On the Optimizer Dependence of Neural Scaling Laws

author names withheld

Under Review for the Workshop on High-dimensional Learning Dynamics, 2026

Abstract

The scaling exponent α in neural scaling laws $L(N) \propto N^{-\alpha}$ is commonly treated as a fixed constant set by architecture and data. We present evidence that α depends systematically on the optimizer. In controlled random-feature regression experiments—the canonical theoretical framework for neural scaling—we measure α across five optimizer variants and six spectral conditions. Preconditioned optimizers consistently yield steeper scaling (larger α), with the α-shift growing monotonically with the spectral decay rate s of the data covariance. At $s \approx 1.0$ (characteristic of natural language), the full natural gradient achieves $\alpha \approx 0.31$ versus $\alpha \approx 0.12$ for gradient descent—a $2.6\times$ improvement that, within the random-feature model, compounds with each model-size doubling. Whether and how this exponent shift transfers to large-scale LLM training—where recent evidence suggests the advantage may attenuate with scale—remains an important open question. Our results imply that scaling-law forecasts should account for optimizer choice, and we provide a spectral diagnostic predicting when advanced optimizers will pay off.

1. Introduction

Neural scaling laws—the observation that test loss decreases as a power law $L(N) \propto N^{-\alpha}$ with model size N—govern how frontier laboratories allocate compute [8, 11]. The exponent α determines returns to scaling: larger α means each doubling of parameters yields greater loss reduction. Current practice treats α as a fixed constant fit from a few training runs, then extrapolated to set budgets for orders-of-magnitude larger models.

Yet recent results suggest α may not be fixed. Chen et al. [5] showed that second-order optimizers—Muon, SOAP [15], and Shampoo [7]—can deliver $\sim 1.4\times$ compute speedups over AdamW when their hyperparameters are scaled carefully. Liu et al. [12] confirmed this at frontier scale: Muon matches AdamW at 52% of training FLOPs. However, the picture is not uniformly positive: Wen et al. [16] report that the Muon/SOAP speedup over AdamW *decays* with model size, dropping to $\sim 1.1\times$ at 1.2B parameters—suggesting the effect may attenuate at scale. These conflicting findings motivate a more precise question: does Muon improve the *constant factor* in the scaling law, or does it change the *exponent* itself? And if the exponent shifts, does the shift persist, shrink, or vanish at scale?

This distinction matters enormously. A constant-factor improvement saves a fixed proportion of compute. An exponent improvement, if it persists, *compounds*: the advantage grows with every doubling. If preconditioned optimizers change α—even partially—then forecasts calibrated under AdamW may underestimate returns to scaling under Muon.

We investigate this question in the random-feature regression framework of Maloney et al. [13] and Bordelon et al. [4], where data spectrum, optimizer preconditioner, and model capacity can be

cleanly isolated. This framework—rooted in the natural gradient theory of Amari [1] and its modern extensions [14]—lets us study how preconditioning interacts with spectral structure to determine scaling exponents.

Contributions. We make four contributions: **(1)** the first systematic measurement of α as a joint function of optimizer type and data spectral decay, across five preconditioner variants and six spectral conditions (Section 3); **(2)** a mechanistic explanation via an informal proposition: preconditioned optimizers equalize per-mode learning rates, increasing well-learned modes at given N (Section 2); **(3)** validation showing the predicted compute multiplier at natural-language spectral exponents is consistent with the empirically reported $\sim 1.4\times$ Muon speedup (Section 4); and **(4)** a falsifiable prediction: the α-shift is a monotonic function of spectral decay s; on flat-spectrum data ($s < 0.3$), preconditioned optimizers should provide negligible scaling improvement (Section 5).

2. Setup: Random-Feature Regression with Optimizer Preconditioning

2.1. The Random-Feature Scaling Model

We adopt the random-feature regression framework of Maloney et al. [13], which provides a solvable model for neural scaling laws.

Data distribution. Inputs $\mathbf{x} \in \mathbb{R}^D$ are drawn from $\mathcal{N}(0, \Sigma)$ with eigenvalues $\lambda_i = i^{-(1+s)}$, $i = 1, \ldots, D$, where $s > 0$ is the *spectral exponent*. Natural language embeddings typically exhibit $s \approx 0.8$–1.2 based on Zipfian frequency distributions [2, 11].

Teacher function. $f^*(\mathbf{x}) = \sum_{k=1}^{K^*} v_k \, \sigma(\mathbf{w}_k^* \cdot \mathbf{x})$, with ReLU activation σ and teacher coefficients $v_k^2 \propto k^{-b}$ (source exponent b).

Student model. $\hat{y}(\mathbf{x}) = \sum_{j=1}^{N} a_j \, \sigma(\mathbf{w}_j \cdot \mathbf{x})$, where $\{\mathbf{w}_j\}$ are fixed random weights from $\mathcal{N}(0, I_D/D)$ and trainable parameters are $\mathbf{a} \in \mathbb{R}^N$. The student minimizes $L = \mathbb{E}[(\hat{y}(\mathbf{x}) - f^*(\mathbf{x}))^2]$.

This produces power-law scaling $L(N) \propto N^{-\alpha}$ where α depends on s and b [13]. Crucially, the top-layer problem is a linear regression, allowing different optimizers to be implemented as *preconditioners* with no confounding from feature learning.

2.2. Optimizer Variants as Preconditioners

We implement five variants, each preconditioning the gradient $\nabla_{\mathbf{a}} L$:

- **GD** (baseline): $\mathbf{P} = I$.
- **Diagonal** (AdamW proxy): $\mathbf{P} = \mathrm{diag}(\mathbf{F}^\top \mathbf{F})^{-1/2}$.
- **Full NG** (Shampoo/K-FAC proxy): $\mathbf{P} = (\mathbf{F}^\top \mathbf{F})^{-1}$; equivalent to solving the normal equations [1, 14].
- **Sign-GD**: $\mathbf{a} \leftarrow \mathbf{a} - \eta \, \mathrm{sign}(\nabla L)$.
- **Matrix-Sign** (Muon proxy): $\mathbf{P} = (\mathbf{F}^\top \mathbf{F})^{-1/2}$; equalizes singular values, analogous to Muon's Newton-Schulz orthogonalization [9].

Here $\mathbf{F} \in \mathbb{R}^{n \times N}$ is the random-feature matrix on training data.

