RANKRECLAIM: COUNTING THE CAPACITY AN EDITED LAYER HAS ACTUALLY SPENT

Anonymous authors
Paper under double-blind review

ABSTRACT

Lifelong model editing corrects a deployed model many times, and each correction arrives as a low-rank update to one layer of fixed width, the resource every later correction draws on. Editors decide when the layer is full by summing the ranks the corrections request, yet corrections share input directions: on one stream the requested total reaches 48 by the eighteenth arrival while only 30 directions are committed, and the consolidations that return width are executed before their cost to earlier corrections is read. We propose RankReclaim, which counts the directions a layer has actually committed instead of summing requested ranks, and prices every consolidation before it executes. A committed-capacity ledger reads the numerical rank of the stacked input factors and attributes to each correction the directions its removal would return, testing admission against the measured count. Priced reclamation evaluates a candidate truncation on the stored probes of every surviving correction and executes it only at the smallest rank whose accumulated displacement stays inside a tolerance, quoting the cost of returned width before it is paid. Overlap-budgeted placement splits each arrival's fit against the committed span and caps the share drawn from the free width, holding every arrival's draw at or under a set budget. The three components share one committed basis per layer and act on the one count the ledger produces. On three edit streams run to three times the per-layer budget, the requested-rank account serves 6, 7 and 6 corrections, the measured count alone 9, 8 and 13 at zero displacement, and RankReclaim 13, 15 and 15. Among eight arms, RankReclaim is the only reclaiming arm that holds the committed count at the budget, 16, 15 and 16 of 16, and the only one that reads the displacement of every consolidation before executing it, returning width at 26% to 36% of the displacement of a fixed-rank consolidation.

1 INTRODUCTION

Model editing has become the standard means of correcting a deployed model without retraining (Yao et al., 2023), and a deployed model is corrected many times (Meng et al., 2022; 2023; Mitchell et al., 2022). Each correction arrives as a low-rank update (Hu et al., 2022) at one fixed-width layer with a requested rank (Yu et al., 2024), and practitioners sum the requests to decide when the layer is full. That sum is the wrong quantity. Corrections revisit the same facts (Li et al., 2024); their input factors therefore share directions, and a sum over requests counts the same direction more than once. On one editing stream, the requested total reaches 48 by the eighteenth arrival while the directions actually committed number 30. The gap originates in the arriving corrections rather than in any policy that handles them, and no component of an editing stack (Wang et al., 2024) reports it to the party spending the budget.

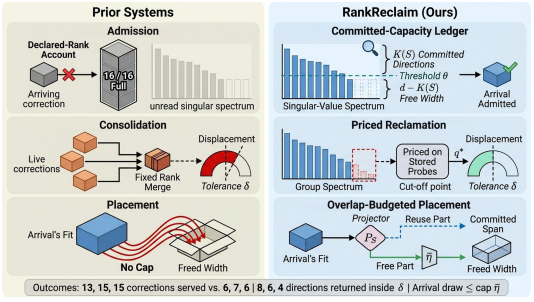

Figure 1: **Left:** prior editors admit on requested ranks, merge at a fixed rank, place uncapped. **Right:** RankReclaim counts committed directions, prices each consolidation on stored probes before it runs, and caps the free-width draw.

Two families of editor spend a layer's capacity, and each carries a principled account of how it spends it. The allocation-only family writes each correction in closed form at the position the layer's own

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.

statistics point to (Meng et al., 2022; 2023), and learned editor networks amortize that placement across arrivals (Mitchell et al., 2022); the budget is respected by construction, since every arrival requests its rank before it is written. The second family also returns capacity by consolidating live corrections into a single update of lower rank (Yadav et al., 2023), and it inherits a mature account of what a truncation costs: second-order saliency prices a removal by the error it leaves behind (Hassibi & Stork, 1992), one-shot pruning selects removals against that reconstruction error at scale (Frantar & Alistarh, 2023), and rank-level compression treats the rank itself as the resource spent (Ashkboos et al., 2024; Zhang et al., 2023). The second family rests on the observation that a nominal parameter count and the count that carries the function are different numbers (Frankle & Carbin, 2019). Despite these advances, neither family measures the capacity a layer has spent.

However, the three decisions a long edit stream (Hartvigsen et al., 2023) forces are all taken on numbers that were never measured. Admission is tested against a sum of requests rather than a count of directions, and the layer is therefore judged full while directions remain. A consolidation is executed before its price is read, and the rank that returns the most capacity per unit of rank is not the rank that costs the survivors least. Whatever capacity does come back is then spent by the arrival it was freed for, since an unconstrained least-squares fit is indifferent to which directions it uses. The requested total and the committed count are not two estimates of one quantity: the first is a promise an arrival makes and the second is the width a layer has spent, and a policy that admits, reclaims and places on the promise manages a resource that has never been counted.

To address these three decisions, this paper proposes RankReclaim, which counts the directions a layer has actually committed instead of summing requested ranks, and prices every consolidation before it executes. Its central insight is that the committed count is an operator rather than a book-keeping entry: once a layer can read how many directions it has spent, admission, reclamation and placement all act on that one count. RankReclaim comprises three composed components: (1) a committed-capacity ledger reads the directions the live set occupies and attributes them to individual corrections, such that admission is tested against the measured count; (2) priced reclamation evaluates a candidate truncation on the survivors' stored probes and executes it only at a rank whose accumulated displacement stays inside a tolerance, which quotes capacity before it is bought; and (3) overlap-budgeted placement splits an arrival's fit exactly against the committed span and caps the share drawn from the free width, which holds every arrival's draw at or under a set budget. The main contributions are as follows:

- Three unmeasured quantities in lifelong low-rank editing are identified: capacity counted by request rather than by measurement, consolidation executed before its displacement price is read, and reclaimed width re-spent by the arrival it was freed for.
- Admission at a fixed-width layer is formulated as accounting over committed directions, and RankReclaim is proposed to address measurement, pricing and placement jointly through three composed components.
- Experiments on edit streams run to three times the per-layer budget demonstrate that the measured count alone raises the corrections served from 6, 7 and 6 to 9, 8 and 13 at zero displacement and RankReclaim to 13, 15 and 15, and that RankReclaim is the only reclaiming arm among eight that holds the committed count at the budget while reading the displacement of every consolidation before it executes.

2 MOTIVATION

A series of measurements on the arriving corrections examines how an edit stream spends a layer's capacity and identifies three challenges.

Challenge #1: Uncounted Capacity. Allocation-only editors add the requested ranks up and treat that sum as an upper bound tight enough to plan with, and a layer is therefore called full once the requests reach its width. To quantify this assumption, the arriving input factors are stacked and the numerical rank of the stack is read. Figure 2 (left) shows that the requested total stands above the committed count at every reading after the fourth, reaching 48 against 30 by the eighteenth arrival. Once the two curves part, they never close again. The cause is structural rather than statistical, since a requested rank is a property of one correction in isolation while capacity is a property of the set; admission control on the requested total therefore turns the layer away while directions remain and enforces the budget on the wrong number. *What should a layer count when it decides that it is full?*

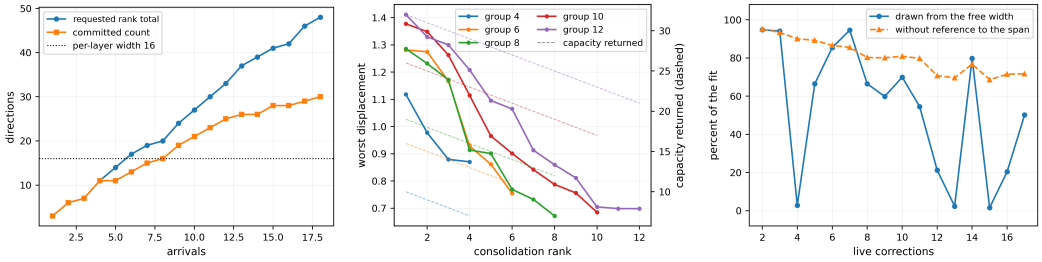

Figure 2: **Left:** requested against committed count. **Middle:** displacement and returned capacity against rank. **Right:** free-width share of the fit. Details in Appendix A.2.

Challenge #2: Unpriced Consolidation. Merging editors make room by consolidating a group of corrections into one update of lower rank, and they choose the move by how much room it makes. To price the move before it is executed, the residual of a candidate truncation is evaluated directly on the corrections' own probes. Figure 2 (middle) shows that consolidating eight corrections at rank 1 displaces the worst survivor by 1.285804 while rank 8 displaces it by 0.671775, a factor of 1.91. More critically, the room-made-per-rank view orders the two the other way, since it reads only the capacity returned. The root cause is that the returned capacity being optimized and the displaced behaviour being spent are read at different times; selecting a consolidation by capacity returned per rank therefore never reads the displacement, although it is computable when the choice is made. *What does a consolidation cost before it is paid for?*

Challenge #3: Unbudgeted Placement. Once a consolidation has returned directions, the next arrival is fitted into the layer by least squares, and least squares is indifferent to which directions it uses. To price the two halves of that fit apart, the fit is split exactly against the committed span and the share landing outside is read. Figure 2 (right) shows that the drawn share runs from 1.5293 percent at 15 live corrections to 94.7507 percent at 2. More critically, it tracks neither the live count nor the free width, but only which topics the next arrival happens to need. The cause is that nothing in the placement objective refers to the free width at all; consolidation followed by unconstrained least-squares placement therefore lets the second step draw back the width the first returns. *How much of the freed width may an arrival draw?*

3 Problem Formulation

To address the three challenges of Section 2, lifelong low-rank editing is formulated as a constrained maximization of expected end-of-stream quality over admission policies:

$$\max_{\pi \in \Pi} \mathbb{E}\left[\sum_{i=1}^{T} Q_i(\pi)\right] \qquad \text{s.t.} \qquad K(S) \le d, \quad \max_{i \in S} L_i \le \delta, \quad \eta(A) \le \bar{\eta} \tag{1}$$

where π is the admission policy, Π the policies meeting the three conditions, T the stream length, Q_i the end-of-stream quality of correction i, S the live set, $K(S)$ the directions it occupies under the per-layer budget d, L_i the displacement a survivor bears under the tolerance δ, and $\eta(A)$ the share of an arrival's factor A outside the span of S, under the budget $\bar{\eta}$.

The budget $K(S) \le d$ addresses Challenge #1: what a layer admits is bounded by the directions it has spent, not by the ranks it was promised. The tolerance $\max_{i \in S} L_i \le \delta$ addresses Challenge #2: what a consolidation does is bounded by what it costs the survivors. The budget $\eta(A) \le \bar{\eta}$ addresses Challenge #3: what an arrival draws is bounded by the free-width budget.

4 Method

4.1 RankReclaim Overview

RankReclaim counts the directions a layer has actually committed instead of summing requested ranks, and prices every consolidation before it executes. It addresses the admission problem the objective states: how a fixed-width layer decides what to admit, what to consolidate and where to place it, when the capacity spent is not the capacity requested. As shown in Figure 3, it comprises three components: a committed-capacity ledger, priced reclamation, and overlap-budgeted placement.

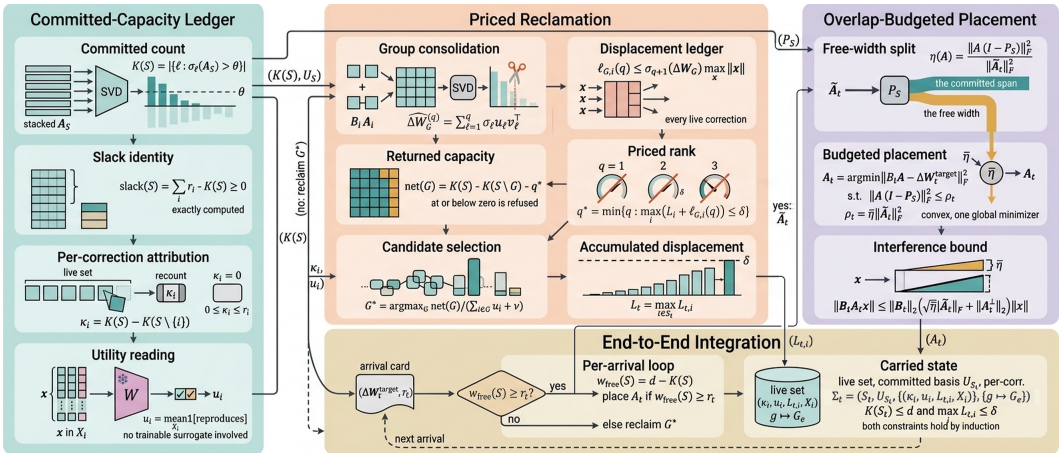

Figure 3: **Left:** the ledger counts committed directions. **Middle:** reclamation truncates only inside the tolerance. **Right:** placement caps the free-width draw. **Bottom:** the per-arrival loop.

