STABILITY–PLASTICITY BALANCE VIA SINGULAR-VECTOR SELECTION IN LLM CONTINUAL LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Domain-specific continual adaptation of LLMs risks catastrophic forgetting, creating a fundamental tension between acquiring new capabilities and preserving those learned during pretraining. PEFT mitigates this problem by restricting the number of trainable parameters, but existing methods lack a principled unit for deciding where plasticity should be allocated and stability should be preserved. We identify the singular-vector channel as a natural unit for managing this trade-off. Each channel represents an input–output transformation, which can be updated to acquire new knowledge or fixed to preserve pretrained capabilities. Based on this perspective, we introduce SVC, a parameter-efficient continual-learning method that selectively updates Singular-Vector Channels. Before fine-tuning, SVC uses domain-specific data to estimate each channel's adaptation benefit and a fixed public general-domain corpus only as a history activation proxy for estimating forgetting cost. It then adaptively selects trainable channels based on these scores via knee-based cost screening, Pareto-front filtering, and Otsu thresholding. Experimental results across four LLM families and eight downstream tasks show that SVC better preserves pretrained capabilities while achieving strong downstream performance relative to existing PEFT baselines. Further analysis of channel scoring and selection demonstrates that selective plasticity at the singular-vector-channel level enables effective continual LLM adaptation.

1 INTRODUCTION

Large language models (LLMs) gain broad knowledge and general-purpose reasoning through large-scale pretraining (Karanikolas et al., 2023), but often require domain-specific fine-tuning to meet specialized data and task demands (Chen et al., 2025). However, such adaptation can cause catastrophic forgetting, degrading pretrained capabilities and reducing reliability in real-world settings (French, 1999; Liu et al., 2020). Therefore, continual domain adaptation requires a careful balance between *plasticity*, to acquire new domain knowledge, and *stability*, to preserve pretrained capabilities.

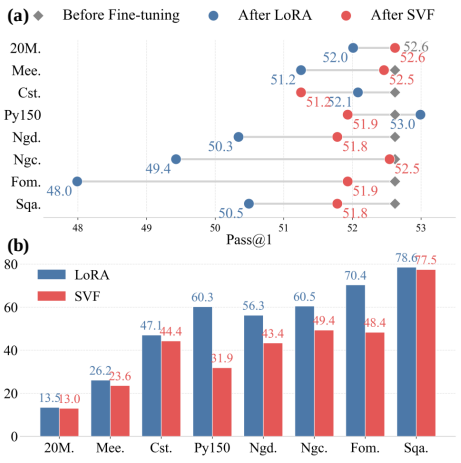

Figure 1: **Stability–plasticity trade-off between LoRA and SVF.** (a) GSM8K retention across fine-tuning tasks. (b) Downstream performance. Details in Appendix C

Traditional continual learning methods (Shi et al., 2025) mitigate forgetting primarily by preserving information from previous learning stages. Regularization-based approaches constrain parameter updates or functional changes to protect previously acquired knowledge (Kirkpatrick et al., 2017; Li & Hoiem, 2017), while replay-based approaches revisit stored or generated samples from earlier tasks (Rebuffi et al., 2017; Rolnick et al., 2019). Applying these strategies to LLMs is challenging because the original pretraining data and training signals are typically unavailable, making it difficult to directly preserve the broad capabilities acquired during pretraining. Moreover, overly strong constraints can impede adaptation to new domains, whereas weak constraints may fail to prevent forgetting. As a

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.

result, achieving a favorable stability-plasticity trade-off remains challenging for continual domain adaptation of LLMs.

Parameter-efficient fine-tuning (PEFT) offers a promising alternative by freezing most pretrained parameters and adapting only a small subset (Ding et al., 2023; Li et al., 2025). A prominent example is low-rank adaptation (LoRA) (Hu et al., 2022), which learns low-rank weight updates to enable effective downstream adaptation while often reducing degradation of pretrained capabilities (Biderman et al., 2024). However, the extent of such preservation can vary substantially across adaptation tasks. As shown in Figure 1(a), when the same pretrained model is independently adapted to each of the eight TRACE tasks (Wang et al., 2023b) using LoRA, the resulting models exhibit markedly different drops in GSM8K performance (Cobbe et al., 2021). This suggests that restricting the update rank alone does not provide consistent control over forgetting.

Singular value fine-tuning (SVF) (Sun et al., 2022) imposes a stronger structural constraint. Given a weight matrix decomposed by singular value decomposition, SVF updates only the singular values while keeping the corresponding left and right singular vectors fixed. Since the singular vectors determine the input and output directions of each rank-one component, whereas the singular values control their strengths (Muller et al., 2004), SVF modifies the magnitude of existing transformations without changing their directions. As shown in Figure 1(a), this stronger directional constraint yields substantially more consistent retention of GSM8K performance across tasks. However, Figure 1(b) shows that it also limits downstream adaptation performance.

These observations reveal a central tension in parameter-efficient adaptation: allowing transformation directions to change improves plasticity but can increase interference with pretrained capabilities, while fixing all directions improves stability at the cost of adaptation. This suggests that the key question is not whether transformation directions should be updated, but *which directions should be updated for a given task*. To enable such task-dependent control, we treat each paired left and right singular-vector direction associated with a singular value as a *singular-vector channel*, which serves as the basic unit for balancing adaptation benefit against forgetting cost.

In this paper, we introduce **SVC**, a parameter-efficient continual adaptation method that selectively updates Singular-Vector Channels. Before fine-tuning, SVC evaluates each channel using two task-dependent signals: its potential benefit for learning the current domain and its potential cost to previously acquired or general-domain capabilities. SVC then identifies candidate channels through knee-based cost screening and Pareto-front filtering in the benefit–cost plane, and applies Otsu thresholding to automatically select a task-specific subset for optimization. During adaptation, all singular values and the singular vectors of the selected channels are updated jointly, while the remaining singular-vector directions are kept fixed. In this way, SVC retains the structural stability of singular value fine-tuning while introducing directional flexibility only where it is beneficial for the current task.

Extensive experiments across four LLM families and eight downstream tasks show that SVC effectively adapts to sequential tasks while consistently preserving pretrained general capabilities. Further analyses validate the effectiveness of channel-level benefit-cost estimation and the multi-stage selection mechanism, showing that selective channel updates better balance model stability and plasticity. Our contributions are summarized as follows:

- We identify singular-vector channels as a principled unit for controlling the stability-plasticity trade-off in parameter-efficient continual adaptation.
- We propose a parameter-efficient continual-learning method for LLMs that adaptively balances stability and plasticity across tasks.
- SVC requires neither prior-task samples nor task-specific modules, yet consistently performs well across models and task orders, demonstrating strong generality and scalability.

2 Related Work

Parameter-Efficient Fine-Tuning (PEFT) adapts LLMs through a small number of trainable parameters while keeping most pretrained weights fixed (Ding et al., 2023). Recent studies have shown that this restricted update regime can reduce catastrophic forgetting (Vu et al., 2022; Wang et al., 2025b), with LoRA exhibiting particularly favorable retention behavior (Biderman et al., 2024).

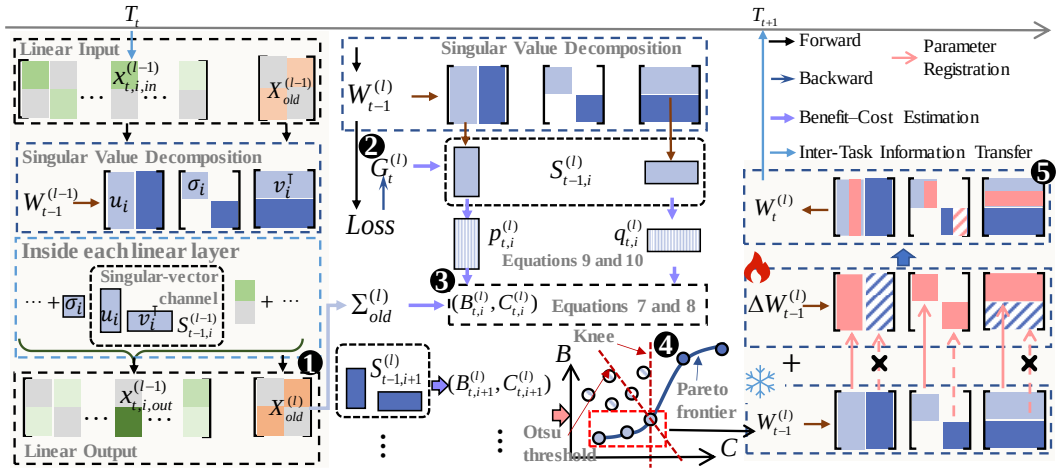

Figure 2: **Overview of SVC.** SVC ① collects reference activation statistics from a publicly available, unlabeled general-domain reference corpus. It then ② computes layer-wise average gradients from current-task data. From these quantities, SVC ③ estimates each singular-vector channel's local adaptation benefit and forgetting cost. SVC next ④ selects trainable channels through knee-based cost screening, Pareto-front filtering, and Otsu thresholding. Finally, SVC ⑤ updates all singular values while updating singular vectors only in selected channels. Post-training, SVC merges the learned increments into the model weights.

O-LoRA (Wang et al., 2023a), a representative LoRA-based continual-learning method, learns successive tasks in orthogonal low-rank subspaces to reduce interference. However, its orthogonality constraint is defined only relative to the low-rank subspaces learned for previous tasks. It does not directly estimate whether changing a particular direction benefits the current domain or compromises pretrained general capabilities. Spectral PEFT methods provide another way to constrain adaptation by exploiting the singular-value structure of pretrained weights. SVF (Sun et al., 2022), SVD-iff (Han et al., 2023), and SAM-PARSER (Peng et al., 2024) keep the left and right singular-vector bases fixed and optimize only the diagonal SVD coefficients, thereby rescaling the existing matched rank-one components. SVFT (Lingam et al., 2024) instead learns a sparse coefficient matrix over fixed singular-vector bases, including off-diagonal coefficients that combine different left and right singular vectors. Across these methods, the set of trainable spectral coefficients is specified before fine-tuning rather than selected using task-specific estimates of adaptation benefit and forgetting cost. SVC instead treats singular-vector channels as the basic units for weighing domain-adaptation benefits against general-capability forgetting costs and adaptively selects trainable directions for each domain.

3 Method

This section presents SVC, covering the preliminaries, the estimation of adaptation benefit and forgetting cost for singular-vector channels, Pareto-frontier-based adaptive channel selection, and the overall fine-tuning procedure. As illustrated in Figure 2, singular value decomposition represents each linear-layer weight matrix as a collection of rank-one components, each defined by a singular value and a corresponding pair of left and right singular vectors.

