包，我们采用余弦衰减学习率计划，从 $2 \times 10^{-5}$ 开始逐渐降低到 $2 \times 10^{-6}$。结果如表 7所示。作为比较，我们展示了 Muon 微调模型与 Adam 微调模型性能相当。这些结果表明，为了获得最佳性能，在预训练阶段应用 Muon 比在监督微调阶段应用更有效。

## 4 讨论

有几个可能的研究方向可以进一步探索和扩展当前发现。

表 7: Adam 和 Muon 优化器应用于 Qwen2.5-7B 预训练模型 SFT 的比较。

| 基准（指标） | # Shots | Adam-SFT | Muon-SFT |
|---|---|---|---|
| 预训练模型 | - | Qwen2.5-7B | |
| MMLU (EM) | 0-shot (CoT) | **71.4** | 70.8 |
| HumanEval (Pass@1) | 0-shot | **79.3** | 77.4 |
| MBPP (Pass@1) | 0-shot | **71.9** | 71.6 |
| GSM8K (EM) | 5-shot | **89.8** | 85.8 |

**将所有参数纳入 Muon 框架** 目前，Muon 优化器与 Adam 优化器配合使用，某些参数仍由 Adam 优化器处理。这种混合方法虽然可行，但提供了改进的机会。将所有参数的优化完全整合到 Muon 框架内是一个重要的研究课题。

**将 Muon 扩展到 Schatten 范数** Muon 优化器可以解释为谱范数下的最速下降方法。鉴于 Schatten 范数的广泛适用性和多功能性，将 Muon 扩展到涵盖一般 Schatten 范数是一个有前景的方向。这种扩展可能解锁额外的优化能力，并可能产生比当前基于谱范数的实现更优的结果。

**理解和解决预训练-微调不匹配问题** 实践中观察到的一个显著现象是，使用 AdamW 预训练的模型在使用 Muon 微调时表现不佳，反之亦然。这种优化器不匹配对有效利用大量 AdamW 预训练检查点库构成了重大障碍，因此需要严格的理论调查。对底层机制的精确理解对于设计稳健有效的解决方案至关重要。

## 5 结论

在本技术报告中，我们对 Muon 在大语言模型训练中的可扩展性进行了全面研究。通过系统分析和改进，我们成功将 Muon 应用于一个使用 5.7 万亿 token 训练的 30 亿/160 亿参数 MoE 模型。我们的结果表明，Muon 能够有效替代 AdamW 成为大规模大语言模型训练的标准优化器，在训练效率和模型性能方面都提供了显著优势。通过开源我们的实现、Moonlight 模型和中间训练检查点，我们旨在促进可扩展优化技术的进一步研究，并加速大语言模型训练方法的发展。

## References


Alter, Orly, Patrick O. Brown, and David Botstein. "Singular value decomposition for genome-wide expression data processing and modeling". In: *Proceedings of the National Academy of Sciences* 97.18 (2000), pp. 10101–10106. DOI: 10.1073/pnas.97.18.10101. eprint: https://www.pnas.org/doi/pdf/10.1073/pnas.97.18.10101. URL: https://www.pnas.org/doi/abs/10.1073/pnas.97.18.10101.

Austin, Jacob et al. *Program Synthesis with Large Language Models*. 2021. arXiv: 2108.07732 [cs.PL]. URL: https://arxiv.org/abs/2108.07732.

Bernstein, Jeremy and Laker Newhouse. *Old Optimizer, New Norm: An Anthology*. 2024. arXiv: 2409.20325 [cs.LG]. URL: https://arxiv.org/abs/2409.20325.

Bi, Xiao et al. "Deepseek llm: Scaling open-source language models with longtermism". In: *arXiv preprint arXiv:2401.02954* (2024).

Cesista, Franz Louis. *Deep Learning Optimizers as Steepest Descent in Normed Spaces*. 2024. URL: http://leloykun.github.io/ponder/steepest-descent-opt/.

Chen, Mark et al. "Evaluating Large Language Models Trained on Code". In: (2021). arXiv: 2107.03374 [cs.LG].

DeepSeek-AI. *DeepSeek-V2: A Strong, Economical, and Efficient Mixture-of-Experts Language Model*. 2024. arXiv: 2405.04434 [cs.CL].

DeepSeek-AI et al. *DeepSeek-V3 Technical Report*. 2024. arXiv: 2412.19437 [cs.CL]. URL: https://arxiv.org/abs/2412.19437.

