POWERSTEP: MEMORY-EFFICIENT ADAPTIVE OPTIMIZATION VIA ℓ_p-NORM STEEPEST DESCENT

Anonymous authors
Paper under double-blind review

ABSTRACT

Adaptive optimizers such as Adam are standard for training Transformers, but storing gradient first and second moments incurs substantial memory overhead. We introduce PowerStep, a memory-efficient optimizer that achieves coordinate-wise adaptivity without storing second-moment statistics. Motivated by ℓ_p-norm steepest descent, PowerStep applies a signed-power transform directly to one momentum buffer. We establish a finite-horizon stationarity bound for exact, unregularized updates, with an $O(1/\sqrt{T})$ term and a noise-dependent residual. Experiments on Transformers from 124M to 235B parameters show competitive validation quality while halving `fp32` optimizer-state memory. Combined with uniform `int8` quantization, PowerStep remains numerically stable and reduces optimizer-state memory by $\sim 8\times$ compared to `fp32` AdamW. PowerStep thus provides a simple, memory-efficient alternative for large-scale training.

1 INTRODUCTION

Neural networks are difficult to train because of vanishing and exploding gradients (Hochreiter, 1991; Bengio et al., 1994) and ill-conditioned loss landscapes (LeCun et al., 1998; Dauphin et al., 2014; Choromanska et al., 2015), which can slow stochastic gradient descent (SGD) or cause divergence. Adaptive gradient methods (Duchi et al., 2011; Tieleman & Hinton, 2012; Kingma & Ba, 2015; Loshchilov & Hutter, 2019) precondition updates using accumulated second-moment statistics. Among these methods, Adam and its decoupled weight decay variant AdamW have become default optimizers for large-scale neural networks, particularly Transformers (Vaswani et al., 2017).

Consider an objective $f(\boldsymbol{\theta}) : \mathbb{R}^d \to \mathbb{R}$. Let $\mathbf{g}_t$ denote the stochastic gradient, and let $\mathbf{m}_t$ and $\mathbf{v}_t$ denote its first- and second-moment estimates. Omitting bias correction, Adam updates parameters as

$$\boldsymbol{\theta}_t = \boldsymbol{\theta}_{t-1} - \eta_t \cdot \mathbf{m}_t/(\sqrt{\mathbf{v}_t} + \epsilon), \tag{1}$$

where η_t is the learning rate and ϵ is a small constant. Dividing the first moment by the square root of the second moment rescales the momentum coordinate-wise, dampening updates in coordinates with large historical gradients relative to those with small ones. Equivalently, Adam applies a diagonal preconditioner $\mathbf{D}_t = \mathrm{diag}((\sqrt{\mathbf{v}_t} + \epsilon)^{-1})$ to the momentum direction. The empirical benefits of this adaptivity in Transformer training remain under active investigation (Balles & Hennig, 2018; Pan & Li, 2022; Kunstner et al., 2023; Zhang et al., 2024; Tomihari & Sato, 2025; Zhao et al., 2025; Jin et al., 2026).

However, the second-moment accumulator $\mathbf{v}_t$ incurs substantial memory overhead. It has the same dimensionality as the model parameters and is typically stored in `fp32`, doubling optimizer-state storage relative to SGD with momentum. For a model with d parameters, the two `fp32` buffers require $8d$ bytes, excluding parameters, gradients and activations. Unlike temporary tensors used during an update, these buffers persist throughout training. The second-moment buffer alone costs four bytes per parameter, making its removal a direct way to reduce persistent memory as models grow. This cost has motivated methods that approximate second-moment statistics through factorization (Shazeer & Stern, 2018) or blockwise sharing (Zheng & Kwok, 2019; Zhang et al., 2025b), quantize optimizer states (Dettmers et al., 2022; Li et al., 2023; Han et al., 2025), or dispense with the second-moment buffer entirely (Bernstein et al., 2018; Balles & Hennig, 2018; Chen et al., 2023; Zhang et al., 2025a).

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

A related line of work explores optimization under ℓ_p-norm geometry. Early work by Grove et al. (2001) and Gentile (2003) established convergence guarantees for ℓ_p-norm algorithms in linear classification and regression. Powerball (Yuan et al., 2019) uses ℓ_p-norm steepest descent to update parameters directly, while pbSGDM, the momentum variant of pbSGD (Zhou et al., 2020), applies a signed-power transform before momentum accumulation. Stacey (Luo et al., 2025) applies the transform to a momentum buffer within a primal–dual interpolation framework for non-convex optimization.

In this paper, we use ℓ_p-norm geometry to design a single-state optimizer. Our contributions are as follows:

- **Algorithm design.** We derive PowerStep from ℓ_p-norm steepest descent and show that applying a signed-power transform directly to a heavy-ball momentum buffer yields coordinate-wise adaptivity without second-moment storage.
- **Theoretical analysis.** We establish a finite-horizon stationarity bound for exact, unregularized updates, with an $O(1/\sqrt{T})$ term and an explicit noise-dependent residual. In the noiseless case, the bound yields $O(1/\sqrt{T})$ stationarity.
- **Empirical validation.** We evaluate dense and mixture-of-experts Transformers from 124M to 235B parameters. PowerStep achieves competitive validation quality while halving `fp32` optimizer-state memory. Combined with uniform `int8` quantization, it remains stable in our experiments and reduces state storage by $\sim 8\times$ relative to `fp32` AdamW.

2 ALGORITHM

PowerStep stores a single heavy-ball momentum buffer (Polyak, 1964) and updates parameters by

$$\mathbf{m}_t = \gamma \mathbf{m}_{t-1} + \mathbf{g}_t, \tag{2}$$

$$\boldsymbol{\theta}_t = \boldsymbol{\theta}_{t-1} - \eta_t \Phi_\beta(\mathbf{m}_t), \tag{3}$$

where $\gamma \in [0, 1)$ is the momentum coefficient controlling the retention of past gradients, and $\beta \in [0, 1]$ is the power exponent controlling the compression of momentum magnitudes. The transform $\Phi_\beta(\mathbf{x}) = \operatorname{sign}(\mathbf{x}) \odot |\mathbf{x}|^\beta$ acts elementwise. Algorithm 1 includes decoupled weight decay. We next motivate the direction through ℓ_p-norm geometry.

2.1 STEEPEST DESCENT UNDER ℓ_p-NORM GEOMETRY

Steepest descent selects an update direction $\mathbf{v}$ that minimizes the first-order approximation to the objective $f(\boldsymbol{\theta})$. In the framework of steepest descent in general normed spaces (Boyd & Vandenberghe, 2004), the optimal direction is defined as

$$\mathbf{v}^* = \arg\min_{\mathbf{v} \in \mathbb{R}^d} \langle \nabla f(\boldsymbol{\theta}), \mathbf{v} \rangle \quad \text{s.t. } \|\mathbf{v}\| \le 1. \tag{4}$$

The geometry of the trust region is dictated by the choice of norm. By adopting the Minkowski ℓ_p norm, $\|\mathbf{v}\|_p = (\sum_{i=1}^d |v_i|^p)^{1/p}$, we define a continuum of geometries. To derive the update, we form the Lagrangian using the p-th power of the norm constraint

$$\mathcal{L}(\mathbf{v}, \mu) = \sum_{i=1}^d g_i v_i + \mu \left(\sum_{i=1}^d |v_i|^p - 1 \right), \tag{5}$$

where g_i is the i-th component of the gradient $\mathbf{g} = \nabla f(\boldsymbol{\theta})$ and $\mu > 0$ is the Lagrange multiplier. Setting $\frac{\partial \mathcal{L}}{\partial v_i} = 0$ yields the coordinate-wise update rule

$$v_i^* \propto -\operatorname{sign}(g_i) \cdot |g_i|^{\frac{1}{p-1}}. \tag{6}$$

For $\mathbf{g} \neq 0$ and $1 < p < \infty$, the normalized solution is $\mathbf{v}^* = -\Phi_\beta(\mathbf{g}) / \|\mathbf{g}\|_{1+\beta}^\beta$, where $\beta = 1/(p-1)$. Powerball (Yuan et al., 2019) and PowerStep use the unnormalized signed-power direction; the learning rate determines its length rather than imposing a fixed trust-region radius.

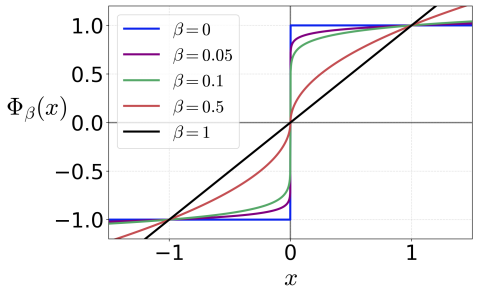


Figure 1: Signed-power transform for several exponents. Coordinate-wise magnitude compression interpolates between a sign update and a linear update.

2.2 Adaptivity via ℓ_p-norm steepest descent

The transform defined above follows the direction in (6) when $\beta = 1/(p-1) \in (0, 1]$ for $p \in [2, \infty)$. At $\beta = 0$, we define $\Phi_0 = \mathrm{sign}$ with $\mathrm{sign}(0) = 0$. As illustrated in Figure 1, for $0 < \beta < 1$, the transform compresses the ratio of two nonzero magnitudes from r to r^β, reducing magnitude disparities while preserving signs.

PowerStep applies this transform after heavy-ball momentum accumulation (Polyak, 1964), allowing temporal smoothing before nonlinear scaling. Powerball instead transforms raw gradients. For a constant gradient $\mathbf{g}$, heavy-ball momentum approaches $\mathbf{g}/(1-\gamma)$, so the signed-power update gains a factor $(1-\gamma)^{-\beta}$. At $(\gamma, \beta) = (0.9, 0.1)$ this factor is only 1.26, providing a scale-based rationale for testing the same learning-rate range as AdamW. The theory studies updates (2)–(3) without weight decay or gradient clipping.

Equivalently, the PowerStep update (3) can be expressed as a preconditioned momentum step,

$$\boldsymbol{\theta}_t = \boldsymbol{\theta}_{t-1} - \eta_t \mathbf{D}_t \mathbf{m}_t, \tag{7}$$

$$\mathbf{D}_t = \mathrm{diag}\big(|\mathbf{m}_t|^{\beta-1}\big), \tag{8}$$

where the i-th diagonal entry is $|m_{t,i}|^{\beta-1}$ for nonzero $m_{t,i}$. At zero, this expression is singular for $\beta < 1$, but the implemented transform $\Phi_\beta(0) = 0$ is well defined. For $0 < \beta < 1$, each entry is inversely related to momentum magnitude. This coordinate-wise scaling is computed directly from $\mathbf{m}_t$ without a second-moment estimator.

At the extremes, PowerStep recovers classical methods: $\beta = 0$ yields $\Phi_0(\mathbf{m}_t) = \mathrm{sign}(\mathbf{m}_t)$, reducing to SignSGD with momentum (Bernstein et al., 2018; Balles & Hennig, 2018); $\beta = 1$ yields $\Phi_1(\mathbf{m}_t) = \mathbf{m}_t$, recovering standard SGD with momentum (Polyak, 1964). Between these limits, PowerStep interpolates between sign-based and linear updates, with $\beta = 0.1$ providing a practically effective trade-off (Section 5).

Algorithm 1 PowerStep

Require: Initial parameters $\boldsymbol{\theta}_0$, learning rates $\{\eta_t\}_{t=1}^T$
Require: Momentum $\gamma \in [0, 1)$, exponent $\beta \in [0, 1]$, weight decay $\lambda \geq 0$
$\mathbf{m}_0 \leftarrow \mathbf{0}$
for $t = 1$ to T **do**
  $\mathbf{g}_t \leftarrow \nabla f_t(\boldsymbol{\theta}_{t-1})$ ▷ Stochastic gradient
  $\mathbf{m}_t \leftarrow \gamma \mathbf{m}_{t-1} + \mathbf{g}_t$ ▷ Accumulate gradients
  $\mathbf{u}_t \leftarrow \mathrm{sign}(\mathbf{m}_t) \odot |\mathbf{m}_t|^\beta$ ▷ Elementwise signed-power transform
  $\boldsymbol{\theta}_t \leftarrow (1 - \eta_t \lambda)\boldsymbol{\theta}_{t-1} - \eta_t \mathbf{u}_t$
end for

2.3 Relation to prior methods

Powerball (Yuan et al., 2019) applies the signed-power transform to raw gradients, providing no temporal smoothing. pbSGDM (Zhou et al., 2020) applies it before momentum accumulation.

PowerStep instead applies it after momentum accumulation, so the nonlinear scaling reflects the accumulated state. These operations do not commute; Section 5 compares their empirical behavior.

Stacey (Luo et al., 2025) applies a stabilized transform $\Phi^{\epsilon}_{\beta}(\mathbf{x}) = \mathrm{sign}(\mathbf{x}) \odot (|\mathbf{x}| + \epsilon)^{\beta}$ to a momentum-like state in a primal–dual framework, making it the closest relative. Stacey maintains two state buffers and aims to accelerate optimization through primal–dual interpolation. PowerStep has a different aim: reducing optimizer-state memory. It removes the auxiliary state and stabilization term, retaining only one momentum buffer. We compare PowerStep with Stacey in Appendix D.