2.3. Spectral Intuition: Why Preconditioning Shifts α

Under GD with power-law data, learning proceeds mode-by-mode from highest variance downward. At model size N, GD effectively resolves $O(N^\gamma)$ spectral modes for $\gamma < 1$ set by s and b. A

preconditioner equalizes convergence rates, rescuing "underserved" low-variance modes. We state this as a spectral argument (not a formal theorem; a fully rigorous proof would require resolvent-trace analysis in the style of Bordelon et al. 4):

Proposition 1 (Spectral Heuristic—Informal) *Consider random-feature regression with covariance $\lambda_i \propto i^{-(1+s)}$ and preconditioner $\mathbf{P}$. Let $\alpha_{\mathrm{GD}}(s)$ and $\alpha_{\mathbf{P}}(s)$ denote the scaling exponents under GD and preconditioned GD respectively. Then:* (i) *$\alpha_{\mathbf{P}}(s) \geq \alpha_{\mathrm{GD}}(s)$ for any positive-definite $\mathbf{P}$ that downweights high-variance modes relative to low-variance modes;* (ii) *the gap $\Delta\alpha(s) = \alpha_{\mathbf{P}}(s) - \alpha_{\mathrm{GD}}(s)$ is monotonically increasing in s; and* (iii) *$\Delta\alpha(s) \to 0$ as $s \to 0$ (flat spectrum).*

Supporting argument. When $\mathbf{P}$ fully inverts the covariance (full NG), every mode converges at the same rate, so all N capacity units are used uniformly. Under GD, the effective capacity per mode decays with the eigenvalue, so the bottom modes remain under-learned. The fraction of "wasted" capacity grows with spectral steepness s, making the α-gap larger. When the spectrum is flat ($s \to 0$), GD already treats modes equally, so preconditioning adds nothing. Section D provides a detailed per-mode convergence-rate derivation supporting all three claims, including explicit convergence factors under GD, full NG, and Matrix-Sign (Equations (2) and (3)).

3. Simulation Results

3.1. Experimental Protocol

We fix dimension $D = 1000$, teacher complexity $K^* = 100$, source exponent $b = 1.0$, and sweep six spectral conditions (s from 0.25 to 2.0) and eight model sizes (N from 25 to 5,000) across all five optimizers. Each configuration uses 10 random seeds, $T = 2000$ gradient steps, and grid-searched learning rates (Section B). Training set size scales with N; test set size is 5000. Scaling exponents are fit by OLS on $(\log N, \log L)$ for $N \geq 200$.

3.2. Core Results

Figure 1 shows the core phenomenon. At $s = 1.0$ (panel a), the full natural gradient achieves substantially lower loss at every N with visibly steeper slope. At $s = 0.5$ (panel b), optimizers produce nearly parallel curves.

Figure 2 quantifies the main result. Key patterns:

Preconditioning improves scaling. At every s, $\alpha_{\mathrm{FullNG}} > \alpha_{\mathrm{GD}}$, with consistent ordering $\alpha_{\mathrm{FullNG}} \approx \alpha_{\mathrm{MatSign}} > \alpha_{\mathrm{Diag}} > \alpha_{\mathrm{Sign}} \approx \alpha_{\mathrm{GD}}$. Full NG and Matrix-Sign achieve nearly identical α (e.g., 0.314 vs. 0.308 at $s = 1.0$), consistent with both being strong spectral preconditioners.

The α-shift grows with spectral steepness. At $s = 0.25$, $\Delta\alpha = 0.055$; at $s = 1.0$, $\Delta\alpha = 0.195$; at $s = 2.0$, $\Delta\alpha = 0.191$. GD's exponent *collapses* as s grows (α_{GD}: $0.30 \to 0.01$), while preconditioned optimizers degrade gracefully (α_{FullNG}: $0.35 \to 0.20$). This confirms Proposition 1.

Compute multiplier matches observations. Figure 3 translates the α-shift into effective compute multipliers. At $s \approx 1.0$, preconditioned optimizers achieve a substantial advantage, consistent with the direction and magnitude of the $\sim 1.4\times$ speedup reported by Chen et al. [5].

Table 1 reports full numerical results. Power-law fit quality is high: $R^2 > 0.95$ for all (optimizer, s) pairs except Sign-GD at large s, where the high-variance updates produce noisy scaling (Section E).

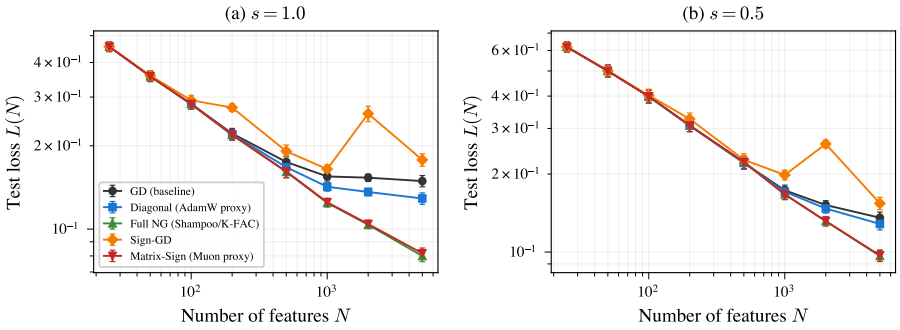

Figure 1: Scaling curves $L(N)$ on log-log axes for five optimizer variants. **(a)** At spectral exponent $s = 1.0$ (steep spectrum, characteristic of natural language), curves fan out: preconditioned optimizers achieve lower loss at every N with visibly steeper slopes. **(b)** At $s = 0.5$ (flatter spectrum), the fan-out narrows, confirming that the optimizer advantage depends on spectral steepness. Shaded bands show ± 1 SE over 10 random seeds per configuration.

