Emergent, Coordinate-Specific Structure in Dense Parameter Updates for Continual Learning

Anonymous authors
Paper under double-blind review

Abstract

Continual learning (CL) requires models to learn new tasks while retaining previous knowledge, yet sequential parameter updates can cause catastrophic forgetting. Parameter-efficient adaptation methods such as LoRA address this challenge by restricting updates to a low-rank subspace, limiting the degrees of freedom of each task update. We ask the converse: what structure emerges when the full parameter space remains unconstrained and is updated continuously across tasks? We freeze a pretrained model and train a same-sized parameter state from zero with standard AdamW. We find that continuous dense updating develops cross-task co-directionality and a contracting effective rank, in contrast to the rank expansion observed in LoRA, and that this structure favours knowledge retention across tasks. Coordinate and trajectory interventions show that retention depends on both where updates are written and whether their accumulation remains continuous: preserving update values while permuting their coordinates sharply reduces retention, while interrupting the trajectory at task boundaries also degrades it. At matched learning amounts, LoRA incurs greater per-task parameter displacement, accounting for much of its additional forgetting, with residual dependence on rank. Across the swept range, the dense adapter achieves better retention than LoRA, while its frontier intersects that of EWC. These results reveal that continuous unconstrained updating itself forms structured parameter dynamics with measurable consequences for continual learning.

1 Introduction

1.1 Removing the restriction

Adapting a large pretrained model to a new task normally means updating all of its parameters, which is expensive to store and to serve. LoRA (Hu et al., 2022) reduces that cost by restricting the learnable update to a pre-specified low-rank subspace, so that only a small number of extra parameters are trained. The restriction also bounds the degrees of freedom each task update may use: whatever the task requires, the update can express it only inside a rank-r subspace chosen before training begins. Parameter-isolation and adapter methods impose the same kind of constraint in other forms (Serrà et al., 2018), (Houlsby et al., 2019), (He et al., 2022).

In continual learning that bound is usually read as a defence against forgetting. We ask the converse. Take the most ordinary unrestricted object we can write down — a frozen pretrained model W_0 and a second parameter state W_A of the same shape, initialised at zero and trained with standard AdamW, so that the effective weights are

$$W_{\text{eff}} = W_0 + W_A \tag{1}$$

with nothing selected, protected, replayed or projected — and let it keep updating across tasks. What happens?

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.

1.2 What we find

The update does not stay unstructured. As tasks accumulate, the successive per-task updates of this plain dense adapter become progressively co-directional and its effective rank contracts, while a continuously updated LoRA under the same protocol moves the opposite way. The structure is worth something: at matched learning amounts the dense adapter retains more than LoRA. We then ask where the structure comes from, and answer with two interventions on the accumulation process itself. Permuting the coordinates of a fixed-magnitude update, so that the written values are unchanged but land elsewhere, sharply reduces retention. Interrupting the same accumulation at every task boundary reduces it too. Retention therefore depends on two properties rather than one: where updates are written, and whether their accumulation stays continuous. The same view prices the constraint we removed — at matched learning amounts LoRA's additional forgetting is largely the larger per-task displacement that the low-rank constraint forces, with a residual dependence on rank — and places the phenomenon against the other family of continual-learning controls, where we report a crossing of frontiers rather than an advantage.

1.3 Contributions

1. We show that a plain dense adapter forms structured updates. A full-size state initialised at zero and updated continuously across tasks with standard AdamW develops directional and spectral structure: successive per-task updates become co-directional and the effective rank contracts, whereas a continuously updated low-rank counterpart expands — the opposite direction (Section 3.2).
2. We show that this structure is worth retention. At matched learning amounts the dense adapter retains more than LoRA across the swept range (Table 1).
3. We show that retention depends on where an update is written, by an isometric intervention. A coordinate permutation holds the written multiset bit-for-bit fixed, so written energy is identical by construction and only the correspondence between value and coordinate changes. The permuted write destroys the just-learned task and costs -0.6796 retention at 124M, -0.6524 at 355M and -0.6245 at 1.55B, while coordinate concentration is unchanged (Section 4.4).
4. We show that retention also depends on whether accumulation stays continuous. Interrupting the trajectory at task boundaries degrades retention, and the machinery that performs the interruption is not what carries the benefit; the component that hurts is the continuous erosion of the accumulating state (Section 4.5).
5. We price the restriction we removed. Sweeping LoRA rank against step size onto matched learning amounts, forgetting is largely explained by the displacement the constraint forces, though a residual rank dependence survives controlling for it (Section 5.1); against the EWC frontier we report a crossing rather than an advantage (Appendix A.19).

Two boundaries constrain these claims, and we state them up front. The placement effect is not detectable on the level-free endpoint of our smaller 6.59M protocol, so we do not present it as protocol-independent. And against EWC we report a crossing of frontiers rather than an advantage (Appendix A.19).

2 Related work

Forgetting, and what is done about it. Sequential parameter updates overwrite earlier knowledge, and the established families answer that in different ways: regularisation penalises movement of parameters deemed important (Kirkpatrick et al., 2017), (Zenke et al., 2017); replay rehearses old data (Robins, 1995), (Beaulieu et al., 2020); architectural methods route tasks to disjoint or protected coordinates (Rusu et al., 2016), (Mallya & Lazebnik, 2018), (Serrà et al., 2018); and parameter-efficient adaptation restricts the update to a subspace chosen in advance (Hu et al., 2022), (Houlsby et al., 2019). Each answers forgetting by imposing something on the update. We ask the complementary question — what a full-size

update forms when nothing is imposed and the state simply keeps accumulating — and then which properties of that accumulation retention depends on.

Restricting the update space. A parallel line of work reduces adaptation cost by constraining which directions the update may use: LoRA factorises it as a low-rank product (Hu et al., 2022), and progressive networks, PackNet, HAT and adapter layers avoid interference by not sharing coordinates at all (Rusu et al., 2016), (Mallya & Lazebnik, 2018), (Serrà et al., 2018), (Houlsby et al., 2019). There the bound on the update is chosen before training and never revisited. We remove it, hold the parameter set fixed, let a full-size state keep updating, and ask what structure that process forms and whether the structure is what carries the benefit.

Fast weights and consolidation. Rapidly written weights alongside slowly changing ones go back to (Hinton & Plaut, 1987), and reappear as fast-weight programmers and linear-attention memory (Ba et al., 2016), (Schlag et al., 2021), (Miconi et al., 2018), (Behrouz et al., 2025), where the fast weights are part of the forward computation. Here the fast state of our diagnostic decomposition (Section 4) is an integrator whose contents are moved into the parameters at a boundary. Complementary-learning-systems accounts make the same architectural distinction (McClelland et al., 1995), (Kumaran et al., 2016); we do not claim to introduce it.

Selective consolidation. Regularisation methods estimate per-parameter importance and penalise movement accordingly (Kirkpatrick et al., 2017), (Zenke et al., 2017), so selection there is explicit and coordinate-level by construction. Our setting removes the importance estimator entirely and asks what remains. Selecting what to transfer into a stable store is itself an established principle: Leimer et al. gate consolidation per synapse on that synapse's own fast component (Leimer et al., 2019), Lindsey and Litwin-Kumar gate it on recall (Lindsey & Litwin-Kumar, 2024), and recent work separates a fast textual state from persistent parameters for continual adaptation of large language models (Tiwari et al., 2026). In each case a selector or a channel split decides what is transferred; our setting has neither, so the coordinate placement of a fixed-magnitude write is the only thing that varies.

Replay and projection. Replay constrains new gradients with old data (Robins, 1995), (Beaulieu et al., 2020) and projection constrains them geometrically (De Lange et al., 2022); both act on the update's direction or data support rather than on where a fixed-energy write lands.

Plasticity loss. Continual training can consume plasticity (Lyle et al., 2022), (Dohare et al., 2024), (He et al., 2025); our decay analysis (Section 4.6) is the same tension from the other side.

Task order and weight-space operations. The order in which tasks arrive changes the outcome (Bell & Lawrence, 2022), (Li & Hiratani, 2025), and in-context learning shows that weight-space updates can implement learned computation (von Oswald et al., 2023). Our permutation acts on the writeback increment and, because the gate is scalar, leaves energy exactly unchanged, which makes it a control rather than a perturbation.

