

Exploring the Role of Representational Drift in Overcoming Plasticity Loss in Artificial Neural Networks

Author:
Arturo Comín Lucio (S6068553)

Supervisors:
Gido van de Ven
Robbert Havekes

Research group:
Machine Learning group
Artificial Intelligence, Bernoulli Institute

MSc Biology
UNIVERSITY OF GRONINGEN

August 2026

Abstract

Biological neural circuits can maintain stable behaviour while internal representations continue to change over time, a phenomenon known as representational drift. Artificial neural networks trained in continual learning settings often suffer from catastrophic forgetting and loss of plasticity, yet how standard continual learning strategies constrain internal representation change remains poorly characterised. This project quantifies representational change under regularization-based, replay-based, and architecture-based continual learning on a 50-context permuted-MNIST benchmark, comparing naive fine-tuning, synaptic intelligence (SI), experience replay (ER), and context-dependent gating (XdG) paired with SI, under both Adam and SGD and under task-incremental and domain-incremental readouts. Change is measured with a multi-metric pipeline (linear CKA, orthogonal Procrustes distance, and parameter and representational cosine distances) and related to forgetting and learnability on new tasks. Under Adam, at matched mean accuracy, SI suppresses geometric change more than ER. Under SGD, forgetting is higher, and the accompanying representational change is correspondingly larger. Accuracy-based loss of plasticity remains small at this horizon, except for SGD with replay. Deliberately inducing internal variability rarely increased measured drift without harming retention, although mild dropout revived silent hidden units.

Contents

1 Introduction **1**
- 1.1 Learning over a lifetime: biological versus artificial systems . . . 1
- 1.2 What is representational drift? . . . 1
- 1.3 Continual learning and the stability-plasticity dilemma . . . 2
- 1.4 Three continual learning strategy families . . . 3
- 1.5 Artificial neural networks as a computational window on biological learning . 4
- 1.6 Gap, objectives, and research questions . . . 4
- 1.7 Hypotheses . . . 5

2 Methods **6**
- 2.1 Experimental overview . . . 6
- 2.2 Benchmark and continual learning protocol . . . 7
- 2.3 Model architecture and training . . . 7
  - 2.3.1 Joint-training capacity calibration . . . 7
  - 2.3.2 Optimiser calibration . . . 8
  - 2.3.3 Training objective . . . 8
- 2.4 Continual learning methods implemented . . . 8
  - 2.4.1 Naive fine-tuning . . . 9
  - 2.4.2 Synaptic intelligence . . . 9
  - 2.4.3 Experience replay . . . 10
  - 2.4.4 Context-dependent gating . . . 10
  - 2.4.5 XdG + SI hybrid . . . 10
  - 2.4.6 Hyperparameter tuning procedure . . . 11
- 2.5 Representational change and drift measurement . . . 11
  - 2.5.1 The importance of metric choice . . . 12
  - 2.5.2 Parameter cosine distance . . . 13
  - 2.5.3 Representational cosine distance . . . 13
  - 2.5.4 Linear CKA change . . . 13
  - 2.5.5 Orthogonal Procrustes distance . . . 14
  - 2.5.6 Primary metrics and post-hoc analyses . . . 14
  - 2.5.7 Metric agreement and rank correlations . . . 15
- 2.6 Performance and plasticity metrics . . . 15
- 2.7 Induced representational-change interventions . . . 16

3 Results **17**
- 3.1 Capacity calibration and hyperparameter tuning . . . 17
- 3.2 Continual learning performance and forgetting . . . 17
- 3.3 Representational change by continual learning strategy . . . 18
- 3.4 Coupling change to learnability and forgetting . . . 22

3.5 Loss of plasticity and structural diagnostics . . . 23
3.6 Metric agreement and measurement validity . . . 25
3.7 Induced change and plasticity recovery . . . 26

4 Discussion **30**
4.1 Main findings . . . 30
4.2 Interpreting change under the stability-plasticity lens . . . 31
4.3 Artificial networks as a model of biological representational change . . . 32
4.4 Limitations . . . 32
4.5 Future work . . . 33
4.6 Conclusion . . . 34

A Appendix: supplementary figures and tables **38**
A.1 Long-horizon loss of plasticity: first approach . . . 38
A.2 Joint-training capacity calibration . . . 40
A.3 SGD learning-rate probe . . . 43
A.4 Method hyperparameter sweeps . . . 44
A.5 Matched comparison summaries for task-IL . . . 49
A.6 Locked-setting trajectories for the remaining combinations . . . 50
A.7 Matched-horizon drift by reference task . . . 55
A.8 Loss-of-plasticity diagnostics under task-IL . . . 57
A.9 Full induced-change Δ table . . . 58
A.10 Induced-change dead-unit and rank curves . . . 63

1 Introduction

1.1 Learning over a lifetime: biological versus artificial systems

Animals learn throughout their lives. New experiences reshape synaptic connections while abilities and memories often remain available after days, months, or years of learning. Artificial neural networks struggle to match this combination of stability and adaptability when data arrive as a continuous stream rather than as a fixed training set [1, 2]. Without dedicated continual-learning mechanisms, updating a network on new tasks typically disrupts weights that supported earlier tasks, producing catastrophic forgetting [1]. Even when forgetting is partially controlled, networks can gradually lose their ability to learn new tasks while still appearing to retain old tasks (a phenomenon known as the loss of plasticity) [3]. Forgetting and loss of plasticity represent two of the main differences between biological and artificial neural networks that need to be addressed [4].

To understand how biological circuits can incorporate new experiences without simply overwriting what was learned before, we turn to neuroscience literature. Advances in electrophysiology and calcium imaging now make it possible to track the same neurons across days and weeks while behaviour remains stable [5–7]. These recordings revealed a counter-intuitive phenomenon: internal population codes for fixed stimuli or tasks can drift substantially at the single-neuron level even when animals perform the same behaviour reliably. This phenomenon was termed representational drift [8, 9]. Whether this observed drift reflects noise, exploration of redundant representational solutions, or an active strategy for memory maintenance is still debated [10, 11]. Linking drift to how biological networks balance stability and plasticity may therefore inform why artificial networks suffer from forgetting and loss of plasticity, and inform strategies for mitigating both problems [4, 12]. This biological variation in neural codes leads us to a more precise definition of how drift behaves across different environments.

1.2 What is representational drift?

We define representational drift as gradual change in internal neural representations over time despite stable task performance [8]. Drift has been documented across hippocampus, sensory cortex, and other areas using long-term electrophysiology and imaging [7, 13]. In hippocampal and posterior-parietal recordings, for example, activity patterns reorganize while animals continue to perform the same task reliably [13]. In the primary visual cortex, drift can be structured rather than random: representational change often aligns with high-variance directions in the population code, producing substantial turnover at the single-neuron level while linear readouts remain accurate [7].

Drift is not a single phenomenon with one timescale or one cause. Both spontaneous and experience-dependent forms are observed over days to weeks of recording; they are distinguished by cause, not by whether time has passed [8, 14]. Spontaneous drift continues

even when the external sensory environment is held constant, and has been attributed to background circuit noise or random synaptic turnover [14, 15]. Experience-dependent drift is driven by ongoing stimulus processing and learning [14, 16]. The fundamental reason for drift in the brain remains an active area of research; evidence points to drift as a process that actively assists learning or acts as a byproduct of a highly optimized evolutionary strategy for flexible representations. Computationally, drift may prevent overfitting by forcing the network to continuously explore equivalent parameter configurations within a redundant representational space [10]. Furthermore, recent long-term recordings in the primary visual cortex suggest that representational drift is driven more fundamentally by the accumulation of stimulus experiences rather than the simple passage of physical time [16], hinting at an active, experience-dependent reorganization process.

In continual-learning experiments with artificial networks, we can always quantify representational change: how hidden-layer activations or weights for a fixed reference input set differ from a baseline snapshot as subsequent tasks are learned [4, 17]. That quantity is parallel to, but not identical to, biological drift. We interpret the measured change as representational drift only when performance on those same reference inputs remains stable, matching the biological definition above (see §2.5). Large geometric change that co-occurs with a decreasing reference-task accuracy is forgetting, not drift. To quantify these changes, choosing the correct metric is essential; recent comparative evaluations indicate that geometric alignment measures like linear centered kernel alignment (CKA) and Procrustes distance are capable of capturing functional, behaviourally relevant representational shifts [18]. These metrics allow us to map artificial representation changes onto the stability-plasticity trade-off that defines the continual learning literature.

1.3 Continual learning and the stability-plasticity dilemma

Within machine learning, continual learning (sometimes referred to as lifelong learning) is the problem of learning from datasets that are not fixed in time, while retaining performance on previously learned tasks [2]. This stands in direct contrast to traditional offline machine learning, where networks are trained on a shuffled, stationary dataset. When data arrive sequentially, the independent and identically distributed assumption of offline learning is violated, causing standard optimization algorithms to fail. The core challenge in this field is addressing what is called the stability-plasticity dilemma: a model should be plastic enough to be able to acquire new tasks, and stable at the same time to not override current knowledge.

Three scenarios have been identified in the continual learning setting, first introduced by van de Ven and Tolias [19]. In task-incremental learning (task-IL), the learner receives an explicit task identifier at both training and test time and routes each example to the correct task-specific readout. In domain-incremental learning (domain-IL), the label space is shared across tasks but task identity is not provided at test time: the model must classify into the same set of classes regardless of which input transformation is active. We do not

study class-incremental learning in this project.

We use permuted MNIST as the primary benchmark and report results under both task-IL (multi-head) and domain-IL (single-head); the readout configurations are specified in Methods. XdG still uses a context index to select its hidden-unit mask even with a shared classifier (§2.4.4).

1.4 Three continual learning strategy families

Without specific interventions, sequentially training a neural network using standard optimisation algorithms typically leads to severe catastrophic forgetting. In this baseline scenario, gradients from new tasks directly overwrite the weights that encoded previous tasks, rapidly degrading performance on early objectives. To address this problem, continual learning methods have been developed to protect past knowledge, with most approaches falling into one of three families: regularization-based, memory-based, or architecture-based strategies [2].

Regularization-based methods penalize changes to parameters that are important for previously learned tasks. One such method is Synaptic Intelligence (SI), which estimates parameter importance during training and adds a quadratic penalty to discourage changes to highly weighted synapses [20]. Because SI does not require storing raw data, it is computationally lightweight. This constraint on parameter changes is intended to favour stability, which can come at the expense of plasticity if the network cannot adapt to subsequent tasks.

Memory-based methods preserve performance by interleaving training on examples of past tasks while learning new ones [21]. The simplest approach in this family is experience replay, which stores a subset of old task examples in a buffer to be mixed with current training batches. This method is intended to allow greater representational flexibility, because the parameters can slide along the overlapping low-loss regions of past and present tasks. However, experience replay is computationally demanding and scales poorly as the number of tasks in the stream increases.

Finally, architecture-based approaches physically isolate parameters or dynamically allocate network components to prevent interference. An example of parameter isolation is context-dependent gating (XdG), which activates only a sparse, task-specific subset of units for each task. The intended advantage of this approach is that learning a new task on a sequestered pathway avoids modifying the weights used by other tasks, which could mitigate both forgetting and the loss of plasticity. However, this method is limited by the physical capacity of the network, as the available units are eventually exhausted, often requiring the network to scale up in size and computational cost.

While these strategy families are designed to preserve task performance, they also implicitly shape how hidden representations evolve. By treating artificial neural networks

as experimental model systems, we can dissect how these diverse algorithmic constraints influence representational geometry over time. Comparing these artificial representational changes with empirical neural recordings could then provide insights into which strategies, or combinations of strategies, are being used by biological brains to maintain stable learning.

1.5 Artificial neural networks as a computational window on biological learning

Artificial neural networks serve as a formal language for expressing and testing computational hypotheses under controlled conditions [22]. Biological experiments can establish that drift occurs and constrain its statistics, but they rarely allow systematic manipulation of learning rules, replay budgets, or consolidation strength while holding stimuli constant [6, 12]. In simulation, we can implement continual-learning strategies with similar performance, sample representations at different points, and relate population geometry to forgetting and learnability on the same benchmark [4, 12].

For example, researchers have used deep network models to study how continuous synaptic changes or background noise can produce drift-like dynamics in sensory representations without disrupting downstream readouts [10]. By implementing specific computational constraints in these networks (such as replay buffers or regularized learning rules) we can observe how different stabilization strategies alter representational trajectories over time. Artificial continual-learning experiments can therefore test whether stability-plasticity trade-offs naturally produce drift-like dynamics when plasticity is preserved without catastrophic forgetting [4]. However, a gap remains in our empirical understanding of how different machine learning interventions directly alter internal representation trajectories during sequential learning.

1.6 Gap, objectives, and research questions

Despite extensive research on the algorithmic performance of regularization, replay, and architecture-based continual learning strategies, there is currently a gap in directly and systematically measuring representational change within the explicit context of catastrophic forgetting and loss of plasticity, and in asking when that change constitutes drift. Neuroscience has developed theories around the functional role of drift in biological circuits, and machine learning has designed diverse interventions to mitigate forgetting, yet we lack a unified comparative analysis of how these strategies modulate the stability, rate, and geometry of internal representations when performance is held comparable. It is currently unknown whether different continual learning families produce systematically different profiles of representational change, whether those profiles constitute drift when accuracy is matched, and whether they relate to retention and to the progressive loss of plasticity documented in long sequential training [3]. Whether the parameter constraints of regularisation, the data

reuse of replay, or architectural isolation best supports long-term learnability is a question that motivates the experiments in this project.

To address this gap, this thesis systematically quantifies representational change across CL strategies on a 50-context permuted MNIST dataset, under both task-IL and domain-IL. The first research question asks how these methods differ in the magnitude and temporal trajectory of that change when evaluated with a shared set of metrics (parameter cosine distance, representational cosine distance, linear CKA, and orthogonal Procrustes distance), and under which conditions the change can be interpreted as drift. As a secondary question, we ask whether deliberately inducing representational change through controlled noise or synaptic turnover can mitigate loss of plasticity without increasing forgetting.

1.7 Hypotheses

Based on the distinct computational constraints of each continual learning family, we hypothesise the following:

Our first hypothesis concerns the magnitude of representational drift across algorithmic families. We hypothesize that, at comparable performance, regularization-based methods, which penalize deviations in parameters weighted by estimated importance, will exhibit smaller representational drift over sequential tasks than replay-based methods, whose parameters can move within overlapping low-loss regions defined by stored and current data.

Our second hypothesis concerns the relationship between drift and learning plasticity. We hypothesize that strategies associated with higher, structured representational drift will show better preservation of the ability to learn new tasks over long streams, whereas methods that strongly restrict parameter movement will show more loss of plasticity on later tasks [3, 4]. This prediction may be difficult to observe at a short task horizon.

As an exploratory third hypothesis, we hypothesize that deliberately inducing representational change through controlled noise or synaptic turnover can mitigate loss of plasticity without increasing forgetting.

2 Methods

2.1 Experimental overview

Our experimental framework builds on the continual learning structures established by van de Ven and Tolias [19]. At its core, we use a fully connected artificial neural network as a simplified model system to study how internal representations change during sequential learning. Handwritten digits from MNIST are presented under fifty different pixel permutations. Each permutation defines a distinct context. Contexts are visited sequentially with hard boundaries, and previous training data are not revisited unless a memory mechanism (experience replay) is enabled.

We first checked whether the loss-of-plasticity signature of Dohare et al. [3] appears in this experimental setup, using a long 800-context naive-Adam run (Appendix A.1). The investigation then follows a four-stage procedure that can be reused for other continual-learning comparisons (Figure 1). Capacity calibration checks that the architecture can solve all contexts when they are available simultaneously (§2.3.1). Hyperparameter tuning locks optimiser settings and method-specific hyperparameters so that strategies reach comparable final accuracy rather than maximal performance of any single method (§2.3.2, §2.4.6). The strategy comparison then runs the 50-context evaluation at these locked settings, logging accuracy, representational change, and loss-of-plasticity diagnostics at every context boundary (§2.2–§2.6). The induced-change experiment applies controlled noise and turnover on selected configurations to test whether extra internal variability can restore plasticity without increasing forgetting (§2.7).

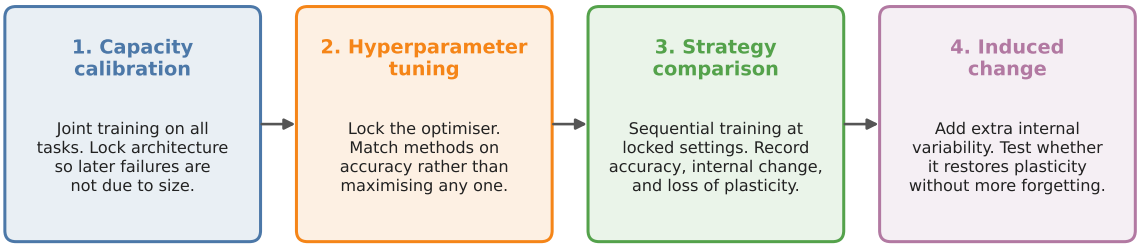


Figure 1: Four-stage experimental procedure. (1) Capacity calibration uses joint training to lock architecture size once the model can solve all tasks together (§2.3.1). (2) Hyperparameter tuning locks the optimiser and method strengths for matched accuracy rather than maximal performance of any single method (§2.3.2, §2.4.6). (3) Strategy comparison evaluates sequential training on accuracy, representational change, and loss of plasticity (§2.2–§2.6). (4) Induced change tests whether extra internal variability can restore plasticity without increasing forgetting (§2.7).

Throughout, we cross two factors that shape interference: the optimiser (Adam versus stochastic gradient descent, SGD) and the readout architecture (single-head versus multi-

head). A single-head network uses one shared ten-unit classifier for all contexts. For naive fine-tuning, SI, and ER, task identity is not provided at training or test time (domain-IL). A multi-head network allocates a separate ten-unit classifier per context (500 output units for 50 contexts). Task identity is provided implicitly through head routing, and only the active head is trained or evaluated (task-IL). XdG is the exception: even with a shared classifier it still receives a context index in order to select the hidden-unit mask (§2.4.4). These four combinations of optimiser and architecture (Adam or SGD with a single-head or multi-head classifier) are tuned and reported separately because the same regularisation strength can behave differently under different gradient noise profiles and output routing schemes. Locked settings shared across combinations are listed in Table 1. Unless noted, reported means are accompanied by 95% confidence intervals computed as $1.96 \times \text{s.d.}/\sqrt{n}$ over seeds.

2.2 Benchmark and continual learning protocol

The primary benchmark is permuted MNIST. Images are zero-padded from 28×28 to 32×32 pixels and flattened to 1,024 input units. Within each context, a fixed random permutation of pixel indices is applied. Permutations are redrawn for each random seed but remain stable within a run. All ten digit classes are present in every context. Contexts are presented sequentially: the network trains for a fixed budget of 5,000 optimiser steps on one permutation, is evaluated, and then advances to the next.

We use the standard MNIST train and test splits for each permuted dataset. After each context, test accuracy is computed on 1,024 randomly sampled test images. Training uses mini-batches of 64 images and 5,000 optimiser steps per context unless stated otherwise. In the single-head (domain-IL) configuration, all ten output units are active and labels remain in $\{0, \ldots, 9\}$. No task index is supplied to naive fine-tuning, SI, or ER. In the multi-head (task-IL) configuration, gradients and evaluation are restricted to the ten output units belonging to the current context. XdG still uses the context index under both readouts (§2.4.4).

To track internal change over time, we compare network states at fixed reference contexts in the stream. Context 1 is the primary online reference during training. Post-hoc analyses additionally use contexts 2, 10, and 25 to capture early- and mid-sequence dynamics.

2.3 Model architecture and training

2.3.1 Joint-training capacity calibration

Before sequential experiments, we checked that the architecture can solve all contexts when they are available simultaneously. On a single local machine (to keep hardware conditions fixed), we swept hidden width $\{100, 400, 1000, 2000\}$, depth of one to three hidden layers, and joint budgets of 50k and 100k iterations, for both Adam and SGD (48 completed runs, Appendix A.2). Sequential training later uses 5,000 steps on each of 50 contexts (250,000 steps in total). The joint grid is therefore a cheaper capacity check, not a matched-budget upper

bound: we did not run joint training at 250k iterations. A two-hidden-layer network with 1,000 ReLU units per layer provided the best accuracy/compute trade-off on that 50k–100k grid. The locked architecture is a fully connected MLP with layer sizes 1,024-1,000-1,000-C, where $C = 10$ for single-head and $C = 500$ for multi-head. Weights are initialised with Xavier normal draws. Dropout and weight decay are disabled in the baseline experiments.

2.3.2 Optimiser calibration

Optimiser hyperparameters were locked in two stages. Single-task search over one permuted context swept learning rates for Adam and SGD at batch sizes $\{16, 64, 128, 256\}$ with other settings held fixed, monitoring test accuracy at ten log-spaced checkpoints up to 10,000 iterations. Multitask refinement across ten contexts (5 seeds, batch size 64, 5,000 steps per context) narrowed learning rates around the single-task optima. Configurations were ranked by mean accuracy over contexts seen so far and by retention of the first context. This yielded Adam at 4.5×10^{-4} ($\beta_1 = 0.9$, $\beta_2 = 0.999$, $\varepsilon = 10^{-8}$) and SGD rates of 0.01 (single-head) and 0.02 (multi-head) with zero momentum for the strategy comparison (Table 1; SGD rate selection is documented in Appendix A.3).

2.3.3 Training objective

All methods optimise a supervised classification objective on the current context. For multi-class permuted MNIST, the primary loss is cross-entropy. Naive fine-tuning minimises this loss alone. When experience replay is active, replayed examples contribute an additional cross-entropy term. The combined loss is

$$\mathcal{L} = \frac{1}{t}\,\mathcal{L}_{\text{cur}} + \left(1 - \frac{1}{t}\right)\mathcal{L}_{\text{replay}},$$

where t is the current context index, so replay weight increases as more tasks accumulate [21]. When synaptic intelligence is active from context 2 onward, a quadratic regulariser

$$\mathcal{L}_{\text{SI}} = c \sum_i \omega_i \left(\theta_i - \theta_i^*\right)^2$$

is added to the total loss, where θ_i^* are parameter values consolidated after previous contexts, ω_i are online importance estimates, and c is the regularisation strength [20]. No replay, SI, or gating terms are applied in the naive baseline.

2.4 Continual learning methods implemented

We compared naive fine-tuning against three protective strategy families: synaptic intelligence (SI) [20], experience replay (ER) [21], and context-dependent gating paired with SI (XdG+SI) [23]. XdG alone is not part of the matched multi-seed comparison: a single-seed diagnostic

Setting	locked value
Benchmark	permuted MNIST, 50 contexts, sequential presentation
Architecture	MLP 1,024-1,000-1,000-C, ReLU, Xavier init, no dropout/weight decay
Steps per context	5,000
Batch size	64
Adam learning rate	4.5×10^{-4}
SGD learning rate	0.01 (single-head), 0.02 (multi-head), momentum 0
Evaluation samples	1,024 test images per context
Representation samples (change metrics)	32 test images per snapshot
LoP diagnostic samples	256 test images

Table 1: Locked experimental settings for the 50-context strategy comparison. Method-specific hyperparameters (SI, ER, XdG+SI) were tuned separately for each combination of optimiser and architecture (Tables 2 and 3).

run was retained only to show that gating without synaptic consolidation cannot keep up with SI or ER on this 50-context stream, motivating the hybrid following the original paired design of Masse, Grant, and Freedman [23]. Each method is described below with its tunable parameters. Hyperparameter selection is detailed in §2.4.6.

2.4.1 Naive fine-tuning

The baseline sequentially minimises cross-entropy on the current context only. No replay buffer, synaptic regulariser, or context mask is applied. Runs use the locked Adam or SGD learning rates from Table 1. This condition provides the reference for catastrophic forgetting, parameter change, and loss of plasticity over 50 contexts.