Methods such as GaLore (Zhao et al., 2024) and LDAdam (Robert et al., 2025) reduce optimizer-state memory by projecting gradients into a low-dimensional subspace, maintaining optimizer states and computing updates there, and projecting the updates back to the full parameter space. This subspace-based compression is orthogonal to PowerStep's single-state update design and could in principle be combined with PowerStep.

3 Convergence analysis

We analyze the exact, unregularized update ($\lambda = 0$), without gradient clipping or state quantization. The guarantee separates a finite-horizon term from a noise-dependent residual. Let $0 < \beta \leq 1$, $q = 1 + \beta$, $p = q/\beta$, and $h = 1 - \gamma > 0$. These are dual exponents, and

$$\langle \mathbf{m}, \Phi_{\beta}(\mathbf{m}) \rangle = \|\mathbf{m}\|_q^q, \qquad \|\Phi_{\beta}(\mathbf{m})\|_p = \|\mathbf{m}\|_q^{\beta}. \tag{9}$$

Assumption 1 (Smoothness and boundedness)**.** *For constants $L_p, G > 0$, the continuously differentiable objective is bounded below by f^*, satisfies $\|\nabla f(\mathbf{x})\|_q \leq G$, and has a Lipschitz gradient in the dual pair:*

$$\|\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})\|_q \leq L_p \|\mathbf{x} - \mathbf{y}\|_p. \tag{10}$$

Assumption 2 (Stochastic oracle)**.** *With $\mathcal{F}_{t-1}$ the history before sampling $\mathbf{g}_t$, write $\bar{\mathbf{g}}_t = \nabla f(\boldsymbol{\theta}_{t-1})$ and $\boldsymbol{\xi}_t = \mathbf{g}_t - \bar{\mathbf{g}}_t$. We assume $\mathbb{E}[\|\boldsymbol{\xi}_t\|_q^q \mid \mathcal{F}_{t-1}] \leq \sigma^q$; the oracle may be biased.*

The q-moment condition need not imply a second moment for $\boldsymbol{\xi}_t$ when $q < 2$. Our proof only bounds the second moment of the *transformed update*, using $2\beta \leq q$.

Theorem 1 (Finite-horizon stationarity bound)**.** *Under Assumptions 1–2, fix a horizon T, use $\eta_t = \eta = c/\sqrt{T}$ with $c > 0$, and initialize $\mathbf{m}_0 = \mathbf{0}$. Let $\nu_t = \mathbb{E}\|\boldsymbol{\xi}_t\|_q^q$. There are explicit constants $C_1, C_2 \geq 0$ and $C_{\beta} > 0$, independent of T, such that*

$$\frac{1}{T}\sum_{t=1}^{T} \mathbb{E}\|\bar{\mathbf{g}}_t\|_q^q \leq \underbrace{\frac{C_1}{\sqrt{T}} + \frac{C_2}{T}}_{R_T} + \frac{C_{\beta}}{T}\sum_{t=1}^{T} \nu_t. \tag{11}$$

Here C_1, C_2 depend on $L_p, G, \sigma, \gamma, \beta, c$ and the initial objective gap; C_{β} depends only on β. Appendix G gives their formulas. Consequently, $\min_{t \leq T} \mathbb{E}\|\bar{\mathbf{g}}_t\|_2^2 \leq G^{1-\beta}(R_T + C_{\beta}\sigma^q)$.

The bound gives $O(T^{-1/2})$ stationarity when $\sigma = 0$; fixed stochastic noise leaves a residual. The constant step size differs from experimental warmup/cosine schedules. Constants may depend on dimension after conversion from Euclidean norms and become loose as $\beta \to 0$; the theorem excludes $\beta = 0$ and does not establish an optimal rate. Appendix G gives the proof, explicit constants, dimension conversions and conditions for a vanishing residual.

4 Low-precision state storage

Dettmers et al. (2022) reduce optimizer-state memory using block-wise dynamic quantization, with a scale per block and a nonuniform codebook spanning different magnitudes. We label the evaluated baseline AdamW (Dettmers `int8`); its local codebooks differ from those in the bitsandbytes implementation. Our simpler scheme linearly quantizes all states in 128-coordinate blocks, including embeddings. PowerStep stores only its momentum: each step reconstructs $\hat{\mathbf{m}}_{t-1}$, computes $\mathbf{a}_t = \mathbf{g}_t + \gamma\hat{\mathbf{m}}_{t-1}$ and $\Phi_{\beta}(\mathbf{a}_t)$ in full precision, then stores $Q(\mathbf{a}_t)$. One byte per coordinate plus one `fp32` scale per block costs $1 + 4/128$ bytes per parameter, a $\sim 8\times$ reduction from AdamW's two `fp32` moments (8 bytes).

Away from zero, the signed-power transform has relative condition number β, attenuating small relative momentum errors for $\beta < 1$. An absolute Hölder bound applies at zero. For AdamW, rounding $v = 10^{-8}$ to zero with $\epsilon = 10^{-8}$ multiplies the reciprocal factor $1/(\sqrt{v} + \epsilon)$ by 10,001, holding the numerator fixed. Appendix E derives these sensitivity bounds; Section 5.3 compares the schemes empirically.

5 EXPERIMENTS

5.1 SETUP

Models and datasets. We evaluate Transformer pretraining from 124M to 235B parameters. Dense models include GPT-2 (124M, 350M) (Radford et al., 2019) and Qwen3 (0.6B, 1.7B, 4B, 8B, 32B) (Yang et al., 2025); MoE models include DeepSeek-V2-Lite (16B total, 2.4B active) (DeepSeek-AI et al., 2024), Qwen3-30B-A3B and Qwen3-235B-A22B. GPT-2 uses OpenWebText (Gokaslan et al., 2019); the other models use C4 (Raffel et al., 2020).

Optimizers and hyperparameters. The main comparison uses AdamW, AdamS, SignSGD with momentum, pbSGDM and PowerStep with `fp32` states; quantized states are labeled explicitly. AdamW and AdamS use $(\beta_1, \beta_2) = (0.9, 0.95)$ and $\epsilon = 10^{-8}$; PowerStep and pbSGDM use $(\gamma, \beta) = (0.9, 0.1)$. The main recipe uses $\eta_{\max} \in [2, 6] \times 10^{-4}$, depending on model size, weight decay 0.1, gradient clipping at 1.0, and 2,000-step warmup followed by cosine decay. GPT-2 uses batch 480, context length 1,024 and 50,000 updates; other models use batch 256, context length 2,048 and 20,000 updates. MoE load-balancing loss has coefficient 10^{-3}. Appendix A gives per-model settings and implementation details.

Infrastructure and precision. We use Ascend hardware with nanoGPT for GPT-2 and Megatron-Core/MindSpeed-LLM for larger models. The large-model recipe uses `bf16`; the matched nanoGPT study in Appendix B uses `fp32` parameters with `bf16` autocast. Its settings, and those of Muon and SFT, are stated separately.

5.2 COMPARISON WITH PRIOR METHODS

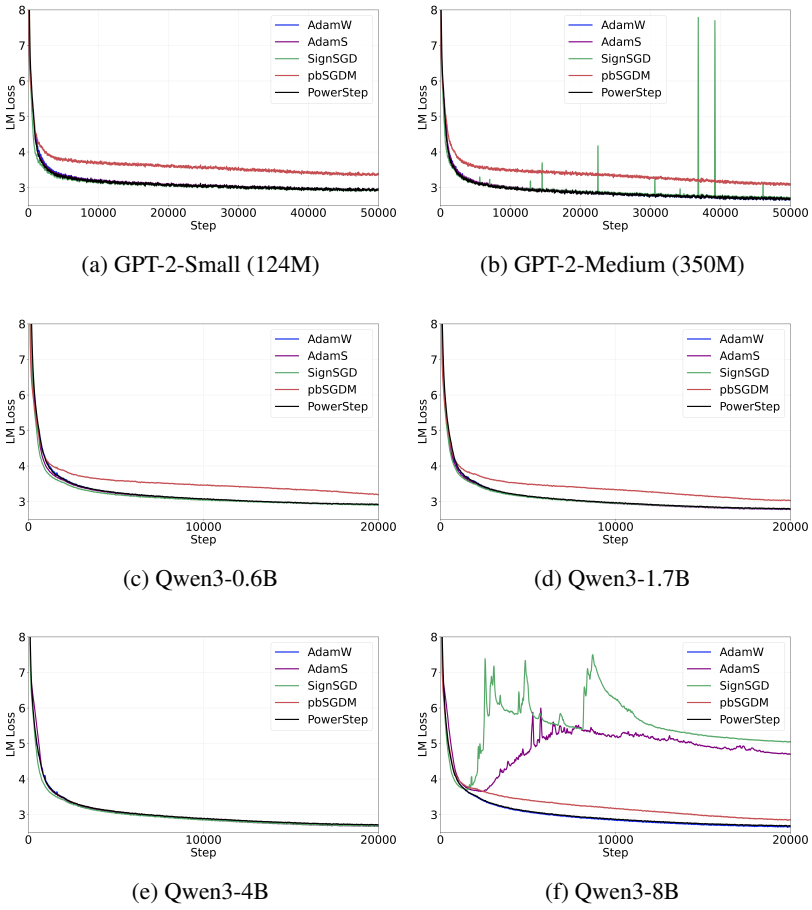
On GPT-2-Medium, AdamW (`fp32`), PowerStep (`fp32`) and PowerStep (`int8`) share data order, validation batches, decay groups and a 9,000-update budget (4.424B token positions). Table 5 in Appendix B reports validation means and sample standard deviations. The `fp32` and `int8` PowerStep configurations achieve slightly lower mean validation losses while reducing state storage from 2.642 to 1.321 and 0.341 GiB per device, respectively. In separate timing measurements using the same training shapes, peak allocation falls from 25.31 to 23.99 and 23.06 GiB, with throughput costs of 1.3% and 3.6%, respectively. Appendix B gives the shared protocol and detailed results; Appendix A specifies the broader model comparison.

Figure 2 extends the comparison to six dense models with `fp32` states, 50,000 updates for GPT-2 and 20,000 for Qwen3. PowerStep's validation loss is 2.949 versus AdamW's 2.950 on GPT-2-Small and 0.006–0.028 higher on the other five models (Table 7). SignSGD is best on Small and Qwen3-0.6B; AdamS is best on Qwen3-4B. The cause of the high AdamS and SignSGD losses on 8B remains unresolved. Together, these results show competitive validation quality across model scales, with the matched study controlling data order and evaluation samples.

5.3 QUANTIZATION

We evaluate how optimizer-state quantization affects training on GPT-2-Small and GPT-2-Medium. Both quantization schemes use 128-coordinate blocks. Uniform `int8` uses equally spaced levels for all state tensors, including embeddings. Dettmers `int8` (Dettmers et al., 2022) uses a nonuniform codebook and retains embedding and small-tensor states in `fp32`. Figure 3 compares uniformly quantized AdamW and PowerStep with their `fp32` references, and includes AdamW with Dettmers `int8`.

PowerStep trains stably with uniform `int8` on both models, whereas AdamW diverges under the same scheme. Dettmers `int8` stabilizes AdamW on GPT-2-Small, reaching validation loss 2.9524



(a) GPT-2-Small (124M)

(b) GPT-2-Medium (350M)

(c) Qwen3-0.6B

(d) Qwen3-1.7B

(e) Qwen3-4B

(f) Qwen3-8B

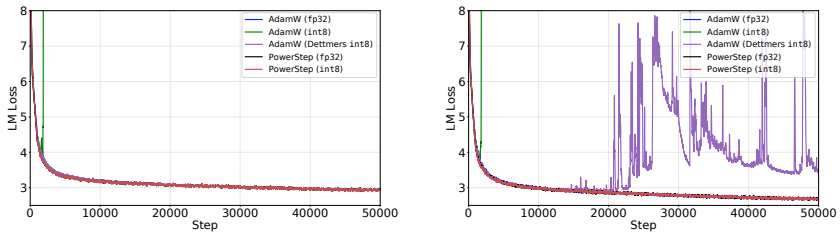
Figure 2: Training losses with `fp32` states. Validation results are in Table 7.

after 50,000 updates, but GPT-2-Medium exhibits repeated loss spikes. Its validation loss spikes at 16,000 updates, reaches 7.8447 at 48,000, and ends at 3.4768 after 50,000 updates.

AdamW divides by $\sqrt{v} + \epsilon$, making its update sensitive to underestimation of the second moment. Low-precision Adam methods use specialized stabilization: Li et al. (2023) use zero-excluding second-moment quantizers and improved normalization for 4-bit states. Q-Adam-mini (Han et al., 2025) reports embedding weight-norm instability from first-moment quantization and mitigates it with stochastic rounding. Stable-SPAM (Huang et al., 2025) uses adaptive gradient normalization, clipping, and moment resets for 4-bit weights and activations. PowerStep supports simple uniform state quantization through a different update structure: it removes the second-moment denominator, and its signed-power transform attenuates small relative momentum errors away from zero, with local relative condition number β. Appendix E analyzes update sensitivity and the AdamW failure timeline; loss traces alone do not identify the trigger of the Medium spikes.

