LANS: TUNING IN LARGER NULL SPACES FOR CONTINUAL LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Continual learning aims to enable models to sequentially acquire knowledge while maintaining a balance between stability and plasticity. To achieve this, null-space-based approaches enforce weight updates to lie in the null spaces of past input features, thereby preserving historical responses. Although this strict constraint offers a strong stability guarantee, it inevitably reduces plasticity, especially when model capacity is limited or the number of tasks grows. In this paper, we revisit the stability condition for the attention mechanism in modern Transformers. Unlike prior approaches that constrain each layer independently, we analyze how updates to the attention parameters affect both the attention affinities and the final output, and derive attention-specific stability conditions. These conditions define a feasible update subspace that is provably larger than the conventional null space, offering greater plasticity while preserving historical attention behavior to first order. Building on these results, we characterize the feasible subspace in closed form and develop **LANS**, a low-rank parameterization that efficiently learns weight increments within the derived constraints. Extensive experiments on multiple class-incremental learning benchmarks demonstrate consistent improvements over existing state-of-the-art approaches. Code is available at `https://anonymous.4open.science/r/lans-207E`.

1 INTRODUCTION

Continual learning (CL) aims to enable models to adapt to evolving environments by continuously acquiring new knowledge from non-stationary data streams. Early efforts in this direction primarily focused on feed-forward and convolutional networks, giving rise to four representative families of approaches: regularization (Li & Hoiem, 2016), replay (Lopez-Paz & Ranzato, 2017), architecture expansion (Yoon et al., 2018), and null-space projection (Zeng et al., 2019; Saha et al., 2021).

Recently, large-scale pre-trained vision transformers (ViTs) (Dosovitskiy et al., 2021) and parameter-efficient fine-tuning (PEFT) techniques, such as visual prompt tuning (Jia et al., 2022) and low-rank adaptation (LoRA) (Hu et al., 2022), have garnered growing interest due to their remarkable generalization capabilities and strong performance across a wide range of vision tasks. Motivated by these advances, researchers have begun to explore the integration of popular PEFT methods into the continual learning setting. Representative efforts introduce additional learnable parameters or modules, including prompts and low-rank adapters (Wang et al., 2022b;a; Smith et al., 2023; Liang & Li, 2024). These approaches are often paired with an input-conditioned selection mechanism that dynamically chooses relevant parameters on the fly.

In contrast, another line of research advocates for continually fine-tuning the pre-trained model without accumulating external modules. Several works in this direction are inspired by null-space projection strategies (Zeng et al., 2019; Saha et al., 2021; Wang et al., 2021), constraining each linear layer's update to the right null space of the corresponding input representation. More recent studies extend this principle to parameter-efficient continual tuning via prompts, adapters, or low-rank updates (Qiao et al., 2024; Lu et al., 2024; Liang & Li, 2024; Qiao et al., 2025). Although this strict orthogonality ensures stability, it progressively shrinks the available update space and reduces plasticity as tasks accumulate. To mitigate this limitation, several attempts have sought to improve plasticity by reusing, relaxing, scaling, or restricting selected projection directions (Lin et al., 2022; Kong et al., 2022; Saha & Roy, 2023). While these strategies partially alleviate the plasticity bottle-

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.

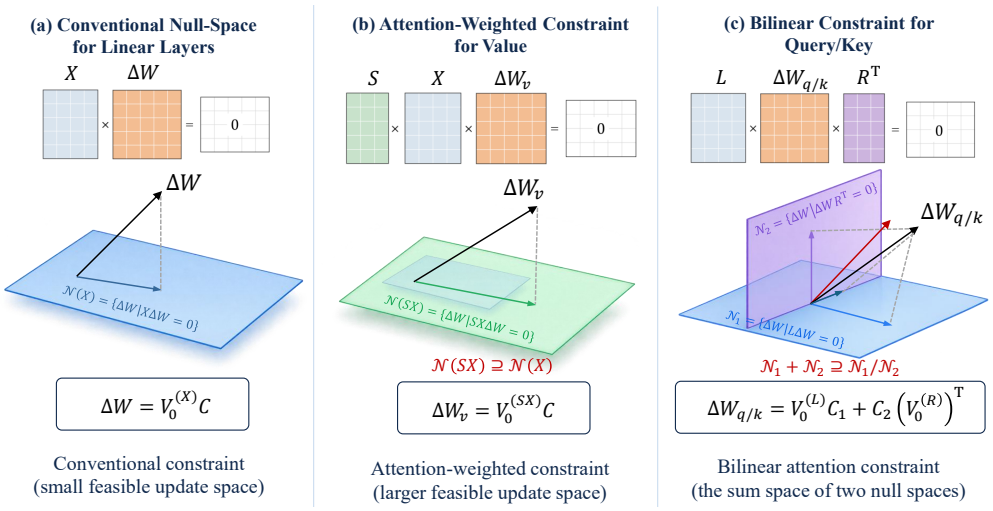


Figure 1: Comparison of conventional null-space methods and our method. (a) Existing methods treat each linear projection independently and constrain its weight update by $X\Delta W = 0$, restricting the update to the null space of the input features. Our formulations in (b) and (c) exploit the interactions within self-attention to derive less restrictive constraints. (b) For the value projection, $SX\Delta W_v = 0$ admits the attention-weighted null space $\mathcal{N}(SX) \supseteq \mathcal{N}(X)$, with strict inclusion when $\mathrm{rank}(SX) < \mathrm{rank}(X)$. (c) For the query and key, the bilinear constraint $L\Delta W_{q/k} R^\top = 0$ admits the sum space $\mathcal{N}_1 + \mathcal{N}_2$, where $\mathcal{N}_1 = \{\Delta W \mid L\Delta W = 0\}$ and $\mathcal{N}_2 = \{\Delta W \mid \Delta W R^\top = 0\}$. Let $V_0^{(X)}$ denote the null-space basis of X. Then the bottom row shows the corresponding null-space parameterizations, with fixed null-space bases $V_0^{(X)}$, $V_0^{(SX)}$, $V_0^{(L)}$, $V_0^{(R)}$, and trainable coefficients C, C_1, and C_2.

neck, the recovered capacity is typically governed by empirical criteria or manually crafted designs, rather than derived from an explicit characterization of a larger, theoretically justified update space. This raises a natural question:

> *Can we identify a larger feasible update space that improves plasticity while retaining theoretically justified stability constraints?*

To address this question, we revisit the stability conditions for self-attention in modern Transformers (Vaswani et al., 2017). Unlike conventional null-space methods that constrain each linear projection independently, we analyze how updates to the query, key, and value projections jointly affect the attention scores and output. This joint analysis reveals that the value and query/key projections admit different stability conditions: value updates ΔW_v must preserve their contribution after attention-weighted aggregation, while query and key updates $\Delta W_{q/k}$ are subject to bilinear constraints that account for their interaction. We then derive closed-form characterizations of these feasible update spaces and establish conditions under which they strictly contain the conventional input null spaces. Intuitively, by accounting for the interactions among projections, the constraints no longer need to hold for each projection in isolation, thereby relaxing the overall restriction and enlarging the feasible update space. This enlargement is illustrated in Fig. 1(b) and (c), in contrast to the conventional linear null space shown in Fig. 1(a).

Building on these theoretical foundations, we propose **LANS** (**LA**rger **N**ull-**S**pace tuning), which learns updates within the derived spaces via fixed subspace bases and trainable coefficients, directly incorporating the constraints into training without explicit gradient projection. As with standard LoRA, the learned updates can be merged into the backbone weights after training, introducing no additional parameter overhead or task-specific routing at inference time. We conduct extensive experiments on multiple benchmarks with different pre-trained backbones. The results demonstrate that LANS consistently outperforms existing null-space-based methods and PEFT-based continual learning approaches, achieving superior plasticity while retaining strong empirical stability.

Our main contributions are summarized as follows:

- We derive attention-specific stability conditions for the query, key, and value projections in the self-attention module, revealing additional update directions that are excluded by conventional layer-wise null-space constraints. We further characterize the closed-form feasible spaces induced by these conditions with rigorous theoretical guarantees.
- We propose **LANS**, which learns weight updates within the derived feasible spaces via fixed subspace bases and trainable coefficients. This parameterization enforces the stability constraints throughout training without requiring explicit gradient projection, while ensuring provably improved plasticity.
- Extensive experiments across multiple class-incremental benchmarks and pre-trained backbones demonstrate that our method consistently outperforms existing approaches, achieving superior plasticity while retaining strong empirical stability.

2 Related Work

Continual Learning. Existing continual learning methods are commonly grouped into regularization-, replay-, architecture-, and optimization-based approaches (Lange et al., 2022). Regularization methods preserve parameters or previous predictions (Kirkpatrick et al., 2017; Li & Hoiem, 2016), whereas replay-based methods retain representative samples or constrain updates using episodic memory (Rebuffi et al., 2017; Chaudhry et al., 2019; Lopez-Paz & Ranzato, 2017). Other approaches reduce classifier bias through representation distillation or post-hoc correction (Hou et al., 2019; Douillard et al., 2020; Zhao et al., 2020), or allocate additional task-dependent capacity (Yoon et al., 2018; Yan et al., 2021). With pre-trained ViTs, parameter-efficient approaches instead learn task-adaptive prompts or low-rank updates (Wang et al., 2022b;a; Smith et al., 2023; Liang & Li, 2024). While effective, task-specific modules may grow with the task sequence and input-conditioned selection can be sensitive to routing errors. We consider the complementary setting in which updates are merged into a shared backbone and stability is enforced through the admissible update space.