2.4.2 Synaptic intelligence

SI tracks an online importance weight ω_i for each parameter during training on a context. Following Zenke, Poole, and Ganguli [20], ω_i is a path integral of the loss gradient along the parameter trajectory, accumulated as $\omega_i \leftarrow \omega_i + W_i/((\Delta\theta_i)^2 + \varepsilon)$ after each context, where W_i estimates that parameter's contribution to the loss decrease and ε is a small damper (here $\varepsilon = 0.1$). After each context, consolidated values θ_i^* and updated ω_i are stored. From context 2 onward, the SI penalty above is added to the loss with regularisation strength c as the primary hyperparameter. Larger c penalises movement in parameters deemed important for past tasks. We swept $c \in \{0.1, 0.2115, 0.4472, 0.9457, 2.0\}$ (Table 2).

2.4.3 Experience replay

ER maintains a memory buffer of past training examples using random per-class storage [21]. After each context, B examples per class from that context are stored and kept, so the buffer grows with the stream ($10B$ examples per completed context). During training on context t, mini-batches of size 64 are drawn from the current task. Replay samples are incorporated through the weighted loss above. In task-IL, each previous context is a separate replay slot: an additional mini-batch of size $\lceil 64/t \rceil$ is drawn per slot, capped by available stored data. In domain-IL, stored examples from all previous contexts are concatenated into a single replay stream with mini-batch size 64. The current batch size is therefore not reduced. We swept $B \in \{10, 20, 50, 100, 150, 500, 1000\}$ examples per class (Table 2).

2.4.4 Context-dependent gating

XdG assigns each context a fixed random binary mask over hidden units in every fully connected hidden layer [23]. The hyperparameter $g \in (0, 1)$ specifies the fraction of units masked off (set to zero excitability) per layer. The remaining $(1 - g)$ fraction stays active for that context. We swept $g \in \{0.2, 0.4, 0.6, 0.8, 0.9\}$ during tuning to select a gating fraction for the hybrid below. XdG-alone runs are diagnostic only (seed 1): they document that gating without consolidation underperforms SI/ER and are not seed-aggregated or treated as a matched comparison condition. For SI, ER, and naive fine-tuning, no gating is used. For XdG and XdG+SI, the context-specific mask for the evaluated task is applied during training, test-accuracy measurement, and representation snapshots. The network is not evaluated with all units unmasked. Because different contexts activate different unit subsets, representational change measured over the full hidden population can remain substantial even when task accuracy is stable.

Selecting the mask requires knowing which context is active. Under the multi-head readout that context index is already used to route the classifier, so gating does not add a new source of task information. Under the single-head readout the classifier is shared and naive fine-tuning, SI, and ER receive no task index, but XdG and XdG+SI still receive the context index in order to apply the correct hidden-unit mask. Single-head XdG is therefore not a strict domain-IL method in the sense of van de Ven and Tolias [19]: the output layer does not use task identity, but the hidden layers do. We keep this oracle gating so that architectural isolation can be compared with a shared readout; we do not infer the mask from the input.

2.4.5 XdG + SI hybrid

Following Masse, Grant, and Freedman [23], we combined XdG with a lighter SI penalty as the architecture-based method in the strategy comparison. The hybrid uses the same context-conditioned masks as XdG alone, including under the single-head readout, so it likewise receives a context index that SI and ER do not. For each combination of optimiser and architecture, we first selected a gating fraction g from the XdG-alone diagnostic sweep,

then swept SI strengths $c \in \{0.01, 0.05, 0.1, 0.2115\}$ at that g (Adam combinations used $g = 0.9$), choosing the hybrid that best matched the SI/ER accuracy profile.

2.4.6 Hyperparameter tuning procedure

Hyperparameters were tuned in phases (SI, then ER, then XdG, then XdG+SI) on the full 50-context stream, separately for each combination of optimiser and architecture, using seed 1. Selected settings were then evaluated with 15 seeds. The goal was performance matching, not declaring a single best method. The seed-1 accuracy curves used for selection are in Appendix A.4.

For SI, we selected the c value that maximised final mean test accuracy averaged over all 50 contexts. For ER, buffer budgets were handpicked from the sweep so that final mean accuracy and reference-task retention were comparable to the chosen SI run (Table 3). For the XdG-alone diagnostic we selected g by final mean accuracy solely to fix the gating fraction for the hybrid. XdG+SI then used that g (or $g = 0.9$ for Adam) with a handpicked lighter SI strength. Locked comparison settings are reported in Table 3 (XdG alone is omitted there because it is not a comparison condition).

Method	Hyperparameter	Sweep values
SI	regularisation strength c	0.1, 0.2115, 0.4472, 0.9457, 2.0
ER	buffer budget per class B	10, 20, 50, 100, 150, 500, 1000
XdG	masked fraction g	0.2, 0.4, 0.6, 0.8, 0.9
XdG+SI	gating g (from XdG), SI strength c	0.01, 0.05, 0.1, 0.2115

Table 2: Hyperparameter grids for continual learning methods. SI was selected by highest final mean accuracy. The XdG g grid was used only to pick a gating fraction for XdG+SI. ER budgets and XdG+SI strengths were handpicked for performance matching to SI.

Combination	SI: c	ER: B (per class)	XdG+SI: g, c
single-head / Adam	0.2115	20	0.9, 0.01
single-head / SGD	0.9457	10	0.2, 0.05
multi-head / Adam	0.4472	100	0.9, 0.05
multi-head / SGD	2.0	150	0.6, 0.05

Table 3: Handpicked hyperparameters for the 50-context strategy comparison (from phased tuning). XdG+SI uses the listed gating fraction g with the listed SI strength c.

2.5 Representational change and drift measurement

We distinguish representational change (any difference in internal activations or weights between two training states) from representational drift in the biological sense: change in

population responses to fixed inputs while task performance on those inputs remains stable [4, 8]. In our continual-learning setting, drift analyses focus on reference-task inputs whose labels and permutations are held fixed. Large activation changes that co-occur with collapsing reference-task accuracy reflect forgetting rather than drift in this narrower sense. Cross-method comparisons emphasise conditions with matched final accuracy so that differences in representational metrics are not trivially explained by performance gaps.

At the end of each context, we snapshot (i) the flattened trainable weight vector and (ii) hidden-layer activations evoked by 32 fixed test images from each context. When the network completes reference context r, that snapshot becomes the baseline. For all later contexts $t \geq r$, we pass the same reference-task test images through the current network and quantify change relative to the baseline. For XdG runs, the reference context's mask is applied during these forward passes, matching training and evaluation conditions. Hidden activations are taken from the feature extractor (both fully connected hidden layers, flattened into a single population vector per image), yielding an activation matrix $H \in \mathbb{R}^{n \times d}$ with $n = 32$ stimuli and $d = 2000$ hidden units.

2.5.1 The importance of metric choice

Representational drift in neuroscience is defined at the population level. Individual units can change tuning while a downstream readout or behaviour remains stable [7, 8]. Continual learning analyses often track only parameter movement or scalar accuracy. Those quantities are useful (regularisation methods such as SI explicitly penalise weight change), but they do not specify whether the geometry of stimulus representations has reorganised. Two networks can differ substantially in weights yet implement nearly the same input–output mapping, or retain accuracy while redistributing activity across hidden units in ways that a mean activation summary would miss. Recent comparative work in NeuroAI therefore treats representational similarity as a measurement problem with explicit invariances [18, 24, 25]. A metric that is invariant to all invertible linear transforms can, in high dimensions, report spurious similarity between unrelated representations [24]. We therefore need measures that operate on the same fixed reference inputs, capture population-level structure relevant to classification, and can be computed identically for every continual-learning method.

No single statistic captures every aspect of this change. Parameter-level distance reports how far optimisation has moved the model in weight space. Activation-level cosines report coarse shifts in average hidden activity. Geometry-based metrics ask whether the relational structure among stimuli is preserved. These levels can diverge. SI can suppress both weight change and measured geometry, whereas ER can stabilise accuracy while allowing substantial remapping at matched performance (see §3.3). We therefore report four metrics jointly. Agreement between CKA and Procrustes indicates robust geometric change. Disagreement with parameter cosine suggests reorganisation that is not reducible to global weight displacement. Stability of representational cosine alongside rising CKA would indicate redistribution across units with little change in the population mean.

2.5.2 Parameter cosine distance

The parameter cosine distance is

$$d_{\text{param}} = 1 - \frac{\theta_t^\top \theta_r}{\|\theta_t\| \, \|\theta_r\|},$$

where θ_r and θ_t are the flattened vectors of all trainable weights at the reference snapshot and at context t. This measure treats the network as a single point in parameter space and is sensitive to any global reorientation of weights, regardless of whether hidden geometry or readout performance changes [17]. It is a natural companion to regularisation-based continual learning, whose objective explicitly discourages movement in θ, and it provides an interpretable upper bound on how much the model has been overwritten. Its limitation is that parameters are only an indirect proxy for representational geometry: large d_{param} can co-occur with stable stimulus–stimulus relationships in hidden space, and small d_{param} does not guarantee that a fixed linear readout would still decode earlier tasks without retraining.

2.5.3 Representational cosine distance

The representational cosine distance compresses the population code to the mean activation vector across the reference test set,

$$d_{\cos} = 1 - \frac{\bar{h}_t^\top \bar{h}_r}{\|\bar{h}_t\| \, \|\bar{h}_r\|}, \qquad \bar{h} = \frac{1}{n} \sum_{i=1}^{n} h_i,$$

where $h_i \in \mathbb{R}^d$ is the hidden representation of the ith reference image. This metric is easy to interpret (it tracks whether the network's average hidden state for the reference task has moved) and serves as a deliberately simple baseline against which geometry-sensitive measures can be judged. Its weakness mirrors a common pitfall in neural data analysis: drift can rearrange tuning across units while leaving average activity approximately unchanged, especially when representations lie in a high-dimensional subspace with redundant degrees of freedom [10, 11]. We include $d_{\cos}$ to expose such cases. We do not rely on it alone to define the magnitude of representational change.

2.5.4 Linear CKA change

Linear centred kernel alignment (CKA) compares two representation matrices $H_r, H_t \in \mathbb{R}^{n \times d}$ with rows aligned to the same stimuli [24]. Each column (hidden unit) is centred across samples, and similarity is computed from the Frobenius inner products of the resulting Gram matrices:

$$\text{CKA}(H_r, H_t) = \frac{\|H_r^\top H_t\|_F^2}{\|H_r^\top H_r\|_F \, \|H_t^\top H_t\|_F},$$

with values in $[0, 1]$. We report CKA change as $d_{\text{CKA}} = 1 - \text{CKA}(H_r, H_t)$. Linear CKA is invariant to orthogonal transformations of feature space (rotations and reflections of hidden

activations that preserve inner products), but, unlike correlation-based measures that are invariant to any invertible linear map, it remains sensitive to meaningful geometric change when the number of features exceeds the number of samples [24]. Intuitively, CKA asks whether pairs of reference stimuli stand in the same relative positions to one another in the two representation spaces, which is closer to what a fixed linear classifier exploits than a single mean vector. Comparative evaluations find that CKA aligns more closely with behavioural similarity across models than linear predictivity or other common baselines [18].

2.5.5 Orthogonal Procrustes distance

Orthogonal Procrustes analysis explicitly finds the rotation in feature space that best aligns two representation matrices before measuring residual disparity [25]. After column-centring H_r and H_t, each matrix is Frobenius-normalised and the orthogonal map $R^\star = \arg\min_{R^\top R=I} \|H_r R - H_t\|_F$ is obtained by singular value decomposition of $H_r^\top H_t$. The Procrustes distance is the normalised residual

$$d_{\text{Proc}} = \frac{\|H_r R^\star - H_t\|_F}{\sqrt{n}}.$$

Lower values indicate that the two representations differ only by an orthogonal reparameterisation. Higher values indicate geometric mismatch that cannot be removed by rotating feature axes. Like CKA, Procrustes emphasises global shape over individual unit identity, analogous to biological drift in which neuron-level turnover coexists with stable population structure [7]. Williams et al. show that Procrustes belongs to a family of proper shape metrics (satisfying symmetry and the triangle inequality) suitable for comparing representational geometry across networks [25]. Close rank agreement between CKA and Procrustes would increase confidence that observed geometric effects are not artefacts of a single similarity index; that check is defined below and reported in §3.6.

2.5.6 Primary metrics and post-hoc analyses

We treat CKA and Procrustes as the primary metrics for functionally relevant representational change, because they capture stimulus–stimulus geometry and have been found to track behavioural correspondence more reliably than mean-activation or weight summaries alone [18]. Parameter and representational cosine distances are retained as interpretable references that anchor the geometry metrics to optimisation-level and mean-activation-level change, respectively. Post-hoc analyses recompute all four metrics for reference contexts $\{1, 2, 10, 25\}$ from stored snapshots without altering the training trajectory. Final values of those series are not comparable across references, because earlier contexts have a longer subsequent stream. We therefore also read each metric after a matched window of 15 subsequent contexts (context 1 evaluated at context 16; context 25 at context 40), which asks whether representations of earlier tasks change more than those of later tasks given the same amount of further training.

2.5.7 Metric agreement and rank correlations

The four change metrics live on different numerical scales, so we compare them by rank rather than by raw magnitude. Spearman ρ is the Pearson correlation of the ranked values, with tied observations assigned average ranks. Values near +1 mean that two quantities order runs or strategies in the same way; values near 0 mean that the orderings are unrelated.

Metric agreement is computed on strategy rankings, not on individual seeds. For each of the four combinations of optimiser and architecture, we take the four strategy means of final CKA change, Procrustes distance, and parameter cosine distance, compute Spearman ρ between each pair of those rankings, and average ρ across combinations. Representational cosine is retained as a mean-activity baseline and is not included in this trio. High agreement between CKA and Procrustes indicates that the geometric change is robust to the choice of similarity index. Lower agreement of either geometry metric with parameter cosine indicates reorganisation that is not reducible to global weight displacement.

To relate representational change to performance, we compute Spearman ρ between per-run final CKA change and reference-task forgetting separately in each optimiser–readout combination, using only SI, ER, and XdG+SI ($n = 45$ per combination). Naive fine-tuning is excluded because it is not accuracy-matched: pooling it with the locked methods confounds overwriting with drift. We do not residualise on mean accuracy, and we do not report a single coefficient pooled across combinations.

2.6 Performance and plasticity metrics

After each context, we log per-context test accuracy and the running mean accuracy over all contexts seen so far [2, 19]. Forgetting on reference context r is the drop from peak test accuracy on r to accuracy on r at the end of the 50-context stream. Retention is the final accuracy on r.

Loss of plasticity (LoP) is distinct from forgetting: a network can retain high accuracy on early tasks yet learn new tasks increasingly poorly [3]. Our primary LoP measure in the strategy comparison is current-task accuracy over time: whether accuracy on newly learned contexts at the end of their training window declines as the stream lengthens. We report this as late learnability, the mean current-task accuracy on contexts 41–50, and as LoP drop, early minus late current-task accuracy, with early taken as the mean on contexts 1–5. A positive LoP drop means later tasks were learned more poorly than early ones. Following Dohare et al. [3], we additionally log structural diagnostics at each context boundary: the fraction of hidden units with exactly zero activation on 256 test samples (dead-unit fraction), the effective rank (participation ratio) of the hidden activation matrix, and mean absolute synaptic weight magnitude. Increasing dead-unit fraction and decreasing effective rank indicate reduced representational flexibility. Those four quantities (current-task accuracy, dead-unit fraction, effective rank, and weight magnitude) are the same diagnostics we first

logged on the 800-context naive-Adam runs in Appendix A.1.

2.7 Induced representational-change interventions

The induced-change experiment tests whether deliberately increasing internal variability can mitigate LoP without accelerating forgetting. Noise is compared within each strategy, against that strategy's own no-noise baseline, so SI-matched accuracy is not required. We therefore give ER a stronger replay setting ($B = 100$ examples per class) rather than the SI-matched $B = 20$ of Table 3. We add four perturbation families during training on the Adam single-head stream, using naive fine-tuning, SI ($c = 0.2115$), this ER setting, and XdG+SI ($g = 0.9$, $c = 0.05$, not the locked $c = 0.01$ of Table 3), motivated by computational studies of representational change [7, 26, 27].

Uniform and targeted dropout act on hidden-unit activations, not on pixels. Dropout is applied to the input of each fully connected map, so placing it on the first map (1,024 $\rightarrow$ 1,000) would randomly zero input pixels. That first map is therefore left un-noised. The second hidden layer then drops first-hidden-unit activations, and the classifier drops second-hidden-unit activations, each with probability p during training [7]. Unlike Aitken et al. [7], who applied dropout only to the penultimate layer of a CNN because change was measured there, both hidden representations are noised here because the change metrics concatenate them. Targeted (variance-dependent) dropout uses the same placement and refreshes per-unit drop probabilities $P_\mu = \text{clip}(A \cdot v_\mu, 0, 1)$ every 500 steps from running activation variance v_μ.

Gradient noise injects additive Gaussian noise with standard deviation σ into gradients of all fully connected weight matrices, including the first hidden layer and the classifier [26]. Synaptic turnover likewise acts on all fully connected weights: it prunes and regrows a fraction f of active connections every 100 steps based on an exponential moving average of connection activity ($\beta = 0.99$) [3, 27].

Each family is evaluated at ten strengths (Table 4), together with a shared no-noise baseline, giving 41 conditions per strategy. All 164 strategy-and-condition combinations are run with 15 seeds (2,460 runs). Because ER and XdG+SI here are not the SI-matched locked settings of Table 3, noise results should not be read as a replicate of the strategy comparison. Results are reported in §3.7.

Family	Parameter	Sweep values
Uniform dropout	p	0.05, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9
Targeted dropout	A	1, 2, 5, 10, 20, 40, 80, 160, 320, 640
Gradient noise	σ	0.001, 0.003, 0.01, 0.03, 0.1, 0.3, 1, 3, 10, 30
Synaptic turnover	f	0.001, 0.002, 0.005, 0.01, 0.02, 0.05, 0.1, 0.2, 0.35, 0.5

Table 4: Induced-change hyperparameter grid. Each family is evaluated at ten strengths plus a shared no-noise baseline (41 conditions per strategy).

3 Results

The strategy comparison evaluates naive fine-tuning, SI, ER, and XdG+SI across 50 permuted-MNIST contexts under all four combinations of optimiser and architecture (Table 3), with 15 independent seeds for each strategy. We lead with Adam single-head (domain-IL). The SGD single-head summary is in Table 6. Task-IL (multi-head) summaries are in Appendix A.5. Unless noted, representational change metrics refer to reference context 1.

3.1 Capacity calibration and hyperparameter tuning

Joint training on all fifty permutations confirmed that the MLP architecture (1,024-1,000-1,000-C) has sufficient capacity when data from all contexts are available simultaneously (Appendix A.2, Figure 14). Single-task and multitask learning-rate searches locked Adam's learning rate at 4.5×10^{-4} and SGD's learning rate at 0.01 (single-head) and 0.02 (multi-head) with zero momentum (Table 1; Appendix A.3).

Method-specific hyperparameters were chosen per combination (Table 3). Sweeps of SI, ER, XdG, and XdG+SI, and the selection rules, are in Appendix A.4. SI was selected by highest final mean accuracy. ER budgets and XdG+SI strengths were handpicked for performance matching to SI. ER budgets are not equated across combinations: domain-IL requires smaller buffers ($B = 10$–20) than task-IL ($B = 100$–150) to reach comparable retention to SI.

3.2 Continual learning performance and forgetting

Table 5 reports the mean and 95% confidence intervals for 15 seeds in the primary Adam single-head configuration. Naive fine-tuning serves as the lower bound (mean accuracy 0.35, forgetting 0.87). SI and ER both raise mean accuracy to ≈0.87–0.88. ER achieves the lowest forgetting (0.15 vs. 0.26 for SI). XdG+SI matches mean accuracy (0.88) but retains reference-task performance poorly (forgetting 0.43), illustrating the difficulty of architectural isolation with a shared readout.

Table 5: Matched comparison summary (Adam, single-head): mean ± 95% CI. Forgetting is peak accuracy−final accuracy on reference context 1.

Strategy	Mean acc	Forgetting	CKA	Procrustes	Param
naive	0.347±0.006	0.874±0.012	0.564±0.042	0.143±0.005	0.750±0.001
SI	0.871±0.004	0.258±0.026	0.135±0.011	0.062±0.002	0.423±0.001
ER	0.878±0.002	0.150±0.009	0.321±0.035	0.099±0.004	0.745±0.001
XdG+SI	0.884±0.004	0.428±0.029	0.055±0.005	0.045±0.002	0.716±0.001

XdG alone cannot match SI/ER on this dataset (Appendix A.4) and is therefore excluded from the matched performance comparison tables. Figure 2 compares the four strategies at locked settings. SI and ER preserve reference-task accuracy throughout the stream. Naive

Table 6: Matched comparison summary (SGD, single-head): mean ± 95% CI. Forgetting is peak accuracy−final accuracy on reference context 1.

Strategy	Mean acc	Forgetting	CKA	Procrustes	Param
naive	0.689±0.008	0.482±0.030	0.256±0.022	0.099±0.003	0.263±0.000
SI	0.797±0.006	0.286±0.040	0.187±0.008	0.081±0.002	0.211±0.000
ER	0.825±0.001	0.112±0.008	0.151±0.011	0.073±0.003	0.116±0.000
XdG+SI	0.797±0.007	0.368±0.025	0.182±0.008	0.083±0.002	0.253±0.000

fine-tuning collapses within the first ten contexts. XdG+SI improves over naive but remains below SI/ER on the reference task.

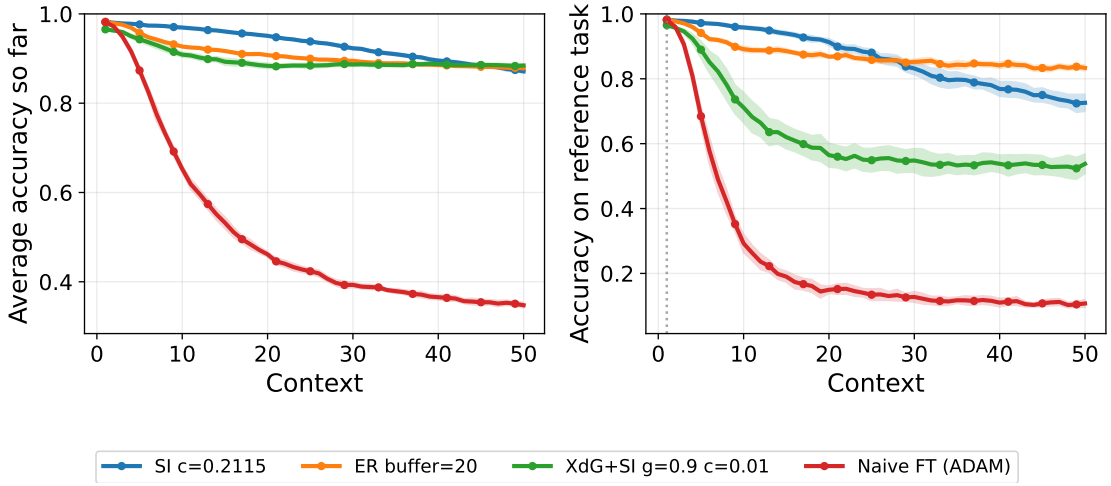


Figure 2: Final method comparison at locked hyperparameters (Adam, single-head, 15 seeds, mean ± 95% CI): naive fine-tuning, SI, ER, and XdG+SI. Left: mean accuracy over contexts seen so far. Right: reference-task (context 1) accuracy.

Under task-IL (multi-head), all matched strategies reach high mean and reference-task accuracy (Appendix A.5, Tables 12–13). Under SGD single-head, naive fine-tuning is less catastrophic than under Adam (mean accuracy 0.69 vs. 0.35; Table 6) but SI/ER ordering persists. Performance, CKA/Procrustes, and cosine trajectories for these other combinations are in Appendix A.6. Domain-IL remains the harder experimental setting and is the focus of subsequent analyses.

3.3 Representational change by continual learning strategy

With performance approximately matched, we ask how hidden representations of a fixed reference input set change. Table 5 reports final CKA change, Procrustes distance, and parameter cosine distance for Adam single-head.

