WHAT SHOULD THE BACKBONE LEARN? ANALYZING HETEROGENEOUS CONTINUAL PET VIA COMMIT-AND-REBASE

Anonymous authors
Paper under double-blind review

ABSTRACT

In continual learning with pre-trained models (PTMs), parameter-efficient tuning (PET) has emerged as a prevailing paradigm, typically allocating a small budget of tunable parameters to tasks. The budget bounds interference with the pre-trained representation, but it also bounds how much a stream can teach the model: whether the parameters are appended alongside the backbone or embedded in a low-rank slice of it, the knowledge of every task must fit within an allowance far smaller than the backbone itself. A stream long enough will exhaust any such allowance, leaving the backbone itself as the natural store, and this raises a fundamental question: of everything a PET module learns, what is actually worth writing into it? The question is hard to even pose, because different PET methods express what they learn in incompatible formats. In this paper, we propose CoRe, an instrument that converts what an arbitrary PET method has learned into a shared weight space through a commit-and-rebase process, so that heterogeneous methods become directly comparable. Across seven PET methods and seven benchmarks, CoRe reproduces the behavior of the original expert more faithfully than existing merging operators, with 97.5% prediction agreement. Analyzing the converted updates, we find that only a small component of each update is invariant across methods and that this component alone moves the decisions of its task, providing a first empirical account of what a backbone should learn from a PET module.

1 INTRODUCTION

Pre-trained models (PTMs) have substantially advanced class-incremental learning (CIL), largely driven by parameter-efficient tuning (PET), in which task knowledge is restricted to a small parameter budget (Wang et al., 2024a; Zhou et al., 2024a). Existing strategies distribute this budget differently: some maintain a shared parameter pool of fixed size (Wang et al., 2022b;a; Smith et al., 2023), forcing later tasks to compete with earlier ones; others allocate a dedicated PET module per task (Zhou et al., 2024b; Sun et al., 2025b), avoiding interference at the cost of an expanding memory footprint; and low-rank adaptation methods embed the budget within a low-rank subspace of the backbone (Hu et al., 2022; Liang & Li, 2024). Despite these structural variations, existing strategies trade one constraint for another. Fixed-capacity continual learners balance preserving earlier knowledge against retaining flexibility for new tasks, and accumulated protection constraints can impair subsequent learning (Kirkpatrick et al., 2017; Schwarz et al., 2018). Task-dedicated PET modules alleviate competition for shared adaptation parameters, but their storage requirements grow with the number of tasks (Zhou et al., 2024b; Sun et al., 2025b). These constraints motivate consolidating newly acquired knowledge into a shared backbone, following the broader continual-learning principle of transferring task-specific learning into a persistent knowledge base (Schwarz et al., 2018).

Consolidating knowledge into the backbone requires determining which components of a PET module's learned update are actually worth writing into it. This question is difficult even to formulate, because heterogeneous PET architectures express knowledge in mutually incompatible representations. Prompts modify attention through additional learnable tokens, adapters introduce nonlinear bottleneck transformations, and low-rank modules parameterize additive weight increments (Jia et al., 2022; Houlsby et al., 2019; Hu et al., 2022; He et al., 2022). As they stand, these updates can-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

not be directly compared, decomposed, or evaluated against one another. Resolving this challenge requires a common coordinate system, a unified weight-space format into which the knowledge of arbitrary PET methods can be mapped while the underlying learner continues to operate.

Two properties of PET make this conversion nontrivial. The first is representability. The effect of a prompt depends on the input through attention and admits no exact static weight-space equivalent (He et al., 2022; Dherin et al., 2025), and nonlinear adapters cannot be folded analytically (Luo et al., 2023). An approximate equivalent can nevertheless be constructed by distillation, matching the classifier-input representation and decision function of an end-of-task expert with a low-rank update on the attention projections (Wang et al., 2024b). The second property concerns the learner itself. A continual method keeps running on its own module state, and once a task's knowledge has been written into the backbone, the retained state applies the same knowledge again on top of it; left uncorrected, the redundancy grows with every conversion. Conversion therefore has to be paired with a realignment of the learner, and CoRe treats the two as separate operations.

To reconcile these incompatible formats without disrupting the running learner, we propose CoRe, a **Co**mmit-and-**Re**base instrument that converts what an arbitrary PET method has learned into a unified weight space. At each consolidation step, CoRe distills the representation and decision boundaries of the PET expert into a low-rank update on the backbone's attention projections. This update is folded in under a second-order regularizer that damps directions to which earlier decisions are sensitive. To prevent duplicate application, in which the retained state re-applies knowledge already written into the backbone, a temporary realignment module offsets this redundancy before being merged into the next base and discarded. By decoupling knowledge writing from state realignment, CoRe allows the knowledge held in the backbone to grow incrementally across steps without accumulating a bank of task modules, ultimately yielding a single consolidated backbone paired with the classifier.

We evaluate CoRe across seven representative PET methods and seven benchmarks, where heterogeneous methods learn identical task streams on a common PTM base. After verifying that the conversion faithfully preserves the decisions of the original expert, our analysis proceeds in three stages. First, we decompose the converted weight updates to separate method-specific components from a format-invariant component shared across methods. Second, we apply matched controls, including label permutation, to establish that this shared component represents a learned task signal rather than an artifact of shared data statistics. Third, we causally validate its functional role by transplanting it alone into the pre-trained backbone. The goal is to identify what portion of a PET update genuinely carries task knowledge, and to establish that this portion is a method-agnostic unit that consolidation can write and reuse.

Our contributions are summarized as follows.

- We propose the first approach that expresses the knowledge of heterogeneous PET methods in a comparable shared weight space.
- We design a paired low-rank structure that separates writing task knowledge into the backbone from realigning the running learner, so that arbitrary PET methods can be consolidated without altering their own algorithms.
- Our analysis provides an empirical account of what a backbone should learn from a PET module, identifying a method-invariant task component that alone moves the decisions of its task as a candidate unit of knowledge transfer in continual learning.

2 Related Work

2.1 PTM-Based Continual Learning

Continual learning with PTMs has converged on parameter-efficient tuning, and existing methods differ chiefly in where they place the parameters that carry task knowledge. Prompt-based methods maintain a shared pool of learnable tokens that are selected and composed through attention (Wang et al., 2022b;a; Smith et al., 2023); the pool caps memory growth, but because every task must be absorbed by the same bounded set of parameters, later tasks increasingly compete with earlier ones for it. Adapter-based methods instead insert bottleneck layers between transformer blocks, either adapting the backbone once at the first session (Zhou et al., 2025) or allocating a dedicated module

per task (Zhou et al., 2024b; Sun et al., 2025b); dedicated allocation avoids competition inside a fixed pool, at the price of storage and routing machinery that grow with the number of tasks. In both families, what a task contributes is held in parameters outside the backbone, and the backbone itself is not used as a store.

Low-rank methods place the budget inside the backbone, as a weight increment that can in principle be merged into it, and they differ in whether the knowledge of the stream ends up in the backbone or stays in the module. InfLoRA (Liang & Li, 2024) folds each task's increment into the weights after restricting it to the orthogonal complement of earlier tasks' input subspace; preserving their representations in this way is sufficient for their decisions but not necessary, so directions those decisions never used are withheld as well. SD-LoRA (Wu et al., 2025) retains the low-rank matrices of every task and recombines them at inference, so the modules function as a task bank. LCA (Tran et al., 2026) keeps a single module, initializing each task's LoRA from the previous one, merging the task vectors by elementwise magnitude selection, and realigning the classifiers to the result; the module remains the sole container of what the stream has learned, so the total deviation from the pre-trained weights never exceeds its rank, and the operands of the merge share one parameterization by construction. CoRe folds each converted increment into the backbone and learns the next expert on the result, so the accumulated update is neither confined to a prescribed subspace nor bounded by the rank of a module; and because the increment is distilled from the function of an arbitrary expert, the operands need not share a parameterization or even exist in the weight space.

2.2 Weight-Space Conversion and Model Merging

Expressing the knowledge of heterogeneous PET methods in one coordinate system requires two operations that prior work treats separately: converting a module into a weight-space form, and merging that form into a model. Analytical folding exists for linear or linearizable modules (Luo et al., 2023; Lian et al., 2022), and distillation has been shown to transfer low-rank modules across base models nearly losslessly (Wang et al., 2024b), but no exact static equivalent exists for prompts, whose effect depends on the input through attention (He et al., 2022; Dherin et al., 2025). Model merging, in turn, combines complete models of the same architecture, whether by resolving sign conflicts between task vectors (Yadav et al., 2023), weighting parameters by their Fisher information (Matena & Raffel, 2022), or, in continual learning, folding each newly fine-tuned model into its predecessor under recency-weighted Fisher averaging (Marouf et al., 2024). In all of these, the operands are symmetric and finished: complete models of the same form, combined once their training is over.

