AUDITING PERMANENCE: EXACT WRITE ACCOUNTING AND A MATCHED-VOLUME CONTROL FOR OPTIMIZER-STATE GATES

Anonymous authors
Paper under double-blind review

ABSTRACT

Gates that decide which updates reach a model's base weights are judged by whether the gated model forgets less. That comparison mixes which updates the gate selects with how many it lets through. We separate the two. A masked AdamW makes every routed write exact and logged. In runs that train only the shared parameters, all 440,401,920 routed coordinates of a 0.6B student end bit-identical to the pretrained weights. The same logging lets us build a matched-volume random control. We apply both to a controller that reads its own Adam-style moments and, per block, commits an update to the base weights, defers the example for replay into low-rank adapters, or withholds it. On a seven-phase continual-distillation stream, with a protocol fixed before the final runs, three findings hold at the level of the three stream seeds. First, selection adds nothing measurable: committing random blocks at the controller's rate, with its own deferral and replay, forgets the same (seed-level 90% interval [-0.014, +0.016]). Second, forgetting follows write volume only coarsely. Writing one to two orders of magnitude more, by committing everything or removing the agreement gate, raises forgetting in every seed. Within the gated range, neither authorized nor realized base-weight change orders the arms. Across 49 configurations the best single predictor counts adapter writes too (Spearman 0.49, against 0.21 for base writes alone). Third, the controller's permanent writes neither help nor measurably hurt retention compared with training only the shared parameters. They improve only a pre-declared secondary metric, in-phase learning (all three seeds). We also show that exact isolation blinds any gate that needs positive evidence from the optimizer's own zero-initialized state.

1 INTRODUCTION

An optimizer sets the size and direction of a parameter change but not *how long* it should last. Every accepted update enters the same weights as rules the model has seen many times. In a task-free stream this is the root of interference (McCloskey & Cohen, 1989; Kirkpatrick et al., 2017). A one-off example, an unsettled pattern and an unreliable teacher's output all become as permanent as the rest of the model. Complementary learning systems theory suggests a remedy: tentative storage holds new evidence, and slow storage absorbs it only once it is stable and recurrent (McClelland et al., 1995; Kumaran et al., 2016). Buffers, low-rank adapters (Hu et al., 2022) and base weights provide the substrates, and many recent methods gate which updates may reach the base weights, often using optimizer statistics (Chen et al., 2025; Tang et al., 2026; Hoy & Celik, 2025).

Such gates are judged by whether the gated model forgets less. That comparison mixes *which* updates the gate selects with *how many* it lets through. Forgetting grows with the amount of change (Kalajdzievski, 2024; He et al., 2026), so a gate that looks selective may only be a throttle. Separating the two requires counting exactly what a gate writes and comparing it with a control that writes the same amount at random. We build both and use them to audit one gate: a controller that reads Adam-style moments it tracks itself (Kingma & Ba, 2015). At every step it decides, for each of the M blocks with the largest gradients, to COMMIT the update to the base weights, to DEFER the example to a buffer that is REPLAYed into adapters if its bucket recurs, or to WITHHOLD it. Stable adapters are periodically CONSOLIDATEd. A masked AdamW enforces each decision exactly, so every run

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.

yields a *write ledger* of which coordinates were opened, by which decision, for which teacher. Our contributions, ordered by how much weight the evidence bears:

1. **A matched-volume random control, and its null.** Random commits at the controller's own rate, with its deferral and replay, forget and learn the same. The pre-declared test is null, and the seed-level 90% interval lies inside the pre-declared resolution (Section 5.1). The control is matched in commits, replay, effective writes and realized drift. It is not matched in merge authorizations, which exceed the controller's by $1.70\times$.
2. **Exact write accounting as the instrument.** Unauthorized coordinates stay bit-exactly unchanged (Theorem 1), every changed routed coordinate is in the ledger (Corollary 1), and the ledger bounds drift, merges included (Proposition 1). Reconciliation shows that 52% of the reference's authorized budget merged adapters that were never written (Section 5.2). One discrepancy remains unexplained: 2 coordinates in 1 of 5 audited zero-write runs (Section D).
3. **Volume governs forgetting only coarsely, and adapter writes count.** One to three orders of magnitude more base writes raise forgetting in every seed. Within the gated range, neither authorized nor realized base-weight change orders the arms. Counting adapter writes explains the two arms that write least to the base but forget most (Section 5.2).
4. **Isolation blinds optimizer-state gates.** Under exact isolation a gate that needs positive evidence from the optimizer's zero-initialized state never opens a closed block (Proposition 3). MoLF tracks momentum for every block (Tang et al., 2026); we prove some such tracker is necessary.
5. **Nulls and negative results.** Relative to training only the shared parameters, the controller's permanent writes neither help nor measurably hurt retention. They help only the pre-declared secondary metric, in-phase learning. Always-on LoRA ends lower on final loss, and text-only targets reverse the ranking of controllers.

Not claimed: a lower final loss than adapter-only training; that choosing *which* blocks to commit helps; that permanent writes *reduce* forgetting relative to shared-only training (below what three seeds resolve); that routed write volume predicts forgetting within the gated range or across objectives (Shenfeld et al., 2026); any effect at 1.7B (Section J); any on-device benefit (we measured on a GPU only).

2 Related work

Gating what an update may change. Many methods restrict which parameters an update may touch: gradient projection (Saha et al., 2021; Wang et al., 2021; Liang & Li, 2024), sparse or masked fine-tuning (Ansell et al., 2022; Du et al., 2024), block-wise training (Pan et al., 2024; Luo et al., 2024), and fixed masks (Cloud et al., 2024; Liu et al., 2026). Two read the optimizer state we read. MoFO updates only the coordinates with the largest momentum (Chen et al., 2025). MoLF routes each module's update to a dense or a LoRA expert by an Adam-moment score (Tang et al., 2026). Others decide when adapters enter the base. Online-LoRA merges on a loss plateau (Wei et al., 2025), ReLoRA on a schedule (Lialin et al., 2024), and SLAO into one running adapter (Qiao & Mahdavi, 2026). I-LoRA interpolates a fast and a slow LoRA with replay (Ren et al., 2024), the closest analogue of our fast-versus-permanent split. STABLE accepts, rescales or rejects each merge against a probe-measured forgetting budget (Hoy & Celik, 2025); we did not implement it, and our random control cannot say whether such outcome evidence selects better than gradient agreement. None of these gates is tested against a random control matched in volume.

How much changes versus which. Forgetting grows with the number of fine-tuned parameters and steps (Kalajdzievski, 2024) and with update magnitude (He et al., 2026). Random parameter subsets fine-tune as well as selected ones (Xu & Zhang, 2024). Across objectives, however, output shift rather than the size of the weight change predicts forgetting (Shenfeld et al., 2026). That is the strongest published challenge to a volume account, and Section 5.2 is our closest response within one objective. A different route to safe permanence adds parameters whose initial contribution is zero: zero-initialized adapters (Hu et al., 2022), zero-gated residual blocks (Lok, 2026), and shared or generated low-rank updates (Kaushik et al., 2026; Xiao et al., 2026b). These make a new parameter safe at insertion. Proposition 3 shows why a gate that needs positive evidence from zero-initialized optimizer state cannot open such parameters by itself.

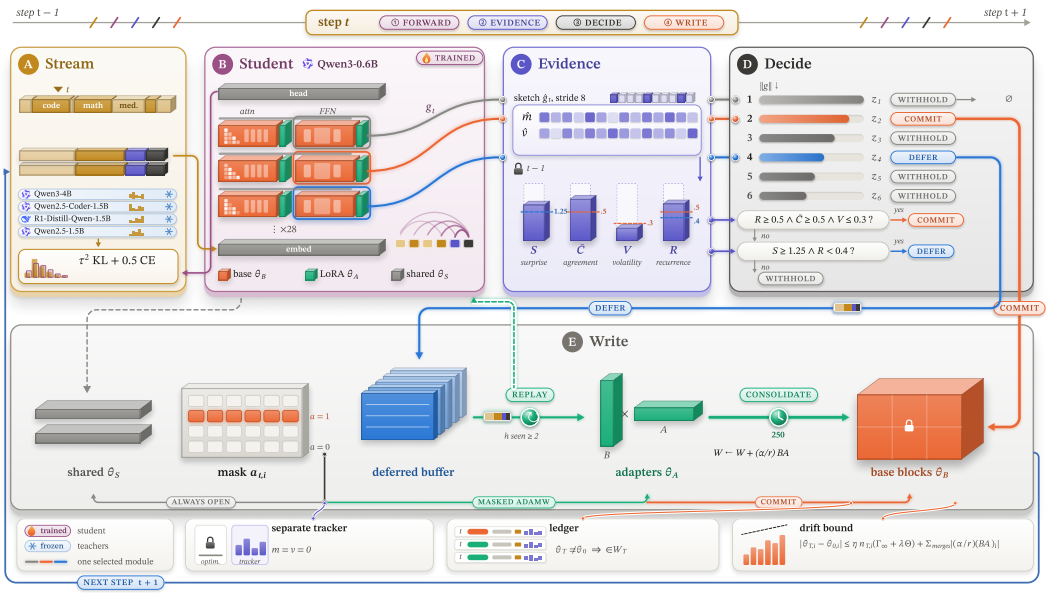

Figure 1: **One step of the gated controller.** A separate tracker (C) turns pre-update moments into four signals, and each of the M largest-gradient modules is decided by Equation (2) (D; confidence gate omitted). Only COMMIT and CONSOLIDATE write routed base weights, and both are logged (E).

Optimizer state as evidence. Adam's moments (Kingma & Ba, 2015; Loshchilov & Hutter, 2019) are read as a gradient signal-to-noise ratio (Liu et al., 2020), a diagonal Fisher (Li et al., 2025), a step-size signal (Zhuang et al., 2020) and a debugging aid (Schneider et al., 2021). Titans gates memory writes by surprise (Behrouz et al., 2025b), and Nested Learning treats optimizers as associative memories (Behrouz et al., 2025a). We borrow these signals. Masked per-coordinate bias correction appears in sparse fine-tuning and sparse training (Ansell et al., 2024; Xiao et al., 2026a).

Continual learning and consolidation. Regularization (Kirkpatrick et al., 2017; Zenke et al., 2017), replay (Rolnick et al., 2019; Buzzega et al., 2020; Ibrahim et al., 2024), parameter isolation (Mallya & Lazebnik, 2018; Serrà et al., 2018) and task-free methods (Aljundi et al., 2019) all ask when durable parameters should change (Shi et al., 2026). Fast weights (Hinton & Plaut, 1987; Ba et al., 2016), Lookahead (Zhang et al., 2019) and CLS-ER (Arani et al., 2022) consolidate on a schedule. Recent LLM work consolidates through prioritised replay (Hazard et al., 2025; Feng et al., 2026), test–retest selection (Han, 2026) or learned sleep (Behrouz et al., 2026a). Accumulated edits to base weights degrade a model (Gupta et al., 2024; Fang et al., 2025; Wang et al., 2024a). LoRA learns and forgets less than full fine-tuning (Biderman et al., 2024). Distillation (Hinton et al., 2015; Kim & Rush, 2016; Gu et al., 2024; Michel et al., 2026) only supplies our gradient stream. Section B extends this section.

3 METHOD

3.1 ROUTED AND SHARED PARAMETERS

The *routed base blocks* $\theta_{\mathcal{B}}$ are each layer's attention (q, k, v, o) and feed-forward (gate, up, down) projections. The *adapters* $\theta_{\mathcal{A}}$ are rank-r LoRA factors on the same projections (Hu et al., 2022). The *shared parameters* $\theta_{\mathcal{S}}$ are the tied embedding and output head and all normalization gains (26% of the model). A routed *module* is one layer's attention or feed-forward block, as base weights or as adapters; the 28-layer student has 112 modules and 4.5×10^8 routed base coordinates. The controller governs $\theta_{\mathcal{B}}$ and $\theta_{\mathcal{A}}$. The shared parameters train with ordinary AdamW on every step and every replay step, in every arm that trains the base model.

3.2 EXACT, DEFAULT-CLOSED WRITE AUTHORIZATION

Let $a_{t,i} \in \{0,1\}$ authorize routed coordinate i at optimizer call t, and let $n_{t,i}$ count its accepted updates. Masked AdamW applies

$$\begin{aligned} n_{t,i} &= n_{t-1,i} + a_{t,i}, \qquad m_{t,i} = a_{t,i}\big(\beta_1 m_{t-1,i} + (1-\beta_1) g_{t,i}\big) + (1 - a_{t,i})\, m_{t-1,i}, \\ v_{t,i} &= a_{t,i}\big(\beta_2 v_{t-1,i} + (1-\beta_2) g_{t,i}^2\big) + (1 - a_{t,i})\, v_{t-1,i}, \\ \theta_{t,i} &= \theta_{t-1,i} - a_{t,i}\,\eta\Big(\lambda\,\theta_{t-1,i} + \frac{m_{t,i}/(1-\beta_1^{n_{t,i}})}{\sqrt{v_{t,i}/(1-\beta_2^{n_{t,i}})} + \epsilon}\Big). \end{aligned} \tag{1}$$

Every routed coordinate is closed unless a decision at step t opens it. Here m, v and n are the *optimizer's* state, and its bias correction is coordinate-local.

Theorem 1 (Exact state isolation)**.** *If $a_{t,i} = 0$, then $\theta_{t,i} = \theta_{t-1,i}$, $m_{t,i} = m_{t-1,i}$, $v_{t,i} = v_{t-1,i}$ and $n_{t,i} = n_{t-1,i}$, and no weight decay is applied. If $a_{t,i} = 1$, coordinate i follows the AdamW recurrence with bias correction evaluated at its own count $n_{t,i}$.*

The proof is immediate from Equation (1); the property holds *bit-exactly* in bf16, since a closed coordinate's stored value is selected, never recomputed, while the merge identity (Proposition 7) holds only up to rounding. The theorem covers the routed parameters only, not the shared ones.

Corollary 1 (Ledger inclusion)**.** *Let $\mathcal{W}_T = \{i : \exists t \le T,\ a_{t,i} = 1\} \cup \{i \in$ blocks merged by* CONSOLIDATE*$\}$. Then $\{i \in \mathcal{B} \cup \mathcal{A} : \theta_{T,i} \neq \theta_{0,i}\} \subseteq \mathcal{W}_T$. Every change to a routed weight is attributable to a logged decision, which records its step, teacher and signals.*

Incidence. Diffing every archived run that has no logged routed write against the pretrained weights (5 runs, 2.2×10^9 coordinate checks) found 2 changed coordinates, both in one surprise-only run, the larger by one bf16 unit. The cause is unknown (Section D). Corollary 1 holds for the audited optimizer and merge paths, with this one exception in 5 runs.

Drift. Along its own accepted calls each coordinate runs ordinary AdamW (Lemma 2), and one step moves it by at most $\eta(\Gamma_\infty + \lambda\Theta)$ with $\Gamma_\infty \approx 7.27$, attained (Proposition 2).

Proposition 1 (Drift is bounded by logged writes, including merges)**.** *Let i be a routed base coordinate with $|\theta_{t,i}| \le \Theta$ for all $t \le T$, let $n_{T,i}$ count its commits, and let $\mathcal{M}_i$ be the logged merges into its block. Then*

$$|\theta_{T,i} - \theta_{0,i}| \;\le\; \eta_{\max}\, n_{T,i}\,(\Gamma_\infty + \lambda\Theta) \;+\; \sum_{m \in \mathcal{M}_i} \Big|\tfrac{\alpha}{r}\big(B_m A_m\big)_i\Big|.$$

Summed over the base weights, the ℓ_1 drift is at most $\eta_{\max}(\Gamma_\infty + \lambda\Theta)$ times the commit coordinate·steps, plus the ℓ_1 norms of the merged deltas.

A merge of a never-written adapter adds zero. On five arms with archived weights the commit term alone exceeds an upper estimate of realized ℓ_1 drift by 15–145$\times$ (Section C), so the bound is an accounting identity, not a predictor. We count *routed permanent writes* as coordinate·steps on base weights: coordinates opened by each COMMIT plus the size of each block that absorbs a CONSOLIDATE. This counts *authorizations*; Section 5.2 also counts *effective* writes (a merge counts only if its adapter was written) and *realized* change.

3.3 EVIDENCE FROM A SEPARATE MOMENT TRACKER

A never-opened block keeps $m = v = 0$ in the optimizer, so a gate reading the optimizer's state can never open it (Proposition 3, Figure 5). The controller therefore keeps a separate *tracker* of bias-corrected Adam-style moments $(\hat{m}, \hat{v})$ on a stride-8 sketch $\hat{g}_t$ of each module's gradient (d sketch coordinates; about 0.45 GB of fp32 state). Unlike the optimizer's state, it is updated at every training step for every module, on one global bias-correction clock, as in MoLF's universal momentum tracking (Tang et al., 2026). At step t all signals are computed from the pre-update $(\hat{m}_{t-1}, \hat{v}_{t-1})$, then every entry is updated and the clock advances once; replay steps do not touch the tracker. The

four signals are

$$S_t = \mathrm{clip}\Big(\tfrac{1}{d}\textstyle\sum_{i=1}^{d} \hat{g}_{t,i}^2/(\hat{v}_{t-1,i}+\epsilon),\ 0,\ 20\Big) \qquad \text{surprise}$$

$$\bar{C}_t = 0.9\,\bar{C}_{t-1} + 0.1\cos(\hat{g}_t, \hat{m}_{t-1}) \qquad \text{agreement}$$

$$V_t = (s_t - \mu_t^2)/(\mu_t^2+\epsilon),\ \ \mu_t = \mathrm{EMA}_{0.1}\|g_t\|,\ s_t = \mathrm{EMA}_{0.1}\|g_t\|^2 \qquad \text{volatility}$$

$$R_t = 0.3\,R_t^{\text{mom}} + 0.5\,R_t^{\text{bucket}} + 0.2\,R_t^{\text{emb}} \qquad \text{recurrence.}$$

R^{mom} is an EMA of $\max(0, \cos(\hat{g}_t, \hat{m}_{t-1}))$. $R^{\text{bucket}} = \frac{c}{c+10}\,\sigma(\bar{S}_{c-100}(h) - \bar{S}_c(h))$ rewards a data bucket h that recurs *and* whose surprise EMA $\bar{S}$ is falling, where c counts its visits; while $c \le 100$ the trend is set to 0, so $R^{\text{bucket}} = \frac{c}{2(c+10)}$. $R^{\text{emb}} = 1 - e^{-H/5}$, where H counts batch embeddings from the last 1000 steps with cosine ≥ 0.85 to the current one. Surprise has expectation 1 under stationarity (Proposition 5), and $\cos(g, m) \to \sqrt{\mathrm{SNR}/(1+\mathrm{SNR})}$ (Proposition 6); the gates read the averaged cosine because the instantaneous one is noisy (Blad et al., 2026). We do not claim these signals as new.

3.4 The decision

At each step the controller ranks the $4L$ modules by gradient norm, keeps the top $M = 6$, and classifies each selected module:

$$z_j = \begin{cases} \text{COMMIT} & \text{if } R \ge \rho_P \ \wedge\ \bar{C} \ge \theta_C \ \wedge\ V \le \theta_V, \\ \text{DEFER} & \text{else if } S \ge \theta_S \ \wedge\ R < \theta_R, \\ \text{WITHHOLD} & \text{otherwise,} \end{cases} \tag{2}$$

where $\rho_P = \theta_C = 0.5$, $\theta_V = 0.3$, $\theta_S = 1.25$ and $\theta_R = 0.40$. COMMIT opens a whole base block for one step; a confidence gate demotes it to WITHHOLD when the batch's mean teacher confidence is below 0.3. WITHHOLD opens nothing on a base block (on an adapter it would open the top $K = 8$ of $r = 16$ ranks, but top-M never selected an adapter in the reference runs). DEFER parks one row of the batch in a FIFO buffer of 1000 entries, appended once per deferring module. REPLAY runs after the optimizer step when the batch's bucket has entries from ≥ 2 distinct earlier steps and not every decision was DEFER. The oldest such step's row, whose entries are then all removed, gets one extra forward/backward pass and optimizer step that opens the top-K ranks of the M adapters with the largest replay gradients. CONSOLIDATE runs every 250 steps and merges adapter j ($W \leftarrow W + \frac{\alpha}{r}BA$) if $R \ge 0.4 \wedge \bar{C} \ge 0.4$, then re-draws A, zeroes B and resets only the adapter's optimizer state. Algorithm 1 gives the full step. Neither WITHHOLD nor DEFER opens a routed coordinate, so a routed weight changes only through the agreement gate or the recurrence path (DEFER, REPLAY, CONSOLIDATE).