Three profiles emerge. Among strategies that retain the reference task well, SI has the lowest CKA change (0.135) and parameter distance (0.42). ER: CKA change about twice that of SI (0.321) with parameter distance near naive (0.75), meaning behaviour is stabilised without freezing the representation. This is the setting in which the extra geometric change can be read as drift. XdG+SI has still lower CKA in this table (0.055) and the lowest CKA in multi-head Adam (0.025; Table 12), but high forgetting in domain-IL (0.43). This means there is a difference between how the network changes and how the representations change due to the subnetwork masking. CKA and Procrustes order these strategies in the same way (§3.6).

The SI–ER contrast is consistent under Adam: ER exceeds SI in CKA change by $\Delta 0.19$ (single-head) and $\Delta 0.18$ (multi-head) while reducing forgetting in domain-IL (Δforgetting $= -0.11$ for single-head). Under SGD single-head, SI forgets more than ER (0.29 vs. 0.11) and shows correspondingly larger CKA change (Table 6). The reason for the difference is that under SGD the extra geometric change tracks extra forgetting, so what we measure is overwriting of earlier task representations rather than drift.

Figure 3 shows CKA and Procrustes trajectories for the matched Adam single-head strategies. ER permits remapping while retaining the reference task. Naive fine-tuning shows the largest geometric change alongside catastrophic forgetting, which we do not interpret as drift.

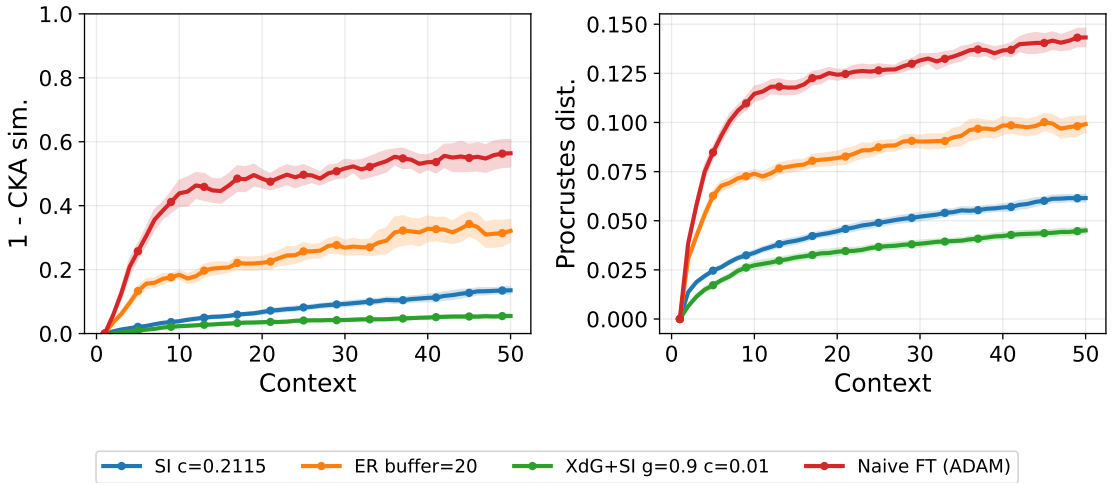


Figure 3: Representational-change trajectories (Adam, single-head, 15 seeds, mean ± 95% CI, reference context 1). Left: linear CKA change ($1 - \text{CKA}$). Right: orthogonal Procrustes distance. ER remaps while retaining the reference task; naive fine-tuning's large change co-occurs with forgetting.

Those measures are all relative to context 1, so they cannot tell us whether earlier-task representations change more than later-task ones: context 1 is observed for 49 subsequent

tasks, context 25 for only 25. We therefore re-evaluate CKA after a matched window of 15 subsequent contexts for each stored reference (Figure 4; numeric values in Appendix A.7). Under Adam single-head, naive fine-tuning does overwrite earlier task representations more: CKA after 15 tasks is 0.46 for context 1 versus 0.32 for context 25 ($\Delta = 0.15$). ER shows a smaller difference ($\Delta = 0.04$) whose 95% CI includes zero. SI reverses the ordering slightly (0.055 vs. 0.091). XdG+SI is essentially flat. The same contrast is strongest under Adam task-IL naive fine-tuning ($\Delta = 0.23$). Under SGD, naive fine-tuning does not show that drop: CKA after 15 tasks is similar at contexts 1 and 25 and highest for mid-sequence references. SI and ER, by contrast, both change earlier references more than later ones. Earlier-task representations therefore do not universally change more. The pattern depends on optimiser and strategy, and it is largest where interference is unconstrained (Adam, naive).

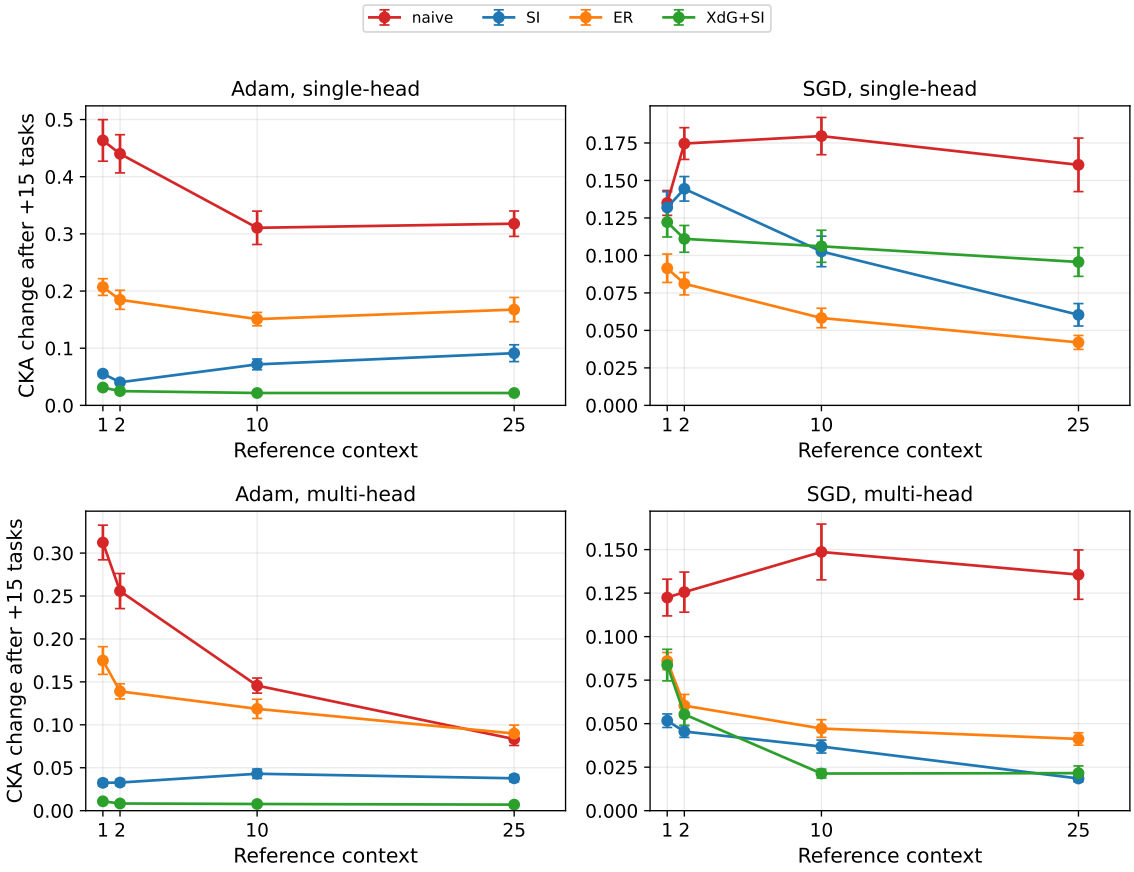


Figure 4: CKA change after 15 subsequent contexts for each stored reference (mean ± 95% CI for 15 seeds). Context 1 is evaluated at context 16 and context 25 at context 40, so remaining stream length is matched. A decreasing trace means earlier-task representations change more than later-task ones given the same amount of further training.

The same runs can also be read at two other levels that biology does not give us jointly: the flattened weight vector and the mean hidden activation on the reference images (Figure 5). Those measurements disagree. ER ends with parameter cosine distance 0.75, nearly the same as naive fine-tuning (0.75), yet representational cosine distance is 0.62 versus 0.90 and CKA change is 0.32 versus 0.56. XdG+SI is the reverse: parameter distance remains high (0.72) while mean-activation change on the gated reference task is the lowest of the matched strategies (0.09), to be expected due to the nature of the strategy. Only SI keeps both weights and representations relatively still (0.42 and 0.16). Weight movement is therefore not a proxy for representational change, and representational change is not a proxy for weight movement.

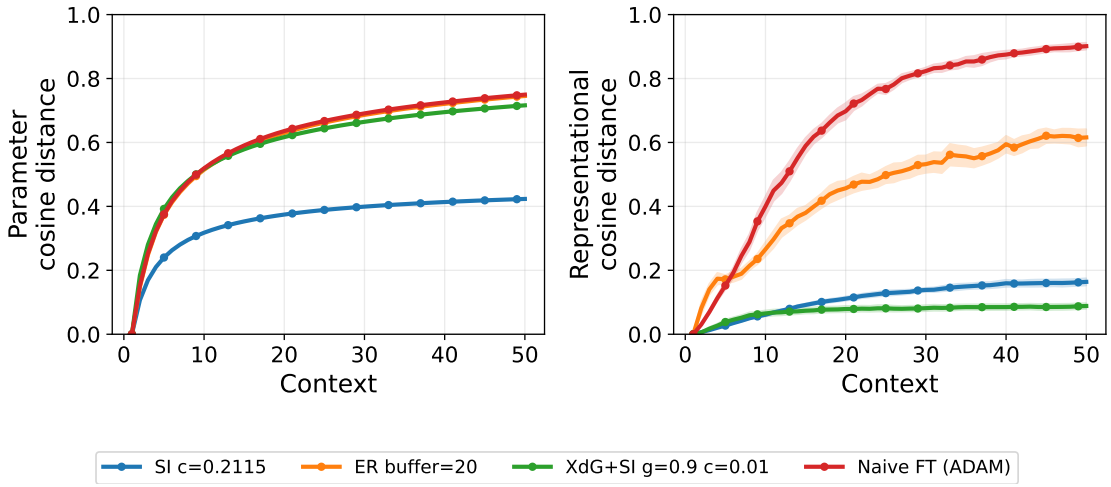


Figure 5: Parameter versus representational cosine distance on the same Adam single-head runs (mean ± 95% CI for 15 seeds, reference context 1). Left: change in the flattened weight vector. Right: change in the mean hidden activation for reference-task images. Biology typically yields only the right-hand measurement; the MLP stores both, so a stability mechanism can be localised to synapses, to the code, or to both. ER and naive fine-tuning overwrite weights to a similar extent; their hidden activity does not.

That split is a reason to treat the network as a model of biological learning rather than only as an engineering baseline. Cortical recordings give population activity, not the complete synaptic matrix of the same cells [6, 8]. In the MLP both are stored on every snapshot, so we can ask whether a stability mechanism acts on synapses, on the code, or on both, a comparison that biological data cannot make. Figure 5 is the clearest illustration: replay can look like naive fine-tuning in weight space while looking much closer to a consolidated code in activity space.

3.4 Coupling change to learnability and forgetting

Among the accuracy-matched methods, CKA change and forgetting are not a single positive association. Table 7 reports Spearman ρ within each optimiser–readout combination for SI, ER, and XdG+SI only ($n = 45$). Under Adam single-head, the primary matched comparison, $\rho = -0.83$: ER remaps more and forgets less, XdG+SI is geometrically still and forgets more on the shared readout, and SI sits in between. SGD single-head has the opposite sign ($\rho = +0.54$), with ER both more stable geometrically and more retentive. The two multi-head combinations are weaker and disagree in sign (+0.29 and −0.34). Geometric change at matched accuracy is therefore not a proxy for forgetting, and the sign of the rank correlation depends on optimiser and readout.

Combination	ρ(CKA, forgetting)	ρ(LoP drop, forgetting)
Adam, single-head	−0.83	−0.41
SGD, single-head	+0.54	−0.74
Adam, multi-head	+0.29	−0.40
SGD, multi-head	−0.34	−0.79

Table 7: Spearman rank correlations among accuracy-matched methods (SI, ER, XdG+SI; 15 seeds; $n = 45$ per combination). Naive fine-tuning is excluded.

Figure 6 shows the same points, including naive fine-tuning for contrast. Naive occupies the high-change, high-forgetting quadrant in every combination; those runs are not used in Table 7. SI and ER under task-IL cluster at low forgetting. XdG+SI under domain-IL shows low geometric change but intermediate-to-high forgetting. Accuracy-based loss of plasticity is treated separately in §3.5.

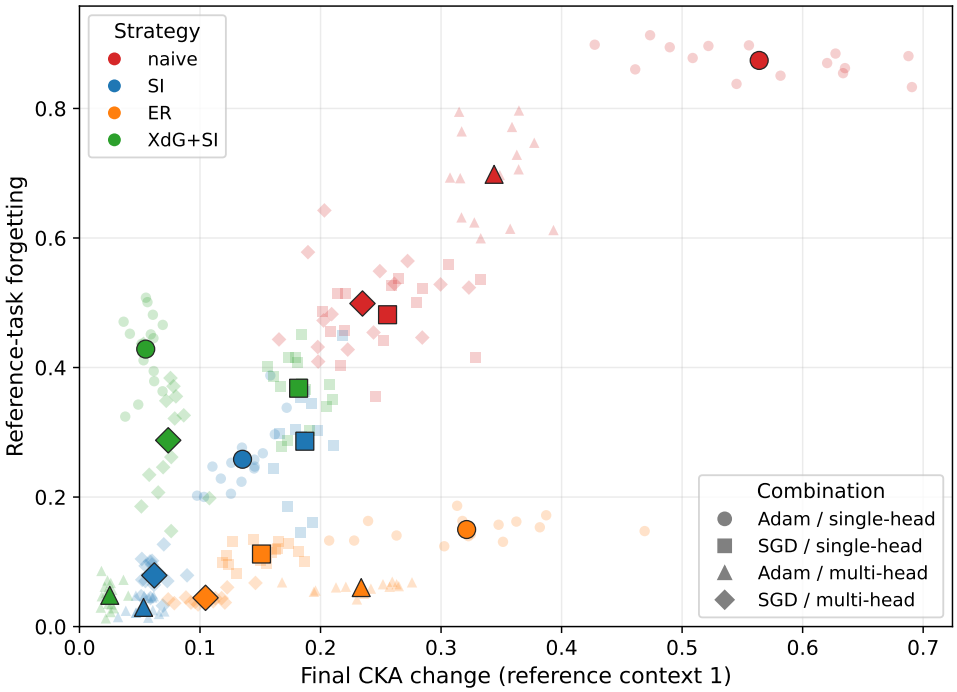


Figure 6: Coupling of final CKA change and reference-task forgetting. Colour is the strategy; marker shape is the optimiser–readout combination. Naive fine-tuning is shown for contrast and is excluded from Table 7. Faint markers are individual seeds; large opaque markers are the means of strategy×combination.

3.5 Loss of plasticity and structural diagnostics

Loss of plasticity is distinct from forgetting. A network can retain early tasks yet learn new ones increasingly poorly [3]. At the 50-context horizon used here, that accuracy-based signature of loss of plasticity is still small under Adam. Late learnability remains above 0.95 for every matched Adam strategy (Table 8). Early learnability (current-task accuracy on contexts 1–5) is near ceiling in the same tables, so the LoP drop is a late-stream change rather than a failure to learn the first tasks. The exception is SGD with experience replay: late learnability falls to 0.89 (single-head) and 0.81 (multi-head), with LoP drops of 0.03 and 0.11 respectively, the only clear functional decline of the kind reported by Dohare et al. [3] (Table 9; Appendix A.8). Among the same accuracy-matched methods, plasticity drop is negatively associated with forgetting in every combination (Table 7: ρ from -0.40 to -0.79): methods that retain early tasks more strongly are the ones that begin to lose late-task plasticity.

The structural diagnostics logged after each context (dead-unit fraction, effective rank, and mean absolute weight) tell a different story, and it is the one that motivates the noise interventions. Under Adam single-head, naive fine-tuning ends with 81% dead hidden units

Table 8: Loss-of-plasticity diagnostics (Adam, single-head): mean ± 95% CI. Early learnability is current-task accuracy on contexts 1–5; late learnability is contexts 41–50; LoP drop is early minus late.

Strategy	Early learn.	Late learn.	LoP drop	Dead units	Eff. rank	Mean $\|w\|$
naive	0.981±0.001	0.976±0.001	0.005±0.001	0.812±0.012	12.0±0.4	0.106±0.000
SI	0.980±0.001	0.954±0.001	0.026±0.002	0.000±0.000	15.8±0.3	0.049±0.000
ER	0.981±0.001	0.974±0.001	0.007±0.001	0.691±0.026	11.5±0.5	0.122±0.000
XdG+SI	0.967±0.001	0.972±0.001	-0.006±0.002	0.577±0.011	11.8±0.3	0.057±0.000

Table 9: Loss-of-plasticity diagnostics (SGD, single-head): mean ± 95% CI. Early learnability is current-task accuracy on contexts 1–5; late learnability is contexts 41–50; LoP drop is early minus late.

Strategy	Early learn.	Late learn.	LoP drop	Dead units	Eff. rank	Mean $\|w\|$
naive	0.947±0.001	0.976±0.001	-0.030±0.002	0.006±0.001	12.7±0.2	0.021±0.000
SI	0.942±0.001	0.931±0.001	0.010±0.002	0.026±0.003	9.4±0.2	0.020±0.000
ER	0.926±0.002	0.893±0.002	0.033±0.002	0.036±0.003	7.7±0.3	0.018±0.000
XdG+SI	0.940±0.001	0.971±0.001	-0.031±0.002	0.068±0.004	11.3±0.2	0.021±0.000

and still learns context 50 at 97.6% accuracy. ER, despite the lowest forgetting in this combination, is almost as saturated (69% dead). SI is the outlier: dead-unit fraction is 0.000, effective rank is highest among the single-head Adam strategies (15.8), and mean absolute weight is less than half that of naive or ER (0.049 vs. 0.11–0.12). The SI penalty acts on weights rather than through the ReLU, so it both caps Adam's weight growth and supplies a restoring gradient for units that would otherwise remain off. SGD produces almost no dead units under any strategy (0.6–7%), so ReLU death is an Adam-domain-IL phenomenon not observed in our other combinations.

Figure 7 makes the decoupling explicit. The high-dead unit cluster is Adam single-head naive and ER. SI sits near the origin (no dead units, a small positive LoP drop). Multi-head SGD ER occupies the opposite corner: few dead units, the largest LoP drop. Restricting to accuracy-matched Adam single-head SI and ER, dead-unit fraction and LoP drop rank oppositely ($\rho = -0.75$, $n = 30$): ER saturates hidden units without a late-task drop, SI keeps every unit alive and still slows slightly. We do not report that coefficient in the other combinations. SGD produces almost no dead units, so there is nothing to rank, and XdG+SI dead-unit counts are confounded by the context mask. Dead ReLUs are therefore not a proxy for the inability to learn new tasks at this horizon, and preserving units (SI) does not by itself prevent the modest late-task slowdown. Task-IL Adam is the other structural extreme: naive and ER keep effective rank near 70 with only 3–7% dead units, while SI compresses rank to 30 (Appendix A.8).

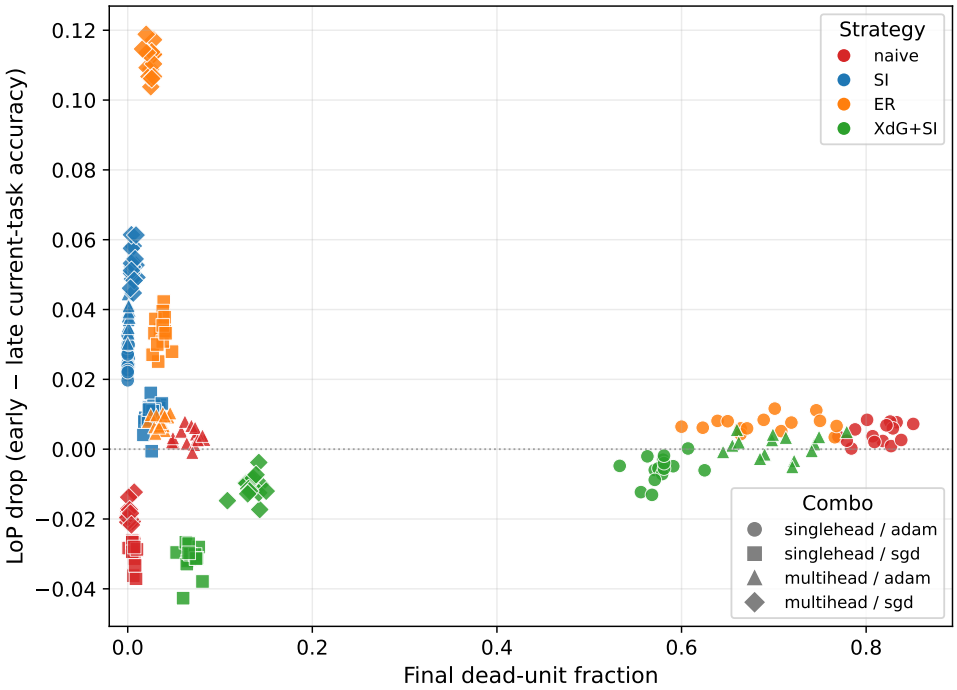


Figure 7: Per-seed coupling of final dead-unit fraction and accuracy-based LoP drop across all strategy-comparison combinations (matched strategies). LoP drop is early minus late current-task accuracy. The two axes dissociate: Adam single-head naive/ER saturate hidden units without losing late learnability; multi-head SGD ER shows the reverse.

3.6 Metric agreement and measurement validity

Averaging Spearman rank correlations of strategy means across the four combinations of optimiser and architecture, agreement between CKA and Procrustes is $\rho = 0.95$. Agreement of either geometry metric with parameter cosine is lower ($\rho = 0.65$ for CKA versus parameter cosine, $\rho = 0.73$ for Procrustes versus parameter cosine). The two geometry metrics therefore order strategies in almost the same way, whereas global weight change does not.

That split is largest under Adam. In both single-head and multi-head, CKA and Procrustes rank XdG+SI lowest, then SI, then ER, then naive. Parameter cosine instead ranks SI lowest by a wide margin and places ER, XdG+SI, and naive close together (Tables 5 and 12). Under SGD the metrics agree more: ER is lowest on all three distances in the single-head setting, and SI is lowest on CKA and Procrustes in the multi-head setting (Tables 6 and 13). The remaining disagreement is again XdG+SI, which can look geometrically still while its weight vector keeps moving. That is largely due to how the two quantities are measured. CKA and Procrustes use hidden activations with the reference-task mask applied, so gated-off units are zeros and the geometry mainly reflects the reference subnetwork. Parameter cosine uses the full unmasked weight vector, including synapses that later contexts overwrite in other gated

subnetworks.

Because CKA and Procrustes agree, the SI-ER contrast in §3.3 is not an artefact of one similarity index. The same weight and activity split on the Adam single-head runs is already in Figure 5.

3.7 Induced change and plasticity recovery

The induced-change experiment tests whether uniform dropout, targeted dropout, gradient noise, or synaptic turnover can restore plasticity without accelerating forgetting when added to naive, SI, ER, and XdG+SI configurations (Adam, single-head, 15 seeds, 41 noise conditions per strategy). All Δ values are within-strategy. Each noise condition is compared to the no-noise baseline of the same method. Because that comparison does not require matching SI, ER uses a stronger replay buffer ($B = 100$) than the SI-matched $B = 20$ of Figure 2. We count an intervention as meeting all three targets if ΔCKA > 0, Δforgetting ≤ 0, and Δlate learnability > 0 simultaneously. Success in inducing drift would be an increase in measured representational change.

Figure 8 confirms the strategy ranking under no noise. Across 2,460 runs, the three-target criterion is rarely met: 0% for gradient noise (600 runs), ≈4% for uniform and targeted dropout (600 runs each), and ≈7% for synaptic turnover (600 runs).