The setting addressed here differs in both operands and direction. The source is a PET module whose parameters are not in weight-space form, and the destination is a backbone on which a learner is still training with that very module attached. Merging is therefore asymmetric, a one-way write from module to backbone, and it occurs while the learner continues. Combining an existing conversion with an existing merging operator does not meet this setting: conversion alone leaves the module in place, and merging alone assumes a finished operand. In particular, once the module's knowledge has been written into the backbone the retained module applies it a second time, and each subsequent write compounds the duplication. CoRe couples the two operations, distilling an arbitrary module into a low-rank form and writing it into the backbone under a temporary realignment that cancels the duplicate application, so that heterogeneous methods can be compared in one weight space without altering the trajectory of the learner that produced them.

3 Method

3.1 Problem Setup

We consider CIL scenarios over a task sequence $\{\mathcal{D}_1, \ldots, \mathcal{D}_T\}$, in which a model observes each task once and must, at any point, classify inputs from all classes seen so far without access to the data of earlier tasks. We assume the learner starts from a pre-trained backbone with weights W_0 and adapts to the stream through PET: an arbitrary PET method $\mathcal{M}$ keeps the pre-trained weights W_0 fixed and confines what it learns to a module state P_t and a classifier H_t, which it updates at task t by its own algorithm. We write $z(x; W, P)$ for the classifier-input representation produced by backbone W with module P attached, $z(x; W)$ for the representation without a module, and

$F(W, P, H) = H(z(\cdot\,; W, P))$ for the induced decision function. Whatever $\mathcal{M}$ acquires over the stream is therefore carried by P_t and H_t, and remains outside W_0.

CoRe writes that knowledge into the backbone. At selected tasks, which we call consolidation steps, the knowledge accumulated in the module is transferred into the weights of the PTM backbone. We write c for the most recent consolidation step, W_c for the backbone it produced, and B_c for the training base handed to $\mathcal{M}$ afterwards; between consolidation steps, W_c is unchanged while $\mathcal{M}$ continues to learn on B_c. Because CoRe is used to observe what existing methods learn, it operates under two assumptions. First, the algorithm of $\mathcal{M}$, including its module transitions, routing, and classifier updates, is left untouched, and CoRe interacts with $\mathcal{M}$ only by supplying the base for the next task. Second, consolidation must be behaviorally transparent to $\mathcal{M}$: the method, continuing with its module on the base it receives, must compute what the consolidated model computes. We state the second assumption as a functional equivalence requirement,

$$F(B_t, P_t, H_t) \;\approx\; F(W_t, H_t), \tag{1}$$

where W_t is the backbone produced at consolidation step t and B_t is the base handed to $\mathcal{M}$ for the next task. The requirement is enforced by optimization and verified on held-out data; it constrains the decision function, and does not guarantee that the subsequent trajectory of $\mathcal{M}$ coincides with the one it would have followed without consolidation.

The goal of CoRe is thus to end the stream with a single backbone W_T that, together with the classifier of $\mathcal{M}$, reproduces the behavior the method would have had with its modules attached. Since this holds for any $\mathcal{M}$, the knowledge acquired by heterogeneous PET methods is expressed in one weight space and becomes directly comparable.

3.2 Commit: Writing an Expert into the Backbone

The commit step converts the expert produced at a consolidation step into an update of the backbone weights, and proceeds in two stages. The first stage expresses the expert in the weight space, by distilling the function it induces into a low-rank update of the attention projections. The second stage writes that update into the backbone under a constraint that limits its effect on decisions established at earlier steps. We describe the two stages in turn.

Distillation is required because the modules used by different methods have no common parameter-level expression. A prompt prepends learnable vectors to the token sequence, at the input or within the key and value projections of attention, an adapter inserts a nonlinear bottleneck between sublayers, and a low-rank module adds a weight increment; only the last of these can be merged into W_c by construction, and the first two alter the computation in ways that no static weight increment reproduces exactly. What the three do share is the function they induce, so CoRe matches that function instead of the parameters that realize it. We take the low-rank form as the target of this conversion for two reasons: it is the only PET family whose parameters are already weight-space increments, so writing it into the backbone requires no further conversion, and it restricts the fitted update to a low-dimensional subspace, which keeps the conversion well conditioned under the limited data of a single task session.

Concretely, let Δ_t denote a rank-r update attached with zero initialization to the query, key, and value projections of the current backbone W_c, where prompt effects are known to act (He et al., 2022; Dherin et al., 2025). Let $z_s = z(x; W_c{+}\Delta_t)$ and $z_T = z(x; B_c, P_t)$ denote the representations of the student and of the expert, the latter evaluated on the base its module was trained on. Holding the expert and the classifier H_t fixed, Δ_t is obtained by minimizing

$$\mathcal{L}_{\text{commit}} = \lambda_f \, d_{\text{feat}}(z_s, z_T) + \lambda_d \, \mathcal{D}\big(H_t(z_s) \,\|\, H_t(z_T)\big) + \lambda_m \, \mathcal{L}_{\text{margin}}, \tag{2}$$

whose three terms constrain the same function at successive points of the network. The representation term $d_{\text{feat}}(z_s, z_T) = \|z_s - z_T\|_2^2 + \big(1 - \cos(z_s, z_T)\big)$ constrains the input to the classifier in both magnitude and direction; the distribution term $\mathcal{D}(\cdot \,\|\, \cdot)$ is the Kullback–Leibler (KL) divergence between the class posteriors obtained by passing the two representations through a frozen copy of H_t; and the margin term $\mathcal{L}_{\text{margin}} = \text{SmoothL1}\big(m(z_s), m(z_T)\big)$, where $m(z)$ denotes the difference between the largest and the second largest logit of $H_t(z)$, together with the corresponding task-routing margin when the method maintains several task blocks, constrains the separation

between competing decisions. The third term is included because agreement in predicted labels does not imply agreement in the confidence with which those labels are separated, and a student that reproduces the expert's predictions with reduced margins degrades under the writes performed at subsequent consolidation steps. Distillation uses only the data of the current task, so no data from earlier tasks is stored or replayed.

The second stage must reconcile two requirements: the write should reproduce the distilled update as closely as possible, and it should not displace the decisions that earlier steps have established. CoRe quantifies the second requirement directly. Let $J_{<t}$ denote the Jacobian, with respect to the projections being written, of the decision margins the stream has committed to protect. A candidate write D displaces those margins by $J_{<t}\,\mathrm{vec}(D)$ to first order, so its expected squared displacement is the quadratic form $\mathrm{vec}(D)^\top H_{<t}\,\mathrm{vec}(D)$ with $H_{<t} = J_{<t}^\top J_{<t}$, a Gauss–Newton curvature of the protected margins. Two properties are imposed on the estimate of this curvature. It is accumulated as a signed cross-layer sketch $S_{<t} \in \mathbb{R}^{s\times d}$ whose capacity s is fixed, so that the cost of protecting the past does not grow with the length of the stream; and it is normalized by its spectral norm,

$$\overline{H}_{<t} = \frac{S_{<t}^\top S_{<t}}{\max\big(\|S_{<t}^\top S_{<t}\|_2,\ \epsilon\big)}, \tag{3}$$

so that the trade-off parameter introduced below has the same meaning at every step regardless of the scale of the accumulated sketch. We approximate the local margin-sensitivity curvature with a fixed-size sketch; Appendix C evaluates its approximation accuracy.

The write is then obtained as the update that stays closest to Δ_t subject to a penalty on the displacement it causes,

$$d_t^* = \arg\min_d \|d - \delta_t\|_2^2 + \lambda\, d^\top \overline{H}_{<t}\, d, \qquad W_t = W_c + \mathrm{unvec}(d_t^*), \tag{4}$$

where $\delta_t = \mathrm{vec}(\Delta_t)$ and the first consolidation step writes δ_1 unchanged. In the eigenbasis of $\overline{H}_{<t}$, the solution scales the component along the i-th eigenvector by $1/(1+\lambda\mu_i) \in (0,1]$, so the most interfering direction is attenuated by $1/(1+\lambda)$ while directions estimated to be inert are preserved, and no scale of the accumulated sketch can suppress the update entirely. Since the curvature is held in factored form, the solution is computed in the sketch space by the Woodbury identity,

$$d_t^* = \delta_t - \beta\, S_{<t}^\top\big(I + \beta\, S_{<t}S_{<t}^\top\big)^{-1} S_{<t}\,\delta_t, \qquad \beta = \lambda \big/ \max\big(\|S_{<t}^\top S_{<t}\|_2,\ \epsilon\big), \tag{5}$$

which requires solving a system of size s rather than of the parameter dimension d.

