AERO at 1x C vs AdamW at 2x C: SlimPajama-6B & OpenWebMath

130M-parameter LLaMA-style pretraining, single RTX 5090 per run. 2026-06-05.

These tests extend the 1x-Chinchilla study by giving **AdamW twice the compute** (2x C = 278k steps, ~4.55B tokens) and asking the decisive question: **does AERO at 1x C still beat AdamW at 2x C?** Everything else (model, data, tokenizer, batch, cosine schedule, seed) is held fixed; only the optimizer and the compute budget change.

> **Headline.** Yes. On SlimPajama, **AERO at 1x C (3.259) beats AdamW at 2x C (3.385) by 0.126 nat while using half the compute.** On OpenWebMath **AERO at 1x C (2.205) also beats AdamW at 2x C (2.264) by 0.059 nat.** Doubling AdamW's budget is not enough to catch AERO at 1x C on either corpus.

Dataset	AERO 1x C	AdamW 1x C	AdamW 2x C	AERO 1x C vs AdamW 2x C
SlimPajama-6B	**3.259**	3.493	3.385	**AERO better by 0.126 nat, at 1/2 compute**
OpenWebMath	**2.205**	2.372	2.264	**AERO better by 0.059 nat, at 1/2 compute**

1. Datasets

All three corpora use the **same GPT-2 BPE tokenizer** and identical packing, so the only thing that changes between studies is the *content*.

1.1 C4-en (baseline reference, from prior 1.3B work)

The Colossal Clean Crawled Corpus (English): one Common Crawl snapshot put through heavy cleaning — English-only language ID, deduplication, bad-word/boilerplate removal, and dropping of short or non-prose lines. The result is **homogeneous, clean web prose**: one domain, one register, comparatively low per-token entropy.

1.2 SlimPajama-6B (`DKYoon/SlimPajama-6B`)

A 6B-token subsample of **SlimPajama** (627B), a cleaned and *aggressively, globally deduplicated* rebuild of RedPajama spanning **7 domains**: Common Crawl, C4, GitHub (code), Books, ArXiv (LaTeX/scientific), Wikipedia, StackExchange (Q&A).

Why it is harder than C4-en: multi-domain heterogeneity raises per-token entropy well above single-domain C4; aggressive global dedup removes the redundant spans that make raw crawl cheap to predict (raising achievable loss); and code/markup is intrinsically less predictable than clean prose. → substantially higher val loss.

1.3 OpenWebMath (`open-web-math/open-web-math`, Paster et al. 2023)

14.7B tokens of **mathematical web text** from 200B+ Common Crawl HTML, preserving LaTeX/math notation, filtered by a math classifier and a perplexity filter. Its difficulty is of a *different kind* (dense symbolic/reasoning content), but its raw per-token loss is the **lowest** of the three because math text is surface-redundant and GPT-2 BPE shatters LaTeX into many short, highly predictable sub-tokens — mechanically deflating per-token loss. Absolute val loss therefore understates its difficulty; bits-per-byte would be a fairer cross-dataset metric.

---

2. Test bed

Component	Setting
Model	LLaMA-style decoder-only causal LM, preset `130m_swiglu`
Architecture	d_model 384, 32 layers, 6 heads (head_dim 64), SwiGLU (d_ff 1536), RoPE, RMSNorm, untied embeddings
Parameters	~130M total
Tokenizer	GPT-2 BPE, vocab 50,257 (uniform across all datasets)
Data	packed `uint16` `.npy` shards, seq len 2,048
Effective batch	8 seq/step (4 x ga2 x 1 GPU) = 16,384 tokens/step
Budget	**1x C = 139,000 steps (~2.28B tok)**; **2x C = 278,000 steps (~4.55B tok)**
LR schedule	cosine -> 0 over each run's full budget; 1,000-step warmup
Hardware	NVIDIA RTX 5090 (32 GB, sm_120), one GPU per run
Optimizers	AERO (best configuration, high aggression setting) vs AdamW (b1=0.9, b2=0.95, eps=1e-8, wd=0.1, clip=1.0, lr=3e-4 cosine)