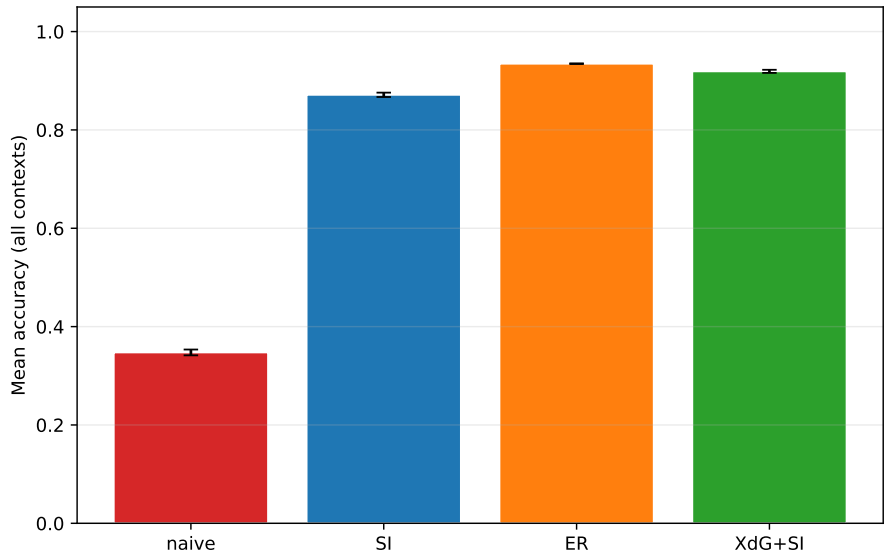


Figure 8: No-noise baseline of the induced-change experiment (Adam, single-head, 15 seeds, mean ± 95% CI). ER uses $B = 100$ and XdG+SI uses $c = 0.05$, not the SI-matched locked settings of Figure 2.

Mild uniform dropout ($p = 0.05$) is the most promising intervention for XdG+SI:

Δforgetting $= -0.09$ with ΔCKA ≈ 0 (Table 21). The gain is in retention, not in measured geometric change. High dropout ($p \geq 0.8$) reduces late learnability across strategies (Figure 9).

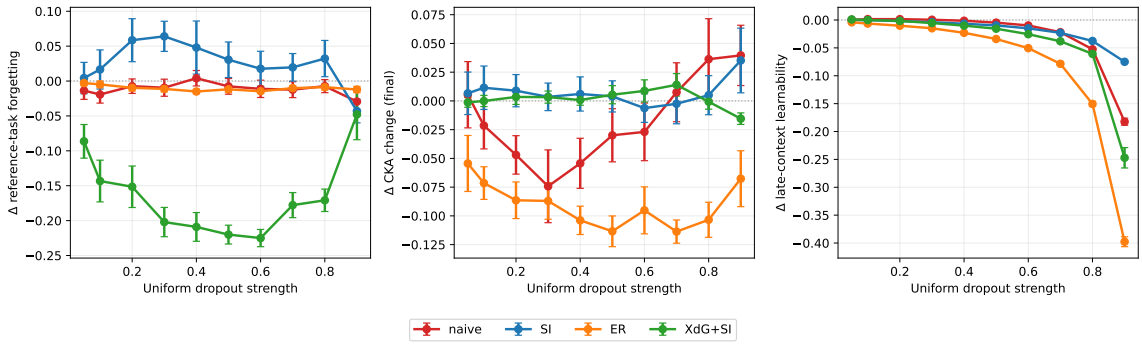


Figure 9: Uniform dropout: change versus the no-noise baseline across noise strengths. Left: Δforgetting. Middle: ΔCKA change. Right: Δlate learnability.

Table 10: Best induced-change setting per strategy under Δforgetting ≤ 0.02 (else least harmful).

Strategy	Best noise	ΔLate	ΔF	ΔCKA
ER	synaptic turnover $f = 0.002$	$+0.000$	$+0.000$	-0.013
SI	targeted dropout $A = 2$	$+0.000$	$+0.008$	$+0.001$
XdG+SI	uniform dropout $p = 0.05$	$+0.001$	-0.086	-0.001
naive	synaptic turnover $f = 0.35$	$+0.003$	$+0.006$	-0.019

Targeted dropout tracks uniform dropout (Figure 10): XdG+SI forgetting falls at moderate A, while CKA change and late learnability do not rise.

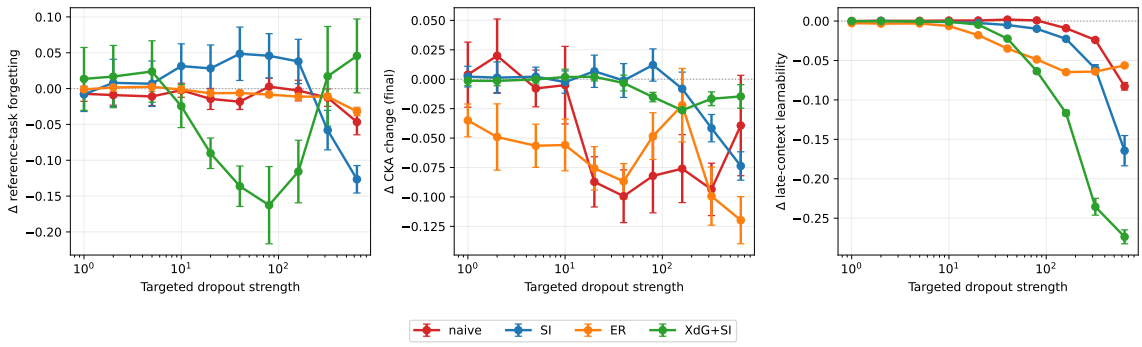


Figure 10: Targeted dropout: change versus the no-noise baseline. The pattern tracks uniform dropout: XdG+SI forgetting falls at moderate A, while CKA change and late learnability do not rise.

From $\sigma \geq 0.003$ onward, gradient noise monotonically degrades mean accuracy and late learnability for all strategies (Figure 11). For naive fine-tuning and SI it also reduces measured CKA change. At $\sigma \geq 10$ under ER, ΔCKA turns positive (+0.28) while forgetting worsens sharply (+0.40), so increased geometric change never co-occurs with retention or plasticity gains.

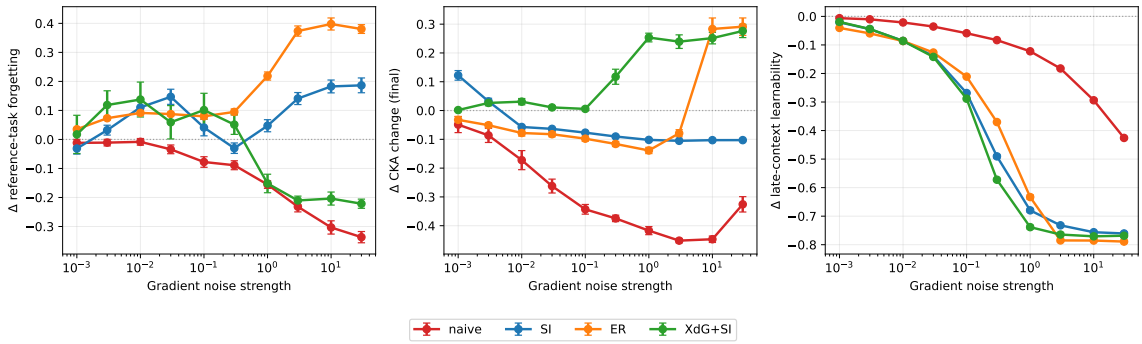


Figure 11: Gradient noise: change versus the no-noise baseline. Low σ suppresses CKA change. High σ under ER increases geometric change only alongside severe forgetting and a drop in late learnability.

Synaptic turnover leaves naive and ER forgetting almost unchanged, but from $f \gtrsim 0.01$ it drives SI and XdG+SI forgetting upward (Figure 12). CKA change increases with that collapse rather than with retained performance. The full table for all 160 non-baseline conditions is in Appendix A.9.

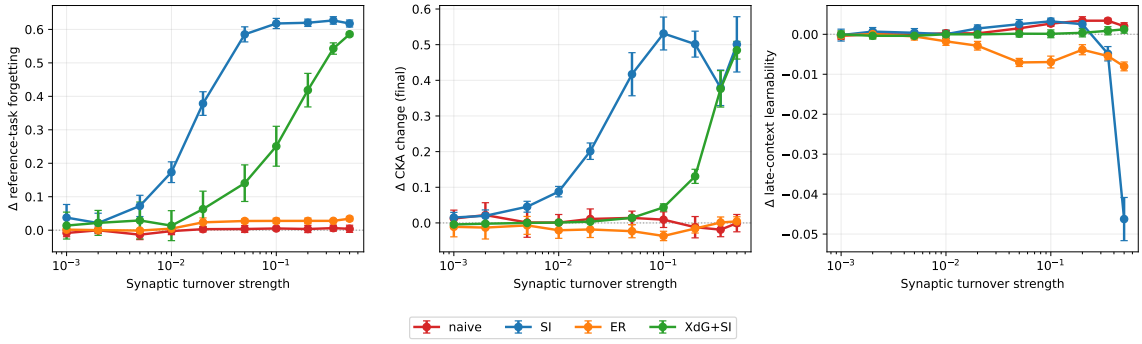


Figure 12: Synaptic turnover: change versus the no-noise baseline. Naive and ER forgetting stay near zero (already high for naive). From $f \gtrsim 0.01$, SI and XdG+SI forgetting rise sharply; CKA change increases with that collapse rather than with retained performance or late-learnability gains.

Because late learnability is already near ceiling under Adam single-head (≈ 0.95–0.98), the three-target criterion has little room to record a plasticity gain. Dead-unit fraction and

effective rank do have room: naive fine-tuning starts at 81% dead units and rank ≈ 12. Uniform dropout reverses that saturation of hidden units without the forgetting cost that the strict criterion was designed to guard against. At $p = 0.3$, dead-unit fraction falls from 0.81 to 0.007 and effective rank rises from 12 to 38, while Δforgetting $= -0.01$ and Δlate learnability ≈ 0 (Appendix A.10, Figure 30; Table 11). Mild dropout on ER is a smaller version of the same pattern (Δdead $= -0.20$ and Δrank $= +14$ at $p = 0.1$). SI has no dead units to revive, so the same interventions are nearly flat on Δdead and only hurt late learnability at high dose. Synaptic turnover, the intervention closest to Dohare et al. [3], does reduce dead units under naive fine-tuning (to 0.48 at $f = 0.35$) but collapses mean accuracy from 0.35 to 0.16. The network stays plastic on the current task because it has already forgotten everything else.

Table 11: Best dead-unit recovery per strategy under Δforgetting ≤ 0.02 and Δlate learnability ≥ -0.01 (Adam, single-head). SI is omitted: dead-unit fraction is already 0 at baseline.

Strategy	Setting	Δdead	Δrank	ΔF
naive	uniform $p = 0.3$	-0.805	$+25.7$	-0.010
ER	uniform $p = 0.1$	-0.203	$+14.3$	-0.005
XdG+SI	uniform $p = 0.05$	-0.056	-0.0	-0.086

Appendix A.10 plots the same Δdead and Δrank curves for all four noise families. Dropout and targeted dropout recover dead units for naive and ER while Δforgetting stays near zero. Gradient noise also lowers dead-unit fraction and can inflate effective rank by $+50$ to $+80$ (Figure 32), but those rank gains co-occur with collapsed late learnability. Rank inflation from gradient noise does therefore not provide plasticity. Synaptic turnover reduces dead units under naive fine-tuning and ER only at high f, where forgetting in SI and XdG+SI has already collapsed (Figure 33). The useful settings are those where dead-unit fraction falls without a rise in forgetting: naive and ER dropout do this; high gradient noise and high synaptic turnover do not.

4 Discussion

4.1 Main findings

The comparison was built around two questions. The first is whether synaptic intelligence and experience replay still differ in how much the hidden code moves when they are matched on mean accuracy. The second is whether extra representational change protects learning on later tasks. Under Adam the first contrast is clear. The second is not testable in a strong form at 50 contexts, because late-task accuracy is still high.

Synaptic intelligence and experience replay both retain the reference task under Adam, yet they do not do so by the same internal route. SI produces the smallest geometric change (lowest CKA and Procrustes) and the smallest parameter displacement. ER, at the same mean accuracy, permits roughly twice the CKA change of SI and a parameter distance close to naive fine-tuning: behaviour is stabilised without freezing the weights. Because reference-task accuracy is held, that extra geometric remapping is the closest analogue in this project to representational drift. CKA and Procrustes order the strategies in the same way, so the SI–ER contrast is not an artefact of one similarity index. Context-dependent gating paired with SI can yield still lower CKA under task-IL, where each context has its own readout.

The same Adam runs also split weight space from activity space (Figure 5). ER overwrites the flattened weight vector almost as much as naive fine-tuning, while the mean hidden activation on reference-task images, and the geometry of those activations, remain much closer to the reference snapshot. SI keeps both the synapses and the code relatively still. Weight movement is therefore not a proxy for representational change. That dissociation is what makes the network useful as a model of biological learning, a point taken up in §4.3.

Under Adam single-head, extra geometric remapping goes with less forgetting, not more ($\rho = -0.83$). That is the setting in which the extra CKA can be read as drift. The other optimiser–readout combinations do not share that sign (Table 7). Naive fine-tuning, which is not accuracy-matched, sits in the high-change, high-forgetting quadrant in Figure 6; those runs are overwriting, not drift, and are not included in the coefficients. XdG+SI under domain-IL remains the complementary case: low measured CKA with high forgetting on a shared readout, so geometric stillness is not retention either.

The SI–ER geometric ordering is specific to Adam. Under SGD, SI forgets more than ER, and the accompanying CKA change is correspondingly larger. That extra change tracks extra forgetting, so it should not be read as extra drift, and it does not undo the Adam result. Crossing optimiser and readout was deliberate. The same regularisation strength, buffer size, or gating fraction can look different under Adam versus SGD and under task-IL versus domain-IL. Small experimental choices (optimiser, whether task identity is available at test time, how hyperparameters are matched) have an impact on the numbers. The design locked those choices rather than reporting a single favourite setting.

Among SI, ER, and XdG+SI, plasticity drop is negatively associated with forgetting in

every optimiser–readout combination (ρ from -0.40 to -0.79; Table 7): methods that retain early tasks more strongly are the ones that begin to lose late-task plasticity. Dead ReLUs are not a stand-in for that loss. Under Adam single-head, ER saturates most hidden units and still learns context 50 at high accuracy; SI keeps every unit alive and still shows a small late-task slowdown. Structural death and functional loss of plasticity come apart at this horizon.

The idea that extra change would protect later learning is not supported at 50 contexts, but the test is weak. Late learnability is already near ceiling under Adam, so there is little room for a protective effect to appear. The one clear late-task drop is SGD with experience replay, which is not the high-change condition, and which also down-weights the current task in the replay loss as the stream lengthens. The induced-change experiment runs into the same ceiling: there is little plasticity to restore. Dropout, gradient noise, and synaptic turnover rarely increased CKA while holding forgetting down and raising late learnability. Mild dropout did revive silent hidden units for naive fine-tuning and ER without adding forgetting, and it improved XdG+SI retention (Δforgetting $= -0.09$). Those are structural and retention effects, not a demonstration that induced drift is beneficial.

4.2 Interpreting change under the stability-plasticity lens

Once representations are measured, not only accuracy, the four strategies sit in different places on the stability-plasticity trade-off. Naive fine-tuning is plastic and unstable: it overwrites weights and the reference code, and it forgets. SI is stable in both parameter space and hidden geometry once importance weights accumulate; the cost, at this scale, is a small late-task slowdown rather than a collapse of current-task accuracy. ER is stable in behaviour while remaining plastic in the weights: the parameters can move through overlapping low-loss regions defined by stored and current data. XdG+SI can look geometrically still on the gated reference subnetwork and still forget when a shared classifier must integrate information across contexts.

CKA, Procrustes, and cosine distances always quantify change. They become evidence of drift only when performance on the probed inputs remains stable. Under that rule, used already for the Adam and SGD comparisons, the Adam SI–ER contrast is a drift result and the SGD SI–ER contrast is not. At 50 contexts the change that remains compatible with retention is more visible in unit activity (dead-unit fraction, effective rank, mean hidden activation) than in late-task accuracy. That is another reason the second question is hard to close here: the interesting variation is in the representation, not in current-task scores.

4.3 Artificial networks as a model of biological representational change

Biological recordings established that population codes for fixed stimuli can reorganise over days while behaviour stays stable [7, 8, 13]. They cannot, in the same animal, deliver the complete synaptic matrix of those cells, hold the learning rule fixed, or swap consolidation for replay while keeping the stimulus set identical [6, 22]. An artificial network is useful here not because it is a brain, but because it is a controllable model of the same computational problem: how to keep learning without overwriting what was already learned.

The practical gain is a joint measurement that neuroscience does not have. Every snapshot in this project stores both the hidden population vector and the full set of weights. Figure 5 makes use of this feature. Replay then looks like function-preserving remapping: nearly naive levels of synaptic displacement with a much more stable activity code, analogous to cortical drift in which neuron-level turnover coexists with a usable linear readout [7]. SI looks like consolidation of synapses and representations, closer to a stability mechanism that freezes both. Without the weight snapshot, those two solutions would be hard to tell apart from activity alone. Without the activity snapshot, SI and ER would look more similar in the engineering metrics that continual learning usually reports.

The same controllability is what lets the comparison be matched on accuracy. Biological drift is defined against stable behaviour. Matching SI and ER on mean accuracy is the artificial counterpart of that constraint. When accuracy is not matched, as with naive fine-tuning or with SI under SGD, large CKA is not evidence that the network has found an equivalent code; it is evidence that the old code has been overwritten. Treating the MLP as a model system makes that distinction testable rather than rhetorical.

A further gain of using an artificial network is complete access to the hidden population. Calcium imaging follows a sampled subset of identified cells over days. Here every hidden unit is available at every snapshot, so identity and turnover can be tracked for the full population rather than a partial field of view. This project reports population geometry (CKA, rank, dead-unit fraction) rather than per-unit turnover. Restricting those summaries to a random subset of units would mainly add noise; whether identity-level turnover would change the SI–ER contrast is a separate check, and we do not expect it to. What the setup can already show is that different artificial stability mechanisms, under the same stimuli and a matched performance budget, produce different representational trajectories, and that those trajectories are visible only if weights and activity are logged together.

4.4 Limitations

Results have been tested on permuted MNIST and to a 50-context horizon. Further testing with other types of datasets would be needed to confirm the results. Loss of plasticity is much weaker here than in the long-horizon reports of Dohare et al. [3]. The 800-context

naive-Adam appendix shows that the signature can appear in this architecture, but the strategy comparison was not run at that length. Permuted MNIST also has no sensory hierarchy, so there is no analogue of areas that drift at different rates [8, 9]. Dropout, gradient noise, and synaptic turnover are not biological synaptic turnover [3, 27]. Because the induced-change interventions did not raise CKA under stable retention, this project cannot say whether biological drift is an active aid to lifelong learning or a byproduct of ongoing plasticity.

The optimiser and the continual-learning scenario are not details. Adam versus SGD, and task-IL versus domain-IL, change both forgetting and measured geometry. Claims about regularisation versus replay therefore have to name those choices. Method-specific hyperparameters were selected on seed 1 and then evaluated with 15 seeds, so the locked SI, ER, and XdG+SI settings can be slightly overfit to that first seed. Geometry metrics use 32 reference images in a 2,000-dimensional hidden space; CKA–Procrustes agreement protects the ranking, not the absolute distances.

Joint training, used to lock network size, was run at 50k and 100k iterations, not at the 250,000 steps of the sequential stream, so it is just a capacity check. Under domain-IL, XdG+SI still receives a context index to select the hidden-unit mask, so the hybrid is not a strict domain-IL method in the sense of van de Ven and Tolias [19].

4.5 Future work

The first suggested follow-up is to repeat the locked SI/ER comparison at a horizon where late-task accuracy actually falls. The 50-context Adam runs never produce the drop reported by Dohare et al. [3]; the 800-context naive-Adam appendix does. Until that drop is present, the question of whether extra change protects later learning, and the induced-change experiment, remain weak tests. Using a lower amount of iterations per task could make this experiment more viable while not compromising too much on performance. The same longer stream would also show whether the Adam SI–ER geometric split survives after importance weights and replay buffers have been accumulating for hundreds of tasks.

Induced-change interventions should be reserved for that longer setting, or for a configuration that already shows loss of plasticity at 50 tasks (SGD with replay is the candidate in the present data). Sweeping dropout on a network that is still learning new tasks at 97% mainly tests whether silent units can be revived, which this project already shows. It does not test whether noise restores learning. Other ways of injecting variability into the model could be tried in that same longer setting, rather than on a network that is still near ceiling.

Any follow-up should still lock the optimiser, or cross it, before claiming a family-level effect: the SGD control here was enough to stop a naive reading of SI versus ER as a universal regularisation-versus-replay law. It should also match methods on accuracy before interpreting geometry. Reporting the highest-accuracy SI run against a small-buffer ER run would have mixed retention with remapping.

The model itself can be moved closer to the biology that motivated the project. An MLP in which every layer has the same role cannot capture the empirical pattern that some cortical areas drift while others remain relatively stable [8, 9]. A hierarchy, or a split between a stable readout and a plastic encoder, would let the same metrics ask where change is tolerated. Tracking unit identity over time, as calcium imaging does, would test whether per-unit turnover adds anything to the SI–ER split; we expect the population metrics already capture it. A second dataset with spatial or semantic structure (for example a class-incremental or domain-incremental vision stream) would show whether the Adam SI–ER split is a permuted-MNIST fact or a fact about how these algorithms reuse features.

4.6 Conclusion

Under Adam, at matched mean accuracy, regularisation (SI) suppresses representational drift more than replay (ER), which remaps the hidden code while retaining the reference task. That is the result that the geometry metrics, the weight–activity split, and the 15-seed Adam comparisons all support. SGD was included as a control: there SI forgets more, and the extra CKA change follows that forgetting rather than reversing the family-level claim. Optimiser identity therefore belongs in the interpretation, not as a qualifier that undoes the Adam finding. Whether extra change would protect later learning is left open, because late-task accuracy barely falls at this horizon.

Relative to the scenario framework of van de Ven and Tolias [19], the plasticity diagnostics of Dohare et al. [3], and the geometry of drift in Aitken et al. [7], this comparison adds the same 50-task stream with SI, ER, and XdG+SI scored on both accuracy and representations. The artificial network is useful as a model of biological learning because it yields the synaptic matrix and the population code together, under learning rules that can be swapped while stimuli are held fixed. Continual learning evaluation should report those representational metrics alongside accuracy: SI and ER can match performance while differing in whether stability lives in the weights, in the code, or in both.

References

[1] Gido M. Van De Ven, Nicholas Soures, and Dhireesha Kudithipudi. "Continual Learning and Catastrophic Forgetting". In: *Learning and Memory: A Comprehensive Reference*. Elsevier, 2025, pp. 153–168. ISBN: 978-0-443-15755-4. DOI: 10.1016/B978-0-443-15754-7.00073-0. URL: https://linkinghub.elsevier.com/retrieve/pii/B9780443157547000730.

[2] Shagun Sodhani et al. *An Introduction to Lifelong Supervised Learning*. July 12, 2022. DOI: 10.48550/arXiv.2207.04354. arXiv: 2207.04354 [cs]. URL: http://arxiv.org/abs/2207.04354 (visited on 01/27/2026).

[3] Shibhansh Dohare et al. "Loss of Plasticity in Deep Continual Learning". In: *Nature* 632.8026 (Aug. 2024), pp. 768–774. ISSN: 1476-4687. DOI: 10.1038/s41586-024-07711-7. URL: https://www.nature.com/articles/s41586-024-07711-7 (visited on 11/28/2025).

[4] Suzanne Van Der Veldt et al. "Learning Continually with Representational Drift". In: *Trends Open* (June 2026), S3117347026000222. ISSN: 31173470. DOI: 10.1016/j.treopn.2026.04.013. URL: https://linkinghub.elsevier.com/retrieve/pii/S3117347026000222.