3.3 Rebase: Realigning the Learner to the Updated Backbone

Writing an expert into the backbone leaves the method in a state that is inconsistent with the model it now runs on. The commit was constructed so that $z(\cdot\,; W_t) \approx z(\cdot\,; B_c, P_t)$, and the method retains P_t because its subsequent tasks depend on it; attaching that module to W_t therefore applies the same adaptation twice. The consequence is not confined to a single step. Each later consolidation step writes an expert that already includes the effect of the retained module, so the discrepancy between the intended and the realized model accumulates with the number of steps. Discarding the module after each write removes the duplication but is not admissible under our first assumption, since prompt-pool accumulation, subspace bookkeeping, and classifier alignment are all carried by the state that would be destroyed. The duplication must therefore be cancelled while the state is preserved.

Two requirements govern how it is cancelled. The first is functional equivalence: the method, continuing with its module on the base it receives, must compute what the consolidated model computes, which is the requirement stated in Eq. (1). The second concerns the conditions under which the method resumes learning. What determines its next trajectory is the base it is handed, not the size of the correction applied to obtain that base, so a correction that restores equivalence while moving the base far from B_c would satisfy the first requirement and still alter how the method behaves afterwards. CoRe therefore controls the displacement of the base itself, and treats the correction as a temporary object.

Accordingly, a low-rank correction C_t is trained on the attention projections, with the backbone, the module, and the classifier all frozen, by minimizing

$$\min_{C_t} \lambda_f \, d_{\text{feat}}\big(z(W_t + C_t, P_t),\, z(W_t)\big) + \lambda_d \, \mathcal{D}\big(H_t(z(W_t + C_t, P_t)) \,\|\, H_t(z(W_t))\big) + \lambda_m \, \mathcal{L}_{\text{margin}} + \lambda_n \frac{\|C_t\|_F^2}{\|B_c\|_F^2} + \lambda_b \frac{\|(W_t + C_t) - B_c\|_F^2}{\|B_c\|_F^2}, \tag{6}$$

where the first three terms are those of Eq. (2), now applied with the consolidated model as the target and the paired model as the student, and the fourth is a norm penalty on the correction. The last term penalizes the displacement of the base and implements the second requirement. Consistently with it, C_t is initialized at the truncated SVD of $B_c - W_t$ within the rank budget, which is the closest reachable point to the base the method was already learning on, so optimization departs from that point only as far as functional equivalence requires.

After optimization the next base is set to $B_t = W_t + C_t$ and the correction is absorbed into it, so no correction is retained and inference never involves C_t. As is common when applying a weight-space direction (Ilharco et al., 2023; Wortsman et al., 2022), a bounded scalar multiple of C_t may additionally be added to W_t when a held-out split of the current task indicates that the write has fallen short of the expert; the criterion and range are given in Appendix A. At the level of the stream, the first task is always a consolidation step and so is the last; between steps the method learns on a fixed base and the next step converts the expert accumulated over that interval, with the interval between steps treated as a free parameter. Every run therefore ends with a backbone W_T and the classifier of the method, with no module or routing remaining. Although each write is low-rank, writes accumulate across steps, so the knowledge held in the backbone grows with the stream rather than being bounded by a fixed budget.

4 Experiments

Our experiments answer three questions in sequence. Section 4.2 establishes that the conversion preserves what it converts, which is a precondition for reading anything off the converted updates. Section 4.3 then asks whether updates learned for the same task by different PET methods share a common component, and Section 4.4 identifies where the component comes from and what it does; Appendices E and F extend the analysis to how the component behaves under the write and what exploiting it affords.

4.1 Experimental Setup

Methods and benchmarks. The analysis covers seven PET-based CIL methods whose modules take each of the three forms in which PET expresses knowledge: prompts in L2P, DualPrompt, CODA-Prompt, and APER-VPT; a bottleneck adapter in APER-Adapter; and low-rank increments in InfLoRA and LCA. Six model merging methods, introduced in Section 4.2, are compared as writing rules. All experiments run in the PILOT framework (Sun et al., 2025a) on seven benchmarks, from a common ImageNet-21K pre-trained ViT-B/16 and with the task sequence, class order, and seeds shared by every method; benchmarks, splits, and hyperparameters are detailed in Appendix A.

Metrics. Conversion fidelity is measured by agreement—the fraction of held-out samples on which the consolidated model and the expert predict the same label—and by the logit KL divergence between their output distributions; Centered Kernel Alignment (CKA) on classifier-input representations and the absolute top-1 gap between the two models are reported as complementary metrics. To compare converted updates, we evaluate two geometric aspects. Direction is the signed cosine between the flattened parameter vectors of two updates, and captures which decisions an update moves. Allocation is the cosine between their energy profiles, where the energy profile of an update is the vector of squared Frobenius norms of its blocks over the twelve layers and three attention projections; it discards sign and captures where an update places its mass.

4.2 Conversion Fidelity

Any conclusion drawn from converted updates is only as trustworthy as the conversion itself, so we first establish that writing an expert into the backbone reproduces that expert's behavior, and that it

Table 1: Fidelity of the consolidated model to the expert it replaces under different writing rules, averaged over seven PET methods and seven benchmarks. Best values in bold.

Writing rule	Agreement (%) ↑	Logit KL ↓	|Top-1 gap| (pp) ↓	CKA ↑
Unconstrained write	93.29 ± 4.43	0.0404	0.750	0.9884
TIES-Merging	90.86 ± 6.36	0.0670	1.234	0.9859
Fisher merging	95.34 ± 3.33	0.0167	0.719	0.9967
Fisher-weighted averaging (CoFiMA)	95.47 ± 3.42	0.0115	0.719	0.9972
Input-subspace projection (InfLoRA)	93.50 ± 4.63	0.0320	1.024	0.9942
Incremental Merging (LCA)	95.80 ± 3.15	0.0157	0.740	**0.9981**
CoRe (ours)	$\mathbf{97.46 \pm 2.10}$	**0.0072**	**0.505**	**0.9981**

does so uniformly across methods. Uniformity is a requirement: a conversion whose quality varied systematically by method would be indistinguishable, in the analyses that follow, from genuine differences between what those methods learn.

Once an expert has been distilled into a low-rank update, writing it into the backbone is a separate operation for which established rules exist, so we compare the interference-regularized write of CoRe against five alternatives applied to the same distilled updates: TIES-Merging (Yadav et al., 2023), Fisher-weighted merging (Matena & Raffel, 2022), Fisher-weighted averaging of CoFiMA (Marouf et al., 2024), the elementwise magnitude-selection rule of LCA (Tran et al., 2026), and the input-subspace projection of InfLoRA (Liang & Li, 2024), which removes from the update every direction occupied by the inputs of earlier tasks. An unconstrained write, which folds the distilled update without any interference control, serves as the lower reference and an ablation of the regularizer. Table 1 reports agreement and logit divergence between the consolidated model and the expert as direct measures of behavioral equivalence, together with CKA of the classifier-input representations and the absolute top-1 gap.

CoRe is the most faithful rule on every quantity, and its variance across methods and benchmarks is the smallest. Relative to the unconstrained write it reduces logit divergence by more than five times, so the interference regularizer improves fidelity rather than trading it for stability; relative to the strongest competitor it halves the divergence and raises agreement by nearly two points. The ordering is also informative about what the write requires. TIES-Merging falls below even the unconstrained write, which follows from its design: it resolves sign conflicts among several finished models, so applied to a single update it trims and rescales a vector with no competitor. The projection of InfLoRA matches the unconstrained write in agreement, widens the top-1 gap, and has the second-largest variance: it removes every direction that earlier inputs occupy, and because the expert's own decisions depend on some of those directions as well, part of what the write was meant to transmit is discarded along with the interference. The three remaining rules improve markedly, and they share what the first two lack, namely that each decides per parameter how much of the update to admit. Fisher merging and CoFiMA do so through a second-order estimate of parameter importance; the magnitude-selection rule of LCA retains the larger of the incoming and existing values, which under a single write behaves as a magnitude gate, and its competitiveness is consistent with its reported effectiveness on PET parameters. CoRe differs from all of them in what its gate is derived from, namely the decision margins the stream has committed to protect, and in solving the write in closed form under a normalized interference budget. Read together, the results indicate that what a criterion protects, and where its curvature is estimated, matter more than whether interference is controlled at all. Fidelity under repeated writes within a stream, and its degradation without realignment, are reported in Appendix D.