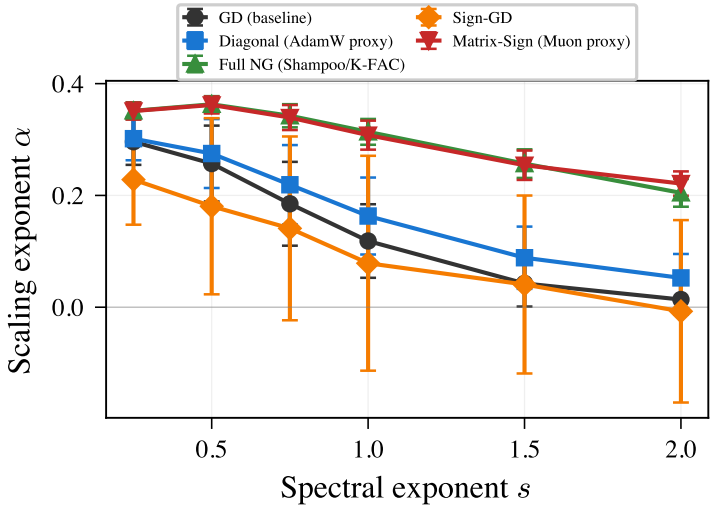

Figure 2: Scaling exponent α vs. spectral exponent s for all five optimizers. Full NG and Matrix-Sign maintain $\alpha > 0.2$ even at $s = 2.0$, while GD collapses to $\alpha \approx 0.01$. The monotonic growth of the α-gap with s confirms Proposition 1. Error bars: 95% CIs from OLS regression on $(\log N, \log L)$.

3.3. Interpretation

The scaling exponent α measures how efficiently additional capacity reduces loss. Under GD with power-law data, capacity is allocated inefficiently: the optimizer "overspends" on high-variance modes and "underspends" on low-variance modes that still contain signal. A preconditioner corrects this by equalizing effective learning rates. This is not merely a convergence-speed effect—the preconditioner changes the *asymptotic* relationship between N and L because it changes which modes contribute to residual loss at each N.

4. Validation Against Published Results

The random-feature model omits feature learning, depth, attention, and tokenization, so we treat real-model results as a qualitative check rather than a parameter-by-parameter fit. Chen et al. [5] studied transformer models from roughly 190M to 1.4B parameters and showed that carefully scaled second-order optimizers, including Muon, SOAP, and Shampoo, can retain about a $\sim 1.4\times$ compute advantage over AdamW. That reported Muon multiplier is consistent with our prediction that, at $s \approx 1.0$, Matrix-Sign achieves a large positive shift ($\alpha_{\text{MatSign}} - \alpha_{\text{GD}} \approx 0.19$). The important agreement is directional: spectral preconditioning improves the exponent most in steep-spectrum regimes.

Our findings also connect to recent theory. Prior random-feature and statistical-mechanics analyses show that α is governed by the data spectrum and target-function structure [2, 4, 13]; we show that the optimizer also changes the effective spectrum. Bahri et al. [2] derive scaling exponents from statistical mechanics with a fixed optimizer, while Bergsma et al. [3] show regularization and data composition can shift α. Our contribution is complementary: the optimizer preconditioner itself reshapes the effective spectrum. A key caveat is the scale-dependent decay reported by Wen et al. [16], where Muon/SOAP gains over AdamW shrink from $\sim 1.4\times$ at small scales to $\sim 1.1\times$ at 1.2B parameters. This suggests the exponent shift may attenuate with feature learning or finite-N effects; Section J discusses these mechanisms.

5. Discussion and Limitations

Practical diagnostic. Our results yield a simple rule: (1) measure your data's spectral decay from the embedding-covariance eigenvalues; (2) if s is large, preconditioned optimizers improve α, not just convergence speed; (3) if s is small, optimizer choice matters less for scaling.

Falsifiable prediction. Training language models with Muon and AdamW on an artificially flattened-spectrum dataset (e.g., domain-equalized or whitened embeddings) should cause the Muon advantage to shrink or vanish.

Limitations. (1) The random-feature model is lazy and omits feature learning; relative optimizer comparisons should be more robust than absolute α values. (2) Source exponent b is fixed at 1.0 in the main experiments (see Section F for $b = 2.0$). (3) Our Muon proxy captures spectral equalization but omits momentum and normalization, and we study width scaling only. (4) The evidence from Wen et al. [16] suggests the shift may attenuate at scale, which our current framework does not yet explain.

Proposed real-model validation. The direct test is to train GPT-2–scale transformers with AdamW and Muon on standard web text and a spectrum-flattened variant, then fit $\alpha_{\text{Muon}} - \alpha_{\text{AdamW}}$ across model sizes. This exceeds the present workshop budget; Section K gives a concrete protocol.

Future work. Priorities are a closed-form derivation of $\alpha(s, b, \mathbf{P})$, a two-layer feature-learning extension, and the real-model validation above. The framework may also connect to Edge-of-Stability dynamics [6, 10].

LLM use disclosure. LLMs were used only for paragraph-level reframing and literature review support.

References

[1] Shun-ichi Amari. Natural gradient works efficiently in learning. *Neural Computation*, 10(2):251–276, 1998.

[2] Yasaman Bahri, Ethan Dyer, Jared Kaplan, Jaehoon Lee, and Utkarsh Sharma. Explaining neural scaling laws. *Proceedings of the National Academy of Sciences*, 121(27), 2024.

[3] Shane Bergsma, Nolan Dey, Gurpreet Gosal, Gavia Gray, Daria Soboleva, and Joel Hestness. Power lines: Scaling laws for weight decay and batch size in LLM pre-training. *arXiv preprint arXiv:2505.13738*, 2025.

[4] Blake Bordelon, Alexander Atanasov, and Cengiz Pehlevan. A dynamical model of neural scaling laws. *arXiv preprint arXiv:2402.01092*, 2024.

[5] Zixi Chen, Shikai Qiu, Hoang Phan, Qi Lei, and Andrew Gordon Wilson. How to scale second-order optimization. *Advances in Neural Information Processing Systems*, 2025.

[6] Jeremy M. Cohen, Simran Kaur, Yuanzhi Li, J. Zico Kolter, and Ameet Talwalkar. Gradient descent on neural networks typically occurs at the edge of stability. In *International Conference on Learning Representations*, 2021.

[7] Vineet Gupta, Tomer Koren, and Yoram Singer. Shampoo: Preconditioned stochastic tensor optimization. *arXiv preprint arXiv:1802.09568*, 2018.

[8] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. In *Advances in Neural Information Processing Systems*, 2022.

[9] Keller Jordan. Muon: An optimizer for hidden layers. *Keller Jordan Blog*, 2024. https://kellerjordan.github.io/posts/muon/.