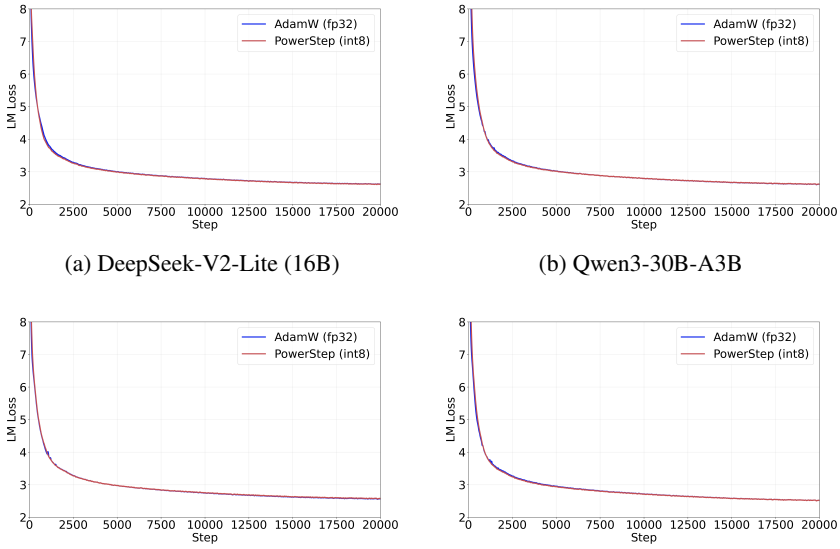
5.4 Large-scale experiments

Figure 4 compares PowerStep with uniformly quantized `int8` momentum states against AdamW with `fp32` states. PowerStep uses the 128-coordinate block-wise linear scheme in Section 4, with one scale per block. Each model trains for 20,000 updates at global batch 256 and context length 2,048. Table 1 reports held-out loss and optimizer-state storage for the dense and sparse MoE models. Across these models, optimizer-state storage decreases by $\sim 8\times$ (87.1%, from $1 - (1 + 4/128)/8$). For Qwen3-235B-A22B, it falls from 6,768.0 to 872.4 MB per device.



(a) GPT-2-Small (b) GPT-2-Medium

Figure 3: Training loss on GPT-2-Small and GPT-2-Medium with `fp32` and `int8` optimizer states. Curves show 50-step moving averages. Both Dettmers runs complete 50,000 updates; Medium exhibits repeated loss spikes. All quantized curves use 128-coordinate blocks. Experimental details are in Appendix E.



(a) DeepSeek-V2-Lite (16B) (b) Qwen3-30B-A3B

(c) Qwen3-32B (d) Qwen3-235B-A22B

Figure 4: Large-model training losses over 20,000 updates: PowerStep with uniform `int8` momentum quantization in 128-coordinate blocks and AdamW with `fp32` states.

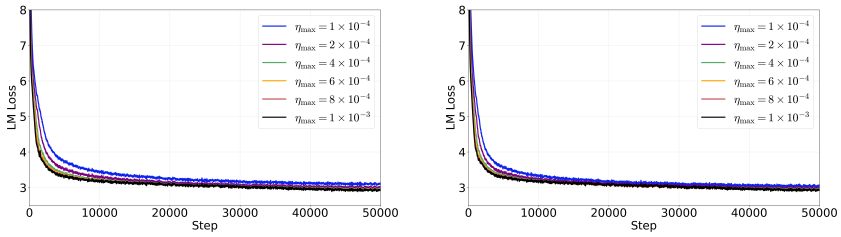
Table 1: Large-model results after 20,000 steps. State-memory columns report the recorded average optimizer-state storage per device, in MB.

	Validation loss		Optimizer state (MB)	
Model	AdamW	`int8` PowerStep	AdamW	`int8` PowerStep
DeepSeek-V2-Lite (16B)	2.623	**2.618**	429.0	**55.3**
Qwen3-30B-A3B	**2.620**	2.624	864.0	**111.4**
Qwen3-32B	**2.572**	2.582	976.5	**125.9**
Qwen3-235B-A22B	2.525	**2.524**	6768.0	**872.4**

5.5 Hyperparameter sensitivity

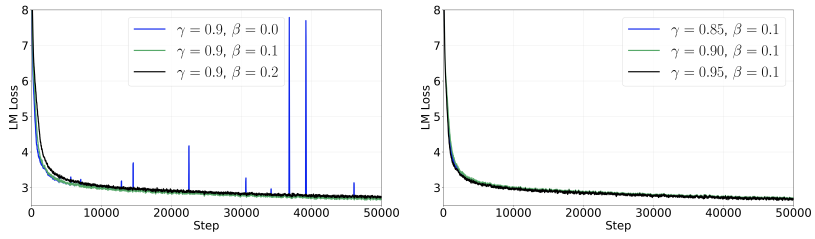
We conduct ablations on the learning rate, power exponent β, and momentum coefficient γ. On GPT-2-Small, we vary $\eta_{\max} \in \{1, 2, 4, 6, 8, 10\} \times 10^{-4}$ with $\eta_{\min} = 0.1\eta_{\max}$, keeping other settings fixed.

Figure 5 shows faster early loss reduction at larger $\eta_{\max}$ for both AdamW and PowerStep. Figure 6 varies $\beta \in \{0, 0.1, 0.2\}$ at $\gamma = 0.9$, and $\gamma \in \{0.85, 0.9, 0.95\}$ at $\beta = 0.1$, on GPT-2-Medium. Smaller β compresses momentum magnitude ratios more strongly toward a sign update. Smaller values reduce loss faster initially; Appendix F.6 explains how near-sign updates can overshoot. In practice, we recommend β near 0.1. Larger γ retains the influence of past gradients for longer; our default is $(\beta, \gamma) = (0.1, 0.9)$. In the separate exponent sweep (Table 11), observed validation endpoints vary by less than 0.04 on each GPT-2 model across $\beta \in \{0.05, 0.10, 0.15\}$.



(a) AdamW (b) PowerStep

Figure 5: Sensitivity to $\eta_{\max}$ on GPT-2-Small (124M): AdamW (left) and PowerStep (right). Within each optimizer, we vary $\eta_{\max}$, with $\eta_{\min} = \eta_{\max}/10$; all other settings are fixed.



(a) Power exponent β (b) Momentum coefficient γ

Figure 6: GPT-2-Medium sensitivity study. The $\beta = 0$ curve has unresolved provenance and is shown for reference only: it differs from the SignSGD run in Figure 2.

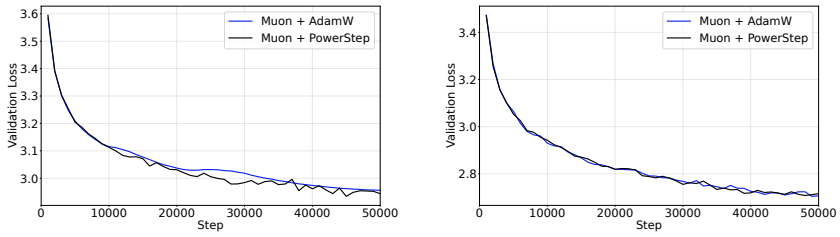
5.6 Compatibility with Muon

We test PowerStep as the auxiliary optimizer for Muon (Jordan, 2024; Bernstein & Newhouse, 2024): Muon updates the two-dimensional matrices inside Transformer blocks, while PowerStep updates embeddings, the tied output head and scalar/vector parameters. Figure 7 uses the same cosine schedule from $\eta_{\max} = 6 \times 10^{-4}$ to $\eta_{\min} = 6 \times 10^{-5}$ for both parameter groups in both hybrids, without warmup. On Small, Muon–AdamW reaches validation losses 2.9558 and 3.0353 across two runs at 50,000 updates (mean 2.9956; Table 12), compared with 2.9438 for Muon–PowerStep. On Medium, the corresponding losses are 2.7065 for Muon–AdamW and 2.7159 for Muon–PowerStep.

The Small Muon–AdamW runs use fixed validation samples, while Muon–PowerStep uses the earlier evaluation protocol. The comparison demonstrates compatibility rather than isolating the auxiliary optimizer. Appendix F.3 reports the reduced-η_t ablation and parameter mappings.

5.7 Supervised fine-tuning

We fine-tune Qwen3-4B-Base on a filtered no-tool Nemotron-Math-v2 subset (NVIDIA, 2026). Both runs complete 14,200 updates with global batch 128, context length 6,144, `bf16`, and linear decay from $\eta_{\max} = 10^{-5}$ to $\eta_{\min} = 10^{-6}$. Figure 8 shows lower training loss for PowerStep; the final 100-update means are 0.0932 versus 0.1053 for AdamW.



(a) GPT-2-Small (b) GPT-2-Medium

Figure 7: Muon hybrid validation loss on GPT-2-Small and GPT-2-Medium. The Small Muon–AdamW curve ends at 2.9558; Table 12 reports the two-run mean. Validation protocols differ; the initial evaluation is omitted.

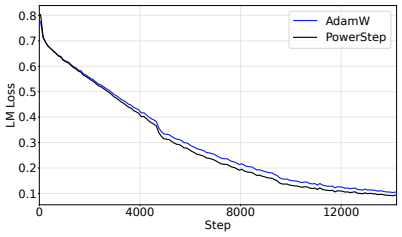


Figure 8: Qwen3-4B SFT training loss, averaged over non-overlapping 100-update windows.

Table 2: Final SFT checkpoint evaluation. Pass@1 is estimated from 16 responses per question: 40 AMC23 questions and 30 per AIME set.

Benchmark	Optimizer	Pass@1 accuracy (%)
AMC23	AdamW	56.09
	PowerStep	**57.34**
AIME24	AdamW	18.54
	PowerStep	**19.38**
AIME25	AdamW	20.21
	PowerStep	**21.25**

Both optimizers start from the same base checkpoint. Training responses contain boxed answers and span 1,000–4,000 tokens; weight decay is 0.1 and gradient clipping is 1.0. PowerStep improves Pass@1 by 1.25 percentage points on AMC23, 0.83 on AIME24 and 1.04 on AIME25 (Table 2).

Evaluation samples 16 responses per question at temperature 0.6, top-$p = 0.95$ and top-$k = 40$, using the same last-boxed-integer scorer. Pass@1 averages the correct-response fraction per question. Appendix F.4 details the protocol and data provenance.

6 Discussion and Conclusion

PowerStep retains competitive validation quality using one momentum buffer, reducing optimizer-state memory and measured peak allocation at a modest throughput cost. Uniform state quantization provides further memory savings.

Future work includes extending PowerStep to lower-bit state storage and combining it with subspace-based compression. Analyzing quantized updates and adaptive choices of β could further connect numerical robustness with convergence, guiding single-state optimizer design for larger models.

AI USE STATEMENT

Generative AI tools (GPT-6, Kimi-K3, and GLM-5.3) assisted with literature comparison, formulation and writing of the finite-horizon analysis and proofs, experimental design, implementation of the matched training and measurement scripts, interpretation and auditing of logs, figure and table preparation, and manuscript editing and formatting. Numerical results in the paper come from executed experiments or archived records, not generated sample values. Verification performed during preparation includes script compilation, short NPU training and checkpoint tests, cross-checks against logs, and numerical checks of proof inequalities; these checks are not a substitute for independent mathematical review. The authors are responsible for the final text, code, analysis and claims.

REPRODUCIBILITY STATEMENT

Algorithm 1 specifies the core update. Section 3 and Appendix G state assumptions and provide proofs. Appendix B specifies the matched experiment and measurement protocol; the anonymous supplementary code includes its training scripts and numerical records. Appendices A and F document the additional experiments and their recorded configurations. Appendix F.4 gives the SFT and scoring protocol.

REFERENCES

Lukas Balles and Philipp Hennig. Dissecting Adam: The sign, magnitude and variance of stochastic gradients. In *International Conference on Machine Learning*, 2018.

Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 1994.

Jeremy Bernstein and Laker Newhouse. Old optimizer, new norm: An anthology. *arXiv preprint arXiv:2409.20325*, 2024.

Jeremy Bernstein, Yu-Xiang Wang, Kamyar Azizzadenesheli, and Animashree Anandkumar. signSGD: Compressed optimisation for non-convex problems. *International Conference on Machine Learning*, 2018.

Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

Xiangning Chen, Chen Liang, Da Huang, Esteban Real, Kaiyuan Wang, Hieu Pham, Xuanyi Dong, Thang Luong, Cho-Jui Hsieh, Yifeng Lu, and Quoc V. Le. Symbolic discovery of optimization algorithms. *Advances in Neural Information Processing Systems*, 2023.

Anna Choromanska, Mikael Henaff, Michael Mathieu, Gérard Ben Arous, and Yann LeCun. The loss surfaces of multilayer networks. *International Conference on Artificial Intelligence and Statistics*, 2015.

Yann N Dauphin, Razvan Pascanu, Caglar Gulcehre, Kyunghyun Cho, Surya Ganguli, and Yoshua Bengio. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. *Advances in Neural Information Processing Systems*, 2014.