4.3 Direction and Allocation

We ask whether updates learned for the same task by different PET methods share anything, and separate two aspects of an update that need not agree. The direction of an update is its signed orientation in the shared coordinate system, which determines which decisions it moves; its allocation is the unsigned distribution of its energy over layers and projections, which determines where its mass is placed. Similarity in each is expressed as a standardized contrast against the both-different pairs of the same dataset, so that the two aspects are read on a common scale.

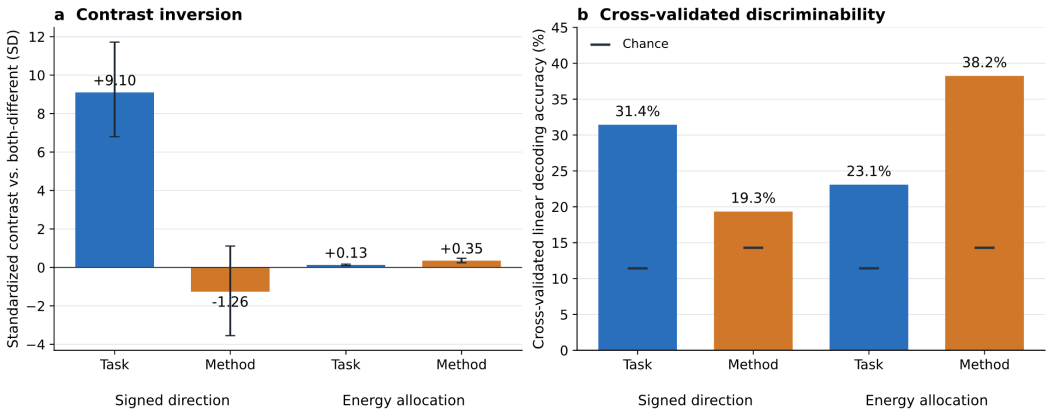


Figure 1: **Task and method contributions to update direction and allocation.** (a) Standardized contrast of same-task and same-method pairs against the both-different reference, in signed direction and in energy allocation. (b) Comparison of chance-adjusted decoding accuracy for task and method identity from each of the two representations.

The two aspects are governed by different factors, and the contrast inverts between them (Figure 1a). In signed direction, pairs that share a task but differ in method lie nine standard deviations above the both-different reference, whereas pairs that share a method but differ in task lie slightly below it; in energy allocation the ordering reverses, with the method contrast exceeding the task contrast. The task is the dominant organizing factor of signed update direction, whereas the producing method dominates energy allocation. Cross-validated decoding confirms the inversion (Figure 1b): allocation identifies the producing method at nearly three times chance, while signed direction identifies the task stage at nearly three times chance, and each channel decodes the other identity only weakly. The claim is one of dominance rather than separability: the shared coordinate distinguishes where a method allocates capacity from which direction it learns. The task-aligned component is small in absolute magnitude, but it is well separated from the reference distribution and is what the remainder of this section examines.

4.4 Identifying the Shared Component

The preceding analysis established that updates for the same task share a direction regardless of the method that produced them, but not what that shared direction represents. Every method in a dataset block is trained on the same images under the same augmentation, so the agreement could reflect the input distribution rather than the function learned on it. To determine which, we corrupt the learned function while holding the input fixed: labels are permuted at ratios from 0 to 100%, and the shared component estimated from the resulting updates is transplanted alone into the pre-trained backbone. A component driven by input statistics would be unaffected, whereas one driven by the learned function would degrade in proportion to the corruption. The transplant gain declines monotonically with the permutation ratio and reaches zero once the labels carry no task information (Figure 2a). What the methods share is therefore the function they learned, not the distribution.

The shared component is also causally responsible for the decisions of the task it was estimated from. For each task, the component is estimated from every method except the one it will be evaluated against, applied alone to the pre-trained backbone without any training, and evaluated on all tasks, with outcomes centered on each evaluation task's mean over donors so that the general benefit of perturbing the backbone is removed. A component transplanted into its own task improves it by six times the gain obtained from components of other tasks (Figure 2b; one-sided Wilcoxon $p = 0.0035$), and the donor task can be recovered from the induced change in the output distribution at more than four times chance. Removal points the same way: deleting the shared component costs significantly more accuracy than deleting a random subspace of equal norm ($p = 0.021$), although only a minority of individual cells respond strongly, which indicates that task information is redundantly distributed and that the shared component is the common denominator across methods. With

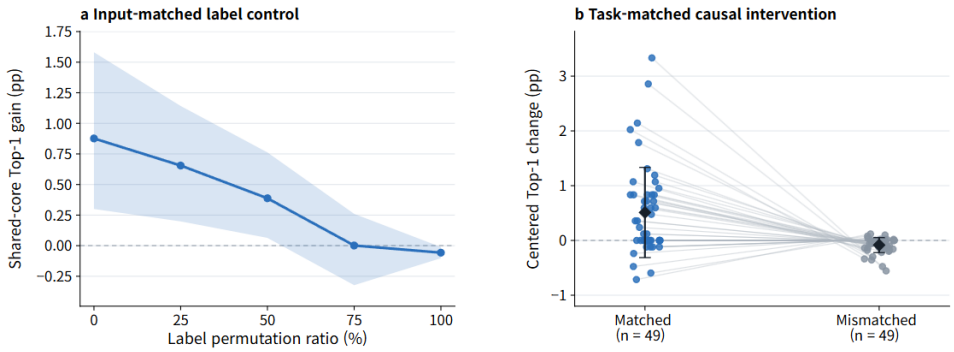


Figure 2: **Characterization and causal role of the shared component.** (a) Effect of label permutation ratio on the transplant gain of the shared component. (b) Comparison of transplant gains between task-matched and task-mismatched shared components.

the learned classifier held fixed, the shared component alone recovers 16.1% of the full converted update's accuracy gain (+0.731 versus +4.541 pp), averaged over the 49 method–benchmark pairs. Scaling each held-out method's own update to the same norm recovers 11.6% (+0.527 pp), providing a magnitude-matched reference.

The write leaves this component intact. Applying the interference-regularized fold of Section 3.2 retains over 99.98% of the update's projection on the shared direction while substantially reducing the estimated risk to protected decisions, and the retention is unchanged when the regularization is strengthened a hundredfold or replaced by projection onto the null space of the accumulated past-margin Jacobian (Appendix E). This outcome is what the preceding analyses predict: the shared direction is specific to its task, the protected margins belong to earlier tasks, and the subspace on which any rank-limited protection criterion acts occupies a vanishing fraction of the coordinate space, so absent a particular alignment between the two, the shared coordinate passes through the write untouched. The consequence for consolidation is practical: at the capacities at which second-order protection is implemented, writing the shared component costs essentially none of the protection budget, a property Appendix F exploits directly.

5 Conclusion

In this paper, we investigated what a backbone should learn from PET modules as continual task streams exhaust external parameter budgets. To bridge the incompatible formats of heterogeneous methods, we introduced CoRe, an instrument that converts arbitrary PET experts into a shared low-rank weight space while the learner continues to run. Our empirical analysis reveals a systematic decomposition: an update's energy allocation is governed by the method, whereas its signed direction is driven by the task. Crucially, this small, format-invariant component is causally sufficient to guide task decisions when transplanted alone into the pre-trained backbone.

Every PET method packages its knowledge differently, but part of what it learns does not belong to the method. Heterogeneous PET methods deposit what they learn in method-specific allocations, yet each update contains a component determined by the task alone; the interference-regularized write passes this component through at essentially no cost to protected decisions, and a minimal low-rank learner that retains it through a compressed write can reach the level of a recent method of the same family without prompt selection, classifier alignment, or merging heuristics. We therefore read the shared component as a candidate unit of knowledge transfer: what a backbone should learn from a PET module is, at minimum, this method-invariant functional core. Two questions follow. The immediate one is estimation: a single auxiliary method already yields a usable direction, but a consolidation rule that estimates the component from a learner's own trajectory, treating seeds, augmentations, or checkpoints as surrogate donors, remains to be built. The further one is scope: the components studied here are task-specific, and whether components pooled from heterogeneous experts compose into a reusable basis that transfers across tasks is the question that ultimately decides what a continual learner should write into its own weights.