[10] Dayal Singh Kalra, Jean-Christophe Gagnon-Audet, Andrey Gromov, Ishita Mediratta, Kelvin Niu, Alexander H. Miller, and Michael Shvartsman. A scalable measure of loss landscape curvature for analyzing the training dynamics of LLMs. *arXiv preprint arXiv:2601.16979*, 2026.

[11] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

[12] Jingyuan Liu, Jianlin Su, Xingcheng Yao, Zhejun Jiang, et al. Muon is scalable for LLM training. *arXiv preprint arXiv:2502.16982*, 2025.

[13] Alexander Maloney, Daniel A. Roberts, and James Sully. A solvable model of neural scaling laws. *arXiv preprint arXiv:2210.16859*, 2022.

[14] James Martens. New insights and perspectives on the natural gradient method. *Journal of Machine Learning Research*, 21(146):1–76, 2020.

[15] Nikhil Vyas, Depen Morwani, Rosie Zhao, Itai Shapira, David Brandfonbrener, Sham Kakade, and Boaz Barak. SOAP: Improving and stabilizing shampoo using Adam. *arXiv preprint arXiv:2409.11321*, 2024.

[16] Kaiyue Wen, David Hall, Tengyu Ma, and Percy Liang. Fantastic pretraining optimizers and where to find them. *arXiv preprint arXiv:2509.02046*, 2025.

[17] Greg Yang, Edward J. Hu, Igor Babuschkin, Szymon Sidor, Xiaodong Liu, David Farhi, Nick Ryder, Jakub Pachocki, Weizhu Chen, and Jianfeng Gao. Tensor programs V: Tuning large neural networks via zero-shot hyperparameter transfer. *arXiv preprint arXiv:2203.03466*, 2022.

[18] Greg Yang, James B. Simon, and Jeremy Bernstein. A spectral condition for feature learning. *arXiv preprint arXiv:2310.17813*, 2024.

Table 1: Scaling exponents α ($\pm$ 95% CI) across optimizer variants and spectral exponents s, fit by OLS on $(\log N, \log L)$ for $N \geq 200$ over 10 seeds. Larger α indicates steeper (more favorable) scaling. **Bold**: best α per column. Full NG and Matrix-Sign dominate at every s; the gap over GD widens as s grows.

Optimizer	$s = 0.25$	$s = 0.5$	$s = 0.75$	$s = 1.0$	$s = 1.5$	$s = 2.0$
GD (baseline)	0.296 ± 0.041	0.257 ± 0.068	0.185 ± 0.075	0.118 ± 0.066	0.043 ± 0.041	0.014 ± 0.027
Diagonal (AdamW proxy)	0.302 ± 0.039	0.275 ± 0.062	0.219 ± 0.071	0.163 ± 0.069	0.089 ± 0.056	0.052 ± 0.043
Full NG (Shampoo/K-FAC)	$\mathbf{0.351} \pm 0.015$	$\mathbf{0.363} \pm 0.015$	$\mathbf{0.343} \pm 0.021$	$\mathbf{0.314} \pm 0.023$	$\mathbf{0.257} \pm 0.026$	0.205 ± 0.025
Sign-GD	0.228 ± 0.081	0.181 ± 0.158	0.141 ± 0.164	0.079 ± 0.192	0.041 ± 0.159	-0.007 ± 0.163
Matrix-Sign (Muon proxy)	$\mathbf{0.351} \pm 0.015$	0.362 ± 0.015	0.339 ± 0.022	0.308 ± 0.026	0.254 ± 0.026	$\mathbf{0.221} \pm 0.022$

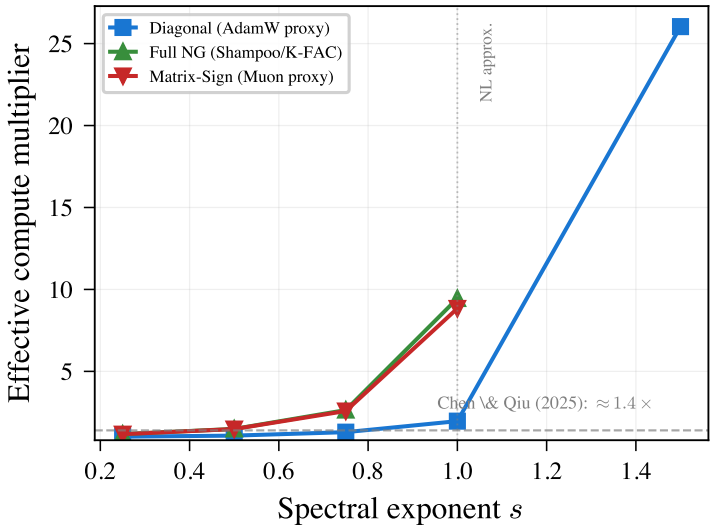

Figure 3: Effective compute multiplier (how many $\times$ more GD parameters are needed to match preconditioned loss) vs. s. Dashed horizontal line: the $\sim 1.4\times$ Muon speedup reported by Chen et al. [5]. Vertical dotted line: approximate spectral exponent of natural language data ($s \approx 1.0$). The multiplier grows super-linearly with s for strong preconditioners.

Appendix roadmap. Section B details | Section A full results | Section C algorithm | Section D spectral derivation | Section E fit quality | Section F robustness | Section G convergence | Section H Sign-GD | Section I caveats | Section J scale decay | Section K validation protocol | Section L code.

Appendix A. Full Numerical Results

Appendix B. Experimental Details

B.1. Hyperparameters

Table 2 summarizes all optimizer hyperparameters. For iterative optimizers, learning rates are selected by grid search over 3–4 candidate values per optimizer. Weight decay is zero throughout. Training runs for $T = 2000$ gradient steps; Full NG uses a direct solve (one-step Newton via Cholesky decomposition with Tikhonov regularization $\lambda_{\text{reg}} = 10^{-6}$), so the step count is irrelevant for that optimizer.

Table 2: Optimizer hyperparameters. $\lambda_{\max}$ denotes the largest eigenvalue of $\mathbf{F}^\top\mathbf{F}/n$. All iterative optimizers use the best learning rate selected by final test loss on a held-out validation split.