3.1 Preliminaries

Continual learning requires a model to learn a sequence of tasks while preserving previously acquired capabilities. Consider a task sequence $\mathcal{T} = \{T_1, T_2, \ldots, T_T\}$, where task T_t is associated with a dataset $\mathcal{D}_t = \{(x_{t,i}, y_{t,i})\}_{i=1}^{n_t}$. Let θ denote the parameters of the model $f(\cdot;\theta)$. When training on task T_t, the model is initialized with θ_{t-1}, obtained after the preceding task, and updates its parameters by minimizing the current-task loss:

$$\theta_t = \arg\min_{\theta} \frac{1}{n_t} \sum_{j=1}^{n_t} L_t(f(x_{t,j};\theta), y_{t,j}), \tag{1}$$

where $L_t(\cdot,\cdot)$ denotes the loss function for task T_t, e.g., cross-entropy.

We treat pretraining as an inaccessible stage that precedes T_1. When learning task T_t, the model has no access to the original pretraining data, pretraining statistics, or earlier-task data. We also keep the model size fixed across tasks to avoid accumulating task-specific modules and increasing storage or deployment overhead.

SVC follows the parameter-efficient fine-tuning paradigm and operates on linear layers. Consider the l-th linear layer before learning task T_t. Its inherited weight matrix is $\mathbf{W}_{t-1}^{(l)} \in \mathbb{R}^{d_{\text{out}}^{(l)} \times d_{\text{in}}^{(l)}}$. Let $\mathbf{X}_{t,\text{in}}^{(l)} \in \mathbb{R}^{N \times d_{\text{in}}^{(l)}}$ denote the input activations of this layer. Here, N denotes the number of activation vectors. The layer output is $\mathbf{X}_{t,\text{out}}^{(l)} = \mathbf{X}_{t,\text{in}}^{(l)} \left(\mathbf{W}_{t-1}^{(l)}\right)^\top$. We decompose the weight matrix as $\mathbf{W}_{t-1}^{(l)} = \mathbf{U}_{t-1}^{(l)} \Sigma_{t-1}^{(l)} \left(\mathbf{V}_{t-1}^{(l)}\right)^\top = \sum_{i=1}^{r_l} \sigma_{t-1,i}^{(l)} \mathbf{u}_{t-1,i}^{(l)} \mathbf{v}_{t-1,i}^{(l)\top}$, where $r_l = \text{rank}(\mathbf{W}_{t-1}^{(l)})$. The corresponding linear transformation can therefore be written as $\mathbf{X}_{t,\text{out}}^{(l)} = \sum_{i=1}^{r_l} \left(\mathbf{X}_{t,\text{in}}^{(l)} \mathbf{v}_{t-1,i}^{(l)}\right) \sigma_{t-1,i}^{(l)} \mathbf{u}_{t-1,i}^{(l)\top}$, where $\sigma_{t-1,i}^{(l)}$, $\mathbf{u}_{t-1,i}^{(l)}$, and $\mathbf{v}_{t-1,i}^{(l)}$ denote the i-th singular value, left singular vector, and right singular vector, respectively. The singular value controls the strength of the corresponding rank-one component, while the right and left singular vectors determine its input and output directions. Motivated by the stable capability retention of singular-value fine-tuning, we treat the paired input and output directions associated with each singular value as a single adaptation unit. We define the corresponding singular-vector channel as [1] $\mathcal{S}_{t-1,i}^{(l)} := \left(\mathbf{u}_{t-1,i}^{(l)}, \mathbf{v}_{t-1,i}^{(l)}\right)$. SVC then selectively enables updates to these channels while keeping all singular values trainable.

3.2 Benefit–Cost Estimation for Singular-Vector Channels

We assess each singular-vector channel by its adaptation benefit and forgetting cost. Since task T_t is learned by minimizing the current-task loss in Equation 1, the loss reduction resulting from a channel update provides a natural measure of its adaptation benefit. Forgetting cost, however, cannot be evaluated in the same way, because the original historical data are unavailable and a unified loss is difficult to construct for capturing changes in the model's broad general capabilities. Inspired by methods such as GPM (Saha et al., 2021) that constrain updates based on activation subspaces, we therefore use linear-layer output drift on historical activations as a local proxy for forgetting cost. Although these activations are unavailable, their effect can be approximated using activation statistics collected from unlabeled general-domain data. Let $\Delta\mathbf{W}_{t,i}^{(l)}$ denote the directional weight change induced by updating singular-vector channel $\mathcal{S}_{t-1,i}^{(l)}$. The corresponding adaptation benefit $B_{t,i}^{(l)}$ and forgetting cost $C_{t,i}^{(l)}$ are conceptually defined as:

$$B_{t,i}^{(l)} = \mathcal{L}_t\left(\mathbf{W}_{t-1}^{(l)}\right) - \mathcal{L}_t\left(\mathbf{W}_{t-1}^{(l)} + \Delta\mathbf{W}_{t,i}^{(l)}\right), \tag{2}$$

$$C_{t,i}^{(l)} = \mathbb{E}_{x \sim \mathcal{D}_{\text{old}}} \left\| \Delta\mathbf{W}_{t,i}^{(l)} x^{(l)} \right\|_2^2 = \text{tr}\left(\Delta\mathbf{W}_{t,i}^{(l)} \Sigma_{\text{old}}^{(l)} \Delta\mathbf{W}_{t,i}^{(l)\top}\right), \tag{3}$$

where $\Sigma_{\text{old}}^{(l)} = \mathbb{E}_{x^{(l)} \sim \mathcal{D}_{\text{old}}}[x^{(l)} x^{(l)\top}] \in \mathbb{R}^{d_{\text{in}}^{(l)} \times d_{\text{in}}^{(l)}}$ denotes the second-moment matrix of historical input activations at layer l, serving here as an ideal reference statistic.

However, separately optimizing each channel to obtain $\Delta\mathbf{W}_{t,i}^{(l)}$ would incur prohibitive computational overhead. We therefore use the average current-task gradient at layer l as the basis for estimating each channel's initial directional update tendency:

$$\mathbf{G}_t^{(l)} = \frac{1}{n_t} \sum_{j=1}^{n_t} \nabla_{\mathbf{W}_{t-1}^{(l)}} L_t\left(f(\mathbf{x}_{t,j}; \theta_{t-1}), \mathbf{y}_{t,j}\right), \tag{4}$$

After omitting positive constants shared by all channels within each objective, including the learning-rate terms, the resulting estimates take the following form:

$$B_{t,i}^{(l)} = \|\mathbf{p}_{t,i}^{(l)}\|_2^2 + \|\mathbf{q}_{t,i}^{(l)}\|_2^2, \tag{5}$$

[1] Section 4.3 examines how including singular values in each channel affects formulation and performance.

$$C_{t,i}^{(l)} = \left(\sigma_{t-1,i}^{(l)}\right)^2 \left[\left(\mathbf{v}_{t-1,i}^{(l)\top} \Sigma_{\text{old}}^{(l)} \mathbf{v}_{t-1,i}^{(l)}\right) \|\mathbf{p}_{t,i}^{(l)}\|_2^2 + \mathbf{q}_{t,i}^{(l)\top} \Sigma_{\text{old}}^{(l)} \mathbf{q}_{t,i}^{(l)}\right], \tag{6}$$

where $\mathbf{p}_{t,i}^{(l)}$ and $\mathbf{q}_{t,i}^{(l)}$ denote the effective gradients for updating the left and right singular-vector directions, respectively:

$$\mathbf{p}_{t,i}^{(l)} = \left(\mathbf{I} - \mathbf{u}_{t-1,i}^{(l)} \mathbf{u}_{t-1,i}^{(l)\top}\right) \sigma_{t-1,i}^{(l)} \mathbf{G}_t^{(l)} \mathbf{v}_{t-1,i}^{(l)}, \tag{7}$$

$$\mathbf{q}_{t,i}^{(l)} = \left(\mathbf{I} - \mathbf{v}_{t-1,i}^{(l)} \mathbf{v}_{t-1,i}^{(l)\top}\right) \sigma_{t-1,i}^{(l)} \mathbf{G}_t^{(l)\top} \mathbf{u}_{t-1,i}^{(l)}. \tag{8}$$

These projected gradients are used only for channel scoring before fine-tuning and do not constrain the subsequent optimization of the selected singular-vector directions.

Intuitively, the adaptation benefit measures the magnitude of the initial directional update tendency. The forgetting cost further weights this tendency by its interaction with the second-order statistics of historical activations.

Equations 3 and 6 show that an ideal evaluation of forgetting cost requires unavailable historical activations. However, historical data enter the resulting score only through the second-moment matrix $\mathbf{\Sigma}_{\text{old}}^{(l)}$, whose estimation requires no ground-truth labels. This allows us to approximate these statistics using a more accessible open-source, unlabeled general-domain corpus. In practice, we replace $\mathbf{\Sigma}_{\text{old}}^{(l)}$ with the second-moment matrix computed from the shared general-domain reference corpus. Detailed derivations of the adaptation benefit and forgetting cost are provided in Appendix A.

3.3 Pareto-Frontier-Based Adaptive Channel Selection

Given the scores estimated in Section 3.2, we seek channels with low forgetting cost and sufficient adaptation benefit. A weighted combination of the two objectives would require a trade-off coefficient that may vary across layers and tasks. We therefore use a three-stage procedure: knee-based cost screening excludes high-cost channels, Pareto-front filtering removes dominated candidates, and an adaptive efficiency cutoff removes channels with unfavorable benefit–cost trade-offs.

First, we log-transform the scores to reduce their scale variation: $b_{t,i}^{(l)} = \log(\max\{B_{t,i}^{(l)}, \epsilon\})$ and $c_{t,i}^{(l)} = \log(\max\{C_{t,i}^{(l)}, \epsilon\})$, where ϵ ensures numerical stability. We sort the log-cost scores as $c_{t,(1)}^{(l)} \leq \cdots \leq c_{t,(r_l)}^{(l)}$ and apply Kneedle to the resulting rank–cost curve, using a convex increasing configuration with sensitivity $S = 1.0$ (Satopaa et al., 2011). The detected knee $k_t^{(l)}$ defines $c_{t,*}^{(l)} = c_{t,(k_t^{(l)})}^{(l)}$. If no knee is detected, we set $c_{t,*}^{(l)} = +\infty$, retaining all channels at this stage. The resulting candidate set is

$$\mathcal{F}_t^{(l)} = \left\{ \mathcal{S}_{t-1,i}^{(l)} \;\middle|\; c_{t,i}^{(l)} \leq c_{t,*}^{(l)} \right\}. \tag{9}$$

Within $\mathcal{F}_t^{(l)}$, we retain the nondominated channels through Pareto-front filtering. A channel $\mathcal{S}_{t-1,i}^{(l)}$ dominates $\mathcal{S}_{t-1,j}^{(l)}$ if it has no lower benefit and no higher cost, with at least one strict inequality. The Pareto frontier is defined as

$$\begin{aligned} \mathcal{P}_t^{(l)} &= \left\{ \mathcal{S}_{t-1,j}^{(l)} \in \mathcal{F}_t^{(l)} \;\middle|\; \nexists\, \mathcal{S}_{t-1,i}^{(l)} \in \mathcal{F}_t^{(l)} : \mathcal{S}_{t-1,i}^{(l)} \succ \mathcal{S}_{t-1,j}^{(l)} \right\}, \\ \mathcal{S}_{t-1,i}^{(l)} \succ \mathcal{S}_{t-1,j}^{(l)} &\iff \left(b_{t,i}^{(l)} \geq b_{t,j}^{(l)}\right) \wedge \left(c_{t,i}^{(l)} \leq c_{t,j}^{(l)}\right) \\ &\quad \wedge \left[\left(b_{t,i}^{(l)} > b_{t,j}^{(l)}\right) \vee \left(c_{t,i}^{(l)} < c_{t,j}^{(l)}\right)\right]. \end{aligned} \tag{10}$$

Pareto-front filtering removes dominated channels, but some remaining channels may still offer insufficient benefit relative to their costs. We therefore define the log-efficiency score $r_{t,i}^{(l)} = b_{t,i}^{(l)} - c_{t,i}^{(l)}$.