AI USE STATEMENT

In this work, we used generative AI tools for providing feedback on research methodology and experiments, implementing methods, and assistance with translation. We have not used generative AI tools for developing theoretical models, formulating or proving mathematical claims, proposing or refining hypotheses, cleaning and reformatting datasets, or interpreting results, and the remaining required disclosure tasks are not applicable to this work. Additionally, we used generative AI tools for creating and editing software code. We have reviewed all AI-assisted work: we checked for potential plagiarism through a manual literature survey, and all LLM-generated code and technical outputs were rigorously verified, tested, and validated for correctness by at least two authors. We take full responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

To ensure the reproducibility of our results, all datasets used in our experiments are publicly available and cited with accessible references. The complete source code, along with environment dependencies and instructions to reproduce our experimental results, is provided in the supplementary material. Furthermore, comprehensive implementation details, hyperparameter configurations, and evaluation procedures are thoroughly documented.

REFERENCES

Benoit Dherin, Michael Munn, Hanna Mazzawi, Michael Wunder, and Javier Gonzalvo. Learning without training: The implicit dynamics of in-context learning. *arXiv preprint arXiv:2507.16003*, 2025.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations (ICLR)*, 2021.

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

Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, Dawn Song, Jacob Steinhardt, and Justin Gilmer. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 8340–8349, 2021a.

Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15262–15271, 2021b.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2790–2799, 2019. URL `https://proceedings.mlr.press/v97/houlsby19a.html`.

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

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

Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning. In *European Conference on Computer Vision*, 2022. URL https://www.ecva.net/papers/eccv_2022/papers_ECCV/html/4175_ECCV_2022_paper.php.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017. doi: 10.1073/pnas.1611835114. URL https://pmc.ncbi.nlm.nih.gov/articles/PMC5380101/.

Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *2013 IEEE International Conference on Computer Vision Workshops*, pp. 554–561, 2013.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Dongze Lian, Daquan Zhou, Jiashi Feng, and Xinchao Wang. Scaling & shifting your features: A new baseline for efficient model tuning. In *Advances in Neural Information Processing Systems*, volume 35, pp. 109–123, 2022.

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

Gen Luo, Minglang Huang, Yiyi Zhou, Xiaoshuai Sun, Guannan Jiang, Zhiyu Wang, and Rongrong Ji. Towards efficient visual adaption via structural re-parameterization. *arXiv preprint arXiv:2302.08106*, 2023.

Imad Eddine Marouf, Subhankar Roy, Enzo Tartaglione, and Stéphane Lathuilière. Weighted ensemble models are strong continual learners. In *European Conference on Computer Vision*, pp. 306–324, 2024.

Michael S. Matena and Colin A. Raffel. Merging models with Fisher-weighted averaging. In *Advances in Neural Information Processing Systems*, volume 35, pp. 17703–17716, 2022.

Jonathan Schwarz, Jelena Luketina, Wojciech M. Czarnecki, Agnieszka Grabska-Barwinska, Yee Whye Teh, Razvan Pascanu, and Raia Hadsell. Progress & compress: A scalable framework for continual learning. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 4528–4537, 2018. URL https://proceedings.mlr.press/v80/schwarz18a.html.

James Seale Smith, Leonid Karlinsky, Vyshnavi Gutta, Paola Cascante-Bonilla, Donghyun Kim, Assaf Arbelle, Rameswar Panda, Rogerio Feris, and Zsolt Kira. CODA-Prompt: Continual decomposed attention-based prompting for rehearsal-free continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11909–11919, 2023.

Hai-Long Sun, Da-Wei Zhou, De-Chuan Zhan, and Han-Jia Ye. PILOT: A pre-trained model-based continual learning toolbox. *SCIENCE CHINA Information Sciences*, 68(4):147101, 2025a. doi: 10.1007/s11432-024-4276-4.

Hai-Long Sun, Da-Wei Zhou, Hanbin Zhao, Le Gan, De-Chuan Zhan, and Han-Jia Ye. MOS: Model surgery for pre-trained model-based class-incremental learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 20699–20707, 2025b.

Tung Tran, Danilo Vasconcellos Vargas, and Khoat Than. LCA: Local classifier alignment for continual learning. In *International Conference on Learning Representations*, 2026.

Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 7 2011.

Liyuan Wang, Xingxing Zhang, Hang Su, and Jun Zhu. A comprehensive survey of continual learning: Theory, method and application. *IEEE transactions on pattern analysis and machine intelligence*, 46(8):5362–5383, 2024a.

Runqian Wang, Soumya Ghosh, David Cox, Diego Antognini, Aude Oliva, Rogerio Feris, and Leonid Karlinsky. Trans-LoRA: Towards data-free transferable parameter efficient finetuning. In *Advances in Neural Information Processing Systems*, volume 37, 2024b.

Zifeng Wang, Zizhao Zhang, Sayna Ebrahimi, Ruoxi Sun, Han Zhang, Chen-Yu Lee, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. DualPrompt: Complementary prompting for rehearsal-free continual learning. In *European Conference on Computer Vision*, pp. 631–648, 2022a.

Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. Learning to prompt for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 139–149, 2022b.

Mitchell Wortsman, Gabriel Ilharco, Jong Wook Kim, Mike Li, Simon Kornblith, Rebecca Roelofs, Raphael Gontijo Lopes, Hannaneh Hajishirzi, Ali Farhadi, Hongseok Namkoong, and Ludwig Schmidt. Robust fine-tuning of zero-shot models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7959–7971, 2022.

Yichen Wu, Hongming Piao, Long-Kai Huang, Renzhen Wang, Wanhua Li, Hanspeter Pfister, Deyu Meng, Kede Ma, and Ying Wei. SD-LoRA: Scalable decoupled low-rank adaptation for class incremental learning. In *International Conference on Learning Representations*, 2025.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin A. Raffel, and Mohit Bansal. TIES-Merging: Resolving interference when merging models. In *Advances in Neural Information Processing Systems*, volume 36, pp. 7093–7115, 2023.

Xiaohua Zhai, Joan Puigcerver, Alexander Kolesnikov, Pierre Ruyssen, Carlos Riquelme, Mario Lucic, Josip Djolonga, Andre Susano Pinto, Maxim Neumann, Alexey Dosovitskiy, Lucas Beyer, Olivier Bachem, Michael Tschannen, Marcin Michalski, Olivier Bousquet, Sylvain Gelly, and Neil Houlsby. A visual task adaptation benchmark. *arXiv preprint arXiv:1910.04867*, 2019.

Yuanhan Zhang, Zhenfei Yin, Jing Shao, and Ziwei Liu. Benchmarking omni-vision representation through the lens of visual realms. In *European Conference on Computer Vision (ECCV)*, pp. 594–611. Springer, 2022.

Da-Wei Zhou, Hai-Long Sun, Jingyi Ning, Han-Jia Ye, and De-Chuan Zhan. Continual learning with pre-trained models: A survey. *arXiv preprint arXiv:2401.16386*, 2024a.

Da-Wei Zhou, Hai-Long Sun, Han-Jia Ye, and De-Chuan Zhan. Expandable subspace ensemble for pre-trained model-based class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 23554–23564, 2024b.

Da-Wei Zhou, Zi-Wen Cai, Han-Jia Ye, De-Chuan Zhan, and Ziwei Liu. Revisiting class-incremental learning with pre-trained models: Generalizability and adaptivity are all you need. *International Journal of Computer Vision*, 133(3):1012–1032, 2025.

A Implementation Details

Benchmarks and protocol. The backbone is a ViT-B/16 pre-trained on ImageNet-21K (Dosovitskiy et al., 2021), and the data pipeline, splits, and preprocessing are those of PILOT without modification, with inputs resized to 224×224. CIFAR-100 (Krizhevsky, 2009) (100 classes), CUB-200 (Wah et al., 2011), ImageNet-R (Hendrycks et al., 2021a), ImageNet-A (Hendrycks et al., 2021b) (200 each), and OmniBenchmark (Zhang et al., 2022) (300) are partitioned into ten tasks of equal class count. Stanford CARS (Krause et al., 2013) (196 classes) uses nine tasks of 20 classes followed by a final task of 16, and the 50-class VTAB (Zhai et al., 2019) subset uses five tasks of ten classes.