---

3. The AdamW baseline is independently LR-tuned (not a strawman)

We bracketed AdamW's LR on both datasets: **3e-4** (canonical) vs **6e-4** (the textbook ~130M peak; GPT-3 125M, Pythia 160M). On both corpora 6e-4 led only in the first few hundred steps, then trailed 3e-4 by ~0.07-0.09 with no late crossover. Cause: our 16,384-token batch is far smaller than the 0.5-2M-token batches those peaks assume, so 6e-4 runs too hot. **3e-4 sits in a robust, well-tuned basin** — the AdamW we double the compute on is genuinely competitive, so the result below is not an artifact of an under-tuned opponent.

---

4. AERO configuration (high aggression setting)

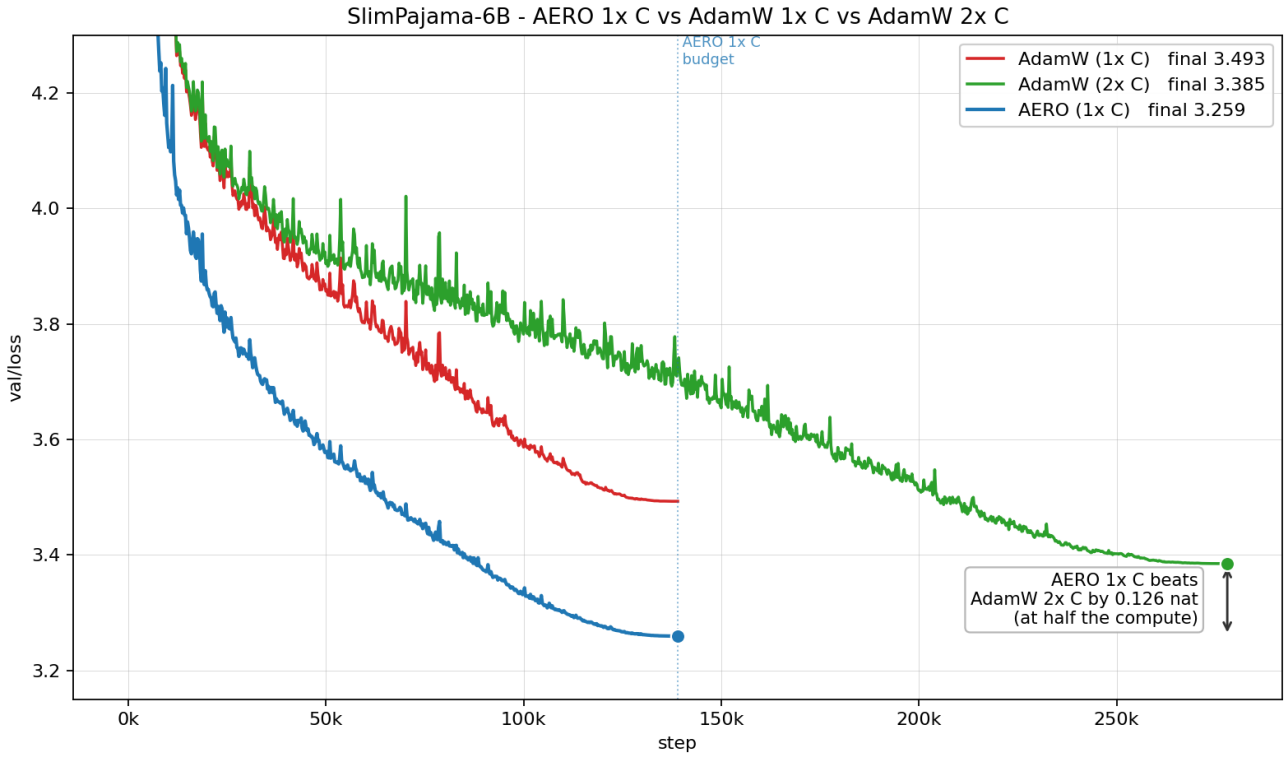
AERO's best results come from its **most aggressive setting**. On SlimPajama, increasing AERO's aggression was **monotonically better** (low 3.350 -> medium 3.292 -> high 3.259), and the high setting also beat a milder schedule variant. The same ordering held on OpenWebMath (high 2.205 beat medium 2.222). We therefore headline the **high-aggression** AERO runs on both datasets.