Franz, Louis Cesista. *The Case for Muon*. Oct. 2024. URL: https://x.com/leloykun/status/1846842887839125941 (visited on 02/18/2025).

Grattafiori, Aaron et al. *The Llama 3 Herd of Models*. 2024. arXiv: 2407.21783 [cs.AI]. URL: https://arxiv.org/abs/2407.21783.

Hendrycks, Dan, Collin Burns, Steven Basart, et al. *Measuring Massive Multitask Language Understanding*. 2021. arXiv: 2009.03300 [cs.CY]. URL: https://arxiv.org/abs/2009.03300.

Hendrycks, Dan, Collin Burns, Saurav Kadavath, et al. *Measuring Mathematical Problem Solving With the MATH Dataset*. 2021. arXiv: 2103.03874 [cs.LG]. URL: https://arxiv.org/abs/2103.03874.

Hoffmann, Jordan et al. *Training Compute-Optimal Large Language Models*. 2022. arXiv: 2203.15556 [cs.CL]. URL: https://arxiv.org/abs/2203.15556.

Huang, Yuzhen et al. *C-Eval: A Multi-Level Multi-Discipline Chinese Evaluation Suite for Foundation Models*. 2023. arXiv: 2305.08322 [cs.CL]. URL: https://arxiv.org/abs/2305.08322.

Jordan, Keller et al. *Muon: An optimizer for hidden layers in neural networks*. 2024. URL: https://kellerjordan.github.io/posts/muon/.

Joshi, Mandar et al. *TriviaQA: A Large Scale Distantly Supervised Challenge Dataset for Reading Comprehension*. 2017. arXiv: 1705.03551 [cs.CL]. URL: https://arxiv.org/abs/1705.03551.

Kaplan, Jared et al. *Scaling Laws for Neural Language Models*. 2020. arXiv: 2001.08361 [cs.LG]. URL: https://arxiv.org/abs/2001.08361.

Kingma, Diederik P. and Jimmy Ba. "Adam: A Method for Stochastic Optimization". In: *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. Ed. by Yoshua Bengio and Yann LeCun. 2015. URL: http://arxiv.org/abs/1412.6980.

Lambert, Nathan et al. "Tülu 3: Pushing Frontiers in Open Language Model Post-Training". In: (2024).

Li, Haonan et al. *CMMLU: Measuring massive multitask language understanding in Chinese*. 2024. arXiv: 2306.09212 [cs.CL]. URL: https://arxiv.org/abs/2306.09212.

Li, Xi-Lin. "Preconditioned Stochastic Gradient Descent". In: *IEEE Transactions on Neural Networks and Learning Systems* 29.5 (May 2018), pp. 1454–1466. ISSN: 2162-2388. DOI: 10.1109/tnnls.2017.2672978. URL: http://dx.doi.org/10.1109/TNNLS.2017.2672978.

– *Preconditioner on Matrix Lie Group for SGD*. 2018. arXiv: 1809.10232 [stat.ML]. URL: https://arxiv.org/abs/1809.10232.

– *Stochastic Hessian Fittings with Lie Groups*. 2024. arXiv: 2402.11858 [stat.ML]. URL: https://arxiv.org/abs/2402.11858.

Li, Xilin. *Black Box Lie Group Preconditioners for SGD*. 2022. arXiv: 2211.04422 [stat.ML]. URL: https://arxiv.org/abs/2211.04422.

Liu, Hong et al. "Sophia: A Scalable Stochastic Second-order Optimizer for Language Model Pre-training". In: *The Twelfth International Conference on Learning Representations*. 2024. URL: https://openreview.net/forum?id=3xHDeA8Noi.

Loshchilov, Ilya and Frank Hutter. "Decoupled Weight Decay Regularization". In: *International Conference on Learning Representations*. 2019. URL: https://openreview.net/forum?id=Bkg6RiCqY7.

OLMo, Team et al. "2 OLMo 2 Furious". In: *arXiv preprint arXiv:2501.00656* (2024).

OpenAI et al. *GPT-4 Technical Report*. 2024. arXiv: 2303.08774 [cs.CL]. URL: https://arxiv.org/abs/2303.08774.

Pethick, Thomas et al. *Training Deep Learning Models with Norm-Constrained LMOs*. 2025. arXiv: 2502.07529 [cs.LG]. URL: https://arxiv.org/abs/2502.07529.

Pooladzandi, Omead and Xi-Lin Li. *Curvature-Informed SGD via General Purpose Lie-Group Preconditioners*. 2024. arXiv: 2402.04553 [cs.LG]. URL: https://arxiv.org/abs/2402.04553.