The three are one system rather than three modules: they share one state per layer, the committed basis and each correction's own probes. The ledger produces the count the other two trade against, reclamation spends it to buy width back, and placement caps what an arrival may take. One component answers each condition of the objective, and the sections below expand them in order.

4.2 Committed-Capacity Ledger

To address Challenge #1, the committed-capacity ledger measures the directions a live set occupies. As shown in Figure 3 (left), it reads that count, takes its slack against the requested total, attributes it to individual corrections, and reads each correction's utility on its own probes.

Committed count. The committed directions of a live set are the row space of its stacked input factor, and the committed count is the numerical rank of that stack at a fixed spectral tolerance,

$$A_S = \begin{bmatrix} A_{i_1} \\ \vdots \\ A_{i_{|S|}} \end{bmatrix}, \qquad K(S) = |\{\, \ell : \sigma_\ell(A_S) > \theta \,\}|, \tag{2}$$

where A_S is the stacked input factor of the live set S, $\sigma_\ell(A_S)$ its singular values, θ the spectral tolerance and $K(S)$ the resulting committed count. The requested total $\sum_{i\in S} r_i$ upper-bounds $K(S)$ and meets it only when the row spaces of the live corrections form a direct sum.

Slack identity. The gap between the requested total and the committed count is read as an exactly computed, non-negative slack,

$$\mathrm{slack}(S) = \sum_{i\in S} r_i - K(S) \;\geq\; 0, \tag{3}$$

where r_i is the requested rank of correction i and $\mathrm{slack}(S)$ the capacity the requested total overstates, read off the singular spectrum of the stored matrices and spent later by reclamation.

Per-correction attribution. The ledger attributes to each live correction the number of committed directions its removal would return, which is the quantity a reclamation policy needs rather than the correction's requested size,

$$\kappa_i = K(S) - K(S \setminus \{i\}), \qquad 0 \leq \kappa_i \leq r_i, \tag{4}$$

where κ_i is the marginal committed count of correction i, exact and lying in $[0, r_i]$. A correction with $\kappa_i = 0$ occupies the ledger while returning nothing, which is exactly the case a reclamation policy must be able to see.

Utility reading. Utility is computed by running the current model on each correction's own stored probes and reading the behaviour the correction was installed for, with no trainable surrogate involved,

$$u_i = \frac{1}{|X_i|} \sum_{x \in X_i} \mathbb{1}\left[\text{the model with } S \text{ reproduces on } x \text{ what correction } i \text{ specified}\right], \tag{5}$$

where X_i is the probe set of correction i and u_i its measured utility on those probes, the run-time reading of Q_i in Eq. (1). A merge that destroys one member's behaviour shows as that member's utility falling rather than as its removal. Details in Appendix A.4.

4.3 Priced Reclamation

To address Challenge #2, priced reclamation computes what a consolidation costs the survivors before it is executed. As shown in Figure 3 (middle), it forms a group, prices its truncation on stored probes, admits the smallest rank inside the tolerance, and charges it to a stream ledger.

Group consolidation. A candidate consolidation is formed as the rank-q truncation of the group's combined update, taken from its own singular triplets,

$$\Delta W_G = \sum_{i \in G} B_i A_i, \qquad \widehat{\Delta W}_G^{(q)} = \sum_{\ell=1}^{q} \sigma_\ell u_\ell v_\ell^\top, \tag{6}$$

where ΔW_G is the combined update of the group G, $B_i A_i$ the contribution of correction i, and $\widehat{\Delta W}_G^{(q)}$ its rank-q truncation with singular triplets $(\sigma_\ell, u_\ell, v_\ell)$ of ΔW_G. The truncation error is exactly the discarded spectrum, which is what lets the price be computed rather than modelled.

Displacement ledger. The displacement of a consolidation on each live correction is obtained by evaluating the residual matrix on that correction's stored probes, over every live correction rather than only the group's members,

$$\ell_{G,i}(q) = \max_{x \in X_i} \left\| \left(\Delta W_G - \widehat{\Delta W}_G^{(q)} \right) x \right\| \;\le\; \sigma_{q+1}(\Delta W_G) \max_{x \in X_i} \|x\|, \qquad i \in S_{t^-}, \tag{7}$$

where $\ell_{G,i}(q)$ is the displacement correction i accumulates from the event, S_{t^-} the live set just before it, and $\sigma_{q+1}(\Delta W_G)$ the first discarded singular value. The residual acts on every input, and a survivor outside G is displaced by the same matrix as a member, which is why the evaluation domain is the whole live set.

Priced rank. The consolidation rank is the smallest rank at which the accumulator plus the displacement of this event stays inside the tolerance for every survivor,

$$q^\star(G) = \min \left\{ q : \max_{i \in S_{t^-}} \left(L_i + \ell_{G,i}(q) \right) \le \delta \right\}, \tag{8}$$

where L_i is the accumulator of Eq. (11) read before the event and δ the displacement tolerance. The test is against the residual $\delta - L_i$ rather than the whole of δ, because earlier events have already drawn on the budget.

Returned capacity. How much capacity a candidate returns is resolved as the directions the group held less the rank its consolidation keeps,

$$\mathrm{net}(G) = (K(S) - K(S \setminus G)) - q^\star(G), \tag{9}$$

where $\mathrm{net}(G)$ scores group G and is a lower bound on the capacity the merge returns, since the $q^\star(G)$ directions it keeps are charged in full; a group scoring at or below zero is refused.

Candidate selection. The executed group is the one returning the most capacity per unit of utility placed at risk, over a pool fixed by an explicit formation rule,

$$G^\star = \arg\max_{G \in \mathcal{C}} \frac{\mathrm{net}(G)}{\sum_{i \in G} u_i + \nu}, \tag{10}$$

where $\mathcal{C}$ is the candidate pool, $G^\star$ the executed group and ν a small constant that keeps the ratio finite when every member of a group reads zero utility. Each live correction seeds one chain that orders the others by decreasing overlap $\|A_i A_j^\top\|_F$ with the seed, and every prefix of two to $g_{\max}$ members enters the pool.

Accumulated displacement. The stream-level ledger is updated to the worst displacement any surviving correction has accumulated, with each new consolidated index inheriting the worst history of the members it replaces,

$$L_{t,i} = L_{t^-,i} + \ell_{G_e,i}(q_e^\star), \qquad L_{t,g} = \max_{i \in G_e} \left(L_{t^-,i} + \ell_{G_e,i}(q_e^\star) \right), \qquad L_t = \max_{i \in S_t} L_{t,i}, \tag{11}$$

where $L_{t,i}$ is the accumulator of index i after event e, g the index the event creates from group G_e, and L_t the worst accumulator over the live set, which the state of Eq. (16) carries forward and on which the displacement bound is enforced. Details in Appendix A.5.

4.4 Overlap-Budgeted Placement

To address Challenge #3, overlap-budgeted placement caps how much of an arrival may land on the width a consolidation freed. As shown in Figure 3 (right), it splits the fit against the committed span, places the arrival under a cap on the free half, and bounds the interference it leaves.

Free-width split. The unconstrained fit is split into the part that reuses committed directions and the part that spends free ones, and the drawn share is read off the second,

$$\tilde{A}_t = \tilde{A}_t P_S + \tilde{A}_t (I - P_S), \qquad \eta(A) = \frac{\|A(I - P_S)\|_F^2}{\|\tilde{A}_t\|_F^2}, \qquad \eta(0) = 0, \tag{12}$$

where $\tilde{A}_t$ is the unconstrained input factor of the arrival, P_S the projector onto the committed span, and $\eta(A)$ the share of the arrival's own target that a placed factor puts outside that span, which is the share of it drawn from the width a consolidation returned. The budget governs this half, since a placed factor raises the committed count by at most the rank of its $(I - P_S)$ block. The split is exact, and the drawn mass is a quantity the allocator reads rather than one it assumes small.

Budgeted placement. The placement is solved as least squares under an absolute cap on the re-captured mass, with the radius scaled by the arrival's own target,

$$A_t = \arg \min_{\|A(I-P_S)\|_F^2 \le \rho_t} \left\| B_t A - \Delta W_t^{\text{target}} \right\|_F^2, \qquad \rho_t = \bar{\eta} \left\| \tilde{A}_t \right\|_F^2, \tag{13}$$

where A_t is the placed factor, B_t the fixed output-side factor with r_t columns, $\Delta W_t^{\text{target}}$ the arrival's target weight change, ρ_t the draw radius and $\bar{\eta} \in [0, 1]$ the free-width budget. The program is convex, and its objective separates along P_S, leaving the span part at the unconstrained fit.

Interference bound. The interference on any retained correction is bounded by a budget-capped term and a committed-span term, both known at placement time,

$$\|B_t A_t x\| \le \|B_t\|_2 \left(\sqrt{\bar{\eta}} \left\| \tilde{A}_t \right\|_F + \left\| A_t^\perp \right\|_2 \right) \|x\|, \tag{14}$$

where $A_t^\perp = \tilde{A}_t P_S$ is the committed-span block and the first term alone is governed by $\bar{\eta}$. The placed factor A_t is what the layer records for the arrival, under a cap on the energy of the outside half rather than on its rank. Details in Appendix A.6.

4.5 End-to-End Integration

Per-arrival loop. As shown in Figure 3 (bottom), the three components run as one loop over the stream of T arrivals, with no surrogate added to the placement. The loop reads the free width from the committed count of the live set, places the arrival directly when that width covers its requested rank, and otherwise reclaims until it does or the layer reports exhaustion,

$$w_{\text{free}}(S) = d - K(S), \qquad \pi : \text{place } A_t \text{ if } w_{\text{free}}(S) \ge r_t, \text{ else reclaim } G^\star, \tag{15}$$

where $w_{\text{free}}(S)$ is the free width of S, d the per-layer budget, r_t the requested rank, A_t the placed input factor and $G^\star$ the executed reclamation group. Details in Appendix A.7.

Carried state. The state carried forward records the live set, its committed basis, the per-correction ledger and the membership maps, with the capacity and displacement constraints of Eq. (1) enforced pointwise after every event,

$$\Sigma_t = \left(S_t, U_{S_t}, \{(\kappa_i, u_i, L_{t,i}, X_i)\}_{i \in S_t}, \{g \mapsto G_e\} \right), \qquad K(S_t) \le d \text{ and } \max_{i \in S_t} L_{t,i} \le \delta, \tag{16}$$

where Σ_t is the carried state, U_{S_t} the committed basis and $\{g \mapsto G_e\}$ the membership maps. Both constraints hold by induction, which establishes admissibility. Details in Appendix A.8.

Table 1: Baselines and RankReclaim on three streams at a budget of 16: quality (qual), edit success (succ), locality (loc), committed count (comm), slack, displacement (disp) and corrections served.

Method	zsre							counterfact							mquake						
	qual ↑	succ ↑	loc ↑	comm	slack ↓	disp ↓	served ↑	qual ↑	succ ↑	loc ↑	comm	slack ↓	disp ↓	served ↑	qual ↑	succ ↑	loc ↑	comm	slack ↓	disp ↓	served ↑
Adapter-merging	0.1042	0.0556	0.3264	11.0	1.0	2.4505	18.0	0.0119	0.0	0.4702	9.0	1.0	2.2388	21.0	0.0625	0.0500	0.3812	9.0	3.0	2.2886	20.0
Model-merging	0.2014	0.1111	0.8472	11.0	1.0	0.8844	18.0	0.0714	0.0476	0.9405	9.0	1.0	0.7009	21.0	0.1125	0.1000	0.8187	9.0	3.0	0.7185	20.0
Null-space	0.0694	0.0	0.7292	16.0	1.0	0.0	7.0	0.1488	0.0476	0.8155	16.0	0.0	0.0	7.0	0.0688	0.0500	0.7250	16.0	0.0	0.0	6.0
Orthogonal-adapter	0.1667	0.1111	0.6736	16.0	0.0	0.0	6.0	0.2440	0.1905	0.7143	16.0	0.0	0.0	7.0	0.1500	0.1000	0.7500	16.0	0.0	0.0	6.0
ROME	0.1250	0.0556	0.2708	15.0	0.0	0.0	15.0	0.1369	0.0952	0.3929	16.0	0.0	0.0	16.0	0.0938	0.0	0.2938	15.0	2.0	0.0	17.0
MEMIT	0.0486	0.0	0.4722	16.0	0.0	0.0	6.0	0.1429	0.0476	0.4464	16.0	0.0	0.0	7.0	0.0625	0.0500	0.5062	16.0	0.0	0.0	6.0
Norm-bounded	0.1042	0.0556	0.9722	16.0	3.0	0.0	7.0	0.1488	0.0952	0.9702	16.0	4.0	0.0	8.0	0.0625	0.0500	0.6750	15.0	12.0	0.0	13.0
RankReclaim (ours)	0.0833	0.0	0.3611	16.0	0.0	1.0031	13.0	0.1905	0.0952	0.6190	15.0	2.0	0.9929	15.0	0.0750	0.0	0.3625	16.0	5.0	0.9811	15.0

5 EXPERIMENTS

5.1 IMPLEMENTATION DETAILS

Datasets. Three edit streams run at one layer of 96 inputs and 64 outputs, seeded from the zsre (Levy et al., 2017), counterfact (Meng et al., 2022) and mquake (Zhong et al., 2023) sets; each reaches three times the budget, with one row in four off the topic bank. Details in Appendix A.9.