The gap. Each family above answers forgetting by imposing something on the update: importance weights, replayed data, disjoint coordinates, or a pre-specified subspace. What has not been characterised is the structure a full-size update forms when none of that is imposed and the state simply keeps accumulating across tasks, nor which properties of that accumulation retention depends on. We isolate two — where an update is written, and whether its accumulation stays continuous — with interventions that hold everything else fixed.

3 The dense adapter, and what its updates form

3.1 The object

We freeze a pretrained backbone and add a second parameter state W_A of the backbone's shape (124,318,464 elements, 0.9990 of the backbone's 124,439,808). W_A is initialised at

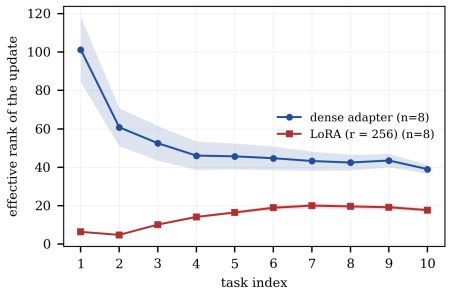

Figure 1: Effective rank of the update as tasks accumulate, for the dense adapter and for a continuously updated LoRA of rank 256 under the same protocol. Means over seeds with the standard deviation shaded; the dense adapter contracts and the low-rank counterpart expands. This is the regularity the paper opens with, and it is descriptive: Section 4.4 shows that the function is carried by coordinates, not by this spectrum.

zero and trained with standard AdamW under a continual-learning curriculum of ten tasks of 60 steps; the backbone never moves. Nothing is selected, protected, replayed or projected, and the update space is unrestricted: this is exactly the object that LoRA restricts, with its degrees of freedom left intact. We call it the dense adapter. Every number below comes from this one protocol unless a section says otherwise, with $n = 8$ to 16 seeds per cell.

Throughout, retention is the mean over tasks of the perplexity on a task at the end of that task divided by the same quantity after the final boundary: 1 is no forgetting, values above 1 are positive transfer between tasks, and lower values are more forgetting. The readout conventions, and why we prefer level-free ones, are in Appendix A.17.1.

3.2 Continuous updating develops directional and spectral structure

As tasks accumulate, the update that the continuous state forms becomes progressively co-directional: the mean cosine between successive per-task updates is $+0.189$, against $+0.008$ for a continuously updated low-rank counterpart under the same protocol. Over the same course its effective rank contracts, to 0.392 of its initial value (Figure 1), whereas the low-rank counterpart expands, to 2.793 — the opposite direction. This is a descriptive regularity, stable across seeds and consistent at every size we ran.

What it is not is a difference in how concentrated the update is. The Top-K support overlap between tasks is nearly identical in the two regimes (0.0346 against 0.0388), and the concentration statistics are indistinguishable between them as well. The contrast therefore lies in the geometry of the update rather than in the distribution of its mass. Section 4.4 shows that it is nonetheless the coordinates that carry the function, which is why we read this regularity as descriptive rather than explanatory.

The rest of the paper depends on there being something in the update to place, so we also record what the state contains after a curriculum. Running two opposite three-domain curricula on the same backbone, it develops a heavy-tailed coordinate profile with no selector present, and the per-seed cosine between that state and the gradient field of the following domain correlates with subsequent adaptation gain at $r \approx +0.87$ (Benjamini–Hochberg $q < 0.1$; leave-one-out $rin[0.81, 0.91]$; permutation $p \approx 0.000$).

This is an association, not an intervention, and we flag it as such: the state develops structure, and that structure is associated with later adaptation. Weakening the direction claim below does not affect the placement result, which does not depend on it.

The structure is carried rather than recomputed: swapping the fast states produced by two opposite curricula changes adaptation on an unseen domain in the same direction at three GPT-2 sizes, with a negligible start-point difference (Appendix A.26). Whether the specific direction is the causal carrier is not established (Appendix A.13).

Table 1: Retention at matched forward adaptation. Rows are the dense adapter's own rungs, so its column is measured and the rank columns are interpolated there; the difference is dense minus best rank, so a positive entry favours the dense adapter. The best rank is rank 256 except in the last row (rank 64). The dense sweep covers $[0.1308, 0.1958]$.

forward	dense	best rank	difference
0.1308	0.9968	0.9982	-0.0014
0.1454	0.9754	0.9669	+0.0084
0.1586	0.9625	0.9263	+0.0362
0.1705	0.9493	0.8898	+0.0596

The structure is worth retention. Table 1 compares the two families at matched learning amounts, and Figure 6 plots them as frontiers: at the dense ladder's own rungs the dense adapter is above the best of the four ranks at three of the four comparable points, and -0.0014 below rank 256 at the lowest, a difference inside seed noise. The two sweeps do not cover the same range — rank 4 reaches a learning amount as low as 0.1006, where its retention is 1.0544, while the dense sweep stops at 0.1308, and at the other end the dense frontier reaches 0.1958, above the highest learning amount reached by any rank frontier (0.1800) — so the comparison is stated over the swept range, and neither sweep extends past it.

4 What properties of the accumulation matter?

4.1 A diagnostic decomposition, and why a write exists at all

The dense adapter is a single object: it accumulates, but it does not expose why the retention appears. To ask which properties of the accumulation matter, we need to be able to move an accumulated update, and to interrupt the accumulation, while changing nothing else. For that we use a minimal decomposition in which each wrapped weight matrix carries a fast state W_f with

$$W_{\text{eff}} = W_{\text{slow}} + gW_f \tag{2}$$

and g a fixed scalar gate. During wake, W_f integrates the learner's own Adam-shaped gradients with a leak,

$$W_f \leftarrow \lambda W_f - \eta\tilde{\nabla}L \tag{3}$$

and at a task boundary a sleep step writes it back,

$$W_{\text{slow}} \leftarrow W_{\text{slow}} + \gamma W_f, \quad W_f \leftarrow \delta W_f \tag{4}$$

The writeback coefficient γ is a scalar, so the rule applies the same multiplier everywhere: it cannot create a coordinate difference, and it cannot protect one either. That is what makes the device useful as an instrument. It is emphatically not proposed as a method: we show in Section 4.5 that adding this machinery back onto the minimal implementation makes retention worse, not better. It appears here because it is the only construction that lets us move the accumulated update without changing anything else.

The full description of the decomposition — the fast state, the scalar writeback and the measurement conventions — is in Appendix A.15.

4.2 Position: an energy-matched permutation of the write

Does it matter where the accumulated update is written, or only how much of it there is? To separate where from how much, we compare the rule's own writeback against a control

that writes the same energy elsewhere. The control permutes the entries of γW_f within each matrix. Since a permutation preserves the multiset of values, the written energy is exactly equal:

$$\|A'\| = \|A\|, \quad A' = \Pi_\pi A \tag{5}$$

Both arms see the same history, the same task boundaries and the same probe seed, so the only difference is the correspondence between a written value and the coordinate that receives it. We call the rule's own writeback direct and the permuted control shufwrite (Equation 5).

The energy term. Comparing shufwrite against nocons, an arm that writes nothing, tests whether the energy of the write does anything on its own. Across the decay settings we swept, the two are statistically indistinguishable on all three metrics (every $|t| \leq 0.86$): the written energy is present in shufwrite and absent in nocons, and the difference does not show up. The amount written, on its own, is not what produces the effect.

The placement term. On the 124M independent protocol, paired by seed, direct reaches a lower final perplexity than shufwrite: $+1.0457 \pm 0.6103$ ($t = +5.94$, 11/12 seeds, $d_z = +1.71$), and $+0.8710 \pm 0.3992$ against nocons ($t = +7.56$, 12/12). Being level-free, this is not the starting point moving: randomised placement with the same energy lands near the no-write floor, aligned placement does not.

The scope condition. On our smaller 6.59M protocol the energy contrast is again null. The forward readout moves towards the aligned arm (+0.0109, $t = +5.42$, 12/12 seeds). That arm also starts from the worse point (+0.2857, $t = +1.82$), and the post-sequence endpoint does not separate the two at significance (-0.1820, $t =$ -1.85, same direction in 11/12 seeds). We therefore do not claim a level-free placement effect at 6.59M, and return to what this implies in Appendix A.20.