Let $\mathcal{I}_{P,t}^{(l)} = \left\{ i \in \{1, \ldots, r_l\} \;\middle|\; \mathcal{S}_{t-1,i}^{(l)} \in \mathcal{P}_t^{(l)} \right\}$, and $\pi_t^{(l)}$ be a permutation of $\mathcal{I}_{P,t}^{(l)}$ that orders its channels by decreasing log-efficiency: $r_{t,\pi_t^{(l)}(1)}^{(l)} \geq \cdots \geq r_{t,\pi_t^{(l)}(|\mathcal{I}_{P,t}^{(l)}|)}^{(l)}$. We apply Otsu's method

to the efficiency scores of the Pareto-front channels (Otsu, 1979). If it yields a valid two-group partition, channels with scores no smaller than the threshold are selected. Otherwise, all channels in $\mathcal{P}_t^{(l)}$ are retained. The resulting cutoff determines the number of selected channels $K_t^{(l)}$. The final selected set is

$$\mathcal{A}_t^{(l)} = \left\{ \mathcal{S}^{(l)}_{t-1,\pi_t^{(l)}(k)} \;\middle|\; 1 \le k \le K_t^{(l)} \right\}. \tag{11}$$

3.4 SVC

We illustrate the SVC fine-tuning procedure on task T_t. Before fine-tuning, SVC computes layer-wise activation second moments from an unlabeled general-domain corpus and average gradients from the current-task data. SVC then applies SVD to each layer and scores its singular-vector channels using Equations 5 and 6. For each layer, SVC obtains $\mathcal{A}_t^{(l)}$ through knee-based cost screening, Pareto-front filtering, and adaptive thresholding, as defined in Equations 9–11. This selection is completed before fine-tuning and remains fixed throughout training on T_t.

During fine-tuning, $\mathbf{W}_{t-1}^{(l)}$ remains frozen, and only structured increments are optimized. Specifically, all singular values are assigned trainable increments, whereas singular-vector increments are registered only for channels in $\mathcal{A}_t^{(l)}$. Let $\delta\sigma_{t,i}^{(l)}$, $\delta\mathbf{u}_{t,i}^{(l)}$, and $\delta\mathbf{v}_{t,i}^{(l)}$ denote the corresponding increments, and let N_{lin} denote the number of linear layers adapted by SVC. The trainable parameter sets for layer l and task T_t are

$$\Phi_t^{(l)} = \left\{ \delta\sigma_{t,i}^{(l)} \right\}_{i=1}^{r_l} \cup \left\{ (\delta\mathbf{u}_{t,i}^{(l)}, \delta\mathbf{v}_{t,i}^{(l)}) \;\middle|\; \mathcal{S}_{t-1,i}^{(l)} \in \mathcal{A}_t^{(l)} \right\}, \qquad \Phi_t = \bigcup_{l=1}^{N_{\text{lin}}} \Phi_t^{(l)}. \tag{12}$$

All incremental parameters are initialized to zero. During fine-tuning, the updated factors and the forward pass of layer l are given by

$$\begin{aligned} \widetilde{\sigma}_{t,i}^{(l)} &= \sigma_{t-1,i}^{(l)} + \delta\sigma_{t,i}^{(l)}, \\ \left(\widetilde{\mathbf{u}}_{t,i}^{(l)}, \widetilde{\mathbf{v}}_{t,i}^{(l)}\right) &= \begin{cases} \left(\mathbf{u}_{t-1,i}^{(l)} + \delta\mathbf{u}_{t,i}^{(l)}, \mathbf{v}_{t-1,i}^{(l)} + \delta\mathbf{v}_{t,i}^{(l)}\right), & \mathcal{S}_{t-1,i}^{(l)} \in \mathcal{A}_t^{(l)}, \\ \left(\mathbf{u}_{t-1,i}^{(l)}, \mathbf{v}_{t-1,i}^{(l)}\right), & \mathcal{S}_{t-1,i}^{(l)} \notin \mathcal{A}_t^{(l)}, \end{cases} \\ \mathbf{X}_{t,\text{out}}^{(l)} &= \sum_{i=1}^{r_l} \left(\mathbf{X}_{t,\text{in}}^{(l)} \widetilde{\mathbf{v}}_{t,i}^{(l)}\right) \widetilde{\sigma}_{t,i}^{(l)} \widetilde{\mathbf{u}}_{t,i}^{(l)\top}. \end{aligned} \tag{13}$$

After fine-tuning, the learned factor changes are merged into the inherited weight: $\mathbf{W}_t^{(l)} = \mathbf{W}_{t-1}^{(l)} + \sum_{i=1}^{r_l} \left[\widetilde{\sigma}_{t,i}^{(l),*} \widetilde{\mathbf{u}}_{t,i}^{(l),*} \widetilde{\mathbf{v}}_{t,i}^{(l),*\top} - \sigma_{t-1,i}^{(l)} \mathbf{u}_{t-1,i}^{(l)} \mathbf{v}_{t-1,i}^{(l)\top} \right]$, where the superscript $*$ denotes the corresponding values after fine-tuning. The merged weight $\mathbf{W}_t^{(l)}$ is then carried forward to task T_{t+1}. Thus, SVC requires no persistent task-specific modules and introduces no task-wise model growth.

4 Experiments

We evaluate SVC across model families and against continual learning baselines.

4.1 Experimental Setup

Datasets and Metrics. For sequential learning, we use TRACE (Wang et al., 2023b), covering science (ScienceQA), finance (FOMC), summarization (MeetingBank), multilingual tasks (German 20Minuten and Chinese C-STANCE), code (Py150), and mathematics (NumGLUE-cm and NumGLUE-ds). Classification and single-token question-answering tasks are evaluated by accuracy, while other generation tasks use the average of ROUGE-L and BLEU. Let $R_{t,i}$ denote performance on task i after learning task t. We report Overall Performance (OP) (Chaudhry et al., 2018), $OP_t = \frac{1}{t}\sum_{i=1}^{t} R_{t,i}$, and Backward Transfer (BWT) (Lopez-Paz & Ranzato, 2017), $BWT_t = \frac{1}{t-1}\sum_{i=1}^{t-1}(R_{t,i} - R_{i,i})$, which measure average learned-task performance and subsequent

changes on previous tasks, respectively. We measure capability preservation using Pass@1 on HumanEval (Chen et al., 2021) for code generation and GSM8K (Cobbe et al., 2021) for mathematical reasoning. To estimate SVC's activation second moments, we sample 30,000 unlabeled examples from the open-source C4 (Raffel et al., 2020) and Stack-v2-Python (Lozhkov et al., 2024) datasets at a 2:1 ratio. This corpus is fixed across all models and tasks. Details are provided in Appendix B.

Baselines. We compare SVC with four baseline categories. **Regularization** methods include EWC (Kirkpatrick et al., 2017), GEM (Lopez-Paz & Ranzato, 2017), and LwF (Li & Hoiem, 2017), which preserve prior knowledge through parameter-importance constraints, gradient projection, and knowledge distillation, respectively. **Replay** reuses past data after each task (Replay) (Rebuffi et al., 2017) or throughout training (Replay-online) (Rolnick et al., 2019). **PEFT** includes SeqLoRA, O-LoRA (Wang et al., 2023a), PiSSA (Meng et al., 2024), and MiLoRA (Wang et al., 2025a). O-LoRA enforces orthogonality among task-specific LoRA subspaces. PiSSA updates adapter components initialized from the largest singular values, whereas MiLoRA uses the smallest-value components while freezing the dominant ones. PiSSA and MiLoRA thus make opposing fixed selections based on singular-value magnitude. In addition, we include OSFT (Nayak et al., 2026), an SVD-based constrained full-parameter fine-tuning method. OSFT uses previous-task data to identify important parameter subspaces and restricts new-task updates to their orthogonal complements. Sequential full-parameter fine-tuning (SeqFT) is also included. Details are provided in Appendix C.

Models and Implementation. We evaluate SVC on Llama3-8B (Grattafiori et al., 2024), Mistral-7B-v0.3 (Jiang et al., 2023), Gemma2-9B-it (Team et al., 2024), and Qwen3.5-9B-Base (Team, 2026), covering four representative open-source model families. All methods use Adam, a batch size of 64, no warmup, and five epochs per task. Learning rates are 10^{-4} for SVC and all PEFT baselines and 10^{-5} otherwise. Given the substantial impact of task order on continual learning (Yoon et al., 2020), drawing on the experimental setups of He et al. (2024) and Wang et al. (2025b), we evaluate all methods under three task orders. All experiments run on four 80-GB NVIDIA A100 GPUs. Further experimental details are provided in Appendix C.

4.2 Main Results

Table 1 reports general-capability preservation and sequential task learning across four models and three task orders. We draw the following observations:
(1) SVC effectively preserves general capabilities. Across the 12 model and task-order settings, SVC achieves or ties the best result in 10 HumanEval settings and 11 GSM8K settings. Against the strongest baseline in each setting, SVC achieves mean gains of 1.68 percentage points on HumanEval and 5.18 percentage points on GSM8K. Together, these results show that SVC provides the strongest overall preservation of general capabilities across model families and task orders.
(2) SVC achieves a strong stability–plasticity balance. SVC obtains the best OP in nearly all settings across the four models and three task orders, while its BWT results also indicate substantially reduced forgetting. Despite storing no previous-task samples, SVC consistently surpasses both replay baselines in OP and remains competitive with them in BWT. Together, these results show that SVC acquires new-task knowledge while limiting interference with previously learned tasks.
(3) SVC is robust across task orders. Although task order affects final performance, SVC maintains strong general-capability preservation and sequential task learning across all three orders. It ranks first or second in 46 of the 48 combinations of model, task order, and metric. These results demonstrate that SVC remains robust to task order across different model families.

4.3 Analysis and Discussion

To understand how SVC balances stability and plasticity, we examine three questions about its channel-level design on Llama3-8B under Order 1.

1. Are benefit and cost scores meaningful? Because SVC selects singular-vector channels to balance learning and forgetting, we test whether the scores defined in Section 3.2 reflect actual update outcomes. We first test whether C identifies risky update directions. For each task and linear layer, we match SVC's channel count and select channels from the highest-, middle-, or lowest-cost region, with all other settings fixed. As shown in Table 2, the retention of pretrained capabilities improves consistently as the cost of the selected channels decreases. High-cost selection also affects

Table 1: Main results across four LLMs and three task orders. HumanEval and GSM8K measure general-capability preservation, while OP and BWT evaluate sequential task learning. Higher is better for all metrics. Best and second-best results are shown in bold and underlined, respectively.