Rajbhandari, Samyam et al. "ZeRO: Memory optimizations Toward Training Trillion Parameter Models". In: (Nov. 2020), pp. 1–16. DOI: 10.1109/sc41405.2020.00024. URL: http://dx.doi.org/10.1109/SC41405.2020.00024.

Roy, Olivier and Martin Vetterli. "The effective rank: A measure of effective dimensionality". In: *2007 15th European Signal Processing Conference*. 2007, pp. 606–610.

Shazeer, Noam. *Fast Transformer Decoding: One Write-Head is All You Need*. 2019. arXiv: 1911.02150 [cs.NE]. URL: https://arxiv.org/abs/1911.02150.

Shoeybi, Mohammad et al. *Megatron-LM: Training Multi-Billion Parameter Language Models Using Model Parallelism*. 2020. arXiv: 1909.08053 [cs.CL]. URL: https://arxiv.org/abs/1909.08053.

Suzgun, Mirac et al. *Challenging BIG-Bench Tasks and Whether Chain-of-Thought Can Solve Them*. 2022. arXiv: 2210.09261 [cs.CL]. URL: https://arxiv.org/abs/2210.09261.

Team, Gemini et al. *Gemini: A Family of Highly Capable Multimodal Models*. 2024. arXiv: 2312.11805 [cs.CL]. URL: https://arxiv.org/abs/2312.11805.

Team, Gemma et al. "Gemma 2: Improving open language models at a practical size". In: *arXiv preprint arXiv:2408.00118* (2024).

Team, Kimi. "Kimi k1.5: Scaling Reinforcement Learning with LLMs". In: (2025).

Vyas, Nikhil et al. "SOAP: Improving and Stabilizing Shampoo using Adam". In: *The Thirteenth International Conference on Learning Representations*. 2025. URL: https://openreview.net/forum?id=IDxZhXrpNf.

Wang, Yubo et al. *MMLU-Pro: A More Robust and Challenging Multi-Task Language Understanding Benchmark*. 2024. arXiv: 2406.01574 [cs.CL]. URL: https://arxiv.org/abs/2406.01574.

Wei, Tianwen et al. *CMATH: Can Your Language Model Pass Chinese Elementary School Math Test?* 2023. arXiv: 2306.16636 [cs.CL]. URL: https://arxiv.org/abs/2306.16636.

Yang, An et al. "Qwen2.5 Technical Report". In: *arXiv preprint arXiv:2412.15115* (2024).

You, Jiacheng. *Jiacheng You's discussion on Muon's Update RMS*. 2025. URL: https://x.com/YouJiacheng/status/1890094769386451309.

Yuan, Huizhuo et al. *MARS: Unleashing the Power of Variance Reduction for Training Large Models*. 2024. arXiv: 2411.10438 [cs.LG].

# A 更新 RMS

### 引理 1 的证明

证明. 不失一般性，考虑正交矩阵 $U \in \mathbb{R}^{n\times n}$ 和 $V \in \mathbb{R}^{m\times m}$，其中 $n \geq m \geq r$。我们将证明对于 $X = U_{[:,:r]}V_{[:r,:]}$（Muon 的更新具有相同格式），RMS 值为 $\sqrt{r/mn}$。根据矩阵乘法的定义：

$$X_{i,j} = \sum_{k=1}^{r} U_{i,k}V_{k,j}$$

RMS 可表示为：

$$\begin{aligned}
\mathrm{RMS}(X)^2 &= \frac{1}{mn}\sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{r} U_{i,k}^2 V_{k,j}^2 \\
&= \frac{1}{mn}\sum_{k=1}^{r}\left(\sum_{i=1}^{n} U_{i,k}^2\right)\left(\sum_{j=1}^{m} V_{k,j}^2\right) \\
&= \frac{1}{mn}\sum_{k=1}^{r} 1 \\
&= \frac{r}{mn}
\end{aligned}$$

因此，$\mathrm{RMS}(X) = \sqrt{r/mn}$。对于常见的满秩矩阵情况，$r = m$，得到 $\mathrm{RMS}(X) = \sqrt{1/n}$。 □

**Muon 与 AdamW 之间一致的更新 RMS** 如 2.2节所讨论的，我们希望匹配 Muon 和 AdamW 优化器之间的更新 RMS。这通过小规模模型实验验证。我们将 Muon 的更新 RMS 设置在 $[0.05, 0.1, 0.2, 0.4, 0.8]$ 范围内，以 AdamW 为基线。我们在表 8中报告了 2k 步（约 2B token）时的损失和代表性权重矩阵 RMS。从结果中，我们发现 0.2 RMS 和 0.4 RMS 表现相似，且远优于其他设置。这些发现与我们观察到的 AdamW 更新 RMS 在 $0.2 \sim 0.4$ 范围内的经验一致。我们选择将 Muon 的更新 RMS 控制在 0.2。