4 Experimental setup

Stream and supervision. A Qwen3-0.6B student (Yang et al., 2025) is distilled for 6000 steps (batch 2, 512 tokens) through the seven-phase stream of Figure 2; recurrent phases fill the slots not taken by their 8 fixed buckets with fresh rows. Each corpus row holds a prompt, cached teacher answer, teacher id, confidence and bucket (domain, subdomain, difficulty). Four live teachers score the cached answers under teacher forcing: Qwen3-4B, Qwen2.5-Coder-1.5B-Instruct, DeepSeek-R1-Distill-Qwen-1.5B and Qwen2.5-1.5B-Instruct (Yang et al., 2025; Hui et al., 2024; Guo et al., 2025). The loss is $\tau^2\mathrm{KL}(p_T^\tau \| p_\theta^\tau) + 0.5\,\mathrm{CE}$ on the teacher text, with $\tau = 4$ and CE weighted by confidence (Hinton et al., 2015; Kim & Rush, 2016). LoRA uses $r = 16$, $\alpha = 32$; masked AdamW uses lr 10^{-4}, weight decay 0.01, bf16, untuned, in all arms (Section G).

Metrics. Every arm is evaluated on the same 32 fixed validation rows per domain. The *retention delta* of domain d at a phase boundary is its validation loss minus its loss at the end of the last phase that trained d. The mixed tail retrains every domain and contributes no deltas. **Forgetting** M_1 is the mean over domains of each domain's worst retention delta. **Plasticity** is the final macro validation loss. **Acquisition** is the own-phase loss: for code, math and medical, the domain's loss at the end of its own phase, averaged; lower means more was learned in-phase. Final loss alone cannot show acquisition, because the mixed tail retrains every domain. The **cost** is routed permanent writes.

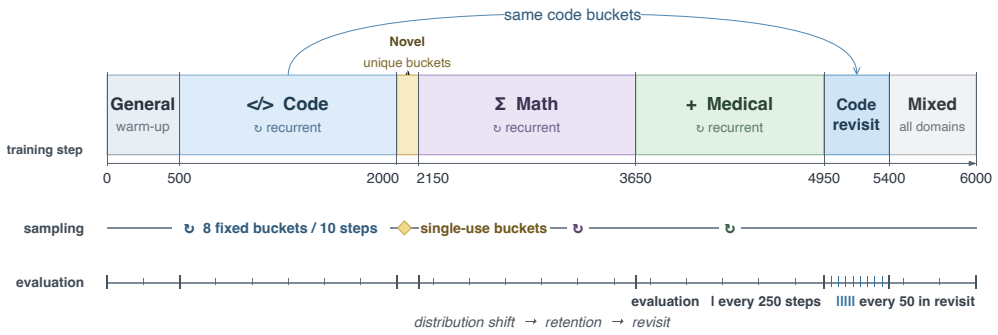

Figure 2: **The seven-phase continual-distillation stream** (steps). Recurrent phases revisit eight fixed buckets every 10 steps; the novel phase uses unique buckets. Evaluation runs at phase boundaries, every 250 steps, and every 50 steps in the code revisit.

Arms. All arms share the stream and objective; only the write rule changes (Table 1; definitions in Section H). The *reference* is the gated controller of Section 3 at the thresholds above. *Evidence-gated routing without permanence*: surprise-only routing, which defers if $S \geq \theta_S$ and otherwise withholds, with replay on (related to Titans' surprise gating, Behrouz et al., 2025b); and the controller with COMMIT and CONSOLIDATE disabled. *Controls on which updates become permanent*: *random relabel*, which keeps the controller's per-phase shares of all three decisions and shuffles which blocks get them; *random commit*, which keeps the controller's own deferral and replay and commits random selected blocks at the controller's per-phase commit rate (the *matched-volume random control*); always-COMMIT; an *SNR rule*, which commits if $\overline{\hat{m}^2/\hat{v}} \geq \tau_{\mathrm{SNR}}$ and otherwise withholds (it has no replay); MoLF's expected-preconditioned-descent argmax (Tang et al., 2026); and full fine-tuning. *Other trainable surfaces*: LoRA only (base and shared frozen); an Online-LoRA-style plateau merge (Wei et al., 2025), which never fired and so coincides with LoRA only; LoRA merged into the base and reset every 250 steps; and training only the shared parameters. Two arms of the earlier campaign are excluded (Section H).

Calibration. θ_S and θ_R were set once, on the controller's first live run, to maximise the gap between the defer rate in the novel phase and the largest defer rate in any recurrent phase. They were then frozen for all arms. θ_C was never tuned. Baseline thresholds were swept (Section J).

4.1 STATISTICAL CONVENTIONS

Unit of analysis. Seeds $\{1337, 2024, 7\}$ set the stream's sampling (recurring buckets, rows per step, novel rows) and the adapter initialization; phase structure and validation rows are fixed. Same-seed reruns see an identical stream, so the *stream seed* is the exchangeable unit. Every comparison in the main text averages an arm's runs within each seed and reports the per-seed paired difference from the reference ($k/3$ seeds worse), its mean, and a t interval on 2 degrees of freedom: 95% for effects, 90% for equivalence against δ_M. Pooled reruns (identical resolved configurations; the reference pool has 10 runs) serve only for descriptive means and standard deviations.

Noise and power. A one-way random-effects decomposition of the reference pool (Table 10) attributes 31% of the variance in M_1 to the stream seed ($\sigma_{\text{stream}} = 0.003$) and the rest to GPU nondeterminism ($\sigma_{\text{GPU}} = 0.005$). With three paired seeds the smallest attainable one-sided exact p is $1/8$ (0.625 after Holm over five tests), so no confirmatory test can reach $p \leq 0.05$ with the seed as the unit. The minimum detectable paired effect ($\alpha = 0.05$ one-sided, power 0.8) is 0.029 in M_1 ($1.4\,\delta_M$) and 0.017 in final loss ($1.1\,\delta_L$). Smaller differences are reported as unresolved, not as equivalence.

Pre-declared tests. The protocol was fixed before any final-batch result. A difference is *resolved* if it is at least $\delta_M = 0.02$ (M_1) or $\delta_L = 0.015$ (final loss), doubled for single-seed arms, *and* an exact one-sided permutation test against the pool gives $p \leq 0.05$. Five confirmatory tests, Holm-corrected, ask whether an arm forgets more than the reference: random relabel, random commit, always-commit,

Table 1: **Arms run on all three stream seeds.** Mean ± sd over pooled runs; *runs* counts pooled runs with identical configurations (the reference pool spans the 3 seeds). Writes are $\log_{10}$ coordinate·steps on routed base weights (0 = none). *perm.*: commit + merge as authorized, added in linear units. *eff.*: a merge counts only if its adapter was written since its last reset. * Full fine-tuning writes every base coordinate outside the controller. *adapter*: adapter coordinate·steps. *Replayed*: rows replayed into the adapters; each replay step also trains the shared parameters. Surface: S = shared, B = routed base blocks, A = adapters. *seeds*: stream seeds in which the arm's seed-mean M_1 exceeds the reference's (Section 4.1). Own-phase: mean over code, math and medical of each domain's loss at the end of its own phase (pre-declared secondary metric). Markers apply the pre-declared run-level rule: ▲ resolved worse, ▽ resolved better, ≈ level. Arms with fewer seeds are in Table 2.

			$\log_{10}$ routed base writes									
Arm	Surface	runs	commit	merge	perm.	eff.	$\log_{10}$ adapter	Replayed	M_1 (forgetting) ↓	seeds	Own-phase ↓	Final loss ↓
Evidence-gated routing												
Gated controller (reference pool)	S+B+A	10	8.7	9.0	9.2	8.9	8.2	274	0.084 ±0.006	–	1.111 ±0.006	1.246 ±0.004
Controller, no bucket-id recurrence	S+B+A	6	9.1	9.0	9.3	9.1	7.8	116	0.093 ±0.006 ≈	3/3	1.114 ±0.004	1.260 ±0.004 ≈
Controller, no DEFER tier (no replay)	S+B	6	8.9	9.1	9.3	8.9	4.4	0	0.096 ±0.008 ≈	3/3	1.135 ±0.009	1.279 ±0.011 ▲
Controller, never COMMIT/CONSOLIDATE	S+A	3	0	0	0	0	8.2	284	0.103 ±0.014 ≈	3/3	1.114 ±0.011	1.251 ±0.008 ≈
Surprise-only gate	S+A	3	0	0	0	0	8.3	416	0.090 ±0.006 ≈	3/3	1.121 ±0.012	1.266 ±0.001 ▲
Surprise-only gate, θ_S=4	S+A	3	0	0	0	0	7.8	133	0.074 ±0.003 ≈	0/3	1.120 ±0.003	1.262 ±0.009 ▲
Controls on which updates become permanent												
Random COMMIT, controller's own DEFER	S+B+A	3	8.7	9.2	9.3	8.9	8.1	272	0.084 ±0.004 ≈	2/3	1.112 ±0.001	1.252 ±0.007 ≈
Random relabel of all decisions	S+B+A	3	8.7	9.1	9.3	8.9	8.8	1305	0.164 ±0.072 ▲	3/3	1.176 ±0.055	1.287 ±0.033 ▲
Always COMMIT	S+B	3	11.4	6.5	11.4	11.4	0	0	0.260 ±0.010 ▲	3/3	1.146 ±0.006	1.257 ±0.013 ≈
SNR commit rule, τ_{SNR}=0.7	S+B	3	8.7	0	8.7	8.7	5.4	0	0.089 ±0.022 ≈	2/3	1.126 ±0.010	1.291 ±0.055 ▲
MoLF EPD rule	S+B+A	4	12.2	0	12.2	12.2	10.4	0	0.283 ±0.071 ▲	3/3	1.213 ±0.028	1.361 ±0.036 ▲
Full fine-tuning	S+B	3	12.4*	0	12.4	12.4	0	0	0.873 ±1.019 ▲	3/3	1.317 ±0.182	1.484 ±0.139 ▲
Controls by trainable surface												
Shared parameters only	S	3	0	0	0	0	0	0	0.084 ±0.012 ≈	1/3	1.136 ±0.004	1.258 ±0.003 ≈
LoRA only (plateau trigger never fired)	A	3	0	0	0	0	10.8	0	0.183 ±0.004 ▲	3/3	1.145 ±0.003	1.197 ±0.008 ▽
LoRA + merge every 250 steps	A+B	3	0	10.0	10.0	9.5	10.8	0	0.207 ±0.008 ▲	3/3	1.124 ±0.002	1.215 ±0.005 ▽

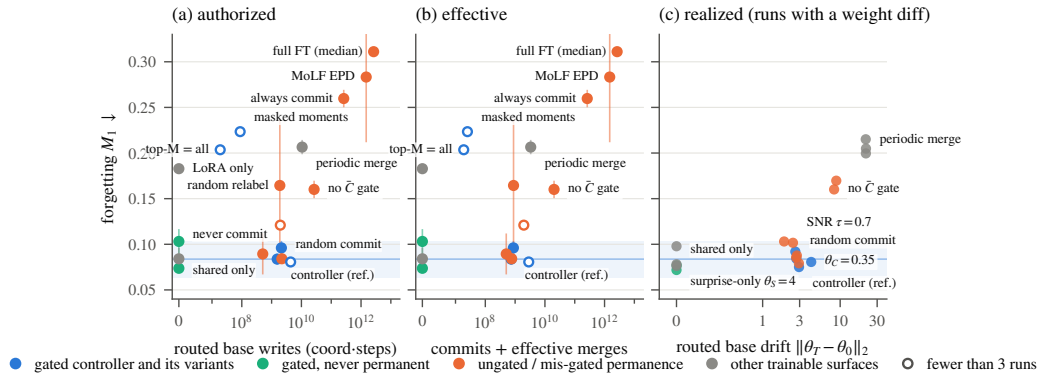

Figure 3: **Forgetting against three ways of counting base-weight change.** (a) Authorized routed writes, the pre-declared cost axis. (b) Effective writes: commits plus merges of written adapters. (c) Realized routed drift, one point per run with archived final weights (19 runs, 8 arms). Hollow: fewer than 3 runs. Band: reference $\pm\delta_M$. Full fine-tuning is drawn at its median; its mean (0.873) is off-scale because one of three seeds diverged.

no $\bar{C}$ gate, and the SNR rule at $\tau_{\text{SNR}} = 0.5$. The random-commit test was added before that arm ran. These tests treat pooled reruns as if they were independent streams. We report them as declared, with the post-hoc seed-level analysis alongside (Table 11). Forgetting and final loss are primary, own-phase loss secondary. Section 5.2 is entirely post hoc.

5 Results

5.1 Main comparison

Differences are paired per seed against the reference and averaged (Section 4.1; all arms in Table 12, run-level tests in Table 11); acquisition is reported because an arm can forget little by learning little.

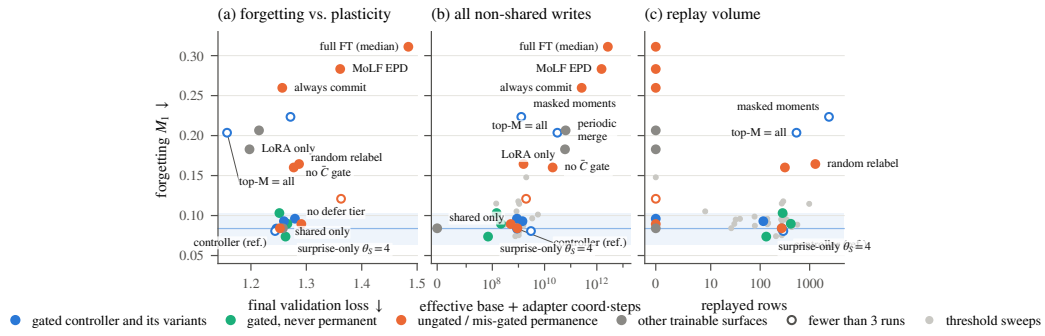

Figure 4: **What tracks forgetting** (post hoc). (a) Forgetting against final loss. (b) Forgetting against all non-shared writes (effective base + adapter coordinate·steps); small grey points are single-seed threshold sweeps. (c) Forgetting against replayed rows. The two arms that write least to the base but forget most (top-M = all, masked moments) sit with LoRA-only in (b).

At the gated volume, selection does not matter. The matched-volume random control (Figure 8) keeps the controller's deferral and replay. It commits as often (76 vs. 74 commits per run) and replays as many rows (272 vs. 274), but it commits randomly chosen blocks. It forgets the same, $\Delta M_1 = +0.001$ (worse in 2/3 seeds). The 90% seed-level interval [-0.014, +0.016] lies inside $\pm\delta_M$, and it learns the same (own-phase $\Delta = +0.001$). The pre-declared test is null (H1b, Holm $p = 0.472$). The control matches the controller in commits, replay, effective writes (4% more) and realized drift (ℓ_2 2.80 vs. 2.77). It authorizes 43% more writes, almost all of them merges of unwritten adapters, so authorizations are the wrong unit for a volume match. Random relabelling of all three decisions forgets more in 3/3 seeds ($\Delta = +0.081$, 95% interval [-0.089, +0.251]). It also replays 4.8× as many rows, so it tests replay and selection together.

Far above the gated volume, forgetting rises. Committing every selected update ($10^{11.4}$ writes) forgets more in 3/3 seeds ($\Delta = +0.176$, [+0.147, +0.206]) with no lower final loss ($\Delta = +0.011$). Removing the $\bar{C}$ gate multiplies commits by 34.2 and merges from 132 to 971, and triples realized drift (8.63 vs. 2.77). Forgetting rises in 3/3 seeds ($\Delta = +0.077$, [+0.043, +0.111]), and so does final loss ($\Delta = +0.031$). MoLF's descent-efficiency rule, full fine-tuning and periodic merging (realized drift 21.46) forget more in every seed. These arms stay the worst on the effective and realized axes of Figure 3, so this part of the volume result survives reconciliation. It does not extend into the gated range. At $\theta_C = 0.35$ (single seed) the controller commits 4.2× as often and drifts 4.23, and it stays level on both axes (0.081, 1.244).

Permanent writes neither help nor measurably hurt retention. The shared-parameters-only control opens no routed block and never replays. Its routed weights end bit-identical in all 3 runs. Its forgetting differs by $\Delta = +0.001$ (worse in 1/3 seeds). The 90% interval [-0.025, +0.027] is wider than $\pm\delta_M$, so three seeds cannot establish equivalence either; the pre-declared pooled-run bound is 0.019. It learns less in every seed (own-phase $\Delta = +0.025$, [+0.022, +0.029]), and its final loss is higher by +0.012, below δ_L. The controller's commits, merges and replays thus buy in-phase learning, on a secondary metric, at a retention cost too small to detect here. Never committing is not better. With COMMIT and CONSOLIDATE disabled, the variant learns as much (own-phase $\Delta = +0.003$) but forgets more in 3/3 seeds ($\Delta = +0.020$, [-0.003, +0.043]), which is unresolved. None of our 52 configurations has both lower forgetting and lower own-phase loss than the reference; only 17 have three seeds, so this describes our sweep, not dominance.

Deferral with replay is the plasticity channel; adapters alone forget more. The SNR rule at $\tau_{\text{SNR}} = 0.7$ commits as often as the controller but has no replay. It forgets about as much ($\Delta = +0.006$) and learns less in 3/3 seeds. Removing the defer tier does the same (M_1 $\Delta = +0.013$; final loss worse in 3/3 seeds). Surprise-only gating replays but never commits. At θ_S=4 it forgets slightly less in 3/3 seeds ($\Delta = -0.010$, below δ_M) and learns less in 3/3. LoRA only reaches the lowest final loss in every seed ($\Delta = -0.048$) but forgets 2.2× as much, gaining its edge in the mixed tail.

5.2 WRITE ACCOUNTING: AUTHORIZED, EFFECTIVE AND REALIZED CHANGE (POST HOC)

Reconciliation. Table 13 reconciles each arm's authorized writes with effective merges, adapter writes, replay, shared steps and, where archived, realized drift. The identity perm. = commit + merge holds in all 104 runs, and the ledger totals match the run summaries. Most merges change nothing: only 24 of the reference's 132 per run absorbed a written adapter (none in the no-defer-tier arm), so 52% of its authorized budget merges unwritten adapters. The shared parameters move far more than the routed ones. Their realized ℓ_2 drift is 179 in the reference, 65$\times$ its routed drift, and nearly the same in every arm that trains them, the shared-only control included.

Does base-weight volume order forgetting? Only across orders of magnitude. On realized drift, Spearman over all 19 runs is 0.62. It is driven by the no-$\bar{C}$ and periodic-merge arms; among the 11 runs of gated committing arms it is -0.04 ($p = 0.915$). Among the 38 configurations within $10\times$ of the reference's authorized writes, forgetting spans 0.074–0.223. Its rank correlation is 0.11 with authorized and 0.10 with effective base writes. The pre-declared volume claim holds as an order-of-magnitude statement only.

Adapter writes count. Two single-seed arms write far less to the base than the reference yet forget far more. Top-M = all writes $10^{7.3}$ to the base and forgets 0.204; the masked-moment ablation writes $10^{7.9}$ and forgets 0.223. Their decision logs explain both: top-M = all selects adapters directly (46.5% of decisions; 218$\times$ the reference's adapter coordinate·steps, about LoRA-only's), and with masked moments surprise saturates, so the arm defers 57.5% of decisions and replays 9.0$\times$ as many rows, each writing adapters and shared parameters. Across configurations the best single predictor is the sum of effective base and adapter writes: Spearman 0.79 on the 16 arms with three seeds and 0.49 on all 49. Authorized base writes give 0.58 and 0.21, replay rows alone -0.04. A log-linear fit is poor (R^2 0.19) but also gives adapter writes the largest positive partial effect (Table 14).

Attribution, rollback and the text-only reversal. In the text-only campaign the teacher-tagged ledger showed a noisy teacher reaching the base through the EPD rule (81% of its decisions) but never through the controller, and reverting attributed merges ran end to end (Figure 12). With text-only targets, two orderings of controllers reverse (Table 15), so rankings are regime-specific. M_1 ranks arms like three alternative forgetting summaries (Spearman ≥ 0.86 on arms with three seeds; Section K). Decision dynamics and ablations: Sections I and J.

6 LIMITATIONS

Power and evaluation. Three stream seeds give a minimum one-sided p of $1/8$ and a minimum detectable effect of about $1.4\,\delta_M$, above our retention comparison with shared-only training. Each domain is evaluated on 32 fixed rows without per-row losses, so evaluation noise is unquantified, and without cached generations we have no downstream metric beyond a 64-probe exact match, the highest-value missing evidence. **The missing control.** The most valuable missing arm is the controller with the shared parameters frozen: they are 26% of the model, train on every step, lie outside Theorem 1 and Corollary 1 and move about 65$\times$ as far as the routed weights, so ungoverned forgetting may enter through them. **Scope and accounting.** One 0.6B student, stream and learning rate, with no replay, EWC or STABLE baselines; rankings are objective-specific (Section 5.2). Realized drift exists for 19 runs only, merge-delta norms were not logged, and one run has an unexplained 1-ULP change (Section D). Replay and recurrence use bucket ids, calibration knew the novel phase, and the controller took 44% of wall-clock; we make no on-device claim.