Methods	Order 1				Order 2				Order 3			
	HumanEval	GSM8K	OP	BWT	HumanEval	GSM8K	OP	BWT	HumanEval	GSM8K	OP	BWT
Llama3-8B												
SeqFT	0.0	2.88	39.22	-10.69	0.61	4.32	29.78	-10.25	0.0	3.11	20.10	-28.79
EWC	0.0	3.26	32.95	-10.05	0.0	5.46	34.33	-11.27	2.44	3.26	36.92	-6.74
GEM	0.0	3.34	39.37	-10.24	0.0	2.58	32.03	-12.44	0.0	2.81	39.07	-7.15
LwF	0.0	3.03	33.06	-9.94	1.83	2.73	27.20	-8.30	1.22	2.50	26.69	-13.53
Replay	0.0	2.05	38.62	-2.94	0.61	3.26	41.90	-0.24	0.61	3.34	39.56	-3.87
Replay-online	0.0	3.11	45.47	-1.99	0.0	3.03	41.33	-1.48	0.0	3.79	40.90	-1.64
SeqLoRA	28.05	44.20	46.04	-4.39	27.44	21.83	44.36	-7.62	**30.49**	36.39	32.14	-2.80
O-LoRA	26.83	42.84	39.73	-7.19	26.22	39.95	41.39	-4.07	29.27	37.91	43.43	-2.43
PiSSA	28.05	39.42	47.81	-2.47	30.49	21.30	47.54	-2.33	28.05	16.45	47.21	-2.92
MiLoRA	29.27	45.49	47.90	-1.55	28.66	41.24	47.66	-2.38	**30.49**	46.17	44.15	-4.21
OSFT	0.0	1.97	28.00	-11.82	0.61	2.65	31.38	-7.47	0.0	2.96	27.37	-13.55
SVC (ours)	**30.49**	**51.55**	**49.62**	**-0.94**	**31.10**	**51.02**	**50.48**	**-0.05**	29.88	**47.31**	**49.21**	**-1.43**
Mistral-7B-v0.3												
SeqFT	0.0	1.14	14.76	-22.20	0.0	1.97	23.79	-12.85	0.0	0.99	13.20	-23.57
EWC	0.0	1.06	17.65	-18.72	0.0	2.05	22.44	-13.20	0.0	1.90	15.11	-22.43
GEM	0.0	2.05	16.80	-21.40	0.0	2.27	26.85	-8.73	0.0	0.68	15.13	-21.82
LwF	0.0	0.0	7.13	-23.47	0.0	0.0	16.37	-9.83	0.0	0.76	12.40	-18.06
Replay	0.0	0.45	17.11	-5.10	0.0	1.97	30.14	**-1.06**	0.0	2.12	31.30	-2.37
Replay-online	0.0	0.0	17.36	-6.46	0.0	1.52	32.66	-1.28	0.0	1.52	33.17	**-1.02**
SeqLoRA	16.46	29.11	45.67	-3.91	15.24	20.85	41.63	-11.31	19.51	12.96	**45.17**	-5.48
O-LoRA	12.20	24.64	40.29	-5.16	17.07	23.35	35.92	-11.31	10.37	12.74	39.36	-5.38
PiSSA	21.34	11.90	38.98	-3.70	16.46	19.86	44.67	-3.50	11.59	3.79	44.26	-1.73
MiLoRA	25.61	30.78	46.44	-2.56	24.39	26.76	42.47	-5.05	**25.61**	28.05	43.45	-2.75
OSFT	0.0	2.43	13.36	-18.42	0.0	0.0	2.20	-19.86	0.0	0.0	2.54	-16.52
SVC (ours)	**27.44**	**36.54**	**48.37**	**-0.37**	**29.88**	**37.07**	**48.70**	-1.75	**25.61**	**37.68**	44.78	-1.53
Gemma2-9B-it												
SeqFT	9.76	5.38	34.09	-12.37	7.32	5.53	34.77	-7.30	7.32	1.74	33.97	-11.17
EWC	7.32	2.73	38.58	-6.96	5.49	6.75	36.85	-8.90	9.15	2.12	38.72	-4.53
GEM	10.37	3.11	32.55	-8.98	10.98	6.67	37.22	-5.70	5.49	1.97	33.89	-13.20
LwF	0.0	2.50	26.51	-12.52	1.22	2.58	23.76	-10.12	1.22	3.87	28.18	-9.94
Replay	6.71	3.26	42.30	**-0.66**	0.61	3.49	39.44	**0.51**	3.05	2.35	41.78	-1.32
Replay-online	3.66	2.58	44.12	-1.95	1.22	2.73	38.61	-3.30	3.05	1.97	40.25	-3.32
SeqLoRA	37.20	73.09	50.19	-6.83	32.32	56.86	51.58	-4.63	28.66	3.94	51.97	-4.56
O-LoRA	41.46	63.08	47.91	-5.88	37.80	57.01	45.43	-2.60	15.85	46.40	46.00	-7.03
PiSSA	49.39	67.63	**51.93**	-3.60	48.78	66.19	53.90	-0.92	48.78	55.04	52.45	-2.95
MiLoRA	50.61	73.54	49.91	-4.21	47.56	73.69	53.20	-1.54	44.51	5.53	54.31	-0.91
OSFT	7.32	3.41	30.35	-4.29	0.0	0.99	19.14	-12.29	0.0	0.76	11.14	-23.95
SVC (ours)	**56.10**	**75.51**	50.25	-1.37	**53.66**	**74.22**	**54.95**	-0.72	**53.66**	**74.91**	**55.84**	**-0.11**
Qwen3.5-9B-Base												
SeqFT	20.73	11.07	50.23	-3.66	23.78	49.58	46.18	-7.47	17.68	4.25	48.51	-5.24
EWC	21.95	11.75	47.88	-5.57	20.12	48.07	47.38	-3.60	16.46	4.70	48.01	-3.59
GEM	18.29	14.40	49.64	-3.38	16.46	41.62	45.15	-7.32	18.29	4.25	47.26	-3.65
LwF	16.46	28.73	38.23	-8.44	21.34	23.73	31.94	-9.66	22.56	30.17	38.27	-9.17
Replay	29.88	16.07	49.36	-0.99	17.07	12.89	49.97	-0.94	12.20	5.23	46.70	-1.04
Replay-online	24.39	12.13	51.16	-0.78	28.66	36.39	51.91	**-0.16**	12.20	12.05	47.25	-1.79
SeqLoRA	50.61	71.19	49.86	-4.58	50.61	66.94	43.09	-7.46	49.39	**74.60**	50.39	-4.64
O-LoRA	28.05	42.99	39.11	-3.80	34.76	39.12	42.01	-6.88	23.17	27.82	45.05	-6.74
PiSSA	**54.27**	71.87	51.46	-1.31	48.17	67.17	48.09	-7.19	51.22	68.92	54.56	0.13
MiLoRA	42.07	68.92	54.34	**0.59**	49.39	70.13	44.71	-10.09	47.56	69.22	54.28	**2.36**
OSFT	11.59	5.31	47.99	-2.04	17.68	48.14	49.04	-1.78	1.22	4.55	45.83	-5.97
SVC (ours)	43.29	**72.25**	**57.05**	0.02	**56.10**	**71.27**	**56.65**	-0.53	**53.05**	70.20	**57.87**	0.76

sequential learning, suggesting that excessive cost can offset adaptation gains. These results show that C distinguishes risky channel updates and provides a reliable basis for controlling forgetting.

To reduce the influence of C when assessing B, we compare channels under cost-matched conditions. For each task and layer, we rank channels by C and separately take $2K$ candidates from the lowest- and middle-cost regions, where K is SVC's channel count. We pair cost-adjacent channels

Table 2: Validation of channel-level benefit and cost scores on Llama3-8B under Order 1.

Metric	High C	Mid C	Low C	Low B, Mid C	High B, Mid C	Low B, Low C	High B, Low C
HumanEval	0.00	28.05	29.27	27.44	28.66	29.88	29.88
GSM8K	2.65	44.05	51.40	45.49	44.35	52.08	51.71
OP	29.27	47.99	51.12	47.26	47.63	50.66	50.28
BWT	-8.32	-1.06	-0.20	-2.21	-2.51	0.05	-1.19
AP	36.55	48.92	51.30	49.19	49.83	50.62	51.32

and select the higher- or lower-B channel from each pair. Within each region, this produces two disjoint K-channel sets with closely matched costs. Because B measures current-task adaptation, we report $AP_t = \frac{1}{t}\sum_{i=1}^{t} R_{i,i}$, which averages performance immediately after each task is learned. As shown in Table 2, the high-B groups achieve higher AP than their low-B counterparts in both cost regions. Although strict cost matching limits the magnitude of the differences between the two groups, the trend remains consistent across both regions. This consistent trend indicates that B captures the contribution of a channel to current-task adaptation. Together, the results for B and C support their use in balancing learning and forgetting at the level of singular-vector channels.

2. How important is each selection stage? Section 3.3 combines knee-based cost screening, Pareto-front filtering, and Otsu thresholding of benefit–cost efficiency scores. We remove each stage in turn, keep all other settings fixed, and let the remaining stages determine the channel count. Figure 3 shows that removing Pareto-front filtering causes the largest degradation, nearly eliminating the retention of code-generation and mathematical-reasoning capabilities and sharply reducing sequential performance. Removing knee-based cost screening has a smaller effect but still weakens mathematical-reasoning retention, lowers sequential performance, and increases forgetting. These results show that reliable channel selection depends on the complete three-stage procedure. Pareto filtering forms the core, while knee-based screening and the Otsu-based cutoff further restrict high-cost and low-efficiency channels to maintain the overall stability-plasticity balance.

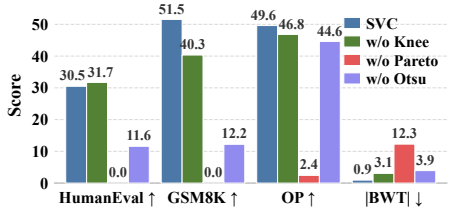

Figure 3: Ablation of the channel-selection stages. BWT is reported in absolute value.

3. Should singular values be selected? By default, SVC selectively updates singular vectors while training all singular values. To test whether singular values should also be selected, we redefine each channel as (σ_i, u_i, v_i), recompute its benefit and cost from the joint perturbation, and apply the same three-stage selection procedure. See Appendix A for the derivation. Compared with default SVC, joint selection decreases HumanEval, GSM8K, OP, and BWT by 1.22, 2.80, 0.65, and 0.56 points, respectively. One possible explanation is that including the singular-value term changes the score distributions, which may reduce the separation between channels and alter both the selected set and its size. Moreover, singular values continue to adjust throughout training, so a local estimate computed before training may not accurately reflect their final contribution. These results support the default SVC design, which selectively updates singular vectors while optimizing all singular values to better balance learning and forgetting.