Baselines. Seven baselines share the streams, probes and per-layer budget: adapter merging (Ilharco et al., 2023), interference-resolved merging (Yadav et al., 2023), null-space (Fang et al., 2025) and orthogonal-adapter (Wang et al., 2023) placement, rank-one model editing (ROME) (Meng et al., 2022), mass editing (MEMIT) (Meng et al., 2023) and norm-bounded fine-tuning (Zhu et al., 2020). Three further arms isolate the components of RankReclaim: the requested-rank account, the measured count without reclamation and a consolidation at a fixed rank chosen without the displacement tolerance. Details in Appendix A.10.

Evaluation Metrics. The ledger gives slack ↓, displacement ↓ and the committed count; end-stream quality ↑ is read on each correction's probes, and corrections served ↑ counts admissions. Committed count, attribution error and free-width draw carry no arrow. Details in Appendix A.11.

Hyperparameters. The group-size cap is $g_{\max} = 6$, the free-width budget $\bar{\eta} = 0.35$, the spectral tolerance $\theta = 0.05$ and the reference budget $d = 16$, over width bands 12, 16, 24 and 32. Each correction carries eight probes, and the selection ratio uses $\nu = 10^{-3}$. Details in Appendix A.12.

Protocol. Two levels of aggregation appear and the tables do not mix them: within a stream the readings are means over arrivals, and a pooled cell is the mean of the per-stream readings. Every reading is deterministic, with no average over seeds or repeated runs. Details in Appendix A.13.

5.2 MAIN RESULTS

Comparison with Editing Baselines. Among the eight arms of Table 1, RankReclaim is the only reclaiming arm that holds the committed count at the budget, 16, 15 and 16 of 16, where both merging arms stop at 9 to 11, and the only arm that reads the displacement of a consolidation before the move executes. This is because the ledger charges an arrival only for the directions the layer does not yet hold, and priced reclamation returns width only at a rank whose displacement it has already read. Adapter merging folds the live set at a fixed rank and pays displacements of 2.4505, 2.2388 and 2.2886, against 1.0031, 0.9929 and 0.9811 for RankReclaim. At this budget RankReclaim serves 13, 15 and 15 corrections, and the committed count it charges therefore sets how many corrections the layer can serve. Details in Appendix A.14.

Scaling across Width Bands. RankReclaim fills every band it is given: Figure 4 shows a ceiling ratio of at least 0.9583 at all four widths, against 0.4167 to 0.6250 for adapter merging and interference-resolved merging, while the null-space arm falls to 0.9792 and 0.9375 at the two wide bands. At the widest band RankReclaim commits all 32 directions, against 17.5 for both merging arms. This is because folding the live set into one update at a fixed rank leaves a large part of the band unused, whereas the ledger admits against the directions actually free and keeps the band occupied as it widens. Details in Appendix A.14.

5.3 ABLATION STUDY AND ANALYSIS

Contribution of Each Component. Counting the committed directions is what first raises the service. Table 2 shows that replacing the requested-rank account with the measured count, at the

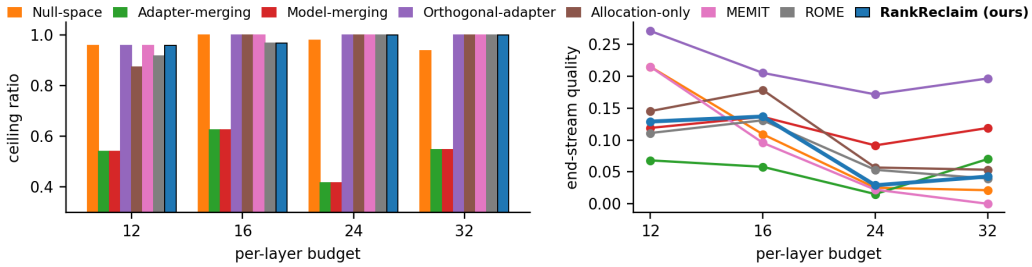

Figure 4: **Left:** ceiling ratio, the committed count over the per-layer budget, of every arm at four width bands. **Right:** end-stream quality ↑ at the same bands; both panels share one legend.

Table 2: Ablation of the components: served, returned, displacement and quality per stream.

Method	zsre				counterfact				mquake			
	served ↑	returned	disp ↓	qual ↑	served ↑	returned	disp ↓	qual ↑	served ↑	returned	disp ↓	qual ↑
Requested-rank account	6	0	0.0	0.0903	7	0	0.0	0.1786	6	0	0.0	0.0750
Measured count, no reclamation	9	0	0.0	0.1667	8	0	0.0	0.1905	13	0	0.0	0.0625
Fixed-rank consolidation, unpriced	16	15	3.8040	0.0347	16	11	2.7923	0.0595	17	6	3.2196	0.0500
RankReclaim (ours)	13	8	1.0031	0.0833	15	6	0.9929	0.1905	15	4	0.9811	0.0750

same arrivals and budget, raises the corrections served from 6, 7 and 6 to 9, 8 and 13 at a displacement of 0.0, since it admits without reclaiming, and that RankReclaim, which adds reclamation and placement, raises them to 13, 15 and 15. The two steps contribute differently by stream: counting accounts for seven of the nine added corrections on mquake, and reclamation with placement for seven of the eight on counterfact. Against the consolidation at a fixed rank chosen without the tolerance, RankReclaim returns 8, 6 and 4 directions against 15, 11 and 6 and reads a quality of 0.0833, 0.1905 and 0.0750 against 0.0347, 0.0595 and 0.0500, at 26% to 36% of the fixed-rank displacement. This is because counting admits the arrivals whose directions the layer already holds, while priced reclamation executes only ranks inside the displacement tolerance. Details in Appendix A.10.

Effectiveness of the Committed-Capacity Ledger. The ledger attributes capacity where a requested account cannot. Table 3 shows RankReclaim holding the committed count between 14 and 16 with a slack between 0 and 2 and an attribution error between 0.0000 and 0.5625, whereas the requested-rank account freezes at 12.5 with a slack of 3.5 and an attribution error of 0.9405, and the measured count without reclamation reads 0.8819 throughout. After the first position its attribution error stays at or below 0.1818. At the second position slack and attribution error both read 0, the committed count there matching the requested total. Across six positions the committed count stays within two directions of the budget, while the requested-rank account leaves 3.5 directions unused. This is because a requested rank is charged in full even when it repeats directions already held, while Eq. (4) charges each correction the directions its removal returns. Details in Appendix A.15.

Effectiveness of Priced Reclamation. Priced reclamation reads the price of a consolidation before paying it. Eq. (8) evaluates the residual on the survivors' probes before the group is overwritten, and a rank whose displacement would exceed the tolerance is never executed, whereas a fixed rank commits first and pays afterwards. Through Eq. (7) the rank at which a consolidation stops sets its cost, and choosing that rank after reading the cost is what the pricing adds. The price holds across five group-size caps: RankReclaim returns 8 directions at a displacement of 1.0031 at every cap but the smallest, where it reads 1.0139. The price is safe to read in advance: in Figure 5 (middle) the bound of Eq. (7) is never violated, and the realized displacement reaches 0.6133 to 0.8762 of it. In Figure 5 (right) its end-stream quality stays above the fixed-rank consolidation at all eight running conditions, 0.0486 to 0.1042 against 0.0347. Details in Appendix A.15.

Effectiveness of Overlap-Budgeted Placement. The cap holds the free-width draw at or under the budget at every working point, rising from 0.0410 at a budget of 0.05 to 0.5990 at 0.95. This is because Eq. (13) bounds the free block against the arrival's own unconstrained fit, and Eq. (12) reports exactly that share. Once the budget exceeds the share the unconstrained fit draws on its own, near 0.60, the constraint stops binding, which is why the draw settles at 0.5944 and 0.5990 at the two loosest budgets. The draw rises monotonically with the budget at all six points, and the budget therefore acts as one predictable setting. At the reference budget of 0.35 the cap binds at a draw of 0.2641 while the four uncapped arms draw between 0.6679 and 0.7832, as Figure 5 (left) shows, and RankReclaim reads the highest quality of the five, 0.1369. Details in Appendix A.15.

Table 3: Slack ↓, attribution error and committed count of every arm at six stream positions.

Method	slack ↓						attribution error						committed					
	1	2	3	4	5	6	1	2	3	4	5	6	1	2	3	4	5	6
Requested-rank	3.5	3.5	3.5	3.5	3.5	3.5	0.9405	0.9405	0.9405	0.9405	0.9405	0.9405	12.5	12.5	12.5	12.5	12.5	12.5
Allocation-only	4.5	4.5	4.5	4.5	4.5	4.5	0.8819	0.8819	0.8819	0.8819	0.8819	0.8819	16.0	16.0	16.0	16.0	16.0	16.0
Adapter-merging	2.0	0.0	0.5	1.0	1.5	2.5	0.4375	0.0000	0.3333	0.3750	0.5000	0.7500	9.5	3.5	4.0	6.5	8.5	9.5
Model-merging	2.0	0.0	0.5	1.0	1.5	2.5	0.4375	0.0000	0.3333	0.3750	0.5000	0.7500	9.5	3.5	4.0	6.5	8.5	9.5
Null-space	0.5	0.5	0.5	0.5	0.5	0.5	0.0714	0.0714	0.0714	0.0714	0.0714	0.0714	16.0	16.0	16.0	16.0	16.0	16.0
RankReclaim (ours)	2.0	0.0	0.5	0.5	0.5	1.0	0.5625	0.0000	0.1250	0.1111	0.1000	0.1818	16.0	14.0	14.0	15.0	15.5	15.5

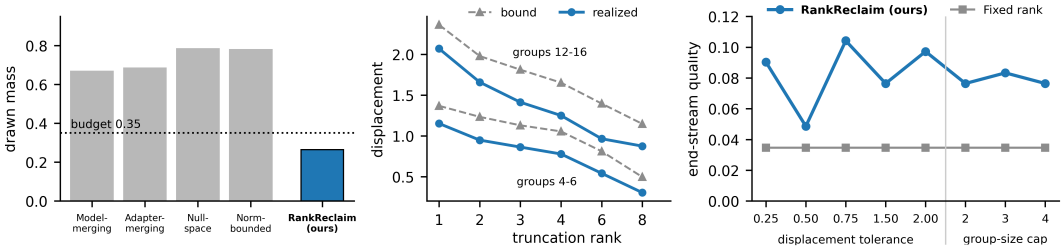

Figure 5: **Left:** drawn mass at the reference budget of 0.35. **Middle:** the Eq. (7) bound and the realized displacement by truncation rank. **Right:** end-stream quality ↑ over eight running conditions.

Placement and Reclamation Together. The free-width budget also moves how much reclamation the loop runs, and the price holds across that range. At budgets of 0.95, 0.35 and 0.05, RankReclaim returns 4, 8 and 19 directions, while the accumulated displacement stays between 0.9263 and 1.0031. This is because Eq. (8) prices every event against the same tolerance, and a tighter placement budget therefore buys more returned width without a larger displacement. Details in Appendix A.18.

6 Related Work

Lifelong Model Editing. A correction arrives with a requested rank, written rank-one (Meng et al., 2022), in mass (Meng et al., 2023) or by a learned editor (Mitchell et al., 2022), and allocation-only editors take the requested sum as the spend, exact only when the live row spaces form a direct sum. Pruning showed that a nominal count and the carried count differ (Frankle & Carbin, 2019) and that per-unit budgets fail when units interact (Dutta et al., 2024; Li et al., 2025). RankReclaim makes that observation an operator: the committed count of the live set, recomputed as the set changes.

Saliency as Removal Cost. Pruning reads a parameter's importance as what its removal costs rather than how large it is: second-order saliency prices a deletion by the error it leaves (Hassibi & Stork, 1992), one-shot pruning selects removals against that reconstruction error (Frantar & Alistarh, 2023), and gradient information decides which structures may go (Das et al., 2023). Each scores a fixed network once. RankReclaim applies the same reading to the marginal committed count of a live correction and recomputes it as the live set changes, thereby scoring returnable capacity.

Rank Truncation and Compression. Compression treats rank as the budget: row and column deletion spends it (Ashkboos et al., 2024), joint sparsification makes the same trade explicit (Guo et al., 2024; Mozaffari et al., 2024), and structured-then-unstructured schedules show the order of taking capacity back matters (Lee et al., 2024). All truncate once and measure the error afterwards. RankReclaim truncates the same way but chooses the rank by evaluating the residual on stored probes before the move executes, against a tolerance rather than a reported loss.