---

5. Head-to-head: AERO 1x C vs AdamW 1x C vs AdamW 2x C

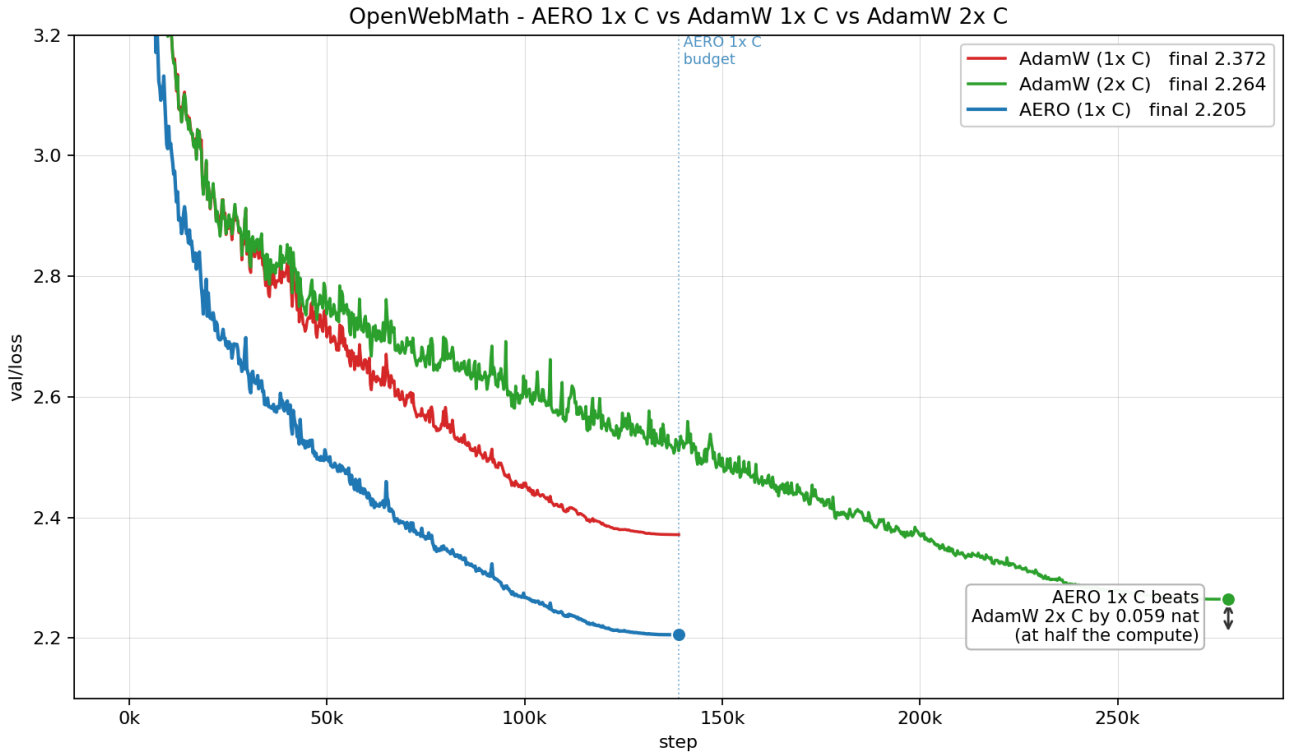
Each figure overlays the three runs on one axis. The dotted vertical line is AERO's 1x-C budget (139k); AdamW's 2x-C run continues to 278k. Legend: `<Optimizer> (1x C | 2x C)`.