7 CONCLUSION

Exact write authorization makes permanence countable, and counting changes the claims: for the gate we audited, selection added nothing measurable, forgetting followed volume only across orders of magnitude and only with adapter writes counted, and permanent writes bought acquisition, not retention. Selection rules should be tested against matched-volume random controls, counting change on every trained surface.

AI USE STATEMENT

We used generative AI tools to aid and polish the writing and to draft sections of the paper; for retrieval and discovery of related work; in research execution, to help implement the training and analysis code, give feedback on the experimental design and interpret results; and to assist in writing the proofs of the mathematical claims. The math and medical training corpora are synthetic: their answers were generated for this study by DeepSeek-R1-Distill-Qwen-1.5B and Qwen2.5-1.5B-Instruct (Section F). No AI tool generated experimental results; every reported number is computed by scripts from logged runs. We have reviewed all AI-assisted work and take responsibility for the final content of this work, including text, claims and artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work studies how parameter updates are committed during continual distillation. It uses public text corpora and publicly released models. It involves no human subjects and no personal data beyond what those public corpora contain. The noisy-teacher experiment shuffles answers within the training stream; it does not construct or release attacks. The ledger of permanent writes is intended to make model changes auditable. We see no direct harmful application beyond those of continual fine-tuning in general.

REPRODUCIBILITY STATEMENT

Section 3 and Section E specify the controller, and Sections G and H give every hyperparameter and arm override. Section C proves the formal statements. Every number, table and figure in the paper is regenerated by one command (`make results figures`) from per-run summaries, metric logs, decision reductions and weight diffs, which record the resolved configuration of each run. The post-hoc analyses of Sections 4.1 and 5.2 and Section K run in the same step. Byte-level checksum tests of every write path, runnable on a CPU, are in the released test suite (Section D), and merges now log the norm of the merged delta. Runs are pooled only when those configurations are identical. The pre-declared statistical protocol and the post-hoc seed-level convention are given in Section 4.1. Anonymized code, run data and the analysis pipeline (`paper_analysis/`) are at `https://anonymous.4open.science/r/TriHOPE-FAE9`; we will release them publicly with the final version.

REFERENCES

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In *The Twelfth International Conference on Learning Representations*, 2024. URL `https://openreview.net/forum?id=3zKtaqxLhW`.

Wasi Uddin Ahmad, Sean Narenthiran, Somshubra Majumdar, Aleksander Ficek, Siddhartha Jain, Jocelyn Huang, Vahid Noroozi, and Boris Ginsburg. OpenCodeReasoning: Advancing data distillation for competitive coding. In *Conference on Language Modeling (COLM)*, 2025.

Rahaf Aljundi, Klaas Kelchtermans, and Tinne Tuytelaars. Task-free continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11246–11255, 2019. doi: 10.1109/CVPR.2019.01151.

Alan Ansell, Edoardo Ponti, Anna Korhonen, and Ivan Vulić. Composable sparse fine-tuning for cross-lingual transfer. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1778–1796, Dublin, Ireland, 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.125. URL `https://aclanthology.org/2022.acl-long.125/`.

Alan Ansell, Ivan Vulić, Hannah Sterz, Anna Korhonen, and Edoardo M. Ponti. Scaling sparse fine-tuning to large language models. *arXiv preprint arXiv:2401.16405*, 2024.

Elahe Arani, Fahad Sarfraz, and Bahram Zonooz. Learning fast, learning slow: A general continual learning method based on complementary learning system. In *International Confer-*

ence on Learning Representations, 2022. URL https://openreview.net/forum?id=uxxFrDwrE7Y.

Kavosh Asadi, Rasool Fakoor, and Shoham Sabach. Resetting the optimizer in deep RL: An empirical study. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, pp. 72284–72324, 2023.

Jordan T. Ash and Ryan P. Adams. On warm-starting neural network training. In *Advances in Neural Information Processing Systems*, volume 33, pp. 3884–3894, 2020.

Jimmy Ba, Geoffrey E. Hinton, Volodymyr Mnih, Joel Z. Leibo, and Catalin Ionescu. Using fast weights to attend to the recent past. In *Advances in Neural Information Processing Systems*, volume 29, 2016. URL https://proceedings.neurips.cc/paper_files/paper/2016/hash/9f44e956e3a2b7b5598c625fcc802c36-Abstract.html.

Lukas Balles and Philipp Hennig. Dissecting Adam: The sign, magnitude and variance of stochastic gradients. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, volume 80 of *Proceedings of Machine Learning Research*, pp. 404–413, 2018.

Jihwan Bang, Hyunseo Koh, Seulki Park, Hwanjun Song, Jung-Woo Ha, and Jonghyun Choi. Online continual learning on a contaminated data stream with blurry task boundaries. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9275–9284, 2022.

Ali Behrouz, Meisam Razaviyayn, Peilin Zhong, and Vahab Mirrokni. Nested learning: The illusion of deep learning architectures. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, pp. 46968–47002. Curran Associates, Inc., 2025a. doi: 10.52202/085713-1565. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/4309616aaed8e848009bc4a7ef73b493-Abstract-Conference.html.

Ali Behrouz, Peilin Zhong, and Vahab Mirrokni. Titans: Learning to memorize at test time. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, pp. 113506–113543. Curran Associates, Inc., 2025b. doi: 10.52202/085713-3786. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/a4ca07aa108036f80cbb5b82285fd4b1-Abstract-Conference.html.

Ali Behrouz, Farnoosh Hashemi, Adel Javanmard, and Vahab Mirrokni. Language models need sleep: Learning to self-modify and consolidate memories. *arXiv preprint arXiv:2606.03979*, 2026a. URL https://arxiv.org/abs/2606.03979.

Ali Behrouz, Zeman Li, Praneeth Kacham, Majid Daliri, Yuan Deng, Peilin Zhong, Meisam Razaviyayn, and Vahab Mirrokni. ATLAS: Learning to optimally memorize the context at test time. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026b. URL https://openreview.net/forum?id=HV1R5bMSpa. Venue per OpenReview (ICML 2026); PMLR pages not verified.

Dan Biderman, Jacob Portes, Jose Javier Gonzalez Ortiz, Mansheej Paul, Philip Greengard, Connor Jennings, Daniel King, Sam Havens, Vitaliy Chiley, Jonathan Frankle, Cody Blakeney, and John Patrick Cunningham. LoRA learns less and forgets less. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=aloEru2qCG. Featured Certification.

Samuel Blad, Martin Längkvist, and Amy Loutfi. Measuring learning progress via gradient-momentum coupling. *arXiv preprint arXiv:2605.05856*, 2026. URL https://arxiv.org/abs/2605.05856.

Lucas Bourtoule, Varun Chandrasekaran, Christopher A. Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu Zhang, David Lie, and Nicolas Papernot. Machine unlearning. In *2021 IEEE Symposium on Security and Privacy (SP)*, pp. 141–159, 2021. doi: 10.1109/SP40001.2021.00019.

Collin Burns, Pavel Izmailov, Jan Hendrik Kirchner, Bowen Baker, Leo Gao, Leopold Aschenbrenner, Yining Chen, Adrien Ecoffet, Manas Joglekar, Jan Leike, et al. Weak-to-strong generalization: Eliciting strong capabilities with weak supervision. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, volume 235 of *Proceedings of Machine Learning Research*, pp. 4971–5012, 2024.

Pietro Buzzega, Matteo Boschini, Angelo Porrello, Davide Abati, and Simone Calderara. Dark experience for general continual learning: a strong, simple baseline. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 15920–15930. Curran Associates, Inc., 2020.

Yupeng Chen, Senmiao Wang, Yushun Zhang, Zhihang Lin, Haozhe Zhang, Weijian Sun, Tian Ding, and Ruoyu Sun. MoFO: Momentum-filtered optimizer for mitigating forgetting in LLM fine-tuning. *Transactions on Machine Learning Research*, 2025. URL https://openreview.net/forum?id=T1qXIDn9my.

Yung-Sung Chuang, Shang-Yu Su, and Yun-Nung Chen. Lifelong language knowledge distillation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 2914–2924, 2020. doi: 10.18653/v1/2020.emnlp-main.233.

Alex Cloud, Jacob Goldman-Wetzler, Evžen Wybitul, Joseph Miller, and Alexander Matt Turner. Gradient routing: Masking gradients to localize computation in neural networks. *arXiv preprint arXiv:2410.04332*, 2024. URL https://arxiv.org/abs/2410.04332.

Payel Das, Subhajit Chaudhury, Elliot Nelson, Igor Melnyk, Sarathkrishna Swaminathan, Sihui Dai, Aurélie Lozano, Georgios Kollias, Vijil Chenthamarakshan, Jiří Navrátil, et al. Larimar: Large language models with episodic memory control. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 10109–10126. PMLR, 2024.

Tim Dettmers and Luke Zettlemoyer. Sparse networks from scratch: Faster training without losing performance. *arXiv preprint arXiv:1907.04840*, 2019.

Shibhansh Dohare, J. Fernando Hernandez-Garcia, Qingfeng Lan, Parash Rahman, A. Rupam Mahmood, and Richard S. Sutton. Loss of plasticity in deep continual learning. *Nature*, 632(8026): 768–774, 2024. doi: 10.1038/s41586-024-07711-7.

Shangchen Du, Shan You, Xiaojie Li, Jianlong Wu, Fei Wang, Chen Qian, and Changshui Zhang. Agree to disagree: Adaptive ensemble knowledge distillation in gradient space. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pp. 12345–12355, 2020.

Wenyu Du, Shuang Cheng, Tongxu Luo, Zihan Qiu, Zeyu Huang, Ka Chun Cheung, Reynold Cheng, and Jie Fu. Unlocking continual learning abilities in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 6503–6522, Miami, Florida, USA, 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-emnlp.379. URL https://aclanthology.org/2024.findings-emnlp.379/.

Ronen Eldan and Mark Russinovich. Who's Harry Potter? approximate unlearning in LLMs, 2023.

Utku Evci, Trevor Gale, Jacob Menick, Pablo Samuel Castro, and Erich Elsen. Rigging the lottery: Making all tickets winners. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, volume 119 of *Proceedings of Machine Learning Research*, pp. 2943–2952, 2020.

Junfeng Fang, Houcheng Jiang, Kun Wang, Yunshan Ma, Jie Shi, Xiang Wang, Xiangnan He, and Tat-Seng Chua. AlphaEdit: Null-space constrained knowledge editing for language models. In *International Conference on Learning Representations (ICLR)*, 2025. URL https://openreview.net/forum?id=HvSytvg3Jh.

Mehrdad Farajtabar, Navid Azizan, Alex Mott, and Ang Li. Orthogonal gradient descent for continual learning. In *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pp. 3762–3773, 2020. URL https://proceedings.mlr.press/v108/farajtabar20a.html.

Yujie Feng, Xu Chu, Yongxin Xu, Guangyuan Shi, Bo Liu, and Xiao-Ming Wu. TaSL: Continual dialog state tracking via task skill localization and consolidation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1266–1279, Bangkok, Thailand, 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.69. URL https://aclanthology.org/2024.acl-long.69/.

Yujie Feng, Hao Wang, Jian Li, Xu Chu, Zhaolu Kang, Yiran Liu, Yasha Wang, Philip S. Yu, and Xiao-Ming Wu. FOREVER: Forgetting curve-inspired memory replay for language model continual learning. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2026. URL https://arxiv.org/abs/2601.03938.

Glaive AI. reasoning-v1-20m: Synthetic reasoning traces for diverse non-code/math topics. Hugging Face dataset https://huggingface.co/datasets/glaiveai/reasoning-v1-20m, 2025. Responses generated by DeepSeek-R1-Distill-Llama-70B; Apache-2.0.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. MiniLLM: Knowledge distillation of large language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=5h0qf7IBZZ.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning. *Nature*, 645(8081):633–638, 2025. doi: 10.1038/s41586-025-09422-z.

Akshat Gupta, Anurag Rao, and Gopala Anumanchipalli. Model editing at scale leads to gradual and catastrophic forgetting. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 15202–15232, 2024. doi: 10.18653/v1/2024.findings-acl.902.

Haoliang Han. EVAF: A test-retest protocol for selective parametric consolidation, 2026. URL https://arxiv.org/abs/2606.29916.

Tom Hartvigsen, Swami Sankaranarayanan, Hamid Palangi, Yoon Kim, and Marzyeh Ghassemi. Aging with GRACE: Lifelong model editing with discrete key-value adaptors. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 47934–47959. Curran Associates, Inc., 2023. doi: 10.52202/075280-2079. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/95b6e2ff961580e03c0a662a63a71812-Abstract-Conference.html.

Hugo Hazard, Zafeirios Fountas, Martin A. Benfeghoul, Adnan Oomerjee, Jun Wang, and Haitham Bou-Ammar. SuRe: Surprise-driven prioritised replay for continual LLM learning, 2025. URL https://arxiv.org/abs/2511.22367.

JinLi He, Liang Bai, and Xian Yang. Exploring the impact of parameter update magnitude on forgetting and generalization of continual learning, 2026. URL https://arxiv.org/abs/2602.20796.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. URL https://arxiv.org/abs/1503.02531. NIPS 2014 Deep Learning Workshop.

Geoffrey E. Hinton and David C. Plaut. Using fast weights to deblur old memories. In *Proceedings of the Ninth Annual Conference of the Cognitive Science Society*, pp. 177–186, Seattle, WA, 1987. URL https://www.cs.toronto.edu/~hinton/absps/fastweights87.pdf.

William Hoy and Nurcin Celik. STABLE: Gated continual learning for large language models, 2025. URL https://arxiv.org/abs/2510.16089.

Cheng-Yu Hsieh, Chun-Liang Li, Chih-Kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alex Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas Pfister. Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 8003–8017, 2023. doi: 10.18653/v1/2023.findings-acl.507.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Nathan Hu, Eric Mitchell, Christopher D. Manning, and Chelsea Finn. Meta-learning online adaptation of language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 4418–4432, 2023. doi: 10.18653/v1/2023.emnlp-main.268.

Jianheng Huang, Leyang Cui, Ante Wang, Chengyi Yang, Xinting Liao, Linfeng Song, Junfeng Yao, and Jinsong Su. Mitigating catastrophic forgetting in large language models with self-synthesized rehearsal. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1416–1428, 2024. doi: 10.18653/v1/2024.acl-long.77.

Tianjin Huang, Ziquan Zhu, Gaojie Jin, Lu Liu, Zhangyang Wang, and Shiwei Liu. SPAM: Spike-aware Adam with momentum reset for stable LLM training. In *International Conference on Learning Representations (ICLR)*, 2025.

Zeyu Huang, Yikang Shen, Xiaofeng Zhang, Jie Zhou, Wenge Rong, and Zhang Xiong. Transformer-patcher: One mistake worth one neuron. In *International Conference on Learning Representations (ICLR)*, 2023.

Binyuan Hui, Jian Yang, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, Jiajun Zhang, Bowen Yu, Keming Lu, et al. Qwen2.5-Coder technical report. *arXiv preprint arXiv:2409.12186*, 2024. URL https://arxiv.org/abs/2409.12186.

Tingfeng Hui, Zhenyu Zhang, Shuohuan Wang, Weiran Xu, Yu Sun, and Hua Wu. HFT: Half fine-tuning for large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12791–12819, Vienna, Austria, 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.acl-long.626. URL https://aclanthology.org/2025.acl-long.626/.

Adam Ibrahim, Benjamin Thérien, Kshitij Gupta, Mats L. Richter, Quentin Anthony, Timothée Lesort, Eugene Belilovsky, and Irina Rish. Simple and scalable strategies to continually pre-train large language models. *Transactions on Machine Learning Research*, 2024. URL https://openreview.net/forum?id=DimPeeCxKO.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic. In *The Eleventh International Conference on Learning Representations (ICLR)*, 2023. URL https://openreview.net/forum?id=6t0Kwf8-jrj.

Joel Jang, Seonghyeon Ye, Sohee Yang, Joongbo Shin, Janghoon Han, Gyeonghun Kim, Stanley Jungkyu Choi, and Minjoon Seo. Towards continual knowledge learning of language models. In *International Conference on Learning Representations (ICLR)*, 2022. URL https://openreview.net/forum?id=vfsRB5MImo9.

Gangwei Jiang, Caigao Jiang, Zhaoyi Li, Siqiao Xue, Jun Zhou, Linqi Song, Defu Lian, and Ying Wei. Unlocking the power of function vectors for characterizing and mitigating catastrophic forgetting in continual instruction tuning. In *International Conference on Learning Representations (ICLR)*, 2025. URL https://openreview.net/forum?id=gc8QAQfXv6.

Damjan Kalajdzievski. Scaling laws for forgetting when fine-tuning large language models, 2024. URL https://arxiv.org/abs/2401.05605.

Prakhar Kaushik, Ankit Vaidya, Shravan Chaudhari, Rama Chellappa, and Alan Yuille. Shared LoRA subspaces for almost strict continual learning. *arXiv preprint arXiv:2602.06043*, 2026.

Zixuan Ke, Yijia Shao, Haowei Lin, Tatsuya Konishi, Gyuhak Kim, and Bing Liu. Continual pre-training of language models. In *International Conference on Learning Representations*, 2023.

Chris Dongjoo Kim, Jinseo Jeong, Sangwoo Moon, and Gunhee Kim. Continual learning on noisy data streams via self-purified replay. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 537–547, 2021.

Yoon Kim and Alexander M. Rush. Sequence-level knowledge distillation. In Jian Su, Kevin Duh, and Xavier Carreras (eds.), *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 1317–1327, Austin, Texas, November 2016. Association for Computational Linguistics. doi: 10.18653/v1/D16-1139. URL https://aclanthology.org/D16-1139/.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations (ICLR)*, 2015. URL https://arxiv.org/abs/1412.6980.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017. doi: 10.1073/pnas.1611835114.

Jongwoo Ko, Sungnyun Kim, Tianyi Chen, and Se-Young Yun. DistiLLM: Towards streamlined distillation for large language models. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, volume 235 of *Proceedings of Machine Learning Research*, pp. 24872–24895, 2024.

Saurabh Kumar, Henrik Marklund, and Benjamin Van Roy. Maintaining plasticity in continual learning via regenerative regularization. In *Proceedings of The 3rd Conference on Lifelong Learning Agents (CoLLAs 2024)*, volume 274 of *Proceedings of Machine Learning Research*, pp. 410–430. PMLR, 2025.

Dharshan Kumaran, Demis Hassabis, and James L. McClelland. What learning systems do intelligent agents need? complementary learning systems theory updated. *Trends in Cognitive Sciences*, 20 (7):512–534, 2016. doi: 10.1016/j.tics.2016.05.004.

YuXin Li, Felix Dangel, Derek Tam, and Colin Raffel. Fishers for free? approximating the Fisher information matrix by recycling the squared gradient accumulator. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 34252–34270, 2025. URL https://proceedings.mlr.press/v267/li25j.html.

Vladislav Lialin, Sherin Muckatira, Namrata Shivagunde, and Anna Rumshisky. ReLoRA: High-rank training through low-rank updates. In *The Twelfth International Conference on Learning Representations (ICLR)*, 2024. URL https://openreview.net/forum?id=DLJznSp6X3.

Yan-Shuo Liang and Wu-Jun Li. InfLoRA: Interference-free low-rank adaptation for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 23638–23647, 2024.

Jessy Lin, Luke Zettlemoyer, Gargi Ghosh, Wen-Tau Yih, Aram Markosyan, Vincent-Pierre Berges, and Barlas Oğuz. Continual learning via sparse memory finetuning, 2025. URL https://arxiv.org/abs/2510.15103.

Jinlong Liu, Yunzhi Bai, Guoqing Jiang, Ting Chen, and Huayan Wang. Understanding why neural networks generalize well through GSNR of parameters. In *International Conference on Learning Representations (ICLR)*, 2020. URL https://openreview.net/forum?id=HyevIJStwH.

Yazheng Liu, Yuxuan Wan, Rui Xu, Xi Zhang, Sihong Xie, and Hui Xiong. Attribution-guided continual learning for large language models. *arXiv preprint arXiv:2605.05285*, 2026. URL https://arxiv.org/abs/2605.05285.

Dante Lok. Gate-zero growth: A geometric framework for function-preserving continual learning. *arXiv preprint arXiv:2607.14571*, 2026.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Keming Lu, Bowen Yu, Fei Huang, Yang Fan, Runji Lin, and Chang Zhou. Online merging optimizers for boosting rewards and mitigating tax in alignment. arXiv preprint arXiv:2405.17931, 2024.

Wentao Lu. ReCoLoRA: Spectrum-aware recursive consolidation for continual LLM fine-tuning, 2026. URL https://arxiv.org/abs/2607.07719.

Qijun Luo, Hengxu Yu, and Xiao Li. BAdam: A memory efficient full parameter optimization method for large language models. In *Advances in Neural Information Processing Systems*, volume 37, pp. 24926–24958, 2024. doi: 10.52202/079017-0786.