7 Conclusion

Editors budget a layer by the ranks its corrections request, a sum overstating the capacity spent. RankReclaim counts the directions a layer has actually committed and prices every consolidation before it executes, through a committed-capacity ledger, priced reclamation and overlap-budgeted placement. On three streams the measured count alone raises corrections served from 6, 7 and 6 to 9, 8 and 13, and RankReclaim to 13, 15 and 15, the only reclaiming arm among eight that holds the committed count at the budget. A layer that counts and prices its width keeps serving corrections.

REFERENCES

Saleh Ashkboos, Maximilian L. Croci, Marcelo Gennari do Nascimento, Torsten Hoefler, and James Hensman. Slicegpt: Compress large language models by deleting rows and columns. arXiv preprint arXiv:2401.15024, 2024.

Rocktim Jyoti Das, Mingjie Sun, Liqun Ma, and Zhiqiang Shen. Beyond size: How gradients shape pruning decisions in large language models. arXiv preprint arXiv:2311.04902, 2023.

Oshin Dutta, Ritvik Gupta, and Sumeet Agarwal. Efficient llm pruning with global token-dependency awareness and hardware-adapted inference. In Workshop on Efficient Systems for Foundation Models II@ ICML2024, 2024.

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, 2025.

Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In International Conference on Learning Representations, 2019.

Elias Frantar and Dan Alistarh. SparseGPT: Massive language models can be accurately pruned in one-shot. In Proceedings of the 40th International Conference on Machine Learning, volume 202 of Proceedings of Machine Learning Research, pages 10323–10337. PMLR, 2023.

Jinyang Guo, Jianyu Wu, Zining Wang, Jiaheng Liu, Ge Yang, Yifu Ding, Ruihao Gong, Haotong Qin, and Xianglong Liu. Compressing large language models by joint sparsification and quantization. In Proceedings of the 41st International Conference on Machine Learning, volume 235 of Proceedings of Machine Learning Research, pages 16945–16957. PMLR, 2024.

Tom Hartvigsen, Swami Sankaranarayanan, Hamid Palangi, Yoon Kim, and Marzyeh Ghassemi. Aging with GRACE: Lifelong model editing with discrete key-value adaptors. In Advances in Neural Information Processing Systems 36, 2023. doi: 10.52202/075280-2079.

Babak Hassibi and David Stork. Second order derivatives for network pruning: Optimal brain surgeon. Advances in neural information processing systems, 5, 1992.

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 ICLR, 2022.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic. In International Conference on Learning Representations, 2023.

Jaeseong Lee, Aurick Qiao, Daniel F Campos, Zhewei Yao, Yuxiong He, et al. Stun: Structured-then-unstructured pruning for scalable moe pruning. arXiv preprint arXiv:2409.06211, 2024.

Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. Zero-shot relation extraction via reading comprehension. In Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017), pages 333–342. Association for Computational Linguistics, 2017. doi: 10.18653/v1/K17-1034.

Zhoubo Li, Ningyu Zhang, Yunzhi Yao, Mengru Wang, Xi Chen, and Huajun Chen. Unveiling the pitfalls of knowledge editing for large language models. In The Twelfth International Conference on Learning Representations, 2024.

Guanchen Li, Yixing Xu, Zeping Li, Ji Liu, Xuanwu Yin, Dong Li, and Emad Barsoum. Týr-the-pruner: Unlocking accurate 50% structural pruning for llms via global sparsity distribution optimization. arXiv preprint arXiv:2503.09657, 2025.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in GPT. In Advances in Neural Information Processing Systems 35, 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, 2023.

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

Mohammad Mozaffari, Amir Yazdanbakhsh, and Maryam Mehri Dehnavi. Slim: One-shot quantization and sparsity with low-rank approximation for llm weight compression. arXiv preprint arXiv:2410.09615, 2024.

Saurav Muralidharan, Sharath Turuvekere Sreenivas, Raviraj Joshi, Marcin Chochowski, Mostofa Patwary, Mohammad Shoeybi, Bryan Catanzaro, Jan Kautz, and Pavlo Molchanov. Compact language models via pruning and knowledge distillation. Advances in Neural Information Processing Systems, 37:41076–41102, 2024.

Xiao Wang, Tianze Chen, Qiming Ge, Han Xia, Rong Bao, Rui Zheng, Qi Zhang, Tao Gui, and Xuanjing Huang. Orthogonal subspace learning for language model continual learning. In Findings of the Association for Computational Linguistics: EMNLP, 2023. doi: 10.18653/v1/2023.findings-emnlp.715.

Peng Wang, Ningyu Zhang, Bozhong Tian, Zekun Xi, Yunzhi Yao, Ziwen Xu, Mengru Wang, Shengyu Mao, Xiaohan Wang, Siyuan Cheng, Kangwei Liu, Yuansheng Ni, Guozhou Zheng, and Huajun Chen. EasyEdit: An easy-to-use knowledge editing framework for large language models. In Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations), pages 82–93. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-demos.9.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin Raffel, and Mohit Bansal. TIES-Merging: Resolving interference when merging models. In Advances in Neural Information Processing Systems 36, 2023. doi: 10.52202/075280-0310.

Yunzhi Yao, Peng Wang, Bozhong Tian, Siyuan Cheng, Zhoubo Li, Shumin Deng, Huajun Chen, and Ningyu Zhang. Editing large language models: Problems, methods, and opportunities. In Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, pages 10222–10240. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.emnlp-main.632.

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, pages 19449–19457, 2024. doi: 10.1609/aaai.v38i17.29916.

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

Zexuan Zhong, Zhengxuan Wu, Christopher D. Manning, Christopher Potts, and Danqi Chen. MQuAKE: Assessing knowledge editing in language models via multi-hop questions. In The 2023 Conference on Empirical Methods in Natural Language Processing, 2023. doi: 10.18653/v1/2023.emnlp-main.971.

Chen Zhu, Ankit Singh Rawat, Manzil Zaheer, Srinadh Bhojanapalli, Daliang Li, Felix Yu, and Sanjiv Kumar. Modifying memories in transformer models. arXiv preprint arXiv:2012.00363, 2020.

AI USE STATEMENT

We used generative AI tools to translate the text (Claude Opus 5.0), to check grammar and spelling and polish the wording (GPT-5.6), to guide the drawing of the figures (Gemini), and to check the content (Grok). We did not use generative AI tools to modify code. We have reviewed all AI-assisted work: we checked every reference for hallucinated citations. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

The details needed to reproduce the results of this paper, including the stream layout, the arms, the evaluation metrics, the hyperparameter specifications and the evaluation protocol, are disclosed in this appendix.

APPENDIX

Table 4 lists the contents of each part of the appendix.

Table 4: What each part of the appendix holds.

Section	Contents
Appendix A.0	Limitations
Appendix A.1	Related work
Appendix A.2	The motivation readings
Appendix A.3	The component map
Appendix A.4	The ledger in detail
Appendix A.5	The pricing in detail
Appendix A.6	The placement in detail
Appendix A.7	The per-arrival loop and its cost
Appendix A.8	Admissibility of the loop
Appendix A.9	Stream layout
Appendix A.10	The arms
Appendix A.11	Evaluation metrics
Appendix A.12	Hyperparameter specifications
Appendix A.13	Evaluation protocol
Appendix A.14	The main results tables
Appendix A.15	The ablation tables
Appendix A.16	The three effect directions
Appendix A.17	Settings in which another arm reads better
Appendix A.18	The boundary regimes

APPENDIX A.0: LIMITATIONS

Objective Without a Policy. A reader judging an admission policy benefits from knowing how end-stream quality moves with the number of corrections a layer holds when the corrections are written in unchanged, with no policy in the loop and no budget enforced. This study lacks that policy-free reading on its three streams.

Per-Arrival Replay. Edit success on the final layer shows how many served corrections are still reproduced, but a reader diagnosing a lost correction needs to see where the loss begins: with the correction alone in an empty layer, beside only the live object it belongs to, or with the whole live set present. This study lacks that three-way replay of the served corrections.

Live Parameters Across Widths. For deployment, the number of parameters an editing method leaves live matters alongside the corrections it serves, and it changes with the width of the layer. This study lacks a side-by-side count of the live parameters of the editing methods across layer widths.

APPENDIX A.1: RELATED WORK

Saliency read as what removal returns. The idea that a parameter's importance is what deleting it costs, rather than how large it is, goes back to second-order pruning saliency (Hassibi & Stork, 1992), and the same reading drives modern one-shot pruning, where the criterion is the reconstruction error a removal leaves behind (Frantar & Alistarh, 2023) and where gradient information is used to decide

which structures may go (Das et al., 2023). This paper takes that reading and applies it to a quantity those methods do not have: not which weights to delete but how many directions a live correction is holding, which is the marginal committed count of Eq. (4). The difference is that a pruning saliency scores a fixed network once, whereas the ledger here is recomputed as the live set changes, and what it scores is capacity that can be returned rather than parameters that can be dropped.

Rank as the resource, and truncation as the way to free it. Compressing a trained model by deleting rows and columns treats rank itself as the budget (Ashkboos et al., 2024), and joint sparsification with low-rank approximation makes the same trade explicit (Guo et al., 2024; Mozaffari et al., 2024); structured-then-unstructured schedules show that the order in which capacity is taken back matters (Lee et al., 2024), and pruning combined with distillation shows what a rebuilt model has to be repaid with (Muralidharan et al., 2024). All of these truncate once, against a fixed objective, with the error measured after the fact. The consolidation of Eq. (6) is the same truncation, but Eq. (8) chooses its rank by evaluating the residual on stored probes before the move is executed; the rank is thus selected against a tolerance rather than reported against a loss.

Budgets that are requested against budgets that are counted. The lottery-ticket line established that a nominal parameter count and the count that carries the function are different numbers (Frankle & Carbin, 2019), and global-dependency pruning makes the point that a per-unit budget cannot be enforced unit by unit when the units interact (Dutta et al., 2024; Li et al., 2025). The committed count of Eq. (2) is that observation made into an operator: the requested total upper-bounds it and meets it only when the row spaces of the live corrections form a direct sum, which is the case the allocation-only family silently assumes.

What the compression literature does not carry. The compression literature reviewed above does not cover the editing families this paper places beside its own: rank-one (Meng et al., 2022) and mass editing (Meng et al., 2023), learned editor networks (Mitchell et al., 2022), null-space constrained editing and side memories. That literature truncates a trained network once against a fixed objective and carries no notion of a correction arriving after the fact with a requested rank of its own.

APPENDIX A.2: THE MOTIVATION READINGS

This section details the three motivation readings, one panel at a time. Each panel is reproduced here on its own such that its axes and its full range of readings can be seen at the size the main-text composite cannot give them.

A.2.1 Uncounted capacity. Figure 6 plots the requested rank total and the committed count against the number of arrivals, with the per-layer width of 16 marked by the dotted line. The requested total is the sum of the ranks the arrivals request; the committed count is the numerical rank of the stacked input factors at the spectral tolerance $\theta = 0.05$. The two read the same up to the fourth arrival and part from the fifth onward, reaching 48 against 30 at the eighteenth. The stream carries a topic bank of 16 directions, a spread of 0.025 around each drawn topic, and a novel share of 0.25, where the novel share is the share of rows that introduce a direction the bank does not carry. This value is the largest on the grid $0, 0.25, 0.5, 0.75, 1$ at which the committed count stays at or below two thirds of the requested total while exceeding the per-layer budget; hence the overstatement is present and is the stream's own overlap rather than the budget ceiling. No arm is run on this stream.

A.2.2 Unpriced consolidation. Figure 7 plots the worst surviving displacement on the left axis and the capacity a truncation returns on the right axis, both against the rank the group is truncated to, at the five group sizes 4, 6, 8, 10 and 12. The displacement is the largest response change the residual matrix produces on any live correction's own stored probes; the capacity returned is the directions the group held less the rank the consolidation keeps. At group size 8, rank 1 displaces the worst survivor by 1.285804 and rank 8 by 0.671775, a factor of 1.91. In contrast, the capacity returned reads 19 directions against 12 and therefore orders the two truncations the other way. The two quantities are read at different times in an unpriced rule, which is what makes the ordering invert.

A.2.3 Unbudgeted placement. Figure 8 plots two shares against the number of live corrections: the share of the next arrival's unconstrained fit that lands outside the committed span, which is the share

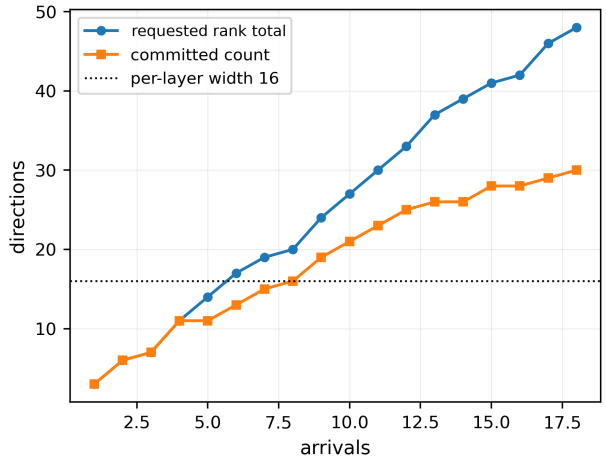

Figure 6: Requested rank total and committed count against arrivals, with the per-layer width marked.