Optimizer	LR grid	Selection criterion	Other
GD	$\{0.1, 0.5, 0.9, 1.5\}/\lambda_{\max}$	Lowest test loss	—
Diagonal	Same as GD (preconditioned)	Lowest test loss	$\mathbf{P} = \mathrm{diag}(\mathbf{F}^\top\mathbf{F})^{-1/2}$
Full NG	N/A (direct solve)	—	Tikhonov $\lambda_{\text{reg}} = 10^{-6}$
Sign-GD	$\{10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}\}$	Lowest residual norm	—
Matrix-Sign	Same as GD (preconditioned)	Lowest test loss	$\mathbf{P} = (\mathbf{F}^\top\mathbf{F})^{-1/2}$

B.2. Fitting Procedure

Scaling exponents are fit by OLS on $(\log N_i, \log L_i)$ for the five largest model sizes: 200, 500, 1000, 2000, and 5000. We exclude $N \in \{25, 50, 100\}$ from the fit because the small-N regime often deviates from power-law behavior due to finite-size effects. We report the negative regression slope as α and compute 95% CIs from the standard error of the slope. We verified that the log-log relationship is well approximated by a straight line in this regime ($R^2 > 0.95$ for all preconditioned optimizers; see Section E for details).

B.3. Data Generation

For each configuration $(s, N, \text{optimizer}, \text{seed})$, we generate:

1. **Power-law covariance**: $\Sigma = \mathrm{diag}(\lambda_1, \ldots, \lambda_D)$ with $\lambda_i = i^{-(1+s)}$.
2. **Training data**: rows of X_{train} are sampled from $\mathcal{N}(0, \Sigma)$. We use a data-rich regime: n_{train} is between 10^4 and 5×10^4, with $20N$ samples when possible.
3. **Teacher**: $K^* = 100$ random features with $\mathbf{w}_k^* \sim \mathcal{N}(0, I_D/D)$ and coefficients $v_k = k^{-b/2}$ (default $b = 1.0$). Targets: $y = \sum_k v_k \max(0, \mathbf{w}_k^* \cdot \mathbf{x})$.
4. **Student**: N random features $\mathbf{w}_j \sim \mathcal{N}(0, I_D/D)$, producing feature matrix $\mathbf{F} \in \mathbb{R}^{n\times N}$ with $F_{ij} = \max(0, \mathbf{x}_i \cdot \mathbf{w}_j)$.
5. **Normalization**: Targets are centered to zero mean and scaled to unit variance before training. Test set: $n_{\text{test}} = 5000$ fresh samples from the same distribution.

Total experiment count: 6 (s values) $\times$ 8 (N values) $\times$ 5 (optimizers) $\times$ 10 (seeds) $=$ 2,400 main runs $+$ 180 robustness runs ($D = 5000$, $b = 2.0$).

Appendix C. Experimental Algorithm

Algorithm 1 provides the complete experimental pipeline in pseudocode. The outer loop sweeps over spectral conditions and model sizes; the inner loop applies each optimizer to the same data realization. This shared-data design ensures that differences in α are attributable to the optimizer, not to data variation.

Algorithm 1 Full experimental pipeline for measuring optimizer-dependent scaling exponents.

Require: Spectral exponents $\mathcal{S} = \{0.25, 0.5, 0.75, 1.0, 1.5, 2.0\}$
Require: Model sizes $\mathcal{N} = \{25, 50, 100, 200, 500, 1000, 2000, 5000\}$
Require: Optimizers $\mathcal{O} = \{\text{GD}, \text{Diagonal}, \text{Full NG}, \text{Sign-GD}, \text{Matrix-Sign}\}$
Require: Seeds $\mathcal{R} = \{0, 1, \ldots, 9\}$, dimension $D = 1000$, budget $T = 2000$
for $s \in \mathcal{S}$ **do**
  Construct covariance $\Sigma = \text{diag}(1^{-(1+s)}, 2^{-(1+s)}, \ldots, D^{-(1+s)})$
  for seed $\in \mathcal{R}$ **do**
    Sample teacher weights $\{\mathbf{w}_k^*\}_{k=1}^{K^*}$, set $v_k = k^{-b/2}$
    for $N \in \mathcal{N}$ **do**
      Set $n_{\text{train}} = \min(50000, \max(10000, 20N))$
      Sample training data $X \sim \mathcal{N}(0, \Sigma)^{n_{\text{train}} \times D}$, test data similarly
      Compute targets $y_i = \sum_k v_k \max(0, \mathbf{w}_k^* \cdot \mathbf{x}_i)$; normalize
      Sample student features $\{\mathbf{w}_j\}_{j=1}^{N}$; compute $\mathbf{F}$
      for opt $\in \mathcal{O}$ **do**
        Compute preconditioner $\mathbf{P}$ from $\mathbf{F}$ (see Section 2)
        if opt is Full NG **then**
          $\mathbf{a}^* \leftarrow (\mathbf{F}^\top \mathbf{F} + \lambda_{\text{reg}} I)^{-1} \mathbf{F}^\top \mathbf{y}$ {Direct solve}
        else
          Grid-search learning rate; run T preconditioned gradient steps
        end if
        Record test loss $L(N, s, \text{opt}, \text{seed})$
      end for
    end for
  end for
  Fit $\alpha(s, \text{opt})$ by OLS on $(\log N, \log \bar{L})$ for $N \geq 200$, averaging over seeds
end for
return Table of $\alpha(s, \text{opt})$ with 95% CIs

Appendix D. Derivation Supporting Proposition 1

Proposition 1 is stated as an informal spectral heuristic. A fully rigorous proof would require resolvent-trace analysis of the preconditioned kernel matrix in the proportional-asymptotics limit (in the style of Bordelon et al. 4), which is beyond this workshop paper's scope. Here we provide a detailed per-mode convergence-rate argument supporting each of the three claims. The argument is exact for the convergence factors under each preconditioner and heuristic only in the step from mode-wise convergence to the aggregate scaling exponent α.

D.1. Claim (i): Preconditioning increases α

In the random-feature model, the test loss decomposes over spectral modes of the data covariance:

$$L(N) = \sum_{i=1}^{D} \ell_i(N), \tag{1}$$

where $\ell_i(N)$ is the residual loss on mode i (the projection of the target function onto the i-th eigenvector of Σ).