DeepSeek-AI, Aixin Liu, Bei Feng, Bin Wang, Bingxuan Wang, Bo Liu, Chenggang Zhao, Chengqi Deng, Chong Ruan, Damai Dai, et al. Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model. *arXiv preprint arXiv:2405.04434*, 2024.

Tim Dettmers, Mike Lewis, Sam Shleifer, and Luke Zettlemoyer. 8-bit optimizers via block-wise quantization. *International Conference on Learning Representations*, 2022.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 2011.

Claudio Gentile. The robustness of the p-norm algorithms. *Machine Learning*, 2003.

Aaron Gokaslan, Vanya Cohen, Ellie Pavlick, and Stefanie Tellex. Openwebtext corpus, 2019. https://skylion007.github.io/OpenWebTextCorpus/.

Adam J. Grove, Nick Littlestone, and Dale Schuurmans. General convergence results for linear discriminant updates. *Machine Learning*, 2001.

Yizhou Han, Chaohao Yang, Congliang Chen, Xingjian Wang, and Ruoyu Sun. Q-adam-mini: Memory-efficient 8-bit quantized optimizer for large language model training. *ICML Workshop ES-FoMo III*, 2025.

Sepp Hochreiter. Untersuchungen zu dynamischen neuronalen netzen. *Diploma thesis, Technische Universität München*, 1991.

Tianjin Huang, Haotian Hu, Zhenyu Zhang, Gaojie Jin, Xiang Li, Li Shen, Tianlong Chen, Lu Liu, Qingsong Wen, Zhangyang Wang, and Shiwei Liu. Stable-SPAM: How to train in 4-bit more stably than 16-bit Adam. *arXiv preprint arXiv:2502.17055v1*, 2025.

Ruinan Jin, Yingbin Liang, and Shaofeng Zou. Why adam can beat sgd: Second-moment normalization yields sharper tails. *arXiv preprint arXiv:2603.03099*, 2026.

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

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 2015.

Frederik Kunstner, Jacques Chen, Jonathan Wilder Lavington, and Mark Schmidt. Noise is not the main factor behind the gap between sgd and adam on transformers, but sign descent might be. *International Conference on Learning Representations*, 2023.

Yann LeCun, Léon Bottou, Genevieve B Orr, and Klaus-Robert Müller. Efficient backprop. *Neural Networks: Tricks of the Trade*, 1998.

Bingrui Li, Jianfei Chen, and Jun Zhu. Memory-efficient optimizers with 4-bit states. *Advances in Neural Information Processing Systems*, 2023.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *International Conference on Learning Representations*, 2019.

Xinyu Luo, Site Bai, Bolian Li, Petros Drineas, Ruqi Zhang, and Brian Bullins. Stacey: Promoting stochastic steepest descent via accelerated ℓ_p-smooth nonconvex optimization. *International Conference on Machine Learning*, 2025.

Chao Ma, Wenbo Gong, Meyer Scetbon, and Edward Meeds. SWAN: SGD with normalization and whitening enables stateless LLM training. *International Conference on Machine Learning*, 2025.

NVIDIA. Nemotron-math-v2. Dataset, 2026. https://huggingface.co/datasets/nvidia/Nemotron-Math-v2.

Yan Pan and Yuanzhi Li. Toward understanding why adam converges faster than SGD for transformers. *OPT 2022: Optimization for Machine Learning Workshop*, 2022.

Boris T Polyak. Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 1964.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI Technical Report*, 2019.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 2020.

Thomas Robert, Mher Safaryan, Ionut-Vlad Modoranu, and Dan Alistarh. LDAdam: Adaptive optimization from low-dimensional gradient statistics. *International Conference on Learning Representations*, 2025.

Meyer Scetbon, Chao Ma, Wenbo Gong, and Edward Meeds. Gradient multi-normalization for stateless and scalable LLM training. *arXiv preprint arXiv:2502.06742*, 2025.

Noam Shazeer and Mitchell Stern. Adafactor: Adaptive learning rates with sublinear memory cost. *International Conference on Machine Learning*, 2018.

Tijmen Tieleman and Geoffrey Hinton. Lecture 6.5—RMSProp: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural Networks for Machine Learning, 2012. Lecture slides.

Akiyoshi Tomihari and Issei Sato. Understanding why Adam outperforms SGD: Gradient heterogeneity in transformers. *arXiv preprint arXiv:2502.00213v1*, 2025.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Ye Yuan, Mu Li, Jun Liu, and Claire Tomlin. On the powerball method: variants of descent methods for accelerated optimization. *IEEE Control Systems Letters*, 2019.

Huishuai Zhang, Bohan Wang, and Luoxin Chen. AdamS: Momentum itself can be a normalizer for LLM pretraining and post-training. *Conference on Empirical Methods in Natural Language Processing*, 2025a.

Yushun Zhang, Congliang Chen, Tian Ding, Ziniu Li, Ruoyu Sun, and Zhi-Quan Luo. Why transformers need adam: A hessian perspective. *Advances in Neural Information Processing Systems*, 2024.

Yushun Zhang, Congliang Chen, Ziniu Li, Tian Ding, Chenwei Wu, Diederik (Durk) Kingma, Yinyu Ye, Zhi-Quan Luo, and Ruoyu Sun. Adam-mini: Use fewer learning rates to gain more. *International Conference on Learning Representations*, 2025b.

Jiawei Zhao, Zhenyu Zhang, Beidi Chen, Zhangyang Wang, Anima Anandkumar, and Yuandong Tian. GaLore: Memory-efficient LLM training by gradient low-rank projection. In *Proceedings of the 41st International Conference on Machine Learning*. PMLR, 2024.

Rosie Zhao, Depen Morwani, David Brandfonbrener, Nikhil Vyas, and Sham Kakade. Deconstructing what makes a good optimizer for autoregressive language models. *International Conference on Learning Representations*, 2025.

Shuai Zheng and James T. Kwok. Blockwise adaptivity: Faster training and better generalization in deep learning. *arXiv preprint arXiv:1905.09899*, 2019.

Beitong Zhou, Jun Liu, Weigao Sun, Ruijuan Chen, Claire Tomlin, and Ye Yuan. pbSGD: Powered stochastic gradient descent methods for accelerated non-convex optimization. *International Joint Conferences on Artificial Intelligence*, 2020.

A EXPERIMENTAL DETAILS

A.1 HYPERPARAMETERS

Table 3 reports the main pretraining learning rates. Weight decay is 0.1, global gradient norm clipping is 1.0, and the MoE auxiliary load-balancing coefficient is 10^{-3}. Warmup takes 2,000 steps, followed by cosine decay to the stated minimum at step 50,000 for GPT-2 or step 20,000 for all other models. Global batch size counts sequences; nominal token totals are batch size times context length times steps, without correcting for padding.

The main comparison uses a common schedule within each model. Section 5.5 varies the learning rate for AdamW and PowerStep on GPT-2-Small; Table 11 separately varies AdamW's β_2 and PowerStep's exponent. In particular, pbSGDM uses the same γ and β as PowerStep in the main comparison.

Table 3: Training hyperparameters

Model	Batch size	Context length	$\eta_{\max}$	$\eta_{\min}$
GPT-2-Small (124M)	480	1024	6×10^{-4}	6×10^{-5}
GPT-2-Medium (350M)	480	1024	6×10^{-4}	6×10^{-5}
Qwen3-0.6B	256	2048	5×10^{-4}	5×10^{-5}
Qwen3-1.7B	256	2048	5×10^{-4}	5×10^{-5}
Qwen3-4B	256	2048	5×10^{-4}	5×10^{-5}
Qwen3-8B	256	2048	3×10^{-4}	3×10^{-5}
DeepSeek-V2-Lite (16B)	256	2048	2×10^{-4}	2×10^{-5}
Qwen3-30B-A3B	256	2048	2×10^{-4}	2×10^{-5}
Qwen3-32B	256	2048	2×10^{-4}	2×10^{-5}
Qwen3-235B-A22B	256	2048	2×10^{-4}	2×10^{-5}

A.2 PARALLELISM CONFIGURATION

Table 4 lists the parallelism configuration for each model. We denote data parallelism by DP, tensor parallelism by TP, pipeline parallelism by PP and expert parallelism by EP. The total number of NPUs satisfies NPUs $=$ DP $\times$ TP $\times$ PP. For MoE architectures, expert parallelism partitions the data-parallel ranks, with EP $\leq$ DP; it is not an additional factor in the device count.

Table 4: Parallelism configuration across models

Model	NPUs	DP	TP	PP	EP
GPT-2-Small (124M)	8	8	1	1	N/A
GPT-2-Medium (350M)	8	8	1	1	N/A
Qwen3-0.6B	8	8	1	1	N/A
Qwen3-1.7B	8	8	1	1	N/A
Qwen3-4B	8	8	1	1	N/A
Qwen3-8B	32	16	1	2	N/A
DeepSeek-V2-Lite (16B)	256	256	1	1	8
Qwen3-30B-A3B	256	64	1	4	4
Qwen3-32B	256	16	8	2	N/A
Qwen3-235B-A22B	256	64	1	4	8

B MATCHED THREE-SEED STUDY AND EFFICIENCY MEASUREMENTS

B.1 MATCHED THREE-SEED STUDY

Three optimizer configurations use GPT-2-Medium with the same three initialization seeds, data order, validation samples and decay groups. Each run uses eight Ascend 910 devices, global batch 480 and context length 1,024, with 9,000 updates (4.424B token positions). η_t warms up for 900

steps to $\eta_{\max} = 6 \times 10^{-4}$ and decays to $\eta_{\min} = 6 \times 10^{-5}$. The horizon was fixed from hardware timing before inspecting losses; all configurations use the same schedule.

Table 5: Matched three-seed study on GPT-2-Medium at 9,000 updates. Memory is GiB per device; peak allocation is measured separately.

Configuration	Validation loss (mean ± SD)	State	Peak
AdamW (`fp32`)	2.9563±0.0026	2.642	25.31
PowerStep (`fp32`)	2.9487±0.0056	1.321	23.99
PowerStep (`int8`)	**2.9484±0.0047**	**0.341**	**23.06**

Table 6 lists every completed seed; the supplementary data provide their validation trajectories.

B.2 Detailed protocol and individual results

Design. Each method uses GPT-2-Medium with 24 layers, 16 heads, hidden size 1024, vocabulary 50,304, no bias or dropout, and context length 1024. All methods use `fp32` parameters with `bf16` autocast. Eight devices each process microbatches of 12 sequences and accumulate five gradients before an update, giving global batch 480. Gradients are globally synchronized and clipped to norm 1.0. Weight decay 0.1 applies to parameters with dimension at least two, with no decay on scalar/vector parameters, identically for all methods. AdamW uses $(\beta_1, \beta_2) = (0.9, 0.95)$ and $\epsilon = 10^{-8}$; PowerStep uses $(\gamma, \beta) = (0.9, 0.1)$. All runs start from scratch.

For each seed $s \in \{1337, 1338, 1339\}$, model initialization uses s, and rank r samples training batches with a separate CPU generator seeded by $s + 100000 + r$. Evaluation uses another generator with seed $918273 + r$, reset at each evaluation, keeping its samples fixed across methods, seeds and checkpoints. Each rank evaluates 25 batches, giving 2,400 sequences per checkpoint. Validation consumes no training-generator randomness. Learning rates increase linearly for 900 updates to 6×10^{-4}, followed by cosine decay to 6×10^{-5} at update 9,000. Evaluations and checkpoints occur every 1,000 updates, including a final evaluation before the checkpoint is saved. The data use the same tokenized OpenWebText stream as the nanoGPT experiments; the provided code expects uint16 token files.

Fixed resource budget and reporting. We allocated at most 24 hours on four nodes. Before launching the fresh training runs, a common horizon was selected solely from the slowest measured step time, three seeds, a 15% timing allowance and a 30-minute reserve, rounded down to 1,000 updates and capped at 10,000. This yielded 9,000 updates. All three seeds have completed for every method. Table 6 lists all three common seeds and their individual endpoints. We report the mean and sample standard deviation. No seed was excluded because of its loss.

Table 6: All three completed 9,000-step validation endpoints per method. The final column reports the mean and sample SD across the three seeds.

Optimizer	1337	1338	1339	n	Mean ± SD
AdamW (`fp32`)	2.9593	2.9549	2.9547	3	2.9563±0.0026
PowerStep (`fp32`)	2.9535	2.9500	**2.9426**	3	2.9487±0.0056
PowerStep (`int8`)	**2.9530**	**2.9436**	2.9487	3	**2.9484±0.0047**

Timing and memory. Measurements use PyTorch 2.7.1 with torch-npu, eight Ascend 910 devices per node, and the same training shapes and precision as above. Each independent process performs 20 warmup updates followed by 60 timed updates. The window includes batch construction and transfer, forward/backward passes, clipping and optimizer updates, but excludes evaluation and disk I/O. Devices synchronize at its boundaries. The maximum duration across ranks determines throughput. We reset allocator peak statistics after warmup, then report peak allocated and reserved memory and deduplicated persistent optimizer tensor storage. Reported peak allocation excludes driver/runtime allocations outside the framework allocator. State figures exclude model parameters, gradients and transient optimizer workspace. Node-wise measurements are supplied as machine-readable records.