Yun Luo, Zhen Yang, Fandong Meng, Yafu Li, Jie Zhou, and Yue Zhang. An empirical study of catastrophic forgetting in large language models during continual fine-tuning. *IEEE Transactions on Audio, Speech and Language Processing*, 33:3776–3786, 2025. doi: 10.1109/TASLPRO.2025.3606231.

Clare Lyle, Zeyu Zheng, Evgenii Nikishin, Bernardo Avila Pires, Razvan Pascanu, and Will Dabney. Understanding plasticity in neural networks. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 23190–23211. PMLR, 2023. URL https://proceedings.mlr.press/v202/lyle23b.html.

Clare Lyle, Zeyu Zheng, Khimya Khetarpal, Hado van Hasselt, Razvan Pascanu, James Martens, and Will Dabney. Disentangling the causes of plasticity loss in neural networks. In *Proceedings of The 3rd Conference on Lifelong Learning Agents (CoLLAs 2024)*, volume 274 of *Proceedings of Machine Learning Research*, pp. 750–783. PMLR, 2025.

Arun Mallya and Svetlana Lazebnik. PackNet: Adding multiple tasks to a single network by iterative pruning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7765–7773, 2018. doi: 10.1109/CVPR.2018.00810.

Daniel Marczak, Bartłomiej Twardowski, Tomasz Trzciński, and Sebastian Cygert. MagMax: Leveraging model merging for seamless continual learning. In *Computer Vision – ECCV 2024*, Lecture Notes in Computer Science, pp. 379–395. Springer, 2024. doi: 10.1007/978-3-031-73013-9_22.

Imad Eddine Marouf, Subhankar Roy, Enzo Tartaglione, and Stéphane Lathuilière. Weighted ensemble models are strong continual learners. In *Computer Vision – ECCV 2024*, Lecture Notes in Computer Science, pp. 306–324. Springer, 2024. doi: 10.1007/978-3-031-73209-6_18.

James L. McClelland, Bruce L. McNaughton, and Randall C. O'Reilly. Why there are complementary learning systems in the hippocampus and neocortex: Insights from the successes and failures of connectionist models of learning and memory. *Psychological Review*, 102(3):419–457, 1995. doi: 10.1037/0033-295X.102.3.419.

Michael McCloskey and Neal J. Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of Learning and Motivation*, volume 24, pp. 109–165. Academic Press, 1989. doi: 10.1016/S0079-7421(08)60536-8.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in GPT. In *Advances in Neural Information Processing Systems*, volume 35, pp. 17359–17372, 2022.

Kevin Meng, Arnab Sen Sharma, Alex Andonian, Yonatan Belinkov, and David Bau. Mass-editing memory in a transformer. In *International Conference on Learning Representations (ICLR)*, 2023.

Nicolas Michel, Maorong Wang, Ling Xiao, and Toshihiko Yamasaki. Rethinking momentum knowledge distillation in online continual learning. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, volume 235 of *Proceedings of Machine Learning Research*, pp. 35607–35622, 2024.

Nicolas Michel, Maorong Wang, Jiangpeng He, and Toshihiko Yamasaki. Continual distillation of teachers from different domains. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

Eric Mitchell, Charles Lin, Antoine Bosselut, Chelsea Finn, and Christopher D. Manning. Fast model editing at scale. In *International Conference on Learning Representations (ICLR)*, 2022.

Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G. Patil, Ion Stoica, and Joseph E. Gonzalez. MemGPT: Towards LLMs as operating systems, 2023.

Rui Pan, Xiang Liu, Shizhe Diao, Renjie Pi, Jipeng Zhang, Chi Han, and Tong Zhang. LISA: Layerwise importance sampling for memory-efficient large language model fine-tuning. In *Advances in Neural Information Processing Systems*, volume 37, pp. 57018–57049, 2024. doi: 10.52202/079017-1817.

Subarnaduti Paul, Yohan Jung, Mohammad Emtiyaz Khan, Siddharth Swaroop, Thomas Möllenhoff, and Martin Mundt. Fast and slow variational continual learning. arXiv preprint arXiv:2606.24007, 2026.

Fuli Qiao and Mehrdad Mahdavi. Merge before forget: A single LoRA continual learning via continual merging. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 8133–8159, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/0dedca10992d34035c9a9776254cfa8c-Abstract-Conference.html.

Amrutha Varshini Ramesh, Vignesh Ganapathiraman, Issam H. Laradji, and Mark Schmidt. BlockLLM: Memory-efficient adaptation of LLMs by selecting and optimizing the right coordinate blocks. arXiv preprint arXiv:2406.17296, 2024. Also presented at OPT 2024: 16th Annual Workshop on Optimization for Machine Learning.

Sashank J. Reddi, Satyen Kale, and Sanjiv Kumar. On the convergence of Adam and beyond. In *International Conference on Learning Representations (ICLR)*, 2018. URL https://openreview.net/forum?id=ryQu7f-RZ.

Weijieying Ren, Xinlong Li, Lei Wang, Tianxiang Zhao, and Wei Qin. Analyzing and reducing catastrophic forgetting in parameter efficient tuning. *arXiv preprint arXiv:2402.18865*, 2024.

David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. Experience replay for continual learning. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

Gobinda Saha, Isha Garg, and Kaushik Roy. Gradient projection memory for continual learning. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=3AOj0RCNC2.

Imanol Schlag, Kazuki Irie, and Jürgen Schmidhuber. Linear transformers are secretly fast weight programmers. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 9355–9366, 2021. URL https://proceedings.mlr.press/v139/schlag21a.html.

Jürgen Schmidhuber. Learning to control fast-weight memories: An alternative to dynamic recurrent networks. *Neural Computation*, 4(1):131–139, 1992. doi: 10.1162/neco.1992.4.1.131.

Frank Schneider, Felix Dangel, and Philipp Hennig. Cockpit: A practical debugging tool for the training of deep neural networks. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 20825–20837. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/hash/ae3539867aaeec609a4260c6feb725f4-Abstract.html.

Joan Serrà, Dídac Surís, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 4548–4557. PMLR, 2018. URL https://proceedings.mlr.press/v80/serra18a.html.

Noam Shazeer and Mitchell Stern. Adafactor: Adaptive learning rates with sublinear memory cost. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, volume 80 of *Proceedings of Machine Learning Research*, pp. 4596–4604, 2018.

Idan Shenfeld, Jyothish Pari, and Pulkit Agrawal. RL's razor: Why online reinforcement learning forgets less. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=GO0X9nQot3.

Haizhou Shi, Zihao Xu, Hengyi Wang, Weiyi Qin, Wenyuan Wang, Yibin Wang, Zifeng Wang, Sayna Ebrahimi, and Hao Wang. Continual learning of large language models: A comprehensive survey. *ACM Computing Surveys*, 58(5):1–42, 2026. doi: 10.1145/3735633.

Weixi Song, Zuchao Li, Lefei Zhang, Hai Zhao, and Bo Du. Sparse is enough in fine-tuning pre-trained large language models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 46121–46135, 2024. URL https://proceedings.mlr.press/v235/song24e.html.

Fan-Keng Sun, Cheng-Hao Ho, and Hung-Yi Lee. LAMOL: LAnguage MOdeling for lifelong language learning. In *International Conference on Learning Representations (ICLR)*, 2020. URL https://openreview.net/forum?id=Skgxcn4YDS.

Yu Sun, Xinhao Li, Karan Dalal, Jiarui Xu, Arjun Vikram, Genghan Zhang, Yann Dubois, Xinlei Chen, Xiaolong Wang, Sanmi Koyejo, et al. Learning to (learn at test time): RNNs with expressive hidden states. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 57503–57522. PMLR, 2025.

Jihoon Tack, Jaehyung Kim, Eric Mitchell, Jinwoo Shin, Yee Whye Teh, and Jonathan Richard Schwarz. Online adaptation of language models with a memory of amortized contexts. In *Advances in Neural Information Processing Systems*, volume 37, pp. 130109–130135, 2024.

Haozhan Tang, Xiuqi Zhu, Xinyin Zhang, Boxun Li, Virginia Smith, and Kevin Kuo. Beyond LoRA vs. full fine-tuning: Gradient-guided optimizer routing for LLM adaptation. *arXiv preprint arXiv:2605.07111*, 2026. URL https://arxiv.org/abs/2605.07111.

Lukas Thede, Karsten Roth, Matthias Bethge, Zeynep Akata, and Thomas Hartvigsen. WikiBigEdit: Understanding the limits of lifelong knowledge editing in LLMs. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 59326–59354. PMLR, 2025.

Chenxu Wang, Yilin Lyu, Zicheng Sun, and Liping Jing. Continual gradient low-rank projection fine-tuning for LLMs. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 14815–14829, Vienna, Austria, 2025a. Association for Computational Linguistics. doi: 10.18653/v1/2025.acl-long.721. URL https://aclanthology.org/2025.acl-long.721/.

Ke Wang, Yiming Qin, Nikolaos Dimitriadis, Alessandro Favero, and Pascal Frossard. MEMOIR: Lifelong model editing with minimal overwrite and informed retention for LLMs. In *Advances in Neural Information Processing Systems*, 2025b. URL https://openreview.net/forum?id=t94tALZvZE.

Peng Wang, Zexi Li, Ningyu Zhang, Ziwen Xu, Yunzhi Yao, Yong Jiang, Pengjun Xie, Fei Huang, and Huajun Chen. WISE: Rethinking the knowledge memory for lifelong model editing of large language models. In *Advances in Neural Information Processing Systems*, volume 37, pp. 53764–53797, 2024a.

Shipeng Wang, Xiaorong Li, Jian Sun, and Zongben Xu. Training networks in null space of feature covariance for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 184–193, 2021.

Xiao Wang, Yuansen Zhang, Tianze Chen, Songyang Gao, Senjie Jin, Xianjun Yang, Zhiheng Xi, Rui Zheng, Yicheng Zou, Tao Gui, et al. TRACE: A comprehensive benchmark for continual learning in large language models, 2023.

Yu Wang, Yifan Gao, Xiusi Chen, Haoming Jiang, Shiyang Li, Jingfeng Yang, Qingyu Yin, Zheng Li, Xian Li, Bing Yin, et al. MEMORYLLM: Towards self-updatable large language models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 50453–50466. PMLR, 2024b.

Yu Wang, Dmitry Krotov, Yuanzhe Hu, Yifan Gao, Wangchunshu Zhou, Julian McAuley, Dan Gutfreund, Rogerio Feris, and Zexue He. M+: Extending MemoryLLM with scalable long-term memory. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 63308–63323. PMLR, 2025c.

Xiwen Wei, Guihong Li, and Radu Marculescu. Online-LoRA: Task-free online continual learning via low rank adaptation. In *Proceedings of the Winter Conference on Applications of Computer Vision (WACV)*, pp. 6634–6645, 2025. doi: 10.1109/WACV61041.2025.00646.

Taiqiang Wu, Chaofan Tao, Jiahao Wang, Runming Yang, Zhe Zhao, and Ngai Wong. Rethinking Kullback-Leibler divergence in knowledge distillation for large language models. In *Proceedings of the 31st International Conference on Computational Linguistics (COLING)*, pp. 5737–5755, 2025. URL https://aclanthology.org/2025.coling-main.383/.

Tongtong Wu, Linhao Luo, Yuan-Fang Li, Shirui Pan, Thuy-Trang Vu, and Gholamreza Haffari. Continual learning for large language models: A survey, 2024.

Qiao Xiao, Boqian Wu, Patrik Okanovic, Tomasz Sternal, Maurice van Keulen, Elena Mocanu, Mykola Pechenizkiy, Decebal Constantin Mocanu, and Torsten Hoefler. Memory-efficient LLM training with dynamic sparsity: From stability to practical scaling. In *Proceedings of the 43rd International Conference on Machine Learning (ICML)*, 2026a. URL https://openreview.net/forum?id=ivVPgBZewP.

ShiLin Xiao, Tianxiang Xu, Canran Xiao, Weihao Luo, Liwei Hou, and Chuangxin Zhao. Meta-UCF: Unified task-conditioned LoRA generation for continual learning in large language models. In *The Fourteenth International Conference on Learning Representations*, 2026b. URL https://openreview.net/forum?id=iNg5KL7eTC.

Shuo Xie and Zhiyuan Li. Implicit bias of AdamW: ℓ_∞-norm constrained optimization. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, volume 235 of *Proceedings of Machine Learning Research*, pp. 54488–54510, 2024.

Jing Xu and Jingzhao Zhang. Random masking finds winning tickets for parameter efficient fine-tuning. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 55501–55524, 2024. URL https://arxiv.org/abs/2405.02596.

Atsuki Yamaguchi, Terufumi Morishita, Aline Villavicencio, and Nikolaos Aletras. Mitigating catastrophic forgetting in target language adaptation of LLMs via source-shielded updates. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 18949–18975, San Diego, California, United States, 2026. Association for Computational Linguistics. doi: 10.18653/v1/2026.acl-long.865. URL https://aclanthology.org/2026.acl-long.865/.

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. URL https://arxiv.org/abs/2505.09388.

Lang Yu, Qin Chen, Jie Zhou, and Liang He. MELO: Enhancing model editing with neuron-indexed dynamic LoRA. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 19449–19457, 2024. doi: 10.1609/aaai.v38i17.29916.

Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 3987–3995. PMLR, 2017. URL https://proceedings.mlr.press/v70/zenke17a.html.

Hengyuan Zhang, Yanru Wu, Dawei Li, Sak Yang, Rui Zhao, Yong Jiang, and Fei Tan. Balancing speciality and versatility: a coarse to fine framework for supervised fine-tuning large language model. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 7467–7509, Bangkok, Thailand, 2024a. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.445. URL https://aclanthology.org/2024.findings-acl.445/.

Michael R. Zhang, James Lucas, Jimmy Ba, and Geoffrey E. Hinton. Lookahead optimizer: k steps forward, 1 step back. In *Advances in Neural Information Processing Systems*, volume 32, 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/hash/90fd4f88f588ae64038134f1eeaa023f-Abstract.html.

Qingru Zhang, Minshuo Chen, Alexander Bukharin, Nikos Karampatziakis, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. Adaptive budget allocation for parameter-efficient fine-tuning. In *International Conference on Learning Representations*, 2023.

Wenxuan Zhang, Paul Janson, Rahaf Aljundi, and Mohamed Elhoseiny. Overcoming generic knowledge loss with selective parameter update. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 24046–24056, 2024b.

Yushun Zhang, Congliang Chen, Naichen Shi, Ruoyu Sun, and Zhi-Quan Luo. Adam can converge without any modification on update rules. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, pp. 28386–28399, 2022.

Zheyuan Zhang, Alvin Zhang, Daniel Khashabi, and Tianmin Shu. Continual learning mechanisms compose for long-horizon memorization, 2026. URL https://arxiv.org/abs/2609.06986.

Junhao Zheng, Xidi Cai, Shengjie Qiu, and Qianli Ma. Spurious forgetting in continual learning of language models. In *International Conference on Learning Representations (ICLR)*, 2025. URL https://openreview.net/forum?id=ScI7IlKGdI.

Didi Zhu, Zhongyi Sun, Zexi Li, Tao Shen, Ke Yan, Shouhong Ding, Chao Wu, and Kun Kuang. Model tailor: Mitigating catastrophic forgetting in multi-modal large language models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 62581–62598, 2024. URL https://proceedings.mlr.press/v235/zhu24l.html.

Juntang Zhuang, Tommy Tang, Yifan Ding, Sekhar C. Tatikonda, Nicha Dvornek, Xenophon Papademetris, and James Duncan. AdaBelief optimizer: Adapting stepsizes by the belief in observed gradients. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 18795–18806. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/hash/d9d4f495e875a2e075a1a4a6e1b9770f-Abstract.html.

Fangyu Zou, Li Shen, Zequn Jie, Weizhong Zhang, and Wei Liu. A sufficient condition for convergences of Adam and RMSProp. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11127–11135, 2019.

Adam Zweiger, Jyothish Pari, Han Guo, Yoon Kim, and Pulkit Agrawal. Self-adapting language models. In *Advances in Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=JsNUE84Hxi. Author list per OpenReview; arXiv v2 additionally lists Ekin Akyürek.

A CLAIMS AT A GLANCE

Each claim with its evidence, whether that evidence is pre-declared or post hoc, and how it relates to prior work. What is not claimed is stated in Section 1.

Claim	Evidence	Status	Relation to prior work
Selection does not matter at the gated volume	matched-volume random commit: H1b Holm $p = 0.472$; seed-level 90% interval [-0.014, +0.016] inside $\pm\delta_M$; matched in effective writes and realized drift	pre-declared test (null); interval post hoc	new test for permanence gates; extends Xu & Zhang (2024) to forgetting
Unauthorized routed coordinates never change	Theorem 1 and Corollary 1; 0 of 440,401,920 changed in 3 shared-only runs; 2 unexplained coordinates in 1 of 5 zero-write runs (Section D)	proof; audit	per-coordinate bias correction as in Ansell et al. (2024)
Logged writes bound drift, merges included	Proposition 1; commit term exceeds realized ℓ_1 by 15–145$\times$	proof; post-hoc check	extends Reddi et al. (2018); Zou et al. (2019)
Far more base writes, more forgetting	always-commit, no $\bar{C}$: worse in 3/3 seeds (Holm $p = 0.017$, 0.017 at run level); also on effective and realized axes	pre-declared	consistent with Kalajdzievski (2024); He et al. (2026)
Within the gated range, base writes do not order forgetting	Spearman 0.11 (authorized), -0.04 (realized)	post hoc	qualifies the above; cf. Shenfeld et al. (2026)
Adapter writes count	all non-shared writes: Spearman 0.49 vs. 0.21; counterexamples explained by adapter writes	post hoc	new
Permanent writes buy acquisition, not retention	shared-only: $M_1\ \Delta = +0.001$ (equivalence not established); own-phase worse in 3/3 seeds	pre-declared metrics; seed-level post hoc	new
Replay is the learning channel	no defer tier: final loss worse in 3/3 seeds	pre-declared (final)	consistent with Rolnick et al. (2019); Buzzega et al. (2020)
Optimizer-state gates are blind under isolation	Proposition 3; masked-moment ablation commits once	proof; single seed	MoLF uses the remedy (Tang et al., 2026)
Rankings depend on the regime	text-only targets reverse two orderings (Table 15)	descriptive	evaluation caution

B EXTENDED RELATED WORK

This section places the paper more fully than Section 2 has room for. For each line of work we say what it decides, and from which signal.

Selective and constrained updates. One family constrains the *direction* of an update so that it avoids subspaces important to earlier tasks. Examples are orthogonal gradient descent (Farajtabar et al., 2020), gradient projection memory (Saha et al., 2021), null-space Adam (Wang et al., 2021), InfLoRA (Liang & Li, 2024) and low-rank projection for LLMs (Wang et al., 2025a). These methods need task boundaries to build the protected subspace, and every projected update is written. A second family chooses *which coordinates or blocks* to train:

- lottery-ticket and scalable sparse fine-tuning (Ansell et al., 2022; 2024);
- gradient-based sparse fine-tuning (Song et al., 2024) and adaptive rank budgets (Zhang et al., 2023);
- selective updates that protect generic knowledge (Zhang et al., 2024b;a; Hui et al., 2025; Feng et al., 2024; Zhu et al., 2024; Yamaguchi et al., 2026);
- activation-magnitude gating (Du et al., 2024);
- layer and block selection (Pan et al., 2024; Luo et al., 2024; Ramesh et al., 2024); BlockLLM, like our top-M step, ranks blocks by gradient norm.

MoFO (Chen et al., 2025) is the closest in signal: it updates only the coordinates with the largest momentum and mitigates forgetting in LLM fine-tuning. In all of these, a selected update goes straight into the weights being trained. There is no tier that defers an update, and no separate decision about permanence. Continual pre-training of language models also uses soft importance masks (Ke et al., 2023).

How much changes versus which. Several recent results concern the amount rather than the location of change. Kalajdzievski (2024) fit forgetting during LoRA and full fine-tuning of Llama-2 models as a shifted power law in the number of fine-tuned parameters and the number of update steps, and find an inverse linear relation between fine-tuning performance and forgetting. He et al. (2026) describe forgetting as knowledge degradation caused by task-specific drift in parameter space and modulate the update magnitude by gradient orientation. Xu & Zhang (2024) show that fine-tuning a random mask of parameters matches LoRA once the learning rate is raised, so precise selection is not needed for task performance. Shenfeld et al. (2026) report the opposite caution: across supervised and on-policy reinforcement fine-tuning, the KL divergence from the base model on the new task predicts forgetting, while the magnitude, sparsity and rank of the weight change do not. Our budget result is consistent with the first three within one distillation objective, and does not contradict the fourth, which compares objectives. What we add is a controlled test of selection at matched permanence volume, made possible by exact accounting.