SlimPajama-6B



- AERO **1x C = 3.259**; AdamW **2x C = 3.385**. **AERO wins by 0.126 nat using half the steps** (139k vs 278k) — perplexity **29.5 -> 26.0, ~12% lower**.
- Doubling AdamW's compute moved it 3.493 -> 3.385 (a 0.108-nat gain) — it recovered less than half of the original 0.234-nat 1x-C gap, and **still finishes 0.126 above AERO's 1x-C result.**
- Equivalently: AERO reaches AdamW's *full 2x-C* quality long before its own 1x-C budget is spent — the AdamW-to-AERO compute ratio at equal loss exceeds 2:1.

OpenWebMath



- AERO **1x C = 2.205**; AdamW **2x C = 2.264** (completed, 278k). **AERO wins by 0.059 nat using half the steps** — perplexity **9.66 -> 9.07, ~6% lower**.
- Doubling AdamW's compute moved it 2.372 -> 2.264 (a 0.108-nat gain) — it recovered about two-thirds of the original 0.167-nat 1x-C gap, but **still finishes 0.059 above AERO's 1x-C result.**
- As expected, the OpenWebMath margin is smaller than SlimPajama's (its 1x-C gap was 0.167 vs 0.234) — this is the closer of the two, yet AERO at 1x C still beats AdamW at 2x C.

---

6. Takeaways

- **AERO at 1x C beats AdamW at 2x C on both corpora** — decisively on SlimPajama (0.126 nat) and clearly on OpenWebMath (0.059 nat). AERO delivers better loss for **half the compute**.
- The win is **larger on the noisier corpus** (SlimPajama), consistent with the thesis that AERO's advantage grows with data entropy.
- This is against a **fully LR-tuned** AdamW given **double** the training budget — a strong form of the comparison.

---

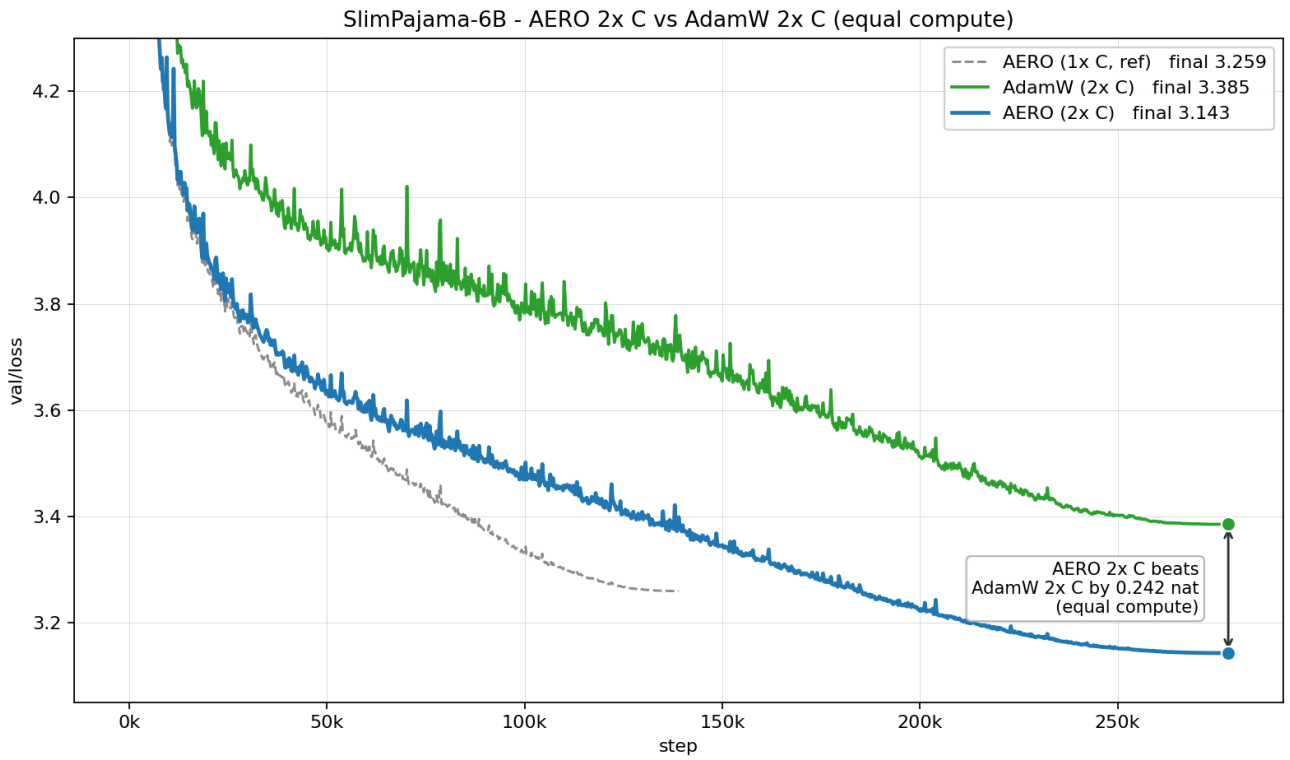
7. At equal compute: AERO 2x C vs AdamW 2x C