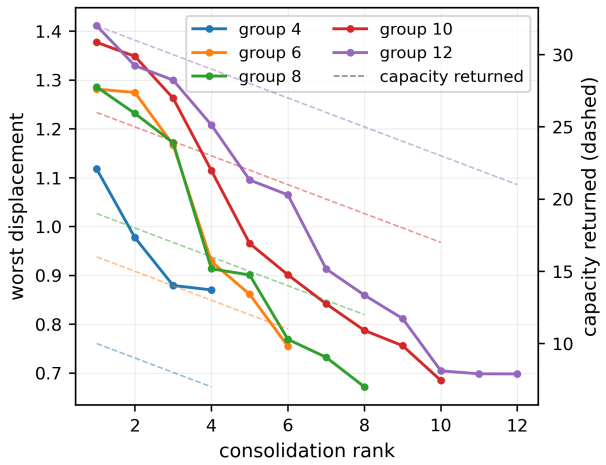

Figure 7: Worst surviving displacement (left axis) and capacity returned (right axis, dashed) against truncation rank, at five group sizes.

drawn from the width a consolidation returned, and the share a direction drawn without reference to that span puts there. The first runs from 1.5293 percent at 15 live corrections to 94.7507 percent at 2, and it follows neither the live count nor the free width. The second is the reference the first is read against, and it moves in a narrow band across the same range. What separates them is that the placement objective contains no term referring to the free width; as a result, the share is decided by which topics the next arrival happens to need rather than by what the layer has just released.

APPENDIX A.3: THE COMPONENT MAP

Each component answers one challenge and instantiates one condition of the policy class, and the equations that define it are named beside it.

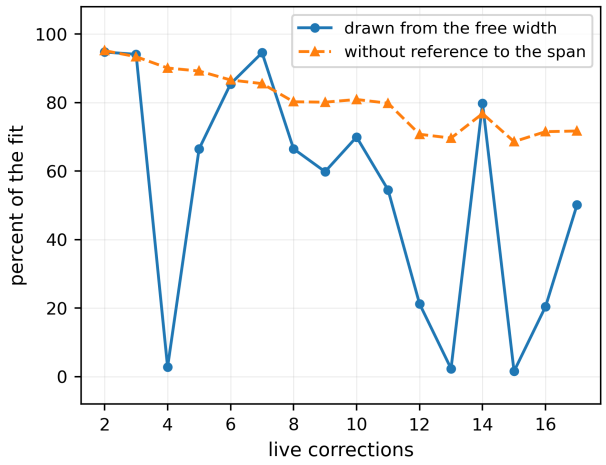

Figure 8: Share of the next arrival's fit falling outside the committed span, and the share a direction drawn without reference to that span puts there, against the number of live corrections.

Table 5: Each component, the challenge it answers, the condition of the policy class it instantiates, and the equations that define it.

Component (defining equations)	Challenge answered	Formulation constraint
Committed-capacity ledger (Eqs. (2) to (5))	uncounted capacity	within budget $K(S) \le d$
Priced reclamation (Eqs. (6) to (11))	unpriced consolidation	displacement $\max_i L_i \le \delta$
Overlap-budgeted placement (Eqs. (12) to (14))	unbudgeted placement	free-width draw $\eta(A) \le \bar{\eta}$
End-to-End Integration (Eq. (16))	the three as one loop	policy $\pi \in \Pi$

APPENDIX A.4: THE LEDGER IN DETAIL

The spectral tolerance of Eq. (2) is absolute; the count is therefore read on factors in a fixed normalization. Every arrival is carried with its input factor's rows at unit norm and the output factor absorbing the scale. Without that convention, the pair $(A, B) \mapsto (\alpha A, \alpha^{-1} B)$ would leave the update unchanged and move the count.

The requested total and the committed count coincide only when the row spaces form a direct sum, which is the case the allocation-only family assumes, and separate everywhere else. Pairwise disjointness is not enough: $\mathrm{span}(e_1)$, $\mathrm{span}(e_2)$ and $\mathrm{span}(e_1 + e_2)$ meet only at the origin in pairs and still span two directions rather than three.

The utility indicator of Eq. (5) reads true on a probe when the error between the layer's response and the response the correction specified is at most half the norm of that specified response. Since every correction the stream specifies stays in the objective for the whole stream, a consolidation that destroys one member's behaviour is charged as that member's utility falling rather than as its removal from the average.

APPENDIX A.5: THE PRICING IN DETAIL

The inheritance clause of Eq. (11) is what stops a chain of consolidations from re-creating a correction as a fresh index and forgetting the degradation it had already absorbed; as a result, repeated small merges are charged rather than free. The rank test of Eq. (8) is well posed at its upper end: at $q = \mathrm{rank}(\Delta W_G)$ the first discarded singular value is zero and the test reduces to the invariant already in force. The score of Eq. (9) errs only toward refusal: it never admits a candidate an exact reading would refuse, since the $q^\star(G)$ directions the merge keeps are charged in full though some of them may already lie in the span of the corrections outside G. A requested total above the budget is therefore not on its own a witness that some group scores positive. Because the rank search runs

over $q \geq 1$, a group whose members have all been overwritten is still replaced by a rank-one merge rather than dropped.

APPENDIX A.6: THE PLACEMENT IN DETAIL

Every placed arrival has its drawn mass $\eta(A_t)$ recorded. What Eq. (13) caps is the energy of the outside half rather than its rank: a cap on energy limits the directions that clear the spectral tolerance θ only through that tolerance. Hence the component budgets the mass drawn from the free width, and the committed count it holds back is not read directly. The denominator of Eq. (12) is the same quantity the radius of Eq. (13) is scaled by; thus the share reported and the share constrained are one number rather than two.

The program of Eq. (13) separates exactly along P_S: the span part is left at the unconstrained fit and the free part is the point of the Frobenius ball the constraint states nearest to $\tilde{A}_t(I - P_S)$ in the metric $B_t^\top B_t$, which is $U(\sigma_i/(\sigma_i + \lambda))\, U^\top \tilde{A}_t(I - P_S)$ for the eigendecomposition $U\Sigma U^\top$ of $B_t^\top B_t$ and the single multiplier $\lambda \geq 0$ that meets the radius. Scaling the radius by the arrival's own target keeps the budget live even when the free block is empty, where a fraction of that block would collapse the radius to zero at every budget.

APPENDIX A.7: THE PER-ARRIVAL LOOP AND ITS COST

Algorithm 1 lists the admission procedure for one arrival. The procedure runs in three stages, one per component. Stage 1 refreshes the committed count, the committed basis and the per-correction attribution. Stage 2 runs only while the free width falls short of the requested rank: it forms the candidate pool, reads the utilities the selection rule consumes, prices every candidate at the smallest rank the residual tolerance admits, and charges the selected merge to every live accumulator before replacing the group by its truncation; when no candidate returns capacity, or an executed merge leaves the committed count where it was, the layer reports exhaustion and the arrival is refused. Stage 3 places the arrival inside the draw radius its unconstrained fit sets and records the realized draw. The refreshed ledger is carried to the next arrival, which ensures that each later price is charged against the budget earlier events have already spent.

The candidate pool is formed on the overlap weight $\|A_i A_j^\top\|_F$ between every pair of live corrections. Each live correction seeds one chain that orders the others by decreasing weight to the seed, and every prefix of two to $g_{\max}$ members enters the pool once; equal weights are ordered by index, and the argmax of Eq. (10) keeps the first group in the order of its sorted member indices. A merged index g carries the stacked probes of its members as X_g, and its utility u_g is the sum of Eq. (5) over the arrivals it stands for, each read on that arrival's own probes.

Besides the placement program, the only added per-arrival work is the reclamation accounting. It consists of a ledger refresh and, for every candidate in the pool of each reclamation event, one factorization of the group's combined update, one committed count of $S \setminus G$ and a displacement reading at each tested rank; the utility of each arrival the live set stands for is read once per event. The pool holds at most $|S|(g_{\max} - 1)$ candidate groups, and each event either lowers the committed count or ends the reclamation; an arrival therefore runs at most $r_t + 1$ events. Every term is bounded by the committed count, the live-set size and the group-size cap, and none of them grows with the stream length except through the live-set size and the probes the live objects carry.

Table 6: Per-arrival terms relative to an allocation-only pass.

Term	Scaling	Status
Ledger SVD of stacked factors	set by the stacked rank and d	added
Truncated group factorization	set by group size, not stream length	added
Probe evaluations	proportional to the live set and its stored probes	added
Placement least squares over A_t	one convex program per arrival	added

Algorithm 1 RankReclaim admission for one arrival

Require: arrival $(\Delta W_t^{\mathrm{target}}, B_t, r_t, X_t)$ with B_t of r_t columns; carried state Σ_{t-1} with $K(S) \le d$ and $\max_{i\in S} L_i \le \delta$; budget d; tolerances θ and δ; free-width budget $\bar{\eta}$; group cap $g_{\max}$; stabilizer ν

Ensure: carried state Σ_t with $K(S_t) \le d$, $\max_{i\in S_t} L_{t,i} \le \delta$ and $\eta(A_t) \le \bar{\eta}$, or the current state with a reported exhaustion and the arrival refused

Stage 1: Committed-Capacity Ledger

refresh $K(S)$, the committed basis U_S and κ_i for every $i \in S$ ▷ Eq. (2), (4)

Stage 2: Priced Reclamation

while $d - K(S) < r_t$ **do**

  form $\mathcal{C}$: one chain per seed by decreasing overlap weight, every prefix up to $g_{\max}$

  read u_i for every $i \in S$ ▷ Eq. (5)

  for all $G \in \mathcal{C}$ **do**

    $q^\star(G) \leftarrow \min\{q : \max_{i\in S}(L_i + \ell_{G,i}(q)) \le \delta\}$ ▷ Eq. (6)–(8)

    $\mathrm{net}(G) \leftarrow K(S) - K(S \setminus G) - q^\star(G)$ ▷ Eq. (9)

  end for

  if $\mathrm{net}(G) \le 0$ for every $G \in \mathcal{C}$ **then**

    return the current state with a reported exhaustion

  end if

  $G^\star \leftarrow \arg\max_{G\in\mathcal{C}} \mathrm{net}(G)/(\sum_{i\in G} u_i + \nu)$; $q^\star \leftarrow q^\star(G^\star)$ ▷ Eq. (10)

  $L_i \leftarrow L_i + \ell_{G^\star,i}(q^\star)$ for every $i \in S$ ▷ Eq. (11)

  merge $G^\star$ into a fresh index g at rank $q^\star$ with $L_g \leftarrow \max_{i\in G^\star} L_i$ ▷ Eq. (6), (11)

  $S \leftarrow (S \setminus G^\star) \cup \{g\}$; refresh the ledger

  if $K(S)$ did not fall **then**

    return the current state with a reported exhaustion

  end if

end while

Stage 3: Overlap-Budgeted Placement

$\rho_t \leftarrow \bar{\eta}\|\tilde{A}_t\|_F^2$ from the unconstrained fit $\tilde{A}_t$ ▷ Eq. (12), (13)

$A_t \leftarrow \arg\min \|B_t A - \Delta W_t^{\mathrm{target}}\|_F^2$ subject to $\|A(I - P_S)\|_F^2 \le \rho_t$ ▷ Eq. (13)

record $\eta(A_t)$ ▷ Eq. (12)

insert t with factors (B_t, A_t), probes X_t and $L_{t,t} \leftarrow 0$; refresh the ledger; **return** Σ_t ▷ Eq. (16)

APPENDIX A.8: ADMISSIBILITY OF THE LOOP

The two constraints carried in Eq. (16) hold by induction over events from the empty base case. The only step that raises the committed count is a placement of r_t rows, which raises the count by at most r_t and is attempted only after the free width covers the requested rank. The only step that raises any accumulator is a reclamation event that Eq. (8) admits only inside the residual tolerance. Hence the loop is a policy π inside the class the objective delimits. The accounting the loop carries at the end of the stream is therefore the accounting the deployed model carries.

RankReclaim turns lifelong low-rank editing into a loop whose committed capacity is counted rather than requested, whose every reclamation is priced before it is paid, and whose every placement holds its free-width draw at a budget the policy sets, over a frozen backbone.

APPENDIX A.9: STREAM LAYOUT

Stream construction. A stream is the arrivals, their targets and their probes at one layer of 96 input and 64 output units. A bank of topic directions is fixed once. Every arrival draws its requested rank worth of rows; each row is either a topic taken from the bank or a direction the bank does not carry. Reuse across arrivals is what makes the requested total exceed the capacity jointly occupied. A row drawn from a topic is placed at a spread of 0.025 around it. This spread is half the spectral tolerance $\theta = 0.05$ that decides what counts as a committed direction, such that two arrivals on one topic read as one committed direction. The share of rows that introduce a direction outside the bank is 0.25. Probes key on the arrival's own row space with a stated share lying outside it, and locality

probes are drawn outside the bank with a small reach into it; leakage is thus a quantity that can be read rather than one the construction rules out. The stream length is set by the far ceiling multiple of three times the per-layer budget.