Null-Space-Based Methods. Null-space methods reduce interference by projecting updates away from feature subspaces associated with previous tasks. OWM, GPM, and Adam-NSCL construct such subspaces from historical inputs, gradient-relevant bases, or feature covariances (Zeng et al., 2019; Saha et al., 2021; Wang et al., 2021). Subsequent methods improve plasticity by reusing, enlarging, scaling, or restricting selected projection directions (Lin et al., 2022; Kong et al., 2022; Saha & Roy, 2023). The journal extension of Adam-NSCL further explores adaptively scaled projections and complementary self-supervised objectives (Wang et al., 2025). More recently, null-space constraints have been incorporated into prompt, adapter, and low-rank continual tuning (Qiao et al., 2024; Lu et al., 2024; Liang & Li, 2024; Qiao et al., 2025; Zhang et al., 2026; Luo et al., 2026). Despite different parameterizations, these methods largely impose the layer-wise conditions $X_\tau \Delta W = 0$ for all $\tau \leq t$, whose common feasible space shrinks as historical features accumulate. In contrast, LANS derives stability conditions for the complete self-attention operation, exploiting attention-weighted value aggregation and query–key coupling to identify additional stability-preserving directions.

3 Preliminaries

3.1 Problem Definition

Consider a task stream $\{\mathcal{T}_1, \mathcal{T}_2, \ldots\}$, where each task $\mathcal{T}_\tau$ is associated with a dataset $\mathcal{D}_\tau = \{(X_{\tau,j}, Y_{\tau,j})\}_{j=1}^{n_\tau}$. Let $W_t = \{W_t^1, \ldots, W_t^L\}$ denote the parameters of an L-layer model f after completing task $\mathcal{T}_t$. Thus, t indexes the latest task learned by the current model, whereas τ indexes an arbitrary historical task with $\tau \leq t$. When learning the new task $\mathcal{T}_{t+1}$, we solve:

$$\min_{W_{t+1}} \mathcal{L}\big(f(X_{t+1}, W_{t+1}), Y_{t+1}\big), \tag{1}$$

$$\text{s.t.} \quad f(X_\tau, W_{t+1}) = f(X_\tau, W_t), \quad \forall \tau \in \{1, \ldots, t\}. \tag{2}$$

The objective (1) captures plasticity, whereas Eq. (2) requires the updated model W_{t+1} to match the pre-update model W_t on every task $\mathcal{T}_\tau$ with $\tau \leq t$.

3.2 Layer-wise Linear Null Space for CL

Directly enforcing Eq. (2) is challenging because f is nonlinear and historical samples may be unavailable. A practical relaxation preserves the response of each linear layer on historical features:

$$X_\tau^l W_t^l = X_\tau^l W_{t+1}^l, \quad \forall \tau \leq t,\ l \leq L, \tag{3}$$

where X_τ^l denotes the input features of layer l extracted from task $\mathcal{T}_\tau$ by the pre-update model W_t. Omitting the layer index and defining the current update as $\Delta W = W_{t+1} - W_t$, Eq. (3) becomes

$$X_\tau \Delta W = 0, \quad \forall \tau \leq t. \tag{4}$$

For an arbitrary historical task τ, Eq. (4) requires every column of ΔW to lie in the right null space of X_τ. Conventional methods (Saha et al., 2021; Wang et al., 2021) enforce this requirement by projecting an optimizer update ΔW onto the corresponding null space:

$$\Delta W' \leftarrow V_0^{(X_\tau)} (V_0^{(X_\tau)})^\top \Delta W, \tag{5}$$

where $V_0^{(X_\tau)} \in \mathbb{R}^{d \times r_x}$ is the right-null-space basis of X_τ and r_x is its nullity. Alternatively, the update can be parameterized directly within the desired null space:

$$\Delta W = V_0^{(X_\tau)} C, \tag{6}$$

where $C \in \mathbb{R}^{r_x \times d'}$ is an arbitrary matrix that can be learned end-to-end. This reparameterization is adopted in recent LoRA-based continual learning works (Liang & Li, 2024; Luo et al., 2026).

Recent advances have extended null-space projection to other parameter-efficient fine-tuning techniques, including adapter tuning and prompt tuning (Qiao et al., 2024; Lu et al., 2024; Qiao et al., 2025). Despite these efforts, their theoretical foundation remains essentially the same as Eq. (4): updates must satisfy increasingly many historical feature constraints. As more tasks are learned, the admissible directions therefore shrink progressively, limiting model plasticity. Some heuristic (Lin et al., 2022; Kong et al., 2022; Saha & Roy, 2023) attempts relax this restriction using task correlations or parameter importance, but without explicitly characterizing a larger exact stability-preserving update space.

This paper aims to derive stability conditions that are provably less restrictive than the conventional null-space constraint, thereby preserving the stability while enabling substantially greater plasticity.

4 Method

In this section, we begin by revisiting the attention module in modern Transformers, from which we derive stability conditions for the query, key and value projections. Based on these conditions, we characterize the resulting space structures and propose the corresponding learning algorithm.

4.1 Stability Conditions for the Self-Attention Module

Unlike feed-forward networks (FFNs), self-attention contains structured interactions among the query, key, and value projections. For notational simplicity, we analyze single-head attention and omit the head index throughout this subsection. The resulting conditions extend naturally to multi-head attention by applying them independently to each head (Appendix A.5).

Formally, given a sequence of n tokens represented by $X \in \mathbb{R}^{n \times d}$, the output of a single-head self-attention module is

$$\text{Self-Attn}(X) = \text{SoftMax}\left(\frac{QK^\top}{\sqrt{d'}}\right) V = \text{SoftMax}\left(\frac{XW_q W_k^\top X^\top}{\sqrt{d'}}\right) XW_v \tag{7}$$

where $W_{\{q,k,v\}} \in \mathbb{R}^{d \times d'}$ are learnable projection matrices, $Q = XW_q$, $K = XW_k$, and $V = XW_v$ are the query, key, and value matrices, respectively. The above computation can be decomposed into

three sequential operations: **affinity**, **softmax** (applied row-wise), and **aggregation**:

$$\begin{cases} A = XW_qW_k^\top X^\top \in \mathbb{R}^{n\times n}, \\ S = \text{SoftMax}(A/\sqrt{d'}) \in \mathbb{R}^{n\times n}, \\ Z = SXW_v \in \mathbb{R}^{n\times d'}. \end{cases} \tag{8}$$

In conventional null-space-based approaches, the stability constraint mandates that the parameter update must lie in the null space of the historical input features. Applied to the attention module, this constraint is imposed simultaneously on all three projection matrices:

$$\Delta Q = X_\tau \Delta W_q = 0, \quad \Delta K = X_\tau \Delta W_k = 0, \quad \Delta V = X_\tau \Delta W_v = 0, \tag{9}$$

where X_τ denotes the input features of the attention module from an arbitrary historical task $\mathcal{T}_\tau$ ($\tau \le t$), and ΔW_i ($i \in \{q, k, v\}$) denotes the update to W_i while learning $\mathcal{T}_{t+1}$. Although this uniform constraint preserves all three projected features, it imposes an unnecessarily rigid bottleneck on plasticity because Q, K, and V play distinct roles within attention.

To alleviate this rigidity, we derive less restrictive conditions at the level of the attention computation. To preserve the historical attention output Z, we separately eliminate its two sources of variation. First, we require the query–key updates to induce no first-order variation in the affinity logits A, thereby preserving the historical attention weights S. With S preserved, we then require the value update to induce no variation in the aggregated output. These jointly sufficient conditions are

$$\begin{cases} \Delta A = X_\tau W_q \Delta W_k^\top X_\tau^\top + X_\tau \Delta W_q W_k^\top X_\tau^\top = 0, & \text{(10a)} \\ \Delta Z = S_\tau X_\tau \Delta W_v = 0. & \text{(10b)} \end{cases}$$

Here, W_q, W_k, and W_v denote the pre-update projections, and $S_\tau = \text{SoftMax}(X_\tau W_q W_k^\top X_\tau^\top/\sqrt{d'})$ is the attention-weight matrix computed from historical inputs X_τ. ΔA denotes the first-order affinity variation, obtained by omitting the second-order interaction $X_\tau \Delta W_q \Delta W_k^\top X_\tau^\top$.

(a) Linear constraint for ΔW_v with a larger null space. The second condition, $S_\tau X_\tau \Delta W_v = 0$, generalizes the traditional constraint $X_\tau \Delta W_v = 0$ in Eq. (9) by incorporating the attention weights S_τ. This relaxation enlarges the feasible subspace, since

$$\mathcal{N}(S_\tau X_\tau) = \{\Delta W \mid S_\tau X_\tau \Delta W = 0\} \supseteq \mathcal{N}(X_\tau) = \{\Delta W \mid X_\tau \Delta W = 0\}, \tag{11}$$

with strict inclusion if and only if $\text{rank}(S_\tau X_\tau) < \text{rank}(X_\tau)$. Under this condition, constraining ΔW_v to $\mathcal{N}(S_\tau X_\tau)$ rather than to $\mathcal{N}(X_\tau)$ admits additional update directions, thereby providing greater plasticity.

(b) Bilinear constraints for ΔW_q and ΔW_k. Eq. (10a) preserves the historical affinity matrix by allowing the variations induced by ΔW_q and ΔW_k to cancel each other. Although this yields the most general stability-preserving updates, it couples the two variables, making the feasible space and an efficient parameterization hard to characterize.[1] We thus first adopt a separable sufficient condition requiring each variation to vanish independently, which decouples the problem into two bilinear constraints:

$$X_\tau \Delta W_q K_\tau^\top = 0, \quad X_\tau \Delta W_k Q_\tau^\top = 0. \tag{12}$$

Both constraints in Eq. (12) share the following canonical form $L\Delta W R^\top = 0$, where L and R are two (potentially different) feature matrices. This common form enables a unified treatment of both constraints while still being less restrictive than the standard null-space constraint. To fully characterize the structure of its solution space, we present the following theorems.

Theorem 1 (Bilinear Null-Space Characterization). *Let $L \in \mathbb{R}^{n\times d}$ and $R \in \mathbb{R}^{m\times d'}$ be two given matrices of ranks k_1 and k_2, respectively. We consider the tall-matrix regime,[2] where $n > d$ and $m > d'$. Let $V_0^{(L)} \in \mathbb{R}^{d\times r_1}$ and $V_0^{(R)} \in \mathbb{R}^{d'\times r_2}$ be bases of the right null spaces of L and R, where $r_1 = d - k_1$ and $r_2 = d' - k_2$. Denote*

$$\mathcal{N}_1 = \{\Delta W \in \mathbb{R}^{d\times d'} \mid L\Delta W = 0\}, \quad \textit{and } \mathcal{N}_2 = \{\Delta W \in \mathbb{R}^{d\times d'} \mid \Delta W R^\top = 0\}. \tag{13}$$

Then the solution space for $L\Delta W R^\top = 0$ is the sum space of $\mathcal{N}_1$ and $\mathcal{N}_2$:

$$\mathcal{N}_+ = \mathcal{N}_1 + \mathcal{N}_2 = \{V_0^{(L)} C_1 + C_2 (V_0^{(R)})^\top \mid C_1 \in \mathbb{R}^{r_1\times d'},\ C_2 \in \mathbb{R}^{d\times r_2}\}. \tag{14}$$

[1] The coupled case is deferred to Appendix A.4.