On Medium, each method runs on all four physical nodes with rotated order. These are four hardware replicates of the same benchmark setup.

C VALIDATION RESULTS

Validation losses for dense and large-scale models are reported in Tables 7 and 1, respectively. The impact of optimizer state quantization on validation performance is examined separately in Table 8.

The matched study is reported separately in Appendix B.

Table 7: Validation losses on dense models, all states `fp32`. The lowest validation loss in each row is bold.

Model	AdamW	AdamS	SignSGD	pbSGDM	PowerStep
GPT-2-Small (124M)	2.950	2.951	**2.942**	3.377	2.949
GPT-2-Medium (350M)	**2.711**	2.725	2.736	3.110	2.717
Qwen3-0.6B	2.912	2.912	**2.897**	3.198	2.923
Qwen3-1.7B	**2.780**	2.782	2.794	3.034	2.799
Qwen3-4B	2.690	**2.672**	2.680	2.712	2.712
Qwen3-8B	**2.660**	4.689	5.045	2.853	2.688

Table 8: Validation losses under state quantization. Uniform all-layer `int8` AdamW diverged and is omitted. Dettmers `int8` reaches 2.9524 on Small and 3.4768 on Medium at 50,000 updates (Section 5.3).

Model	AdamW (`fp32`)	PowerStep (`fp32`)	PowerStep (`int8`)
GPT-2-Small (124M)	2.950	2.949	**2.948**
GPT-2-Medium (350M)	**2.711**	2.717	2.716

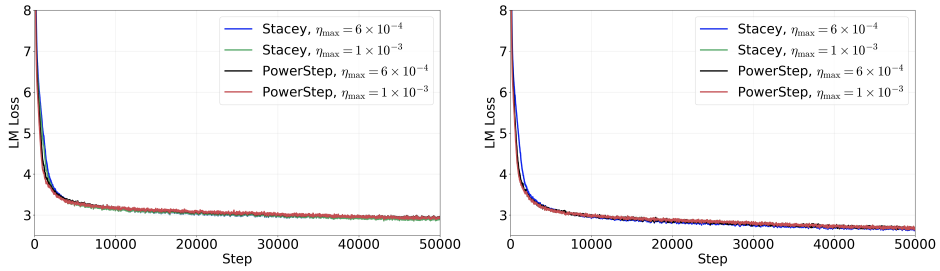
D COMPARISON TO STACEY

PowerStep can be viewed as a simplified, memory-efficient variant of Stacey-$(p, 2)$ (Luo et al., 2025) that removes the primal–dual auxiliary variables and the ϵ-stabilization term and employs heavy-ball momentum. To assess whether these simplifications incur any performance cost, we conduct a direct comparison between the two optimizers on GPT-2-Small (124M) and GPT-2-Medium (350M). For Stacey-$(p, 2)$, we adopt the hyperparameters from (Luo et al., 2025): $\alpha = 0.1$, $\beta_1 = 0.9$, $\beta_2 = 0.99$, $\tau = 0.001$, and $\epsilon = 1 \times 10^{-8}$. We set $p = 11$, corresponding to $\beta = 1/(p-1) = 0.1$, which matches PowerStep's power exponent. For learning rates, we evaluate $\eta_{\max} \in \{6 \times 10^{-4}, 1 \times 10^{-3}\}$ with $\eta_{\min} = 0.1 \cdot \eta_{\max}$ across both optimizers. Other pretraining settings follow Appendix A. Figure 9 reports the training loss trajectories. The plotted trajectories show faster early loss reduction for PowerStep and close final training losses.

The archived file labeled as Medium Stacey at $\eta_{\max} = 10^{-3}$ contains PowerStep code, so that curve's optimizer identity is not verified. Table 9 retains the independently audited validation records, which include lower losses for Stacey. The panels retain this trajectory for reference; the disputed label is not used to draw conclusions about Stacey.

Table 9: Validation losses for Stacey and PowerStep after 50,000 updates at the indicated $\eta_{\max}$.

Model	$\eta_{\max}$	Stacey	PowerStep
GPT-2-Small	1×10^{-3}	**2.9100**	2.9407
GPT-2-Medium	6×10^{-4}	**2.6918**	2.7170



(a) GPT-2-Small (124M) (b) GPT-2-Medium (350M)

Figure 9: Training loss comparison between PowerStep and Stacey-$(p, 2)$. The archived Medium Stacey label at $\eta_{\max} = 10^{-3}$ has an unresolved provenance mismatch; see the text.

E Quantization: implementation and sensitivity

We analyze how stored-state errors affect the updates under the tested quantization scheme. Both quantizers use 128-coordinate blocks. The Dettmers configuration uses signed nonuniform codes for the first moment and unsigned codes for the second, retaining embedding and small-tensor states in `fp32`. Its local codebooks differ from those in the bitsandbytes implementation. Both GPT-2 runs use $\eta_{\max} = 6 \times 10^{-4}$, 2,000 warmup updates and 50,000 updates with cosine decay. Figure 3 shows trailing 50-update averages of uninterrupted per-update training logs; evaluation-average losses are excluded.

E.1 Linear block quantization and memory accounting

Uniform quantization applies linear `int8` quantization to blocks of 128 values in every layer, including embeddings. This differs from the nonuniform dynamic recipe of Dettmers et al. (2022). A common symmetric nearest-rounding model is $Q(x_i) = s\,\mathrm{round}(x_i/s)$, with $s = \max_i |x_i|/127$ and $|Q(x_i) - x_i| \le s/2$; zero blocks are exact. The error is absolute relative to the block maximum, not a uniform coordinate-relative error. `int8` has integer codes, not floating-point mantissa bits. This rounding model abstracts from kernel-specific tie-breaking and scale-rounding conventions.

One `int8` state plus one `fp32` scale per 128 values uses $1 + 4/128 = 1.03125$ bytes per parameter, versus 8 bytes for two `fp32` moments: an idealized $\sim 8\times$ ratio, consistent with Table 1. This excludes parameters, gradients, master weights, activations, communication, padding and workspace. At equal `int8` precision, one state approximately halves the two-state payload, subject to metadata and tensor exemptions.

E.2 Error enters through the stored previous state

For $0 < \beta \le 1$, with dequantized stored buffer $\widehat{\mathbf{m}}_{t-1}$ and full-precision accumulator $\mathbf{a}_t$, the update in Section 4 is

$$\mathbf{a}_t = \gamma \widehat{\mathbf{m}}_{t-1} + \mathbf{g}_t, \qquad \widehat{\mathbf{m}}_t = Q(\mathbf{a}_t), \qquad \mathbf{u}_t = \Phi_\beta(\mathbf{a}_t). \tag{12}$$

Thus the current rounding error is stored for the next iteration; $\mathbf{u}_t$ is not computed from $Q(\mathbf{a}_t)$. To isolate state error, define a reference buffer on the *same gradient sequence* by $\mathbf{m}_t = \gamma \mathbf{m}_{t-1} + \mathbf{g}_t$. With $\mathbf{e}_t = \widehat{\mathbf{m}}_t - \mathbf{m}_t$ and $\boldsymbol{\delta}_t = Q(\mathbf{a}_t) - \mathbf{a}_t$, starting from $\widehat{\mathbf{m}}_0 = \mathbf{m}_0 = \mathbf{0}$,

$$\mathbf{e}_t = \gamma \mathbf{e}_{t-1} + \boldsymbol{\delta}_t, \qquad \mathbf{a}_t - \mathbf{m}_t = \gamma \mathbf{e}_{t-1}, \qquad \|\mathbf{e}_t\|_q \le \sum_{s=1}^{t} \gamma^{t-s} \|\boldsymbol{\delta}_s\|_q. \tag{13}$$

The signed-power transform satisfies

$$\|\Phi_\beta(\mathbf{a}_t) - \Phi_\beta(\mathbf{m}_t)\|_p \le 2^{1-\beta} \gamma^\beta \|\mathbf{e}_{t-1}\|_q^\beta. \tag{14}$$

The transform is continuous at zero, but neither globally bounded nor Lipschitz there for $\beta < 1$. Moreover, $(1/128)^{0.1} \simeq 0.616$: a Hölder bound does not prove attenuation of small absolute perturbations or preservation of coordinate signs. The same-gradient bound also does not control two training trajectories with different future gradients.

E.3 Relative sensitivity away from zero

Relative sensitivity can be quantified when a coordinate is resolved relative to its quantization error. For $x \neq 0$ and $|\delta| \leq \rho|x|$ with $0 \leq \rho < 1$, the sign is unchanged and the mean-value theorem gives

$$\frac{|\Phi_\beta(x+\delta) - \Phi_\beta(x)|}{|\Phi_\beta(x)|} \leq \beta(1-\rho)^{\beta-1}\frac{|\delta|}{|x|}. \tag{15}$$

Indeed, every point between x and $x+\delta$ has magnitude at least $(1-\rho)|x|$, and $|\Phi'_\beta(z)| = \beta|z|^{\beta-1}$. Thus the local relative condition number is β: small relative perturbations are attenuated to first order when $\beta < 1$. For example, at $\beta = 0.1$ and $\rho = 0.01$, the relative output error is at most 0.00101. This is a conditional local result, not a uniform consequence of block quantization. Coordinates near zero may have large relative errors or be rounded to zero, where only the absolute Hölder bound applies. For the implemented update, $x = m_{t,i}$ and $\delta = \gamma e_{t-1,i}$, so the condition concerns accumulated previous-state error, not just one rounding step.

E.4 Why second-moment quantization can be sensitive

For $v > 0$, the reciprocal factor $r(v) = 1/(\sqrt{v} + \epsilon)$ has derivative

$$r'(v) = -\frac{1}{2\sqrt{v}(\sqrt{v}+\epsilon)^2}. \tag{16}$$

Taylor expansion requires a small perturbation in the positive domain. Rounding v to zero instead changes the reciprocal exactly from $1/(\sqrt{v}+\epsilon)$ to $1/\epsilon$. For example, with $v = 10^{-8}$ and $\epsilon = 10^{-8}$, rounding v to zero increases this reciprocal factor by exactly $(\sqrt{v}+\epsilon)/\epsilon = 10{,}001$. This example holds the numerator fixed and concerns the reciprocal factor only; it is not a predicted full-update ratio for the implementation. The condition for ϵ to dominate is $v \ll \epsilon^2$, not $v \ll \epsilon$. A first-order expansion with $|\delta^v| \gg v$ is invalid and cannot justify a quantitative amplification estimate. The update also depends on the correlated first moment, so a small denominator alone does not prove divergence. Nonuniform codebooks and selective precision can mitigate this sensitivity (Dettmers et al., 2022; Li et al., 2023). Figure 3 contrasts the linear stress test with a local Dettmers baseline; uniform quantization failure is not evidence against 8-bit AdamW generally.

A nonuniform codebook improves resolution across magnitudes but does not guarantee small relative errors for every coordinate. Rounding errors in stored moments enter subsequent recurrences. If the second moment is underestimated without a corresponding decrease in the first, the update can be amplified. Q-Adam-mini (Han et al., 2025) provides another example of Adam-state quantization instability: it reports embedding weight-norm instability from first-moment quantization and mitigates it with stochastic rounding. This supports the need for care when quantizing Adam, but does not identify the cause of our Medium loss spikes, since embedding states remain in `fp32` in the evaluated Dettmers configuration.

Independent low-bit Adam studies identify related failure modes. Dettmers et al. (2022, Table 3) report 90% unstable runs for unblocked linear 8-bit Adam on a 209M language model, versus 0% for their block-wise dynamic scheme. These 209M percentages summarize runs across hyperparameter settings. Their nonuniform but unblocked 1.3B configuration without the stable embedding layer is unstable in all five seeds (100%); adding that layer reduces the reported rate to 80%. At 4 bits, Li et al. (2023) identify the second-moment zero-point problem: rounding small positive states to zero can greatly distort the inverse-square-root factor. Their GPT-2-Medium E2E-NLG fine-tuning ablation (Table 1) reports 66% unstable runs for B128/DE and 0% for B128/DE-0; neither row uses stable embeddings, stochastic rounding or factorization. DE-0 excludes zero from the dynamic-exponent codebook. The latter result concerns 4-bit states and supports the mechanism rather than an 8-bit failure rate. Together these studies establish that Adam-state quantization can require specific stabilization measures; they do not imply that all 8-bit Adam implementations are unstable.

Location of the observed Medium instability. After 10,000 updates, the first raw training loss above 4 occurs at completed update 15,998 (loss 4.1295). Validation rises at 16,000 updates, temporarily recovers by 19,000, and deteriorates again after 20,000. These are finite loss spikes; the run reaches its full 50,000-update budget. The implementation forms $\widetilde{\mathbf{m}}_t = \beta_1\widehat{\mathbf{m}}_{t-1} + (1-\beta_1)\mathbf{g}_t$ and $\widetilde{\mathbf{v}}_t = \beta_2\widehat{\mathbf{v}}_{t-1} + (1-\beta_2)\mathbf{g}_t^{\odot 2}$ in `fp32` and uses these tensors, with bias correction, in the current

update. Quantized copies are stored for the next iteration. Thus a second moment rounded down at one step can affect a later denominator; the current squared-gradient contribution can partly replenish it. First- and second-moment errors need not cancel in their ratio. Identifying a causal trigger would require layerwise moment-error and update-norm records around the first excursion; the available loss logs alone do not provide that localization.