5 Conclusion

We present Selective Singular-Vector Channel Adaptation (SVC), which treats singular-vector channels as the basic units for balancing adaptation benefits against forgetting costs in continual learning for large language models. SVC estimates both quantities using current-task gradients and activation statistics from an unlabeled general-domain reference corpus, and adaptively selects trainable channels accordingly. Experiments across four LLM families and three task orders show that SVC improves continual-learning performance while preserving pretrained general capabilities, offering a task-adaptive route to stable and scalable continual domain adaptation for LLMs. In future work, we will investigate reference-corpus construction methods that provide broader capability coverage and greater robustness to distribution shifts, as well as channel-selection strategies that account for interactions among channels and adjust dynamically throughout training. We will further evaluate SVC over longer task sequences and a broader range of general-capability benchmarks to characterize the limits of its robustness and computational efficiency.

AI USE STATEMENT

The generative AI model GPT-5.6 was used solely to improve the grammar, wording, clarity, and readability of text written by the authors. It was not used to generate or validate any scientific content, including research ideas, methods, equations, code, data, results, figures, references, or conclusions. All AI-assisted edits were reviewed by the authors, who take full responsibility for the final content.

REPRODUCIBILITY STATEMENT

To support reproducibility, we provide the source code, data, and intermediate experimental outputs at `https://anonymous.4open.science/r/SVC-79F4`. The main experimental settings and evaluation protocols are described in Section 4. Appendix A provides detailed derivations of the adaptation-benefit and forgetting-cost formulations, Appendix B documents the datasets and data usage, and Appendix C presents the baseline configurations, implementation details, and task orders.

REFERENCES

Dan Biderman, Jacob Portes, Jose Javier Gonzalez Ortiz, Mansheej Paul, Philip Greengard, Connor Jennings, Daniel King, Sam Havens, Vitaliy Chiley, Jonathan Frankle, Cody Blakeney, and John Patrick Cunningham. LoRA learns less and forgets less. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL `https://openreview.net/forum?id=aloEru2qCG`. Featured Certification.

Arslan Chaudhry, Puneet K Dokania, Thalaiyasingam Ajanthan, and Philip HS Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *European conference on computer vision*, pp. 556–572. Springer, 2018.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Yanyuan Chen, Dexuan Xu, Yu Huang, Songkun Zhan, Hanpin Wang, Dongxue Chen, Xueping Wang, Meikang Qiu, and Hang Li. Mimo: A medical vision language model with visual referring multimodal input and pixel grounding multimodal output. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 24732–24741, 2025.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Ning Ding, Yujia Qin, Guang Yang, Fuchao Wei, Zonghan Yang, Yusheng Su, Shengding Hu, Yulin Chen, Chi-Min Chan, Weize Chen, et al. Parameter-efficient fine-tuning of large-scale pre-trained language models. *Nature machine intelligence*, 5(3):220–235, 2023.

Robert M French. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4):128–135, 1999.

Annette Rios Gonzales, Nicolas Spring, Tannon Kew, Marek Kostrzewa, Andreas Säuberli, Mathias Müller, and Sarah Ebling. A new dataset and efficient baselines for document-level text simplification in german. In *Proceedings of the Third Workshop on New Frontiers in Summarization*, pp. 152–161, 2021.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Ligong Han, Yinxiao Li, Han Zhang, Peyman Milanfar, Dimitris Metaxas, and Feng Yang. Svdiff: Compact parameter space for diffusion fine-tuning. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 7289–7300. IEEE, 2023.

Jinghan He, Haiyun Guo, Kuan Zhu, Zihan Zhao, Ming Tang, and Jinqiao Wang. Seekr: Selective attention-guided knowledge retention for continual learning of large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 3254–3266, 2024.

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. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Yebowen Hu, Timothy Ganter, Hanieh Deilamsalehy, Franck Dernoncourt, Hassan Foroosh, and Fei Liu. Meetingbank: A benchmark dataset for meeting summarization. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 16409–16423, 2023.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023. URL https://arxiv.org/abs/2310.06825.

Nikitas Karanikolas, Eirini Manga, Nikoletta Samaridi, Eleni Tousidou, and Michael Vassilakopoulos. Large language models versus natural language understanding and generation. In *Proceedings of the 27th Pan-Hellenic Conference on Progress in Computing and Informatics*, pp. 278–290, 2023.

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

Xinlong Li, Weijieying Ren, Wei Qin, Lei Wang, Tianxiang Zhao, and Richang Hong. Analyzing and reducing catastrophic forgetting in parameter efficient tuning. In *ICASSP 2025 - 2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1–5, 2025. doi: 10.1109/ICASSP49660.2025.10889361.

Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017.

Vijay Lingam, Atula Tejaswi, Aditya Vavre, Aneesh Shetty, Gautham K Gudur, Joydeep Ghosh, Alex Dimakis, Eunsol Choi, Aleksandar Bojchevski, and Sujay Sanghavi. Svft: Parameter-efficient fine-tuning with singular vectors. *Advances in neural information processing systems*, 37:41425–41446, 2024.

Hong Liu, Mingsheng Long, Jianmin Wang, and Yu Wang. Learning to adapt to evolving domains. *Advances in neural information processing systems*, 33:22338–22348, 2020.

David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. *Advances in neural information processing systems*, 30, 2017.

Anton Lozhkov, Raymond Li, Loubna Ben Allal, Federico Cassano, Joel Lamy-Poirier, Nouamane Tazi, Ao Tang, Dmytro Pykhtar, Jiawei Liu, Yuxiang Wei, et al. Starcoder 2 and the stack v2: The next generation. *arXiv preprint arXiv:2402.19173*, 2024.

Pan Lu, Swaroop Mishra, Tanglin Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. Learn to explain: Multimodal reasoning via thought chains for science question answering. *Advances in Neural Information Processing Systems*, 35:2507–2521, 2022.

Shuai Lu, Daya Guo, Shuo Ren, Junjie Huang, Alexey Svyatkovskiy, Ambrosio Blanco, Colin Clement, Dawn Drain, Daxin Jiang, Duyu Tang, et al. Codexglue: A machine learning benchmark dataset for code understanding and generation. *arXiv preprint arXiv:2102.04664*, 2021.

Fanxu Meng, Zhaohui Wang, and Muhan Zhang. Pissa: Principal singular values and singular vectors adaptation of large language models. *Advances in Neural Information Processing Systems*, 37:121038–121072, 2024.

Swaroop Mishra, Arindam Mitra, Neeraj Varshney, Bhavdeep Sachdeva, Peter Clark, Chitta Baral, and Ashwin Kalyan. Numglue: A suite of fundamental yet challenging mathematical reasoning tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3505–3523. Association for Computational Linguistics, 2022.

Neil Muller, Lourenço Magaia, and Ben M Herbst. Singular value decomposition, eigenfaces, and 3d reconstructions. *SIAM review*, 46(3):518–545, 2004.

Nikhil Shivakumar Nayak, Krishnateja Killamsetty, Ligong Han, Abhishek Bhandwaldar, Prateek Chanda, Kai Xu, Oleg Silkin, Mustafa Eyceoz, Hao Wang, Aldo Pareja, and Akash Srivastava. Sculpting subspaces: Constrained full fine-tuning in LLMs for continual learning. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=vQcyqsGJDw.

Nobuyuki Otsu. A threshold selection method from gray-level histograms. *IEEE transactions on systems, man, and cybernetics*, 9(1):62–66, 1979.

Zelin Peng, Zhengqin Xu, Zhilin Zeng, Xiaokang Yang, and Wei Shen. Sam-parser: Fine-tuning sam efficiently by parameter space reconstruction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 4515–4523, 2024.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67, 2020.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 2001–2010, 2017.

David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. Experience replay for continual learning. *Advances in neural information processing systems*, 32, 2019.

Gobinda Saha, Isha Garg, and Kaushik Roy. Gradient projection memory for continual learning. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=3AOj0RCNC2.

Ville Satopaa, Jeannie Albrecht, David Irwin, and Barath Raghavan. Finding a" kneedle" in a haystack: Detecting knee points in system behavior. In *2011 31st international conference on distributed computing systems workshops*, pp. 166–171. IEEE, 2011.

Agam Shah, Suvan Paturi, and Sudheer Chava. Trillion dollar words: A new financial dataset, task & market analysis. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 6664–6679, 2023.

Haizhou Shi, Zihao Xu, Hengyi Wang, Weiyi Qin, Wenyuan Wang, Yibin Wang, Zifeng Wang, Sayna Ebrahimi, and Hao Wang. Continual learning of large language models: A comprehensive survey. *ACM Computing Surveys*, 58(5):1–42, 2025.

Yanpeng Sun, Qiang Chen, Xiangyu He, Jian Wang, Haocheng Feng, Junyu Han, Errui Ding, Jian Cheng, Zechao Li, and Jingdong Wang. Singular value fine-tuning: Few-shot segmentation requires few-parameters fine-tuning. *Advances in neural information processing systems*, 35: 37484–37496, 2022.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, et al. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*, 2024.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

Tu Vu, Aditya Barua, Brian Lester, Daniel Cer, Mohit Iyyer, and Noah Constant. Overcoming catastrophic forgetting in zero-shot cross-lingual generation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 9279–9300, 2022.

Hanqing Wang, Yixia Li, Shuo Wang, Guanhua Chen, and Yun Chen. Milora: Harnessing minor singular components for parameter-efficient llm finetuning. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 4823–4836, 2025a.

Lingxiang Wang, Hainan Zhang, and Zhiming Zheng. Parameter importance-driven continual learning for foundation models. *arXiv preprint arXiv:2511.15375*, 2025b.

Xiao Wang, Tianze Chen, Qiming Ge, Han Xia, Rong Bao, Rui Zheng, Qi Zhang, Tao Gui, and Xuan-Jing Huang. Orthogonal subspace learning for language model continual learning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 10658–10671, 2023a.

Xiao Wang, Yuansen Zhang, Tianze Chen, Songyang Gao, Senjie Jin, Xianjun Yang, Zhiheng Xi, Rui Zheng, Yicheng Zou, Tao Gui, et al. Trace: A comprehensive benchmark for continual learning in large language models. *arXiv preprint arXiv:2310.06762*, 2023b.

Jaehong Yoon, Saehoon Kim, Eunho Yang, and Sung Ju Hwang. Scalable and order-robust continual learning with additive parameter decomposition. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=r1gdj2EKPB.

Chenye Zhao, Yingjie Li, and Cornelia Caragea. C-stance: A large dataset for chinese zero-shot stance detection. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13369–13385, 2023.