[2] In practice, the rows of L and R aggregate token-level observations over samples and historical tasks rather than representing a single sample length; consequently, this regime normally holds.

Remark 1. Theorem 1 shows that the feasible space of $L\Delta W R^\top = 0$ is the sum $\mathcal{N}_1 + \mathcal{N}_2$. Compared with the conventional space $\mathcal{N}_1$ induced by $L\Delta W = 0$, this bilinear constraint provides $k_1 r_2$ additional degrees of freedom and yields a strict enlargement whenever $k_1 > 0$ and $r_2 > 0$.

Theorem 2 (Dimension Gain). *Under the notation of Theorem 1,* $\dim(\mathcal{N}_+) = dd' - k_1 k_2$, *and*

$$\dim(\mathcal{N}_+) - \dim(\mathcal{N}_1) = k_1 r_2, \qquad \dim(\mathcal{N}_+) - \dim(\mathcal{N}_2) = r_1 k_2. \tag{15}$$

Each inclusion $\mathcal{N}_i \subseteq \mathcal{N}_+$ is strict if and only if the corresponding gain is positive.

Remark 2. For a single attention head, let r_x, r_q, r_k, and r_{sx} denote the nullities of X_τ, Q_τ, K_τ, and $S_\tau X_\tau$, respectively. The input-null-space constraints used by InfLoRA admit $D_{\text{InfLoRA}} = 3r_x d'$ degrees of freedom across the query, key, and value updates. Under the separable query–key constraints, LANS instead admits

$$D_{\text{LANS}} = \underbrace{r_x d' + (d - r_x) r_k}_{\text{query}} + \underbrace{r_x d' + (d - r_x) r_q}_{\text{key}} + \underbrace{r_{sx} d'}_{\text{value}}. \tag{16}$$

Consequently, the dimension gain is

$$D_{\text{LANS}} - D_{\text{InfLoRA}} = (d - r_x)(r_k + r_q) + (r_{sx} - r_x) d' \geq 0, \tag{17}$$

where $r_{sx} \geq r_x$ follows from $\mathcal{N}(X_\tau) \subseteq \mathcal{N}(S_\tau X_\tau)$. The first term quantifies the additional query–key directions, while the second captures the gain from attention-weighted value aggregation. These dimensions describe the feasible spaces before any rank-budget truncation.

4.2 Larger Null-Space Tuning with Parameterization

In the context of parameter-efficient fine-tuning, it is desirable to learn weight increments in parameterized form rather than update entire weight matrices. Building on Theorem 1, we directly parameterize increments within the desired feasible spaces, avoiding the construction and projection of full updates. Specifically, ΔW_q and ΔW_k are expressed as

$$\Delta W_q = V_0^{(X_\tau)} C_{q,1} + C_{q,2} \big(V_0^{(K_\tau)}\big)^\top, \tag{18}$$

$$\Delta W_k = V_0^{(X_\tau)} C_{k,1} + C_{k,2} \big(V_0^{(Q_\tau)}\big)^\top, \tag{19}$$

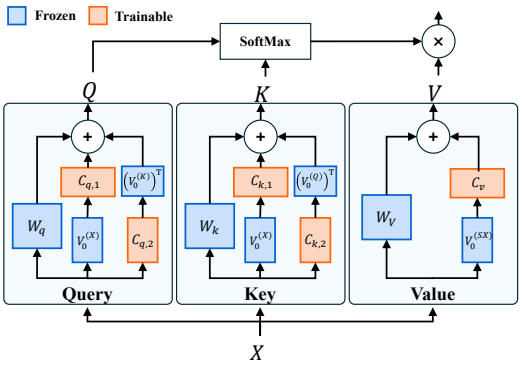


Figure 2: LANS forward propagation.

where $C_{q,1} \in \mathbb{R}^{r_x \times d'}$, $C_{q,2} \in \mathbb{R}^{d \times r_k}$, $C_{k,1} \in \mathbb{R}^{r_x \times d'}$, and $C_{k,2} \in \mathbb{R}^{d \times r_q}$ are trainable coefficient matrices optimized end-to-end during the new task. The bases $V_0^{(X_\tau)} \in \mathbb{R}^{d \times r_x}$, $V_0^{(K_\tau)} \in \mathbb{R}^{d' \times r_k}$, and $V_0^{(Q_\tau)} \in \mathbb{R}^{d' \times r_q}$ are constructed from accumulated historical covariance statistics before each task and held fixed while that task is learned. The parameterization for W_v is simpler, as it only involves a single linear constraint. We can directly parameterize the update within the null space of $S_\tau X_\tau$:

$$\Delta W_v = V_0^{(S_\tau X_\tau)} C_v, \tag{20}$$

where $C_v \in \mathbb{R}^{r_{sx} \times d'}$ is a trainable coefficient matrix, and $V_0^{(S_\tau X_\tau)} \in \mathbb{R}^{d \times r_{sx}}$ is the null-space basis of $S_\tau X_\tau$. The full forward propagation of LANS is illustrated in Fig. 2.

Training Procedure. For the first task, we learn standard LoRA updates $\Delta W = AB$, with both matrices optimized jointly. After finishing one task, the learned updates are merged into the backbone, and the feature covariances at each layer are accumulated to compute the null-space bases. During the subsequent task, we extract approximate null-space bases from these statistics and then parameterize the updates using Eqs. (18, 19, 20). All trainable coefficient matrices are initialized to zero and optimized end-to-end, while the subspace bases and backbone weights remain frozen. More details are provided in Appendix B.

Table 1: Results with ViT-B/16-IN21K on three 10-task benchmarks. Results are averaged over five runs. Bold and underline denote the best and second-best continual-learning results in each column, respectively. Exp. indicates whether a method introduces expandable parameters or modules.

Method	Exp.	CIFAR-100		ImageNet-R		CUB-200	
		FAA (↑)	CAA (↑)	FAA (↑)	CAA (↑)	FAA (↑)	CAA (↑)
Joint	-	93.22 ± 0.16	-	80.76 ± 0.73	-	89.00 ± 0.34	-
L2P	✓	82.76 ± 1.17	88.48 ± 0.83	66.49 ± 0.40	72.83 ± 0.56	62.21 ± 1.92	73.83 ± 1.67
DualPrompt	✓	85.56 ± 0.33	90.33 ± 0.33	68.50 ± 0.52	72.59 ± 0.24	66.00 ± 0.57	77.92 ± 0.50
CODA-Prompt	✓	86.56 ± 0.77	90.61 ± 0.36	75.25 ± 0.56	81.26 ± 0.76	72.63 ± 0.76	80.54 ± 0.54
HiDe-Prompt	✓	88.39 ± 0.49	92.09 ± 0.28	75.66 ± 0.18	78.41 ± 0.17	<u>87.13 ± 0.15</u>	87.53 ± 0.23
EASE	✓	88.17 ± 0.38	92.96 ± 0.48	76.25 ± 0.34	79.03 ± 0.61	86.47 ± 0.23	91.60 ± 1.05
MoTE	✓	88.52 ± 0.15	92.62 ± 0.16	74.32 ± 0.13	79.58 ± 0.16	86.75 ± 0.24	89.65 ± 1.69
FLY-CL	✗	89.72 ± 0.20	92.97 ± 0.83	69.26 ± 0.66	76.57 ± 0.76	86.07 ± 0.47	89.37 ± 0.67
SLCA	✗	91.53 ± 0.28	94.09 ± 0.87	77.00 ± 0.33	81.17 ± 0.64	84.71 ± 0.40	90.94 ± 0.68
SLCA++	✗	91.46 ± 0.18	<u>94.20 ± 0.71</u>	78.09 ± 0.22	<u>82.95 ± 0.78</u>	86.59 ± 0.29	<u>91.63 ± 0.72</u>
InfLoRA	✗	87.06 ± 0.25	91.59 ± 0.13	75.65 ± 0.14	80.82 ± 0.24	80.26 ± 1.21	84.04 ± 0.55
VPT-NSP^2	✗	<u>91.74 ± 0.63</u>	93.24 ± 0.35	<u>78.88 ± 0.50</u>	81.57 ± 0.32	82.06 ± 0.68	88.66 ± 0.82
LANS	✗	**92.02 ± 0.14**	**94.91 ± 0.55**	**79.05 ± 0.41**	**83.28 ± 0.37**	**88.67 ± 0.49**	**92.01 ± 0.65**

5 EXPERIMENTS

We evaluate class-incremental learning on CIFAR-100 (Krizhevsky, 2009), ImageNet-R (Hendrycks et al., 2021), and CUB-200 (Wah et al., 2011). Each benchmark is divided into 10 disjoint tasks with an equal number of classes per task, and task identity is unavailable at inference time. Unless stated otherwise, we report the mean and standard deviation over five runs with different random seeds. Our primary backbone is ViT-B/16 (Dosovitskiy et al., 2021) pre-trained on ImageNet-21K (Deng et al., 2009). We additionally use the DeiT-S/16 architecture (Touvron et al., 2021) with the ImageNet-611 pre-trained weights introduced by Kim et al. (2023), for which classes overlapping with the downstream benchmarks were removed from pre-training. Its smaller representation dimension provides a capacity-stress setting in which accumulated feature matrices approach full rank more rapidly, making differences in the update space easier to observe.