表 8: Muon 更新 RMS 实验

| 优化器 | AdamW | 0.05 RMS* | 0.1 RMS | 0.2 RMS | 0.4 RMS | 0.8 RMS |
|---|---|---|---|---|---|---|
| LM 训练损失 | 3.512 | 3.355 | 3.239 | **3.198** | 3.199 | 3.386 |
| LM 验证损失 | 3.679 | 3.503 | 3.374 | 3.325 | **3.314** | 3.543 |
| AttnQ 权重 RMS | 1.01e-2 | 5.74e-3 | 8.44e-3 | 1.57e-2 | 2.95e-2 | 7.23e-2 |
| Mlp 权重 RMS | 1.25e-2 | 8.01e-3 | 1.27e-2 | 2.35e-2 | 4.51e-2 | 8.73e-2 |

* 除第一列外，所有其他候选都使用具有受控 RMS 的 Muon。

# B AdamW 基线扩展定律

为了确保我们实验的公平性和准确性，我们在专有数据集上进行了一系列实验，以得出对 AdamW 最优的扩展定律参数。这包括在受限计算预算（FLOPs，$C$）下确定最优模型大小（$N$）、训练 token 数（$D$）、学习率（$\eta$）和批次大小（$B$）。(Kaplan et al. 2020; Hoffmann et al. 2022; Bi et al. 2024) 表 9展示了我们系统参数搜索过程的结果。

表 9: 扩展定律参数与计算预算（FLOPs）之间的经验关系

| $N(C)$ | $D(C)$ | $\eta(C)$ | $B(C)$ |
|---|---|---|---|
| $0.0483359 \cdot C^{0.5112684}$ | $3.4480927 \cdot C^{0.4887316}$ | $0.0127339 \cdot C^{-0.0574752}$ | $0.0065202 \cdot C^{0.4137915}$ |

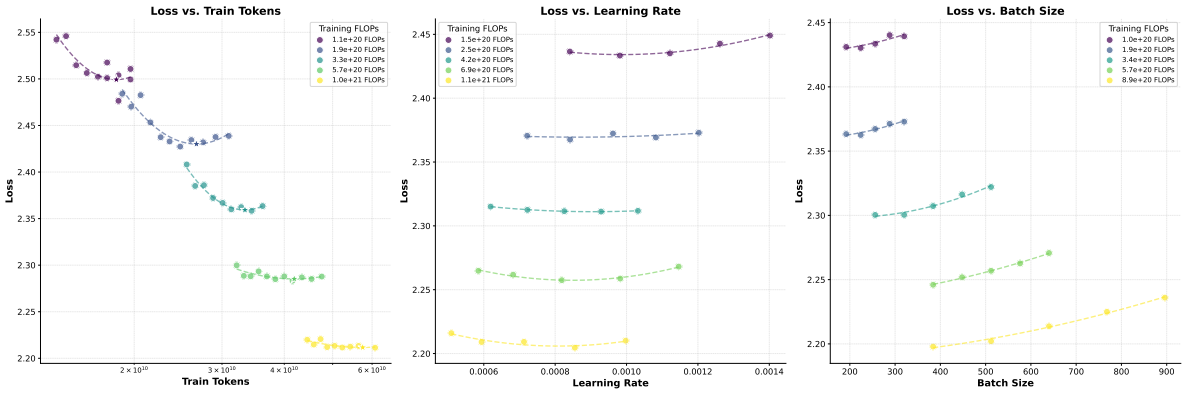



图 5: 跨 FLOPs 预算的扩展定律超参数优化景观

**超参数搜索** 为了系统地识别 AdamW 基线中的最优扩展定律超参数，我们采用了多阶段搜索协议。首先，我们选择多个计算预算（FLOPs 级别），并根据先前研究的经验指南初始化模型大小、学习率和批次大小。对于每个固定的 FLOPs 约束，我们在调整训练 token 数 $D$ 的同时改变模型大小 $N$，以保持 $C = 6ND$，从而探索模型容量与数据效率之间的权衡。每种配置都训练到收敛，并记录验证损失以确定 $N$ 和 $D$ 的帕累托最优组合。随后，在固定最优 $N - D$ 对的情况下，我们通过网格搜索细化学习率和批次大小，确保跨配置的稳定性和收敛性。为了减轻局部极小值并增强鲁棒性，该迭代过程重复 2-3 次，逐步缩小超参数空间。