4.3 A graded version of the same manipulation

A two-point contrast shows that two placements differ, not that placement is a quantity with a scale. We therefore grade it: the coordinate fidelity of a write is the fraction of written values that land on the coordinate they were computed for, and we permute a controlled fraction of them, which preserves the written multiset and hence the energy exactly. The endpoint penalty increases monotonically as fidelity decreases. The levels, their paired statistics and the start-point guard on them are in Appendix A.24.

4.4 Position and retention, at three model sizes

The permutation contrasts above are read on a level-free endpoint. The same intervention can be made on retention, where the effect is far larger and can be measured at three model sizes.

At each task boundary the accumulated update is written into the persistent weights with a scalar multiplier $c = 0.5$, and the accumulating state is then reset. The control writes $\gamma\, w_{\text{fast}}[\pi]$, with π a fixed permutation, in place of $\gamma\, w_{\text{fast}}$; both arms share every hyperparameter, share the wake, and differ only in how the boundary writes.

Two properties make this a strict single-variable intervention, and both are verified rather than assumed.

The written values are unchanged, exactly. A permutation is an isometry, so $\|\text{add}\| = \gamma\|w_{\text{fast}}\|$ holds identically, not approximately: the multiset of written values, and therefore every symmetric statistic of the write, is bit-for-bit the same. Only the coordinates differ.

The pre-boundary state is unchanged, bit for bit. The wake is identical in every arm; the boundary is the only place they differ. The perplexity recorded immediately after the first task's wake — before any boundary runs — must therefore agree across arms. It does, to ten

decimal places, in every file containing both arms: eight seeds at 124M, eight at 355M and both seeds at 1.55B (e.g. 8.5725351376 for all four arms). The arms cannot have diverged before the intervention.

All four arms are identical before the first boundary and diverge only at it: writing to the update's own coordinates changes the just-learned task's perplexity by a factor of 0.999, while writing the same values permuted changes it by 8.58 (Table 5).

Over the full sequence retention drops by -0.6796 (t = -88.30), -0.6524 (t = -76.54) and -0.6245 (t = -40.26) at 124M, 355M and 1.55B, with the permuted arm worse in 8/8, 8/8 and 2/2 seeds. These averages run over the stages before the last. The final stage's term compares a reading taken at the end of training with a denominator taken after the final boundary, so it measures that boundary, rather than the retention of earlier history. Including it gives -0.7201, -0.7137 and -0.7096. The ordering "write to own coordinates > partial write > no write > permuted write" is stable at all three scales.

It is not a broken model. The permuted arm is better at learning the new domain: its forward adaptation gain rises from 0.2473 to 0.3001, and its final perplexity improves from 40.91 to 37.51. The intervention does not damage plasticity; it removes the ability to retain, which is the dissociation the claim requires.

It is not a change in how the update is distributed. Coordinate concentration is essentially unchanged by the permutation: the top-1% share moves from 0.0487 to 0.0498 and normalised entropy from 0.9766 to 0.9766. The effect is therefore attributable to where the update lands, not to how its mass is distributed.

4.5 Continuity: interrupting the accumulation at boundaries

Does retention also depend on whether the accumulation stays continuous? The boundary write is the instrument that makes the manipulation above possible, and it interrupts the trajectory: at every task boundary the accumulating state is reset and its contents are moved into the persistent weights. That interruption is not what the benefit rests on. Adding the boundary's components back one at a time onto the minimal implementation — a frozen backbone plus one full-size state, continuously updated across tasks — makes retention worse, not better.

Protocol. Every contrast in this subsection is computed inside a single harness, so all arms share one retention definition $R = \text{mean}_t\, a_t/e_t$, where a_t is the perplexity on task t at the end of task t and e_t the same quantity after the final boundary. We report within-harness paired differences only; these absolute values are not comparable with the retention columns of the tables above, which use different bookkeeping.

Retention is 0.871 for the minimal implementation, 0.743 when the boundary writes $c = g(1-\delta)$ (the conserving value), 0.704 at the shipped $c = 0.15$, 0.694 with the attenuation but no boundary, and 0.474 when the boundary writes nothing. Removing the machinery altogether is better than keeping it: the minimal implementation beats the design that separates the state into slow and fast parts and interrupts the trajectory at task switches, by +0.1676 (t = +18.78, 12/12 seeds).

The decomposition of that machinery is informative. The wake attenuation alone accounts for almost all of the loss: dropping it while leaving the trajectory otherwise uninterrupted gains -0.1776 (t = -23.77, 12/12 seeds). The boundary, by contrast, buys back only +0.0100 (t = +3.48). The component that hurts is the continuous erosion of the accumulating state, not the boundary itself.

The boundary is nevertheless doing something real when it writes: a boundary that writes nothing costs +0.2297 relative to one that writes (t = +57.26, 12/12). And when a boundary write is used, it must conserve the effective weight: the write coefficient that leaves W_{eff} unchanged across the boundary is $c = g(1-\delta)$, and setting it there gains +0.0397 over the shipped value (t = +13.61, 12/12).

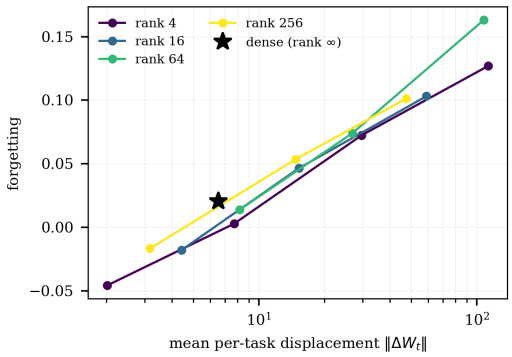

Figure 2: Forgetting against mean per-task displacement, at a common learning amount. One line per LoRA rank, each point a step size that reaches the same forward adaptation; the black star is the dense adapter, which is lower than every rank in both coordinates at its own learning amount. Displacement and forgetting rise together within a rank, and lower rank is displaced further for the same functional change. $n = 8$–16 seeds per point.

4.6 A boundary: decay is a trade-off, not a monotone knob

Sweeping the fast-state decay with everything else pinned gives a U-shaped level-free endpoint whose optimum is not where retention is best, so decay trades one axis against the other rather than acting as a pure gain (Appendix A.23).

5 Pricing the restriction we removed

5.1 Capacity-limited adaptation pays in displacement

The preceding sections characterise what the dense adapter forms, and split the retention it carries into two properties of the accumulation: where an update is written, and whether it stays continuous. We now ask what the restriction we removed actually costs, using the same intervention family on the constrained side.

We sweep LoRA rank against step size so that each rank reaches the same learning amount (forward adaptation), which is the only form of matching that is meaningful here: equal step size across ranks does not produce equal functional change. Four ranks each reach a common window of forward adaptation $[0.1323, 0.1482]$.

At a common forward adaptation inside that window the mean per-task displacement $\|\Delta W_t\|$ decreases with rank, apart from one exception at the top edge where rank 16 crosses above rank 4, and forgetting tracks it. The dense adapter is lower than every rank in both coordinates at its own learning amount (Figure 2).

Forgetting is largely accounted for by displacement: regressing it on displacement across all cells gives $R^2 = 0.793$. Displacement is the dominant channel by a wide margin, with a standardised coefficient of 0.055 against 0.014 for rank. Rank is not, however, fully absorbed: adding it as a second predictor raises the fit to $R^2 = 0.849$. The cost of the low-rank constraint is therefore carried mainly by the displacement it forces, with a residual dependence on rank that displacement does not explain.

One qualification. Displacement is nearly invariant to the random seed within a cell (for example, the eight seeds of one rank-16 cell span 14.83–15.67), so displacement accounts for variation between cells and not for seed-to-seed variation within one.

5.2 Relation to other continual-learning controls

The dense adapter is competitive with the EWC frontier rather than ahead of it: the two families cross at forward ≈ 0.182, with EWC leading below the crossing and the dense

adapter above it. Plain AdamW on the same store, with no boundary machinery at all, reproduces the device's forward gain and retains more at every matched position. Both comparisons, with their protocols and caveats, are in Appendix A.19 and Appendix A.20.