5.1 MAIN RESULTS

Results on ViT-B/16-IN21K. Table 1 shows that LANS achieves the best continual-learning results across all three benchmarks. On CIFAR-100, it reaches 92.02 FAA and 94.91 CAA, improving over the strongest baselines by 0.28 and 0.71 percentage points, respectively. The corresponding gains are $0.17/0.33$ points on ImageNet-R and $1.54/0.38$ points on CUB-200. Compared directly with InfLoRA, a closely related low-rank null-space method, LANS improves FAA/CAA by $4.96/3.32$, $3.40/2.46$, and $8.41/7.97$ points on the three datasets, respectively. These consistent gains support our central claim that attention-aware constraints preserve more useful update directions without relying on task-specific backbone expansion.

Results on DeiT-S/16-IN611. The advantage of LANS becomes clearer with DeiT-S/16-IN611, which has fewer parameters and is pre-trained on the substantially smaller ImageNet-611 dataset rather than ImageNet-21K. As shown in Table 2, LANS ranks first in every column. Relative to the strongest competing method for each metric, it improves FAA/CAA by $3.17/2.45$ points on CIFAR-100, $1.09/3.61$ points on ImageNet-R, and $1.92/4.65$ points on CUB-200. Compared specifically with InfLoRA, LANS improves FAA/CAA by $15.59/11.24$, $6.75/5.03$, and $27.72/20.66$ points on the three datasets,

Table 2: Results with DeiT-S/16-IN611 on 10-Split CIFAR-100, ImageNet-R and CUB-200.

Method	CIFAR-100		ImageNet-R		CUB-200	
	FAA	CAA	FAA	CAA	FAA	CAA
L2P	67.47	78.46	46.59	53.02	42.36	56.74
DualPrompt	67.88	78.37	49.94	54.99	45.35	58.02
CODA-Prompt	64.80	76.11	53.43	63.68	44.72	57.58
HiDe-Prompt	72.42	81.63	53.68	59.82	63.79	69.72
EASE	66.29	77.04	50.23	64.54	48.84	59.62
MoTE	67.47	78.45	49.05	54.84	48.52	58.26
FLY-CL	71.77	80.82	45.07	48.47	57.35	65.34
SLCA	75.16	<u>83.89</u>	<u>60.73</u>	65.87	69.66	73.55
SLCA++	<u>75.41</u>	83.32	60.22	<u>66.90</u>	<u>70.66</u>	<u>75.04</u>
InfLoRA	62.99	75.10	55.07	65.48	44.86	59.03
VPT-NSP^2	70.83	80.24	52.56	63.37	35.98	50.76
LANS	**78.58**	**86.34**	**61.82**	**70.51**	**72.58**	**78.69**

Table 3: Results on long task sequences with two pre-trained backbones. Higher FAA and CAA are better. **Bold** and underline denote the best and second-best results, respectively.

Method	ViT-B/IN21K 20-Split				ViT-B/IN21K 50-Split				DeiT-S/IN611 20-Split				DeiT-S/IN611 50-Split			
	CIFAR-100		ImageNet-R		CIFAR-100		ImageNet-R		CIFAR-100		ImageNet-R		CIFAR-100		ImageNet-R	
	FAA	CAA	FAA	CAA	FAA	CAA	FAA	CAA	FAA	CAA	FAA	CAA	FAA	CAA	FAA	CAA
L2P	81.64	87.56	63.77	72.40	66.41	74.97	54.11	64.97	55.86	69.84	41.99	52.98	39.80	56.60	34.12	46.80
DualPrompt	81.00	86.69	60.42	67.92	67.44	76.96	50.92	58.79	56.42	69.39	46.98	56.79	42.17	55.88	40.84	49.91
CODA-Prompt	81.77	87.82	66.82	74.91	56.55	69.29	50.22	59.81	57.37	70.67	46.02	57.24	34.98	49.03	33.94	44.79
EASE	84.69	89.92	74.05	81.21	79.19	85.76	67.78	76.67	60.26	73.80	47.18	57.94	57.25	70.74	40.93	52.83
SLCA++	89.78	93.82	76.00	81.57	83.82	90.82	65.33	75.65	72.81	81.96	56.78	66.89	65.68	78.60	47.05	59.85
InfLoRA	83.37	89.52	72.32	79.95	56.67	70.56	61.38	70.97	50.46	67.39	47.62	60.51	26.13	48.95	26.47	40.31
VPT-NSP^2	88.50	92.69	76.63	82.71	79.99	85.76	70.37	76.03	63.51	75.04	45.20	57.73	48.96	62.19	33.28	44.62
LANS	**90.57**	**94.18**	**77.38**	**83.15**	**86.30**	**91.11**	**71.37**	**77.85**	**76.74**	**84.61**	**58.67**	**67.72**	**73.24**	**82.36**	**52.15**	**62.29**

respectively. These larger margins show that LANS remains effective when the backbone offers less capacity and weaker transferable representations, a setting in which preserving sufficient freedom for continual adaptation is especially important.

Results on Long Task Sequences. Table 3 evaluates 20-task and 50-task sequences on CIFAR-100 and ImageNet-R with both pre-trained backbones. LANS achieves the highest FAA and CAA in all settings. With ViT-B/16-IN21K, its FAA/CAA gains over the strongest competing result for each metric on 50-task sequences are $2.48/0.29$ points on CIFAR-100 and $1.00/1.18$ points on ImageNet-R. The advantage is more pronounced with the smaller, less extensively pre-trained DeiT-S/16-IN611: on 50-task sequences, LANS outperforms SLCA++, the strongest baseline in these settings, by $7.56/3.76$ and $5.10/2.44$ points, respectively. These margins exceed those on the corresponding 20-task sequences ($3.93/2.65$ and $1.89/0.83$ points). Although performance decreases as the sequences become longer, LANS remains consistently competitive across both backbones, supporting the effectiveness of its enlarged update spaces under repeated adaptation.

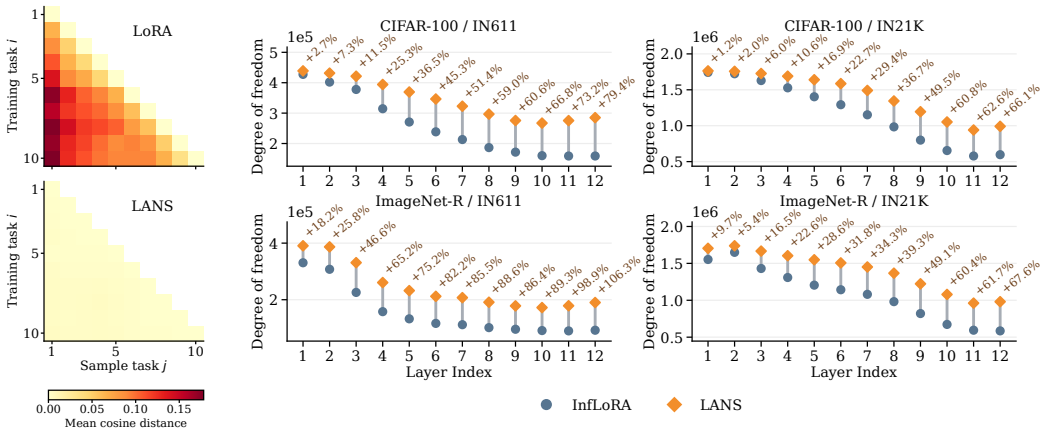
5.2 Ablation and Analysis

Component Ablation. Table 4 evaluates all combinations of the LANS query, key, and value modules, with ordinary LoRA used for the remaining projections. The full configuration achieves the best FAA and CAA, and removing any module reduces performance, supporting their complementary contributions. Among individual modules, the value module provides the largest gains over LoRA: $5.37/1.92$ points on CIFAR-100 and $8.43/4.33$ points on ImageNet-R. The benefits of the query and key modules become more pronounced when combined with the value module. Adding either improves all four metrics, while adding both yields further FAA/CAA gains of $2.24/2.32$ and $5.17/2.45$ points over value-only adaptation, respectively. A plausible explanation lies in their distinct computational roles: values carry the content aggregated into the output, whereas queries and keys determine how it is weighted. The larger value-side gains are consistent with the benefits of directly adapting representational content while limiting interference with historical outputs; the additional query–key gains suggest that adjusting the associated attention weights further improves content utilization.

Table 4: Component ablations with DeiT-S/16-IN611. A checkmark denotes LANS; an unmarked projection uses ordinary LoRA. All variants use the same classifier-calibration protocol.

LANS modules			CIFAR-100		ImageNet-R	
ΔW_q	ΔW_k	ΔW_v	FAA	CAA	FAA	CAA
			70.97	82.10	48.22	63.73
✓			71.35	82.27	49.75	64.45
	✓		70.81	82.07	50.52	64.48
		✓	76.34	84.02	56.65	68.06
✓	✓		74.23	83.30	54.65	65.74
✓		✓	78.57	86.10	60.58	69.65
	✓	✓	78.28	86.00	60.75	70.08
✓	✓	✓	78.58	86.34	61.82	70.51

Stability–Plasticity Analysis. We assess stability using the mean cosine distance between task-j representations after tasks j and i ($i > j$). In Fig. 4a, LANS exhibits consistently lower drift than LoRA across all historical task–stage pairs on CIFAR-100 with DeiT-S/16-IN611. Averaged over



(a) Stability (b) Plasticity

Figure 4: Stability–plasticity analysis. (a) Representation drift of LoRA (top) and LANS (bottom) on CIFAR-100 with DeiT-S/16-IN611. Entry (i, j) is the mean cosine distance of task-j features between stages j and i. (b) Feasible degrees of freedom on CIFAR-100 (top) and ImageNet-R (bottom), with DeiT-S/16-IN611 (left) and ViT-B/16-IN21K (right).

the off-diagonal entries, the drift decreases from 0.09958 to 0.00224, a 97.8% reduction. After the final task, the mean drift over the nine preceding tasks remains only 0.00246, compared with 0.09897 for LoRA, demonstrating sustained preservation of historical representations. To assess plasticity, we measure the dimensions of the feasible query, key, and value update spaces. Fig. 4b shows that LANS provides more degrees of freedom than InfLoRA at every layer across both datasets and backbones, with larger relative gains generally observed in deeper layers. At the final layer, the gains reach $79.4\%/66.1\%$ on CIFAR-100 and $106.3\%/67.6\%$ on ImageNet-R for IN611/IN21K, respectively. Together, these results show that LANS combines strong empirical representation stability with an enlarged feasible update space, retaining greater capacity for adaptation without the substantial historical drift observed under unconstrained LoRA.