优化过程进一步在图 5中说明，它描绘了损失景观作为训练 token、学习率和批次大小在不同 FLOPs 预算下的函数。每个碗状曲线代表特定 FLOPs 水平的损失表面，具有对应于最优超参数配置的不同全局最小值。

## C 模型架构

Muon 与模型架构无关，我们使用了与 Deepseek-V3-Small 相似的模型，如DeepSeek-AI et al. 2024所述，因为它是一个具有开源权重的强基线模型。我们在 Moonlight 模型中做了几处小修改，在此列出：

**多 token 预测（MTP）** MTP 在我们的实验中未显示出对预训练的显著益处。为简单起见，我们未将 MTP 层引入 Moonlight 模型。

**无辅助损失偏置更新** 在DeepSeek-AI et al. 2024中，无辅助损失偏置通过以下方式更新：$b_i = b_i + u \times \text{sign}(e_i)$，其中 $u$ 是更新率，$b_i$ 是第 i 个专家的偏置，$e_i$ 是专家的违规率。我们稍微修改了更新规则为：$b_i = b_i + u \times (\text{sign}(e_i) - \text{sign}(e).\text{mean}())$，其中 $\text{sign}(e).\text{mean}()$ 是所有专家违规率符号的平均值，以控制偏置的大小，同时不改变 topk 选择逻辑。

**门控缩放因子** Deepseek-V2-Lite 未使用门控缩放因子，Deepseek-V3 使用了 2.5 的缩放因子。我们使用 2.446 的缩放因子来控制与稠密模型相似的输出 rms。计算我们门控缩放因子的代码见图 6。

```python
import numpy as np

def sigmoid(x):
    return 1 / (1 + np.exp(-x))

def calc_gate_scaling_factor(num_experts: int, topk: int, iter_times: int):
    """Calculate the gate scaling factor for MoE.

    Args:
        num_experts (int): The number of experts.
        topk (int): The number of experts to select.
        iter_timers (int): The number of iterations.

    Returns:
        float: The gate scaling factor.
    """
    factors = []
    for _ in range(iter_times):

        # mock gaussian logits
        logits = np.random.randn(num_experts)
        # select topk logits
        p = np.sort(sigmoid(logits))[::-1]
        p = p[:topk]
        # renormalize
        p = p / p.sum()
        # calculate the scaling factor
        factors.append( 1/ (p**2).sum()**0.5)
    return np.mean(factors)
```