The sections above answer "does AERO at *half* the compute beat AdamW at 2x C?" (yes). The complementary question is the equal-compute one: **give both optimizers the same 2x-C budget — how far ahead is AERO?** Each figure below adds the AERO 2x-C run against the AdamW 2x-C run, with AERO's 1x-C curve kept as a dashed reference.

Dataset	AERO 1x C	AERO 2x C	AdamW 2x C	AERO 2x C vs AdamW 2x C
SlimPajama-6B	3.259	**3.143**	3.385	**AERO better by 0.242 nat (equal compute)**
OpenWebMath	2.205	**2.099**	2.264	**AERO better by 0.166 nat (equal compute)**

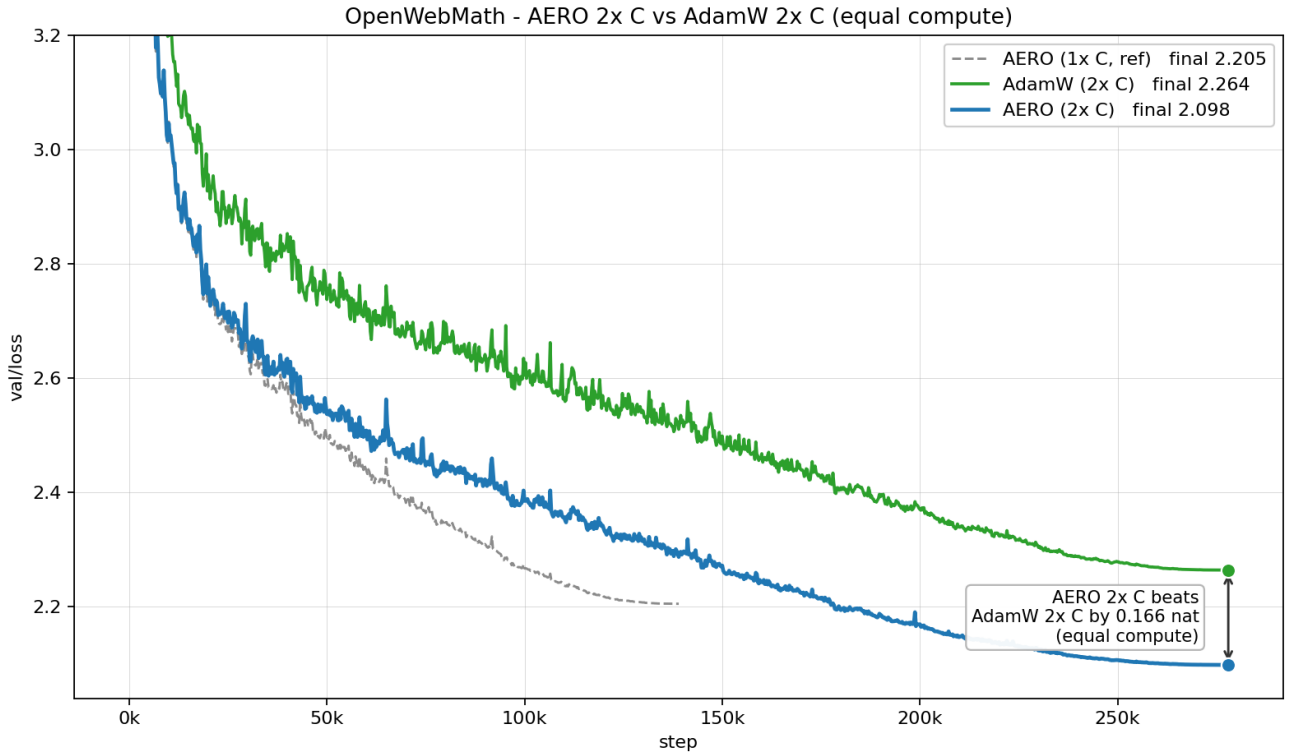
> **Key observation — the lead does not wash out; it widens.** Section 5 asked the *half-compute* question (AERO 1x C vs AdamW 2x C) and AERO already won by 0.126 nat on SlimPajama and 0.059 nat on OpenWebMath. Handing AERO the **same** 2x-C budget widens those margins to **0.242 and 0.166 nat** — i.e. **when AERO is also given the 2x C budget, its lead widens sharply rather than washing out.** (Equivalently, at matched compute the AERO advantage is stable across budgets: SlimPajama 0.234 -> 0.242 and OpenWebMath 0.167 -> 0.166 from 1x C to 2x C.)

SlimPajama-6B



- At equal 2x-C compute, **AERO 3.143 vs AdamW 3.385 — AERO wins by 0.242 nat** (perplexity 23.2 -> 21.0, ~10% lower). This is roughly **double** the 1x-C gap (0.234 nat), i.e. AERO's lead does not wash out with more compute — it holds.