[5] Laura N. Driscoll et al. "Dynamic Reorganization of Neuronal Activity Patterns in Parietal Cortex". In: *Cell* 170.5 (Aug. 24, 2017), 986–999.e16. ISSN: 1097-4172. DOI: 10.1016/j.cell.2017.07.021. PMID: 28823559.

[6] Charles Micou and Timothy O'Leary. "Representational Drift as a Window into Neural and Behavioural Plasticity". In: *Current Opinion in Neurobiology* 81 (Aug. 1, 2023), p. 102746. ISSN: 0959-4388. DOI: 10.1016/j.conb.2023.102746. URL: https://www.sciencedirect.com/science/article/pii/S0959438823000715 (visited on 05/01/2026).

[7] Kyle Aitken et al. "The Geometry of Representational Drift in Natural and Artificial Neural Networks". In: *PLOS Computational Biology* 18.11 (Nov. 28, 2022). Ed. by Peter E. Latham, e1010716. ISSN: 1553-7358. DOI: 10.1371/journal.pcbi.1010716. URL: https://dx.plos.org/10.1371/journal.pcbi.1010716 (visited on 04/28/2026).

[8] Laura N. Driscoll, Lea Duncker, and Christopher D. Harvey. "Representational Drift: Emerging Theories for Continual Learning and Experimental Future Directions". In: *Current Opinion in Neurobiology* 76 (Oct. 1, 2022), p. 102609. ISSN: 0959-4388. DOI: 10.1016/j.conb.2022.102609. URL: https://www.sciencedirect.com/science/article/pii/S0959438822001039 (visited on 02/03/2026).

[9] Paul Masset, Shanshan Qin, and Jacob A. Zavatone-Veth. "Drifting Neuronal Representations: Bug or Feature?" In: *Biological Cybernetics* 116.3 (June 1, 2022), pp. 253–266. ISSN: 1432-0770. DOI: 10.1007/s00422-021-00916-3. URL: https://doi.org/10.1007/s00422-021-00916-3 (visited on 05/06/2026).

[10] Michael E. Rule and Timothy O'Leary. "Self-Healing Codes: How Stable Neural Populations Can Track Continually Reconfiguring Neural Representations". In: *Pro-*

ceedings of the National Academy of Sciences 119.7 (Feb. 15, 2022), e2106692119. DOI: 10.1073/pnas.2106692119. URL: https://www.pnas.org/doi/abs/10.1073/pnas.2106692119 (visited on 12/10/2025).

[11] Maanasa Natrajan and James E. Fitzgerald. "Stability through Plasticity: Finding Robust Memories through Representational Drift". In: *Proceedings of the National Academy of Sciences* 122.45 (Nov. 11, 2025), e2500077122. DOI: 10.1073/pnas.2500077122. URL: https://www.pnas.org/doi/abs/10.1073/pnas.2500077122 (visited on 01/23/2026).

[12] Daniel Anthes and Sushrut Thorat. "Continual Learning in Artificial Neural Networks as a Computational Framework for Understanding Representational Drift in Neuroscience". In: *Cogn. Comput. Neurosci* (2024). URL: https://2024.ccneuro.org/pdf/567_Paper_authored_CCN2024-authored.pdf.

[13] Shanshan Qin et al. "Coordinated Drift of Receptive Fields in Hebbian/Anti-Hebbian Network Models during Noisy Representation Learning". In: *Nature Neuroscience* 26.2 (Feb. 2023), pp. 339–349. ISSN: 1546-1726. DOI: 10.1038/s41593-022-01225-z. URL: https://www.nature.com/articles/s41593-022-01225-z (visited on 05/15/2026).

[14] Guillermo B. Morales, Miguel A. Muñoz, and Yuhai Tu. "Representational Drift and Learning-Induced Stabilization in the Piriform Cortex". In: *Proceedings of the National Academy of Sciences* 122.29 (July 22, 2025), e2501811122. DOI: 10.1073/pnas.2501811122. URL: https://www.pnas.org/doi/10.1073/pnas.2501811122 (visited on 02/03/2026).

[15] Aviv Ratzon, Dori Derdikman, and Omri Barak. "Representational Drift as a Result of Implicit Regularization". In: *eLife* 12 (Apr. 10, 2024). DOI: 10.7554/eLife.90069.2. URL: https://elifesciences.org/reviewed-preprints/90069 (visited on 02/03/2026).

[16] Siyuan Zhao et al. *Realigning Representational Drift in Mouse Visual Cortex by Flexible Brain-Machine Interfaces*. May 28, 2024. DOI: 10.1101/2024.05.23.595627. URL: https://www.biorxiv.org/content/10.1101/2024.05.23.595627v1. Pre-published.

[17] Joonkyu Kim, Yejin Kim, and Jy-yong Sohn. *Measuring Representational Shifts in Continual Learning: A Linear Transformation Perspective*. June 12, 2025. DOI: 10.48550/arXiv.2505.20970. arXiv: 2505.20970 [cs]. URL: http://arxiv.org/abs/2505.20970 (visited on 01/30/2026). Pre-published.

[18] Yiqing Bo et al. "Evaluating Representational Similarity Measures from the Lens of Functional Correspondence". In: *8th Annual Conference on Cognitive Computational Neuroscience*. 2025. URL: https://openreview.net/forum?id=JGANEMteY8.

[19] Gido M. van de Ven and Andreas S. Tolias. *Three Scenarios for Continual Learning*. arXiv.org. Apr. 15, 2019. URL: https://arxiv.org/abs/1904.07734v1 (visited on 05/29/2026).

[20] Friedemann Zenke, Ben Poole, and Surya Ganguli. "Continual Learning Through Synaptic Intelligence". In: *Proceedings of the 34th International Conference on Machine*

Learning. Ed. by Doina Precup and Yee Whye Teh. Vol. 70. Proceedings of Machine Learning Research. PMLR, June 2017, pp. 3987–3995. URL: https://proceedings.mlr.press/v70/zenke17a.html.

[21] David Rolnick et al. "Experience Replay for Continual Learning". In: *Advances in Neural Information Processing Systems*. Ed. by H. Wallach et al. Vol. 32. Curran Associates, Inc., 2019. URL: https://proceedings.neurips.cc/paper_files/paper/2019/file/fa7cdfad1a5aaf8370ebeda47a1ff1c3-Paper.pdf.

[22] Tal Golan et al. *Deep Neural Networks Are Not a Single Hypothesis but a Language for Expressing Computational Hypotheses*. Mar. 21, 2023. DOI: 10.31234/osf.io/tr7gx. URL: https://osf.io/preprints/psyarxiv/tr7gx_v1/ (visited on 12/10/2025). Pre-published.

[23] Nicolas Y. Masse, Gregory D. Grant, and David J. Freedman. "Alleviating Catastrophic Forgetting Using Context-Dependent Gating and Synaptic Stabilization". In: *Proceedings of the National Academy of Sciences* 115.44 (Oct. 30, 2018). ISSN: 0027-8424, 1091-6490. DOI: 10.1073/pnas.1803839115. URL: https://pnas.org/doi/full/10.1073/pnas.1803839115 (visited on 05/15/2026).

[24] Simon Kornblith et al. "Similarity of Neural Network Representations Revisited". In: *ICML2019*. 2019. URL: https://debug-ml-iclr2019.github.io/cameraready/DebugML-19_paper_9.pdf.

[25] Alex H. Williams et al. "Generalized Shape Metrics on Neural Representations". In: *Advances in neural information processing systems* 34 (Dec. 2021), pp. 4738–4750. ISSN: 1049-5258. PMID: 38170102. URL: https://pmc.ncbi.nlm.nih.gov/articles/PMC10760997/.

[26] Farhad Pashakhanloo and Alexei Koulakov. "Stochastic Gradient Descent-Induced Drift of Representation in a Two-Layer Neural Network". In: *Proceedings of the 40th International Conference on Machine Learning*. International Conference on Machine Learning. PMLR, July 3, 2023, pp. 27401–27419. URL: https://proceedings.mlr.press/v202/pashakhanloo23a.html (visited on 01/27/2026).

[27] Dhruva V Raman and Timothy O'Leary. "Optimal Plasticity for Memory Maintenance during Ongoing Synaptic Change". In: *eLife* 10 (Sept. 14, 2021). Ed. by Srdjan Ostojic et al., e62912. ISSN: 2050-084X. DOI: 10.7554/eLife.62912. URL: https://doi.org/10.7554/eLife.62912 (visited on 05/01/2026).

A Appendix: supplementary figures and tables

A.1 Long-horizon loss of plasticity: first approach

Before the 50-context strategy comparison, we asked whether loss of plasticity as described by Dohare et al. [3] shows up at all in this training setup. We followed their sequential protocol: a fully connected network with layer sizes 1,024-2,000-2,000-2,000-10, Kaiming initialisation, naive Adam, batch size 1, and 60,000 steps on each pixel permutation, with no replay, synaptic regulariser, or gating. Each context is a new permutation of MNIST. We ran 800 contexts at three learning rates (10^{-5}, 10^{-4}, and 3×10^{-4}) and averaged across seeds.

Figure 13 is that first check. Online accuracy is measured on the permutation being trained at that moment, so a drop is a failure to keep learning new tasks, not forgetting of earlier ones. That drop is visible at every learning rate and is largest at 3×10^{-4}, where current-task accuracy falls from about 0.94 towards 0.86 and becomes noisy late in the stream. The other three panels give the structural counterpart. Dead-unit fraction rises with learning rate: at 3×10^{-4} almost every hidden unit is inactive within the first 150 contexts; at 10^{-4} the same saturation takes until about context 200; at 10^{-5} units die more slowly and the fraction is still around 0.8 at context 800. Effective rank of the hidden activity collapses in all three runs, from tens of dimensions down to a handful. Mean absolute weight grows throughout, fastest at the largest learning rate.

A single seed at 3×10^{-4} looks like the green traces: current-task accuracy trends down, dead-unit fraction sits near one, rank stays low, and weights keep increasing. Averaging across seeds does not remove the pattern, so it is not a one-run accident.

We retained these four diagnostics for the strategy comparison. The 800-context protocol is too slow to combine with SI, replay, gating, and representational snapshots, and it uses a different width and depth from the two-hidden-layer network used in the main experiments. Those experiments therefore ask the same plasticity questions on a 50-context stream, where the comparisons are feasible. At that shorter horizon, accuracy-based loss of plasticity is still small under Adam (see §3.5), while the structural signatures (dead units, rank compression, weight growth) already appear.

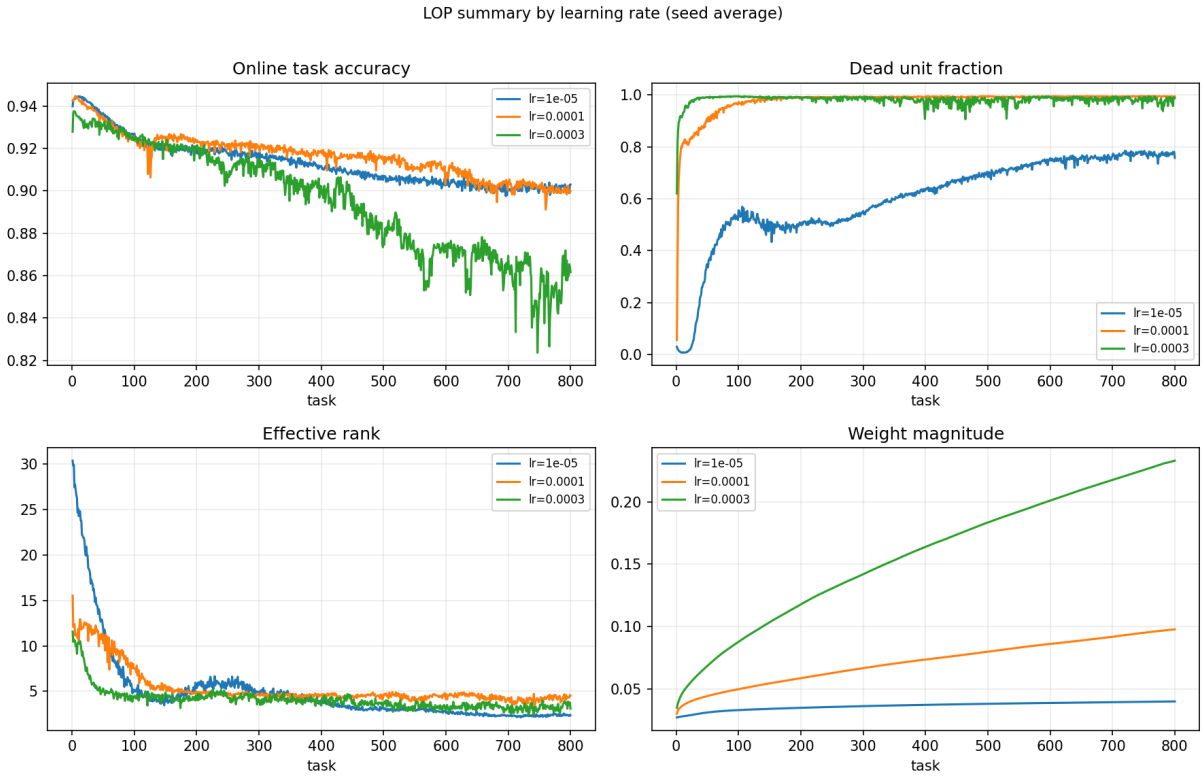


Figure 13: Naive Adam on 800 sequential permuted-MNIST contexts (MLP 1,024-2,000-2,000-2,000-10, batch size 1, 60,000 steps per context, seed average). Panels: current-task accuracy, dead-unit fraction, effective rank of hidden activity, and mean absolute weight. Higher learning rates accelerate the same loss-of-plasticity pattern.

A.2 Joint-training capacity calibration

Figure 14 reports the local joint-training grid used to lock network size. All fifty permuted-MNIST contexts were trained simultaneously (48 completed runs, widths $\{100, 400, 1000, 2000\}$, depths of one to three hidden layers, joint budgets 50k and 100k iterations). Sequential training uses 250,000 steps in total (5,000 per context). Under Adam, mean test accuracy already exceeds 94% at width 1,000 and approaches 97–98% at width 2,000 with the 100k budget; doubling the joint budget buys about 1–2 percentage points at roughly twice the wall-clock cost. SGD saturates lower ($\approx 90\%$) and is more budget-sensitive: 50k runs remain in the mid-80s, while 100k runs cluster near 90%. The locked MLP (1,024-1,000-1,000-C, two hidden layers) sits on the Adam accuracy plateau while remaining cheaper than width 2,000 or three hidden layers (Figure 15).

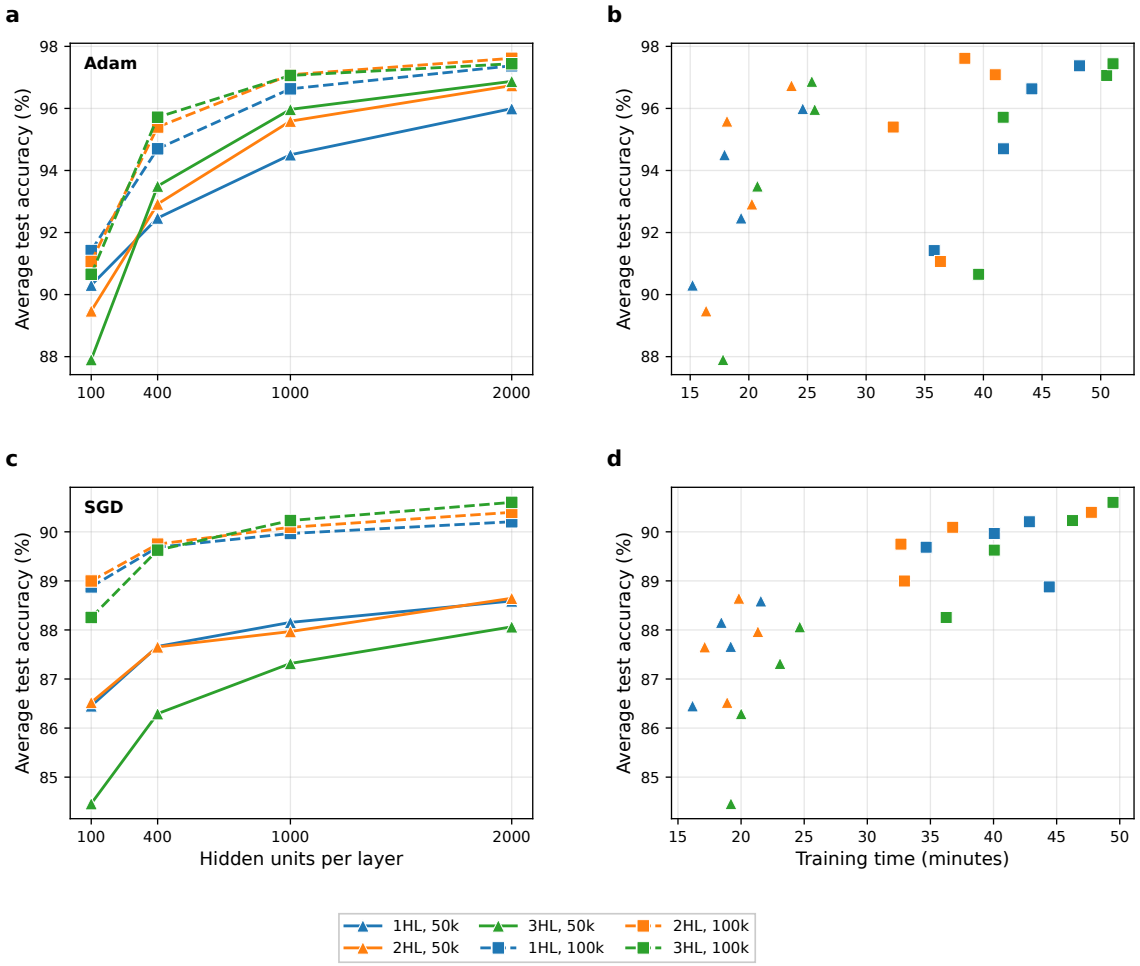


Figure 14: Joint-training capacity on 50-context permuted MNIST (local, same-hardware grid). Average test accuracy versus hidden width (a, c) and versus wall-clock training time (b, d), for Adam (top) and SGD (bottom). Marker/line families encode depth (1HL–3HL) and joint budget (50k vs. 100k).

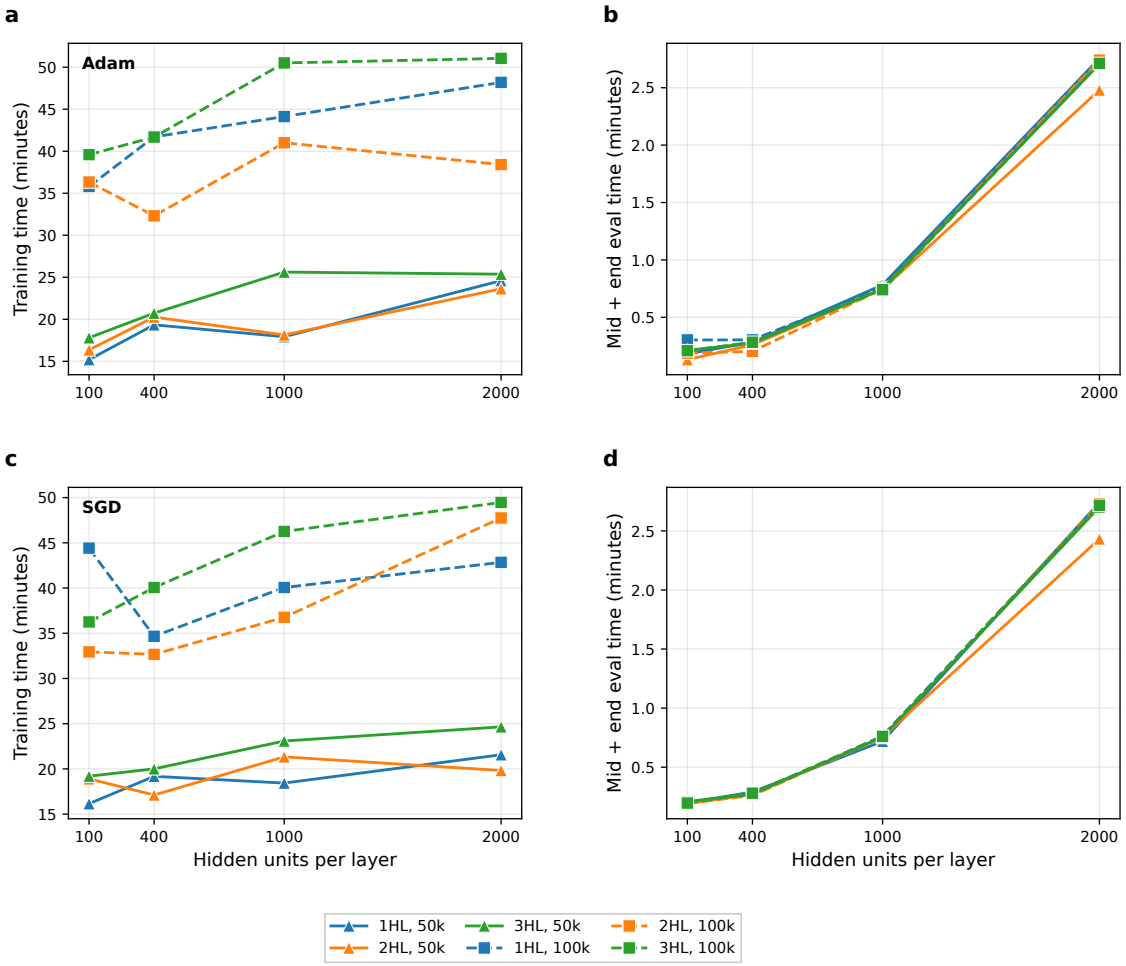


Figure 15: Wall-clock cost of the same joint-capacity grid. Training time (a, c) scales mainly with the joint iteration budget, with secondary growth in width and depth. Mid- and end-of-training evaluation time (b, d) is essentially independent of budget and depth, and grows steeply with width.

A.3 SGD learning-rate probe

An initial SGD rate of 0.2 (zero momentum) was stable for naive fine-tuning but collapsed under SI. Follow-up probes under SI at high regularisation strength selected the lower locked rates of 0.01 (single-head) and 0.02 (multi-head). Figure 16 shows that rates near the initial candidate of 0.2 collapsed, while rates in the 0.01–0.02 range preserved training stability.

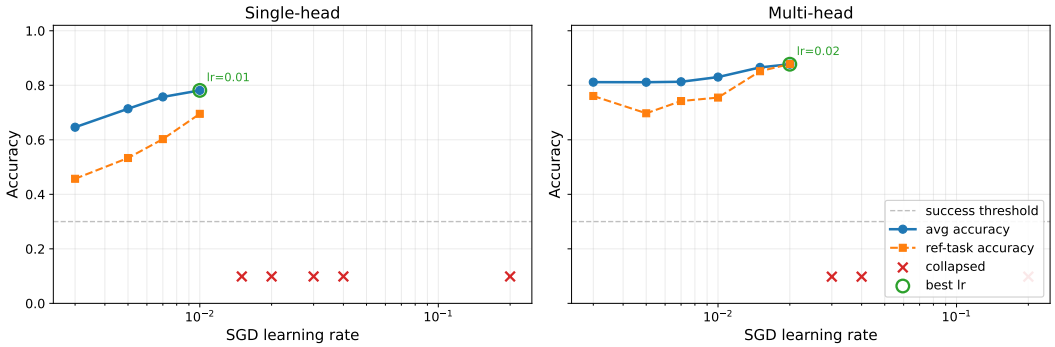


Figure 16: SGD learning-rate probe under SI (combined single-head and multi-head). Lower rates (0.01 to 0.02) preserve training stability relative to the initial candidate rate of 0.2.