6 Limitations

The capacity-matched baseline exists only at 124M, and the placement contrast is not measured at comparable starting points at 355M and 774M; the claim is scoped to one protocol (10 tasks of 60 steps, 20-Newsgroups) and one backbone (GPT-2), with the adapter variants of the baseline not implemented; the placement effect is not detectable on the level-free endpoint of the 6.59M protocol, and we do not know which difference between the protocols is responsible; only the 0% level of the fidelity curve is start-matched, so its interior levels establish a direction rather than certified magnitudes; and the causal status of the history-specific direction remains open (Appendix A.9, Appendix A.13).

7 Conclusion

Continual learning asks a model to keep what it has learned while it keeps learning, and the standard answer restricts the update to a low-rank subspace. Asked the converse — what a full-size update forms when nothing is restricted — the update does not stay unstructured: its cross-task directions become co-directional and its effective rank contracts, while a continuously updated low-rank counterpart expands. That structure favours retention, and retention depends on two properties of the accumulation. The first is where an update is written: writing the same values permuted instead costs -0.6796, -0.6524 and -0.6245 retention at 124M, 355M and 1.55B. The second is whether it keeps accumulating without interruption. We do not claim that coordinate alignment is the unique mechanism of selective consolidation, nor that the specific direction carried by the history is causal (Appendix A.13). The same view prices the constraint we removed — at matched learning amounts its additional forgetting is largely the displacement it forces, with a residual rank dependence — so across the swept range the dense adapter retains more than LoRA, while against EWC we report a crossing of frontiers rather than an advantage.

AI use statement

(This section is required and does not count toward the page limit.)

In this work, we used generative AI tools for the following tasks with required disclosure: implementing methods (the training, evaluation and audit scripts used to produce every reported number); designing and providing feedback on research methodology and experiments (the calibration of the comparison protocol, the isometric coordinate intervention and its pre-boundary identity check, the write-back ablation ladder, the rank-matched displacement ladder and its mediation analysis, and the start-point decomposition); interpreting results (separating the level term from the retention term, reading the scale ladder as a non-detection, and reading the frontier comparison only over non-dominated rungs); and creating or modifying scientific figures (the figures in this submission were produced by scripts written with AI assistance). We additionally used generative AI tools for tasks with recommended disclosure: drafting parts of the paper and editing it for readability, suggesting a structure for the paper, proposing the title, formatting references, and summarising and locating existing literature. The remaining tasks on the required disclosure list are not applicable to this work. We have reviewed all AI-assisted work: every quantitative statement in the paper is emitted from a single set of bindings drawn from the archived result files, each number was checked against those files, and the code that produced the reported results was executed and its outputs audited against the raw runs recorded in Appendix A. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

Reproducibility statement

(This section is recommended and does not count toward the page limit.)

The protocols, seeds, measurement conventions and numerical-precision choices behind every reported number are recorded in Appendix A. That section states the operating point and the collapse of the per-coordinate gate, the random-number control and seed handling, the numerical precision of the reported quantities, and the statistical conventions used for every paired contrast; it also defines the claim-strength levels applied throughout the paper. Each result file is identified by the batch that produced it, and the exact identity used to recover the starting point, pre = after/(1 − forward), is given with its residual so that the decomposition can be re-derived from the archived endpoints alone. The scale ladder, the fidelity dose curve and the capacity-matched control are reported with their sample sizes and their start-point checks attached, so that a reader can reproduce both the effects and the limits placed on them.

References

Jimmy Ba, Geoffrey E Hinton, Volodymyr Mnih, Joel Z Leibo, and Catalin Ionescu. Using fast weights to attend to the recent past. In Advances in Neural Information Processing Systems, 2016.

Shawn Beaulieu, Lapo Frati, Thomas Miconi, Joel Lehman, Kenneth O. Stanley, Jeff Clune, and Nick Cheney. Learning to continually learn. In European Conference on Artificial Intelligence (ECAI), 2020.

Ali Behrouz, Peilin Zhong, and Vahab Mirrokni. Titans: Learning to memorize at test time. In Advances in Neural Information Processing Systems (NeurIPS), 2025.

Samuel J Bell and Neil D Lawrence. The effect of task ordering in continual learning. arXiv preprint arXiv:2205.13323, 2022.

Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Aleš Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. IEEE Transactions on Pattern Analysis and Machine Intelligence, 44(7):3366–3385, 2022.

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

Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. Towards a unified view of parameter-efficient transfer learning. In International Conference on Learning Representations (ICLR), 2022.

Naicheng He, Kaicheng Guo, et al. Spectral collapse drives loss of plasticity in deep continual learning. In arXiv preprint arXiv:2509.22335, 2025.

Geoffrey E Hinton and David C Plaut. Using fast weights to deblur old memories. In Proceedings of the Annual Conference of the Cognitive Science Society, 1987.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In International Conference on Machine Learning (ICML), pp. 2790–2799, 2019.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. In International Conference on Learning Representations (ICLR), 2022.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. Proceedings of the National Academy of Sciences, 114(13):3521–3526, 2017.

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

Pascal Leimer, Michael Herzog, and Walter Senn. Synaptic weight decay with selective consolidation enables fast learning without catastrophic forgetting. bioRxiv, 2019. doi: 10.1101/613265.

Zihao Li and Naoki Hiratani. Optimal task order for continual learning of multiple tasks. In International Conference on Machine Learning (ICML), 2025.

Jack W. Lindsey and Ashok Litwin-Kumar. Selective consolidation of learning and memory via recall-gated plasticity. eLife, 12:RP90793, 2024. doi: 10.7554/eLife.90793.

Clare Lyle, Mark Rowland, and Will Dabney. Understanding and preventing capacity loss in reinforcement learning. In International Conference on Learning Representations (ICLR), 2022.

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

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

Thomas Miconi, Jeff Clune, and Kenneth O Stanley. Differentiable plasticity: Training plastic neural networks with backpropagation. In International Conference on Machine Learning (ICML), 2018.

Anthony Robins. Catastrophic forgetting, rehearsal and pseudorehearsal. Connection Science, 7(2):123–146, 1995.

Andrei A. Rusu, Neil C. Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. In arXiv preprint arXiv:1606.04671, 2016.

Imanol Schlag, Kazuki Irie, and Jürgen Schmidhuber. Linear transformers are secretly fast weight programmers. In International Conference on Machine Learning (ICML), pp. 9355–9366, 2021.

Joan Serrà, Dídac Suris, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. In Proceedings of the 35th International Conference on Machine Learning (ICML), volume 80 of PMLR, pp. 4548–4557, 2018.

Rishabh Tiwari, Kusha Sareen, Lakshya A. Agrawal, Joseph E. Gonzalez, Matei Zaharia, Kurt Keutzer, Inderjit S. Dhillon, Rishabh Agarwal, and Devvrit Khatri. Learning, fast and slow: Towards llms that adapt continually. arXiv preprint arXiv:2605.12484, 2026.

Johannes von Oswald, Eyvind Niklasson, Edoardo Randazzo, João Sacramento, Alexander Mordvintsev, Andrey Zhmoginov, and Max Vladymyrov. Transformers learn in-context by gradient descent. In International Conference on Machine Learning (ICML), pp. 35151–35174, 2023.

Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In International Conference on Machine Learning (ICML), 2017.

A Experimental audit, measurement conventions, and corrections

This appendix records the implementation facts and corrections that determine what the numbers in the main text mean. The main text cites conclusions; the audit trail is here so that each conclusion can be traced to a mechanism rather than to a convention.

A.1 Metrics that contain the starting point

The starting point pre used by gain-40 and by the forward axis is measured after the treatment, because writeback moves the model and therefore moves the level at which the next probe begins. Decomposing gain-40 per seed with the exact identity $\Delta\text{gain-40} = \text{post}_B\Delta\text{pre}/(\text{pre}_A\text{pre}_B) - \Delta\text{post}/\text{pre}_A$ (residual about 10^{-18}) separates a start-point term from an endpoint term. Over the writeback-gain axis the start-point share of the apparent effect rises from 11.5% at $\gamma = 0.5$ to 77.4% at $\gamma = 1.5$. Amplifying the write therefore amplifies displacement faster than it amplifies benefit, which is why the load-bearing readouts in this paper are the end-of-probe perplexities and retention = mean(after/end).