APPENDIX

A DERIVATION OF ADAPTATION BENEFIT AND FORGETTING COST

For notational simplicity, we suppress the task index t and layer index l throughout this subsection, while retaining the channel index i. Accordingly, W, G_{new}, $\mathcal{L}_{\text{new}}$, and Σ_{old} correspond to $\mathbf{W}_{t-1}^{(l)}$, $\mathbf{G}_t^{(l)}$, $\mathcal{L}_t$, and $\mathbf{\Sigma}_{\text{old}}^{(l)}$, respectively. The task and layer indices are restored directly in the main-text expressions. Consider a linear layer $y = Wx$, where $W \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ is the weight matrix before fine-tuning and $x \in \mathbb{R}^{d_{\text{in}}}$ is the input activation. The SVD of W is

$$W = U\Sigma V^\top = \sum_{i=1}^{r} \sigma_i \mathbf{u}_i \mathbf{v}_i^\top, \tag{14}$$

where $r = \text{rank}(W)$. We refer to $\mathcal{S}_i = (\mathbf{u}_i, \mathbf{v}_i)$ as the i-th singular-vector channel and to σ_i as its associated singular value. Together, they define the rank-one component $\sigma_i \mathbf{u}_i \mathbf{v}_i^\top$. The following benefit and cost analysis is carried out at the granularity of individual singular-vector channels.

The adaptation benefit is naturally characterized by the reduction in domain loss. Forgetting, however, cannot be fully characterized by a single loss, so following methods such as GPM, we instead measure it by the output drift of the linear layer on old-task input activations. Let $\mathcal{L}_{\text{new}}(W)$ denote the new-task loss, and let $\Sigma_{\text{old}} = \mathbb{E}_{x \sim \mathcal{D}_{\text{old}}}[xx^\top] \in \mathbb{R}^{d_{\text{in}} \times d_{\text{in}}}$ denote the second-moment matrix of old-task input activations. For the local update ΔW_i associated with the i-th singular-vector channel, we define its adaptation benefit B_i and forgetting cost C_i as

$$B_i = \mathcal{L}_{\text{new}}(W) - \mathcal{L}_{\text{new}}(W + \Delta W_i) \tag{15}$$

$$C_i = \mathbb{E}_{x \sim \mathcal{D}_{\text{old}}}[\|\Delta W_i x\|_2^2] = \text{tr}\left(\Delta W_i \Sigma_{\text{old}} \Delta W_i^\top\right) \tag{16}$$

In the default SVC formulation, channel selection determines whether the two singular-vector directions $(\mathbf{u}_i, \mathbf{v}_i)$ are trainable, whereas all singular values are optimized regardless of the selection result. We therefore first derive the default benefit and cost scores from the singular-vector perturbations. We then include the singular-value perturbation separately to obtain the alternative scores used in the corresponding ablation.

For completeness, consider small perturbations of the associated singular value and the two singular vectors:

$$\sigma_i \to \sigma_i + \delta\sigma_i \qquad \mathbf{u}_i \to \mathbf{u}_i + \delta\mathbf{u}_i, \qquad \mathbf{v}_i \to \mathbf{v}_i + \delta\mathbf{v}_i$$

The local weight update is

$$\begin{aligned}\Delta W_i &= \delta\sigma_i \mathbf{u}_i\mathbf{v}_i^\top + \sigma_i\delta\mathbf{u}_i\mathbf{v}_i^\top + \sigma_i\mathbf{u}_i\delta\mathbf{v}_i^\top + O(\delta^2)\\ &= (\delta\sigma_i + \sigma_i\mathbf{u}_i^\top\delta\mathbf{u}_i + \sigma_i\delta\mathbf{v}_i^\top\mathbf{v}_i)\mathbf{u}_i\mathbf{v}_i^\top + \sigma_i\delta\mathbf{u}_{i,\perp}\mathbf{v}_i^\top + \sigma_i\mathbf{u}_i\delta\mathbf{v}_{i,\perp}^\top + O(\delta^2)\\ &= \Delta\sigma_i\mathbf{u}_i\mathbf{v}_i^\top + \sigma_i\delta\mathbf{u}_{i,\perp}\mathbf{v}_i^\top + \sigma_i\mathbf{u}_i\delta\mathbf{v}_{i,\perp}^\top + O(\delta^2) \end{aligned} \tag{17}$$

where $\Delta\sigma_i = \delta\sigma_i + \sigma_i\mathbf{u}_i^\top\delta\mathbf{u}_i + \sigma_i\delta\mathbf{v}_i^\top\mathbf{v}_i$, $\mathbf{u}_i^\top\delta\mathbf{u}_{i,\perp} = 0$, and $\mathbf{v}_i^\top\delta\mathbf{v}_{i,\perp} = 0$. Ignoring second-order perturbation terms, the default channel score isolates the orthogonal singular-vector perturbations. The singular-value perturbation is considered separately in the ablation derived below.

$$\Delta W_i^\perp = \sigma_i\delta\mathbf{u}_{i,\perp}\mathbf{v}_i^\top + \sigma_i\mathbf{u}_i\delta\mathbf{v}_{i,\perp}^\top. \tag{18}$$

Let $G_{\text{new}} = \nabla_W\mathcal{L}_{\text{new}}(W)$ denote the gradient of the new-task loss with respect to the weight matrix. A first-order Taylor expansion of $\mathcal{L}_{\text{new}}$ at W gives

$$\mathcal{L}_{\text{new}}(W + \Delta W_i^\perp) - \mathcal{L}_{\text{new}}(W) \approx \left\langle G_{\text{new}}, \Delta W_i^\perp \right\rangle. \tag{19}$$

Expanding the inner product with equation 18 substituted in,

$$\begin{aligned}\left\langle G_{\text{new}}, \Delta W_i^\perp \right\rangle &= \left\langle G_{\text{new}}, \sigma_i\delta\mathbf{u}_{i,\perp}\mathbf{v}_i^\top + \sigma_i\mathbf{u}_i\delta\mathbf{v}_{i,\perp}^\top \right\rangle\\ &= \sigma_i \operatorname{tr}\left(G_{\text{new}}^\top\delta\mathbf{u}_{i,\perp}\mathbf{v}_i^\top\right) + \sigma_i \operatorname{tr}\left(G_{\text{new}}^\top\mathbf{u}_i\delta\mathbf{v}_{i,\perp}^\top\right)\\ &= \sigma_i \operatorname{tr}\left(\mathbf{v}_i^\top G_{\text{new}}^\top\delta\mathbf{u}_{i,\perp}\right) + \sigma_i \operatorname{tr}\left(\delta\mathbf{v}_{i,\perp}^\top G_{\text{new}}^\top\mathbf{u}_i\right)\\ &= \langle\sigma_i G_{\text{new}}\mathbf{v}_i, \delta\mathbf{u}_{i,\perp}\rangle + \left\langle\sigma_i G_{\text{new}}^\top\mathbf{u}_i, \delta\mathbf{v}_{i,\perp}\right\rangle. \end{aligned} \tag{20}$$

This identifies the gradients associated with $\delta\mathbf{u}_{i,\perp}$ and $\delta\mathbf{v}_{i,\perp}$:

$$\mathbf{g}_{u_i} = \sigma_i G_{\text{new}}\mathbf{v}_i, \qquad \mathbf{g}_{v_i} = \sigma_i G_{\text{new}}^\top\mathbf{u}_i. \tag{21}$$

Since $\delta\mathbf{u}_{i,\perp}$ and $\delta\mathbf{v}_{i,\perp}$ must satisfy $\mathbf{u}_i^\top\delta\mathbf{u}_{i,\perp} = 0$ and $\mathbf{v}_i^\top\delta\mathbf{v}_{i,\perp} = 0$, we project $\mathbf{g}_{u_i}, \mathbf{g}_{v_i}$ onto the orthogonal complements of $\mathbf{u}_i, \mathbf{v}_i$ to obtain the feasible effective gradients:

$$\mathbf{p}_i = (I - \mathbf{u}_i\mathbf{u}_i^\top)\mathbf{g}_{u_i}, \qquad \mathbf{q}_i = (I - \mathbf{v}_i\mathbf{v}_i^\top)\mathbf{g}_{v_i}. \tag{22}$$

Updating along $-\mathbf{p}_i, -\mathbf{q}_i$ with step size η and substituting into equation 18 yields the local weight update along the orthogonal perturbation directions:

$$\Delta W_i^\perp = -\eta\sigma_i\mathbf{p}_i\mathbf{v}_i^\top - \eta\sigma_i\mathbf{u}_i\mathbf{q}_i^\top \tag{23}$$

Substituting equation 23 into the benefit definition, equation 15, and using $\mathbf{u}_i^\top\mathbf{p}_i = 0$, $\mathbf{v}_i^\top\mathbf{q}_i = 0$ to eliminate the cross terms

$$\begin{aligned}B_i &= \mathcal{L}_{\text{new}}(W) - \mathcal{L}_{\text{new}}(W + \Delta W_i^\perp)\\ &\approx -\left\langle G_{\text{new}}, \Delta W_i^\perp \right\rangle\\ &= \eta\sigma_i\left\langle G_{\text{new}}, \mathbf{p}_i\mathbf{v}_i^\top + \mathbf{u}_i\mathbf{q}_i^\top \right\rangle\\ &= \eta\sigma_i \operatorname{tr}(G_{\text{new}}^\top\mathbf{p}_i\mathbf{v}_i^\top) + \eta\sigma_i \operatorname{tr}(G_{\text{new}}^\top\mathbf{u}_i\mathbf{q}_i^\top)\\ &= \eta \operatorname{tr}((\sigma_i G_{\text{new}}\mathbf{v}_i)^\top\mathbf{p}_i) + \eta \operatorname{tr}(\mathbf{q}_i^\top(\sigma_i G_{\text{new}}^\top\mathbf{u}_i))\\ &= \eta \operatorname{tr}((\mathbf{p}_i + \sigma_i(\mathbf{u}_i^\top G_{\text{new}}\mathbf{v}_i)\mathbf{u}_i)^\top\mathbf{p}_i) + \eta \operatorname{tr}(\mathbf{q}_i^\top(\mathbf{q}_i + \sigma_i(\mathbf{v}_i^\top G_{\text{new}}^\top\mathbf{u}_i)\mathbf{v}_i))\\ &= \eta\left(\|\mathbf{p}_i\|_2^2 + \|\mathbf{q}_i\|_2^2\right). \end{aligned} \tag{24}$$