Update Direction Alignment. We compare task-wise query and key weight increments with unconstrained LoRA on ImageNet-R using DeiT-S/16-IN611. For each projection, we concatenate increments across all 12 layers and compute their cosine similarity to the corresponding LoRA update. The comparison uses an InfLoRA-form input-null-space baseline implemented within the same training framework. As shown in Fig. 3, LANS achieves higher alignment for both projections across tasks 2–10, with mean similarities of $0.225/0.235$ for query/key updates, compared with $0.140/0.122$ for the baseline. These results show that LANS learns update directions more closely aligned with unconstrained adaptation.

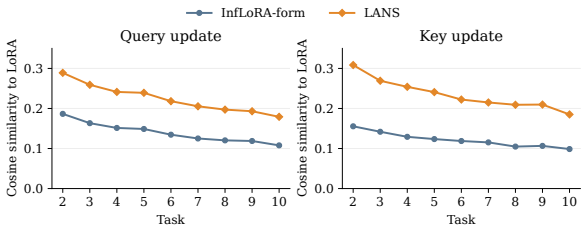


Figure 3: Query and key update alignment with unconstrained LoRA.

6 Conclusion

We presented LANS, an attention-aware null-space tuning framework for continual learning with pre-trained Transformers. By exploiting the structure of self-attention, we derived exact value-stability constraints for fixed attention weights and first-order query–key affinity constraints, and characterized the resulting feasible spaces and their dimension gains over conventional input-null-space constraints. LANS learns parameterized updates within these spaces and merges them into the backbone without additional inference-time overhead. Experiments demonstrate consistent performance improvements, particularly under weaker pre-training and longer task sequences, while representation-drift and feasible-space analyses provide complementary evidence of historical stability and greater capacity for adaptation.

AI USE STATEMENT

In this work, we used generative AI tools for polishing the manuscript. We have reviewed all AI-assisted content. Specifically, all AI-generated or AI-polished text was carefully checked and revised by the authors to ensure factual accuracy and alignment with our technical claims. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REFERENCES

Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with A-GEM. In *International Conference on Learning Representations*, 2019.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009.

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*, 2021.

Arthur Douillard, Matthieu Cord, Charles Ollion, Thomas Robert, and Eduardo Valle. PODNet: Pooled outputs distillation for small-tasks incremental learning. In *European Conference on Computer Vision*, pp. 86–102, 2020.

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 *IEEE/CVF International Conference on Computer Vision*, pp. 8320–8329, 2021. doi: 10.1109/ICCV48922.2021.00823.

Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Learning a unified classifier incrementally via rebalancing. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 831–839, 2019.

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

Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge J. Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning. In *European Conference on Computer Vision*, pp. 709–727, 2022. doi: 10.1007/978-3-031-19827-4_41.

Gyuhak Kim, Changnan Xiao, Tatsuya Konishi, and Bing Liu. Learnability and algorithm for continual learning. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 16877–16896. PMLR, 2023.

James Kirkpatrick, Razvan Pascanu, Neil C. 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.

Yajing Kong, Liu Liu, Zhen Wang, and Dacheng Tao. Balancing stability and plasticity through advanced null space in continual learning. In *European Conference on Computer Vision*, pp. 219–236, 2022. doi: 10.1007/978-3-031-19809-0_13.

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

Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Ales Leonardis, Gregory G. Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(7):3366–3385, 2022. doi: 10.1109/TPAMI.2021.3057446.

Linjie Li, Zhenyu Wu, and Yang Ji. MoTE: Mixture of task-specific experts for pre-trained model-based class-incremental learning. *Knowledge-Based Systems*, 324:113795, 2025. doi: 10.1016/j.knosys.2025.113795.

Zhizhong Li and Derek Hoiem. Learning without forgetting. In *European Conference on Computer Vision*, pp. 614–629, 2016. doi: 10.1007/978-3-319-46493-0_37.

Yan-Shuo Liang and Wu-Jun Li. InfLoRA: Interference-free low-rank adaptation for continual learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 23638–23647, 2024. doi: 10.1109/CVPR52733.2024.02231.

Sen Lin, Li Yang, Deliang Fan, and Junshan Zhang. TRGP: Trust region gradient projection for continual learning. In *International Conference on Learning Representations*, 2022.

David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. In *Advances in Neural Information Processing Systems*, pp. 6467–6476, 2017.

Yue Lu, Shizhou Zhang, De Cheng, Yinghui Xing, Nannan Wang, Peng Wang, and Yanning Zhang. Visual prompt tuning in null space for continual learning. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Mao-Lin Luo, Zi-Hao Zhou, Yi-Lin Zhang, Yuanyu Wan, Tong Wei, and Min-Ling Zhang. KeepLoRA: Continual learning with residual gradient adaptation. *arXiv preprint arXiv:2601.19659*, 2026.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Z. Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, pp. 8024–8035, 2019.

Jingyang Qiao, Zhizhong Zhang, Xin Tan, Chengwei Chen, Yanyun Qu, Yong Peng, and Yuan Xie. Prompt gradient projection for continual learning. In *International Conference on Learning Representations*, 2024.

Jingyang Qiao, Zhizhong Zhang, Xin Tan, Yanyun Qu, Wensheng Zhang, Zhi Han, and Yuan Xie. Gradient projection for continual parameter-efficient tuning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(10):9316–9329, 2025. doi: 10.1109/TPAMI.2025.3587032.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H. Lampert. iCaRL: Incremental classifier and representation learning. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5533–5542, 2017.

Gobinda Saha and Kaushik Roy. Continual learning with scaled gradient projection. In *AAAI Conference on Artificial Intelligence*, pp. 9677–9685, 2023. doi: 10.1609/AAAI.V37I8.26157.

Gobinda Saha, Isha Garg, and Kaushik Roy. Gradient projection memory for continual learning. In *International Conference on Learning Representations*, 2021.

James Seale Smith, Leonid Karlinsky, Vyshnavi Gutta, Paola Cascante-Bonilla, Donghyun Kim, Assaf Arbelle, Rameswar Panda, Rogério Feris, and Zsolt Kira. CODA-Prompt: Continual decomposed attention-based prompting for rehearsal-free continual learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11909–11919, 2023. doi: 10.1109/CVPR52729.2023.01146.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, pp. 10347–10357, 2021.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pp. 5998–6008, 2017.

Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-UCSD birds-200-2011 dataset. Technical report, California Institute of Technology, 2011.

Liyuan Wang, Jingyi Xie, Xingxing Zhang, Mingyi Huang, Hang Su, and Jun Zhu. Hierarchical decomposition of prompt-based continual learning: Rethinking obscured sub-optimality. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Shipeng Wang, Xiaorong Li, Jian Sun, and Zongben Xu. Training networks in null space of feature covariance for continual learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 184–193, 2021. doi: 10.1109/CVPR46437.2021.00025.

Shipeng Wang, Xiaorong Li, Jian Sun, and Zongben Xu. Training networks in null space of feature covariance with self-supervision for incremental learning. *IEEE Trans. Pattern Anal. Mach. Intell.*, 47(4):2563–2580, 2025. doi: 10.1109/TPAMI.2024.3522258.

Zifeng Wang, Zizhao Zhang, Sayna Ebrahimi, Ruoxi Sun, Han Zhang, Chen-Yu Lee, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer G. Dy, and Tomas Pfister. Dualprompt: Complementary prompting for rehearsal-free continual learning. In *European Conference on Computer Vision*, pp. 631–648, 2022a. doi: 10.1007/978-3-031-19809-0_36.

Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer G. Dy, and Tomas Pfister. Learning to prompt for continual learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 139–149, 2022b. doi: 10.1109/CVPR52688.2022.00024.

Ross Wightman. Pytorch image models, 2019. Software library.

Shipeng Yan, Jiangwei Xie, and Xuming He. Der: Dynamically expandable representation for class incremental learning. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3014–3023, 2021.

Jaehong Yoon, Eunho Yang, Jeongtae Lee, and Sung Ju Hwang. Lifelong learning with dynamically expandable networks. In *International Conference on Learning Representations*, 2018.

Guanxiong Zeng, Yang Chen, Bo Cui, and Shan Yu. Continual learning of context-dependent processing in neural networks. *Nature Machine Intelligence*, 1:364–372, 2019. doi: 10.1038/S42256-019-0080-X.

Gengwei Zhang, Liyuan Wang, Guoliang Kang, Ling Chen, and Yunchao Wei. SLCA: Slow learner with classifier alignment for continual learning on a pre-trained model. In *IEEE/CVF International Conference on Computer Vision*, pp. 19091–19101, 2023.

Gengwei Zhang, Liyuan Wang, Guoliang Kang, Ling Chen, and Yunchao Wei. SLCA++: Unleash the power of sequential fine-tuning for continual learning with pre-training. *arXiv preprint arXiv:2408.08295*, 2024.

Shizhou Zhang, Yue Lu, De Cheng, Yinghui Xing, Nannan Wang, Peng Wang, and Yanning Zhang. VPT-NSP^2++: Importance-aware visual prompt tuning in null space for continual learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 48(4):4318–4335, 2026. doi: 10.1109/TPAMI.2025.3642298.

Bowen Zhao, Xi Xiao, Guojun Gan, Bin Zhang, and Shutao Xia. Maintaining discrimination and fairness in class incremental learning. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 13205–13214, 2020.

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 *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 23554–23564, 2024. doi: 10.1109/CVPR52733.2024.02223.

Heming Zou, Yunliang Zang, Wutong Xu, and Xiangyang Ji. FLY-CL: A fly-inspired framework for enhancing efficient decorrelation and reduced training time in pre-trained model-based continual representation learning. In *International Conference on Learning Representations*, 2026.