A.2 Start-point decomposition of the capacity-matched control

pre is not stored in the archived records but is exactly recoverable from forward = 1 − after/pre, hence pre = after/(1 − forward). The maximum absolute residual over all arms and all tasks is 1.7×10^{-16}, so the decomposition is a recomputation and needs no new runs. Results are in Table 4. Two corrections are recorded. First, the $n = 6$ reading that the two families start at the same point does not survive extension to $n = 12$, and the main text no longer uses it. Second, pre is not single-valued in forward along the frontier: between adapter@lr3e-4 and adapter@lr1e-3, forward moves by +0.0043 while pre moves by +18.1, a slope of about 4200 PPL per unit forward. Interpolating across that segment lets the weight decide the level, so adapter@lr1e-3 is excluded from the interpolation basis; keeping it inflated the default point's Δpre to −1.489. On the full basis three of twelve seeds cross the segment; on the excluded basis none do.

A.3 Semantics of the no-write arm

In an early implementation the fast state was decayed unconditionally at every sleep, so the arm intended to write nothing was not a no-op: it still shrank the fast state. The implementation was corrected so that decay follows the write. Under the corrected semantics that arm means "neither written nor decayed" and is no longer equivalent to a 0% write baseline, so it is referenced with the corrected meaning throughout.

A.4 The operating point, and the collapse of the per-coordinate gate

The operating point is fixed by three independent observations. A per-coordinate gate used in an earlier version was measured to be a single scalar across $39,886,848$ coordinates ($n_{\text{eff}} = 0.9999999819$; the fraction with $|g - 0.5| > 0.005$ is zero), so the module provided no coordinate resolution. Pinning it to 0.25 improved both axes simultaneously. Pinning it to its learned mean of 0.5 was indistinguishable from leaving it free ($\Delta\text{post} = -0.0019$, $t = -0.03$), and deleting the module entirely changed none of nine metrics beyond the noise of repeated runs of an identical configuration ($|\Delta\text{post}| = 0.009$ against a noise band of 0.033–0.058). We therefore use a constant gate of 0.25.

A.5 Reproducibility and random-number control

An early batch showed unexplained variance whose cause was dropout in the backbone together with the absence of any seeding. All subsequent batches are explicitly seeded and use common random numbers across seeds. Two rounds of the decay grid were rerun on a different CPU instance and the level-free endpoint reproduced bit-for-bit (40.719581/40.719581 and 40.709594/40.709594), so numbers from different instances can be pooled.

The 124M fidelity experiment additionally required a check that its reference arms were comparable to the archived ones. They are: direct and shufwrite reproduce the archived 124M values bit-for-bit over all shared seeds (24 of 24 stored quantities differ by 0.000). An analogous check on the smaller harness failed, with a worst difference of 0.0072 in the relative-gain metric, which is why the fidelity experiment measures its reference arms inside its own batch rather than reusing archived ones.

A.6 Numerical precision

We compared fp32 ($n = 12$) against bf16 ($n = 4$) at 124M. The placement and energy contrasts are stable across precisions (τ-shuf of -0.02287 against -0.02270, a difference of 1.7×10^{-4}), so relative-gain comparisons may be pooled. Level-free endpoints may not be pooled: final PPL inflates by 30–45% under bf16 (the shufwrite placement contrast moves from +1.0457 to +1.5113; the two are the same contrast at two precisions, not two results). All perplexity tables here compare within a single precision.

A.7 Statistical conventions

All contrasts are paired by seed ($n = 6$ or $n = 12$), and we report paired t statistics with a threshold of $|t| >= 2$. No multiplicity correction is applied to the main results. Benjamini–Hochberg control is used once, for the correlation table in Section 3.2, where the leave-one-out range is also reported so that a single seed cannot drive the result. Point estimates with $n < 6$ are not cited: the same batch has previously produced opposite readings at $n = 2$ and $n = 6$.

A.8 Claim-strength levels used in this paper

We use four levels explicitly, and each main-text claim is written at the level its evidence supports.

1. Observation. Something is present. Stated directly.
2. Association. Two quantities covary without intervention. Written as "is associated with".
3. Intervention. A matched manipulation changes the outcome. Written as "changes" or "reduces".
4. Mechanism. The manipulation is the causal route. Written only where a directional intervention with adequate power exists.

The emergence result (Section 3.2) is L2. The placement result (Section 4.2 and Section 4.3) is L3. Nothing in this paper is written at L4.

A.9 Scale ladder

Scope of this ladder. It reads the level-free endpoint for the Scheme-A contrast (direct against shufwrite), at 124M, 355M and 774M. That is a different quantity from the retention-side ladder of Section 4.4, which measures the Scheme-B contrast at 124M, 355M and 1.55B and does not shrink. The two are not in conflict — they read different endpoints on partly different model sizes — but they must not be quoted interchangeably, and the shrinking reported here does not carry over to retention.

We ran the same three arms (direct, shufwrite, nocons) at 124M, 355M and 774M. Every level has only these three arms and no external baseline. The level-free endpoint contrast shrinks as the model grows: +1.511 ($t = +6.08$, $n = 4$), +0.886 ($t = +3.34$, $n = 4$), and +0.161 ($t = +1.17$, $n = 3$), which at 774M is not distinguishable from zero.

The numbers move monotonically, and the effect shrinks faster than the level does (the endpoint level falls to 80% of its 124M value while the contrast falls to 10.7%), so it is not merely a rescaling by level. But two cautions limit what we are willing to say.

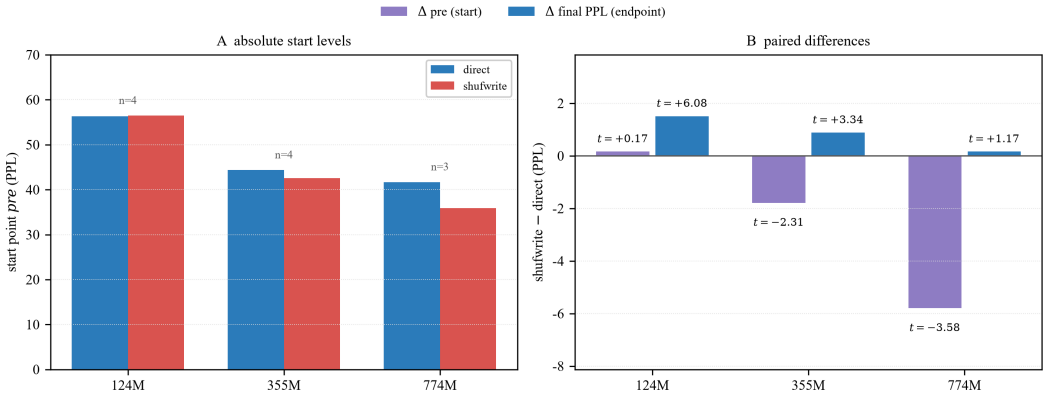

Figure 3: The scale ladder with its start-point check. A: absolute start levels. B: paired differences. The start-point difference changes sign and grows with scale while the endpoint difference shrinks; at 355M and 774M the start-point difference is 2.0 and 36.0 times the endpoint difference, in the opposite direction. The two larger levels therefore do not provide comparable starting points, and the ladder cannot be read as a scale trend.

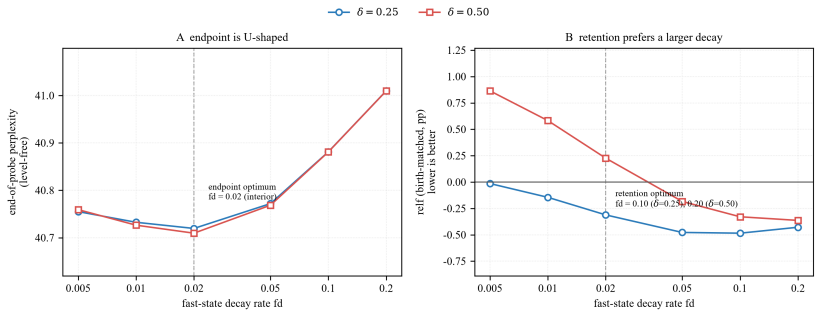

Figure 4: Fast-state decay as a dose axis ($n = 12$, paired). The level-free endpoint is U-shaped with its optimum at $\lambda = 0.02$; retention is best at 0.05–0.10. Each adjacent step agrees in sign across all 12 seeds with $|t| > 2$.