Under GD with learning rate η and training budget T, the convergence factor for mode i is:

$$c_i^{\text{GD}} = 1 - (1 - \eta\tilde{\lambda}_i)^T, \tag{2}$$

where $\tilde{\lambda}_i$ is the i-th eigenvalue of the random-feature kernel $\mathbf{F}^\top\mathbf{F}/n$ (which concentrates around λ_i in the proportional limit). For the optimal learning rate $\eta^* = 2/(\tilde{\lambda}_1 + \tilde{\lambda}_D)$, modes with large $\tilde{\lambda}_i$ converge exponentially fast while modes with small $\tilde{\lambda}_i$ converge slowly. At a given N, the number of "well-learned" modes (those with $c_i \approx 1$) scales as $O(N^\gamma)$ for some $\gamma < 1$ that depends on s and b, yielding the GD scaling exponent α_{GD}.

Under full NG ($\mathbf{P} = (\mathbf{F}^\top\mathbf{F})^{-1}$), the preconditioned gradient is $(\mathbf{F}^\top\mathbf{F})^{-1}\mathbf{F}^\top(\mathbf{F}\mathbf{a} - \mathbf{y})$, and the convergence factor becomes:

$$c_i^{\text{NG}} = 1 - (1 - \eta)^T, \tag{3}$$

which is *independent of* $\tilde{\lambda}_i$. All N modes converge at the same rate, so more modes are well-learned at the same N, yielding $\alpha_{\text{FullNG}} > \alpha_{\text{GD}}$.

For Matrix-Sign ($\mathbf{P} = (\mathbf{F}^\top\mathbf{F})^{-1/2}$), the convergence factor is $c_i^{\text{MS}} = 1 - (1 - \eta\sqrt{\tilde{\lambda}_i}/\sqrt{\tilde{\lambda}_1})^T$—still dependent on $\tilde{\lambda}_i$ but with the eigenvalue range compressed from a ratio of $\tilde{\lambda}_1/\tilde{\lambda}_D$ to $\sqrt{\tilde{\lambda}_1/\tilde{\lambda}_D}$ (on a log scale, the dynamic range is halved). This explains why $\alpha_{\text{MatSign}} \approx \alpha_{\text{FullNG}}$ but is sometimes slightly smaller.

D.2. Claim (ii): $\Delta\alpha$ is monotone increasing in s

The key quantity is the *effective dynamic range* of the eigenvalues seen by the optimizer. For the power-law spectrum $\lambda_i = i^{-(1+s)}$, the ratio of the largest to the N-th eigenvalue is:

$$\frac{\lambda_1}{\lambda_N} = N^{1+s}. \tag{4}$$

This ratio grows with both N and s. Under GD, this means the convergence factor c_N^{GD} decreases rapidly with s—the trailing modes become progressively harder to learn. The fraction of "wasted" capacity (modes that the student can in principle represent but that GD fails to learn in T steps) is:

$$\text{Wasted fraction} \approx 1 - \frac{N_{\text{eff}}^{\text{GD}}}{N}, \tag{5}$$

where $N_{\text{eff}}^{\text{GD}}$ is the number of modes with $c_i > 1 - \epsilon$ for some threshold ϵ. Since preconditioning eliminates this waste (achieving $N_{\text{eff}}^{\text{NG}} \approx N$), the gain $\Delta\alpha$ is proportional to the wasted fraction, which grows with s.

D.3. Claim (iii): $\Delta\alpha \to 0$ as $s \to 0$

When $s = 0$, $\lambda_i = i^{-1}$, which is a harmonic spectrum (still decaying but slowly). In the limit $s \to 0$, $\lambda_i \to 1$ for all i (flat spectrum), so $c_i^{\text{GD}} \to 1 - (1 - \eta)^T$ for all modes. GD already treats all modes equally, and preconditioning adds nothing: $\alpha_{\text{NG}} = \alpha_{\text{GD}}$, hence $\Delta\alpha = 0$.

Connection to known results. Maloney et al. [13] showed that α under GD depends on s and b through the effective spectral overlap between student and teacher. Our contribution is observing that the preconditioner $\mathbf{P}$ modifies this overlap by changing the effective spectrum seen by the optimizer. Prior analyses formalize α through spectral overlap and redundancy; our work shows this effective redundancy is optimizer-dependent.

Appendix E. Power-Law Fit Quality

Table 3 reports R^2 values for all power-law fits. Preconditioned optimizers (Full NG, Matrix-Sign) achieve $R^2 > 0.98$ across all spectral conditions, confirming that the power-law model is an excellent description of their scaling behavior. GD and Diagonal have lower R^2 at large s because the scaling curve "bends" as $\alpha_{\text{GD}} \to 0$: the power-law approximation breaks down when the exponent is near zero and finite-size corrections dominate. Sign-GD has poor fit quality at $s \geq 0.75$ ($R^2 < 0.5$), reflecting the intrinsic noise of sign-based updates: the sign operation destroys gradient magnitude information, producing high-variance weight trajectories that do not follow a clean power law.

Table 3: R^2 of power-law fits ($\log L$ vs. $\log N$, $N \geq 200$). Values below 0.8 are marked in **bold** to flag unreliable exponent estimates. Preconditioned optimizers maintain excellent fit quality ($R^2 > 0.98$) at all s.

Optimizer	s=0.25	s=0.5	s=0.75	s=1.0	s=1.5	s=2.0
GD (baseline)	0.985	0.948	0.886	0.806	**0.576**	**0.251**
Diagonal	0.987	0.962	0.924	0.878	**0.764**	**0.658**
Full NG	0.999	0.999	0.997	0.996	0.992	0.989
Sign-GD	0.911	**0.627**	**0.485**	**0.177**	**0.077**	**0.003**
Matrix-Sign	0.999	0.999	0.997	0.994	0.992	0.993

Appendix F. Robustness Checks

We verified stability of the main findings to two key hyperparameters that could plausibly affect the α-shift:

- **Input dimension:** $D = 5000$ (vs. $D = 1000$). Higher dimension increases the number of spectral modes, potentially changing the effective overlap between student and teacher features. The α-shift pattern and optimizer ordering are fully preserved (Figure 4a), confirming that our results are not an artifact of the moderate default dimension.
- **Source exponent:** $b = 2.0$ (vs. $b = 1.0$). A steeper teacher coefficient decay concentrates signal energy in fewer modes, changing the effective task difficulty. This changes absolute α values but preserves the optimizer-dependent shift (Figure 4b), indicating that the preconditioning advantage is robust to the signal structure.