F SUPPLEMENTARY EXPERIMENTS

The following studies use the configurations specified in each subsection. Results are reported separately from the matched three-seed study in Appendix B.

F.1 MULTIPLE SEEDS

AdamW and PowerStep were evaluated at the main-study $\eta_{\max} = 6 \times 10^{-4}$ for 50,000 steps. Table 10 reports the means, sample standard deviations and standard errors of the mean (SEM). Small PowerStep endpoints are 2.9632, 2.9625, 2.9465, 2.9550 and 2.9486; Small AdamW endpoints are 2.9525, 2.9628, 2.9528, 2.9472 and 2.9503. Medium PowerStep endpoints are 2.7143, 2.7189, 2.7342, 2.7180 and 2.7170; Medium AdamW endpoints are 2.7494, 6.7695, 2.7180, 6.7551 and 2.7105. Four runs in each group have named seed records, while one is matched to an original archived run. Two near-identical Medium original-run candidates leave a minor provenance ambiguity that does not change the rounded mean/SEM. Thus these historical aggregates are weaker evidence than the new explicitly paired experiment. The high-loss AdamW runs show instability under that recipe, not an isolated causal effect of the second moment; lower learning rates and implementation differences could change the result.

Table 10: Five-run aggregates. Uncertainty is SEM ($s/\sqrt{5}$), not SD. High-loss counts use the descriptive threshold final validation loss > 4; Medium AdamW includes both high-loss endpoints.

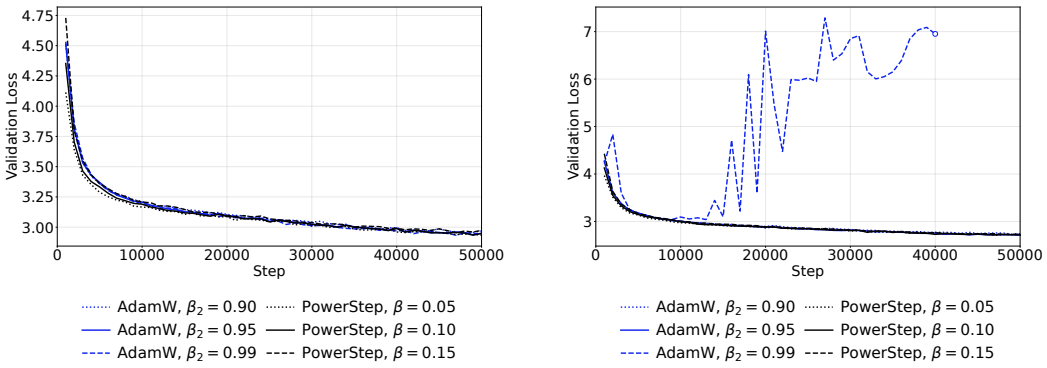
Model	Optimizer	Mean ± SEM	Sample SD	High-loss runs
Small	AdamW	**2.9531±0.0026**	0.0059	0/5
	PowerStep	2.9552±0.0034	0.0077	0/5
Medium	AdamW	4.3405±0.9887	2.2108	2/5
	PowerStep	**2.7205±0.0035**	0.0079	0/5

F.2 POWERSTEP EXPONENT AND ADAMW SECOND-MOMENT SWEEP

Figure 10 and Table 11 vary AdamW's second-moment coefficient β_2 and PowerStep's exponent β, holding $\eta_{\max} = 6 \times 10^{-4}$ fixed. The default rows reuse the main-study baselines. All runs reach 50,000 updates except Medium AdamW with $\beta_2 = 0.99$, whose last recorded evaluation is at 40,000 updates. PowerStep remains stable across the tested exponents, while AdamW with $\beta_2 = 0.99$ becomes unstable on Medium. We exclude two PowerStep endpoints from the fixed-rate sweep because they use different $\eta_{\max}$: 3.0163 on Small at 10^{-2} and 2.7096 on Medium at 10^{-3}. The unmatched Medium AdamW summary value 2.7258 is also excluded.

Table 11: Parameter-sweep validation losses at $\eta_{\max} = 6 \times 10^{-4}$. Baseline rows reuse the default configuration.

Optimizer	Parameter	Small	Medium
AdamW	$\beta_2 = 0.90$	2.9686	2.7427
AdamW	$\beta_2 = 0.95$ (baseline)	2.9503	**2.7105**
AdamW	$\beta_2 = 0.99$	2.9397	(collapse)
PowerStep	$\beta = 0.05$	**2.9350**	2.7291
PowerStep	$\beta = 0.10$ (baseline)	2.9486	2.7170
PowerStep	$\beta = 0.15$	2.9740	2.7332



(a) GPT-2-Small (b) GPT-2-Medium

Figure 10: Validation curves for the AdamW second-moment coefficient β_2 and PowerStep exponent β at $\eta_{\max} = 6 \times 10^{-4}$. Solid curves use the default settings; dotted and dashed curves vary one coefficient. The open circle marks the last recorded evaluation of the unstable Medium AdamW run.

F.3 Matrix-update hybrids

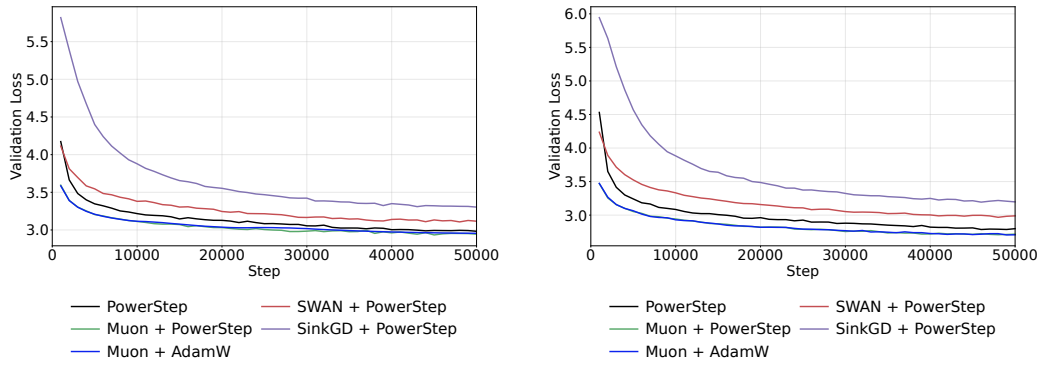
We evaluate Muon (Jordan, 2024), SWAN (Ma et al., 2025) and SinkGD (Scetbon et al., 2025) on hidden matrix parameters, using PowerStep for remaining parameters, and compare Muon–PowerStep with Muon–AdamW. Figure 11 shows validation curves, and Table 12 gives final validation losses at 50,000 steps. For Muon, the matrix group is exactly the two-dimensional parameters in the Transformer blocks; all remaining parameters, including tied embeddings/output weights and vector/scalar parameters, form the auxiliary group. Both groups use cosine decay from $\eta_{\max} = 6 \times 10^{-4}$ to $\eta_{\min} = 6 \times 10^{-5}$ without warmup. Muon uses momentum 0.95, Nesterov updates, five Newton–Schulz steps, and update scaling by the square root of the larger matrix dimension; the Muon implementation applies no weight decay. Auxiliary weight decay is 0.1. PowerStep uses $(\gamma, \beta) = (0.9, 0.1)$, while AdamW uses $(\beta_1, \beta_2) = (0.9, 0.95)$. The earlier study uses Small/Medium seeds 2400/2401 for Muon–PowerStep and 2402/2403 for Muon–AdamW. Validation occurs every 1,000 updates. These settings describe the Muon runs; the SWAN and SinkGD results use their separately recorded configurations. The Small Muon–AdamW ablation uses seeds 2400 and 2402, each with two schedules for η_t. The schedules share initialization and data order within each seed. Fixed validation uses 200 batches with seed 872401 and preserves the training RNG. At 50,000 updates, the shared schedule gives 3.0353 and 2.9558; scaling only the Muon matrix-group η_t by 0.2 gives 2.9337 and 2.9372 (mean 2.9354, sample SD 0.0025). The auxiliary AdamW retains $\eta_{\max} = 6 \times 10^{-4}$ and $\eta_{\min} = 6 \times 10^{-5}$. Figure 7 shows the seed-2402 trajectory with the shared schedule (2.9558); the hybrid table reports the mean of both runs with the shared schedule, 2.9956. All four trajectories are supplied.

Table 12: Hybrid validation losses. Small Muon–AdamW reports the mean of two runs with the shared schedule. The first two columns specify the hidden-matrix and remaining-parameter updates. The all-PowerStep row uses the hybrid-study recipe.

Hidden matrices	Remaining parameters	GPT-2-Small	GPT-2-Medium
PowerStep	PowerStep	2.9825	2.7979
SWAN	PowerStep	3.1162	2.9890
SinkGD	PowerStep	3.3035	3.1973
Muon	PowerStep	**2.9438**	2.7159
Muon	AdamW	2.9956	**2.7065**

Muon–PowerStep improves over the all-PowerStep row in both models. On Small, Muon–AdamW averages 2.9956 with fixed validation samples, while Muon–PowerStep reaches 2.9438 under the earlier evaluation protocol. On Medium, Muon–AdamW is lower by 0.0094. SWAN and SinkGD with PowerStep have higher losses than the Muon hybrids under the tested configurations.

Replacing AdamW by PowerStep only for the remaining d_r parameters saves $4d_r$ bytes of `fp32` moment storage. If Muon retains one `fp32` buffer for d_m hidden-matrix parameters, the two hybrid state payloads are $4d_m + 8d_r$ and $4d_m + 4d_r$ bytes. Consequently, this substitution does not generally halve the *whole hybrid's* optimizer memory. These are buffer counts, not peak-memory measurements; SWAN/SinkGD hybrids likewise retain the PowerStep state on the remaining group even if their matrix update is stateless.



(a) GPT-2-Small (b) GPT-2-Medium

Figure 11: Validation curves for matrix-update hybrids. PowerStep denotes the all-PowerStep configuration; other labels give the matrix optimizer and auxiliary optimizer. The Muon + PowerStep and Muon + AdamW curves largely overlap. The Small Muon–AdamW curve uses the same run as Figure 7.

F.4 Supervised fine-tuning

Qwen3-4B-Base was fine-tuned with AdamW or PowerStep on a filtered high-quality, no-tool Nemotron-Math-v2 subset (NVIDIA, 2026), selecting responses with boxed answers and lengths between 1,000 and 4,000 tokens. The recorded training scripts use 14,200 updates, global batch 128, microbatch 1, context length 6,144, tensor parallelism 4, eight nodes with eight devices each, `bf16`, seed 42, $\eta_{\max} = 10^{-5}$, $\eta_{\min} = 10^{-6}$, weight decay 0.1 and gradient clipping at 1.0. AdamW uses $(0.9, 0.95)$ for its moment coefficients. Both scripts load the same base checkpoint and do not load an earlier optimizer state. Final training losses are 0.0731 and 0.0612 for AdamW and PowerStep, respectively; the final 100-update means are 0.1053 and 0.0932. The schedule is linear with no warmup.

The archived generation script samples with temperature 0.6, top-$p = 0.95$ and top-$k = 40$, with context length at most 32,768 and at most the remaining context length in generated tokens. Four nodes with four generation workers each produce 16 responses per question. For each model and AIME year, the raw shards contain exactly 30 questions and 480 unique question/sample pairs, without missing questions, duplicate pairs or malformed JSON records. Sampling-seed provenance and the full dataset revision/decontamination procedure remain unavailable.

We re-score both optimizers with one explicit rule: extract the *last* boxed expression, accept it only if it is an integer (allowing whitespace and comma separators), and compare with the integer reference answer. Missing or unparseable answers count as incorrect. For N questions with c_i correct responses among 16 samples, we estimate Pass@1 as $N^{-1} \sum_{i=1}^{N} c_i/16$. Since each question has the same number of samples, this equals the fraction of correct responses among all $16N$ responses, including invalid ones in the denominator.

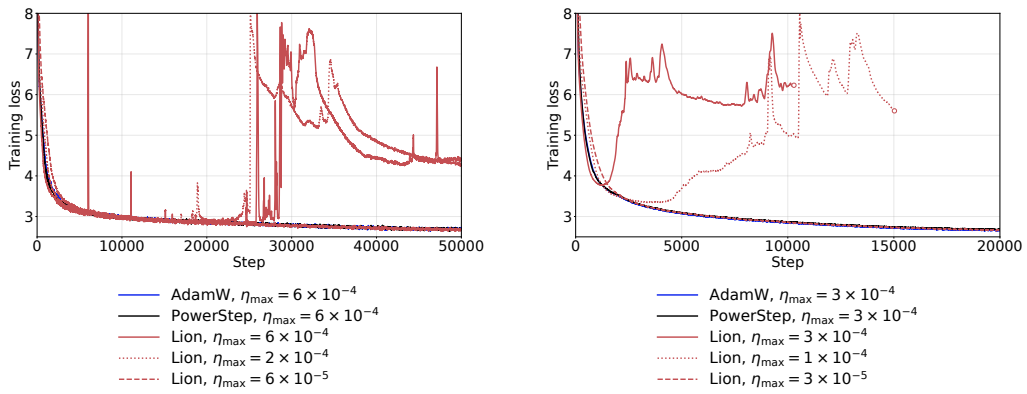
The AMC23 evaluation contains 40 questions and 640 responses per SFT checkpoint, with sample indices 0–15 for every question and no duplicate or conflicting records. We apply the same last-boxed-integer rule: AdamW has 359 correct responses and 34 questions with at least one correct response, versus 367 and 35 for PowerStep. There are 91 and 84 unparseable responses, respectively. The archived AMC23 set is a 40-question subset, not the full contest collection. The launch script

uses the same final SFT checkpoints and decoding settings. The Pass@1 differences are small, and contamination controls remain unverified. This independent SFT experiment does not evaluate the pretraining checkpoints in Section 5.