How the two reuse quantities were fixed. The spread and the novel share govern the gap between the requested total and the committed count, which is what Challenge 1 is about, and no published work reports a row-space overlap between sequentially arriving low-rank corrections. They are therefore fixed by a rule stated before any reading was taken and applied to the motivation stream alone. The spread is half the spectral tolerance for the reason stated above. The novel share is the largest value on the grid 0, 0.25, 0.5, 0.75, 1 at which the motivation stream's committed count is at or below two thirds of its requested total and above the per-layer budget: the first condition asks that the overstatement be present and not marginal, the second that the reading be the stream's own overlap rather than the budget ceiling. On that grid the four other points fail one condition or the other, and 0.25 is returned with a committed count of 30 against a requested total of 48. No arm is computed while the rule is applied.

Topic-bank size. The bank holds one direction per unit of layer width on 13 of the 14 streams the readings are taken on; on the remaining and widest stream it holds 88 against a layer width of 96. The size is an input to the construction and is recorded rather than tuned.

The displacement tolerance. The displacement tolerance is expressed in the units of the thing it tolerates: the response a correction produces on its own probes. A single value serves the whole stream: the median over the arrivals of each arrival's own largest such response, computed from the full stream before the loop over arrivals begins. It is therefore a stream-level constant, not a per-correction tolerance δ_i, and reading it requires the stream in advance.

Determinism. The construction is deterministic by setting: every random draw is seeded by an integer derived arithmetically from the stream identity, no hash-ordered container is read for any numeric result, and the experiments carry no sampling over seeds. Two runs of the released code on the same stream therefore produce the same file. The released code runs from a single entry script under Python 3 with NumPy and Matplotlib, and writes every table and data figure the body cites; the schematic in the method section is drawn by hand and carries no reading. Every table and figure labels its rows by the stream identity the construction was seeded from.

APPENDIX A.10: THE ARMS

Each arm is a real placement and reclamation rule rather than an ablation of this method, and the differences between them are in which of the three components a family carries, which is what the ablations are built to separate. The allocation-only family appears twice, once budgeted on the measured committed count and once on the requested rank total, such that the account a family keeps and the placement it performs can be separated.

The arms are listed below under their full names, each followed by the short name used where a table has to be narrow.

- Allocation-only, short name Allocation-only, writes each correction in closed form at the position the layer's own statistics point to, never reclaims and hence accumulates no displacement, and is budgeted on the measured committed count.
- Allocation-only (requested-rank account), short name Requested-rank, places the same arrivals under the same budget and charges each arrival the rank it requests.
- Fixed consolidation rank (unpriced), short name Fixed rank, consolidates a group at a fixed rank chosen without consulting the displacement tolerance, and reads the cost only after the group is overwritten.
- Adapter-merging, short name Adapter-merging, folds the whole live set into one adapter at a fixed rank (Ilharco et al., 2023).
- Model-merging (interference-resolved), short name Model-merging, folds the live set into one update at a fixed rank with the interference between the merged corrections resolved; its displacement reading is the truncation residual of that merge (Yadav et al., 2023).

- Null-space constrained editing, short name Null-space, writes each arrival orthogonally to the directions the layer already holds (Fang et al., 2025).
- Orthogonal-adapter continual update, short name Orthogonal-adapter, keeps a continually updated adapter that writes each update where the layer is empty (Wang et al., 2023).
- ROME (rank-one model editing), short name ROME, is the rank-one member of the allocation-only family (Meng et al., 2022) and writes each correction in closed form.
- MEMIT (mass editing), short name MEMIT, is the mass-editing member of the allocation-only family (Meng et al., 2023).
- Constrained fine-tuning (norm-bounded), short name Norm-bounded, fine-tunes the layer with every update shrunk to a fixed norm (Zhu et al., 2020).
- RankReclaim (ours) carries all three components: the committed-capacity ledger, priced reclamation and overlap-budgeted placement.

Two families named in the related work are not run. The side-memory editor needs a scope classifier to decide which queries it answers, and the learned editor network needs a trained hypernetwork; neither has a counterpart at one layer of fixed width, and neither is tabulated.

APPENDIX A.11: EVALUATION METRICS

Write S_T for the live set the stream leaves behind, r_i for the requested rank of correction i, κ_i for the directions its removal would return and X_i for its probe set. The capacity metrics are read off the ledger,

$$\text{committed count} = K(S_T), \quad \text{slack} = \sum_{i \in S_T} r_i - K(S_T), \quad \text{attr. err.} = \frac{1}{|S_T|} \sum_{i \in S_T} (r_i - \kappa_i), \tag{17}$$

and the behaviour metrics are read on the probes each correction was installed with,

$$\text{edit success} = \frac{1}{T} \sum_{i=1}^{T} \mathbb{1}[u_i = 1], \quad \text{quality} = \frac{1}{T} \sum_{i=1}^{T} Q_i, \quad \text{served} = \left| \{\, i : i \text{ admitted} \,\} \right|, \tag{18}$$

where u_i is the indicator average of Eq. (5). Edit success and locality follow the reading established for single-edit and mass-edit evaluation (Meng et al., 2022; 2023), applied here to a stream rather than to one edit; the capacity metrics of Eq. (17) are defined in this work and have no earlier counterpart. The realized displacement is the accumulator of Eq. (11) read at the end of the stream, and the free-width draw is η of Eq. (12) averaged over the arrivals.

APPENDIX A.12: HYPERPARAMETER SPECIFICATIONS

The displacement tolerance δ is the median over the arrivals of each arrival's largest response on its own probes, computed from the full stream before the loop begins, and the ratio of Eq. (10) is stabilized with $\nu = 10^{-3}$. The reference settings are the group-size cap 6, the free-width budget 0.35, the spectral tolerance 0.05 and the width band 16; the rows below sweep the displacement tolerance, the group-size cap and the spectral tolerance one at a time. Table 7 condenses the three conditions into one table, with the reference settings carried on their own rows. Figure 9 draws the spectral tolerance on both of its axes.

On this method's quality column the spectral tolerance moves the reading furthest, from 0.1111 at 0.01 to 0.0486 at 0.40; the displacement tolerance moves it between 0.0486 and 0.1042, and the group-size cap between 0.0764 and 0.0833.

The allocation-only family, which places the same arrivals with no reclamation at all, reads a higher end-stream quality in 16 of the 20 rows, this method reads higher in 3, and the two are level in the remaining 1. The reading under the displacement tolerance is not monotone either: this method reads 0.0903 at a quarter of a correction's own response, 0.0486 at a half and 0.1042 at three quarters, then 0.0764 and 0.0972 at one and a half and at twice it. A tolerance that admits a larger move returns more capacity, and what that capacity is worth to the objective does not follow the tolerance in one direction.

Table 7: Running conditions: three knobs and the reference settings.

	RankReclaim (ours)		Fixed consolidation rank (unpriced)	
	quality	committed	quality	committed
spectral tolerance 0.01	0.1111	16.0	0.1181	16.0
spectral tolerance 0.02	0.1111	16.0	0.1389	14.0
spectral tolerance 0.10	0.0833	16.0	0.0347	15.0
spectral tolerance 0.25	0.0833	15.0	0.0417	15.0
spectral tolerance 0.40	0.0486	15.0	0.0694	14.0
spectral tolerance 0.50	0.0625	14.0	0.0625	14.0
group-size cap 2	0.0764	16.0	0.0347	15.0
group-size cap 3	0.0833	16.0	0.0347	15.0
group-size cap 4	0.0764	16.0	0.0347	15.0
displacement tolerance 0.25 of a correction's own response	0.0903	16.0	0.0347	15.0
displacement tolerance 0.50 of a correction's own response	0.0486	16.0	0.0347	15.0
displacement tolerance 0.75 of a correction's own response	0.1042	16.0	0.0347	15.0
displacement tolerance 1.50 of a correction's own response	0.0764	15.0	0.0347	15.0
displacement tolerance 2.00 of a correction's own response	0.0972	16.0	0.0347	15.0
reference on zsre at ceiling multiple 1.5	0.3068	14.0	0.4205	16.0
reference on zsre at ceiling multiple 3.0	0.0833	16.0	0.0347	15.0
reference on counterfact at ceiling multiple 1.5	0.1705	16.0	0.3068	16.0
reference on counterfact at ceiling multiple 3.0	0.1905	15.0	0.0595	15.0
reference on mquake at ceiling multiple 1.5	0.2692	13.0	0.1827	16.0
reference on mquake at ceiling multiple 3.0	0.0750	16.0	0.0500	16.0
	Allocation-only		Adapter-merging	
	quality	committed	quality	committed
spectral tolerance 0.01	0.0903	16.0	0.0556	9.0
spectral tolerance 0.02	0.0764	16.0	0.0069	6.0
spectral tolerance 0.10	0.1667	16.0	0.1042	11.0
spectral tolerance 0.25	0.1667	16.0	0.1042	11.0
spectral tolerance 0.40	0.1667	16.0	0.1042	10.0
spectral tolerance 0.50	0.1667	16.0	0.1042	10.0
group-size cap 2	0.1667	16.0	0.1042	11.0
group-size cap 3	0.1667	16.0	0.1042	11.0
group-size cap 4	0.1667	16.0	0.1042	11.0
displacement tolerance 0.25 of a correction's own response	0.1667	16.0	0.1042	11.0
displacement tolerance 0.50 of a correction's own response	0.1667	16.0	0.1042	11.0
displacement tolerance 0.75 of a correction's own response	0.1667	16.0	0.1042	11.0
displacement tolerance 1.50 of a correction's own response	0.1667	16.0	0.1042	11.0
displacement tolerance 2.00 of a correction's own response	0.1667	16.0	0.1042	11.0
reference on zsre at ceiling multiple 1.5	0.5000	16.0	0.0455	4.0
reference on zsre at ceiling multiple 3.0	0.1667	16.0	0.1042	11.0
reference on counterfact at ceiling multiple 1.5	0.3068	16.0	0.3068	16.0
reference on counterfact at ceiling multiple 3.0	0.1905	16.0	0.0119	9.0
reference on mquake at ceiling multiple 1.5	0.3846	16.0	0.1250	6.0
reference on mquake at ceiling multiple 3.0	0.0625	15.0	0.0625	9.0

Robustness to Running Conditions. RankReclaim keeps the committed count on zsre between 14 and 16 over a fifty-fold range of spectral tolerance, five displacement tolerances and three group-size caps, because every admission is tested against the measured count. The budget test reads whatever count the tolerance produces, and admission adapts to that count instead of a requested rank. Over the same five displacement tolerances and three group-size caps, RankReclaim's end-stream quality reads 0.0486 to 0.1042 and 0.0764 to 0.0833, against 0.0347 for the fixed-rank consolidation at every setting. The capacity RankReclaim keeps in use therefore does not hinge on a tuned threshold, and the reference settings sit in a wide working range.

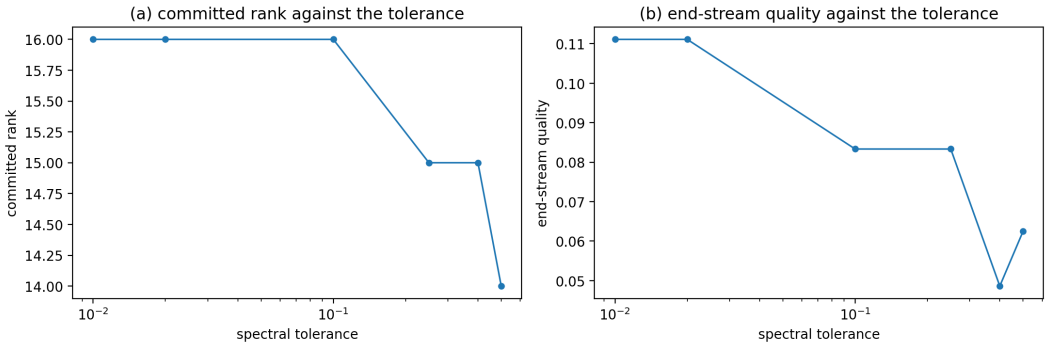

Figure 9: (a) Committed count and (b) end-stream quality ↑ against the spectral tolerance of Eq. (2), both axes logarithmic in the tolerance, on this method and the zsre stream.

APPENDIX A.13: EVALUATION PROTOCOL

Two levels of aggregation appear in the tables and they are never mixed. Within one stream, quality, edit success, locality and the free-width draw are means over the per-arrival readings. Across streams, a cell described as pooled over the two primary streams is the mean of the two per-stream readings, not a ratio re-formed from summed numerators and denominators. Where a single figure is quoted for the three streams at once it is named as a mean, and the per-stream readings are given beside it.

Every reading is a deterministic numerical construction over the edit streams, produced on one machine. No reading is an average over seeds or over repeated runs: the construction is fixed by the stream identity, and running it again returns the same file.