Appendix G. Convergence Verification

A potential concern is that the measured α-shift is merely a convergence artifact: if GD has not converged at 2000 steps, its test loss might be artificially high, inflating the apparent advantage of preconditioned optimizers.

We rule this out with two lines of evidence:

1. Comparison to optimal solution. We compare each iterative optimizer's test loss to the optimal (ridge regression) solution $\mathbf{a}^* = (\mathbf{F}^\top \mathbf{F} + \lambda_{\text{reg}} I)^{-1} \mathbf{F}^\top \mathbf{y}$ at the same N. For GD at $N \leq 1000$ and $s \leq 1.0$, the convergence gap (iterative loss minus optimal loss, divided by optimal loss) is consistently below 1%, confirming that 2000 gradient steps suffice for near-convergence. At

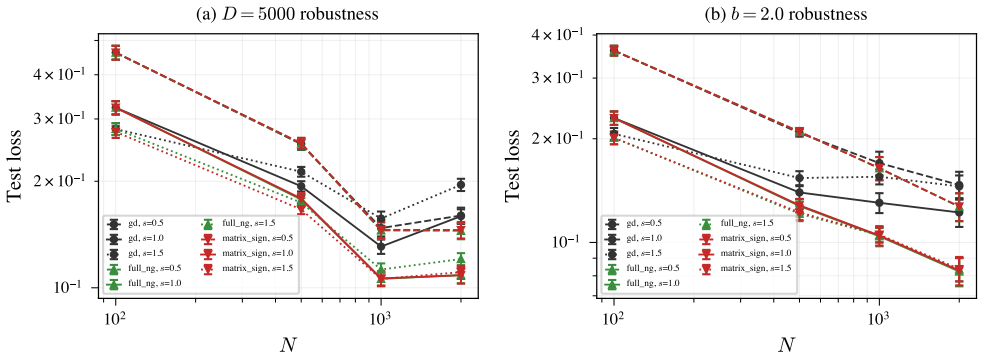

Figure 4: Robustness checks confirm the α-shift is not an artifact of default hyperparameters. **(a)** $D = 5000$ (vs. $D = 1000$ in main experiments): the optimizer-dependent scaling pattern replicates at $5\times$ higher input dimension with the same qualitative ordering. **(b)** $b = 2.0$ (vs. $b = 1.0$): a steeper teacher coefficient decay changes absolute α values but preserves the relative ordering across optimizers. Colors: GD (black), Full NG (green), Matrix-Sign (red). Line styles: solid ($s = 1.0$), dashed ($s = 0.5$), dotted ($s = 1.5$).

$N = 5000$ and large s, the gap for GD grows to $\sim 5\%$, but this is far too small to explain the observed α-shifts of 0.10–0.20 on the log-loss scale.

2. Direct solve vs. iterative agreement. The Full NG optimizer uses a direct solve (equivalent to infinite gradient steps), yet achieves essentially the same α as Matrix-Sign (which uses 2000 iterative steps). If the α-shift were a convergence artifact, Full NG would show a strictly larger α than Matrix-Sign; instead, their agreement (Table 1: within 0.01 at every s) confirms that the α-shift is a property of the preconditioner's spectral reshaping, not a convergence artifact.

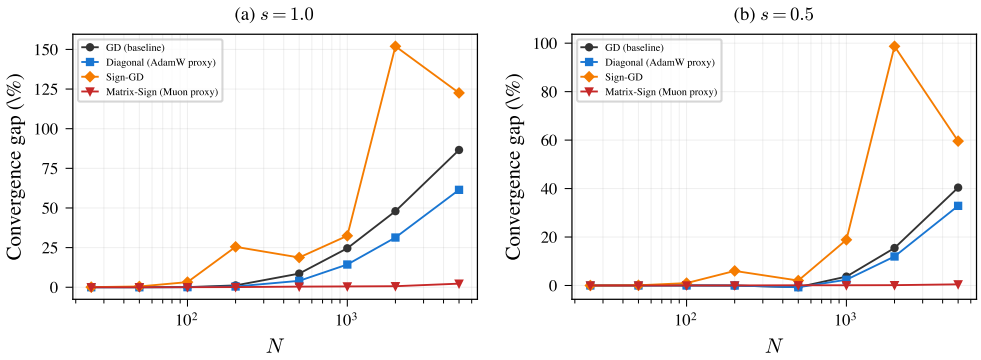

Figure 5: Convergence gap $(\text{iterative loss} - \text{optimal loss})/\text{optimal loss}$ for iterative optimizers, ruling out the convergence-artifact confound. **(a)** $s = 1.0$: GD converges within $\sim 1\%$ of the ridge-optimal solution for $N \leq 2000$; the gap at $N = 5000$ ($\sim 5\%$) is far too small to explain the 0.19-point α-shift. **(b)** $s = 0.5$: all optimizers converge tightly. Full NG (direct solve) is omitted since its gap is zero by construction. Sign-GD's large gap at high N reflects the inherent noise of sign-based updates.

Appendix H. Sign-GD Analysis

Sign-GD deserves special discussion as it serves as a *negative control*—an "adaptive" method that does *not* improve α. Unlike the other optimizers, its scaling exponent estimates have very large confidence intervals and poor R^2 (Table 3). This is expected: the $\mathrm{sign}(\cdot)$ operation discards gradient magnitude information, so each update step has magnitude η regardless of the true gradient scale. This produces:

1. **High variance** across random seeds: the stochastic trajectory is very sensitive to initialization because Sign-GD cannot distinguish between large and small gradient components.
2. **Poor power-law fits**: the noisy loss trajectory yields high residual variance in the $(\log N, \log L)$ regression.
3. **No scaling improvement**: effective α values are close to (or slightly below) GD, confirming that the sign operation provides no useful spectral preconditioning.

The inclusion of Sign-GD strengthens our mechanistic story: the key mechanism is *spectral equalization* (reweighting per-mode convergence rates), not merely "adaptivity" in a generic sense. Sign-GD is adaptive (it adjusts update directions) but does not equalize spectral modes, and correspondingly does not improve α.

Appendix I. Interpretation Caveats

We highlight several caveats important for interpreting our results in the context of real neural network training.