Table 2: CoRe hyperparameters shared across subject methods and benchmarks. The 20% validation fraction refers to the capped compensation pool, not the full training set.

Stage	Hyperparameter	Value
Commit: distillation	rank r of Δ_t, per Q/K/V projection	64
	$(\lambda_f, \lambda_d, \lambda_m)$	$(1, 1, 1)$
	epochs / minimum optimizer updates	12 / 400
	learning rate	10^{-3}
	batch size	32
	optimizer	AdamW
	distillation temperature	2
Commit: write	interference strength λ	1
	number of signed probes P	8
	rank per probe and Q/K/V block q	16
	current-task sketch sample budget N	1,024
	numerical zero-curvature tolerance	10^{-20}
Rebase	rank of C_t, per Q/K/V projection	64
	feature / decision / margin weights	$1/1/1$
	(λ_n, λ_b)	$(0.01, 1)$
	epochs / learning rate	$12/10^{-3}$
	batch size	32
	initialization	stationary-base SVD
Schedule and repair	consolidation interval K	2
	repair grid ρ	$\{0, 0.125, 0.25, 0.375, 0.5\}$
	compensation calibration pool	at most 1,024
	validation fraction of that pool	20%

CoRe hyperparameters. Table 2 gives the configuration for interference-regularized CoRe. The conversion and rebase hyperparameters are shared across all seven subject methods and seven benchmarks, without method-specific CoRe tuning. Both assimilation and compensation act on the query, key, and value projections in all 12 transformer blocks; the FFN and attention output projections are not assigned additional LoRA factors. The subject PET parameters and classifier are not optimized during these two conversion stages. Both stages use AdamW with zero weight decay, cosine learning-rate decay, and gradient-norm clipping at 1.0. The assimilation loss combines feature fidelity, temperature-2 logit distillation, and teacher-decision margin fidelity with unit weights; the hard-label cross-entropy coefficient is zero. Margin fidelity matches the teacher's top-1 margin and, when more than one task has been seen, its cross-task routing margin. Rebase uses the same feature and margin terms; its decision term includes temperature-2 KL and squared distance between centered, unit-normalized logits.

Estimation of the shared component. CoRe provides a weight-space representation at the interface between expert acquisition and backbone writing. We use this interface to identify the component shared across methods, and compare its coordinates before and after the write to measure how consolidation transforms it. Stream-level measurements follow the task-indexed representations generated during continual learning. For label controls and causal transplantation, we hold the recipient backbone fixed at W_0 and use first-task conversions, where all methods share this initialization. This controlled setting isolates the functional effect of the transplanted component from differences in accumulated backbone history. The corresponding component is then examined under recurrent writing in Appendix E.

Let $\delta_\tau^{(m)} = \mathrm{vec}(\Delta_\tau^{(m)})$ denote the converted update and $u_\tau^{(m)} = \delta_\tau^{(m)}/\|\delta_\tau^{(m)}\|_2$ its unit normalization. Excluding the method against which the estimate is evaluated, we estimate the shared direction as follows.

$$c_\tau^{(-m)} = \arg\max_{\|c\|_2=1} \sum_{m' \neq m} \left(c^\top u_\tau^{(m')}\right)^2, \tag{7}$$

with its sign chosen to give nonnegative aggregate alignment with the donor updates. The shared component of the held-out update is its orthogonal projection onto this direction,

$$s_\tau^{(m)} = \left(c_\tau^{(-m)\top} \delta_\tau^{(m)}\right) c_\tau^{(-m)}, \tag{8}$$

so the held-out method contributes nothing to the direction and only its own update determines the amplitude. In cross-task transplantation, each donor component is rescaled to the norm of the target's shared component; the removal experiment subtracts $s_\tau^{(m)}$ itself and is compared against one-dimensional random directions matched to the same removed norm; the label-permutation experiment applies components at their original amplitudes. Independent re-distillations of the same expert under different LoRA initializations provide an empirical reference for the coordinate variability introduced by distillation itself.

Computational cost. Backbone writing and rebase occur only at consolidation steps, whereas the analysis implementation additionally performs diagnostic distillation and fidelity evaluation at intermediate tasks. Consequently, its end-to-end runtime includes more than consolidation-only overhead. At $P = 8$ and $q = 16$, the persistent sketch contains FP16 left/right factors and FP32 singular values, requiring 14,174,208 bytes, or approximately 13.52 MiB, for all 12 ViT-B/16 Q/K/V blocks. This figure excludes temporary sketch-construction buffers, model and optimizer states, and serialized-file metadata. The sketch storage is independent of stream length. All experiments run on a single NVIDIA GeForce RTX 4090 with 24 GB of GPU memory.

B THE CORE ALGORITHM

Algorithm 1 states the full procedure of Section 3. The method trains on every task by its own algorithm, and CoRe acts at consolidation steps, where three operations run in sequence: the commit distills the expert into Δ_t and folds it into the backbone under the interference criterion of Eq. (4), the rebase fits the correction C_t that cancels the duplicate application and sets the next base, and the sketch is then extended with the margins the write has committed to protect. Both low-rank modules are discarded at the end of the step, so what carries across consolidation steps is the backbone, the base handed to the method, and the sketch.

Two points of notation. $\mathrm{BlockSVD}_r$ operates separately on each Q/K/V projection, and Rank_q denotes rank-q recompression stored as low-rank factors. The Rademacher sign $\xi_{p,i}$ is shared across all blocks for a given probe and example, which preserves signed cross-layer interactions; the sketch is updated by adding the new probe contributions to the existing ones and recompressing, so its size does not grow with the stream.

Algorithm 1 CoRe: continual conversion and consolidation of PET experts

Require: pre-trained weights W_0; PET method $\mathcal{M}$; stream $\{\mathcal{D}_t\}_{t=1}^{T}$; interval K; LoRA rank r; interference strength λ; sketch parameters (P, q, N); repair grid $\mathcal{R}$

$W \leftarrow W_0$; $B \leftarrow W_0$
$(P_0, H_0) \leftarrow \mathcal{M}.\mathrm{Initialize}()$
$Z_{p,b} \leftarrow 0$ for all probes p and Q/K/V blocks b; $S \leftarrow 0$
for $t = 1, \ldots, T$ **do**
  $(P_t, H_t) \leftarrow \mathcal{M}.\mathrm{TrainTask}(B, P_{t-1}, H_{t-1}, \mathcal{D}_t)$
  if $t = T$ or $(t-1) \bmod K = 0$ **then**
    ▷ Commit
    Initialize rank-r Q/K/V LoRA with zero effective update Δ_t
    Optimize Δ_t on $\mathcal{D}_t$ using $\mathcal{L}_{\mathrm{commit}}$, with target $F(B, P_t, H_t)$ and fixed W, P_t, H_t
    $\delta_t \leftarrow \mathrm{vec}(\Delta_t)$; $d_t \leftarrow \delta_t$
    $h \leftarrow \lambda_{\max}(SS^\top)$
    if $h > \epsilon$ and $\|\delta_t\|_2^2 > \epsilon$ **then**
      $\beta \leftarrow \lambda / h$
      $u \leftarrow \mathrm{Solve}(I + \beta SS^\top,\ S\delta_t)$
      $d_t \leftarrow \delta_t - \beta S^\top u$
    end if
    $\bar{W} \leftarrow W + \mathrm{unvec}(d_t)$
    ▷ Rebase
    $(\mathcal{D}_{\mathrm{opt}}, \mathcal{D}_{\mathrm{val}}) \leftarrow \mathrm{Split}(\mathrm{Sample}_{\le N}(\mathcal{D}_t))$
    $C_t \leftarrow \mathrm{BlockSVD}_r(B - \bar{W})$
    Optimize C_t on $\mathcal{D}_{\mathrm{opt}}$ using $\mathcal{L}_{\mathrm{rebase}}$, with fixed $\bar{W}, P_t, H_t$
    $B^+ \leftarrow \bar{W} + C_t$; $\rho_t \leftarrow 0$
    if $\mathrm{Acc}_{\mathcal{D}_{\mathrm{val}}}(F(\bar{W}, H_t)) < \mathrm{Acc}_{\mathcal{D}_{\mathrm{val}}}(F(B^+, P_t, H_t))$ **then**
      $\rho_t \leftarrow \arg\max_{\rho \in \mathcal{R}} \mathrm{Acc}_{\mathcal{D}_{\mathrm{val}}}(F(\bar{W} + \rho C_t, H_t))$
    end if
    $W \leftarrow \bar{W} + \rho_t C_t$; $B \leftarrow B^+$
    Discard the temporary LoRA modules Δ_t, C_t
    ▷ UpdateSketch
    $\mathcal{E}_t = \{(x_i, y_i)\}_{i=1}^{n} \leftarrow \mathrm{Sample}_{\le N}(\mathcal{D}_t)$
    $z_i \leftarrow F(W, H_t; x_i)$; $m_i \leftarrow z_{i,y_i} - \max_{j \neq y_i} z_{i,j}$
    for $p = 1, \ldots, P$ **do**
      Draw independent $\xi_{p,i} \sim \mathrm{Rademacher}(\{-1, +1\})$ for $i = 1, \ldots, n$
      for each Q/K/V block b **do**
        $G_{p,b} \leftarrow \frac{1}{\sqrt{n}} \sum_{i=1}^{n} \xi_{p,i} \nabla_{W_b} m_i$
        $Z_{p,b} \leftarrow \mathrm{Rank}_q(Z_{p,b} + G_{p,b})$
      end for
    end for
  end if
end for
return W, H_T