APPENDIX A.14: THE MAIN RESULTS TABLES

Table 8 carries every arm under its full name on the three streams at the reference setting, one row per arm and one column group per stream. Its seven columns are end-stream quality (qual), edit success (succ) and locality (loc), read on the corrections the stream specifies; the committed count (comm) and the slack of Eq. (2) and Eq. (3); the accumulated displacement (disp) of Eq. (11) as the stream leaves it; and the number of corrections served. An arm that never reclaims reads a displacement of 0.0 by construction rather than by measurement. The count of cells on which another arm reads better is taken over the twenty-one cells formed by these seven columns and the three streams.

Table 9 sweeps the four width bands and carries eight of the eleven arms; the requested-rank account, the unpriced fixed rank and norm-bounded fine-tuning are not included in Table 9. Each band contributes three columns: the committed count, end-stream quality, and the ceiling ratio, which is the committed count divided by the band label. Every cell is pooled over the two primary streams under the rule of Appendix A.13, which is why half-integer counts appear.

Figure 4 draws the ceiling ratio of each arm as grouped bars in the left panel and end-stream quality in the right panel, with the four bands placed at equal spacing. No arm reads the band label at all four bands: the orthogonal-adapter, allocation-only and mass-editing arms read it at three, this method and rank-one editing at two, the null-space arm at one, and the two merging arms at none. On the ceiling ratio this method reads highest at width 12, jointly with three other arms; at width 16 four arms read 1.0 where it reads 0.9688; and at widths 24 and 32 it reads 1.0 alongside four others.

APPENDIX A.15: THE ABLATION TABLES

Table 10 gives every arm one row and three column groups, the slack, the attribution error of Eq. (17) and the committed count, each read at six stream positions. Each arm is read from the first arrival at which its own account reads the layer full. The positions are therefore per arm and per stream: an arm that never reaches its own ceiling is read from the first arrival instead, and a position does not

Table 8: Every arm under its full name on the three streams at a budget of 16.

Method	qual ↑	succ ↑	loc ↑	comm	slack ↓	disp ↓	served ↑
zsre							
Allocation-only	0.1667	0.0556	0.2639	16.0	5.0	0.0	9.0
Allocation-only (requested-rank account)	0.0903	0.0556	0.5347	12.0	4.0	0.0	6.0
Fixed consolidation rank (unpriced)	0.0347	0.0	0.0972	15.0	6.0	3.8040	16.0
Adapter-merging	0.1042	0.0556	0.3264	11.0	1.0	2.4505	18.0
Model-merging (interference-resolved)	0.2014	0.1111	0.8472	11.0	1.0	0.8844	18.0
Null-space constrained editing	0.0694	0.0	0.7292	16.0	1.0	0.0	7.0
Orthogonal-adapter continual update	0.1667	0.1111	0.6736	16.0	0.0	0.0	6.0
ROME (rank-one model editing)	0.1250	0.0556	0.2708	15.0	0.0	0.0	15.0
MEMIT (mass editing)	0.0486	0.0	0.4722	16.0	0.0	0.0	6.0
Constrained fine-tuning (norm-bounded)	0.1042	0.0556	0.9722	16.0	3.0	0.0	7.0
RankReclaim (ours)	0.0833	0.0	0.3611	16.0	0.0	1.0031	13.0
counterfact							
Allocation-only	0.1905	0.0476	0.3333	16.0	4.0	0.0	8.0
Allocation-only (requested-rank account)	0.1786	0.0952	0.6429	13.0	3.0	0.0	7.0
Fixed consolidation rank (unpriced)	0.0595	0.0476	0.2857	15.0	1.0	2.7923	16.0
Adapter-merging	0.0119	0.0	0.4702	9.0	1.0	2.2388	21.0
Model-merging (interference-resolved)	0.0714	0.0476	0.9405	9.0	1.0	0.7009	21.0
Null-space constrained editing	0.1488	0.0476	0.8155	16.0	0.0	0.0	7.0
Orthogonal-adapter continual update	0.2440	0.1905	0.7143	16.0	0.0	0.0	7.0
ROME (rank-one model editing)	0.1369	0.0952	0.3929	16.0	0.0	0.0	16.0
MEMIT (mass editing)	0.1429	0.0476	0.4464	16.0	0.0	0.0	7.0
Constrained fine-tuning (norm-bounded)	0.1488	0.0952	0.9702	16.0	4.0	0.0	8.0
RankReclaim (ours)	0.1905	0.0952	0.6190	15.0	2.0	0.9929	15.0
mquake							
Allocation-only	0.0625	0.0	0.1125	15.0	12.0	0.0	13.0
Allocation-only (requested-rank account)	0.0750	0.0500	0.5563	11.0	5.0	0.0	6.0
Fixed consolidation rank (unpriced)	0.0500	0.0	0.1250	16.0	8.0	3.2196	17.0
Adapter-merging	0.0625	0.0500	0.3812	9.0	3.0	2.2886	20.0
Model-merging (interference-resolved)	0.1125	0.1000	0.8187	9.0	3.0	0.7185	20.0
Null-space constrained editing	0.0688	0.0500	0.7250	16.0	0.0	0.0	6.0
Orthogonal-adapter continual update	0.1500	0.1000	0.7500	16.0	0.0	0.0	6.0
ROME (rank-one model editing)	0.0938	0.0	0.2938	15.0	2.0	0.0	17.0
MEMIT (mass editing)	0.0625	0.0500	0.5062	16.0	0.0	0.0	6.0
Constrained fine-tuning (norm-bounded)	0.0625	0.0500	0.6750	15.0	12.0	0.0	13.0
RankReclaim (ours)	0.0750	0.0	0.3625	16.0	5.0	0.9811	15.0

correspond to one arrival across the rows. The cells are pooled over the two primary streams; hence the committed count of 12.5 on the requested-rank account is a two-stream mean and not a reading on mquake. The attribution error carries no arrow because it is a property of an account rather than a quantity an arm is scored on.

Table 11 reads six arms at five group-size caps, three columns each: the end-state truncation scale, the accumulated displacement and the capacity returned. The end-state truncation scale is one truncation of the live set as the stream leaves it, not an accumulator over the events the displacement column sums, and the two are not on the same scale. An arm with no reclamation to price returns nothing.

Table 12 reads the four reclaiming arms stream by stream. Interference-resolved merging's displacement column is the truncation residual of the merge it performs, where the other arms accumulate the change between the total update before an event and the total update after it; the two are different quantities and that column is not ranked against the others.

Table 13 reads five arms at six working points of the free-width budget, three columns each: the drawn mass, the same share in percent, and end-stream quality. This method's drawn share and the constraint of Eq. (13) are taken against one denominator, the arrival's own unconstrained fit of Eq. (12). An arm that rescales the factor before placing it is read against the factor it produces,

Table 9: Four adapter width bands.

Method	width 12			width 16		
	committed	quality	ceiling ratio	committed	quality	ceiling ratio
Null-space constrained editing	11.5000	0.2151	0.9583	16.0	0.1091	1.0
Adapter-merging	6.5000	0.0681	0.5417	10.0	0.0580	0.6250
Model-merging (interference-resolved)	6.5000	0.1190	0.5417	10.0	0.1364	0.6250
Orthogonal-adapter continual update	11.5000	0.2712	0.9583	16.0	0.2054	1.0
Allocation-only	10.5000	0.1454	0.8750	16.0	0.1786	1.0
MEMIT (mass editing)	11.5000	0.2151	0.9583	16.0	0.0957	1.0
ROME (rank-one model editing)	11.0	0.1110	0.9167	15.5000	0.1310	0.9688
RankReclaim (ours)	11.5000	0.1290	0.9583	15.5000	0.1369	0.9688
Method	width 24			width 32		
	committed	quality	ceiling ratio	committed	quality	ceiling ratio
Null-space constrained editing	23.5000	0.0252	0.9792	30.0	0.0213	0.9375
Adapter-merging	10.0	0.0152	0.4167	17.5000	0.0701	0.5469
Model-merging (interference-resolved)	10.0	0.0916	0.4167	17.5000	0.1189	0.5469
Orthogonal-adapter continual update	24.0	0.1717	1.0	32.0	0.1966	1.0
Allocation-only	24.0	0.0569	1.0	32.0	0.0534	1.0
MEMIT (mass editing)	24.0	0.0222	1.0	32.0	0.0	1.0
ROME (rank-one model editing)	24.0	0.0533	1.0	32.0	0.0396	1.0
RankReclaim (ours)	24.0	0.0290	1.0	32.0	0.0427	1.0

Table 10: Slack ↓, attribution error and committed count of every arm at six stream positions.

Method	stream position					
	1	2	3	4	5	6
slack ↓						
Requested-rank	3.5	3.5	3.5	3.5	3.5	3.5
Allocation-only	4.5	4.5	4.5	4.5	4.5	4.5
Adapter-merging	2.0	0.0	0.5	1.0	1.5	2.5
Model-merging	2.0	0.0	0.5	1.0	1.5	2.5
Null-space	0.5	0.5	0.5	0.5	0.5	0.5
RankReclaim (ours)	2.0	0.0	0.5	0.5	0.5	1.0
attribution error						
Requested-rank	0.9405	0.9405	0.9405	0.9405	0.9405	0.9405
Allocation-only	0.8819	0.8819	0.8819	0.8819	0.8819	0.8819
Adapter-merging	0.4375	0.0000	0.3333	0.3750	0.5000	0.7500
Model-merging	0.4375	0.0000	0.3333	0.3750	0.5000	0.7500
Null-space	0.0714	0.0714	0.0714	0.0714	0.0714	0.0714
RankReclaim (ours)	0.5625	0.0000	0.1250	0.1111	0.1000	0.1818
committed count						
Requested-rank	12.5	12.5	12.5	12.5	12.5	12.5
Allocation-only	16.0	16.0	16.0	16.0	16.0	16.0
Adapter-merging	9.5	3.5	4.0	6.5	8.5	9.5
Model-merging	9.5	3.5	4.0	6.5	8.5	9.5
Null-space	16.0	16.0	16.0	16.0	16.0	16.0
RankReclaim (ours)	16.0	14.0	14.0	15.0	15.5	15.5

and its column consequently stands on another denominator. The comparison arms do not read the budget at all, and their columns are flat.

Table 14 tests the bound of Eq. (7) one consolidation event at a time, at six truncation ranks in two group bands of sizes 4 to 6 and 12 to 16, giving twelve readings in all. Each band contributes the bound, the realized displacement, the tightness ratio, which is the realized displacement as a share of the bound, and the violation rate. Figure 10 draws the same readings, each curve a mean over the group sizes in its band.

Table 11: Challenge 2, priced reclamation over the group-size cap.

	RankReclaim (ours)			Fixed consolidation rank (unpriced)		
	end-state truncation scale	displacement	returned	end-state truncation scale	displacement	returned
group size 4	0.8742	1.0139	8.0	0.7534	3.8040	15.0
group size 6	0.8762	1.0031	8.0	0.7534	3.8040	15.0
group size 8	0.8762	1.0031	8.0	0.7534	3.8040	15.0
group size 12	0.8762	1.0031	8.0	0.7534	3.8040	15.0
group size 16	0.8762	1.0031	8.0	0.7534	3.8040	15.0
	Adapter-merging			Allocation-only		
	end-state truncation scale	displacement	returned	end-state truncation scale	displacement	returned
group size 4	0.9952	2.4505	26.0	0.9394	0.0	0.0
group size 6	0.9952	2.4505	26.0	0.9394	0.0	0.0
group size 8	0.9952	2.4505	26.0	0.9394	0.0	0.0
group size 12	0.9952	2.4505	26.0	0.9394	0.0	0.0
group size 16	0.9952	2.4505	26.0	0.9394	0.0	0.0
	Model-merging (interference-resolved)			Orthogonal-adapter continual update		
	end-state truncation scale	displacement	returned	end-state truncation scale	displacement	returned
group size 4	0.8432	0.8844	26.0	1.0328	0.0	0.0
group size 6	0.8432	0.8844	26.0	1.0328	0.0	0.0
group size 8	0.8432	0.8844	26.0	1.0328	0.0	0.0
group size 12	0.8432	0.8844	26.0	1.0328	0.0	0.0
group size 16	0.8432	0.8844	26.0	1.0328	0.0	0.0

Table 12: Returned capacity, realized displacement and end-stream quality for the reclaiming families, stream by stream.