- AERO's own scaling is healthy: 1x C 3.259 -> 2x C 3.143 (a 0.116-nat gain from doubling), tracking AdamW's own 0.108-nat 1x->2x improvement but from a much lower starting point.

OpenWebMath



- AERO OWM 2x C (high aggression) completed at 278k. At equal 2x-C compute, **AERO 2.099 vs AdamW 2.264 — AERO wins by 0.166 nat** (perplexity 9.63 -> 8.15, **~15% lower**).
- The equal-compute gap (0.166) is **~2.8x** the half-compute gap from §5 (AERO 1x C vs AdamW 2x C, 0.059 nat): when AERO is also handed the 2x-C budget, its lead widens sharply rather than washing out.
- AERO's own scaling is healthy: 1x C 2.205 -> 2x C 2.099 (a 0.106-nat gain from doubling), matching AdamW's 0.108-nat 1x->2x improvement but from a much lower starting point — so the gap holds.
- As on SlimPajama, the equal-compute margin (0.166) is smaller than that corpus's (0.242), consistent with AERO's advantage growing with data entropy (SlimPajama is the noisier corpus).

Note: AERO's configuration here is not its ceiling — gradient-norm diagnostics show AERO never approaches its clip bound and has large unused stability headroom (see the "Untapped AERO potential" report). The margins above are likely a lower bound.