Optimizer state and its analysis. Our signals are standard statistics of Adam's moments. m^2/v is the GSNR (Liu et al., 2020). v approximates the Fisher diagonal (Li et al., 2025). Agreement between gradient and momentum drives AdaBelief (Zhuang et al., 2020), and gradient–momentum coupling has been used to measure learning progress (Blad et al., 2026). Balles & Hennig (2018) decompose Adam into a sign and a variance-adapted magnitude, which is why a coordinate's first accepted step is sign-like.

The per-update bound of Proposition 2 applies the Cauchy–Schwarz argument used in convergence analyses of Adam (Reddi et al., 2018, App. C), (Zou et al., 2019, Lemma 20). We apply it to bias-corrected moments indexed by the coordinate-local counter, and show that the bound is attained. Zhang et al. (2022) bound the per-coordinate step more loosely, and Xie & Li (2024) note that single Adam steps can exceed the learning rate. Coordinate-local bias correction for weights that enter training at different times is used in scalable sparse fine-tuning (Ansell et al., 2024) and in dynamic sparse training, where a global clock inflates a regrown weight's first step (Xiao et al., 2026a). Sparse-to-dense training keeps momentum for zero weights (Dettmers & Zettlemoyer, 2019; Evci et al., 2020), which is in the spirit of our tracker for closed blocks. Resetting or filtering moments has been studied for spikes (Huang et al., 2025), for non-stationary objectives (Asadi et al., 2023) and for factored second moments (Shazeer & Stern, 2018).

Consolidation, merging and fast weights. Fast weights go back to Hinton & Plaut (1987) and Schmidhuber (1992), with modern forms in attention (Ba et al., 2016; Schlag et al., 2021) and test-time training (Sun et al., 2025; Behrouz et al., 2025b; 2026b). Slow–fast optimizers and memories consolidate by interpolation or averaging: Lookahead (Zhang et al., 2019), CLS-ER (Arani et al., 2022), online merging optimizers (Lu et al., 2024), continual model merging and averaging (Marczak et al., 2024; Marouf et al., 2024), and fast–slow variational continual learning (Paul et al., 2026). Adapter-based schedules include ReLoRA (Lialin et al., 2024), Online-LoRA (Wei et al., 2025), SLAO (Qiao & Mahdavi, 2026) and STABLE (Hoy & Celik, 2025). I-LoRA keeps a fast and a slow LoRA and interpolates between them, with replay (Ren et al., 2024); its slow LoRA plays the role our base weights play, and its interpolation replaces our gated merge. Share learns one low-rank subspace shared across tasks (Kaushik et al., 2026), and Meta-UCF generates task-conditioned LoRA updates with a hypernetwork (Xiao et al., 2026b). Gate-zero growth inserts residual blocks behind zero-initialized gates so that growth preserves the current function (Lok, 2026). The consolidation step in these methods runs on a schedule, at task boundaries, or when a loss or probe criterion fires, and applies to all weights or all adapters. Ours merges only the adapters whose own signals pass a gate, in a stream without boundaries. Recent work also consolidates through surprise-prioritised replay (Hazard et al., 2025), test–retest selection (Han, 2026), learned sleep (Behrouz et al., 2026a) or self-generated updates (Zweiger et al., 2025). Composing several continual-learning mechanisms is studied by Zhang et al. (2026), and recursive spectral consolidation of LoRA by Lu (2026).

Continual learning of LLMs. Surveys (Shi et al., 2026; Wu et al., 2024) and benchmarks (Wang et al., 2023; Jang et al., 2022) document forgetting in LLM streams. Several analyses bear on how forgetting should be measured. Zheng et al. (2025) attribute part of it to lost alignment rather than lost knowledge, Jiang et al. (2025) to biased function activation, and Luo et al. (2025) measure it empirically. Replay remains a strong baseline:

- continual pre-training with replay and learning-rate re-warming (Ibrahim et al., 2024);
- generative and self-synthesized rehearsal (Sun et al., 2020; Huang et al., 2024);
- online adaptation, where CaMeLS reweights tokens (Hu et al., 2023) and MAC stores amortized contexts (Tack et al., 2024).

Our deferred tier is a replay path that is triggered only by recurrence, and it writes only to adapters. Plasticity loss and its remedies (Dohare et al., 2024; Lyle et al., 2023; 2025; Ash & Adams, 2020; Kumar et al., 2025) motivate reporting final loss next to forgetting.

Model editing and side memories. Locate-then-edit methods write edits directly into base weights (Meng et al., 2022; 2023), and accumulated edits degrade the model (Gupta et al., 2024; Thede et al., 2025). Remedies include null-space constraints (Fang et al., 2025), learned editors (Mitchell et al., 2022), codebooks and patches (Hartvigsen et al., 2023; Huang et al., 2023; Yu et al., 2024), and side memories with routing (Wang et al., 2024a; 2025b). Memory-augmented models keep new knowledge in latent pools (Wang et al., 2024b; 2025c; Das et al., 2024) or in paged context (Packer et al., 2023). These methods decide where knowledge lives *at inference*. We decide, during training, whether each update may reach the base weights.

Distillation, noisy supervision and reversibility. Our objective follows logit and sequence-level distillation (Hinton et al., 2015; Kim & Rush, 2016). Recent LLM variants change the divergence (Gu et al., 2024; Agarwal et al., 2024; Ko et al., 2024; Wu et al., 2025) or the supervision (Hsieh et al., 2023). With several teachers, gradients can be reconciled in gradient space (Du et al., 2020). Continual distillation from a stream of domain teachers is studied by Michel et al. (2026), and online continual distillation by Michel et al. (2024) and Chuang et al. (2020). Noisy or contaminated streams are handled by purified replay (Kim et al., 2021; Bang et al., 2022), and weak supervisors are studied by Burns et al. (2024). For reversibility, exact unlearning shards training so that a datum's influence can be removed (Bourtoule et al., 2021), approximate unlearning fine-tunes it away (Eldan & Russinovich, 2023), and task arithmetic subtracts task vectors (Ilharco et al., 2023). Our ledger makes committed writes auditable and merges revertible through pre-merge checkpoints (Section M). It does not unlearn exactly what the shared parameters absorbed.

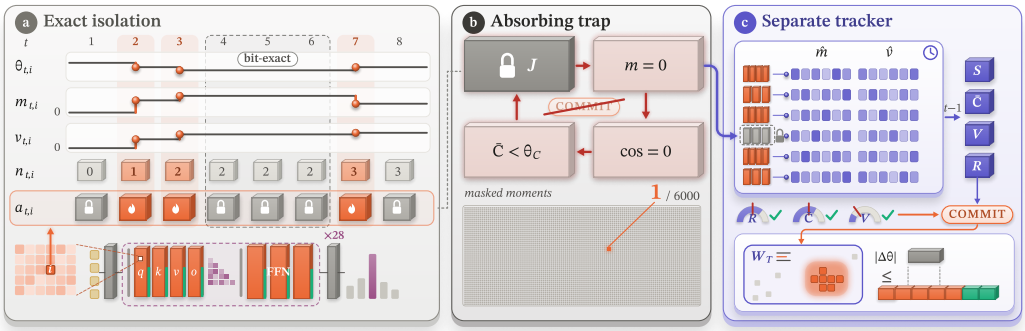

Figure 5: **Isolation and its blind spot.** (a) Theorem 1 for one routed coordinate: θ, m, v and the count n change only at steps with $a_{t,i} = 1$ and stay bit-exact otherwise. (b) Proposition 3: a closed block keeps $m = 0$, so its agreement is 0, $\bar{C} < \theta_C$, and it is never committed; reading the optimizer's own masked moments, the controller makes 1 commit in 6000 steps. (c) The separate tracker updates every module, closed ones included, so the gate can open them; each commit enters the ledger $\mathcal{W}_T$, which bounds drift (Proposition 1).

C Formal results

C.1 Setting

Let $\theta \in \mathbb{R}^D$ be partitioned into routed coordinates $\mathcal{R} = \mathcal{B} \cup \mathcal{A}$ (base blocks and adapters) and shared coordinates $\mathcal{S}$. Optimizer calls are indexed by $t = 1, 2, \ldots$; a replay step is one more call. At call t the optimizer receives a real gradient $g_t \in \mathbb{R}^D$ and an authorization vector $a_t \in \{0, 1\}^D$ with $a_{t,i} = 1$ for every $i \in \mathcal{S}$. For routed coordinates, $a_{t,i} = 1$ only if a logged decision opened i at call t. Masked AdamW keeps, for every coordinate, a first moment m, a second moment v, an optional AMSGrad maximum r and an accepted-update counter n, all initialized to 0. It applies

$$n_{t,i} = n_{t-1,i} + a_{t,i}, \tag{3}$$

$$m_{t,i} = m_{t-1,i} + a_{t,i}(1-\beta_1)(g_{t,i} - m_{t-1,i}), \qquad v_{t,i} = v_{t-1,i} + a_{t,i}(1-\beta_2)(g_{t,i}^2 - v_{t-1,i}), \tag{4}$$

$$r_{t,i} = r_{t-1,i} + a_{t,i}\big(\max(r_{t-1,i}, v_{t,i}) - r_{t-1,i}\big) \quad \text{(AMSGrad only)}, \tag{5}$$

$$\theta_{t,i} = \theta_{t-1,i} - a_{t,i}\,\eta_t\Big(\lambda\theta_{t-1,i} + \frac{\hat{m}_{t,i}}{\sqrt{\hat{s}_{t,i}} + \epsilon}\Big), \quad \hat{m}_{t,i} = \frac{m_{t,i}}{1-\beta_1^{n_{t,i}}},\ \hat{s}_{t,i} = \frac{s_{t,i}}{1-\beta_2^{n_{t,i}}}, \tag{6}$$

where $s = v$ (AdamW) or $s = r$ (AMSGrad), $\beta_1, \beta_2 \in [0, 1)$ and $\epsilon > 0$. The hat quantities are read only when $a_{t,i} = 1$, and then $n_{t,i} \geq 1$. Equation (1) in the main text is the AdamW case. All statements are in exact arithmetic. The implementation stores parameters and moments in bf16, so equalities that involve a merge hold only up to rounding. Isolation of closed coordinates, however, holds bit-exactly, because a closed coordinate is never written.

C.2 Exact isolation and its consequences

Proof of Theorem 1. Fix t and i with $a_{t,i} = 0$. Then Equations (3) to (5) give $n_{t,i} = n_{t-1,i}$, $m_{t,i} = m_{t-1,i}$, $v_{t,i} = v_{t-1,i}$ and $r_{t,i} = r_{t-1,i}$. Equation (6) multiplies both the decay term $\lambda\theta_{t-1,i}$ and the adaptive term by $a_{t,i} = 0$, so $\theta_{t,i} = \theta_{t-1,i}$ and no weight decay is applied. If $a_{t,i} = 1$, Equations (4) and (5) are the Adam and AMSGrad recurrences. Equation (6) applies decoupled decay followed by the bias-corrected adaptive step, with exponent $n_{t,i}$, the number of updates coordinate i has accepted, in place of a shared clock. □

Remark 1 (Why close coordinates instead of attenuating gradients). If every accepted gradient of a coordinate is multiplied by a constant $c > 0$, then $\hat{m}$ scales by c and $\sqrt{\hat{v}}$ scales by c. The step $\hat{m}/\sqrt{\hat{v}}$ is therefore unchanged for $\epsilon \to 0$; Adam is invariant to rescaling of the gradient (Kingma & Ba, 2015). Shrinking a coordinate's gradients before AdamW does not shrink the steps it takes

once the moments have adapted. Whether the coordinate moves is changed only by authorization, which Theorem 1 makes exact. Lin et al. (2025) warn that adaptive step sizes, momentum and weight decay can interact with sparse updates in unexpected ways. Theorem 1 and Corollary 2 spell out the interaction under masking: a closed coordinate's state is frozen, and its bias correction does not advance.

Corollary 2 (Coordinate-local bias correction)**.** *If coordinate i has accepted k updates before call t and $a_{t,i} = 1$, the update at t uses the factors $1 - \beta_1^{k+1}$ and $1 - \beta_2^{k+1}$, however many calls it spent closed.*

Proof. Per-weight step counters for bias correction also appear in sparse fine-tuning and dynamic sparse training (Ansell et al., 2024; Xiao et al., 2026a); here the counter follows from exact masking. Unrolling Equation (3) gives $n_{t-1,i} = \sum_{\tau<t} a_{\tau,i} = k$, so $n_{t,i} = k + 1$. By Theorem 1, closed calls change neither the counter nor the moments being corrected. □

Lemma 1 (All-open reduction)**.** *If $a_{t,i} = 1$ for all t and i, masked AdamW is AdamW (Loshchilov & Hutter, 2019), including its AMSGrad variant.*

Proof. Here $n_{t,i} = t$ for every i. Equations (4) and (5) then become the standard recurrences and Equation (6) the standard bias-corrected update, coordinate by coordinate. □

The next lemma is the structural fact behind everything that follows. Seen along its own accepted calls, a routed coordinate runs ordinary AdamW.

Lemma 2 (Subsequence equivalence)**.** *Let $t_1 < t_2 < \dots$ be the calls at which $a_{t,i} = 1$. The values $(\theta_{t_k,i}, m_{t_k,i}, v_{t_k,i}, r_{t_k,i})_{k\geq 1}$ are exactly those of scalar AdamW (or AMSGrad) run from $\theta_{0,i}$ and zero state on the gradients $g_{t_1,i}, g_{t_2,i}, \dots$ with learning rates $\eta_{t_1}, \eta_{t_2}, \dots$. Between accepted calls, the state is constant.*

Proof. By Theorem 1, every call strictly between t_k and t_{k+1} leaves the state unchanged, so the state entering call t_{k+1} is the state leaving call t_k. At t_{k+1}, Equation (3) sets $n_{t_{k+1},i} = k+1$. Equations (4) to (6) then perform the $(k + 1)$-th AdamW step on $g_{t_{k+1},i}$. Induction on k completes the proof. □

Proof of Corollary 1 (ledger inclusion). The only operations that write a routed coordinate are the optimizer update of Equation (6), and a merge $W \leftarrow W + \frac{\alpha}{r} BA$ together with the adapter reset that follows it. Take $i \in \mathcal{R} \setminus \mathcal{W}_T$. It is never authorized, so by Theorem 1 and induction over $t \leq T$ the optimizer never changes it. It is not in a merged block by definition of $\mathcal{W}_T$, so no merge changes it either. Hence $\theta_{T,i} = \theta_{0,i}$. Every authorization and every merge in $\mathcal{W}_T$ is emitted by the controller as a logged event, together with its step, module, teacher and signal values. □

C.3 The ledger bounds how far a coordinate can move

Let $w_k^{(n)} = \frac{(1-\beta_1)\beta_1^{n-k}}{1-\beta_1^n}$ and $u_k^{(n)} = \frac{(1-\beta_2)\beta_2^{n-k}}{1-\beta_2^n}$ for $k = 1, \dots, n$. Each set of weights sums to one. By Lemma 2, after a coordinate's n-th accepted update we have $\hat{m} = \sum_k w_k^{(n)} \gamma_k$ and $\hat{v} = \sum_k u_k^{(n)} \gamma_k^2$, where γ_k is the gradient at its k-th accepted call.

Proposition 2 (Per-update movement)**.** *Let $\rho = \beta_1^2/\beta_2 < 1$ and*

$$\Gamma(n)^2 = \sum_{k=1}^{n} \frac{(w_k^{(n)})^2}{u_k^{(n)}} = \frac{(1-\beta_1)^2(1-\beta_2^n)}{(1-\beta_1^n)^2(1-\beta_2)} \cdot \frac{1-\rho^n}{1-\rho}.$$

Then, for every gradient sequence, the adaptive term of AdamW after n accepted updates satisfies $|\hat{m}|/(\sqrt{\hat{v}} + \epsilon) \leq |\hat{m}|/\sqrt{\hat{v}} \leq \Gamma(n)$, with $0/0 := 0$. The second inequality is attained by $\gamma_k \propto w_k^{(n)}/u_k^{(n)}$, so $\Gamma(n)$ is the supremum. For $\beta = (0.9, 0.999)$, $\Gamma(1) = 1$, $\Gamma(n)$ is non-decreasing (verified numerically up to $n = 2 \cdot 10^4$), and $\Gamma_\infty = \sqrt{(1-\beta_1)^2/((1-\beta_2)(1-\rho))} \approx 7.27$.

Proof. The argument is the Cauchy–Schwarz step used in convergence analyses of Adam (Reddi et al., 2018, App. C), (Zou et al., 2019, Lemma 20), here applied to bias-corrected moments and to the coordinate-local count. By the Cauchy–Schwarz inequality, $|\hat{m}| = \left|\sum_k \frac{w_k}{\sqrt{u_k}} \cdot \sqrt{u_k}\gamma_k\right| \le \left(\sum_k \frac{w_k^2}{u_k}\right)^{1/2}\left(\sum_k u_k\gamma_k^2\right)^{1/2} = \Gamma(n)\sqrt{\hat{v}}$. Equality holds when $\sqrt{u_k}\gamma_k$ is proportional to $w_k/\sqrt{u_k}$. The closed form follows from summing the geometric series $\sum_{j=0}^{n-1}\rho^j$. Adding $\epsilon > 0$ to the denominator only decreases the ratio. □

Remark 2. Γ_∞ exceeds the "effective step size" $(1-\beta_1)/\sqrt{1-\beta_2} \approx 3.16$ that Kingma & Ba (2015, Sec. 2.1) describe for a sparse-gradient case. That figure is informal, and single Adam steps can exceed the learning rate (Xie & Li, 2024). Over arbitrary gradient sequences, the exact supremum is $\Gamma(n)$. The limit Γ_∞ is the constant that appears in Reddi et al. (2018, App. C) without bias correction.

Proof of Proposition 1. Between logged events a base coordinate changes only at its accepted calls and at merges into its block (Theorem 1; a merge adds $\frac{\alpha}{r}BA$ to the block and nothing else writes it). At the k-th accepted call it moves by $\eta_{t_k}|\lambda\theta_{t_k-1,i} + \hat{m}/(\sqrt{\hat{v}}+\epsilon)| \le \eta_{t_k}(\lambda\Theta + \Gamma(k))$ by Lemma 2 and Proposition 2. At merge m it moves by exactly $|\frac{\alpha}{r}(B_mA_m)_i|$ in exact arithmetic. The triangle inequality over all these events, with $\Gamma(k) \le \Gamma_\infty$ and $\eta_{t_k} \le \eta_{\max}$, gives the bound. For a base coordinate every accepted call is a COMMIT, so $\sum_i n_{T,i}$ is its commit coordinate·steps. A merge of an adapter that has not been written since its last reset has $B_m = 0$ and contributes nothing, which is why effective merges, not all merges, enter the second term. □

Without the merge term the bound would hold only for coordinates outside merged blocks. The campaign's event logs record which block was merged, when and for which teachers, but not the norm of the merged delta, so the merge term cannot be evaluated for these runs. The released code now logs $\|\frac{\alpha}{r}BA\|_1$, $\|\cdot\|_2$ and $\|\cdot\|_\infty$ with every merge, and a CPU test checks that the logged ℓ_1 equals the realized base change (Section D). Here we evaluate only the commit term. It is a valid check of the bound because the merge term is non-negative. For five arms with archived final weights we compare $\eta\,\Gamma_\infty \cdot (\text{commit coord} \cdot \text{steps})$ with an upper estimate of the realized routed ℓ_1 drift, $\sqrt{k}\,\|\theta_T - \theta_0\|_2$ over the k changed coordinates. For the reference these are 4.1×10^5 and 2.1×10^4. Across the five arms the commit term alone is $15\times$ to $145\times$ larger. The bound holds with wide slack. Most committed Adam steps are far smaller than the worst case $\Gamma_\infty\eta$, so the bound is an accounting identity and not a predictor of forgetting.

C.4 Isolation and observability

Proposition 3 (A closed block stays closed under optimizer-state evidence)**.** *Suppose the commit decision for a routed block J requires $\bar{C} \ge \theta_C > 0$, where $\bar{C}$ starts at 0 and averages $\cos(g_t, m_{t-1})$ computed from the optimizer's own first moment over J, with the convention $\cos(\cdot, 0) = 0$. Suppose also that no other path authorizes J. If J has never been authorized before call t, it is not authorized at call t. Hence a never-opened block is never opened.*

Proof. By induction on t. If J was never authorized before t, then by Theorem 1 and the zero initialization $m_{t-1,i} = 0$ for all $i \in J$. So $\cos(g_t, m_{t-1}) = 0$, $\bar{C}_t = 0 < \theta_C$, and the commit condition fails at t. □

Remark 3 (Scope of Proposition 3). The argument needs three things: the gate requires *positive* evidence ($\theta_C > 0$), the evidence is computed from the optimizer's own state, and that state starts at zero. It does not cover gates that open by default and close on evidence (for example, commit unless $\bar{C} < 0$). It does not cover gates that read a non-zero-initialized or separately tracked state, as MoLF and our tracker do, or gates whose evidence depends only on the current gradient, such as surprise computed against a fixed scale. Nor does it cover blocks that another path, such as replay, opens first.