C Accuracy of the Curvature Sketch

Evaluation protocol. We characterize how sketch capacity affects the approximation of local margin-sensitivity curvature. At a fixed checkpoint, let J collect the per-example gradients of the ground-truth decision margin with respect to the Q/K/V weights. For n examples, the reference directional response is

$$r(d) = d^\top H d = \frac{1}{n}\|Jd\|_2^2, \qquad H = \frac{1}{n} J^\top J.$$

Table 3: Curvature-sketch approximation characteristics. Values are averaged over five probe seeds within each checkpoint and then equally across the three checkpoints.

P	q	$E_{\text{response}} \downarrow$	Spearman $\uparrow$	$E_{\text{compression}} \downarrow$
8	∞	0.424	0.848	–
8	8	0.475	0.768	0.478
8	16	0.410	0.791	0.281
8	32	0.387	0.816	0.155
32	∞	0.189	0.945	–
32	16	0.282	0.900	0.275

The sketch estimates this response as $\hat{r}_q(d) = \|S_q d\|_2^2/P$, where P is the number of signed probes and q is the compression rank per Q/K/V block. The factor $1/P$ cancels under the spectral normalization used by the commit operator.

We reuse the first consolidated checkpoints of APER-VPT/CIFAR-100, APER-Adapter/CUB, and InfLoRA/CARS. For each checkpoint, we compute per-example Jacobians on 64 fixed current-task training examples, covering all 21,233,664 Q/K/V coordinates. All configurations use identical examples and evaluation transforms. We examine five saved assimilation directions from tasks 3, 5, 7, 9, and 10, together with five coordinate-sign-randomized controls that preserve the task-3 update's coordinate magnitudes. All directions are normalized to unit norm. Each configuration is evaluated with five probe seeds.

Approximation metrics. We measure relative error in directional responses and their Spearman rank correlation. To distinguish probe sampling from blockwise compression, we also compare each compressed sketch with its uncompressed counterpart constructed from the same probes:

$$E_{\text{response}} = \frac{\|\hat{\mathbf{r}}_q - \mathbf{r}\|_2}{\|\mathbf{r}\|_2}, \qquad E_{\text{compression}} = \frac{\|\hat{\mathbf{r}}_q - \hat{\mathbf{r}}_\infty\|_2}{\|\hat{\mathbf{r}}_\infty\|_2}.$$

Here, $q = \infty$ denotes signed probes without blockwise rank compression.

Results. The default configuration achieves a mean directional-response correlation of 0.791 with the full-Jacobian reference. The corresponding correlations are 0.593 for APER-VPT/CIFAR-100, 0.893 for APER-Adapter/CUB, and 0.886 for InfLoRA/CARS. Considering only the five saved assimilation directions gives a mean correlation of 0.693.

The capacity sweep reveals two complementary controls over approximation quality. Increasing the number of probes from 8 to 32 reduces the uncompressed response error from 0.424 to 0.189 and raises correlation from 0.848 to 0.945. Increasing the block rank at fixed $P = 8$ reduces compression error from 0.478 at $q = 8$ to 0.281 at $q = 16$ and 0.155 at $q = 32$. With $q = 16$, increasing P to 32 improves the combined response error to 0.282 and correlation to 0.900. These results identify probe count and block rank as distinct controls over sampling variability and representation loss.

Effect on the write operator. We further characterize how the approximation propagates through the spectrally normalized write operator. Using $\lambda = 1$, we fold the saved task-3 assimilation direction and evaluate the resulting response with the full Jacobian. The default sketch yields mean post-write/pre-write response ratios of 0.972, 0.811, and 0.935 for APER-VPT/CIFAR-100, APER-Adapter/CUB, and InfLoRA/CARS, respectively. Using the full empirical curvature yields corresponding ratios of 0.757, 0.630, and 0.589. This comparison quantifies the effect of finite sketch capacity on the magnitude of the induced weight-space adjustment.

Implementation implications. The sketch provides a fixed-size representation of directional margin sensitivity whose resolution can be controlled through P and q. Across the evaluated configurations, additional probes improve response estimation, while higher block rank improves the representation of each probe. Persistent storage scales as $\mathcal{O}(LPq(d_{\text{in}} + d_{\text{out}}))$ for L projection blocks and remains independent of stream length. The measurements therefore provide an explicit storage–resolution characterization of the curvature component used by CoRe, making its approximation behavior measurable within the common weight-space analysis.

Table 4: Fidelity of the consolidated model to the expert it replaces, by benchmark, averaged over the seven subject methods.

Benchmark	Agreement (%) ↑	Logit KL ↓	|Top-1 gap| (pp) ↓	CKA ↑
CIFAR-100	97.70	0.00939	0.654	0.9956
CUB-200	98.15	0.00321	0.574	0.9986
CARS	95.40	0.00631	0.588	0.9989
ImageNet-R	96.11	0.00756	0.649	0.9975
ImageNet-A	96.96	0.01420	0.449	0.9983
OmniBenchmark	98.09	0.00933	0.543	0.9980
VTAB	99.81	0.00008	0.078	1.0000
Overall	97.46	0.00715	0.505	0.9981

D Conversion Fidelity: Extended Results

Fidelity by dataset. Table 4 disaggregates the fidelity results of Table 1 by benchmark, averaging over the seven subject methods. Fidelity is uniform across domains as well as across methods: agreement remains above 95% on every benchmark, from 95.40% on CARS, the finest-grained benchmark, to 99.81% on VTAB, and the absolute top-1 gap stays below 0.7 pp throughout. The behavioral equivalence established in Section 4.2 therefore does not depend on a particular data distribution, and no benchmark contributes disproportionately to the averages reported in the main text.

Degradation without realignment. Section 3.3 argues that once an expert has been written into the backbone, reattaching the retained module applies the same adaptation twice, and that the resulting discrepancy compounds across consolidation steps. Figure 3 tests this directly by running the stream with the rebase step removed, so that the method receives the consolidated backbone W_t itself as its next base. The two APER variants are used because they adapt only at the first session and keep their module fixed thereafter, so the module state is identical with and without rebase and any difference in agreement is attributable to the realignment alone. Without rebase, agreement drops immediately after the first write and continues to decline over the stream, ending at 94.61% for APER-VPT and 94.87% for APER-Adapter; with rebase, the same runs end at 98.03% and 96.83%. The duplication is therefore not absorbed by the fixed module: even when the PET state itself never changes, recombining it with a backbone that has absorbed its knowledge requires an explicit correction, and rebase performs that function.

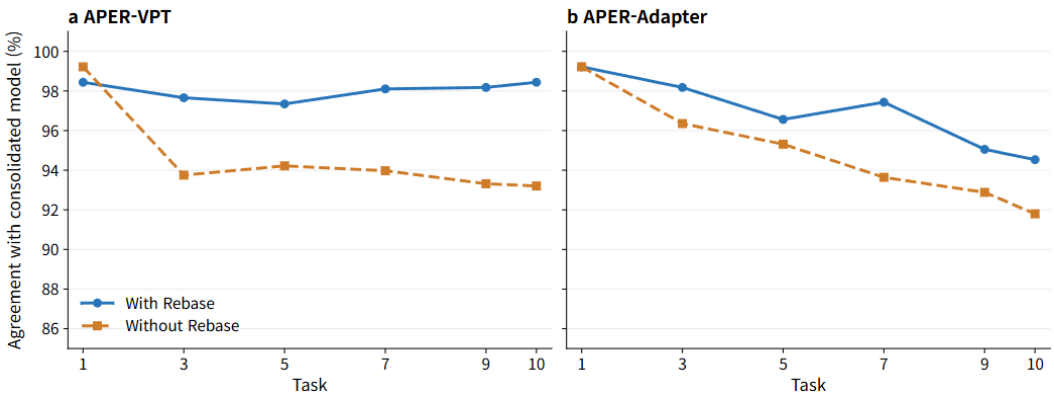


Figure 3: Agreement with the consolidated model at each task with and without the rebase step, for (a) APER-VPT and (b) APER-Adapter on CIFAR100.