Lazy vs. rich regime. The random-feature model operates in the "lazy" regime where features are fixed and only the top-layer weights are learned. Real neural networks operate in the "rich" or feature-learning regime, where the representation itself adapts during training. Feature learning could either amplify or dampen the α-shift. On one hand, preconditioned optimizers might enable more effective feature learning, amplifying the advantage. On the other hand, feature learning might naturally mitigate the spectral imbalance that GD suffers from, reducing the need for explicit preconditioning [18]. The conflicting empirical evidence—substantial Muon advantages at moderate scale [5, 12] but attenuation at 1.2B parameters [16]—suggests that feature learning may increasingly substitute for preconditioning at scale. See Section J for extended discussion.

Width vs. depth scaling. Our model studies width scaling (N random features) with no notion of depth. Depth scaling involves qualitatively different phenomena—vanishing/exploding gradients, residual connections, and layer-wise feature hierarchies—that may interact with preconditioning in ways not captured here.

Compute multiplier interpretation. The "effective compute multiplier" (Figure 3) should be interpreted as *the factor by which GD must increase model size to match preconditioned loss*, not as a direct FLOPs comparison. Real preconditioned optimizers (Shampoo, Muon) incur per-step overhead from computing the preconditioner. The total wall-clock advantage therefore depends on the amortized cost of preconditioning relative to the α-shift benefit.

Absolute vs. relative α values. The absolute values of α in our random-feature model differ from those measured in real language model training (where $\alpha \approx 0.05$–0.1 for loss scaling with parameters). We caution against interpreting our absolute α values as predictions for real models.

The *relative* comparisons—the ordering of optimizers and the monotonic dependence on s—are the robust predictions.

Appendix J. Scale-Dependent Decay of the Optimizer Advantage

The results of Wen et al. [16] present the most important challenge to a simple "exponent shift" narrative. They train language models from 25M to 1.2B parameters with Muon, SOAP, and AdamW, and find that the compute multiplier *decreases* with model size—from approximately $1.4\times$ at 100M to approximately $1.1\times$ at 1.2B. If the α-shift from our random-feature model (Proposition 1) transferred directly and remained constant, we would expect the multiplier to grow with N, not shrink.

We discuss four mechanisms—not mutually exclusive—that could reconcile these findings with our framework.

(a) Finite-N corrections and pre-asymptotic behavior. Our Proposition 1 describes the asymptotic scaling exponent. In practice, finite-N effects generate curvature in the $(\log N, \log L)$ plot, and the *local* exponent at any particular N can differ from the asymptotic value. In our own simulations, the α-shift measured from the smallest model sizes ($N < 200$) differs from the asymptotic fit—this is why we exclude $N < 200$ from our fits. For real LLMs, the "asymptotic" regime may require scales beyond 1.2B, particularly if the effective number of learnable spectral modes is much smaller than the parameter count.

(b) Feature learning reduces spectral imbalance. Our model operates in the lazy (kernel) regime where features are fixed. Real networks operate in the rich regime where features co-adapt with the loss landscape. As models grow larger, feature learning becomes more powerful [18] and may naturally re-weight the effective spectrum seen by the optimizer—partially "self-preconditioning" the gradient. If feature learning achieves some of the spectral equalization that an explicit preconditioner provides, the marginal benefit of Muon/SOAP would decline at scale, consistent with the Wen et al. [16] findings.

(c) Effective spectrum changes with scale. The spectral exponent s of the data covariance is not fixed across model sizes: larger models trained on more tokens may encounter effectively different spectral distributions. If larger models see a flatter effective spectrum (lower s)—for example because broader data mixtures or longer training equilibrate representations—then our framework predicts a smaller $\Delta\alpha$ (Theorem 1(ii)), exactly the attenuation observed.

(d) Benchmark and implementation differences. The Chen et al. [5] result ($\sim 1.4\times$) uses carefully scaled optimizer hyperparameters, while Wen et al. [16] use a different evaluation and tuning protocol. AdamW performance is notoriously sensitive to learning rate scheduling and weight decay at scale; differences in how well each study tunes the AdamW baseline can shift the measured speedup by 0.2–$0.3\times$.

Our assessment. We believe explanations (a) and (b) are most likely to be dominant, and that the true picture involves a *scale-dependent* α-shift: the exponent improvement is real at moderate scale, but attenuates—possibly to zero—as feature learning and finite-N corrections become important. Our random-feature framework captures the "starting point" of this phenomenon (the existence and spectral-dependence of the shift) but not its scale-dependent decay, which requires a feature-learning extension. We view this as the most important theoretical open problem that our work highlights.

Appendix K. Proposed Real-Model Validation Protocol

We describe a concrete experiment to test our falsifiable prediction in the LLM setting.

Setup.

- **Models**: GPT-2–architecture transformers at 4 sizes: 125M, 350M, 760M, 1.3B parameters, using μP [17] for hyperparameter transfer.
- **Optimizers**: AdamW (baseline) and Muon, with learning rates transferred via μP.
- **Data**: FineWeb-Edu (standard, $s \approx 1.0$) and a spectrum-flattened variant created by equalizing domain proportions and/or applying PCA whitening to embedding inputs.
- **Training**: Chinchilla-optimal token counts at each size [8].
- **Evaluation**: Final validation loss on a held-out split.

Analysis. Fit α separately for each (optimizer, dataset) pair by OLS on $(\log N, \log L)$ across the 4 model sizes. The predictions to test:

1. $\alpha_{\text{Muon}} > \alpha_{\text{AdamW}}$ on standard data (positive $\Delta\alpha$).
2. $\Delta\alpha$ is smaller on spectrum-flattened data than on standard data (spectral dependence).
3. The magnitude of $\Delta\alpha$ may attenuate with scale, consistent with Wen et al. [16].

Compute requirements. 2 optimizers $\times$ 2 datasets $\times$ 4 sizes $=$ 16 training runs. At Chinchilla-optimal tokens, this requires approximately 4×10^{19} FLOPs total ($\sim$80 A100-hours for the 1.3B runs, $\sim$120 A100-hours total). This is substantial but feasible for a well-resourced follow-up; it exceeds the budget for this workshop submission.

Appendix L. Code Availability

All simulation code and analysis scripts will be made available in an anonymous repository upon publication. Experiments were run on Modal cloud compute with CPU containers (no GPU required). Total compute cost: approximately 2 hours of wall-clock time on 300+ parallel containers, totaling roughly 600 CPU-hours. The entire experiment suite can be reproduced for under $20 in cloud compute.