First, the starting points are not comparable at the larger levels. The paired start-point difference is +0.163 ($t = +0.17$) at 124M, -1.784 ($t = -2.31$) at 355M and -5.791 ($t = -3.58$) at 774M. At the larger two, the displacement is significant, runs opposite to the endpoint contrast, and is larger than the endpoint contrast itself. An endpoint difference measured while the starting point moves by 36 times that difference is not a clean estimate.

Second, the 774M endpoint contrast is not distinguishable from zero at $n = 3$. "The cost at 774M is only 0.16" is therefore a non-detection, not a measured cost.

We are consequently unable to distinguish two explanations for the shrinking: that the relative influence of writeback placement genuinely declines with scale, or that larger models tolerate a misplaced write better. Separating them would require a start-matched contrast at each scale, which a three-arm ladder cannot provide, since all three arms differ in their starting point. We report the ladder as a scope limit, not as a scale result. The 1.5B level has no valid data: that run failed for memory reasons and must not be read as a size effect.

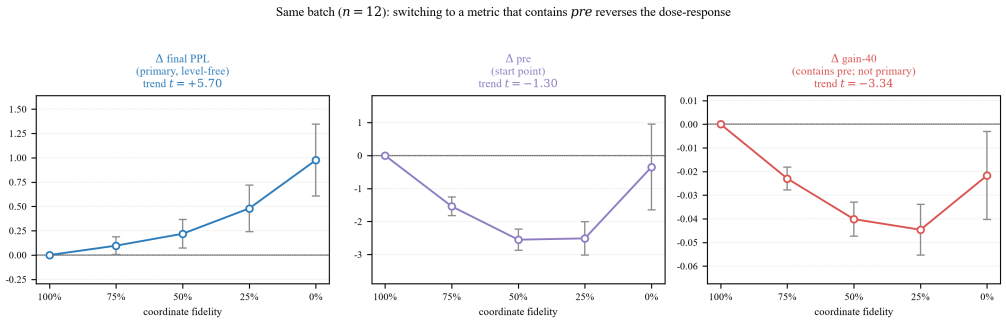

Figure 5: The same batch under three readouts ($n = 12$). Left: the level-free endpoint, trend $t = +5.70$. Middle: the starting point itself, trend $t =$ -1.30, with no detectable dose relationship. Right: gain-40, which contains pre, trend $t =$ -3.34, the opposite direction.

A.10 Fast-state decay as a dose axis

A.11 The same batch under three readouts

A.12 Why the comparison was designed this way

The comparison above would have come out the other way if the baseline had not been swept, or if the baseline had been a full fine-tune rather than a same-sized store. Both designs are common, and both would have produced an advantage for the device that reflects the comparison rather than the device. The discipline we adopted (frontier against frontier, matched capacity, level-free readouts, and decomposition wherever a start-dependent readout is unavoidable) is what made the negative result visible. It is also what made us discard several of our own earlier readings, catalogued in Appendix A.

A.13 Open items

1. Adapter versions of EWC, SI and Replay are not implemented. Importance estimates and replay buffers would have to be confined to a frozen backbone plus adapter; the current implementation refuses them explicitly rather than allowing a penalty computed over a frozen backbone to degrade silently to zero.
2. The history-specific direction is L2 only. A projection arm perturbs $\|W_f\|$ by about 3×10^{-4}, which has no power either way. A directional boost arm has been run, but its effect disappears under start-point decomposition: the intervention raises pre by up to 6.5 PPL ($t = +5.5$) while the endpoint barely moves ($|t| <= 1.9$), and giving a random arm the boost arm's starting point returns its gain to the random level. That arm's effect is therefore a start-point displacement, not causal evidence.
3. History specificity is bounded at about 7.1%, not yet separated from "any perturbation of comparable size does the same damage".
4. The 1.5B level has no valid data.
5. The scale ladder has three arms per level, no external baseline, and incomparable starting points at 355M and 774M.
6. Only the 0% level of the fidelity curve is start-matched, and the propagation slope that would certify the interior levels' magnitudes is not identified.

A.14 The two families as frontiers

The main-text table gives the matched-forward contrast at the dense adapter's own rungs. Plotted as frontiers (Figure 6) the two families are ordered the same way: the dense frontier lies above all four rank frontiers over the range both sweeps cover, with the single exception

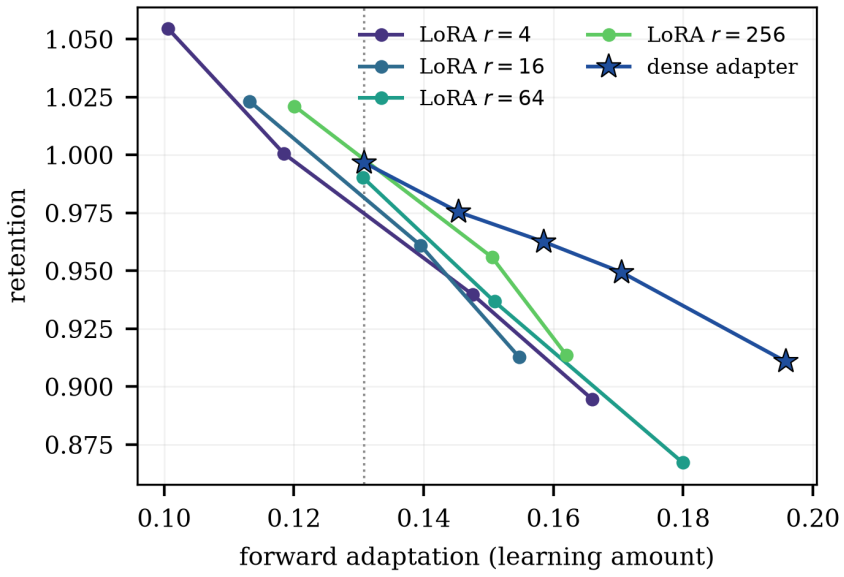

Figure 6: The dense adapter and the four LoRA ranks in the retention–learning plane, each family reduced to its non-dominated subset. Over the learning amounts the dense sweep covers, the dense frontier is above every rank frontier except at its lowest rung, where rank 256 is -0.0014 higher. The dotted line marks where the dense sweep starts: below it the rank families keep going, to learning amounts the dense ladder never measured, and there retention exceeds 1 (positive transfer between tasks), which is why the comparison is stated over the swept range only. Same cells as Table 1.

noted in the caption, while the rank families extend to lower learning amounts at which retention is above 1. Neither family is swept past the range shown.

A.15 The diagnostic decomposition in full

A.16 The fast state

The fast state is a leaky integral of the learner's own gradients: at each wake step we take the gradient of the training loss with respect to the effective weights, shape it with Adam, and accumulate it with decay λ (Equation 3). Two properties matter: the state is a function of the history of gradients rather than of one batch, so it can carry structure across steps, and its update is coordinate-wise, each coordinate integrating its own gradient stream. Neither property presupposes coordinate-level selection.

A.17 Scalar writeback and the fixed operating point

Sleep applies Equation 4. Because γ is a scalar, the written increment is γW_f with the same multiplier everywhere. A permutation of the coordinates of W_f therefore leaves $\|\gamma W_f\|$ unchanged, which is what allows us to build a matched-energy control (Section 4.2).

We fix a single operating point and do not tune it per experiment: in the 6.59M protocol it is $g = 0.25$, no plasticity gate, $\gamma = 1.0$, $\delta = 0.50$, and a leak of $\lambda = 0.02$. The choice of λ is itself a result (Section 4.6): the interior optimum of the decay axis, not a free parameter.

A.17.1 Measurement: why we prefer level-free readouts

A subtlety governs every number here. The starting point pre of a probe is measured after the treatment, because writeback moves the model and the level the probe begins from. Any metric that divides by pre or subtracts it mixes the treatment with a displacement of the starting level; an intervention that merely raises pre will look like a gain.

Table 2: Coordinate-fidelity dose curve (124M, $n = 12$, paired against the 100% anchor). Left three columns are the level-free endpoint. Right two columns are the start-point guard: the three interior levels sit significantly below the anchor at pre, while the 0% level does not differ from it. The endpoint ordering therefore does not run the same way as the start-point movement.