F.5 Comparison with Lion

Lion (Chen et al., 2023) also uses one momentum buffer, but computes a sign update with separate momentum coefficients for the update direction and stored state. Figure 12 compares Lion at three values of $\eta_{\max}$ with AdamW and PowerStep at their main-study settings. For GPT-2-Medium, Lion uses $\eta_{\max} \in \{6 \times 10^{-5}, 2 \times 10^{-4}, 6 \times 10^{-4}\}$; for Qwen3-8B, it uses $\eta_{\max} \in \{3 \times 10^{-5}, 10^{-4}, 3 \times 10^{-4}\}$. The lowest value follows Lion's recommendation to reduce the learning rate relative to AdamW, together with the associated optimizer settings (Chen et al., 2023).

At the recommended lower $\eta_{\max}$, Lion achieves validation loss close to AdamW and PowerStep on both models. Using the main-study $\eta_{\max}$ makes Lion unstable; the intermediate value also produces later instability despite faster initial progress on Medium. Lion becomes unstable at learning rates where AdamW and PowerStep remain stable in these runs; Appendix F.6 examines the step-size mechanism. Lion requires a smaller $\eta_{\max}$, which slows its initial progress compared with AdamW and PowerStep at their main-study settings.



(a) GPT-2-Medium (b) Qwen3-8B

Figure 12: Training curves for Lion at three values of $\eta_{\max}$, compared with AdamW and PowerStep at the main-study settings: $\eta_{\max} = 6 \times 10^{-4}$ on GPT-2-Medium and 3×10^{-4} on Qwen3-8B. Curves show trailing 50-update averages. Colors identify optimizers; line styles distinguish Lion configurations. All GPT-2-Medium runs reach 50,000 updates. Open circles mark the recorded Qwen3-8B Lion endpoints: 10,291 updates at 3×10^{-4} (runtime interruption) and 15,027 at 10^{-4} (partial trace).

F.6 Step-size sensitivity of sign and signed-power updates

Both sign updates and PowerStep use the sign operator to choose direction. PowerStep additionally retains magnitude information. For a nonzero scalar momentum m, their step magnitudes are η_t and $\eta_t |m|^{\beta}$, respectively. A sign update takes a full coordinate step even when the momentum is arbitrarily small. PowerStep's step vanishes as $m \to 0$ for every $\beta > 0$. Changing the momentum from $+\delta$ to $-\delta$, with $\delta > 0$, changes the update by

$$2\eta_t \quad \text{for sign,} \qquad 2\eta_t \delta^{\beta} \quad \text{for signed power.} \tag{17}$$

Although $\operatorname{sign}(m)$ is discontinuous at zero, its product with $|m|^{\beta}$ is continuous there. This reduces the update jump caused by sign changes of small momentum coordinates. At $\beta = 0.1$, however, the attenuation is slow: $(10^{-6})^{0.1} \approx 0.251$.

A scalar quadratic isolates the overshoot mechanism. Let $f(x) = ax^2/2$ with $a > 0$, and consider exact gradients without momentum or weight decay. For $x_t \neq 0$, sign descent gives $x_{t+1} =$

$x_t - \eta_t \operatorname{sign}(x_t)$, so

$$f(x_{t+1}) - f(x_t) = -a\eta_t|x_t| + \tfrac{1}{2}a\eta_t^2 < 0 \quad \Longleftrightarrow \quad 0 < \eta_t < 2|x_t|. \tag{18}$$

The descent threshold shrinks near the minimum, while the step magnitude remains fixed. With a constant positive step size, the iterates generally enter a two-point cycle around zero unless they reach zero exactly. For the signed-power update $x_{t+1} = x_t - \eta_t \operatorname{sign}(x_t)|ax_t|^\beta$, the same calculation yields

$$f(x_{t+1}) - f(x_t) = -\eta_t a^{1+\beta}|x_t|^{1+\beta} + \tfrac{1}{2}\eta_t^2 a^{1+2\beta}|x_t|^{2\beta}, \tag{19}$$

$$f(x_{t+1}) < f(x_t) \quad \Longleftrightarrow \quad 0 < \eta_t a^\beta < 2|x_t|^{1-\beta}. \tag{20}$$

Signed power lets the absolute step shrink with the gradient, but for $0 < \beta < 1$ a fixed step size can still produce oscillation near the minimum. Momentum can also lag behind the current gradient. Thus PowerStep softens the fixed-step and zero-crossing mechanisms rather than eliminating overshoot or guaranteeing stability.

Chen et al. (2023) likewise attribute Lion's smaller recommended learning rate to the larger update norm produced by the sign operation. This scalar analysis explains why retaining magnitude information can help, but does not establish a universal stability ordering against AdamW, whose normalized update can also become sign-like. Requiring a smaller learning rate is distinct from having a narrower stable learning-rate range; Figure 12 directly illustrates the former under the tested configurations.

G COMPLETE CONVERGENCE PROOFS

Throughout this appendix, $0 < \beta \le 1$, $q = 1 + \beta$, $p = q/\beta$, $h = 1 - \gamma$, and the assumptions are those of Section 3. All vector norms below are explicitly indicated. The update is exact, with no weight decay, clipping, or quantization. Expectations refer to the stochastic oracle, and the horizon-dependent step size is constant during each run. Write $\bar{\mathbf{g}}_t = \nabla f(\boldsymbol{\theta}_{t-1})$, $\mathbf{u}_t = \Phi_\beta(\mathbf{m}_t)$, and

$$\bar{\mathbf{G}}_t = \sum_{k=0}^{t-1} \gamma^k \bar{\mathbf{g}}_{t-k}, \qquad \mathbf{N}_t = \sum_{k=0}^{t-1} \gamma^k \boldsymbol{\xi}_{t-k}, \qquad \mathbf{m}_t = \bar{\mathbf{G}}_t + \mathbf{N}_t, \qquad \mathbf{r}_t = \bar{\mathbf{g}}_t - h\bar{\mathbf{G}}_t. \tag{21}$$

Here $\bar{\mathbf{G}}_t$ is an accumulated true-gradient component, not a conditional expectation of $\mathbf{m}_t$ at the current iterate. Past noise is correlated with later iterates; no independence between $\mathbf{N}_t$ and $\bar{\mathbf{G}}_t$ is assumed.

Explicit constants in Theorem 1. Let $\Delta_0 = f(\boldsymbol{\theta}_0) - f^*$. Define

$$\begin{aligned} M &= \left(\frac{G+\sigma}{h}\right)^{2\beta}, & D &= \frac{L_p\sqrt{M}\,\gamma}{h}, \\ A &= 2^{-q}h^{-\beta}, & B &= \sqrt{M} + \tfrac{1}{2}h^{-\beta}(2G)^\beta, \\ K_\beta &= \frac{\beta}{2}\left(\frac{2^{2-\beta}}{q}\right)^{q/\beta}. \end{aligned} \tag{22}$$

Theorem 1 uses

$$C_1 = \frac{1}{A}\left[\frac{\Delta_0}{c} + c\left(\frac{L_p M}{2} + BD\right)\right], \qquad C_2 = \frac{BG\gamma}{Ah}, \qquad C_\beta = 2^q K_\beta. \tag{23}$$

Thus $R_T = C_1/\sqrt{T} + C_2/T$ retains the initialization term and all parameter dependencies.

Proof idea and scope. Write $\mathbf{m}_t = \bar{\mathbf{G}}_t + \mathbf{N}_t$, where $\bar{\mathbf{G}}_t = \sum_{k=0}^{t-1}\gamma^k\bar{\mathbf{g}}_{t-k}$ and $\mathbf{N}_t = \sum_{k=0}^{t-1}\gamma^k\boldsymbol{\xi}_{t-k}$. Align the update through $\bar{\mathbf{G}}_t$ and control $\bar{\mathbf{g}}_t - h\bar{\mathbf{G}}_t$ in mean square. This makes the drift contribution linear in η, rather than η^β. A coordinatewise Hölder bound and Young's inequality control the noise without removing its residual. The following lemmas account for initialization and give explicit constants. The constant horizon-dependent step avoids the logarithm in $\sum_t (c/\sqrt{t})^2$; it differs from the experimental warmup/cosine schedule.

When $\sigma = 0$, the bound gives $O(T^{-1/2})$ stationarity. A vanishing average noise moment also removes the residual. The corollaries below state sufficient conditions for growing batches and a small relative-noise condition. These are additional assumptions, not properties established for the language-model experiments. We make no minimax-optimality claim: a neighborhood bound is weaker than exact stationarity, and increasing the batch changes the number of oracle calls.

Dimension and the choice of β. There is no explicit d in (11) when L_p, G, σ are fixed in the native norms. This does not imply dimension independence at fixed Euclidean constants. If those constants are L_2, G_2, σ_2, valid choices are

$$L_p = d^{(1-\beta)/(1+\beta)}L_2, \quad G = d^{1/q-1/2}G_2, \quad \sigma^q = d^{1-q/2}\sigma_2^q. \tag{24}$$

For example, at $\beta = 0.1$ and $d = 10^5$, the Euclidean-to-native conversion gives $L_p = d^{9/11}L_2 \approx 1.23 \times 10^4 L_2$. This factor concerns conversion from a fixed Euclidean smoothness constant, not an additional factor when native-norm constants are assumed directly. At the experimental exponent $\beta = 0.1$, $K_\beta \approx 3.429 \times 10^4$ and $C_\beta = 2^{1.1}K_\beta \approx 7.350 \times 10^4$. The resulting noise-residual bound is therefore a qualitative guarantee, not a quantitative prediction of training performance. More generally, C_β becomes loose as $\beta \to 0$; the theorem neither covers $\beta = 0$ nor predicts the empirical preference for $\beta = 0.1$. Even at $\beta = 1$, the residual in this generic bound need not be tight. Removing bounded gradients and proving sharper noise-dependent guarantees remain open here.

G.1 TRANSFORM AND MOMENT BOUNDS

Lemma 1 (Dual geometry and scalar perturbations). *For* $\mathbf{x}, \mathbf{y}, \mathbf{m} \in \mathbb{R}^d$,

$$\langle \mathbf{m}, \Phi_\beta(\mathbf{m}) \rangle = \|\mathbf{m}\|_q^q, \quad \|\Phi_\beta(\mathbf{m})\|_p = \|\mathbf{m}\|_q^\beta, \quad \|\Phi_\beta(\mathbf{x}) - \Phi_\beta(\mathbf{y})\|_p \le 2^{1-\beta}\|\mathbf{x} - \mathbf{y}\|_q^\beta. \tag{25}$$

Moreover, for real a, n,

$$2^{1-\beta}|a||n|^\beta \le \tfrac{1}{2}|a|^q + K_\beta |n|^q, \qquad K_\beta = \frac{\beta}{2}\left(\frac{2^{2-\beta}}{q}\right)^{q/\beta}. \tag{26}$$

Proof. The identities follow coordinatewise from $\beta p = q$. For same-sign scalars, subadditivity of z^β gives $|\Phi_\beta(x) - \Phi_\beta(y)| \le |x - y|^\beta$. For opposite signs, concavity gives $|x|^\beta + |y|^\beta \le 2^{1-\beta}(|x| + |y|)^\beta$. Raising the scalar bound to p, summing, and taking the p-th root proves the vector bound. Finally maximize $2^{1-\beta}an^\beta - a^q/2$ over $a \ge 0$ for fixed $n \ge 0$. For $n > 0$ the maximizer is $a = (2^{2-\beta}/q)^{1/\beta} n$, giving (26); the zero cases are immediate. □

Lemma 2 (Uniform update moment and accumulated noise). *Let* $\nu_t = \mathbb{E}\|\boldsymbol{\xi}_t\|_q^q \le \sigma^q$. *Then*

$$\mathbb{E}\|\mathbf{u}_t\|_p^2 \le M = \left(\frac{G + \sigma}{h}\right)^{2\beta}, \tag{27}$$

$$\mathbb{E}\|\mathbf{N}_t\|_q^q \le h^{1-q} \sum_{k=0}^{t-1} \gamma^k \nu_{t-k}, \qquad \sum_{t=1}^{T} \mathbb{E}\|\mathbf{N}_t\|_q^q \le h^{-q} \sum_{t=1}^{T} \nu_t. \tag{28}$$