Similarly, substituting equation 23 into the cost definition, equation 16,

$$\begin{aligned}
C_i &= \operatorname{tr}\left(\Delta W_i^{\perp}\Sigma_{\text{old}}(\Delta W_i^{\perp})^{\top}\right) \\
&= \eta^2\sigma_i^2 \operatorname{tr}\left[\left(\mathbf{p}_i\mathbf{v}_i^{\top} + \mathbf{u}_i\mathbf{q}_i^{\top}\right)\Sigma_{\text{old}}\left(\mathbf{v}_i\mathbf{p}_i^{\top} + \mathbf{q}_i\mathbf{u}_i^{\top}\right)\right] \\
&= \eta^2\sigma_i^2 \operatorname{tr}\left(\mathbf{p}_i\mathbf{v}_i^{\top}\Sigma_{\text{old}}\mathbf{v}_i\mathbf{p}_i^{\top}\right) + \eta^2\sigma_i^2 \operatorname{tr}\left(\mathbf{p}_i\mathbf{v}_i^{\top}\Sigma_{\text{old}}\mathbf{q}_i\mathbf{u}_i^{\top}\right) \\
&\quad + \eta^2\sigma_i^2 \operatorname{tr}\left(\mathbf{u}_i\mathbf{q}_i^{\top}\Sigma_{\text{old}}\mathbf{v}_i\mathbf{p}_i^{\top}\right) + \eta^2\sigma_i^2 \operatorname{tr}\left(\mathbf{u}_i\mathbf{q}_i^{\top}\Sigma_{\text{old}}\mathbf{q}_i\mathbf{u}_i^{\top}\right) \\
&= \eta^2\sigma_i^2\left[(\mathbf{v}_i^{\top}\Sigma_{\text{old}}\mathbf{v}_i)\operatorname{tr}(\mathbf{p}_i\mathbf{p}_i^{\top}) + (\mathbf{v}_i^{\top}\Sigma_{\text{old}}\mathbf{q}_i)\operatorname{tr}(\mathbf{u}_i^{\top}\mathbf{p}_i)\right. \\
&\quad \left.+(\mathbf{q}_i^{\top}\Sigma_{\text{old}}\mathbf{v}_i)\operatorname{tr}(\mathbf{p}_i^{\top}\mathbf{u}_i) + (\mathbf{q}_i^{\top}\Sigma_{\text{old}}\mathbf{q}_i)\operatorname{tr}(\mathbf{u}_i^{\top}\mathbf{u}_i)\right] \\
&= \eta^2\sigma_i^2\left[(\mathbf{v}_i^{\top}\Sigma_{\text{old}}\mathbf{v}_i)\|\mathbf{p}_i\|_2^2 + \mathbf{q}_i^{\top}\Sigma_{\text{old}}\mathbf{q}_i\right].
\end{aligned} \tag{25}$$

After omitting the positive learning-rate factors shared by all channels, equation 24 and equation 25 yield the default benefit and forgetting-cost scores reported in the main text.

Singular-value-inclusive ablation. For the ablation in which the channel is instead defined as $\widetilde{\mathcal{S}}_i = (\sigma_i, \mathbf{u}_i, \mathbf{v}_i)$, we include the singular-value perturbation in the same local analysis.

Incorporating the singular-value perturbation, gradient descent gives

$$\Delta\sigma_i = -\eta g_i, \qquad g_i \triangleq \mathbf{u}_i^{\top} G_{\text{new}}\mathbf{v}_i. \tag{26}$$

The full update of channel i is

$$\Delta W_i = -\eta\left(g_i\mathbf{u}_i\mathbf{v}_i^{\top} + \sigma_i\mathbf{p}_i\mathbf{v}_i^{\top} + \sigma_i\mathbf{u}_i\mathbf{q}_i^{\top}\right). \tag{27}$$

Substituting into equation 15,

$$\begin{aligned}
\widetilde{B}_i &= \mathcal{L}_{\text{new}}(W) - \mathcal{L}_{\text{new}}(W + \Delta W_i) \\
&\approx -\langle G_{\text{new}}, \Delta W_i\rangle \\
&= \eta\left\langle G_{\text{new}},\, g_i\mathbf{u}_i\mathbf{v}_i^{\top} + \sigma_i\mathbf{p}_i\mathbf{v}_i^{\top} + \sigma_i\mathbf{u}_i\mathbf{q}_i^{\top}\right\rangle \\
&= \eta g_i \operatorname{tr}(G_{\text{new}}^{\top}\mathbf{u}_i\mathbf{v}_i^{\top}) + \eta\sigma_i \operatorname{tr}(G_{\text{new}}^{\top}\mathbf{p}_i\mathbf{v}_i^{\top}) + \eta\sigma_i \operatorname{tr}(G_{\text{new}}^{\top}\mathbf{u}_i\mathbf{q}_i^{\top}) \\
&= \eta g_i\left(\mathbf{v}_i^{\top}G_{\text{new}}^{\top}\mathbf{u}_i\right) + \eta \operatorname{tr}\big((\sigma_i G_{\text{new}}\mathbf{v}_i)^{\top}\mathbf{p}_i\big) + \eta \operatorname{tr}\big(\mathbf{q}_i^{\top}(\sigma_i G_{\text{new}}^{\top}\mathbf{u}_i)\big) \\
&= \eta g_i^2 + \eta \operatorname{tr}\big((\mathbf{p}_i + \sigma_i(\mathbf{u}_i^{\top}G_{\text{new}}\mathbf{v}_i)\mathbf{u}_i)^{\top}\mathbf{p}_i\big) + \eta \operatorname{tr}\big(\mathbf{q}_i^{\top}(\mathbf{q}_i + \sigma_i(\mathbf{v}_i^{\top}G_{\text{new}}^{\top}\mathbf{u}_i)\mathbf{v}_i)\big) \\
&= \eta\left(g_i^2 + \|\mathbf{p}_i\|_2^2 + \|\mathbf{q}_i\|_2^2\right).
\end{aligned} \tag{28}$$

Substituting into equation 16,

$$\begin{aligned}
\widetilde{C}_i &= \operatorname{tr}\left(\Delta W_i\Sigma_{\text{old}}\Delta W_i^{\top}\right) \\
&= \eta^2 \operatorname{tr}\left[\left(g_i\mathbf{u}_i\mathbf{v}_i^{\top} + \sigma_i\mathbf{p}_i\mathbf{v}_i^{\top} + \sigma_i\mathbf{u}_i\mathbf{q}_i^{\top}\right)\Sigma_{\text{old}}\left(g_i\mathbf{v}_i\mathbf{u}_i^{\top} + \sigma_i\mathbf{v}_i\mathbf{p}_i^{\top} + \sigma_i\mathbf{q}_i\mathbf{u}_i^{\top}\right)\right] \\
&= \eta^2 g_i^2 \operatorname{tr}\left(\mathbf{u}_i\mathbf{v}_i^{\top}\Sigma_{\text{old}}\mathbf{v}_i\mathbf{u}_i^{\top}\right) + \eta^2 g_i\sigma_i \operatorname{tr}\left(\mathbf{u}_i\mathbf{v}_i^{\top}\Sigma_{\text{old}}\mathbf{v}_i\mathbf{p}_i^{\top}\right) + \eta^2 g_i\sigma_i \operatorname{tr}\left(\mathbf{u}_i\mathbf{v}_i^{\top}\Sigma_{\text{old}}\mathbf{q}_i\mathbf{u}_i^{\top}\right) \\
&\quad + \eta^2\sigma_i g_i \operatorname{tr}\left(\mathbf{p}_i\mathbf{v}_i^{\top}\Sigma_{\text{old}}\mathbf{v}_i\mathbf{u}_i^{\top}\right) + \eta^2\sigma_i^2 \operatorname{tr}\left(\mathbf{p}_i\mathbf{v}_i^{\top}\Sigma_{\text{old}}\mathbf{v}_i\mathbf{p}_i^{\top}\right) + \eta^2\sigma_i^2 \operatorname{tr}\left(\mathbf{p}_i\mathbf{v}_i^{\top}\Sigma_{\text{old}}\mathbf{q}_i\mathbf{u}_i^{\top}\right) \\
&\quad + \eta^2\sigma_i g_i \operatorname{tr}\left(\mathbf{u}_i\mathbf{q}_i^{\top}\Sigma_{\text{old}}\mathbf{v}_i\mathbf{u}_i^{\top}\right) + \eta^2\sigma_i^2 \operatorname{tr}\left(\mathbf{u}_i\mathbf{q}_i^{\top}\Sigma_{\text{old}}\mathbf{v}_i\mathbf{p}_i^{\top}\right) + \eta^2\sigma_i^2 \operatorname{tr}\left(\mathbf{u}_i\mathbf{q}_i^{\top}\Sigma_{\text{old}}\mathbf{q}_i\mathbf{u}_i^{\top}\right).
\end{aligned} \tag{29}$$

Reducing each term via $\mathrm{tr}(\mathbf{a}\mathbf{b}^\top M) = \mathbf{b}^\top M\mathbf{a}$, and eliminating cross terms with $\mathbf{u}_i^\top \mathbf{p}_i = 0$, $\mathbf{v}_i^\top \mathbf{q}_i = 0$,

$$\begin{aligned}
\mathrm{tr}\left(\mathbf{u}_i\mathbf{v}_i^\top \Sigma_{\mathrm{old}}\mathbf{v}_i\mathbf{u}_i^\top\right) &= (\mathbf{v}_i^\top \Sigma_{\mathrm{old}}\mathbf{v}_i)(\mathbf{u}_i^\top \mathbf{u}_i) = \mathbf{v}_i^\top \Sigma_{\mathrm{old}}\mathbf{v}_i,\\
\mathrm{tr}\left(\mathbf{u}_i\mathbf{v}_i^\top \Sigma_{\mathrm{old}}\mathbf{v}_i\mathbf{p}_i^\top\right) &= (\mathbf{v}_i^\top \Sigma_{\mathrm{old}}\mathbf{v}_i)(\mathbf{p}_i^\top \mathbf{u}_i) = 0,\\
\mathrm{tr}\left(\mathbf{u}_i\mathbf{v}_i^\top \Sigma_{\mathrm{old}}\mathbf{q}_i\mathbf{u}_i^\top\right) &= (\mathbf{v}_i^\top \Sigma_{\mathrm{old}}\mathbf{q}_i)(\mathbf{u}_i^\top \mathbf{u}_i) = \mathbf{v}_i^\top \Sigma_{\mathrm{old}}\mathbf{q}_i,\\
\mathrm{tr}\left(\mathbf{p}_i\mathbf{v}_i^\top \Sigma_{\mathrm{old}}\mathbf{v}_i\mathbf{u}_i^\top\right) &= (\mathbf{v}_i^\top \Sigma_{\mathrm{old}}\mathbf{v}_i)(\mathbf{u}_i^\top \mathbf{p}_i) = 0,\\
\mathrm{tr}\left(\mathbf{p}_i\mathbf{v}_i^\top \Sigma_{\mathrm{old}}\mathbf{v}_i\mathbf{p}_i^\top\right) &= (\mathbf{v}_i^\top \Sigma_{\mathrm{old}}\mathbf{v}_i)\|\mathbf{p}_i\|_2^2,\\
\mathrm{tr}\left(\mathbf{p}_i\mathbf{v}_i^\top \Sigma_{\mathrm{old}}\mathbf{q}_i\mathbf{u}_i^\top\right) &= (\mathbf{v}_i^\top \Sigma_{\mathrm{old}}\mathbf{q}_i)(\mathbf{u}_i^\top \mathbf{p}_i) = 0,\\
\mathrm{tr}\left(\mathbf{u}_i\mathbf{q}_i^\top \Sigma_{\mathrm{old}}\mathbf{v}_i\mathbf{u}_i^\top\right) &= (\mathbf{q}_i^\top \Sigma_{\mathrm{old}}\mathbf{v}_i)(\mathbf{u}_i^\top \mathbf{u}_i) = \mathbf{v}_i^\top \Sigma_{\mathrm{old}}\mathbf{q}_i,\\
\mathrm{tr}\left(\mathbf{u}_i\mathbf{q}_i^\top \Sigma_{\mathrm{old}}\mathbf{v}_i\mathbf{p}_i^\top\right) &= (\mathbf{q}_i^\top \Sigma_{\mathrm{old}}\mathbf{v}_i)(\mathbf{p}_i^\top \mathbf{u}_i) = 0,\\
\mathrm{tr}\left(\mathbf{u}_i\mathbf{q}_i^\top \Sigma_{\mathrm{old}}\mathbf{q}_i\mathbf{u}_i^\top\right) &= (\mathbf{q}_i^\top \Sigma_{\mathrm{old}}\mathbf{q}_i)(\mathbf{u}_i^\top \mathbf{u}_i) = \mathbf{q}_i^\top \Sigma_{\mathrm{old}}\mathbf{q}_i. \qquad (30)
\end{aligned}$$