Fidelity	Δ final PPL	t	Same sign	Δpre	t
100%	—	—	—	—	—
75%	+0.0965	+2.37	10/12	-1.5401	-12.03
50%	+0.2187	+3.24	10/12	-2.5500	-17.51
25%	+0.4794	+4.40	11/12	-2.5104	-10.96
0%	+0.9757	+5.82	11/12	-0.3506	-0.59

We therefore separate two readout classes and report them separately. Level-free readouts are endpoints and within-arm ratios: the end-of-probe perplexity, written post PPL or final PPL, and retention = mean(after/end). Start-dependent readouts contain pre: the relative gain, written gain-40 = (pre − post)/pre, its unnormalised version drop = pre − post, and the forward axis. Where we must use a start-dependent quantity we report its exact decomposition alongside it; Appendix A gives the identity, its residual, and the corrections this discipline forced on earlier readings of our own data.

The practical consequence shapes the results section: where a level-free readout is available we lead with it, and where it is null while a start-dependent one is significant we say so rather than reporting the significant one.

A.18 Relation to other continual-learning controls

A.19 Comparison with the regularisation frontier

The dense adapter is competitive with the regularisation frontier rather than ahead of it, so we state the result as a crossing. We sweep EWC over eight penalty strengths and the dense adapter over six learning rates, and compare the two families at matched forward adaptation. Both are swept families, since comparing a swept family against a single operating point is not a comparison; and both are reduced to their non-dominated subsets first, one rung dropped from the dense family and three from the EWC family (Figure 10 in the appendix).

The frontiers cross at forward ≈ 0.182. EWC retains more at low learning amounts, by up to +0.0385 at forward 0.150; the dense adapter retains more at high learning amounts, by up to +0.0025 at forward 0.186. We claim no overall advantage over EWC, and note only that the dense adapter reaches this frontier without Fisher estimation, a replay buffer, or per-parameter importance weights.

A.20 Plain AdamW on the same store

The baseline in that comparison is the object this paper is about: a same-sized side weight driven by plain AdamW on a frozen backbone, with no boundary machinery at all. Under strict capacity matching at 124M it reproduces the device's forward gain and retains more at every matched position, so the writeback mechanism does not push the frontier outward. That points the same way as the continuity ablation above — the interruption is what costs retention — and we read it as corroboration rather than as an independent test of that claim. The design, the frontier, the start-point decomposition, the $n = 6 \to n = 12$ power artefact that changed the reading, and the caveats that qualify this comparison are given in Appendix A.22.

A.21 Tables for the results in the main text

The tables supporting the main-text results are collected here so that the main text stays within the page limit. Each caption states the sample size and the paired contrast it reports.

Table 3: Retention gap at matched forward, mechanism against the interpolated baseline frontier (124M, $n = 12$, paired). Retention is the load-bearing axis because it contains no starting point. The device is lower at every one of its frontier points.

Mechanism arm	forward	Δ retention	t
$\eta = 1.2 \times 10^{-4}$	+0.0936	−0.0124	−17.27
$\eta = 4 \times 10^{-4}$	+0.1386	−0.0167	−9.40
$\eta = 6 \times 10^{-4}$ (default)	+0.1523	−0.0292	−14.85
$\eta = 1.2 \times 10^{-3}$	+0.1824	−0.0368	−11.43
$\eta = 4 \times 10^{-3}$	+0.1979	−0.0631	−3.25

Table 4: Start-point decomposition of the default operating point, at both sample sizes. At $n = 6$ the starting point looked matched; at $n = 12$ the same signed shift resolves. The retention gap survives at both sample sizes and is the load-bearing quantity. Companion endpoint values: Δafter = -0.342 (t = -1.50) at $n = 6$ and -0.228 (t = -2.18) at $n = 12$.

Sample	Δpre	t	Δ retention	t
$n = 6$ (original)	-0.590	-1.74	-0.0277	-17.40
$n = 12$ (extended)	-0.390	-2.12	-0.0292	-14.85

A.22 The capacity-matched cost comparison (moved from the main text)

Everything so far changes what is written while holding the writeback rule fixed. It does not answer the comparison a practitioner would ask: relative to simply training a same-sized side weight directly, what does this device buy? We ran that comparison at 124M under strict capacity matching, and the answer is a boundary rather than a win. Design. The baseline receives the object the device writes into: the fast store from make_state(), 124,318,464 elements, 0.9990 of the backbone's 124,439,808, read out through the same forward pass and gate. It freezes the backbone and drives that store with plain AdamW only: no leak, no writeback. Both sides are swept into frontiers (seven learning rates for the baseline, seven step sizes for the device), because comparing a swept frontier against a single operating point, or the reverse, is not a comparison. Protocol: 10 tasks of 60 steps, new backbone, $n = 12$. Result. The device's frontier lies inside the baseline's (Figure 7, Table 3). Six of its seven operating points are bracketed (some baseline setting is significantly better on one axis and not significantly worse on the other), and two are dominated on both axes simultaneously. The one point that is not bracketed falls outside the range the baseline was swept over, and trades a 0.0076 advantage in forward against a 0.0077 deficit in retention: an exchange, not an advantage. Plain AdamW on the same store reproduces the device's forward gain and retains more at every matched position. Where the cost lives. In the device's conservative corner it reaches a comparable forward with $|W_f| = 0.63$–1.78 against 9.4–17.5 for the baseline: the written amount differs by about an order of magnitude and the retention gap does not shrink accordingly. The cost therefore tracks the act of writing the increment into the slow weights, not its size: the step off the pretrained point does not scale with how much is written. The start-point question for this comparison. The forward axis contains pre, so we decompose it using the exact recovery of pre from after and forward recorded in Appendix A. It has to be reported with its sample size attached, because extending it changed the reading: At $n = 6$ the two families appeared to sit at the same starting point (Δpre = -0.590, t = -1.74, not significant) and to end at the same level (Δafter = -0.342, t = -1.50, not significant). Extending to $n = 12$ shows that the "same starting point" reading was a power artefact: the point estimate grows and the shift resolves (Δpre = -0.390, t = -2.12; Δafter = -0.228, t = -2.18). The correct statement is therefore not that the two families start identically. It is that the level gap is small (0.39 PPL) relative to the retention gap, and that retention is significant under every basis we tried (Table 4). Two measurement facts behind these numbers are recorded in Appendix A, because both change what they mean: the matched-point retention gap is 0.013–0.028 at $n = 6$ and 0.012–0.063 at $n = 12$, so the earlier interval is the $n = 6$ value of the same quantity rather

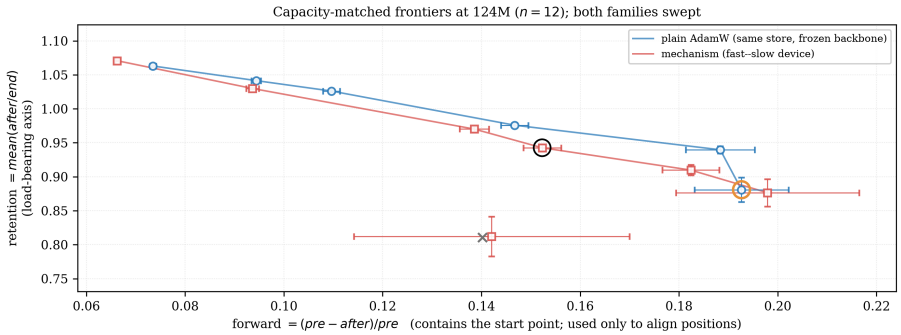

Figure 7: Capacity-matched frontier at 124M ($n = 12$). Blue: plain AdamW on the same fast store with a frozen backbone. Red: the device. Both families are swept. The device's frontier sits inside the baseline's. The orange circle marks adapter@lr1e-3, which stays on the frontier only through a +2.3% gain in forward bought with +18.1 PPL of level, and is excluded from the start-point interpolation; the black circle marks the device's default operating point, Δ retention $= -0.0292$ ($t = -14.85$, 12/12). Grey crosses are dominated baseline rungs; error bars are ± 2.2 standard errors.

than a superseded one; and the interpolation producing them is ill-conditioned at one arm, which is why that arm is excluded from the basis (the excluded basis gives -0.390). What this does not say. The baseline's EWC, SI and Replay adapter variants are not implemented (Appendix A.13), so the claim here is exactly "does not push the frontier outward relative to plain AdamW", not "is worse than importance-based or replay methods".