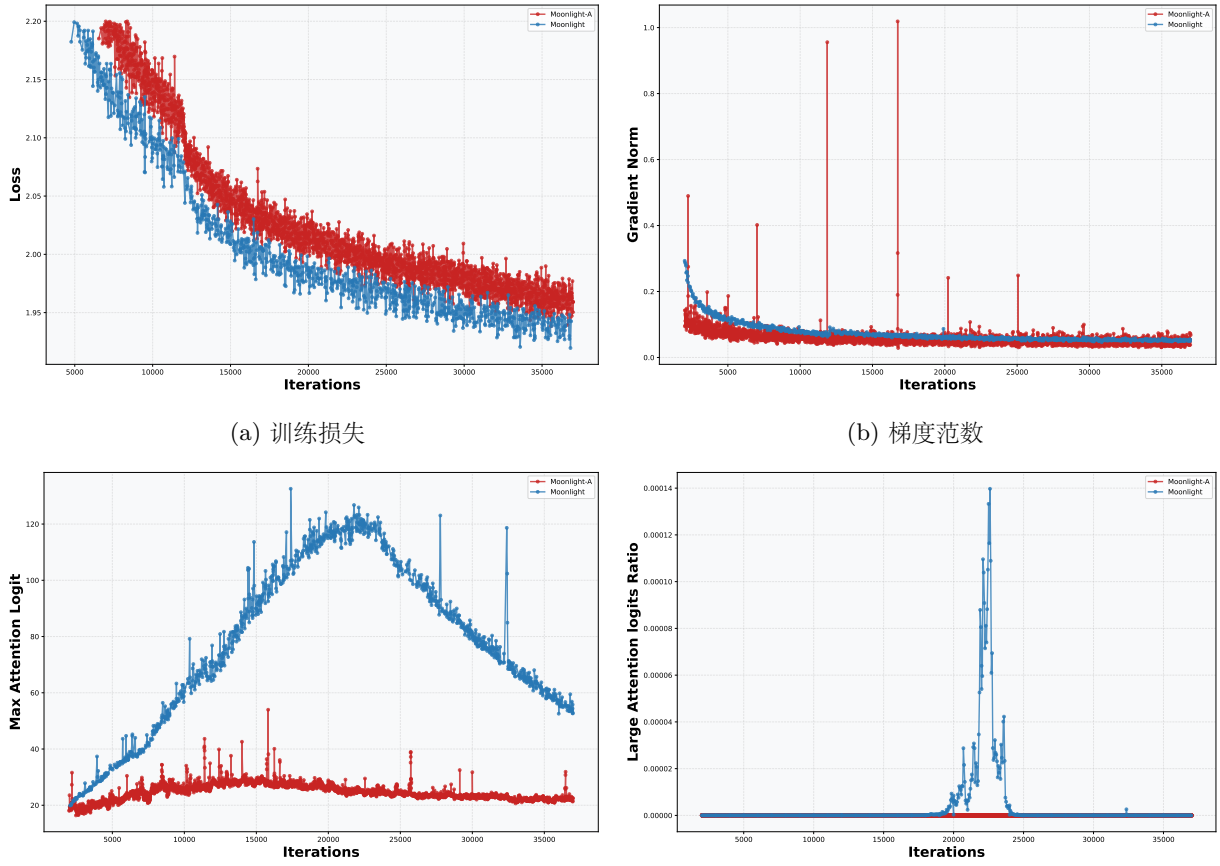
图 6: 计算门控缩放因子的 Python 实现。

## D 训练稳定性

**无损失或梯度范数尖峰** Moonlight 训练过程非常平稳，我们没有遇到任何损失尖峰或梯度范数尖峰。损失和梯度范数曲线见图 7（Moonlight 以蓝色显示，使用 AdamW 训练的 Moonlight-A 以红色显示）

**最大注意力 Logit** 在训练期间，我们观察到虽然训练损失和梯度范数在整个过程中保持稳定，但最大注意力 logit（计算为全局批次中单个最大 logit 值）在初始训练阶段在特定层表现出明显的上升趋势，超过 100 的阈值。值得注意的是，AdamW 在控制这一指标方面比其他优化器表现出更健康的行为。

为了进一步调查这一现象的影响，我们引入了大注意力 logits 比率指标，定义为批次中超过 100 的注意力 logits 比例。如图 7所示，该比率始终保持较低（约 $10^{-4}$），表明极端大的 logit 值是稀疏的。此外，随着训练的进行，最大 logit 值逐渐降低，表明优化动态变得更加健康。




(a) 训练损失 (b) 梯度范数

(c) 最大注意力 Logit（第 1 层） (d) 大注意力 Logit 比率（第 1 层）

图 7: Moonlight 与 Moonlight-A 的训练动态对比

**RMSNorm Gamma 权重衰减** 值得注意的是，对 RMSNorm gamma 参数应用权重衰减对于确保训练稳定性至关重要，因为它有效防止每层输出 RMS 值过高。

## E 与更高计算成本模型的比较

表 10展示了我们使用 Muon 优化的 Moonlight 模型与使用更大计算资源训练的公开可用模型之间的比较分析，包括 `LLama3.1-8B` (Grattafiori et al. 2024)、`Gemma-9B` (Gemma Team et al. 2024) 和 `Qwen2.5-7B` (Yang et al. 2024)。图 8展示了 Moonlight 与领域中可比模型的 GSM8k 性能基准。

## F 权重矩阵的奇异值分布

我们通过绘制每个矩阵奇异值按降序排列的线图来可视化权重矩阵的奇异值分布，并按最大值归一化。如图 9和 10所示，我们发现，对于大多数权重矩阵，Muon 优化的奇异值分布比 AdamW 更平坦，这进一步证实了 Muon 能够提供更多样化更新谱的假设。