A Theoretical Derivations

A.1 Stability Condition Derivations for Self-Attention Module

For the self-attention module, the output change ΔZ induced by ΔW_v can be expressed as:

$$\Delta Z = S_\tau X_\tau (W_v + \Delta W_v) - S_\tau X_\tau W_v = S_\tau X_\tau \Delta W_v. \tag{21}$$

Similarly, the variation of the attention score matrix ΔA caused by ΔW_q and ΔW_k is given by:

$$\Delta A = X_\tau (W_q + \Delta W_q)(W_k + \Delta W_k)^\top X_\tau^\top - X_\tau W_q W_k^\top X_\tau^\top \tag{22}$$

$$= X_\tau W_q \Delta W_k^\top X_\tau^\top + X_\tau \Delta W_q W_k^\top X_\tau^\top + X_\tau \Delta W_q \Delta W_k^\top X_\tau^\top. \tag{23}$$

Under the small-update assumption, neglecting the second-order term yields the following first-order approximation:

$$\Delta A \approx X_\tau W_q \Delta W_k^\top X_\tau^\top + X_\tau \Delta W_q W_k^\top X_\tau^\top. \tag{24}$$

We empirically verify the validity of this approximation in Appendix C.

A.2 Proof of Theorem 1

Before proving Theorem 1, we first present the following lemma, which characterizes the solution spaces of the linear equations $X\Delta W = 0$ and $\Delta W X^\top = 0$.

Lemma 1. *Let $X \in \mathbb{R}^{n\times d}$ $(n > d)$ be a real matrix of rank k. Its singular value decomposition is*

$$X = \begin{bmatrix} U_1 & U_0 \end{bmatrix} \begin{bmatrix} \Lambda & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} V_1 & V_0 \end{bmatrix}^{\mathrm{T}}, \tag{25}$$

where $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_k)$, $U_1 \in \mathbb{R}^{n\times k}$ and $V_1 \in \mathbb{R}^{d\times k}$ contain the singular vectors associated with nonzero singular values, while $U_0 \in \mathbb{R}^{n\times(n-k)}$ and $V_0 \in \mathbb{R}^{d\times(d-k)}$ contain the remaining singular vectors. Let $r = d - k$ denote the right-null-space dimension.

Then the solution space for the equation $X\Delta W = 0$ (where $\Delta W \in \mathbb{R}^{d\times d'}$) can be represented as

$$\{V_0 C | C \in \mathbb{R}^{r\times d'}\}. \tag{26}$$

Likewise, for $\Delta W X^\top = 0$ with $X \in \mathbb{R}^{n\times d'}$, let $V_0 \in \mathbb{R}^{d'\times r}$ be a basis of the right null space of this X, where $r = d' - \mathrm{rank}(X)$. For $\Delta W \in \mathbb{R}^{d\times d'}$, the solution space is

$$\{C V_0^\top | C \in \mathbb{R}^{d\times r}\}. \tag{27}$$

Now we present the proof of Theorem 1.

Proof. Let

$$P_L = V_1^{(L)} (V_1^{(L)})^\top, \qquad P_R = V_1^{(R)} (V_1^{(R)})^\top \tag{28}$$

denote the orthogonal projectors onto the row spaces of L and R, respectively. Their orthogonal complements are

$$P_{L,0} = I - P_L = V_0^{(L)} (V_0^{(L)})^\top, \qquad P_{R,0} = I - P_R = V_0^{(R)} (V_0^{(R)})^\top. \tag{29}$$

Since $P_L = L^\dagger L$ and $P_R = R^\dagger R$ where † denotes the Moore–Penrose pseudoinverse, we have

$$L\Delta W R^\top = 0 \quad \Longleftrightarrow \quad P_L \Delta W P_R = 0. \tag{30}$$

For any $\Delta W \in \mathbb{R}^{d\times d'}$, its decomposition with respect to these two orthogonal subspaces is

$$\Delta W = (P_{L,0} + P_L)\Delta W = P_{L,0}\Delta W + P_L \Delta W P_{R,0} + P_L \Delta W P_R. \tag{31}$$

If $L\Delta W R^\top = 0$, then $P_L \Delta W P_R = 0$, and consequently

$$\Delta W = P_{L,0}\Delta W + P_L \Delta W P_{R,0} \tag{32}$$

$$= V_0^{(L)} (V_0^{(L)})^\top \Delta W + P_L \Delta W V_0^{(R)} (V_0^{(R)})^\top \tag{33}$$

$$= V_0^{(L)} C_1 + C_2 (V_0^{(R)})^\top, \tag{34}$$

where

$$C_1 = (V_0^{(L)})^\top \Delta W, \qquad C_2 = P_L \Delta W V_0^{(R)}. \tag{35}$$

Thus, every solution of $L\Delta W R^\top = 0$ belongs to $\mathcal{N}_1 + \mathcal{N}_2$.

Conversely, for any

$$\Delta W = V_0^{(L)} C_1 + C_2 (V_0^{(R)})^\top, \tag{36}$$

we have

$$L\Delta W R^\top = L V_0^{(L)} C_1 R^\top + L C_2 (V_0^{(R)})^\top R^\top = 0, \tag{37}$$

since $L V_0^{(L)} = 0$ and $R V_0^{(R)} = 0$. Therefore,

$$\ker(\Delta W \mapsto L\Delta W R^\top) = \mathcal{N}_1 + \mathcal{N}_2. \tag{38}$$

□

A.3 Proof of Theorem 2

Proof. Consider the linear map $\Phi(\Delta W) = L\Delta W R^\top$ on $\mathbb{R}^{d\times d'}$. By vectorization, $\mathrm{vec}(\Phi(\Delta W)) = (R\otimes L)\,\mathrm{vec}(\Delta W)$, where $\otimes$ denotes the Kronecker product. Hence $\mathrm{rank}(\Phi) = \mathrm{rank}(R)\,\mathrm{rank}(L) = k_1 k_2$. The rank–nullity theorem gives

$$\dim(\mathcal{N}_+) = dd' - k_1 k_2 = r_1 d' + d r_2 - r_1 r_2. \tag{39}$$

Since $\dim(\mathcal{N}_1) = r_1 d'$ and $\dim(\mathcal{N}_2) = d r_2$, we obtain

$$\dim(\mathcal{N}_+) - \dim(\mathcal{N}_1) = k_1 r_2, \qquad \dim(\mathcal{N}_+) - \dim(\mathcal{N}_2) = r_1 k_2. \tag{40}$$

Both $\mathcal{N}_1$ and $\mathcal{N}_2$ are subspaces of $\mathcal{N}_+$ by Theorem 1. Each inclusion is therefore strict if and only if its corresponding dimension gain is positive. □

Applied to the query and key constraints in Eq. (12), the respective dimension gains over the input-null-space constraint are $\mathrm{rank}(X_\tau) r_k$ and $\mathrm{rank}(X_\tau) r_q$.

A.4 LANS with Coupled Query-Key Characterization

In the preceding discussion, we decouple the affinity-preservation condition for query and key into two independent constraints, namely $X_\tau \Delta W_q K_\tau^\top = 0$ and $X_\tau \Delta W_k Q_\tau^\top = 0$, and construct their parameterizations in Eqs. (18) and (19). Although the separable constraints admit efficient closed-form parameterizations, requiring the two affinity variations to vanish independently is stronger than necessary. We next characterize the original coupled condition in Eq. (10a).

Theorem 3 (Coupled Query–Key Characterization)**.** *Let $X_\tau \in \mathbb{R}^{n\times d}$, $W_q, W_k \in \mathbb{R}^{d\times d'}$, and define $Q_\tau = X_\tau W_q$ and $K_\tau = X_\tau W_k$. Let $V_0^{(X_\tau)}$, $V_0^{(Q_\tau)}$, and $V_0^{(K_\tau)}$ be bases of their respective right null spaces, with dimensions r_x, r_q, and r_k. A pair $(\Delta W_q, \Delta W_k)$ satisfies the coupled first-order condition*

$$X_\tau \Delta W_q K_\tau^\top + Q_\tau \Delta W_k^\top X_\tau^\top = 0 \tag{41}$$

if and only if it admits the parameterization

$$\Delta W_q = V_0^{(X_\tau)} C_{q,1} + C_{q,2} \big(V_0^{(K_\tau)}\big)^\top - W_q M, \tag{42}$$

$$\Delta W_k = V_0^{(X_\tau)} C_{k,1} + C_{k,2} \big(V_0^{(Q_\tau)}\big)^\top + W_k M^\top, \tag{43}$$

where $C_{q,1}, C_{k,1} \in \mathbb{R}^{r_x\times d'}$, $C_{q,2} \in \mathbb{R}^{d\times r_k}$, $C_{k,2} \in \mathbb{R}^{d\times r_q}$, and $M \in \mathbb{R}^{d'\times d'}$ are arbitrary coefficient matrices.

Proof. For brevity, we omit the historical-task index τ throughout the proof.

Sufficiency. Suppose the updates have the stated form. Using $X V_0^{(X)} = 0$, $Q V_0^{(Q)} = 0$, and $K V_0^{(K)} = 0$, we obtain

$$X \Delta W_q K^\top = -QMK^\top, \tag{44}$$

$$Q \Delta W_k^\top X^\top = QMK^\top. \tag{45}$$

Their sum is zero, so the coupled condition holds.