A.23 Fast-state decay as a trade-off (moved from the main text)

If the fast state were simply a smoother of recent history, more decay would be monotonically better. It is not: sweeping the leak over six values with everything else pinned gives a U-shaped level-free endpoint with its optimum at $\lambda = 0.02$, while retention is best at 0.05–0.10. The two optima are not in the same place, so λ is a knob that trades one axis against the other rather than a pure gain. Each adjacent step agrees in sign across all 12 seeds with $|t| > 2$ (Figure 4 in the appendix).

The obvious alternative explanation is that large decay simply empties the fast state, leaving nothing to write. It does not: at $\lambda = 0.20$ the state still retains 17–19% of its norm at the optimum. The endpoint degrades because the integrator compresses history too aggressively and loses task-internal temporal structure, not because the state disappears.

A.24 A graded version of the same manipulation (moved from the main text)

A two-point contrast shows that two particular placements differ. It does not show that placement is a quantity with a scale. We therefore make the manipulation graded. Define the coordinate fidelity of a write as the fraction of written values that land on the coordinate they were computed for. To realise fidelity $(1 - f)$, choose a random subset of size fN and permute the values within that subset, leaving the remaining $(1 - f)N$ coordinates untouched. The written multiset is unchanged, so energy is again exactly preserved; only the fraction of the write that keeps its coordinate varies. $f = 0$ reduces bit-for-bit to direct and $f = 1$ to a full permutation. All five fidelity levels run inside a single job per seed, so the comparison is paired by construction.

The curve is monotone (Figure 8, Table 2). All four non-anchor levels differ from the aligned control at their own level of significance, and all ten level pairs agree in direction with "lower fidelity is worse". The increments are +0.0965, +0.2187, +0.4794 and +0.9757 (Table 2). A linear contrast across the five levels (the right statistic for a graded claim, with weights that sum to zero so that any per-seed shift cancels exactly) gives $t = +5.70$ with 11/12 seeds

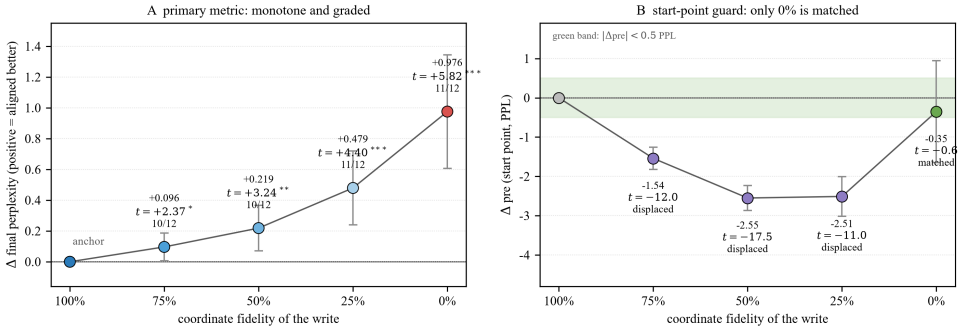

Figure 8: Placement as a graded quantity (124M, $n = 12$). A: the level-free endpoint as a function of coordinate fidelity. Error bars are 97.5% confidence intervals; each point is labelled with its paired t and the number of seeds agreeing in sign. The endpoint penalty increases as coordinate fidelity decreases. B: the start-point guard on the same levels. The green band is the range in which $|\Delta\text{pre}|$ is too small to matter (0.5 PPL). Only the 0% level falls inside it, and that level is also where the endpoint effect is largest.

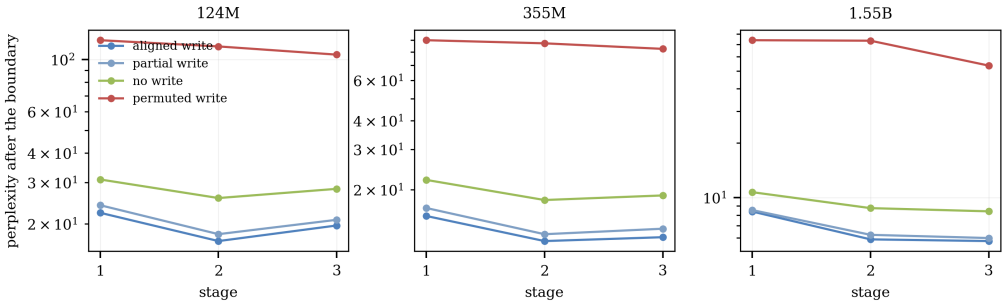

Figure 9: Perplexity on the task just learned, measured after each boundary (log scale). All four arms are identical before the first boundary; they diverge only at it, and in the same order at every model size. The permuted write is off the top of the useful range by an order of magnitude.

agreeing. The endpoint penalty increases as coordinate fidelity decreases, with the largest contrast at the start-matched 0% level.

Two controls support the shape. The 0% endpoint is not an artefact of one permutation draw: the graded family's 0% arm and the independently constructed full permutation differ by -0.0665 (t = -1.26, 7/12). And the two arms shared with the archived 124M batch (direct and shufwrite) reproduce the archived values bit-for-bit across all shared seeds, so the curve is on the same scale as the two-point contrast above.

Two controls on this curve are recorded in the appendix. The 0% endpoint is not an artefact of a single permutation draw, and only the 0% level is start-matched, so the interior levels establish the direction and the shape of the relation rather than certified magnitudes. We also report that reading the same batch through a start-dependent readout reverses the apparent trend, for the mechanical reason that the interior levels sit below the anchor at pre (Appendix A).

A.25 The first boundary, at three model sizes

Fig. 9 plots the perplexity of the just-learned task immediately after the boundary, for all four arms, at each size. The ordering is the same everywhere: writing the accumulated update to its own coordinates leaves the task essentially intact, a partial write costs a little, writing nothing costs more, and writing the same values permuted destroys it.

Table 5: The first boundary, measured on the just-learned task. All four arms are identical before the boundary and diverge only at it. Perplexity; 1.55B, seed 0.

arm	post-wake	post-boundary	factor
write to own coordinates	8.5725	8.5682	×0.999
partial write	8.5725	8.8437	×1.03
no write	8.5725	11.1151	×1.30
write to permuted coordinates	8.5725	73.5595	×8.58

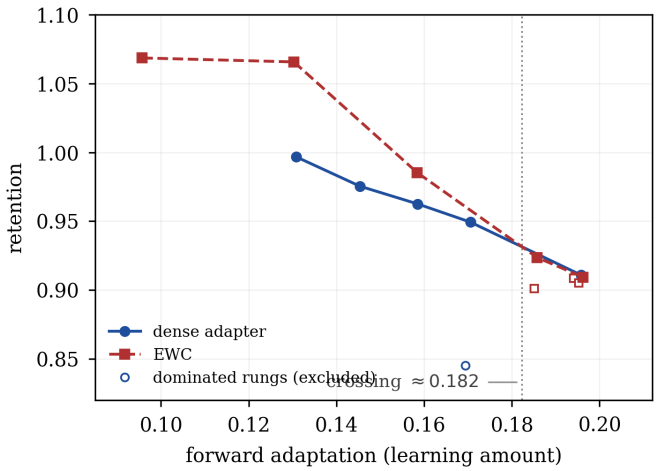

Figure 10: Retention against forward adaptation for two swept families. Blue circles are the dense adapter, red squares EWC. Filled markers are the non-dominated rungs; open markers are dominated rungs, excluded from the frontier and from the interpolation. The isolated red marker at the top left is the strongest penalty, $\lambda = 10^7$, whose retention exceeds 1 because the tasks share a vocabulary (positive transfer). The frontiers cross at forward ≈ 0.182.

A.26 Supporting detail moved from the main text

The structure is carried rather than recomputed. Swapping the fast states produced by two curricula changes adaptation on an unseen domain with a negligible start-point difference, in the same direction at three GPT-2 sizes (Δ final $= +0.775/ + 0.442/ + 0.905$, $t = +9.33/ + 4.14/ + 5.91$). Whether the specific direction is the causal carrier is not established (Appendix A.13).