Under exact isolation, then, the optimizer's state records only what the gate has already let in, and it cannot provide the evidence for letting anything else in. This matches the ablation that reads masked AdamW's own moments: it committed once in 6000 steps (Section J). Its decision log shows that the one commit was on an adapter module, a surface that replay opens first. The controller breaks the cycle with a separate tracker, in the spirit of sparse training methods that keep momentum for weights that are currently zero (Dettmers & Zettlemoyer, 2019; Evci et al., 2020). Its moments $(\tilde{m}, \tilde{v})$ are

updated at every call for every module, with the optimizer's β and a global clock, so it holds evidence for closed blocks as well.

Proposition 4 (Tracker agrees with Adam where both are defined)**.** *On a sketch coordinate that is authorized at every call, with the tracker's clock equal to the call index, the tracker's bias-corrected moments equal masked AdamW's $\hat{m}, \hat{v}$.*

Proof. Under these conditions both apply the same recurrence to the same gradients, with the same exponent in the bias correction, by Lemma 1. □

C.5 What the signals measure

Proposition 5 (Surprise is calibrated to 1)**.** *If the tracker's second moment equals the gradient's second moment, $\hat{v}_{t-1,i} = \mathbb{E}[g_{t,i}^2] > 0$ on every sketch coordinate, then without clipping and with $\epsilon = 0$, $\mathbb{E}[S_t] = 1$.*

Proof. $\mathbb{E}[S_t] = \frac{1}{d}\sum_i \mathbb{E}[g_{t,i}^2]/\hat{v}_{t-1,i} = 1$ by linearity. □

Surprise is therefore a whitened gradient energy: its value is about 1 when a batch pushes a block as hard as that block's history predicts. The defer threshold $\theta_S = 1.25$ asks for 25% more energy than predicted.

Proposition 6 (Agreement measures gradient signal-to-noise)**.** *Let $g = \mu + \xi \in \mathbb{R}^d$, where the coordinates of ξ are i.i.d. with mean 0, variance $\sigma^2 > 0$ and finite fourth moment, and $\|\mu\|^2/d \to a > 0$ as $d \to \infty$. With the momentum equal to the mean direction, $m = \mu$,*

$$\cos(g, m) \xrightarrow{p} \sqrt{\frac{\mathrm{SNR}}{1 + \mathrm{SNR}}}, \qquad \mathrm{SNR} = a/\sigma^2.$$

Proof. $\cos(g, \mu) = \frac{\|\mu\|^2 + \langle \mu, \xi \rangle}{\|\mu\| \, \|\mu + \xi\|}$. Here $\frac{1}{d}\langle \mu, \xi \rangle$ has mean 0 and variance $\sigma^2\|\mu\|^2/d^2 \to 0$, and $\frac{1}{d}\|\xi\|^2 \to \sigma^2$ by the weak law of large numbers. Hence $\frac{1}{d}\|\mu + \xi\|^2 \to a + \sigma^2$, and the ratio converges to $a/\sqrt{a(a + \sigma^2)}$. □

In the same model, the per-coordinate statistic m^2/v estimates $\mu_i^2/(\mu_i^2 + \sigma^2)$. This is the gradient signal-to-noise ratio of Liu et al. (2020), and it follows from $\mathbb{E}[g^2] = \mathbb{E}[g]^2 + \mathrm{Var}(g)$. So the agreement gate $\bar{C}$ and the SNR rule of Section 4 read closely related quantities. They differ in two ways: $\bar{C}$ aggregates over a whole block before thresholding and is averaged over time. We check the convergence numerically: at $d = 10^6$, $\mathrm{SNR} = 0.25$, the cosine is 0.4465 against the limit 0.4472.

C.6 Consolidation and the adapter budget

Proposition 7 (Merge identity)**.** *Let a LoRA layer compute $f(x) = Wx + b + sBAx$ with $s = \alpha/r$ and identity dropout. Replace W by $W^+ = W + sBA$, then re-draw A and set $B^+ = 0$. The layer computes the same function for every input x, and so does any network in which a finite set of such layers is replaced.*

Proof. For the layer, $W^+x + b + sB^+A^+x = Wx + sBAx + b$. For the network, order the replaced layers along the forward pass; each replacement leaves every downstream input unchanged, and induction completes the proof. Training-time dropout is excluded, because a random mask applied to x is not a fixed linear map. □

ReLoRA also merges low-rank updates into the base and resets optimizer state (Lialin et al., 2024). After a merge, the adapter's moments, AMSGrad maximum and counters are reset to zero. By Corollary 2, its next accepted update therefore uses first-update bias correction.

Proposition 8 (Exact rank budget)**.** *For $A \in \mathbb{R}^{r \times d_{\mathrm{in}}}$ and $B \in \mathbb{R}^{d_{\mathrm{out}} \times r}$, opening $K \le r$ rank components (rows of A and the matching columns of B) opens exactly $K(d_{\mathrm{in}} + d_{\mathrm{out}})$ of the $r(d_{\mathrm{in}} + d_{\mathrm{out}})$ adapter coordinates.*

Proof. Component k owns row k of A (d_{in} coordinates) and column k of B (d_{out} coordinates). Different components own disjoint sets, so the counts add. □

Correspondence with the implementation, and scope. Before any action is processed, the writer gives every routed parameter a zero mask. COMMIT opens a base block, adapt and replay open top-K rank components of adapters, and DEFER and WITHHOLD open nothing. Merges run between optimizer calls. The results above therefore confine every routed change to logged events. They do not cover the shared parameters, which are ordinary AdamW parameters and update on every call, including replay calls. The test suite checks Theorem 1, Corollary 2, Lemma 1, Proposition 7 and Proposition 8 on small models. The shared-parameters-only arm checks Theorem 1 at scale: its routed weights must end bit-identical to the pretrained ones. In all 3 runs, 0 of the 440,401,920 routed coordinates differ from the pretrained Qwen3-0.6B weights after 6000 steps in which the shared parameters trained on every step, and every adapter's B factor is still zero. The surprise-only gate at $\theta_S = 4$ also never opens a routed block, but it replays into the adapters. One of its two checked runs has 2 changed routed coordinates with no logged write; Section D reports what we ruled out.

D LEDGER AUDIT AND THE ONE DISCREPANCY

Scope and incidence. We can compare the ledger with the weights only where the final weights were archived. The snapshot holds a coordinate-level diff against the pretrained Qwen3-0.6B weights for 19 runs (Table 13). For runs with routed writes the diff cannot separate logged from unlogged changes, because it records counts and norms but not indices. For runs with *no* logged routed write, any change is unlogged. There are 5 such runs: the three shared-only runs and two surprise-only runs at $\theta_S = 4$ (2.2×10^9 coordinate checks). 2 coordinates changed, both in one surprise-only run (seed 2024). One changed by 0.00195, one bf16 unit at a magnitude in $[0.25, 0.5)$. The other changed by about 3×10^{-8}, consistent with one bf16 unit at a magnitude near 5×10^{-6}. The incidence is 1 of 5 zero-write runs and 2 of 2.2×10^9 checked coordinates. Both zero-write arms that replay into adapters (2 runs) are surprise-only runs, so the only affected arm is one that trains adapters.

What we ruled out. We audited the code on the released branch and added CPU tests (`tests/test_checksum_invariants.py`). The tests hash every parameter's raw bytes before and after every optimizer call and merge, in bf16.

- *The closed optimizer path.* A fully closed parameter is skipped before any arithmetic. A partially open one is written as `torch.where(mask, new, old)`, which returns the stored bf16 value unchanged for closed coordinates, and weight decay is inside the selected branch. The tests confirm byte-identical closed coordinates over random masks, for fully closed tensors, and when the open branch is non-finite.
- *Fused or multi-tensor kernels.* Masked AdamW refuses `fused`, `foreach`, `capturable` and `differentiable` at construction, so only the single-tensor path runs.
- *Aliasing with shared tensors.* The tied embedding and LM head are one shared parameter. Each routed projection is a fresh tensor copied from the pretrained linear at LoRA injection, so no routed tensor shares storage with a shared one. The tests check this by storage pointer.
- *Merges.* The arm runs with consolidation disabled, its ledger records 0 merges, and adapter-only steps (as in replay) leave every base weight byte-identical in the tests.
- *Serialization and precision round trips.* A bf16 checkpoint save and load is byte-exact, and bf16→fp32→bf16 is exact. The run was never interrupted or resumed.
- *Other loss terms.* The trust-region weight in the configuration is inert: no reference logits are passed, so the term is never computed. No loss term writes parameters.

What remains. We could not localize the two coordinates, because the snapshot does not keep indices or checkpoints. Three possibilities remain. (i) A write path on the campaign branch that differs from the released code we audited; the analysis scripts that produced the diff are on that branch. (ii) An error in the diff itself, for example a comparison against a differently loaded reference. (iii) A hardware fault. Until the checkpoint is re-examined, Corollary 1 holds for the audited paths and was violated once, by one bf16 unit, in 5 audited runs.

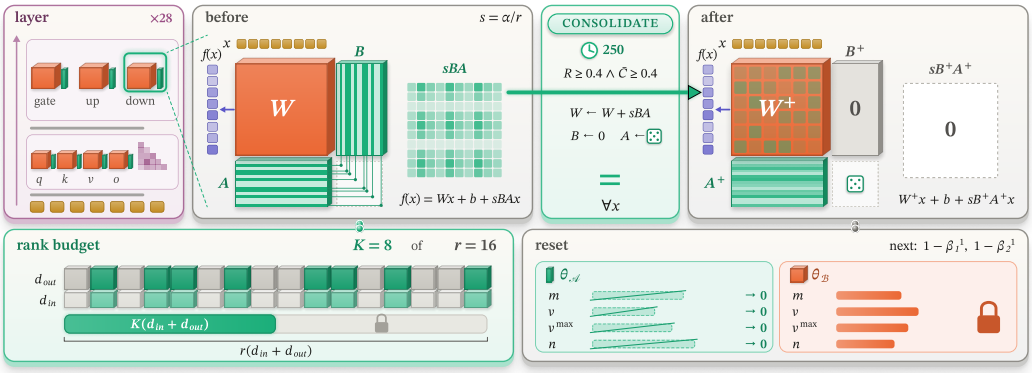

Figure 6: **Consolidating one adapter.** Every 250 steps an adapter with $R \geq 0.4 \wedge \bar{C} \geq 0.4$ is merged, $W \leftarrow W + sBA$ with $s = \alpha/r$; then $B \leftarrow 0$ and A is re-drawn, so the layer computes the same function up to rounding of W. Only the adapter's optimizer state is reset, so its next step uses first-step bias correction; the base block's state is kept. Replay opens the top $K = 8$ of $r = 16$ ranks (rank budget). The $v^{\max}$ row applies only with AMSGrad, which is off in all runs.

E ALGORITHM

Algorithm 1: One training step of the gated controller

Input: batch (x, y), bucket h, teacher k, confidence c; state: shadow moments, buffer $\mathcal{R}$, masked AdamW

compute loss $\mathcal{L} = \tau^2 \mathrm{KL}(p^{\tau}_{T_k} \,\|\, p^{\tau}_{\theta}) + \lambda_{\mathrm{CE}}\mathrm{CE} + 10^{-6}\|\theta_{\mathcal{A}}\|_1$ and backpropagate once;
foreach *routed module* j **do**
    read $g_t^{(j)}$ and the sketch $\hat{g}_t^{(j)}$; compute $S, \bar{C}, V, R$ from the tracker's pre-update $(\hat{m}_{t-1}, \hat{v}_{t-1})$
update every module's tracker moments with $\hat{g}_t^{(j)}$; advance the tracker clock once;
select the top-M modules by $\|g_t^{(j)}\|$;
foreach *selected module* j **do**
    $z_j \leftarrow$ Equation (2); **if** $c < 0.3$ *and* $z_j =$ COMMIT **then** $z_j \leftarrow$ WITHHOLD;
    if $z_j =$ COMMIT **then** open every coordinate of j;
    if $z_j =$ DEFER **then** append (row, h, k, t) to $\mathcal{R}$; `// one entry per deferring module`
    ;
    if $z_j =$ WITHHOLD *and* j *is an adapter* **then** open its top-K rank components;
    log (step, j, z_j, coordinates opened, h, k, signals);
masked AdamW step (shared parameters always open);
if $\mathcal{R}$ *holds entries of bucket* h *from* ≥ 2 *distinct earlier steps and not all* $z_j =$ DEFER **then**
    take the oldest such step's row and remove all its entries; forward/backward on it (tracker not updated); open top-K ranks of the top-M adapters; masked AdamW step; log replay;
if $t \bmod 250 = 0$ **then**
    foreach *adapter* j *with* $R \geq 0.4 \wedge \bar{C} \geq 0.4$ **do**
        log $\|\frac{\alpha}{r} B_j A_j\|$; $W_j \leftarrow W_j + \frac{\alpha}{r} B_j A_j$; re-draw A_j; $B_j \leftarrow 0$; reset the adapter's moments and counters (base block's untouched); log merge with teacher attribution

Mapping to the code. The released code labels the actions R, F and P. COMMIT is P on a base block, and DEFER is R. WITHHOLD is F on a base block, which opens nothing. The *adapt* action is F on an adapter, which opens top-K ranks; the top-M rule never selected one directly in our runs. The code counts replay writes as F actions on adapters. Because top-M always selected base blocks, adapters learned only through replay. In the original design, withholding a base block would have written that block's adapter; that variant is untested. Two sets of moments must be kept apart. The *optimizer's* moments and counters are frozen while a coordinate is closed (Theorem 1): a block that reopens after a long closure resumes from the optimizer state recorded when it was last open, and its bias correction uses its own count. The *tracker's* moments, which feed the signals, advance at every

step for every module on one global clock, as MoLF's universal tracking does for its routing scores. MoLF also ages the optimizer state of losers on the shared clock; we do not, because that would break Theorem 1. DEFER entries are appended once per deferring module, so the 1000-entry buffer holds between about 167 and 1000 distinct rows. Recurrence and replay count distinct steps, and a replayed step's duplicate entries are removed together. K is a fraction of the rank: $K = 0.5\,r = 8$ of $r = 16$ components. The design also allowed two other ways to make a write permanent: distilling the adapted student into the base before resetting the adapter, and base updates under a trust region. We use direct merges, and a COMMIT is an unregularized AdamW step at the shared learning rate.

F DATA AND SUPERVISION

Corpora. Each domain's corpus is capped at 40k rows and shares one schema: prompt, cached teacher answer, teacher id, per-answer confidence and a bucket (domain, subdomain, difficulty). The general rows derive from reasoning-v1 (Glaive AI, 2025), whose answers were written by DeepSeek-R1-Distill-Llama-70B. The code rows derive from OpenCodeReasoning (Ahmad et al., 2025), whose answers were written by DeepSeek-R1. The math and medical rows were generated for this study by DeepSeek-R1-Distill-Qwen-1.5B and Qwen2.5-1.5B-Instruct. The four processed corpora are public and will be linked after review. The code corpus has only three native buckets, so its buckets combine subdomain, difficulty and problem source (12 buckets). Recurrent phases exclude their chosen buckets from background draws, so background rows cannot inflate a bucket's recurrence count.

What the teachers are. The student is supervised by four frozen teachers that score the cached answer under teacher forcing: the KL target is the teacher's distribution over the fixed prompt-plus-answer sequence. The math and medical teachers are the models that wrote their domains' answers. Every teacher must share the student's vocabulary, so the general and code teachers (Qwen3-4B, Qwen2.5-Coder-1.5B-Instruct) are stand-ins that score another model's text. A row's teacher is fixed by its domain. The design also allows routing each sample to the teacher whose prototype (the mean embedding of seed examples) is closest in a shared embedding space; that router is implemented but unused here, because every row names its domain. Teacher selection is not part of this study.

Teacher confidence. For math and medical, confidence is derived from the authoring model's token distribution; its correlation with negative entropy is 0.99. For general and code, it is a score shipped with the corpus. Medical is the only domain where a material share of rows falls below the 0.3 gate. The confidence gate and confidence weighting should be read with this in mind.

Truncation. The student sees at most 512 tokens, and at most half of them are prompt. It therefore trains on the opening of each reasoning trace, and math answers are additionally cut at about 1000 characters.

Noisy-teacher stream. In the math phase, half of the math buckets, together with the matching background steps, have their answers shuffled among the corrupted rows. The corrupted rows keep their own confidence, except in the low-confidence arm, where it is set to 0.2. In the text-only campaign these rows carry their own teacher tag (Section M). In the live campaign they are scored by the math teacher like every other math row.

G HYPERPARAMETERS AND STREAM

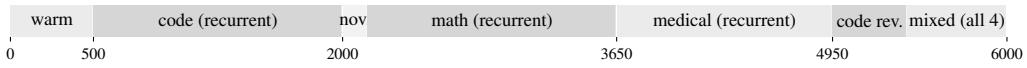

Figure 7: The seven-phase stream (steps). Evaluation happens at every phase boundary, every 250 steps, and every 50 steps within *code revisit*.

Student	Qwen3-0.6B (28 layers), bf16 weights and autocast
Adapters	LoRA $r = 16$, $\alpha = 32$, dropout 0, on q, k, v, o, gate, up, down
Optimizer	masked AdamW, lr 10^{-4} (all groups), $\beta = (0.9, 0.999)$, $\epsilon = 10^{-8}$, wd 0.01
Stream	6000 steps, batch 2, 512 tokens (prompt $\leq 50\%$), 7 phases (Figure 7)
Objective	$\tau = 4$, $\lambda_{\text{KD}} = 1$, $\lambda_{\text{CE}} = 0.5$ (confidence-weighted, floor 0.1), ℓ_1 on adapters 10^{-6}; no trust-region, anti-forgetting or consistency term
Teachers	Qwen3-4B, Qwen2.5-Coder-1.5B-Instruct, DeepSeek-R1-Distill-Qwen-1.5B, Qwen2.5-1.5B-Instruct
Shadow moments	stride-8 sketch, β as the optimizer, global bias-correction clock
Signals	surprise clip 20; $\bar{C}$ EMA 0.1 (cosine taken as 0 while $\|\hat{m}\| < 10^{-6}$); V EMA 0.1 on the full gradient norm; R weights (0.3, 0.5, 0.2)
Recurrence	R^{mom} EMA 0.05; bucket surprise EMA 0.1 per visit, $k = 10$, trend window 100 visits; embedding: mean-pooled input embedding of the batch's first row, window 1000, cos $\geq$ 0.85, $\kappa = 5$
Decision	$M = 6$; $\theta_C = 0.5$, $\theta_V = 0.3$, $R_{\text{commit}} = 0.5$; $\theta_S = 1.25$, $\theta_R = 0.40$; $K = 8/16$
Consolidation	every 250 steps, $R \geq 0.4 \wedge \bar{C} \geq 0.4$, direct merge, reset adapter state
Replay	$\geq$2 earlier deferrals of the bucket; 1 row per trigger; buffer 1000 (FIFO)
Evaluation	32 validation rows per domain (fixed); every 250 steps, every 50 in *code revisit*, all phase boundaries
Seeds	1337, 2024, 7 (stream sampling and adapter initialization; phase structure and validation rows fixed)

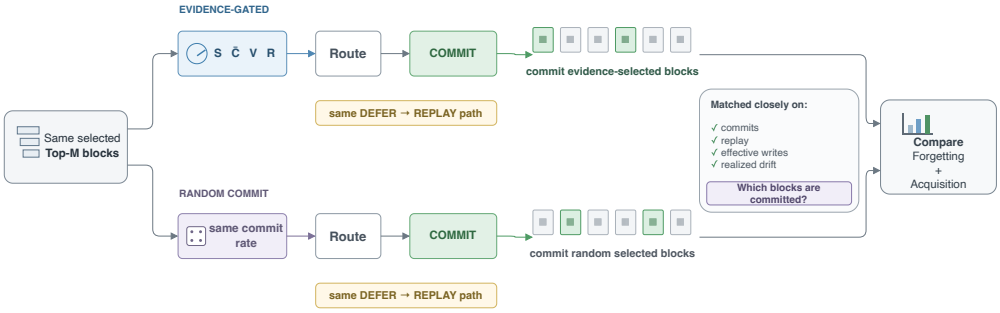

Figure 8: **Matched-volume control for selection.** The controller and the random-COMMIT control receive the same selected top-M blocks and keep the controller's DEFER–REPLAY path. The controller chooses permanent blocks from the evidence $(S, \bar{C}, V, R)$; the control commits randomly at the controller's per-phase commit rate. The runs match closely in commits, replay, effective writes and realized drift, so the comparison isolates whether *which* blocks are committed gives a measurable advantage (Section 5.1).