Table 5: Full-task AIA of each unmodified method (left of each cell) and of the same method under CoRe (right), as 0–1 proportions. AIA averages all 10 task stages (5 for VTAB)

Method	CIFAR-100	CUB-200	CARS	ImageNet-R	ImageNet-A	OmniBench.	VTAB
APER-VPT	.8964 / .8975	.9031 / .9021	.4809 / .4803	.7788 / .7737	.6485 / .6329	.8157 / .8146	.8877 / .8823
APER-Adapter	.9126 / .9091	.9039 / .9035	.5428 / .5406	.7813 / .7851	.6574 / .6547	.8068 / .8037	.8855 / .8860
L2P	.8739 / .8737	.7677 / .7680	.5594 / .5830	.7699 / .7637	.5264 / .5202	.7610 / .7455	.7737 / .7623
DualPrompt	.8532 / .8560	.8086 / .8067	.5208 / .5358	.7356 / .7409	.5760 / .5414	.7629 / .7573	.8292 / .8391
CODA-Prompt	.9112 / .9097	.8450 / .8434	.5421 / .5318	.7834 / .7742	.6326 / .6112	.7152 / .7089	.8926 / .8812
InfLoRA	.9115 / .9036	.8130 / .8070	.6605 / .6483	.8226 / .8188	.6291 / .6089	.8021 / .7785	.8643 / .8430
LCA	.9215 / .9037	.8568 / .8330	.6744 / .6799	.8309 / .8096	.7131 / .6674	.7761 / .7366	.8652 / .8586

Fidelity under repeated writes. Behavioral equivalence at individual steps does not by itself guarantee that a full stream of recurring writes leaves the continual-learning trajectory intact, so Table 5 compares, for every method and benchmark, the AIA of the unmodified method against that of the same method run under CoRe with the $K = 2$ schedule, matched in seed, class increment, class order, and backbone. The consolidated runs track their unmodified counterparts across the full grid: over the 49 method–benchmark pairs, the two AIA landscapes correlate at Pearson $r = 0.9947$ and Spearman $\rho = 0.9936$ (two-sided $p < 10^{-45}$ in both cases), so the relative standing of methods and the difficulty ordering of benchmarks are preserved under consolidation. These statistics describe association; CoRe makes no claim of matching or improving accuracy, and individual cells deviate in both directions, consistent with Eq. (1) constraining the decision function at each step but not the subsequent optimization path.

E Identifying the Shared Component: Extended Results

Preservation under the interference-regularized write. The write of Eq. (4) damps the directions to which earlier decisions are sensitive, so a natural concern is that it also damps the shared component. We test this directly. For every consolidation step of all seven methods on all seven benchmarks, the distilled update Δ_t saved before the write is compared with the pure fold recomputed from it, with rebase and conditional repair removed so that only the write intervenes; the recomputed folds reproduce the update norms and displacements of the original runs. The shared direction is estimated from the other six methods and held fixed across the comparison. The fold leaves the shared coordinate essentially untouched, retaining over 99.99% of the update's projection on it, while reducing the estimated risk to protected decisions to roughly a third of its unregularized value. Stronger regularization does not change this: sweeping the interference strength up to $\lambda = 100$ drives the past risk below 1% while retention stays above 99.98%, and projecting the update onto the null space of the accumulated past-margin Jacobian, which removes the measured risk almost entirely, retains the shared projection to the same degree. This is the outcome the geometry predicts. The sketch spans at most $P = 8$ directions of a coordinate space of dimension $d \approx 2.1 \times 10^7$, and the shared direction, being specific to the current task, has no cause to lie inside a subspace derived from the margins of earlier tasks; the measurements confirm this expected dissociation and quantify it.

The absolute alignment between an update and its shared direction, about 0.02, is read against two references rather than against unity. In a space of this dimension the cosine between random directions concentrates at zero with standard deviation $1/\sqrt{d} \approx 2 \times 10^{-4}$, two orders of magnitude below the observed value, and independent re-distillations of the same expert, which bound the alignment attainable between functionally equivalent solutions in this coordinate system, reach about 0.06. The shared coordinate is small because task information is redundantly distributed across method-specific components, as Section 4.4 establishes; its significance rests on the transplant and removal interventions there, not on its share of update energy.

F Beyond Analysis: Exploiting the Shared Component

The preceding analyses identify a format-invariant component that carries task signal (Section 4.4) and show that the interference-regularized write leaves it intact (Appendix E). This appendix asks whether these findings have practical bearing, in two steps: whether the component can be estimated

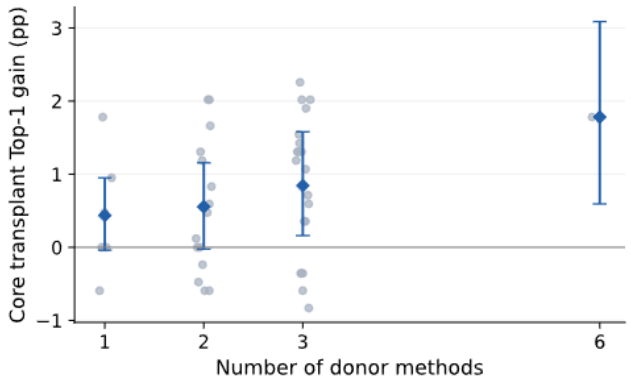


Figure 4: Transplant gain of the shared component as a function of the number of donor methods used to estimate it. CARS, LCA recipient

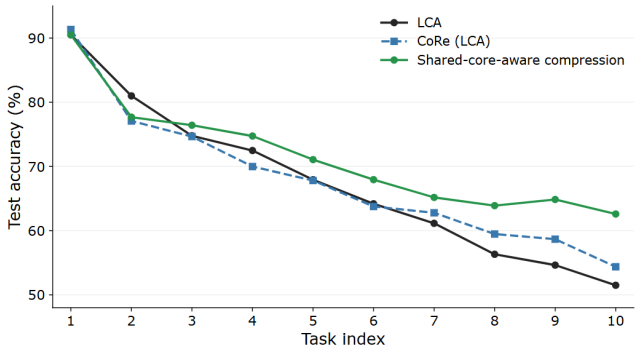


Figure 5: Test Top-1 accuracy over all classes observed through each task on CARS, for LCA, the LCA stream consolidated by CoRe, and the shared-core-aware compressed learner.

from few methods, and whether a minimal learner that uses it is competitive with a recent method of the same low-rank family.

Estimation from few donors. The shared direction has so far been estimated from six donor methods. Figure 4 varies this number: for each donor count n, the direction is estimated from n methods and transplanted as in Section 4.4, with the held-out method fixed. The transplant gain is positive at every count, rising from 0.44 pp with a single donor to 1.8 pp with six, and its confidence interval excludes zero already at $n = 1$. Diversity sharpens the estimate, but it is not a precondition: with one donor the estimate reduces to that donor's converted update direction itself, so a single auxiliary method suffices to obtain a usable shared direction.

Rule. The learner is a rank-64 Q/K/V LoRA trained on each task with cross-entropy on the current backbone. Before the write, the learned update is compressed to the leading 32 singular atoms of each projection, and the atom coefficients are adjusted by the closed-form least change that makes the compressed update reproduce the full update's shared coordinate $c_t^\top a_t$; the result passes through the interference fold of Section 3.2, with the fold correction bounded relative to the update norm, and is written into the backbone. The shared direction c_t is estimated from the converted updates of the other methods; by the donor-count result above, a single auxiliary method would suffice.

Result. On CARS, the rule reaches 71.49% AIA and 62.59% final accuracy, exceeding LCA (Tran et al., 2026) on both measures and the LCA stream consolidated by CoRe throughout the second half of the stream (Figure 5). This demonstrates the feasibility of the shared component as a unit worth acting on: identifying and retaining it during the write is already enough to reach the level of a recent method of the same family.