Necessity. Conversely, suppose that $X\Delta W_q K^\top + Q\Delta W_k^\top X^\top = 0$. Define

$$Y = X\Delta W_q K^\top = -Q\Delta W_k^\top X^\top. \tag{46}$$

The first representation implies $\mathrm{col}(Y^\top) \subseteq \mathrm{col}(K)$, whereas the second implies $\mathrm{col}(Y) \subseteq \mathrm{col}(Q)$. Consequently,

$$QQ^\dagger Y = Y, \qquad YKK^\dagger = Y, \tag{47}$$

where † denotes the Moore–Penrose pseudoinverse. Set

$$M = -Q^\dagger Y (K^\dagger)^\top. \tag{48}$$

Since $QQ^\dagger$ and $KK^\dagger$ are orthogonal projectors,

$$QMK^\top = -QQ^\dagger Y (KK^\dagger)^\top = -Y. \tag{49}$$

Now define

$$E_q = \Delta W_q + W_q M, \qquad E_k = \Delta W_k - W_k M^\top. \tag{50}$$

Using $Q = XW_q$ and $K = XW_k$, we have

$$XE_q K^\top = Y + QMK^\top = 0, \tag{51}$$

$$QE_k^\top X^\top = -Y - QMK^\top = 0. \tag{52}$$

Transposing the second equality gives $XE_kQ^\top = 0$. By Theorem 1, there exist coefficient matrices of compatible dimensions such that

$$E_q = V_0^{(X)} C_{q,1} + C_{q,2}\big(V_0^{(K)}\big)^\top, \tag{53}$$

$$E_k = V_0^{(X)} C_{k,1} + C_{k,2}\big(V_0^{(Q)}\big)^\top. \tag{54}$$

Substituting the definitions of E_q and E_k yields Eq. (42) and Eq. (43), completing the proof. □

Discussion In Eq. (42) and Eq. (43), the terms involving M induce equal and opposite first-order affinity variations, allowing nonzero query and key contributions to cancel. This permits additional coupled update directions at the cost of d'^2 extra trainable parameters per head. Considering the resulting increase in training overhead, we do not adopt this coupled scheme in our implementation.

A.5 Extension to Multi-Head Attention

We now extend our proposed LANS to multi-head attention (MHA). Suppose there are H heads and the dimension of each head is $d_h = d/H$. Let $W^h_{\{q,k,v\}} \in \mathbb{R}^{d\times d_h}$ denote the query, key, and value projection matrices for the h-th head. And let $Q^h_\tau = X_\tau W^h_q$, $K^h_\tau = X_\tau W^h_k$, $V^h_\tau = X_\tau W^h_v$ denote the projected query, key and value for the h-th head.

For W^h_q, its update ΔW^h_q should satisfy $X_\tau \Delta W^h_q (K^h_\tau)^\top = 0$, yielding the parameterization

$$\Delta W^h_q = V_0^{(X_\tau)} C^h_{q,1} + C^h_{q,2} (V_0^{(K^h_\tau)})^T. \tag{55}$$

Then the update ΔW_q is formulated as

$$\begin{aligned}
\Delta W_q &= \left[\Delta W_q^1, \ldots, \Delta W_q^H\right] \\
&= \left[V_0^{(X_\tau)} C^1_{q,1} + C^1_{q,2}(V_0^{(K^1_\tau)})^\top, \ldots, V_0^{(X_\tau)} C^H_{q,1} + C^H_{q,2}(V_0^{(K^H_\tau)})^\top\right] \\
&= V_0^{(X_\tau)} C_{q,1} + \left[C^1_{q,2}(V_0^{(K^1_\tau)})^T, \ldots, C^H_{q,2}(V_0^{(K^H_\tau)})^T\right],
\end{aligned} \tag{56}$$

where $C_{q,1} = \left[C^1_{q,1}, \ldots, C^H_{q,1}\right] \in \mathbb{R}^{r_x \times d}$ and $C^h_{q,2} \in \mathbb{R}^{d \times r_{k,h}}$ are trainable matrices, while $V_0^{(X_\tau)} \in \mathbb{R}^{d\times r_x}$ and $V_0^{(K^h_\tau)} \in \mathbb{R}^{d_h \times r_{k,h}}$ are fixed during training.

Similarly, for the key projection, we have

$$\Delta W_k = V_0^{(X_\tau)} C_{k,1} + \left[C^1_{k,2}(V_0^{(Q^1_\tau)})^T, \ldots, C^H_{k,2}(V_0^{(Q^H_\tau)})^T\right], \tag{57}$$

Algorithm 1 LANS

Require: Pretrained model f, task datasets $\{\mathcal{D}_t\}_{t=1}^{T}$
Output: Updated model f' with merged updates
Initialize empty historical covariance statistics for X_τ, Q_τ, K_τ, and $S_\tau X_\tau$
for $t = 1, \ldots, T$ **do**
    if $t = 1$ **then**
        Train LoRA updates $\Delta W = AB$ for each linear layer
    else
        Construct residual bases from historical covariances with threshold ϵ (Appendix B.2)
        Parameterize $\Delta W_q, \Delta W_k, \Delta W_v$ with (18), (19), and (20)
        Train coefficient matrices end-to-end while keeping the bases frozen
    end if
    Merge the learned updates ΔW into the pre-trained parameters
    Compute and update covariances with the current dataset $\mathcal{D}_t$ (Appendix B.1)
end for

For the value projection of the h-th head, the head-specific value update can be parameterized as

$$\Delta W_v^h = V_0^{(S_\tau^h X_\tau)} C_v^h, \tag{58}$$

where $V_0^{(S_\tau^h X_\tau)} \in \mathbb{R}^{d \times r_{v,h}}$ and $C_v^h \in \mathbb{R}^{r_{v,h} \times d_h}$. Concatenating the updates of all heads gives

$$\Delta W_v = \left[V_0^{(S_\tau^{(1)} X_\tau)} C_v^{(1)}, \ldots, V_0^{(S_\tau^{(H)} X_\tau)} C_v^{(H)} \right]. \tag{59}$$

B Algorithmic Details

Algorithm 1 summarizes the training procedure of LANS.

B.1 Computation and Update of Null-Space Basis via Covariances

Our method requires the null-space bases for multiple components, i.e., $V_0^{(X_\tau)}$, $V_0^{(Q_\tau)}$, $V_0^{(K_\tau)}$, and $V_0^{(S_\tau X_\tau)}$ $(\tau \leq t)$. A practical concern is the storage and maintenance of the input features required for computing these bases. Fortunately, they can be efficiently derived from the uncentered covariance matrix (UCM) of the input feature matrix X_τ, which serves as a central bridge.

Computation of null-space basis via uncentered covariance. After completing task $\mathcal{T}_t$, we obtain input features $\{X_{t,j} \in \mathbb{R}^{n \times d}\}_{j=1}^{n_t}$, where n_t is the number of samples and each $X_{t,j}$ is a sequence of length n with feature dimension d. The UCM is defined as

$$\mathcal{C}_t(X_t) = \frac{1}{n_t} \sum_{j=1}^{n_t} X_{t,j}^\top X_{t,j} \in \mathbb{R}^{d \times d}. \tag{60}$$

The null-space basis $V_0^{(X_t)}$ can be obtained directly from the eigenvectors of $\mathcal{C}_t(X_t)$ corresponding to zero (or negligibly small) eigenvalues, since it shares the same null space as X_t.

For $V_0^{(Q_t)}$ and $V_0^{(K_t)}$, we do not need to store Q_t and K_t themselves or maintain separate covariance matrices for them. Both can be derived on the fly from the stored $\mathcal{C}_t(X_t)$ and the current projection matrices W_q and W_k:

$$\mathcal{C}_t(Q_t) = \frac{1}{n_t} \sum_{j=1}^{n_t} W_q^\top X_{t,j}^\top X_{t,j} W_q = W_q^\top \mathcal{C}_t(X_t) W_q, \tag{61}$$

$$\mathcal{C}_t(K_t) = \frac{1}{n_t} \sum_{j=1}^{n_t} W_k^\top X_{t,j}^\top X_{t,j} W_k = W_k^\top \mathcal{C}_t(X_t) W_k. \tag{62}$$

Then the null-space bases $V_0^{(Q_t)}$ and $V_0^{(K_t)}$ are given by the eigenvectors of $\mathcal{C}_t(Q_t)$ and $\mathcal{C}_t(K_t)$ corresponding to zero (or negligibly small) eigenvalues, respectively.

For the value projection, the weighted input $S_t X_t$ does not admit a simple covariance transformation through a fixed projection matrix because the attention matrix S_t is sample-dependent. We therefore explicitly compute its uncentered covariance:

$$\mathcal{C}_t(S_t X_t) = \frac{1}{n_t} \sum_{j=1}^{n_t} (S_{t,j} X_{t,j})^\top (S_{t,j} X_{t,j}). \tag{63}$$

Incremental update of covariances across tasks. When a new task $\mathcal{T}_{t+1}$ arrives, we compute its per-task covariance $\mathcal{C}_{t+1}(\cdot)$ using the same procedure. To maintain a unified covariance summarizing all tasks seen so far, we update it as

$$\mathcal{C}_{1:t+1}(\cdot) = \frac{n_{t+1}}{N_{t+1}} \mathcal{C}_{t+1}(\cdot) + \frac{N_t}{N_{t+1}} \mathcal{C}_{1:t}(\cdot), \tag{64}$$

where $N_t = \sum_{s=1}^{t} n_s$ denotes the total number of samples observed through task t, and $\mathcal{C}_{1:t}(\cdot)$ is the accumulated covariance. This online update provides a consistent summary of historical data without storing individual samples or per-task covariance matrices, so the memory footprint remains constant with respect to the number of tasks.

The accumulated covariance also enforces all task-wise constraints used in the main text. In particular,

$$\mathcal{C}_{1:t}(X) = \frac{1}{N_t} \sum_{\tau=1}^{t} n_\tau \mathcal{C}_\tau(X_\tau), \qquad v^\top \mathcal{C}_{1:t}(X) v = \frac{1}{N_t} \sum_{\tau=1}^{t} \sum_{j=1}^{n_\tau} \|X_{\tau,j} v\|_2^2. \tag{65}$$

Because every term on the right is non-negative, v is a zero-eigenvalue direction of $\mathcal{C}_{1:t}(X)$ if and only if $X_{\tau,j} v = 0$ for every sample of every historical task $\tau \le t$. Thus, extracting a basis from the accumulated covariance implements the task-wise null-space conditions jointly, without explicitly storing or concatenating historical features. The same argument applies to the accumulated covariances of Q, K, and SX.