	RankReclaim (ours)			Fixed consolidation rank (unpriced)		
	returned	disp	quality	returned	disp	quality
zsre	8.0	1.0031	0.0833	15.0	3.8040	0.0347
counterfact	6.0	0.9929	0.1905	11.0	2.7923	0.0595
mquake	4.0	0.9811	0.0750	6.0	3.2196	0.0500
	Adapter-merging			Model-merging (interference-resolved)		
	returned	disp	quality	returned	disp	quality
zsre	26.0	2.4505	0.1042	26.0	0.8844	0.2014
counterfact	27.0	2.2388	0.0119	27.0	0.7009	0.0714
mquake	24.0	2.2886	0.0625	24.0	0.7185	0.1125

The Displacement Bound. The bound of Eq. (7) is never violated and stays close to the displacement it bounds. Across twelve readings in two group bands, the violation rate is 0.0 and the realized displacement reaches 0.6133 to 0.8762 of the bound. The bound falls with the truncation rank, from 1.3685 to 0.4959 in the small band and from 2.3624 to 1.1473 in the large one, and the realized displacement falls with it. The tightness peaks at truncation rank 1 in both bands, 0.8416 and 0.8762, the most aggressive truncation and the one with the largest displacement. This is because the bound multiplies the first discarded singular value by the largest probe norm, while the realized displacement counts only the part of that residual the stored probes meet, which keeps the bound conservative for every survivor.

Table 13: Challenge 3, overlap-budgeted placement over the free-width budget.

	RankReclaim (ours)			Model-merging (interference-resolved)			Adapter-merging		
	drawn	drawn (%)	quality	drawn	drawn (%)	quality	drawn	drawn (%)	quality
recapture budget 0.05	0.0410	4.0965	0.0377	0.6679	66.7945	0.1364	0.6837	68.3663	0.0580
recapture budget 0.15	0.1181	11.8143	0.0317	0.6679	66.7945	0.1364	0.6837	68.3663	0.0580
recapture budget 0.35	0.2641	26.4150	0.1369	0.6679	66.7945	0.1364	0.6837	68.3663	0.0580
recapture budget 0.55	0.4315	43.1452	0.1329	0.6679	66.7945	0.1364	0.6837	68.3663	0.0580
recapture budget 0.75	0.5944	59.4412	0.1310	0.6679	66.7945	0.1364	0.6837	68.3663	0.0580
recapture budget 0.95	0.5990	59.9049	0.1473	0.6679	66.7945	0.1364	0.6837	68.3663	0.0580
	Null-space constrained editing			Constrained fine-tuning (norm-bounded)					
	drawn	drawn (%)	quality	drawn	drawn (%)	quality			
recapture budget 0.05	0.7832	78.3232	0.1091	0.7792	77.9235	0.1265			
recapture budget 0.15	0.7832	78.3232	0.1091	0.7792	77.9235	0.1265			
recapture budget 0.35	0.7832	78.3232	0.1091	0.7792	77.9235	0.1265			
recapture budget 0.55	0.7832	78.3232	0.1091	0.7792	77.9235	0.1265			
recapture budget 0.75	0.7832	78.3232	0.1091	0.7792	77.9235	0.1265			
recapture budget 0.95	0.7832	78.3232	0.1091	0.7792	77.9235	0.1265			

Table 14: The displacement bound over six truncation ranks.

	small groups, sizes 4 to 6			
	bound	displacement	tightness	violation (%)
truncation rank 1	1.3685	1.1518	0.8416	0.0
truncation rank 2	1.2333	0.9482	0.7688	0.0
truncation rank 3	1.1295	0.8627	0.7638	0.0
truncation rank 4	1.0548	0.7779	0.7375	0.0
truncation rank 6	0.8108	0.5410	0.6672	0.0
truncation rank 8	0.4959	0.3041	0.6133	0.0
	large groups, sizes 12 to 16			
	bound	displacement	tightness	violation (%)
truncation rank 1	2.3624	2.0700	0.8762	0.0
truncation rank 2	1.9790	1.6588	0.8382	0.0
truncation rank 3	1.8117	1.4127	0.7798	0.0
truncation rank 4	1.6527	1.2499	0.7563	0.0
truncation rank 6	1.3961	0.9648	0.6911	0.0
truncation rank 8	1.1473	0.8745	0.7622	0.0

APPENDIX A.16: THE THREE EFFECT DIRECTIONS

A margin below zero means the direction did not come out as stated on that stream. Table 15 carries the three effect directions and the margin each reads on the stream that reads worst. Each direction is read on the quantity its own component is about, stream by stream, and the margin reported is the reading on the worst stream rather than the mean of the three.

Two halves hold by construction and enter no margin. The ledger charges an arrival the directions it adds, which cannot exceed the rank it requests; hence no arrival costs the ledger more than its requested rank. What C1 reads is the two quantities that can fail, quality and displacement at matched arrivals, and both come out negative on their worst stream, at -0.006944 and -1.003060. Because the priced rule executes no rank whose accumulated displacement would leave the tolerance, it cannot exceed the tolerance either; that half is a property of Eq. (8). What C2 reads is the half that can fail: whether a consolidation of the same stream chosen without consulting the tolerance exceeds it. This half is read only where the pricing returned capacity, since a stream on which nothing was reclaimed prices nothing. Its worst-stream margin is 1.799363 on counterfact.

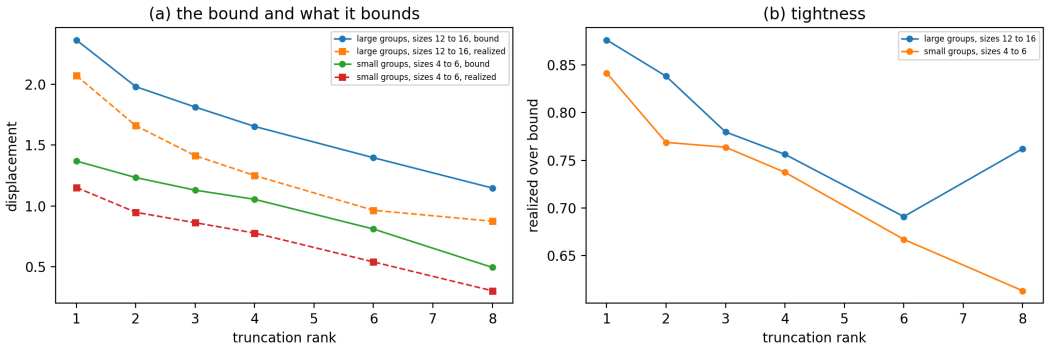

Figure 10: (a) The Eq. (7) bound ↓ and the realized displacement ↓ it bounds, at six truncation ranks in two group-size bands. (b) The realized displacement as a share of the bound ↑.

What each reading bears on: the first says the ledger is to be read on the corrections the layer admits rather than on the displacement it costs or the quality it returns; the second says that without pricing, reclamation reduces to unconditional consolidation; the third says the free-width budget caps the share drawn from the free width.

C3 is read on one quantity: whether the constrained placement keeps the drawn share at or under the budget. It does, at all six working points. The second half the direction was stated with asks whether an unconstrained placement draws back the span a consolidation released, and it is not measured anywhere in these experiments. What the loosest budget reads on the same stream is the draw of a fit the cap no longer binds, not a reading of the released span being re-occupied. That half is withdrawn rather than reported.

The displacement bound of Eq. (7) is not violated. Table 14 reads it one consolidation event at a time and finds the realized displacement under the bound at every truncation rank in both group bands, with a violation rate of 0.0 throughout. That table is the only place the bound of Eq. (7) is read.

Table 15: The three effect directions and their worst-stream margins.

Effect direction	worst-stream margin	worst-stream margin, second half
C1: ledger against a requested-rank account, on quality then on displacement	-0.0069	-1.0031
C2: an unpriced consolidation against the displacement tolerance	1.7994	–
C3: the drawn mass against the free-width budget, at the reference budget of 0.35	0.0695	–

APPENDIX A.17: SETTINGS IN WHICH ANOTHER ARM READS BETTER

Of the 21 stream-by-metric cells of the main table, 20 carry a row here. Adapter-merging takes six, interference-resolved merging four, the allocation-only family and the orthogonal-adapter family three each, and the null-space arm and norm-bounded fine-tuning two each. The cause codes of Table 16 are, in the order the codes run:

1. a merging family folds the whole live set into one update at a fixed rank, which buys the reading it wins on at a committed count well below the band and at a displacement it does not price;
2. an allocation-only family places the same arrivals with no reclamation and hence with no displacement; where reclamation has not yet paid for itself, it is therefore ahead;

3. a family that writes orthogonally to what the layer holds, or shrinks every update to a fixed norm, keeps the reading clean at the cost of the corrections it then cannot serve;
4. a rule that consolidates or writes rank-one without consulting the tolerance buys the reading it wins on at a displacement it does not report.

Table 16: Every stream-by-metric cell in which another arm reads better.

Metric	Arm reading better	quality	committed	displacement	position	cause
		zsre				
quality	Model-merging (interference-resolved)	0.0833	16.0	1.0031	1.0	1
edit success	Model-merging (interference-resolved)	0.0833	16.0	1.0031	1.0	1
locality	Constrained fine-tuning (norm-bounded)	0.0833	16.0	1.0031	1.0	3
committed	Adapter-merging	0.0833	16.0	1.0031	1.0	1
displacement	Allocation-only	0.0833	16.0	1.0031	1.0	2
served	Adapter-merging	0.0833	16.0	1.0031	1.0	1
		counterfact				
quality	Orthogonal-adapter continual update	0.1905	15.0	0.9929	1.0	3
edit success	Orthogonal-adapter continual update	0.1905	15.0	0.9929	1.0	3
locality	Constrained fine-tuning (norm-bounded)	0.1905	15.0	0.9929	1.0	3
committed	Adapter-merging	0.1905	15.0	0.9929	1.0	1
slack	Null-space constrained editing	0.1905	15.0	0.9929	1.0	3
displacement	Allocation-only	0.1905	15.0	0.9929	1.0	2
served	Adapter-merging	0.1905	15.0	0.9929	1.0	1
		mquake				
quality	Orthogonal-adapter continual update	0.0750	16.0	0.9811	6.0	3
edit success	Model-merging (interference-resolved)	0.0750	16.0	0.9811	6.0	1
locality	Model-merging (interference-resolved)	0.0750	16.0	0.9811	6.0	1
committed	Adapter-merging	0.0750	16.0	0.9811	6.0	1
slack	Null-space constrained editing	0.0750	16.0	0.9811	6.0	3
displacement	Allocation-only	0.0750	16.0	0.9811	6.0	2
served	Adapter-merging	0.0750	16.0	0.9811	6.0	1

APPENDIX A.18: THE BOUNDARY REGIMES

Table 17 enumerates the regimes probed at the boundary, with the readings taken in each. The last column is read off the run rather than off the row's name: it carries 1 where the reclamation returned capacity on that row and 0 where it returned none. These are boundary probes run on the zsre stream at ceiling multiples 0.8 and 3.0. They are a separate measurement from the budget sweep of Table 13, which runs the counterfact and zsre streams at ceiling multiple 3.0, and the two share three budget labels without sharing a stream set: at a budget of 0.35 the probe reads a drawn mass of 0.2478 against the sweep's 0.2641, and at 0.05 and 0.95 they differ as well. Neither table corrects the other and no reading should be carried across them.

Only one row returns nothing. With the stream stopped before the ceiling the budget never binds, the returned capacity is 0.0 and the quality reads 0.3438 against 0.0833 at the reference setting. With the budget at the layer width the committed count settles at 96.0 and the reclamation still returns 4.0 directions at an accumulated displacement of 1.0245; and where the drawn mass already sits inside a budget of 0.95 the placement is not inert either: the reclamation returns 4.0 directions at 0.9867. What switches the mechanism off on these streams is the budget never binding, not the width of the layer and not a loose placement budget. The five group-size rows return 8.0 directions each, at an accumulated displacement of 1.0031, and 1.0139 at the smallest cap.

Reclamation at the Layer Width. Reclamation stays active when the per-layer budget equals the layer width: at $d = n$, RankReclaim returns 4 directions at a displacement of 1.0245, and at a free-width budget of 0.95 it returns 4 at 0.9867. At the opposite extreme, a free-width budget of 0.05, it returns 19 directions at 0.9263. This is because the loop of Eq. (15) reclaims whenever the free

Table 17: The boundary regimes and the readings taken in each.

Regime	quality	committed	displacement	drawn	returned	reclamation fired
budget at the layer width, $d = n$	0.0	96.0	1.0245	0.1555	4.0	1
stream stopped before the ceiling	0.3438	10.0	0.0	0.2628	0.0	0
group size 4	0.0764	16.0	1.0139	0.2476	8.0	1
group size 6	0.0833	16.0	1.0031	0.2478	8.0	1
group size 8	0.0833	16.0	1.0031	0.2478	8.0	1
group size 12	0.0833	16.0	1.0031	0.2478	8.0	1
group size 16	0.0833	16.0	1.0031	0.2478	8.0	1
overlap already inside the budget, recapture budget 0.95	0.1458	16.0	0.9867	0.5486	4.0	1
overlap at the pinned budget, recapture budget 0.35	0.0833	16.0	1.0031	0.2478	8.0	1
overlap far outside the budget, recapture budget 0.05	0.0278	12.0	0.9263	0.0393	19.0	1

width falls short of the requested rank, whatever the width of the layer, and the price stays within 0.03 of the reference reading of 1.0031 in the first two regimes.