A.4 Method hyperparameter sweeps

Method hyperparameters were tuned on seed 1 of the 50-context stream, separately for each combination of optimiser and architecture. The figures below show only average test accuracy over contexts seen so far, which is the quantity used for selection. Current-task accuracy is omitted because every setting learned the active permutation; per-task forgetting traces are omitted because they follow the same ranking as the mean-accuracy curve. The locked setting in each panel is drawn thicker. After SI is locked, later sweeps overlay that SI run as a dashed reference so ER matching and XdG+SI competitiveness can be read directly.

SI regularisation strength c was chosen as the value with the highest final mean accuracy (Figure 17). Under Adam this is $c = 0.2115$ (single-head) and $c = 0.4472$ (multi-head). Under SGD, stronger regularisation is needed: $c = 0.9457$ (single-head) and $c = 2.0$ (multi-head). The largest c is not always best: under Adam single-head, $c = 2.0$ begins to hurt later-task learning, which is why that sweep does not pick the strongest penalty.

ER buffers were not maximised. Larger buffers raise mean accuracy, but the locked B was handpicked so that ER matched the selected SI run rather than saturating replay (Figure 18; Table 3). Domain-IL therefore uses small buffers (B = 10–20); task-IL uses larger ones (B = 100–150). Multi-head SGD $B = 150$ sits between the seed-1 grid values 100 and 500 and was trained only in the strategy comparison.

XdG gating fraction g was selected by final mean accuracy solely to lock the mask density for the hybrid (Figure 19). Even the best g remains below SI/ER on this stream, especially under a shared readout, so XdG alone is a diagnostic rather than a matched comparison condition.

At that locked g, we swept a lighter SI penalty and kept the smallest c that stayed competitive with SI (Figure 20). That is why Adam single-head uses $c = 0.01$ even though larger c scores slightly higher: the hybrid is meant to test architectural isolation with only light consolidation.

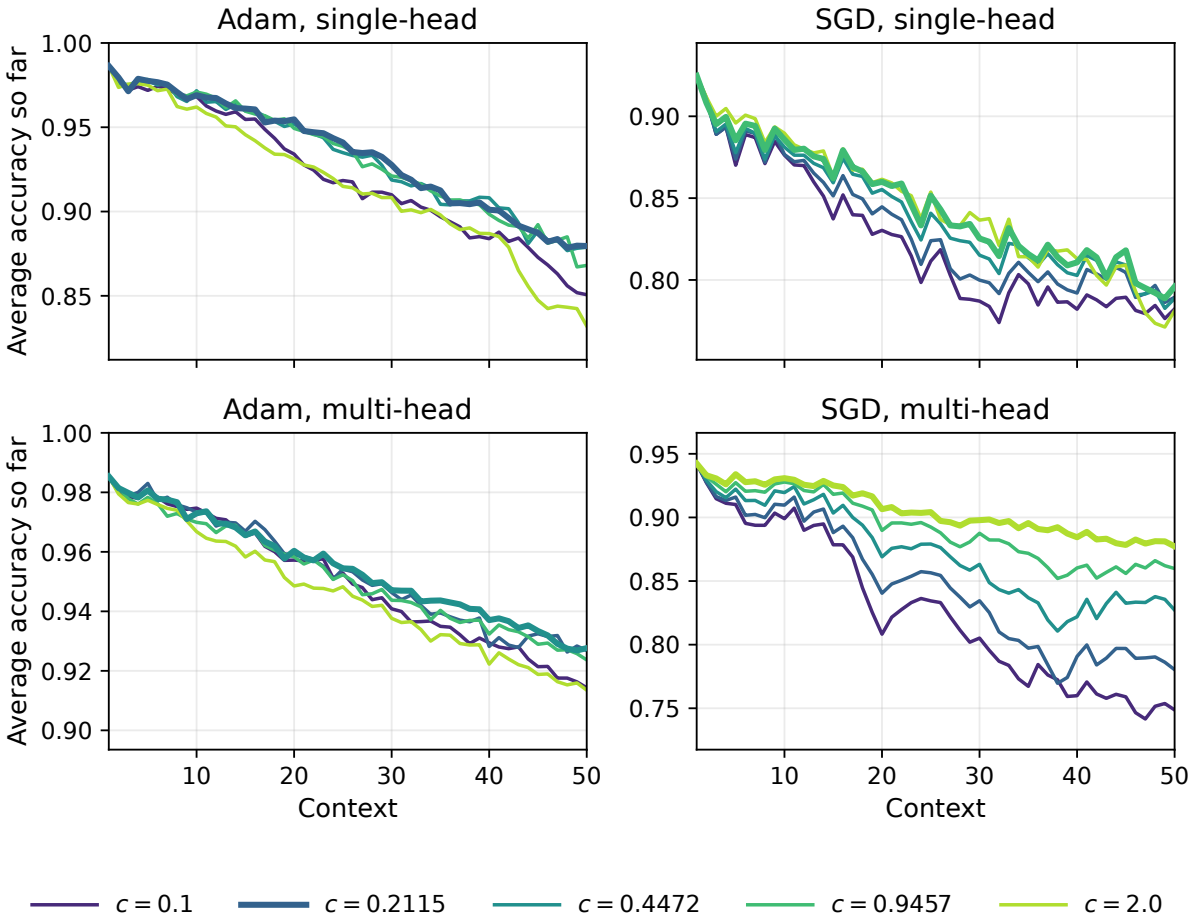


Figure 17: SI regularisation-strength sweep (seed 1). Average accuracy over contexts seen so far. The selected c in each panel maximises final mean accuracy.

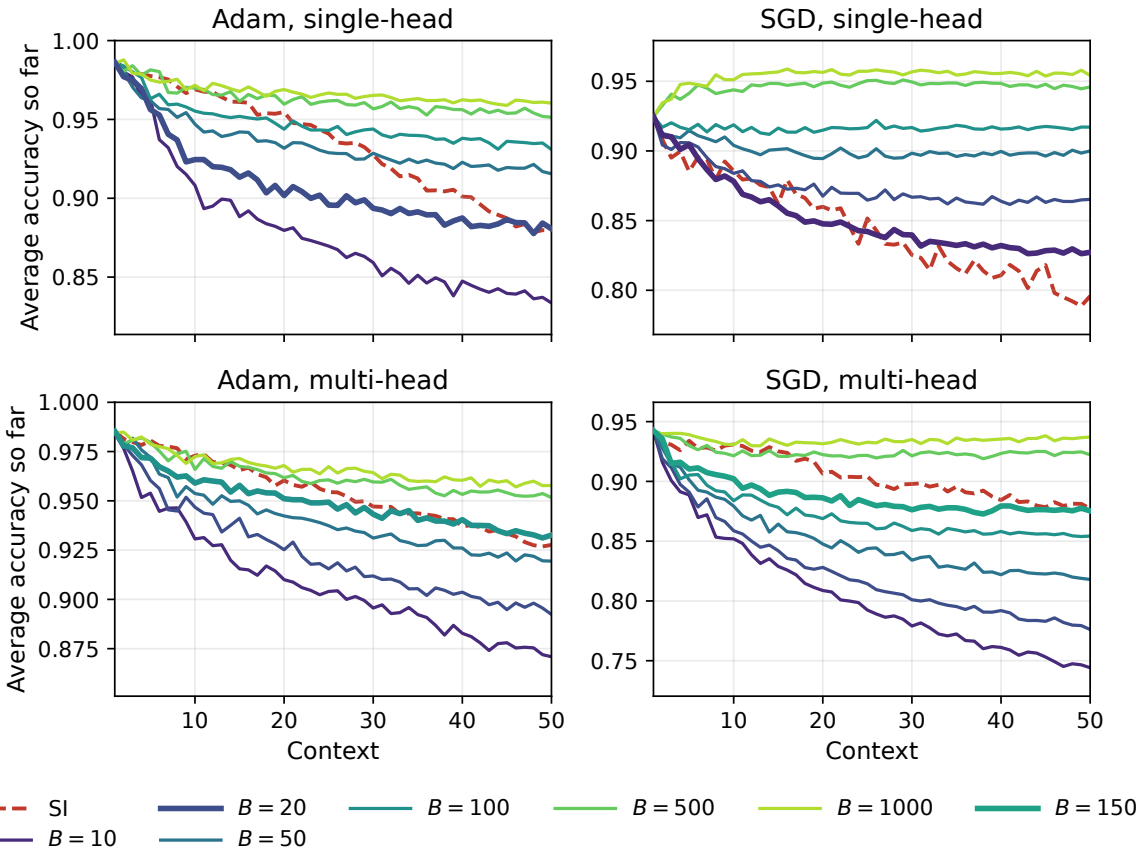


Figure 18: Experience-replay buffer sweep (seed 1). Average accuracy over contexts seen so far. The dashed line is the locked SI run in that panel. The selected B is the matching budget, not the largest one.

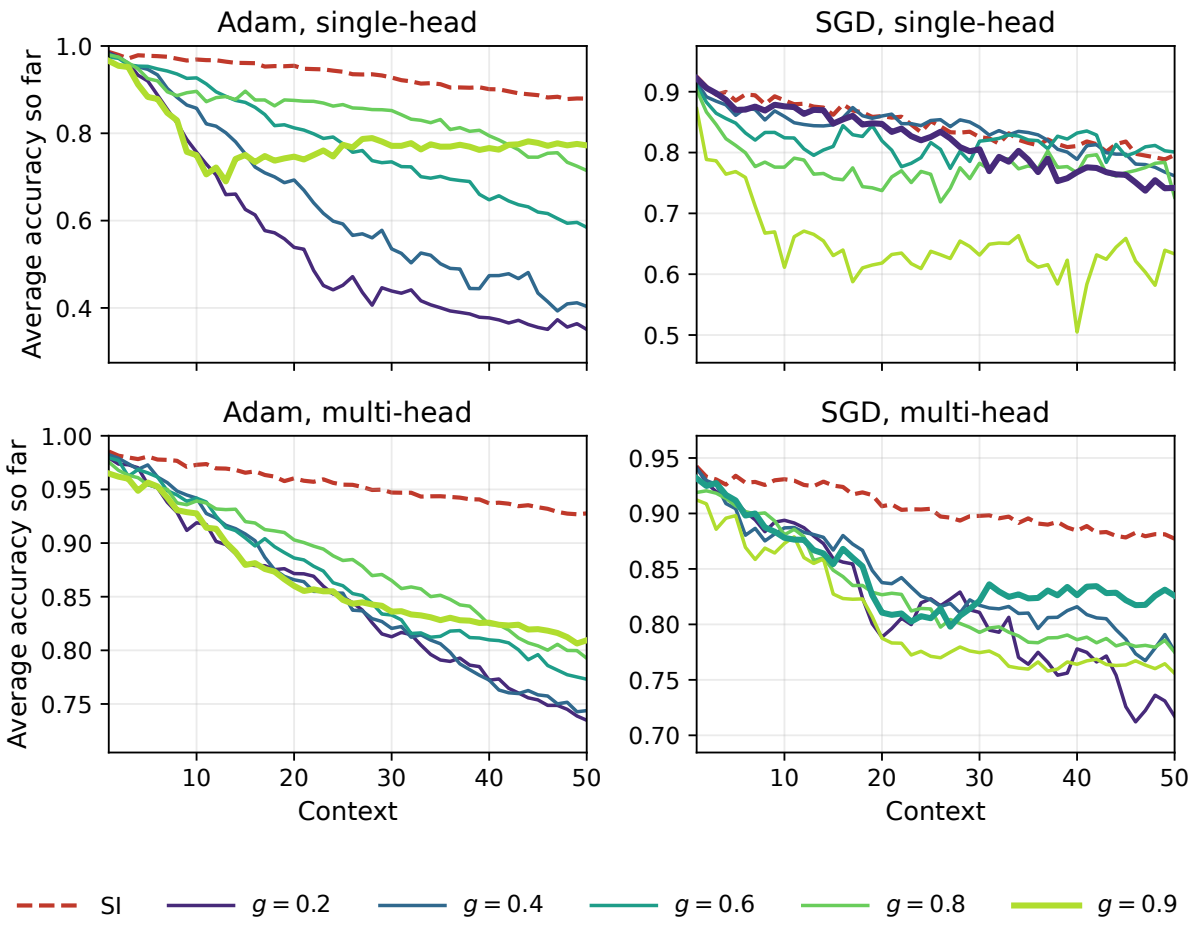


Figure 19: Context-dependent gating sweep without synaptic consolidation (seed 1). Average accuracy over contexts seen so far. The dashed line is the locked SI run in that panel. Selected g is the best XdG setting, used only to fix the hybrid's gating fraction.

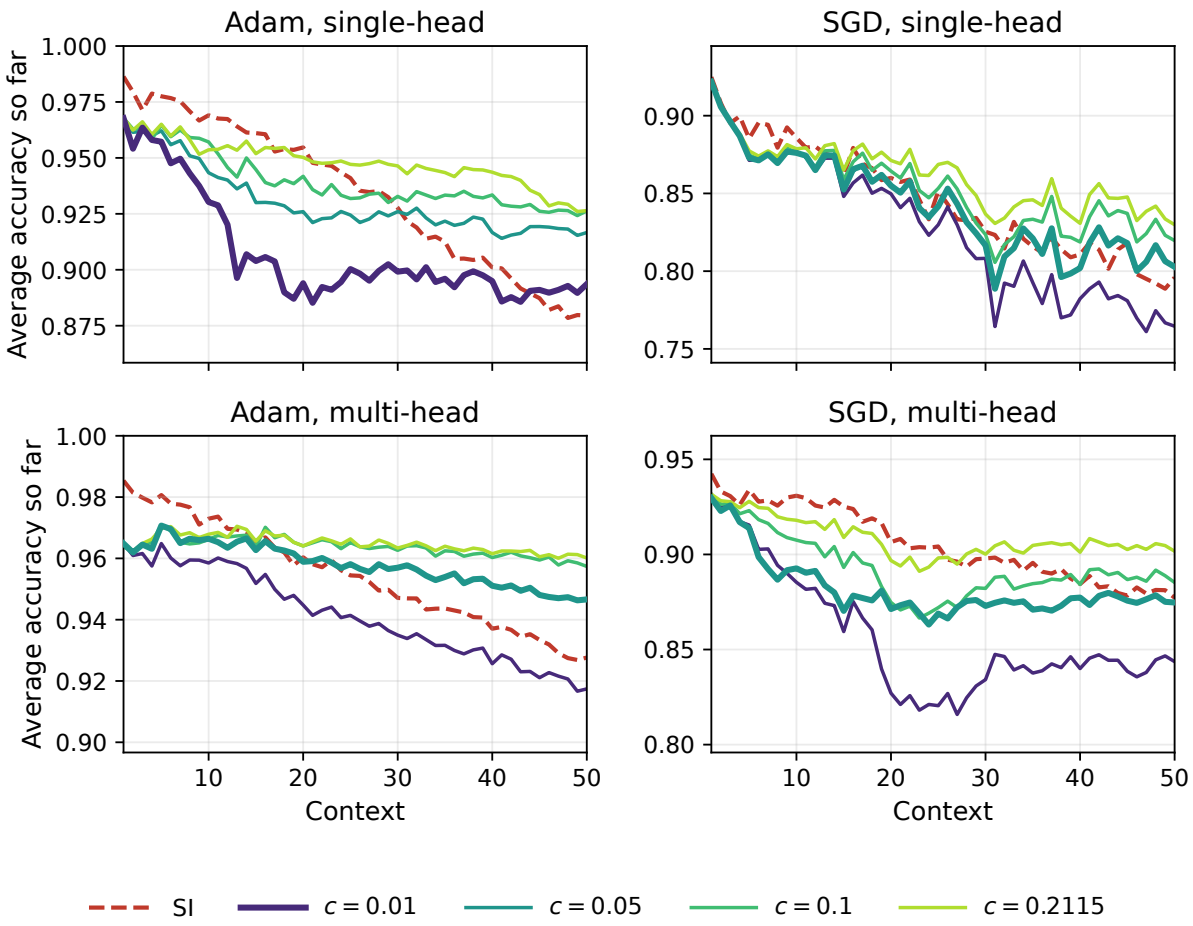


Figure 20: XdG+SI hybrid sweep (seed 1) at the locked gating fraction g. Average accuracy over contexts seen so far. The dashed line is the locked SI run in that panel. Selected c is the lightest SI penalty that remains competitive with SI.

A.5 Matched comparison summaries for task-IL

The main text reports domain-IL (single-head) matched comparison summaries in Tables 5–6. The tables below give the same metrics under task-IL (multi-head). XdG alone is omitted throughout: it is a single-seed diagnostic and is not performance-matched to SI/ER (Figure 19).

Table 12: Matched comparison summary (Adam, multi-head): mean ± 95% CI. Forgetting is peak accuracy−final accuracy on reference context 1.

Strategy	Mean acc	Forgetting	CKA	Procrustes	Param
naive	0.692±0.004	0.698±0.035	0.344±0.014	0.115±0.001	0.802±0.000
SI	0.925±0.001	0.030±0.006	0.053±0.005	0.044±0.001	0.474±0.001
ER	0.933±0.001	0.060±0.004	0.234±0.016	0.092±0.002	0.757±0.001
XdG+SI	0.950±0.002	0.048±0.011	0.025±0.003	0.031±0.001	0.733±0.001

Table 13: Matched comparison summary (SGD, multi-head): mean ± 95% CI. Forgetting is peak accuracy−final accuracy on reference context 1.

Strategy	Mean acc	Forgetting	CKA	Procrustes	Param
naive	0.675±0.008	0.499±0.034	0.235±0.023	0.097±0.003	0.371±0.000
SI	0.877±0.002	0.079±0.013	0.062±0.005	0.050±0.002	0.367±0.000
ER	0.876±0.001	0.044±0.005	0.105±0.009	0.067±0.002	0.337±0.000
XdG+SI	0.868±0.003	0.288±0.043	0.074±0.007	0.052±0.002	0.398±0.000

A.6 Locked-setting trajectories for the remaining combinations

The main text shows performance, CKA/Procrustes, and cosine-distance trajectories only for Adam single-head (Figures 2–5). The same 15-seed locked comparisons for SGD single-head and both task-IL combinations follow, grouped by combination.

SGD, single-head. Naive fine-tuning is less catastrophic than under Adam, but SI and ER still retain the reference task more strongly (Figure 21). Representational and cosine trajectories are in Figures 22 and 23.

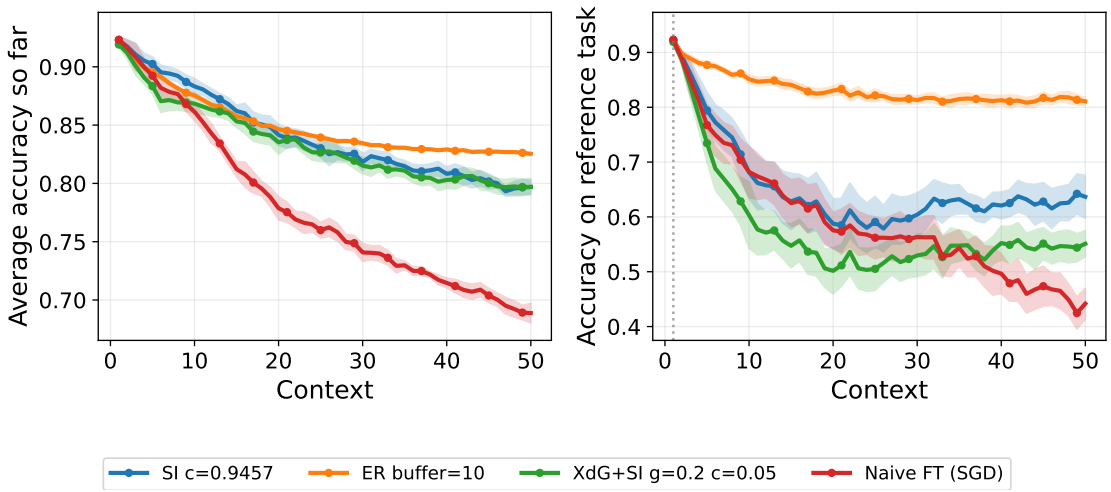


Figure 21: Final method comparison at locked hyperparameters (SGD, single-head, 15 seeds, mean ± 95% CI): naive fine-tuning, SI, ER, and XdG+SI. Left: mean accuracy over contexts seen so far. Right: reference-task (context 1) accuracy.

Adam, multi-head. Under task-IL all matched strategies retain the reference task (Figure 24). Drift and cosine panels are Figures 25 and 26.

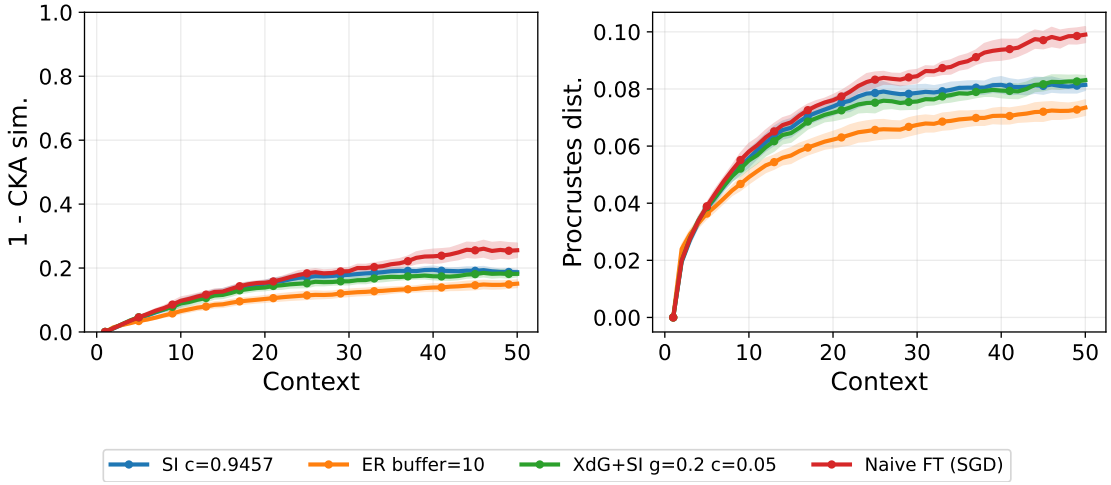


Figure 22: Representational-change trajectories (SGD, single-head, 15 seeds, mean ± 95% CI, reference context 1). Left: linear CKA change (1 − CKA). Right: orthogonal Procrustes distance.

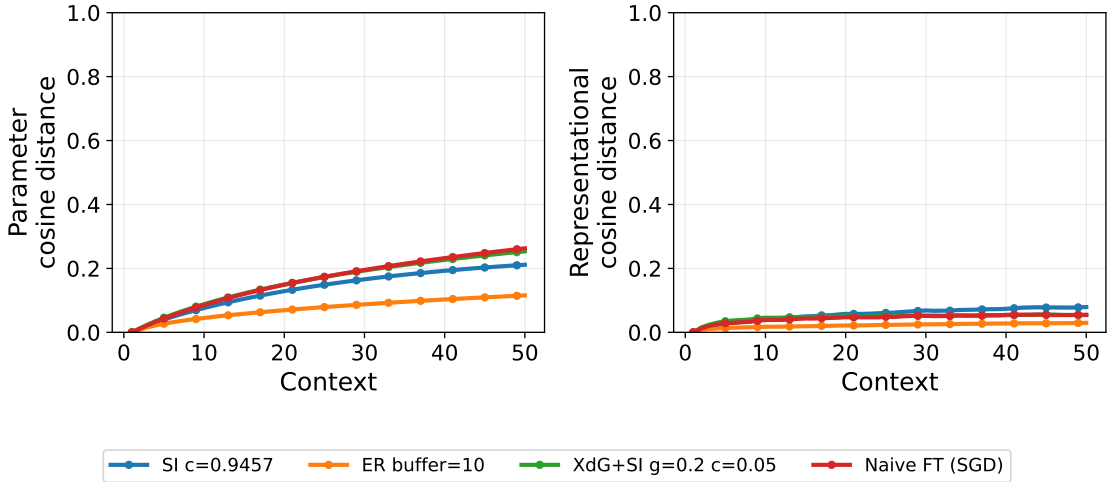


Figure 23: Parameter versus representational cosine distance (SGD, single-head, 15 seeds, mean ± 95% CI, reference context 1). Left: change in the flattened weight vector. Right: change in the mean hidden activation for reference-task images.