Approximate null space with threshold. The preceding theoretical results characterize exact null spaces. In practice, accumulated feature covariance matrices can be full rank, and feature noise and floating-point errors make exactly zero singular values unreliable. Approximating null spaces using singular vectors associated with small singular values is a common practice in null-space-based continual learning (Wang et al., 2021; 2025; Lu et al., 2024). Following this practice, we estimate a near-null subspace using a cumulative spectral-energy threshold ϵ. Specifically, we compute $\mathcal{C}_{1:t}(\cdot) = USV^\top$, sort the singular values in descending order, and retain the minimal set of leading components whose cumulative squared energy accounts for at least an ϵ fraction of the total spectral energy:

$$\frac{\sum_{i=1}^{k} \sigma_i^2}{\sum_{i=1}^{d} \sigma_i^2} \ge \epsilon, \tag{66}$$

where k is the number of retained main components. The remaining singular vectors corresponding to the trailing singular values are retained as the residual-space basis. Thus, the practical parameterization restricts updates to directions with low historical feature responses rather than strictly zero responses. The exact stability guarantees apply to exact null-space bases; with these residual bases, the constraints are enforced approximately, and the resulting stability is assessed empirically in Fig. 4a.

B.2 Truncation and Zero-Padding of Null-Space Bases

As the covariance statistics are updated after each task, the residual dimension may change and tends to decrease as historical features occupy more directions in the input space. To accommodate these varying dimensions while enabling efficient batched computation across attention heads, we use fixed-shape LoRA factors obtained by truncating or zero-padding the residual bases. Specifically, let $V_0^{(X)} \in \mathbb{R}^{d \times r_x}$ denote the residual basis obtained from the covariance matrix of a historical

feature X, where r_x is the number of available residual directions. Given a prescribed LoRA rank r, we select

$$\bar{V}_0^{(X)} = \begin{cases} V_0^{(X)}[:, r_x - r : r_x], & r_x \geq r, \\ \left[V_0^{(X)}, \mathbf{0}_{d\times(r-r_x)}\right], & r_x < r, \end{cases} \tag{67}$$

where the columns of $V_0^{(X)}$ are ordered according to descending covariance eigenvalues. Therefore, when more than r residual directions are available, we retain the r directions associated with the smallest eigenvalues, as they induce the least response on historical features. When fewer than r directions are available, the remaining columns are zero-padded instead of being filled with unconstrained directions. This preserves the fixed parameter shape without introducing updates outside the selected residual subspace.

The same truncation and zero-padding procedure is applied independently to each attention head, using a shared rank budget $r_h = r/H$, where r is divisible by H. Although the available residual dimensions may differ across heads, the resulting factors have identical shapes and can therefore be stacked for GPU-accelerated batched matrix multiplication. For example, for the value projection, the padded bases and trainable coefficients in Eq. (59) are stacked into tensors of shapes $H \times d \times r_h$ and $H \times r_h \times d_h$, respectively. Their products can be computed simultaneously using `torch.bmm`, yielding a tensor of shape $H \times d \times d_h$. Permuting and reshaping this tensor produces the concatenated update $\Delta W_v \in \mathbb{R}^{d\times d}$ without explicitly constructing a block matrix.

C Additional Results

C.1 Experiment Details

Compared Methods. We compare LANS with representative rehearsal-free methods for pre-trained model-based continual learning. The expandable baselines include prompt-based methods L2P (Wang et al., 2022b), DualPrompt (Wang et al., 2022a), CODA-Prompt (Smith et al., 2023), and HiDe-Prompt (Wang et al., 2023), as well as adapter- or expert-based methods EASE (Zhou et al., 2024) and MoTE (Li et al., 2025). We also include non-expanding baselines that retain a shared backbone across tasks: FLY-CL (Zou et al., 2026), SLCA (Zhang et al., 2023), SLCA++ (Zhang et al., 2024), InfLoRA (Liang & Li, 2024), and VPT-NSP^2 (Lu et al., 2024; Zhang et al., 2026). InfLoRA and VPT-NSP^2 are the most closely related null-space baselines: the former constrains low-rank weight updates, whereas the latter applies null-space projection to visual prompts. Joint training uses data from all tasks simultaneously and serves as an offline reference.

Evaluation Metrics. Let $a_{t,j}$ denote the classification accuracy on task $\mathcal{T}_j$ after the model has completed training on task $\mathcal{T}_t$, where $j \leq t$. We report Final Average Accuracy (FAA), which measures performance over all tasks after the complete task sequence,

$$\mathrm{FAA} = \frac{1}{T}\sum_{j=1}^{T} a_{T,j}, \tag{68}$$

and Continual Average Accuracy (CAA), which averages the model's performance over all evaluation stages,

$$\mathrm{CAA} = \frac{1}{T}\sum_{t=1}^{T}\left(\frac{1}{t}\sum_{j=1}^{t} a_{t,j}\right). \tag{69}$$

FAA emphasizes the final retained performance, whereas CAA summarizes accuracy throughout the entire learning trajectory. Higher values are better for both metrics.

Implementation Details. We implement LANS in PyTorch (Paszke et al., 2019) using the timm library (Wightman, 2019). All input images are resized to 224×224, and models are trained with a batch size of 128. We train for 50 epochs per task, except for CIFAR-100 with ViT-B/16-IN21K (40 epochs) and CUB-200 with DeiT-S/16-IN611 (90 epochs). We use SGD with an initial learning rate of 0.01, weight decay of 5×10^{-4}, a momentum of 0.9, and cosine learning-rate annealing. The residual-subspace threshold ϵ is set to 0.95. Following the classifier-alignment protocol of

SLCA++ (Zhang et al., 2024), the classifier is further calibrated using Gaussian features sampled from the estimated class statistics for 20 epochs, or 40 epochs for ImageNet-R with ViT-B/16-IN21K. All experiments are performed on four NVIDIA RTX 4090 GPUs.

Classifier alignment (CA) is a widely adopted technique in pre-trained model-based continual learning and is also used by the competitive baselines HiDe-Prompt (Wang et al., 2023), SLCA (Zhang et al., 2023), SLCA++ (Zhang et al., 2024), and VPT-NSP^2 (Lu et al., 2024) in our comparisons. For the component ablations (Table 4), all variants, including ordinary LoRA, share the same calibration protocol and budget within each experimental setting. These comparisons therefore quantify the incremental benefits of the LANS modules with calibration held fixed. We likewise apply a common calibration protocol and budget across the compared methods in the update-alignment and stability–plasticity analyses (Figs. 3 and 4).

C.2 On the Validity of Omitting the Second-Order Term in Attention

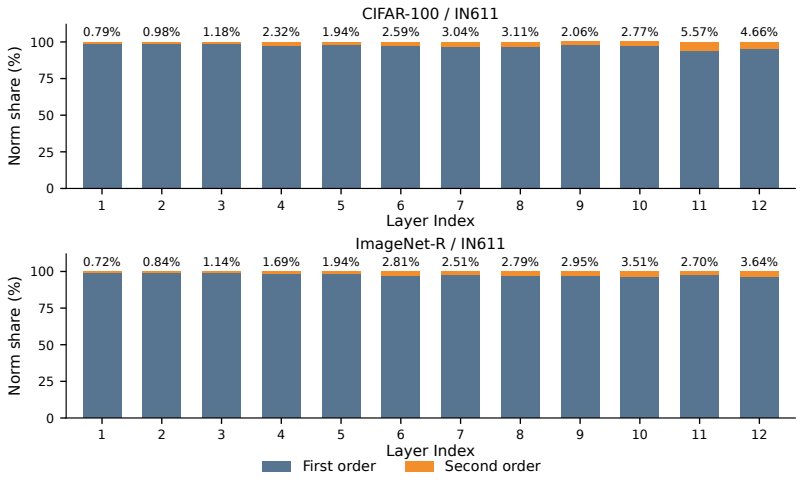


Figure 5: First- and second-order norm shares for the task-two LoRA update on DeiT-S/16-IN611.

We examine the magnitude of the second-order term omitted in the attention stability approximation. We train standard LoRA sequentially on two tasks, obtaining query and key projections $W_{q,1}, W_{k,1}$ and $W_{q,2}, W_{k,2}$. Let $\Delta W_q = W_{q,2} - W_{q,1}$ and $\Delta W_k = W_{k,2} - W_{k,1}$. For historical inputs X_1 extracted using the task-one model, the first- and second-order changes in the attention logits are

$$\Delta A_{1\mathrm{st}} = X_1 \Delta W_q W_{k,1}^\top X_1^\top + X_1 W_{q,1} \Delta W_k^\top X_1^\top, \tag{70}$$

$$\Delta A_{2\mathrm{nd}} = X_1 \Delta W_q \Delta W_k^\top X_1^\top. \tag{71}$$

Fig. 5 compares the two terms on CIFAR-100 and ImageNet-R with DeiT-S/16-IN611. The first-order term dominates across all layers, with the second-order share remaining below 5.6% on CIFAR-100 and 3.7% on ImageNet-R.

Effect on Attention Probabilities. Unlike the standard-LoRA analysis above, this experiment evaluates LANS itself by comparing attention probabilities computed with and without the second-order term. Here, ΔW_q and ΔW_k denote the full task-two weight increments learned by LANS, measured between task-boundary checkpoints rather than individual optimization steps. We compute $\Delta A_{1\mathrm{st}}$ and $\Delta A_{2\mathrm{nd}}$ using these increments and the same fixed historical inputs X_1. With A_1 denoting the corresponding unscaled task-one affinity matrix and d_h the head dimension, let $\hat{S} = \mathrm{softmax}((A_1 + \Delta A_{1\mathrm{st}})/\sqrt{d_h})$ and $S = \mathrm{softmax}((A_1 + \Delta A_{1\mathrm{st}} + \Delta A_{2\mathrm{nd}})/\sqrt{d_h})$. Their relative error is

$$e_{\mathrm{attn}} = \left(\frac{\sum_{\ell,h,x} \|\hat{S}_{\ell,h}(x) - S_{\ell,h}(x)\|_F^2}{\sum_{\ell,h,x} \|S_{\ell,h}(x)\|_F^2} \right)^{1/2}, \tag{72}$$

where the sums run over evaluated layers, heads, and historical images. For LANS, this error is 0.416% on CIFAR-100 and 0.259% on ImageNet-R, indicating that omitting the second-order term

has a small effect on attention probabilities in the evaluated setting. Unlike the relative magnitudes of the logit terms, this metric directly measures the approximation error in attention probabilities even when the first-order variation is suppressed. It does not, however, measure cumulative representation or prediction drift over a long task sequence.