表 10: 不同模型在各种基准上的比较。

| | 基准（指标） | Moonlight | LLAMA3.1-8B | Gemma2-9B | Qwen2.5-7B |
|---|---|---|---|---|---|
| | | | 更高训练计算成本的模型 | | |
| | 激活参数 † | 2.24B | 7.38B | 8.32B | 6.83B |
| | 总参数 † | 15.29B | 7.38B | 8.32B | 6.83B |
| | 训练 Token | 5.7T | 15T | 8T | 18T |
| | 优化器 | Muon | AdamW | 未知 | 未知 |
| 英语 | MMLU | 70.0 | 66.7 | 71.3 | 74.2 |
| | MMLU-pro | 42.4 | 37.1 | 44.7 | 45.0 |
| | BBH | 65.2 | 57.7 | 68.2 | 70.4 |
| | TriviaQA‡ | 66.3 | 70.3 | - | 60.0 |
| 代码 | HumanEval | 48.1 | 37.2 | 37.8 | 57.9 |
| | MBPP | 63.8 | 47.6 | 62.2 | 74.9 |
| 数学 | GSM8K | 77.4 | 57.2 | 70.7 | 85.4 |
| | MATH | 45.3 | 20.3 | 37.7 | 49.8 |

† 报告的参数计数不包含嵌入层参数。‡ 我们使用完整的 TriviaQA 测试集测试所有列出的模型。

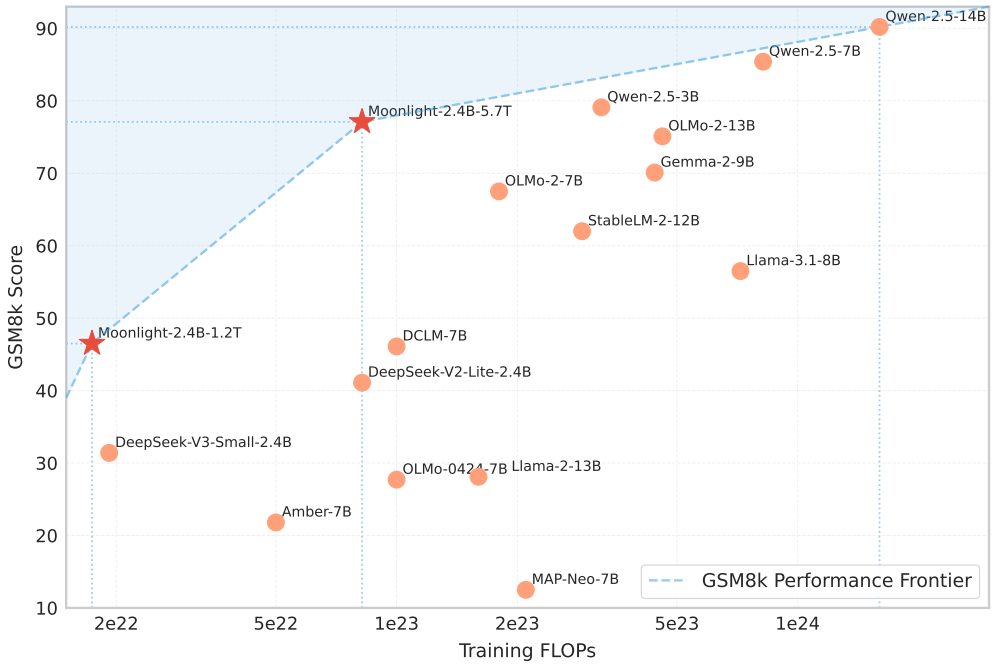



图 8: 我们使用 Muon 优化的 Moonlight 模型与其他可比模型的 GSM8k 性能。

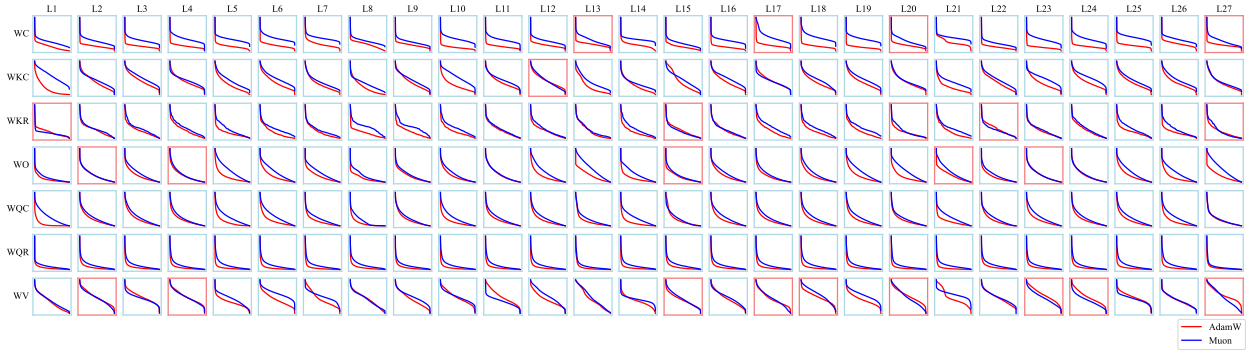



图 9: 注意力层中每个权重矩阵的奇异值分布。我们使用 WC 表示每层中将隐藏状态压缩到键和值共享潜在空间的权重矩阵，WV 表示从潜在空间向上投影值的权重矩阵，WO 表示输出投影矩阵，WKR、WKC、WQR 和 WQC 分别表示带和不带 RoPE 的键和查询部分的投影矩阵。如果 Muon 优化的相应权重矩阵的奇异熵低于 AdamW，我们将每个线图的轴脊设为红色。