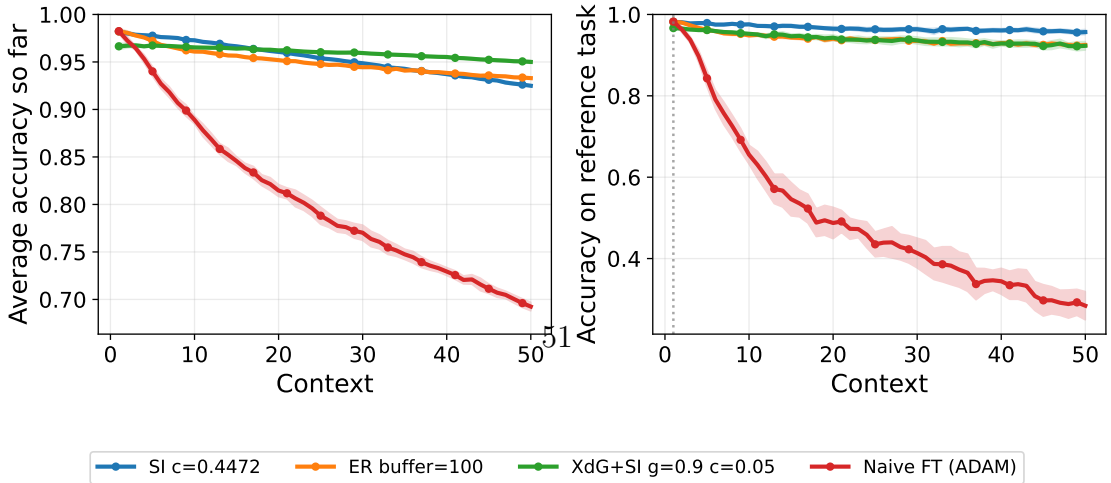


Figure 24: Final method comparison at locked hyperparameters (Adam, multi-head, 15 seeds, mean ± 95% CI): naive fine-tuning, SI, ER, and XdG+SI. Left: mean accuracy over contexts seen so far. Right: reference-task (context 1) accuracy.

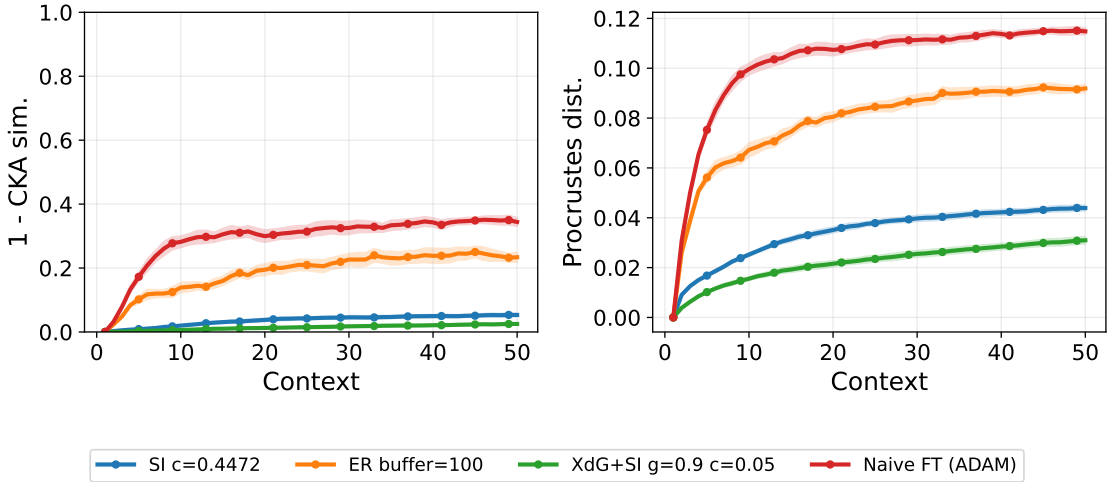


Figure 25: Representational-change trajectories (Adam, multi-head, 15 seeds, mean ± 95% CI, reference context 1). Left: linear CKA change (1 − CKA). Right: orthogonal Procrustes distance.

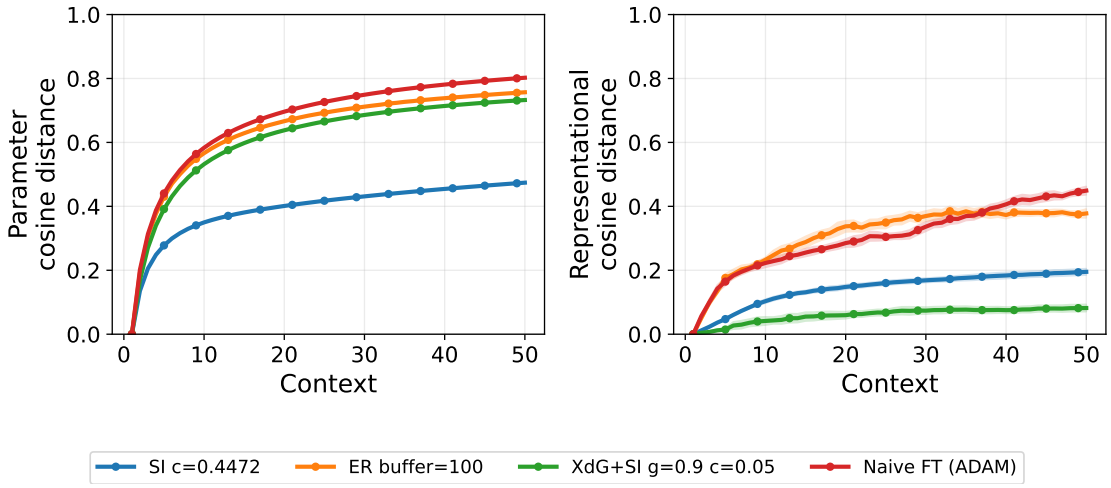


Figure 26: Parameter versus representational cosine distance (Adam, multi-head, 15 seeds, mean ± 95% CI, reference context 1). Left: change in the flattened weight vector. Right: change in the mean hidden activation for reference-task images.

SGD, multi-head. Task-IL under SGD likewise preserves reference-task accuracy for the matched methods (Figure 27). The corresponding geometry and cosine traces are Figures 28 and 29.

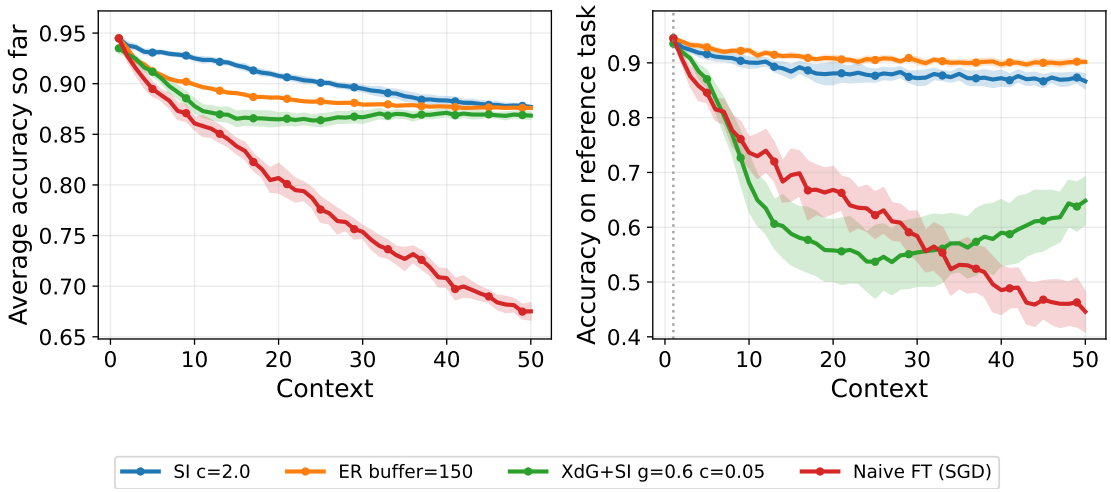


Figure 27: Final method comparison at locked hyperparameters (SGD, multi-head, 15 seeds, mean ± 95% CI): naive fine-tuning, SI, ER, and XdG+SI. Left: mean accuracy over contexts seen so far. Right: reference-task (context 1) accuracy.

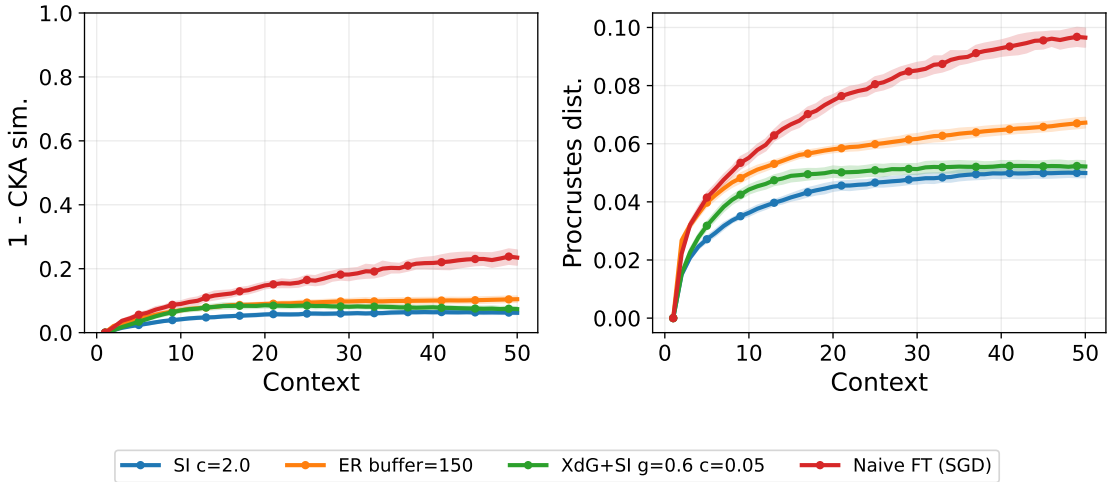


Figure 28: Representational-change trajectories (SGD, multi-head, 15 seeds, mean ± 95% CI, reference context 1). Left: linear CKA change (1 − CKA). Right: orthogonal Procrustes distance.

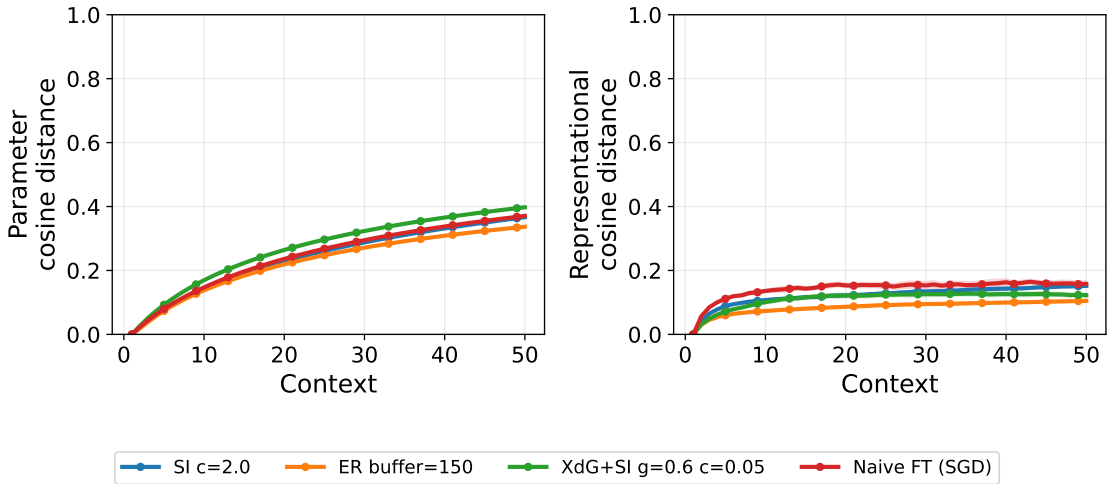


Figure 29: Parameter versus representational cosine distance (SGD, multi-head, 15 seeds, mean ± 95% CI, reference context 1). Left: change in the flattened weight vector. Right: change in the mean hidden activation for reference-task images.

A.7 Matched-horizon drift by reference task

Figure 4 plots CKA after 15 subsequent contexts for all four combinations of optimiser and architecture. The tables below give the corresponding means and 95% confidence intervals. The largest difference between early and late reference-task CKA in the suite is Adam multi-head naive fine-tuning ($\Delta_{1-25} = 0.23$).

Table 14: CKA change after 15 subsequent contexts (Adam, single-head): mean ± 95% CI. Each reference is evaluated at context $r + 15$ so remaining stream length is matched. Δ_{1-25} is ref 1 minus ref 25 (positive: earlier-task representations change more).

Strategy	Ref 1	Ref 2	Ref 10	Ref 25	Δ_{1-25}
naive	0.463±0.036	0.440±0.033	0.311±0.029	0.318±0.022	0.146±0.038
SI	0.055±0.005	0.040±0.004	0.072±0.009	0.091±0.015	-0.036±0.017
ER	0.207±0.015	0.185±0.017	0.151±0.012	0.168±0.021	0.039±0.029
XdG+SI	0.031±0.004	0.025±0.004	0.022±0.003	0.022±0.003	0.010±0.004

Table 15: CKA change after 15 subsequent contexts (SGD, single-head): mean ± 95% CI. Each reference is evaluated at context $r + 15$ so remaining stream length is matched. Δ_{1-25} is ref 1 minus ref 25 (positive: earlier-task representations change more).

Strategy	Ref 1	Ref 2	Ref 10	Ref 25	Δ_{1-25}
naive	0.135±0.008	0.175±0.011	0.180±0.012	0.160±0.018	-0.025±0.017
SI	0.132±0.010	0.144±0.008	0.103±0.010	0.060±0.007	0.072±0.013
ER	0.091±0.009	0.081±0.007	0.058±0.006	0.042±0.005	0.049±0.010
XdG+SI	0.122±0.010	0.111±0.009	0.106±0.011	0.096±0.010	0.027±0.016

Table 16: CKA change after 15 subsequent contexts (Adam, multi-head): mean ± 95% CI. Each reference is evaluated at context $r + 15$ so remaining stream length is matched. Δ_{1-25} is ref 1 minus ref 25 (positive: earlier-task representations change more).

Strategy	Ref 1	Ref 2	Ref 10	Ref 25	Δ_{1-25}
naive	0.312±0.020	0.256±0.020	0.146±0.009	0.083±0.007	0.229±0.022
SI	0.032±0.004	0.033±0.003	0.043±0.005	0.038±0.004	-0.005±0.005
ER	0.175±0.016	0.139±0.009	0.119±0.011	0.090±0.010	0.085±0.018
XdG+SI	0.011±0.002	0.008±0.001	0.008±0.001	0.007±0.001	0.004±0.002

Table 17: CKA change after 15 subsequent contexts (SGD, multi-head): mean ± 95% CI. Each reference is evaluated at context $r + 15$ so remaining stream length is matched. Δ_{1-25} is ref 1 minus ref 25 (positive: earlier-task representations change more).

Strategy	Ref 1	Ref 2	Ref 10	Ref 25	Δ_{1-25}
naive	0.122±0.011	0.126±0.012	0.149±0.016	0.136±0.014	-0.013±0.016
SI	0.052±0.004	0.045±0.003	0.037±0.004	0.018±0.002	0.033±0.005
ER	0.086±0.005	0.060±0.006	0.047±0.005	0.041±0.004	0.045±0.005
XdG+SI	0.084±0.009	0.055±0.006	0.021±0.003	0.022±0.004	0.062±0.010

A.8 Loss-of-plasticity diagnostics under task-IL

The main text reports domain-IL LoP diagnostics in Tables 8–9. The tables below give the same quantities under task-IL (multi-head). Multi-head Adam is the high-rank regime (naive/ER effective rank ≈ 70). Multi-head SGD ER is the largest accuracy-based LoP drop in the strategy comparison (0.11).

Table 18: Loss-of-plasticity diagnostics (Adam, multi-head): mean ± 95% CI. Early learnability is current-task accuracy on contexts 1–5; late learnability is contexts 41–50; LoP drop is early minus late.

Strategy	Early learn.	Late learn.	LoP drop	Dead units	Eff. rank	Mean $\|w\|$
naive	0.982±0.001	0.978±0.001	0.004±0.001	0.067±0.006	69.6±1.2	0.137±0.000
SI	0.979±0.001	0.941±0.001	0.038±0.002	0.001±0.000	30.2±0.9	0.049±0.000
ER	0.981±0.001	0.973±0.000	0.008±0.001	0.035±0.003	73.7±1.3	0.144±0.000
XdG+SI	0.969±0.001	0.968±0.001	0.001±0.002	0.704±0.020	10.5±0.1	0.060±0.000

Table 19: Loss-of-plasticity diagnostics (SGD, multi-head): mean ± 95% CI. Early learnability is current-task accuracy on contexts 1–5; late learnability is contexts 41–50; LoP drop is early minus late.

Strategy	Early learn.	Late learn.	LoP drop	Dead units	Eff. rank	Mean $\|w\|$
naive	0.947±0.001	0.966±0.001	-0.018±0.001	0.002±0.001	13.5±0.3	0.028±0.000
SI	0.946±0.001	0.893±0.002	0.053±0.003	0.006±0.001	15.1±0.3	0.026±0.000
ER	0.922±0.002	0.811±0.001	0.111±0.002	0.024±0.002	11.8±0.4	0.028±0.000
XdG+SI	0.936±0.001	0.947±0.001	-0.011±0.002	0.135±0.005	10.2±0.2	0.028±0.000

A.9 Full induced-change Δ table

Table 20: Induced-drift interventions vs no-noise baseline (Adam, single-head). Mean ± 95% CI.

Strategy	Noise	ΔLate	ΔF	ΔCKA
ER	gradient noise $\sigma = 0.001$	-0.040±0.001	0.035±0.004	-0.032±0.012
ER	gradient noise $\sigma = 0.003$	-0.060±0.002	0.073±0.006	-0.051±0.008
ER	gradient noise $\sigma = 0.01$	-0.086±0.002	0.091±0.008	-0.078±0.011
ER	gradient noise $\sigma = 0.03$	-0.127±0.002	0.087±0.006	-0.082±0.009
ER	gradient noise $\sigma = 0.1$	-0.211±0.003	0.080±0.010	-0.098±0.009
ER	gradient noise $\sigma = 0.3$	-0.370±0.003	0.095±0.010	-0.116±0.009
ER	gradient noise $\sigma = 1$	-0.633±0.008	0.218±0.014	-0.139±0.011
ER	gradient noise $\sigma = 3$	-0.785±0.005	0.374±0.018	-0.079±0.012
ER	gradient noise $\sigma = 10$	-0.785±0.005	0.398±0.021	0.283±0.038
ER	gradient noise $\sigma = 30$	-0.789±0.004	0.380±0.015	0.291±0.030
ER	synaptic turnover $f = 0.001$	-0.000±0.001	0.002±0.004	-0.011±0.028
ER	synaptic turnover $f = 0.002$	0.000±0.001	0.000±0.005	-0.013±0.032
ER	synaptic turnover $f = 0.005$	-0.001±0.001	-0.001±0.004	-0.007±0.025
ER	synaptic turnover $f = 0.01$	-0.002±0.001	0.005±0.003	-0.021±0.023
ER	synaptic turnover $f = 0.02$	-0.003±0.001	0.024±0.013	-0.018±0.023
ER	synaptic turnover $f = 0.05$	-0.007±0.001	0.027±0.006	-0.023±0.018
ER	synaptic turnover $f = 0.1$	-0.007±0.001	0.028±0.008	-0.037±0.013
ER	synaptic turnover $f = 0.2$	-0.004±0.001	0.028±0.008	-0.016±0.013
ER	synaptic turnover $f = 0.35$	-0.005±0.001	0.028±0.004	0.001±0.016
ER	synaptic turnover $f = 0.5$	-0.008±0.001	0.034±0.007	0.004±0.013
ER	targeted dropout $A = 1$	-0.003±0.001	-0.001±0.004	-0.035±0.014
ER	targeted dropout $A = 2$	-0.003±0.001	0.002±0.006	-0.049±0.028
ER	targeted dropout $A = 5$	-0.003±0.001	0.002±0.006	-0.057±0.018
ER	targeted dropout $A = 10$	-0.006±0.001	-0.001±0.005	-0.056±0.022
ER	targeted dropout $A = 20$	-0.018±0.001	-0.007±0.005	-0.076±0.018
ER	targeted dropout $A = 40$	-0.035±0.001	-0.006±0.003	-0.087±0.015
ER	targeted dropout $A = 80$	-0.049±0.002	-0.008±0.004	-0.048±0.020
ER	targeted dropout $A = 160$	-0.065±0.002	-0.011±0.005	-0.022±0.031
ER	targeted dropout $A = 320$	-0.064±0.003	-0.011±0.004	-0.099±0.025
ER	targeted dropout $A = 640$	-0.056±0.002	-0.032±0.006	-0.120±0.020
ER	uniform dropout $p = 0.05$	-0.004±0.001	-0.003±0.004	-0.054±0.024
ER	uniform dropout $p = 0.1$	-0.006±0.001	-0.005±0.005	-0.071±0.014
ER	uniform dropout $p = 0.2$	-0.010±0.001	-0.010±0.005	-0.086±0.016
ER	uniform dropout $p = 0.3$	-0.015±0.001	-0.011±0.004	-0.087±0.016
ER	uniform dropout $p = 0.4$	-0.023±0.001	-0.015±0.003	-0.104±0.012
ER	uniform dropout $p = 0.5$	-0.034±0.001	-0.012±0.005	-0.113±0.013
ER	uniform dropout $p = 0.6$	-0.050±0.002	-0.015±0.003	-0.095±0.020

Continued on next page

Table 20: Induced-drift interventions vs no-noise baseline (Adam, single-head). Mean ± 95% CI. (continued)

Strategy	Noise	ΔLate	ΔF	ΔCKA
ER	uniform dropout $p = 0.7$	-0.079±0.002	-0.010±0.003	-0.114±0.010
ER	uniform dropout $p = 0.8$	-0.151±0.005	-0.009±0.004	-0.103±0.015
ER	uniform dropout $p = 0.9$	-0.398±0.009	-0.012±0.004	-0.068±0.024
SI	gradient noise $\sigma = 0.001$	-0.021±0.001	-0.031±0.019	0.122±0.017
SI	gradient noise $\sigma = 0.003$	-0.044±0.001	0.032±0.017	0.032±0.011
SI	gradient noise $\sigma = 0.01$	-0.086±0.002	0.111±0.022	-0.057±0.003
SI	gradient noise $\sigma = 0.03$	-0.139±0.003	0.147±0.026	-0.064±0.004
SI	gradient noise $\sigma = 0.1$	-0.269±0.003	0.041±0.029	-0.077±0.004
SI	gradient noise $\sigma = 0.3$	-0.490±0.006	-0.030±0.018	-0.090±0.004
SI	gradient noise $\sigma = 1$	-0.680±0.006	0.047±0.021	-0.102±0.003
SI	gradient noise $\sigma = 3$	-0.732±0.006	0.141±0.021	-0.105±0.002
SI	gradient noise $\sigma = 10$	-0.756±0.005	0.183±0.022	-0.103±0.002
SI	gradient noise $\sigma = 30$	-0.761±0.005	0.186±0.026	-0.103±0.003
SI	synaptic turnover $f = 0.001$	-0.000±0.001	0.038±0.039	0.015±0.015
SI	synaptic turnover $f = 0.002$	0.001±0.001	0.021±0.029	0.020±0.016
SI	synaptic turnover $f = 0.005$	0.000±0.001	0.073±0.032	0.045±0.015
SI	synaptic turnover $f = 0.01$	0.000±0.001	0.173±0.031	0.088±0.015
SI	synaptic turnover $f = 0.02$	0.001±0.001	0.378±0.035	0.201±0.023
SI	synaptic turnover $f = 0.05$	0.003±0.001	0.585±0.023	0.417±0.060
SI	synaptic turnover $f = 0.1$	0.003±0.001	0.618±0.015	0.531±0.046
SI	synaptic turnover $f = 0.2$	0.003±0.001	0.620±0.011	0.501±0.036
SI	synaptic turnover $f = 0.35$	-0.005±0.002	0.627±0.011	0.379±0.050
SI	synaptic turnover $f = 0.5$	-0.046±0.005	0.618±0.010	0.501±0.078
SI	targeted dropout $A = 1$	-0.000±0.001	-0.008±0.024	0.002±0.009
SI	targeted dropout $A = 2$	0.000±0.001	0.008±0.033	0.001±0.013
SI	targeted dropout $A = 5$	-0.001±0.001	0.007±0.032	0.002±0.008
SI	targeted dropout $A = 10$	-0.001±0.001	0.031±0.031	-0.002±0.009
SI	targeted dropout $A = 20$	-0.002±0.001	0.028±0.033	0.007±0.014
SI	targeted dropout $A = 40$	-0.005±0.001	0.049±0.037	-0.001±0.014
SI	targeted dropout $A = 80$	-0.010±0.001	0.046±0.031	0.012±0.014
SI	targeted dropout $A = 160$	-0.023±0.002	0.038±0.031	-0.008±0.014
SI	targeted dropout $A = 320$	-0.060±0.005	-0.058±0.028	-0.042±0.012
SI	targeted dropout $A = 640$	-0.164±0.019	-0.127±0.019	-0.074±0.012
SI	uniform dropout $p = 0.05$	-0.000±0.001	0.004±0.022	0.007±0.018
SI	uniform dropout $p = 0.1$	-0.000±0.001	0.017±0.028	0.011±0.019
SI	uniform dropout $p = 0.2$	-0.002±0.001	0.059±0.031	0.009±0.014
SI	uniform dropout $p = 0.3$	-0.004±0.001	0.064±0.022	0.004±0.012
SI	uniform dropout $p = 0.4$	-0.007±0.001	0.048±0.038	0.006±0.015
SI	uniform dropout $p = 0.5$	-0.010±0.001	0.031±0.025	0.004±0.014