H ARM DEFINITIONS

Every arm uses the configuration above and changes only the entries listed here. **Reference (gated controller)**: as above, with replay on. **No bucket id**: recurrence weights $(0.6, 0, 0.4)$. **No defer tier**: DEFER disabled; such blocks fall through to WITHHOLD, and there is no replay. **Never commit**: COMMIT and CONSOLIDATE disabled, replay on. **Defer without replay**: replay off. **Surprise-only**: DEFER if $S \geq \theta_S$, otherwise WITHHOLD. It never commits and never consolidates, and replay is on. **Random relabel**: each selected module's label is drawn from the controller's per-phase shares of the three actions, seeded per step and rank. Replay is on. **Random commit**: modules are classified exactly as by the controller, so its defers and the replays they trigger are its own. the controller's commits are then withheld, and each non-deferred selected module is committed with probability $p_P/(p_F + p_P)$, taken from the controller's per-phase shares. **Always commit**: every selected block is committed. **MoLF EPD**: per block, the dense (base) expert wins if its expected preconditioned descent $\text{mean}(\hat{m}^2/(\sqrt{\hat{v}} + \epsilon))$ exceeds the adapter's; every block routes on every step; top-K = all ranks; no DEFER tier and no merge. **SNR commit rule** (the m^2/v variant used as a baseline by MoLF): COMMIT if $\text{clip}(\text{mean}(\hat{m}^2/(\hat{v} + \epsilon)), 0, 1) \geq \tau_{\text{SNR}}$, otherwise WITHHOLD. Top-M is the same as the controller's. It has no DEFER tier and no merge. Because top-M selects only base blocks,

it commits or does nothing. **Full fine-tuning**: controller and masking off, no adapters. **LoRA only**: controller off, base and shared frozen, all adapters open on every step. **Plateau (Online-LoRA)**: LoRA only, plus merge-and-reset of all adapters when a 200-step loss plateau (relative improvement $< 1\%$) follows a loss peak; it never fired. **Periodic merge**: LoRA only, plus merge-and-reset of all adapters every 250 steps. Two arms of the earlier campaign are excluded: a gold-label arm is scored on different targets, and a gradient-routing arm (Cloud et al., 2024) crashed and was resumed with a changed setting. **Shared only**: every selected block is withheld, and replay and consolidation are off, so only the shared parameters train. Ablations change one component of the controller: $\bar{C}$ pinned to pass (no $\bar{C}$), surprise pinned to θ_S, the optimizer's own masked moments instead of the shadow tracker, $M \in \{2, \text{all}\}$, $K \in \{25\%, 100\%\}$, instantaneous C, volatility pinned to pass, recurrence pinned to 0.5, no confidence gate or weighting, and strict consolidation (0.7/0.6, every 1000 steps). Noisy-teacher stream: in the math phase, half of the math buckets (and matching background steps) have their teacher answers shuffled among the corrupted rows.

I When the controller commits

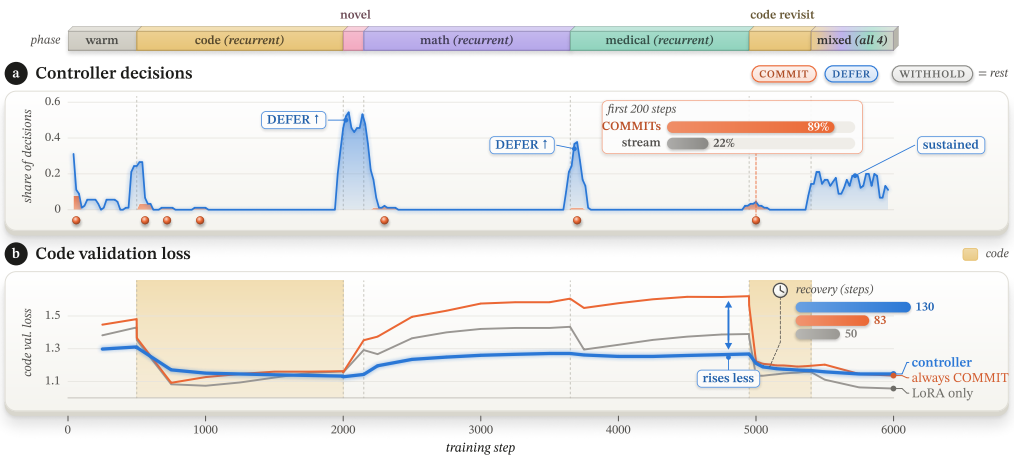

Figure 9: **(a)** Share of block decisions that DEFER (line) and COMMIT (fill) over the stream; the rest are WITHHOLD (reference, mean of 3 seeds, 100-step smoothing). Dots mark logged steps with a commit. Inset: 89% of sampled commits fall in the first 200 steps of a phase, which make up 22% of the stream. **(b)** Code validation loss (mean over seeds) for the reference, always-COMMIT and LoRA only; code phases shaded. Bars: steps to recover in the code revisit.

In the seed-1337 reference run, the 36000 block decisions split into 93.3% WITHHOLD, 6.6% DEFER and 0.17% COMMIT. WITHHOLD is the default (Figure 9). DEFER rises when the input distribution changes: in the novel phase, whose unique buckets are never replayed, at the switch to medical, and in the mixed tail. Commits cluster right after phase switches. 89% of sampled commits fall in the first 200 steps of a phase, windows that make up 22% of the stream. Code loss rises less under the controller during the math and medical phases than under LoRA only or always-commit. It recovers more slowly in the revisit (130 steps vs. 50 and 83).

J All arms and threshold sweeps

Exploratory ablations. Figure 10 and Table 3 (Section J) remove one component at a time. Apart from the agreement gate (above), each ablation is a single seed, so none can be resolved; we read them as directions. Reading the optimizer's own masked moments leaves the controller nearly blind, as Proposition 3 predicts: it commits once in 6000 steps. With the recurrence signal pinned, no block is ever deferred, nothing is replayed, and final loss rises by 0.039. The remaining ablations stay within 2δ on both axes. The dose-response of the $\bar{C}$ gate (Table 8, Figure 11b) is non-monotone. Never committing and never gating both forget more than the operating point, but the curve has only four points and one is a single seed.

Table 2: **Arms with fewer than three seeds** (exploratory; cannot be resolved). Columns as in Table 1.

			$\log_{10}$ routed base writes									
Arm	Surface	runs	commit	merge	perm.	eff.	$\log_{10}$ adapter	Replayed	M_1 (forgetting) ↓	seeds	Own-phase ↓	Final loss ↓
Controls on which updates become permanent												
SNR commit rule, τ_{SNR}=0.5	S+B	1	9.3	0	9.3	9.3	6.4	0	0.121 ≈	1/1	1.201	1.363 ≈
Single-seed ablations discussed in the text												
Controller, θ_C=0.35	S+B+A	1	9.4	9.3	9.6	9.5	8.2	294	0.081 ≈	0/1	1.115	1.244 ≈
Controller, top-M = all blocks	S+B+A	1	7.3	0	7.3	7.3	10.5	542	0.204 ≈	1/1	1.103	1.157 ≈
Controller, optimizer's own (masked) moments	S+B+A	1	0	7.9	7.9	7.4	9.1	2462	0.223 ≈	1/1	1.210	1.271 ≈

Table 3: **Ablations** (Δ relative to the reference pool; markers as in Table 1).

Change to the controller	n	ΔM_1	ΔFinal	$\log_{10}$ perm. writes	Merges
no $\bar{C}$ (agreement gate always passes)	3	+0.076 ▲	+0.031 ▲	10.4	971
no surprise	1	+0.031 ≈	+0.023 ≈	8.9	60
optimizer's own (masked) moments	1	+0.140 ≈	+0.025 ≈	7.9	12
select all blocks (Top-M = all)	1	+0.120 ≈	-0.089 ≈	7.3	0
instantaneous C instead of $\bar{C}$	1	+0.013 ≈	+0.004 ≈	9.6	88
no volatility	1	-0.003 ≈	+0.009 ≈	9.1	91
no recurrence signal	1	+0.005 ≈	+0.039 ≈	9.3	131
no confidence gate / weighting	1	+0.011 ≈	+0.001 ≈	9.2	126
Top-M = 2	1	-0.010 ≈	+0.001 ≈	9.3	179
Top-K = 25% of ranks	1	-0.008 ≈	+0.000 ≈	9.1	85
Top-K = 100% of ranks	1	+0.010 ≈	+0.003 ≈	9.1	101
strict consolidation (0.7 / 0.6, period 1000)	1	+0.004 ≈	+0.010 ≈	8.7	0
Reference pool	10	M_1 0.084 ±0.006		9.2	132

Table 4 lists every configuration on the clean stream, including single-seed threshold sweeps for the controller ($\theta_S \in \{1,2,4\} \times \theta_R \in \{0.2, 0.3, 0.45\}$), surprise-only ($\theta_S \in \{1,2,4\}$), plateau tolerance and MoLF-SNR ($\tau \in \{0.3, 0.5, 0.7\}$). The controller's own sweep spans M_1 0.080–0.105, so the operating point is not a fragile optimum. The gold-label arm is not listed because it is scored on different targets.

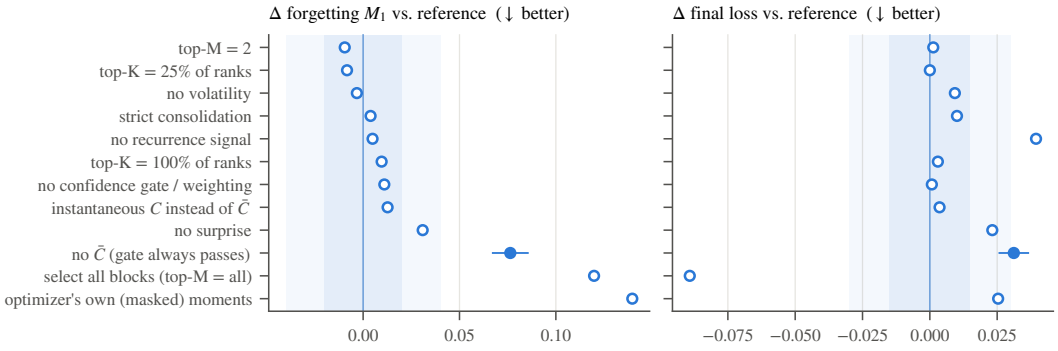

Figure 10: **Ablations of the controller** (numbers in Table 3). Each marker is the change in forgetting (left) or final loss (right) relative to the reference pool. Hollow markers are single seeds. The inner band is $\pm\delta$; the outer band, $\pm 2\delta$, is what a single-seed arm must exceed.

On the noisy-teacher stream (Table 9), the controller forgets 0.084 and the SNR rule 0.129; the controller forgets less in 9/9 run pairings (one run per seed per arm) ($p = 0.050$, the smallest attainable with three runs per arm). The confidence gate has no measurable effect, because it can only demote a COMMIT and commits are rare.

Exploratory runs after the protocol. Three kinds of run were designed or completed after the pre-declared results were known (PREREG addendum 4). None enters a table or a test. (i) A 1.7B student (Qwen3-1.7B) under the controller, one seed, on a rented A100, with the thresholds calibrated

Table 4: **Every configuration on the clean stream**, sorted by forgetting (single-seed sweeps included). $\log_{10}$ permanent writes are authorized routed base coordinate·steps.

Arm	n	M_1	median	Final	$\log_{10}$ perm. writes	Merges
Surprise-only gate, θ_S=4	3	0.074 ±0.003	0.072	1.262 ±0.009	0	0
Controller, Top-M = 2	1	0.074	0.074	1.247	9.3	179
Controller, Top-K = 25% of ranks	1	0.075	0.075	1.246	9.1	85
Controller, θ_S=2, θ_R=0.45	1	0.080	0.080	1.252	9.2	121
Controller, no volatility	1	0.081	0.081	1.255	9.1	91
Controller, θ_C=0.35	1	0.081	0.081	1.244	9.6	239
Gated controller (reference pool)	10	0.084 ±0.006	0.085	1.246 ±0.004	9.2	132
Shared parameters only	3	0.084 ±0.012	0.078	1.258 ±0.003	0	0
Random COMMIT, controller's own DEFER	3	0.084 ±0.004	0.086	1.252 ±0.007	9.3	216
Controller, θ_S=2, θ_R=0.2	1	0.084	0.084	1.272	9.3	144
forced merge at step 2075	1	0.085	0.085	1.244	9.2	154
Controller, θ_S=1, θ_R=0.45	1	0.086	0.086	1.237	9.1	97
Controller, no bucket id, θ_S=2	1	0.087	0.087	1.260	9.4	175
Controller, strict consolidation (0.7 / 0.6, period 1000)	1	0.088	0.088	1.256	8.7	0
Surprise-only, θ_S=1	1	0.089	0.089	1.253	0	0
Controller, no recurrence signal	1	0.089	0.089	1.285	9.3	131
Controller, no bucket id, θ_S=1	1	0.089	0.089	1.265	9.3	123
SNR commit rule, τ_{SNR}=0.7	3	0.089 ±0.022	0.102	1.291 ±0.055	8.7	0
Controller, no bucket id, θ_S=4	1	0.090	0.090	1.272	9.4	140
Surprise-only gate	3	0.090 ±0.006	0.092	1.266 ±0.001	0	0
Controller, θ_S=1, θ_R=0.3	1	0.091	0.091	1.267	9.2	154
Controller, no bucket-id recurrence	6	0.093 ±0.006	0.093	1.260 ±0.004	9.3	122
Controller, Top-K = 100% of ranks	1	0.093	0.093	1.249	9.1	101
Controller, no confidence gate / weighting	1	0.095	0.095	1.247	9.2	126
Controller, θ_S=1, θ_R=0.2	1	0.095	0.095	1.271	9.3	150
forced merge at step 2350	1	0.095	0.095	1.261	9.3	181
Controller, θ_S=4, θ_R=0.3	1	0.096	0.096	1.261	9.3	158
Controller, θ_S=4, θ_R=0.45	1	0.096	0.096	1.252	9.2	159
Controller, no DEFER tier (no replay)	6	0.096 ±0.008	0.095	1.279 ±0.011	9.3	168
Controller, instantaneous C instead of $\bar{C}$	1	0.096	0.096	1.250	9.6	88
Surprise-only, θ_S=2	1	0.097	0.097	1.262	0	0
Controller, DEFER without replay	3	0.097 ±0.014	0.103	1.280 ±0.005	9.3	185
Controller, θ_S=2, θ_R=0.3	1	0.099	0.099	1.274	9.3	155
commit/merge thresholds lowered (0.35)	1	0.101	0.101	1.242	9.8	53
Controller, never COMMIT/CONSOLIDATE	3	0.103 ±0.014	0.095	1.251 ±0.008	0	0
Controller, θ_S=4, θ_R=0.2	1	0.105	0.105	1.269	9.3	174
Controller, no surprise	1	0.115	0.115	1.269	8.9	60
same, commit disabled	1	0.115	0.115	1.260	0	0
forced merge at step 3850	1	0.118	0.118	1.312	9.3	169
SNR commit rule, τ_{SNR}=0.5	1	0.121	0.121	1.363	9.3	0
SNR commit rule, τ_{SNR}=0.3	1	0.148	0.148	1.274	9.3	0
Controller, no $\bar{C}$ (agreement gate always passes)	3	0.160 ±0.010	0.160	1.277 ±0.006	10.4	971
Random relabel of all decisions	3	0.164 ±0.072	0.137	1.287 ±0.033	9.3	175
LoRA only (plateau trigger never fired)	3	0.183 ±0.004	0.184	1.197 ±0.008	0	0
LoRA + plateau merge (tol. 0.01; never fired)	4	0.184 ±0.003	0.184	1.196 ±0.007	0	0
LoRA + plateau merge (tol. 0.02; never fired)	1	0.188	0.188	1.192	0	0
LoRA + plateau merge (tol. 0.005; never fired)	1	0.188	0.188	1.192	0	0
Controller, select all blocks (Top-M = all)	1	0.204	0.204	1.157	7.3	0
LoRA + merge every 250 steps	3	0.207 ±0.008	0.205	1.215 ±0.005	10.0	1344
Controller, optimizer's own (masked) moments	1	0.223	0.223	1.271	7.9	12
Always COMMIT	3	0.260 ±0.010	0.257	1.257 ±0.013	11.4	0
MoLF EPD rule	4	0.283 ±0.071	0.260	1.361 ±0.036	12.2	0
Full fine-tuning	3	0.873 ±1.019	0.311	1.484 ±0.139	12.4	0

Table 5: **Where the writes come from.** Commit and merge writes are routed base coordinate·steps; *full fine-tuning writes every base coordinate at every step, outside the controller. *Eff. merges*: merges whose adapter received a write since its last reset, out of all merges. Without replay the adapters are never written, so all merges are no-ops, although they are counted in Table 1. All write counts are $\log_{10}$ coordinate·steps. Shared coordinate·steps are 1.56×10^8 shared parameters times (steps + replay steps). For every row, $10^{\text{perm.}} = 10^{\text{commit}} + 10^{\text{merge}}$ in Table 1; the reconciliation in Table 13 checks this identity run by run. Controller time is the fraction of wall-clock; wall-clock is measured with three runs sharing one GPU.

Arm	$\log_{10}$ commit	$\log_{10}$ merge	Eff. merges	$\log_{10}$ shared coord·steps	Controller time	Wall-clock
Gated controller (reference pool)	8.7	9.0	24 / 132	12.0	44%	80 min
Controller, no bucket-id recurrence	9.1	9.0	18 / 122	12.0	45%	79 min
Controller, no DEFER tier (no replay)	8.9	9.1	0 / 168	12.0	41%	65 min
Controller, never COMMIT/CONSOLIDATE	0	0	–	12.0	38%	61 min
Surprise-only gate	0	0	–	12.0	38%	62 min
Surprise-only gate, θ_S=4	0	0	–	12.0	43%	81 min
Random COMMIT, controller's own DEFER	8.7	9.2	31 / 216	12.0	45%	88 min
Random relabel of all decisions	8.7	9.1	48 / 175	12.1	44%	91 min
Always COMMIT	11.4	6.5	0.0 / 0.3	12.0	42%	81 min
SNR commit rule, τ_{SNR}=0.5	9.3	0	–	12.0	40%	61 min
SNR commit rule, τ_{SNR}=0.7	8.7	0	–	12.0	45%	77 min
MoLF EPD rule	12.2	0	–	12.0	42%	72 min
Full fine-tuning	12.4*	0	–	12.0	0%	32 min
Shared parameters only	0	0	–	12.0	40%	89 min
LoRA only (plateau trigger never fired)	0	0	–	0	0%	44 min
LoRA + merge every 250 steps	0	10.0	424 / 1344	0	0%	156 min

Table 6: Differences from the reference pool with 95% Welch confidence intervals (arms with $n \geq 2$).

Arm	n	ΔM_1	95% CI	ΔFinal (95% CI)
Controller, no bucket-id recurrence	6	+0.009	[+0.002, +0.016]	+0.013 [+0.009, +0.018]
Controller, no DEFER tier (no replay)	6	+0.012	[+0.004, +0.021]	+0.033 [+0.022, +0.044]
Controller, never COMMIT/CONSOLIDATE	3	+0.019	[-0.012, +0.051]	+0.005 [-0.012, +0.023]
Surprise-only gate	3	+0.006	[-0.005, +0.017]	+0.020 [+0.016, +0.023]
Surprise-only gate, θ_S=4	3	-0.010	[-0.016, -0.004]	+0.016 [-0.004, +0.037]
Random COMMIT, controller's own DEFER	3	+0.000	[-0.008, +0.009]	+0.006 [-0.009, +0.022]
Random relabel of all decisions	3	+0.081	[-0.099, +0.260]	+0.041 [-0.040, +0.121]
Always COMMIT	3	+0.176	[+0.155, +0.197]	+0.011 [-0.020, +0.041]
SNR commit rule, τ_{SNR}=0.7	3	+0.006	[-0.048, +0.060]	+0.045 [-0.092, +0.182]
MoLF EPD rule	4	+0.200	[+0.086, +0.313]	+0.115 [+0.058, +0.172]
Full fine-tuning	3	+0.789	[-1.743, +3.320]	+0.238 [-0.108, +0.584]
Shared parameters only	3	+0.000	[-0.027, +0.027]	+0.012 [+0.006, +0.018]
LoRA only (plateau trigger never fired)	3	+0.099	[+0.091, +0.108]	-0.049 [-0.066, -0.032]
LoRA + merge every 250 steps	3	+0.123	[+0.106, +0.139]	-0.031 [-0.041, -0.021]

at 0.6B. It forgets 0.086 (0.6B pool: 0.084), reaches an own-phase loss of 0.934 and a final loss of 1.039, commits 228 times and replays 103 rows; the controller takes 19% of wall-clock. Its random-commit and shared-only controls were not run, so this run shows only that the controller transfers to a larger student with unchanged thresholds, not what the controller contributes there. Only its run summary and metrics were retrieved. (ii) Two reruns of the controller's configuration (seeds 1337 and 2024) on the same A100s forget 0.089, 0.098 and end at final losses of 1.252, 1.242. The first lies inside the pool's range [0.075, 0.092] and the second just above it, so a change of hardware adds spread beyond that of same-GPU reruns. (iii) A replay-yoked design that permutes the controller's own per-step labels among the selected blocks was started but not completed. Each shuffled cell has one seed (forgetting 0.121 and 0.090 at replay budgets of one and three rows), and the unshuffled cell at the higher budget was stopped before the end. The shuffle did not hold commit volume fixed (1.9× and 2.2× the controller's commits), so unlike the random-commit control it does not isolate allocation, and we draw no conclusion from it.