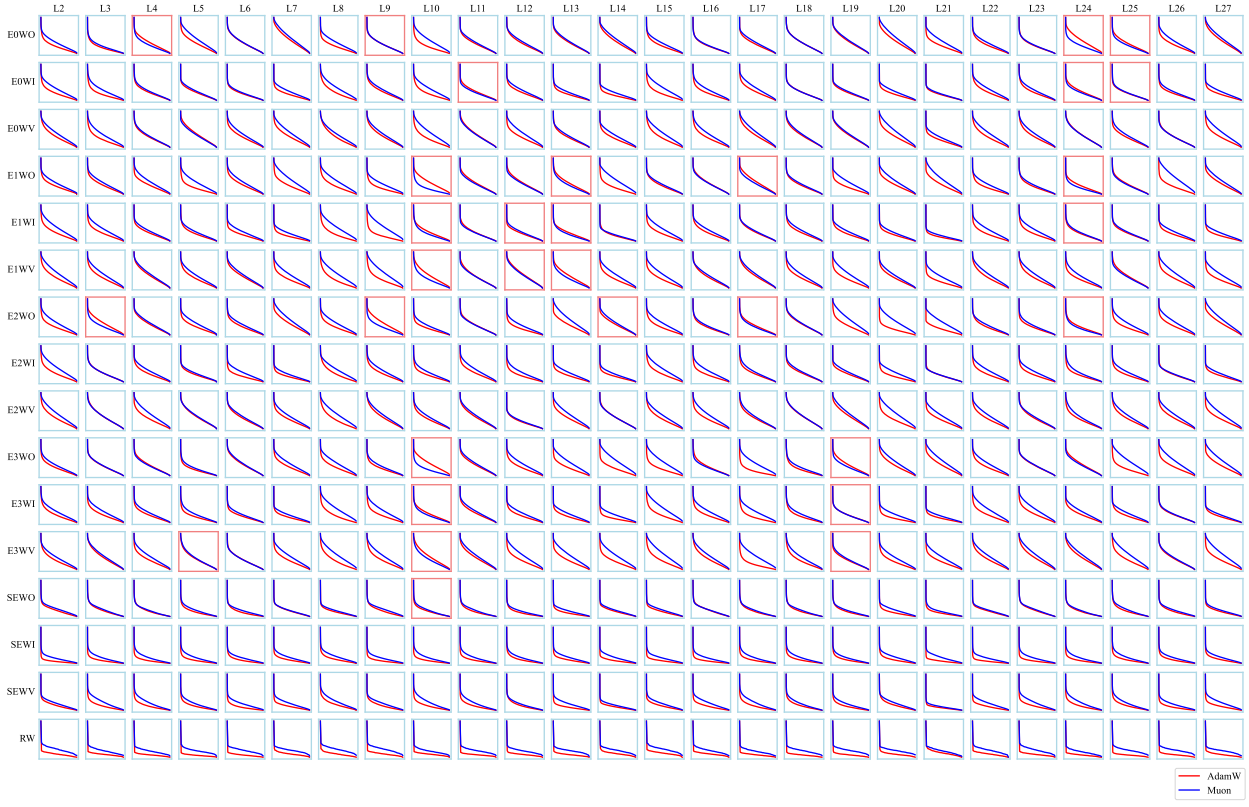



图 10: 前馈网络（FFN）层中每个权重矩阵的奇异值分布。我们使用 WI、WV 和 WO 表示 SwiGLU 激活函数的 FFN 层中涉及的权重矩阵，其中 WI 表示到 $\text{Swish}_1$ 函数的输入投影，WV 表示与 $\text{Swish}_1$ 激活交互的额外输入投影，WO 表示输出投影。我们使用 E0、E2、E3 表示三个任意选择的专家模型，SE 表示共享专家模型中的权重。我们使用 RW 表示路由器中的权重。如果 Muon 优化的相应权重矩阵的奇异熵低于 AdamW，我们将每个线图的轴脊设为红色。