Continued on next page

Table 20: Induced-drift interventions vs no-noise baseline (Adam, single-head). Mean ± 95% CI. (continued)

Strategy	Noise	ΔLate	ΔF	ΔCKA
SI	uniform dropout $p = 0.6$	-0.015±0.001	0.018±0.025	-0.006±0.012
SI	uniform dropout $p = 0.7$	-0.023±0.001	0.020±0.020	-0.002±0.017
SI	uniform dropout $p = 0.8$	-0.038±0.001	0.032±0.026	0.005±0.017
SI	uniform dropout $p = 0.9$	-0.075±0.002	-0.043±0.017	0.035±0.028
XdG+SI	gradient noise $\sigma = 0.001$	-0.019±0.001	0.017±0.066	0.002±0.004
XdG+SI	gradient noise $\sigma = 0.003$	-0.044±0.001	0.119±0.049	0.026±0.010
XdG+SI	gradient noise $\sigma = 0.01$	-0.086±0.001	0.138±0.060	0.031±0.011
XdG+SI	gradient noise $\sigma = 0.03$	-0.142±0.002	0.059±0.057	0.011±0.005
XdG+SI	gradient noise $\sigma = 0.1$	-0.288±0.005	0.101±0.058	0.006±0.006
XdG+SI	gradient noise $\sigma = 0.3$	-0.572±0.005	0.051±0.034	0.117±0.026
XdG+SI	gradient noise $\sigma = 1$	-0.738±0.007	-0.152±0.032	0.254±0.015
XdG+SI	gradient noise $\sigma = 3$	-0.764±0.008	-0.210±0.015	0.239±0.024
XdG+SI	gradient noise $\sigma = 10$	-0.771±0.006	-0.204±0.022	0.252±0.020
XdG+SI	gradient noise $\sigma = 30$	-0.768±0.007	-0.221±0.016	0.276±0.023
XdG+SI	synaptic turnover $f = 0.001$	-0.000±0.001	0.014±0.040	-0.004±0.004
XdG+SI	synaptic turnover $f = 0.002$	-0.000±0.001	0.022±0.037	-0.002±0.004
XdG+SI	synaptic turnover $f = 0.005$	-0.000±0.001	0.029±0.055	0.000±0.005
XdG+SI	synaptic turnover $f = 0.01$	0.000±0.001	0.014±0.045	0.001±0.005
XdG+SI	synaptic turnover $f = 0.02$	-0.000±0.001	0.063±0.054	0.004±0.004
XdG+SI	synaptic turnover $f = 0.05$	0.000±0.001	0.140±0.055	0.014±0.006
XdG+SI	synaptic turnover $f = 0.1$	0.000±0.001	0.251±0.060	0.044±0.010
XdG+SI	synaptic turnover $f = 0.2$	0.000±0.001	0.418±0.050	0.131±0.020
XdG+SI	synaptic turnover $f = 0.35$	0.001±0.001	0.543±0.017	0.376±0.051
XdG+SI	synaptic turnover $f = 0.5$	0.001±0.001	0.586±0.006	0.484±0.025
XdG+SI	targeted dropout $A = 1$	-0.000±0.001	0.013±0.044	-0.001±0.004
XdG+SI	targeted dropout $A = 2$	-0.000±0.001	0.017±0.043	-0.001±0.004
XdG+SI	targeted dropout $A = 5$	-0.001±0.001	0.024±0.043	-0.000±0.004
XdG+SI	targeted dropout $A = 10$	-0.001±0.001	-0.024±0.030	0.002±0.007
XdG+SI	targeted dropout $A = 20$	-0.005±0.001	-0.090±0.021	0.002±0.004
XdG+SI	targeted dropout $A = 40$	-0.022±0.002	-0.136±0.028	-0.003±0.007
XdG+SI	targeted dropout $A = 80$	-0.064±0.003	-0.163±0.054	-0.015±0.004
XdG+SI	targeted dropout $A = 160$	-0.117±0.003	-0.116±0.044	-0.026±0.002
XdG+SI	targeted dropout $A = 320$	-0.235±0.011	0.017±0.070	-0.017±0.006
XdG+SI	targeted dropout $A = 640$	-0.274±0.009	0.046±0.052	-0.015±0.010
XdG+SI	uniform dropout $p = 0.05$	0.001±0.001	-0.086±0.024	-0.001±0.004
XdG+SI	uniform dropout $p = 0.1$	-0.000±0.001	-0.143±0.030	-0.000±0.005
XdG+SI	uniform dropout $p = 0.2$	-0.002±0.001	-0.151±0.030	0.003±0.007
XdG+SI	uniform dropout $p = 0.3$	-0.005±0.001	-0.202±0.021	0.003±0.005
XdG+SI	uniform dropout $p = 0.4$	-0.010±0.001	-0.209±0.021	0.001±0.005

Continued on next page

Table 20: Induced-drift interventions vs no-noise baseline (Adam, single-head). Mean ± 95% CI. (continued)

Strategy	Noise	ΔLate	ΔF	ΔCKA
XdG+SI	uniform dropout $p = 0.5$	-0.016±0.001	-0.220±0.013	0.005±0.008
XdG+SI	uniform dropout $p = 0.6$	-0.025±0.001	-0.225±0.012	0.009±0.010
XdG+SI	uniform dropout $p = 0.7$	-0.038±0.001	-0.178±0.018	0.014±0.010
XdG+SI	uniform dropout $p = 0.8$	-0.061±0.002	-0.171±0.016	-0.001±0.006
XdG+SI	uniform dropout $p = 0.9$	-0.247±0.018	-0.048±0.036	-0.015±0.005
naive	gradient noise $\sigma = 0.001$	-0.006±0.001	-0.012±0.012	-0.049±0.028
naive	gradient noise $\sigma = 0.003$	-0.010±0.001	-0.011±0.011	-0.086±0.025
naive	gradient noise $\sigma = 0.01$	-0.021±0.001	-0.008±0.011	-0.172±0.033
naive	gradient noise $\sigma = 0.03$	-0.036±0.001	-0.034±0.015	-0.262±0.024
naive	gradient noise $\sigma = 0.1$	-0.059±0.002	-0.078±0.018	-0.343±0.017
naive	gradient noise $\sigma = 0.3$	-0.083±0.002	-0.089±0.015	-0.375±0.011
naive	gradient noise $\sigma = 1$	-0.122±0.002	-0.156±0.013	-0.417±0.014
naive	gradient noise $\sigma = 3$	-0.182±0.003	-0.232±0.018	-0.452±0.007
naive	gradient noise $\sigma = 10$	-0.294±0.006	-0.303±0.023	-0.447±0.011
naive	gradient noise $\sigma = 30$	-0.426±0.005	-0.337±0.019	-0.326±0.027
naive	synaptic turnover $f = 0.001$	-0.000±0.001	-0.008±0.011	0.012±0.024
naive	synaptic turnover $f = 0.002$	0.000±0.001	-0.000±0.010	0.022±0.035
naive	synaptic turnover $f = 0.005$	-0.000±0.001	-0.013±0.015	0.001±0.042
naive	synaptic turnover $f = 0.01$	0.000±0.001	-0.003±0.011	0.001±0.023
naive	synaptic turnover $f = 0.02$	0.000±0.001	0.003±0.008	0.011±0.028
naive	synaptic turnover $f = 0.05$	0.002±0.001	0.004±0.010	0.014±0.019
naive	synaptic turnover $f = 0.1$	0.003±0.001	0.005±0.006	0.009±0.022
naive	synaptic turnover $f = 0.2$	0.003±0.001	0.004±0.011	-0.012±0.030
naive	synaptic turnover $f = 0.35$	0.003±0.001	0.006±0.009	-0.019±0.020
naive	synaptic turnover $f = 0.5$	0.002±0.001	0.004±0.011	-0.001±0.024
naive	targeted dropout $A = 1$	0.000±0.001	-0.007±0.010	0.004±0.028
naive	targeted dropout $A = 2$	0.000±0.001	-0.009±0.014	0.020±0.031
naive	targeted dropout $A = 5$	0.000±0.001	-0.011±0.013	-0.008±0.016
naive	targeted dropout $A = 10$	0.001±0.001	-0.002±0.010	-0.005±0.033
naive	targeted dropout $A = 20$	0.001±0.001	-0.014±0.015	-0.087±0.021
naive	targeted dropout $A = 40$	0.002±0.001	-0.018±0.011	-0.099±0.022
naive	targeted dropout $A = 80$	0.001±0.001	0.003±0.012	-0.082±0.031
naive	targeted dropout $A = 160$	-0.009±0.001	-0.003±0.014	-0.076±0.029
naive	targeted dropout $A = 320$	-0.024±0.002	-0.013±0.012	-0.094±0.022
naive	targeted dropout $A = 640$	-0.083±0.005	-0.047±0.018	-0.039±0.043
naive	uniform dropout $p = 0.05$	0.001±0.001	-0.014±0.013	0.005±0.029
naive	uniform dropout $p = 0.1$	0.002±0.001	-0.019±0.012	-0.021±0.020
naive	uniform dropout $p = 0.2$	0.002±0.001	-0.007±0.010	-0.047±0.017
naive	uniform dropout $p = 0.3$	0.000±0.001	-0.010±0.012	-0.074±0.032

Continued on next page

Table 20: Induced-drift interventions vs no-noise baseline (Adam, single-head). Mean ± 95% CI. (continued)

Strategy	Noise	ΔLate	ΔF	ΔCKA
naive	uniform dropout $p = 0.4$	-0.001±0.001	0.004±0.011	-0.054±0.022
naive	uniform dropout $p = 0.5$	-0.005±0.001	-0.008±0.011	-0.030±0.023
naive	uniform dropout $p = 0.6$	-0.010±0.001	-0.011±0.012	-0.027±0.025
naive	uniform dropout $p = 0.7$	-0.022±0.001	-0.012±0.011	0.008±0.026
naive	uniform dropout $p = 0.8$	-0.052±0.002	-0.007±0.009	0.036±0.035
naive	uniform dropout $p = 0.9$	-0.182±0.006	-0.029±0.017	0.040±0.026

Table 21: Best induced-drift setting per strategy under Δforgetting ≤ 0.02 (else least harmful).

Strategy	Best noise	ΔLate	ΔF	ΔCKA
ER	synaptic turnover $f = 0.002$	+0.000	+0.000	-0.013
SI	targeted dropout $A = 2$	+0.000	+0.008	+0.001
XdG+SI	uniform dropout $p = 0.05$	+0.001	-0.086	-0.001
naive	synaptic turnover $f = 0.35$	+0.003	+0.006	-0.019

A.10 Induced-change dead-unit and rank curves

The main text reports Δforgetting and ΔCKA for each noise family (Figures 9–12). The figures below show Δdead-unit fraction and Δeffective rank versus the no-noise baseline. A useful intervention lowers dead-unit fraction while Δforgetting stays at or below zero.

Uniform dropout revives silent units for naive and ER at mild p, while SI is already unsaturated (Figure 30).

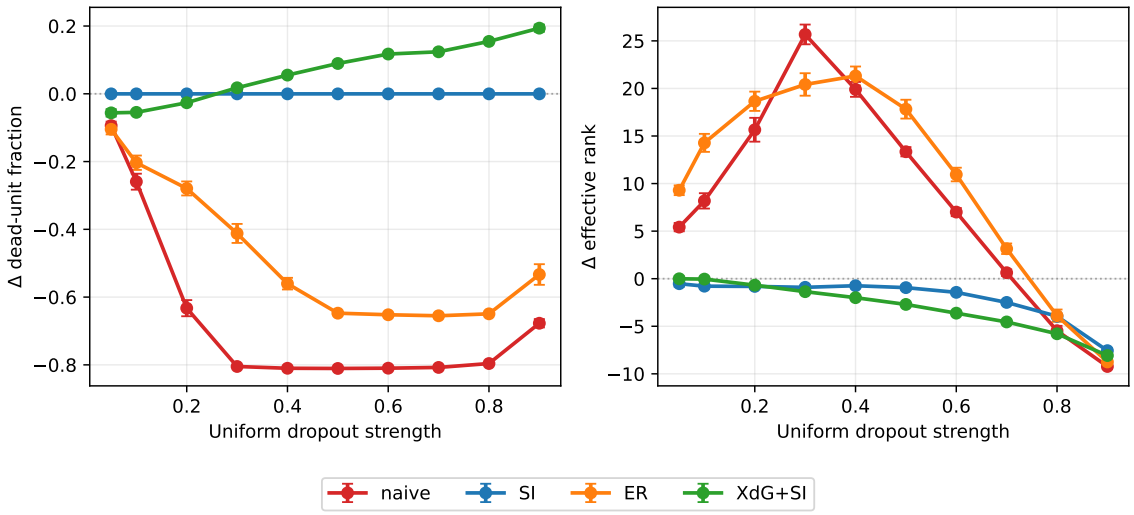


Figure 30: Uniform dropout: change versus the no-noise baseline (Adam, single-head). Left: Δdead-unit fraction. Right: Δeffective rank. Naive and ER revive silent units at mild p; SI is already unsaturated.

Targeted dropout follows the same pattern at large A (Figure 31).

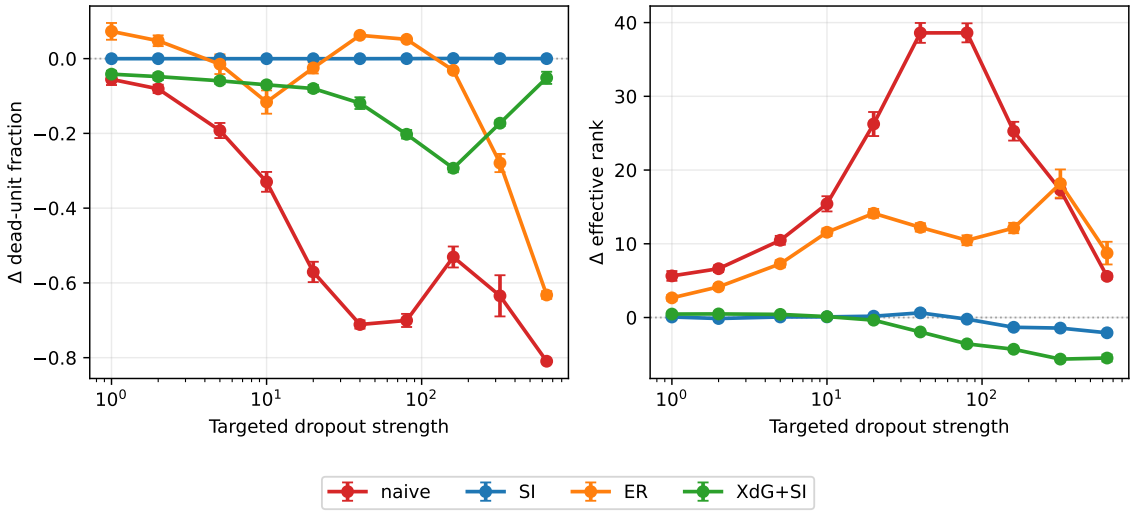


Figure 31: Targeted dropout: Δdead-unit fraction and Δeffective rank versus the no-noise baseline. Naive and ER recover dead units at large A; SI remains flat.

Gradient noise can inflate effective rank by tens of units, but those rank gains co-occur with collapsed late learnability (Figure 32).

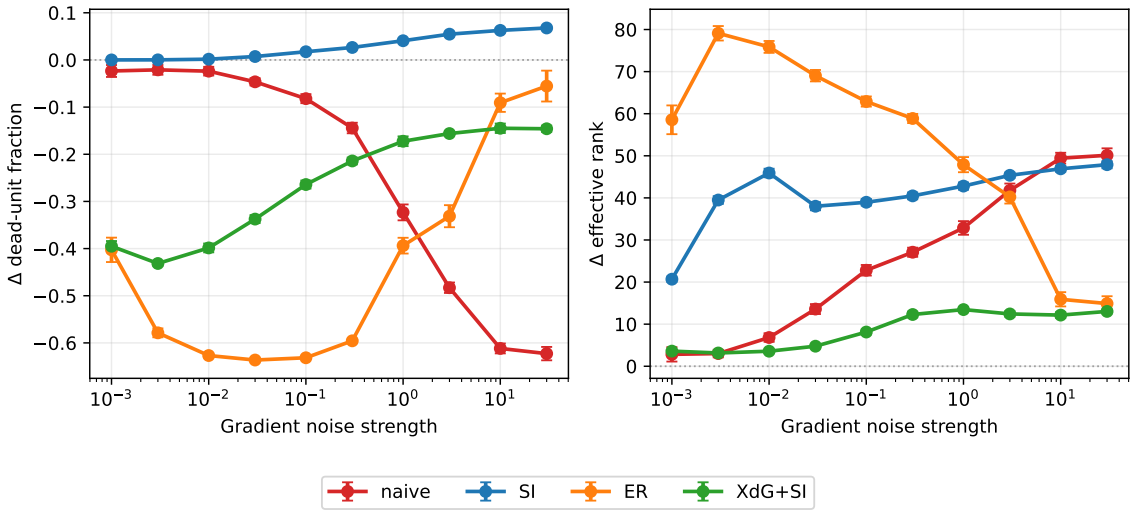


Figure 32: Gradient noise: Δdead-unit fraction and Δeffective rank versus the no-noise baseline. Rank can rise by tens of units while late learnability collapses; that rank inflation is not functional plasticity.

Synaptic turnover lowers dead-unit fraction for naive and ER only at high f, where SI and XdG+SI have already lost retention (Figure 33).

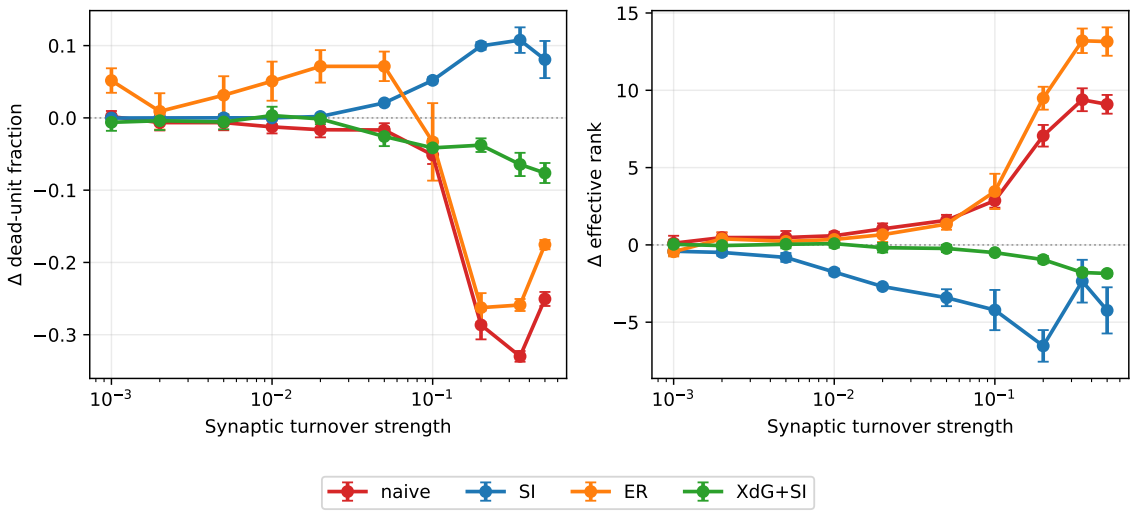


Figure 33: Synaptic turnover: Δdead-unit fraction and Δeffective rank versus the no-noise baseline. Naive and ER dead-unit fraction falls only at high f, where SI and XdG+SI have already lost retention.

Figure 34 summarises the same trade-off at the condition level: points with $\Delta\text{dead} < 0$ and $\Delta\text{forgetting} \leq 0$ recover dead units without added forgetting.

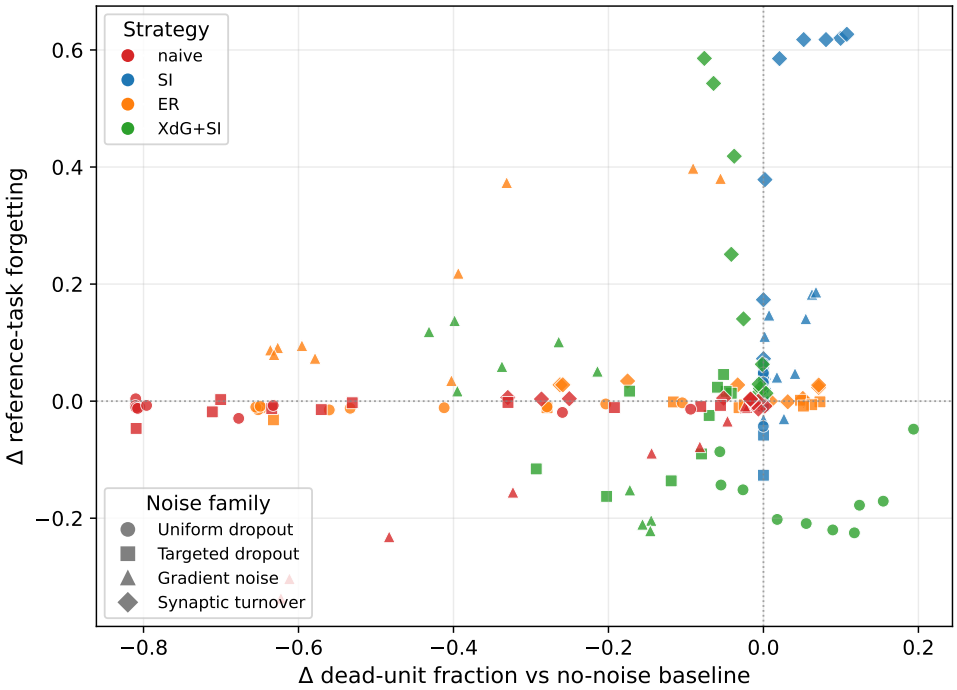


Figure 34: Condition-level coupling of Δdead-unit fraction and Δforgetting in the induced-change experiment (Adam, single-head; means for 15 seeds). Points with Δdead < 0 and Δforgetting ≤ 0 recover dead units without added forgetting.