Proof. Minkowski's inequality in L^q gives $(\mathbb{E}\|\mathbf{g}_t\|_q^q)^{1/q} \le G + \sigma$ and $(\mathbb{E}\|\mathbf{m}_t\|_q^q)^{1/q} \le \sum_{k=0}^{t-1} \gamma^k (G + \sigma) \le (G + \sigma)/h$. Since $2\beta/q \le 1$, Jensen yields $\mathbb{E}\|\mathbf{u}_t\|_p^2 = \mathbb{E}(\|\mathbf{m}_t\|_q^q)^{2\beta/q} \le (\mathbb{E}\|\mathbf{m}_t\|_q^q)^{2\beta/q} \le M$. This does not require a second moment of the untransformed noise. For nonnegative weights w_k, convexity gives $\|\sum_k w_k \mathbf{x}_k\|_q^q \le (\sum_k w_k)^{q-1} \sum_k w_k \|\mathbf{x}_k\|_q^q$. Set $w_k = \gamma^k$ and take expectations. Summing and interchanging the finite sums bounds the remaining geometric sum by $1/h$. No orthogonality or independence of the noise terms is used. Neither this estimate nor the subsequent alignment argument requires a zero conditional mean; oracle bias is included in the noise moment and its residual. □

G.2 DRIFT AND ALIGNMENT

Lemma 3 (Initialization and gradient drift). *With constant step size* η, *let* $D = L_p\sqrt{M}\gamma/h$. *Then*

$$\left(\mathbb{E}\|\mathbf{r}_t\|_q^2\right)^{1/2} \le G\gamma^t + D\eta =: R_t^{\mathrm{drift}}, \qquad \|\mathbf{r}_t\|_q \le 2G \text{ almost surely}. \tag{29}$$

Throughout, R_t^{drift} *denotes this deterministic RMS upper bound; it also bounds* $\mathbb{E}\|\mathbf{r}_t\|_q$ *by Cauchy–Schwarz. Consequently,*

$$\mathbb{E}[\|\mathbf{r}_t\|_q \|\mathbf{u}_t\|_p] \le \sqrt{M} R_t^{\mathrm{drift}}, \qquad \mathbb{E}\|\mathbf{r}_t\|_q^q \le (2G)^{q-1} R_t^{\mathrm{drift}}. \tag{30}$$

Proof. From (21),

$$\mathbf{r}_t = \gamma^t \bar{\mathbf{g}}_t + h \sum_{k=1}^{t-1} \gamma^k (\bar{\mathbf{g}}_t - \bar{\mathbf{g}}_{t-k}). \tag{31}$$

The first term is bounded by $G\gamma^t$. By the gradient Lipschitz assumption, $\|\bar{\mathbf{g}}_t - \bar{\mathbf{g}}_{t-k}\|_q \le L_p \eta \sum_{j=t-k}^{t-1} \|\mathbf{u}_j\|_p$. Minkowski's inequality in L^2 and (27) therefore give

$$(\mathbb{E}\|\mathbf{r}_t\|_q^2)^{1/2} \le G\gamma^t + hL_p\eta\sqrt{M} \sum_{k=1}^{t-1} k\gamma^k \le G\gamma^t + \frac{L_p\eta\sqrt{M}\gamma}{h}. \tag{32}$$

Also $\|\bar{\mathbf{g}}_t\|_q \le G$ and $h\|\bar{\mathbf{G}}_t\|_q \le G(1 - \gamma^t)$, proving the pointwise $2G$ bound. Cauchy–Schwarz gives the first product bound without assuming independence. The second uses $\|\mathbf{r}_t\|_q^q \le (2G)^{q-1}\|\mathbf{r}_t\|_q$ followed by $\mathbb{E}\|\mathbf{r}_t\|_q \le (\mathbb{E}\|\mathbf{r}_t\|_q^2)^{1/2} \le R_t^{\mathrm{drift}}$. In particular, we never substitute the moment bound M as a pointwise bound on $\|\mathbf{u}_t\|_p^2$. □

Lemma 4 (Alignment with the current gradient)**.** *Let A, B be defined in* (22)*. Then*

$$\mathbb{E}\langle \bar{\mathbf{g}}_t, \mathbf{u}_t \rangle \geq A\mathbb{E}\|\bar{\mathbf{g}}_t\|_q^q - hK_\beta \mathbb{E}\|\mathbf{N}_t\|_q^q - BR_t^{\text{drift}}. \tag{33}$$

Proof. The scalar perturbation bound followed by (26) gives

$$\begin{aligned}\langle \bar{\mathbf{G}}_t, \Phi_\beta(\bar{\mathbf{G}}_t + \mathbf{N}_t)\rangle &\geq \|\bar{\mathbf{G}}_t\|_q^q - 2^{1-\beta}\sum_i |\bar{G}_{t,i}||N_{t,i}|^\beta \\ &\geq \tfrac{1}{2}\|\bar{\mathbf{G}}_t\|_q^q - K_\beta\|\mathbf{N}_t\|_q^q. \end{aligned}\tag{34}$$

Since $\bar{\mathbf{g}}_t = h\bar{\mathbf{G}}_t + \mathbf{r}_t$, dual-pair Hölder and Lemma 3 imply

$$\mathbb{E}\langle \bar{\mathbf{g}}_t, \mathbf{u}_t \rangle \geq \tfrac{h}{2}\mathbb{E}\|\bar{\mathbf{G}}_t\|_q^q - hK_\beta \mathbb{E}\|\mathbf{N}_t\|_q^q - \sqrt{M}R_t^{\text{drift}}. \tag{35}$$

Finally $\|\bar{\mathbf{g}}_t\|_q^q \leq 2^{q-1}(h^q\|\bar{\mathbf{G}}_t\|_q^q + \|\mathbf{r}_t\|_q^q)$, hence

$$\tfrac{h}{2}\mathbb{E}\|\bar{\mathbf{G}}_t\|_q^q \geq 2^{-q}h^{1-q}\mathbb{E}\|\bar{\mathbf{g}}_t\|_q^q - \tfrac{1}{2}h^{1-q}(2G)^{q-1}R_t^{\text{drift}}. \tag{36}$$

Substituting $q - 1 = \beta$ proves the result. □

G.3 Proof of Theorem 1

Integrating the gradient Lipschitz assumption along a line segment gives $f(\mathbf{y}) \leq f(\mathbf{x}) + \langle \nabla f(\mathbf{x}), \mathbf{y} - \mathbf{x}\rangle + (L_p/2)\|\mathbf{y} - \mathbf{x}\|_p^2$. The update and the moment bound imply

$$\mathbb{E}f(\boldsymbol{\theta}_t) \leq \mathbb{E}f(\boldsymbol{\theta}_{t-1}) - \eta\mathbb{E}\langle \bar{\mathbf{g}}_t, \mathbf{u}_t\rangle + \tfrac{L_pM}{2}\eta^2. \tag{37}$$

The bounded-gradient assumption and finite update moments ensure integrability at each finite time. Summing and using $f \geq f^*$ yields

$$\eta\sum_{t=1}^T \mathbb{E}\langle \bar{\mathbf{g}}_t, \mathbf{u}_t\rangle \leq \Delta_0 + \tfrac{L_pM}{2}T\eta^2. \tag{38}$$

Insert (33), use (28), and note $\sum_{t=1}^T R_t^{\text{drift}} \leq TD\eta + G\gamma/h$:

$$A\frac{1}{T}\sum_{t=1}^T \mathbb{E}\|\bar{\mathbf{g}}_t\|_q^q \leq \frac{\Delta_0}{\eta T} + \left(\tfrac{L_pM}{2} + BD\right)\eta + \frac{BG\gamma}{hT} + K_\beta h^{1-q}\frac{1}{T}\sum_{t=1}^T \nu_t. \tag{39}$$

Since $h^{1-q}/A = 2^q$, substituting $\eta = c/\sqrt{T}$ gives (11). For the squared Euclidean gradient, $|\bar{g}_{t,i}| \leq G$ implies pointwise

$$\|\bar{\mathbf{g}}_t\|_2^2 = \sum_i |\bar{g}_{t,i}|^2 \leq G^{2-q}\sum_i |\bar{g}_{t,i}|^q = G^{1-\beta}\|\bar{\mathbf{g}}_t\|_q^q. \tag{40}$$

Take expectations and use that the minimum is at most the average. This proves the theorem. □

G.4 Conditions that remove the residual

Corollary 1 (Deterministic and vanishing noise)**.** *In Theorem 1, $\sigma = 0$ gives $O(T^{-1/2})$ for the average squared gradient. More generally, if $\nu_t \leq \bar{\nu}_t \leq \sigma^q$ for a fixed deterministic sequence with $T^{-1}\sum_{t=1}^T \bar{\nu}_t \to 0$, the bound tends to zero as the horizon grows. This is a guarantee for a family of horizon-dependent runs, not an almost-sure convergence claim for a single infinite run.*

Proof. Use the actual moments ν_t in (11); the uniform envelope keeps M, D, B fixed. Both terms on the right tend to zero. □

Corollary 2 (Relative-noise condition)**.** *Replace the noise assumption by $\mathbb{E}[\|\boldsymbol{\xi}_t\|_q^q \mid \mathcal{F}_{t-1}] \leq \rho^q\|\bar{\mathbf{g}}_t\|_q^q$. If $2^qK_\beta\rho^q \leq 1/2$, then*

$$\frac{1}{T}\sum_{t=1}^T \mathbb{E}\|\bar{\mathbf{g}}_t\|_q^q \leq 2R_T = O(T^{-1/2}), \tag{41}$$

where R_T uses $\sigma = \rho G$. The same rate holds for the average squared Euclidean gradient after multiplication by $G^{1-\beta}$.

Proof. Bounded gradients imply the uniform envelope $\sigma = \rho G$, so every lemma still applies. In (11), substitute $\nu_t \le \rho^q \mathbb{E}\|\bar{\mathbf{g}}_t\|_q^q$ and move this term to the left. The assumed coefficient is at most $1/2$. □

This is an explicit, restrictive sufficient condition, especially for small β. It is not implied by unbiasedness or ordinary bounded variance, and it has not been checked in our experiments.

Corollary 3 (Growing independent mini-batches)**.** *Suppose the oracle averages b_t conditionally independent, unbiased per-sample gradients with conditional Euclidean noise variance at most v^2. In addition to Assumption 1, this gives*

$$\nu_t \le d^{1-q/2} v^q b_t^{-q/2}. \tag{42}$$

Any $b_t \to \infty$ removes the residual in the horizon limit. If $b_t = \lceil at \rceil$ for $a > 0$, the squared-gradient bound is $O(T^{-1/2})$ for every fixed $\beta > 0$.

Proof. Conditional independence gives $\mathbb{E}[\|\boldsymbol{\xi}_t\|_2^2 \mid \mathcal{F}_{t-1}] \le v^2/b_t$. The norm inequality $\|\mathbf{x}\|_q \le d^{1/q-1/2}\|\mathbf{x}\|_2$ and concavity of $z^{q/2}$ give the stated moment bound. Cesàro averaging proves the first assertion. For $1 < q < 2$, $T^{-1}\sum_{t=1}^{T} t^{-q/2} = O(T^{-q/2}) = O(T^{-1/2})$; for $q = 2$, it is $O(\log(T)/T) = O(T^{-1/2})$. Apply Theorem 1 with the uniform envelope $\sigma^q = d^{1-q/2}v^q$. □

A mere finite q-moment does not imply the $b_t^{-q/2}$ scaling; the per-sample second-moment assumption above supplies it. Linear batch growth uses $N = \Theta(T^2)$ stochastic gradient evaluations, yielding only $O(N^{-1/4})$ for the squared-gradient bound derived here. We do not call this sample-optimal.

G.5 Interpretation and limits

For $q \le 2 \le p$, norm equivalence gives the valid conversions in (24): applying $\|\mathbf{z}\|_q \le d^{1/q-1/2}\|\mathbf{z}\|_2$ to gradient differences and $\|\mathbf{x}-\mathbf{y}\|_2 \le d^{1/2-1/p}\|\mathbf{x}-\mathbf{y}\|_p$ to displacements yields $L_p = d^{1/q-1/p}L_2$. The bounded-gradient and noise conversions follow in the same way, with Jensen for the noise. Thus native notation removes explicit conversion factors from the proof, but does not remove possible dimensional scaling of the assumptions.

The residual is an upper-bound term, not a matching lower bound or a claim that every trajectory stalls. Unbiased noise alone does not commute with the nonlinear transform: for example, a scalar noise taking 2 with probability $1/3$ and -1 with probability $2/3$ has zero mean but $\mathbb{E}\Phi_\beta(\xi) = (2^\beta - 2)/3 \ne 0$ for $\beta < 1$. Additional distributional or variance-reduction assumptions therefore matter. This example explains why exact alignment cannot simply be inferred from unbiasedness; it is not a counterexample to every fixed-momentum run.

Our proof uses geometric-sum convexity, not a second-moment martingale identity for $q < 2$. It also retains the initialization term $G\gamma^t$. It supplies neither a guarantee for quantization, clipping, and decoupled weight decay nor a proof for scheduled momentum or the practical cosine schedule. Those extensions require separate analyses.