Table 7: Per-domain worst retention deltas and in-phase learning gains (the drop in a domain's loss over its own phase), mean over runs; n as in Table 1.

Arm	Worst retention Δ (general / code / math / medical)				In-phase gain (code / math / medical)		
Gated controller (reference pool)	0.114	0.138	0.051	0.031	0.179	0.223	0.764
Controller, no bucket-id recurrence	0.132	0.165	0.064	0.010	0.167	0.252	0.669
Controller, no DEFER tier (no replay)	0.154	0.169	0.073	-0.012	0.158	0.232	0.672
Controller, never COMMIT/CONSOLIDATE	0.147	0.204	0.045	0.016	0.122	0.175	0.747
Surprise-only gate	0.115	0.150	0.045	0.049	0.255	0.208	0.773
Surprise-only gate, θ_S=4	0.104	0.128	0.046	0.016	0.057	0.181	0.594
Random COMMIT, controller's own DEFER	0.090	0.166	0.051	0.031	0.171	0.194	0.760
Random relabel of all decisions	0.211	0.332	0.101	0.014	0.218	0.116	0.738
Always COMMIT	0.263	0.458	0.190	0.127	0.319	0.392	0.918
SNR commit rule, τ_{SNR}=0.5	0.168	0.212	0.085	0.019	0.075	0.201	0.526
SNR commit rule, τ_{SNR}=0.7	0.089	0.219	0.034	0.015	0.090	0.178	0.668
MoLF EPD rule	0.264	0.569	0.205	0.095	0.576	0.515	1.198
Full fine-tuning	0.790	1.307	0.864	0.528	0.642	0.747	1.025
Shared parameters only	0.126	0.137	0.038	0.036	0.035	0.171	0.594
LoRA only (plateau trigger never fired)	0.219	0.272	0.099	0.142	0.268	0.235	0.782
LoRA + merge every 250 steps	0.222	0.349	0.148	0.107	0.294	0.269	0.872

Table 8: Dose-response of the $\bar{C}$ gate. The commit threshold is θ_C and the consolidation threshold is $\theta_C - 0.1$. "Never commit" disables both; "$\bar{C}$ gate off" lets both always pass.

θ_C (commit)	n	$M_1 \downarrow$	Final $\downarrow$	$\log_{10}$ perm. writes
never commit	3	0.103 ±0.014	1.251 ±0.008	0
0.35	1	0.081	1.244	9.6
0.5	10	0.084 ±0.006	1.246 ±0.004	9.2
$\bar{C}$ gate off	3	0.160 ±0.010	1.277 ±0.006	10.4

K SEED-LEVEL STATISTICS AND WRITE RECONCILIATION (POST HOC)

Everything in this section was computed after the pre-declared results were known, by `scripts/revision.py`, which the same `make results` step runs. It extends the pre-declared analysis and replaces none of it.

Robustness to the forgetting metric. M_1 is a mean of per-domain maxima over 32 validation rows. Across the 18 arms with three seeds it ranks arms almost exactly as three alternatives computed from the same retention histories: Spearman 0.95 with the mean retention delta, 0.94 with the worst domain's delta, and 0.86 with backward transfer (and standard average forgetting) at the end of the code revisit. Across all 53 configurations these are 0.93, 0.90 and 0.63. No arm that differs from the reference by at least δ_M changes sign under any of them.

L TEXT-ONLY SUPERVISION

An earlier campaign used the same stream with the cached teacher text as the only target: cross-entropy only, because no teacher logits were available, so the KL term was zero. It also used different routing thresholds and CE weight. In that campaign, the variant without the defer tier and the MoLF-SNR rule both forgot less than the controller. With the KL term active, both orderings reverse (Section 5). Several factors changed at once, so we do not attribute the reversal to distillation alone. We report it as evidence that controller rankings are regime-dependent (Table 15).

M ATTRIBUTION AND SELECTIVE ROLLBACK

In the text-only campaign, the corrupted teacher's rows carry their own teacher tag, so the ledger can follow them. Under the controller, none of that teacher's decisions reached the base weights through COMMIT, and it touched zero base coordinates directly. The two-tier MoLF-SNR rule contained it equally. The EPD rule, which commits most blocks, sent 81% of that teacher's decisions to the base. To exercise attribution, we forced a merge of all adapters inside the corrupted phase. The ledger then attributed each merge to the teachers whose replayed rows had written the adapter. Restoring

Table 9: **Noisy-teacher stream** (half of the math buckets carry shuffled teacher answers). The last column counts run pairings (one run per seed per arm) in which the controller forgets less.

Arm (noisy-teacher stream)	n	$M_1 \downarrow$	Final $\downarrow$	$\log_{10}$ perm. writes	controller better
Gated controller	3	0.084 $\pm$0.003	1.245 $\pm$0.008	9.2	–
Controller, confidence gate off	3	0.080 $\pm$0.007	1.244 $\pm$0.003	9.2	3/9
Controller, corrupted rows at confidence 0.2	3	0.077 $\pm$0.003	1.242 $\pm$0.006	9.2	0/9
Controller, all adapters merged at step 3000	3	0.086 $\pm$0.009	1.240 $\pm$0.006	9.3	5/9
SNR commit rule, τ_{SNR}=0.5	3	0.129 $\pm$0.026	1.283 $\pm$0.006	9.1	9/9
LoRA only	3	0.176 $\pm$0.006	1.195 $\pm$0.008	0	9/9
MoLF EPD rule	3	0.251 $\pm$0.031	1.356 $\pm$0.020	12.2	9/9
Full fine-tuning	3	0.270 $\pm$0.019	1.411 $\pm$0.020	12.4	9/9

Table 10: **Variance decomposition** of the reference pool: one-way random effects with the stream seed as the factor. "Share stream" is $\sigma^2_{\text{stream}}/(\sigma^2_{\text{stream}} + \sigma^2_{\text{GPU}})$.

Metric	σ_{stream}	σ_{GPU}	share stream	$F_{2,7}$	per-seed pool runs (mean)
M_1	0.0035	0.0052	31%	2.5 ($p = 0.156$)	7: 3 (0.081); 1337: 4 (0.088); 2024: 3 (0.080)
Final loss	0.0019	0.0040	18%	1.7 ($p = 0.247$)	7: 3 (1.245); 1337: 4 (1.249); 2024: 3 (1.243)

the pre-merge checkpoint of the attributed merges and replaying the rest of the stream with those merges blocked completed end to end, and it changed the final losses by at most 0.007. Attribution and rollback are therefore mechanically sound here. On this stream the damage they undo is small, and we do not claim a recovery benefit.

Table 11: **The five confirmatory tests** as pre-declared (run unit, exact permutation over pooled runs, Holm over five), next to the same one-sided question with the stream seed as the unit: exact sign-flip test on per-seed paired differences (minimum $p = 1/8$) and an exact permutation test on seed means (minimum $1/20$). "Seeds worse" counts seeds in which the arm's seed-mean M_1 exceeds the reference's.

Test	Arm	runs	pre-declared, run unit (Holm)	seed unit, paired (Holm)	seed unit, unpaired (Holm)	seeds worse
H1	Random relabel of all decisions	3	0.017	0.125 (0.625)	0.050 (0.250)	3/3
H1b	Random COMMIT, controller's own DEFER	3	0.472	0.500 (1.000)	0.500 (0.500)	2/3
H2	Always COMMIT	3	0.017	0.125 (0.625)	0.050 (0.250)	3/3
H3	Controller, no $\bar{C}$ (agreement gate always passes)	3	0.017	0.125 (0.625)	0.050 (0.250)	3/3
H4	SNR commit rule, τ_{SNR}=0.5	1	0.182	0.500 (1.000)	0.250 (0.500)	1/1

Table 12: **Seed-level paired differences** from the reference for every arm run on all three seeds: per-seed differences of seed means, their mean and a 95% t interval (2 degrees of freedom), and the number of seeds in which the arm is worse. This is the convention used throughout the main text. The pre-declared markers of Table 1 are kept unchanged.

Arm	runs	ΔM_1 vs. reference, per seed			ΔFinal vs. reference, per seed		
		mean [95% CI]	worse	1337 / 2024 / 7	mean	[95% CI]	worse
Surprise-only gate, θ_S=4	3	-0.010 [-0.025, +0.006]	0/3	-0.016 / -0.003 / -0.010	+0.017	[-0.001, +0.035]	3/3
Shared parameters only	3	+0.001 [-0.037, +0.039]	1/3	-0.012 / +0.018 / -0.003	+0.012	[+0.005, +0.019]	3/3
Random COMMIT, controller's own DEFER	3	+0.001 [-0.021, +0.023]	2/3	-0.009 / +0.008 / +0.004	+0.007	[-0.005, +0.018]	3/3
SNR commit rule, τ_{SNR}=0.7	3	+0.006 [-0.060, +0.073]	2/3	-0.025 / +0.023 / +0.020	+0.045	[-0.086, +0.176]	3/3
Surprise-only gate	3	+0.006 [-0.004, +0.017]	3/3	+0.005 / +0.003 / +0.011	+0.020	[+0.011, +0.029]	3/3
Controller, no bucket-id recurrence	6	+0.010 [-0.005, +0.025]	3/3	+0.005 / +0.007 / +0.017	+0.014	[+0.008, +0.019]	3/3
Controller, no DEFER tier (no replay)	6	+0.013 [-0.005, +0.031]	3/3	+0.009 / +0.021 / +0.008	+0.034	[+0.001, +0.066]	3/3
Controller, DEFER without replay	3	+0.014 [-0.018, +0.045]	3/3	+0.015 / +0.001 / +0.026	+0.034	[+0.015, +0.053]	3/3
Controller, never COMMIT/CONSOLIDATE	3	+0.020 [-0.003, +0.043]	3/3	+0.031 / +0.015 / +0.014	+0.006	[-0.007, +0.018]	3/3
Controller, no $\bar{C}$ (agreement gate always passes)	3	+0.077 [+0.043, +0.111]	3/3	+0.062 / +0.090 / +0.079	+0.031	[+0.015, +0.048]	3/3
Random relabel of all decisions	3	+0.081 [-0.089, +0.251]	3/3	+0.158 / +0.057 / +0.028	+0.041	[-0.033, +0.115]	3/3
LoRA only (plateau trigger never fired)	3	+0.100 [+0.090, +0.109]	3/3	+0.098 / +0.104 / +0.096	-0.048	[-0.069, -0.027]	0/3
LoRA + merge every 250 steps	3	+0.123 [+0.095, +0.151]	3/3	+0.111 / +0.125 / +0.134	-0.031	[-0.046, -0.016]	0/3
Always COMMIT	3	+0.176 [+0.147, +0.206]	3/3	+0.169 / +0.190 / +0.170	+0.011	[-0.026, +0.048]	2/3
MoLF EPD rule	4	+0.192 [+0.137, +0.248]	3/3	+0.218 / +0.179 / +0.180	+0.117	[+0.013, +0.221]	3/3
Full fine-tuning	3	+0.789 [-1.749, +3.328]	3/3	+0.169 / +1.969 / +0.230	+0.238	[-0.113, +0.590]	3/3

Table 13: **Write reconciliation.** Routed base writes as authorized (commit, merge, total) and effective (commits plus merges of written adapters; * full fine-tuning's unmasked writes). *eff./all*: effective over all merges per run. *adapter* and *shared*: $\log_{10}$ coordinate·steps on those surfaces. *Realized*: $\|\theta_T - \theta_0\|_2$ of routed and shared weights against the pretrained model, averaged over the runs whose final weights were archived; a superscript gives that number when it is smaller than n, and – means none was archived. Per-merge delta norms and ℓ_1 drift are not in the snapshot. The identity perm. = commit + merge holds in all 104 runs.

		Routed base writes ($\log_{10}$ coord·steps)				Merges		Other surfaces		Realized ℓ_2 drift	
Arm	n	commit	merge	total	effective	eff./all	replayed	adapter	shared	routed	shared
Gated controller (reference pool)	10	8.7	9.0	9.2	8.9	24/132	274	8.2	12.0	$2.77^{(3)}$	179
Controller, no DEFER tier (no replay)	6	8.9	9.1	9.3	8.9	0/168	0	4.4	12.0	–	–
Random COMMIT, controller's own DEFER	3	8.7	9.2	9.3	8.9	31/216	272	8.1	12.0	2.80	179
Random relabel of all decisions	3	8.7	9.1	9.3	8.9	48/175	1305	8.8	12.1	–	–
Controller, no $\bar{C}$ (agreement gate always passes)	3	10.3	9.9	10.4	10.3	171/971	316	8.2	12.0	$8.63^{(2)}$	176
Controller, θ_C=0.35	1	9.4	9.3	9.6	9.5	47/239	294	8.2	12.0	4.23	178
SNR commit rule, τ_{SNR}=0.7	3	8.7	0	8.7	8.7	–	0	5.4	12.0	$2.18^{(2)}$	178
Always COMMIT	3	11.4	6.5	11.4	11.4	0.0/0.3	0	0	12.0	–	–
MoLF EPD rule	4	12.2	0	12.2	12.2	–	0	10.4	12.0	–	–
Full fine-tuning	3	12.4^{*}	0	12.4	12.4	–	0	0	12.0	–	–
Controller, never COMMIT/CONSOLIDATE	3	0	0	0	0	–	284	8.2	12.0	–	–
Surprise-only gate, θ_S=4	3	0	0	0	0	–	133	7.8	12.0	$0.00^{(2)}$	181
Shared parameters only	3	0	0	0	0	–	0	0	12.0	0.00	183
LoRA only (plateau trigger never fired)	3	0	0	0	0	–	0	10.8	0	–	–
LoRA + merge every 250 steps	3	0	10.0	10.0	9.5	424/1344	0	10.8	0	21.46	0
Controller, select all blocks (Top-M = all)	1	7.3	0	7.3	7.3	–	542	10.5	12.0	–	–
Controller, optimizer's own (masked) moments	1	0	7.9	7.9	7.4	3/12	2462	9.1	12.1	–	–

Table 14: **Which write volume predicts forgetting?** OLS of M_1 on $\log_{10}(\cdot + 1)$ of four write volumes across configurations (full fine-tuning and the never-fired plateau arms excluded). Partial R^2 is drop-one; VIF from the other predictors. Shared coordinate·steps are omitted: they equal a constant times (6000 + replayed rows) in every arm that trains the shared parameters, so they are collinear with replay. With 16 and 49 points these fits are descriptive.

	arms with $n \geq 3$ (16 arms, $R^2 = 0.41$)				all configurations (49, $R^2 = 0.19$)			
$\log_{10}(\cdot + 1)$ of	b	p	partial R^2	VIF	b	p	partial R^2	VIF
commit coord·steps	+0.005	0.178	0.16	1.2	+0.001	0.623	0.01	1.5
effective merge coord·steps	-0.000	0.964	0.00	1.7	-0.003	0.241	0.03	1.9
adapter coord·steps	+0.009	0.108	0.22	1.4	+0.008	0.029	0.10	1.6
replayed rows	-0.029	0.071	0.27	1.5	-0.018	0.034	0.10	1.8

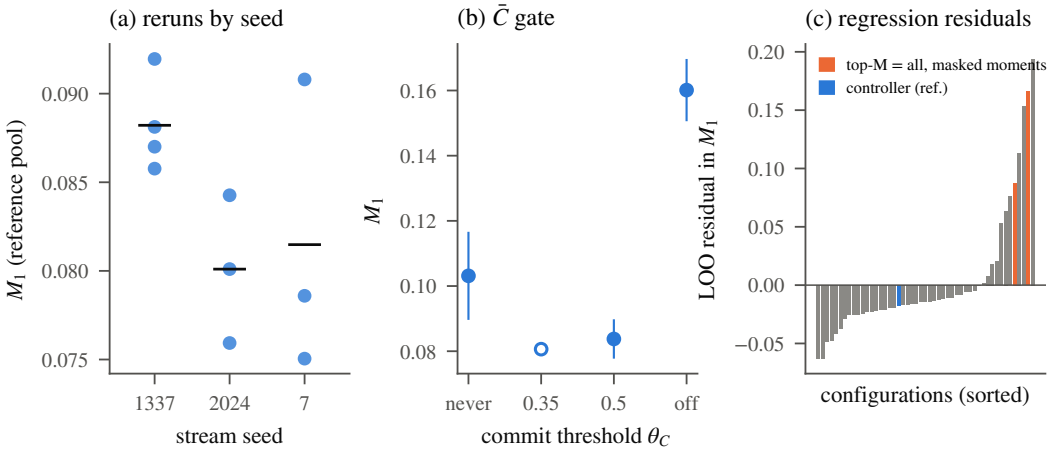

Figure 11: (a) M_1 of the reference pool's runs by stream seed; bars are seed means. (b) Dose-response of the $\bar{C}$ gate (Table 8); the hollow point is a single seed. (c) Leave-one-out residuals of the all-configuration regression in Table 14, sorted; the largest positive residuals belong to always-commit, the masked-moment ablation, MoLF EPD, periodic merging and top-M = all, so the log-linear fit underpredicts forgetting at high write volume and for both counterexamples.

Table 15: Text-only targets (KL inactive; earlier campaign) against live KL distillation (this paper), same stream. Thresholds and CE weight also differ between the campaigns, so the columns are not a controlled comparison.

Arm	M_1 text-only	M_1 live KL	Final text-only	Final live KL
Gated controller (reference pool)	0.113 ±0.007	0.084 ±0.006	1.137 ±0.009	1.246 ±0.004
Controller, no DEFER tier (no replay)	0.071 ±0.008	0.096 ±0.008	1.136 ±0.011	1.279 ±0.011
Controller, never COMMIT/CONSOLIDATE	0.110 ±0.008	0.103 ±0.014	1.141 ±0.011	1.251 ±0.008
Surprise-only gate	0.144 ±0.089	0.090 ±0.006	1.127 ±0.002	1.266 ±0.001
Random relabel of all decisions	0.301 ±0.048	0.164 ±0.072	1.160 ±0.006	1.287 ±0.033
LoRA only (plateau trigger never fired)	0.184 ±0.043	0.183 ±0.004	0.993 ±0.003	1.197 ±0.008
MoLF EPD rule	0.459 ±0.119	0.283 ±0.071	1.414 ±0.016	1.361 ±0.036
Always COMMIT	0.511 ±0.191	0.260 ±0.010	1.197 ±0.007	1.257 ±0.013
Gated controller, noisy stream	0.109 ±0.005	0.084 ±0.003	1.136 ±0.007	1.245 ±0.008
SNR commit rule, τ_{SNR}=0.5, noisy stream	0.039 ±0.004	0.129 ±0.026	1.117 ±0.007	1.283 ±0.006

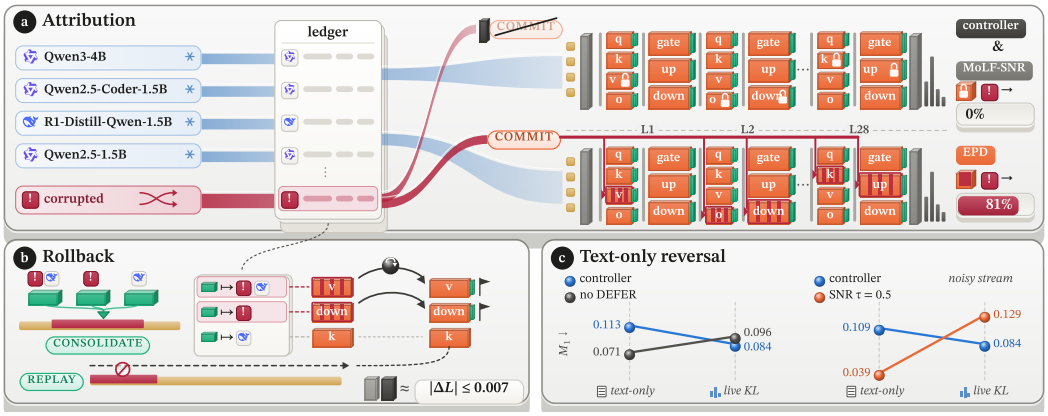

Figure 12: **Attribution, rollback and the text-only reversal** (text-only campaign). (a) The teacher-tagged ledger follows a corrupted teacher: none of its decisions reach the base weights under the controller or the MoLF-SNR rule, while the EPD rule commits 81% of them. (b) Reverting the merges attributed to it and replaying the rest of the stream changes final losses by at most 0.007. (c) Two orderings reverse between text-only targets and live KL (Table 15): the controller against the no-DEFER-tier arm, and against the SNR rule ($\tau_{\mathrm{SNR}} = 0.5$) on the noisy stream.