Substituting back into equation 29 and keeping only the nonzero terms,

$$\begin{aligned}
\widetilde{C}_i &= \eta^2 g_i^2 \left(\mathbf{v}_i^\top \Sigma_{\mathrm{old}}\mathbf{v}_i\right) + 2\eta^2 g_i\sigma_i \left(\mathbf{v}_i^\top \Sigma_{\mathrm{old}}\mathbf{q}_i\right) + \eta^2\sigma_i^2 \left(\mathbf{v}_i^\top \Sigma_{\mathrm{old}}\mathbf{v}_i\right)\|\mathbf{p}_i\|_2^2 + \eta^2\sigma_i^2 \left(\mathbf{q}_i^\top \Sigma_{\mathrm{old}}\mathbf{q}_i\right)\\
&= \eta^2[\sigma_i^2\|\mathbf{p}_i\|_2^2 \left(\mathbf{v}_i^\top \Sigma_{\mathrm{old}}\mathbf{v}_i\right) + \left(g_i\mathbf{v}_i + \sigma_i\mathbf{q}_i\right)^\top \Sigma_{\mathrm{old}} \left(g_i\mathbf{v}_i + \sigma_i\mathbf{q}_i\right)]. \qquad (31)
\end{aligned}$$

The default scores in equation 24-equation 25 therefore quantify the marginal effect of enabling the singular-vector directions, whereas equation 28-equation 31 quantify the joint perturbation of the singular value and its associated singular-vector pair for the corresponding ablation.

B Datasets

B.1 Continual-Learning Tasks

We construct the continual-learning sequences using the eight language tasks from TRACE (Wang et al., 2023b). TRACE combines challenging and heterogeneous tasks for which pretrained language models generally have substantial room for adaptation. The selected tasks cover specialized domains, multiple languages, code completion, and mathematical reasoning. Following the TRACE protocol, we use 1,000 training examples for each task and retain the corresponding test sets for evaluation. Dataset statistics are summarized in Table 3.

Domain-specific tasks. This group evaluates adaptation to scientific, financial, and meeting-related domains.

- **ScienceQA (Sin.)** (Lu et al., 2022) contains questions derived from elementary- and high-school curricula across natural, social, and language science. We retain only the text-based examples, thereby excluding samples that require visual inputs.
- **FOMC (Fom.)** (Shah et al., 2023) is a financial stance classification task in which the model determines whether a Federal Reserve policy statement is hawkish or dovish. We use the combined version containing meeting minutes, press-conference transcripts, and speeches.
- **MeetingBank (Mee.)** (Hu et al., 2023) is a meeting summarization dataset built from city-council proceedings. It requires the model to identify salient information from long transcripts and generate concise summaries.

Multilingual tasks. The multilingual tasks evaluate whether a model can acquire new information and follow task-specific output formats in languages other than English.

- **C-STANCE (Cst.)** (Zhao et al., 2023) is a Chinese stance detection dataset collected from Sina Weibo. We use its target-based setting, in which the targets appearing in the test set are not observed during training.
- **20Minuten (20M.)** (Gonzales et al., 2021) is a German text simplification dataset derived from the Swiss news outlet *20 Minuten*. Each example pairs a news article with a simplified version, making the task a conditional German text-generation problem.

Table 3: Statistics of the continual-learning tasks and capability-preservation benchmarks. A dash indicates that a benchmark is used only for evaluation and is not included in continual training.

Split	Continual-learning tasks								Capability-preservation benchmarks	
	Sin.	Fom.	Mee.	Cst.	20M.	Py150	Ngc.	Ngd.	HumanEval	GSM8K
Train	1,000 examples per task								–	–
Test	2,000	496	692	2,000	200	2,000	81	325	164	1,319

Code-completion task. Py150 (Lu et al., 2021) is constructed from approximately 150,000 Python source files collected from public GitHub repositories. Given a code context, the model must predict the subsequent code content. The task evaluates adaptation under structured inputs and output constraints that differ from ordinary natural-language generation.

Mathematical-reasoning tasks. TRACE includes two tasks derived from NumGLUE (Mishra et al., 2022): **NumGLUE-cm (Ngc.)** and **NumGLUE-ds (Ngd.)**. Both tasks require numerical computation and multi-step reasoning over natural-language problem statements, providing complementary continual-learning tasks for mathematical reasoning.

B.2 Capability-Preservation Benchmarks

We evaluate the preservation of pretrained code-generation and mathematical-reasoning capabilities using HumanEval and GSM8K, respectively. Neither benchmark is included in the continual-training sequences.

HumanEval. HumanEval (Chen et al., 2021) contains 164 hand-written Python programming problems. Each problem specifies a function signature and a natural-language docstring, from which the model must generate a complete function implementation. Correctness is determined by executing the generated code against hidden unit tests. We use the full benchmark to measure whether sequential domain adaptation degrades the model's pretrained code-generation capability.

GSM8K. GSM8K (Cobbe et al., 2021) is a collection of linguistically diverse grade-school mathematics word problems written by human annotators. Its problems typically require multiple steps of elementary arithmetic together with natural-language reasoning. We evaluate models on the complete official test split of 1,319 problems and report the proportion of problems for which the model produces the correct final answer. GSM8K examples are used only for evaluation, allowing changes in performance to reflect the retention of pretrained mathematical-reasoning capability rather than additional training on the benchmark.

B.3 General-Domain Reference Corpus

SVC estimates the potential forgetting cost of each singular-vector channel using activation statistics collected from an unlabeled general-domain reference corpus. We construct this corpus by sampling 30,000 examples from C4 (Raffel et al., 2020) and the Python subset of The Stack v2 (Lozhkov et al., 2024) at a fixed ratio of 2:1. This corresponds to 20,000 C4 examples and 10,000 Python examples. The sampled corpus is fixed and shared across all models, tasks, and task orders.

The reference corpus is not used to optimize model parameters and does not provide supervision for any continual-learning task. We use only its input texts to collect layer-wise activations and estimate the corresponding second-moment matrices required by the forgetting-cost calculation. Thus, the same reference data can be reused throughout the task sequence without storing examples from previous tasks.

C Baselines and Implementation Details

C.1 Baselines

We compare SVC with four groups of continual-learning baselines: regularization and gradient-constraint methods, replay-based methods, parameter-efficient fine-tuning methods, and full or constrained full-parameter fine-tuning methods. Unless otherwise specified, all methods use the same task data, task orders, optimization schedule, and evaluation protocols.

C.1.1 Regularization and Gradient-Constraint Methods

Elastic Weight Consolidation. Elastic Weight Consolidation (EWC) (Kirkpatrick et al., 2017) penalizes changes to parameters that are estimated to be important for previous tasks. When learning task t, its objective is

$$\mathcal{L}_{\mathrm{EWC}} = \mathcal{L}_t(\theta) + \lambda \sum_i F_i \left(\theta_i - \theta_i^*\right)^2, \tag{32}$$

where θ^* denotes the parameters obtained after the preceding task and F_i is the estimated Fisher information for parameter i. We set the regularization coefficient to $\lambda = 0.5$.

Gradient Episodic Memory. Gradient Episodic Memory (GEM) (Lopez-Paz & Ranzato, 2017) maintains an episodic memory $\mathcal{M}_k$ for each previous task k. When learning task t, it computes the current gradient g_t and a reference gradient g_k from each $\mathcal{M}_k$. If $\langle g_t, g_k \rangle < 0$ for any $k < t$, GEM replaces g_t with the solution to

$$\begin{aligned} \min_{\widetilde{g}} \quad & \frac{1}{2}\|\widetilde{g} - g_t\|_2^2, \\ \text{s.t.} \quad & \langle \widetilde{g}, g_k \rangle \geq 0, \qquad \forall k < t. \end{aligned} \tag{33}$$

The resulting gradient is used for parameter updates, limiting interference with the stored examples from earlier tasks.

Learning without Forgetting. Learning without Forgetting (LwF) (Li & Hoiem, 2017) retains a frozen copy of the preceding model and uses its predictions as soft targets. The training objective combines the current-task loss with a knowledge-distillation term:

$$\mathcal{L}_{\mathrm{LwF}} = \mathcal{L}_t^{\mathrm{CE}} + \alpha \mathcal{L}_{\mathrm{KD}}. \tag{34}$$

We use a distillation weight of $\alpha = 0.5$ and a softmax temperature of $T = 2$.

C.1.2 Replay-Based Methods

Both replay baselines maintain a unified memory buffer containing real examples from previously learned tasks. Samples are drawn uniformly across earlier tasks, and no synthetic data or auxiliary generative model is used.

Replay. After training on the current task, Replay performs an additional rehearsal phase using samples drawn from the memory buffer. For each previous task, the buffer stores and replays 1% of its training examples.

Replay-online. Replay-online interleaves historical examples with current-task examples during training. At each stage, approximately 1% of the stored data are sampled from the same memory buffer and mixed with the current-task mini-batches.

C.1.3 Parameter-Efficient Fine-Tuning Methods

For SeqLoRA, O-LoRA, PiSSA, and MiLoRA, we use rank $r = 8$, scaling factor $\alpha = 32$, and apply the adapters to the `q_proj` and `v_proj` modules. SeqLoRA, O-LoRA, and MiLoRA use an adapter dropout of 0.1. For PiSSA, we disable adapter dropout following its original implementation. PiSSA and MiLoRA otherwise use the same LoRA configuration and differ primarily in whether their low-rank factors are initialized from the principal or minor singular components.

Sequential LoRA. Low-Rank Adaptation (LoRA) (Hu et al., 2022) parameterizes the update to a frozen weight matrix W as

$$W' = W + \frac{\alpha}{r} BA, \tag{35}$$

where A and B are trainable low-rank factors. Sequential LoRA (SeqLoRA) continually updates the same adapter parameters across the complete task sequence.

Orthogonal LoRA. Orthogonal LoRA (O-LoRA) (Wang et al., 2023a) assigns successive tasks to task-specific low-rank subspaces and regularizes these subspaces toward orthogonality. It uses the same LoRA configuration as SeqLoRA, with the orthogonality coefficient set to $\gamma = 0.5$.

PiSSA. Principal Singular Values and Singular Vectors Adaptation (PiSSA) (Meng et al., 2024) decomposes each pretrained weight matrix through singular value decomposition. The r largest singular values and their corresponding left and right singular vectors form a rank-r principal component, which is used to initialize the trainable low-rank factors. The remaining singular components are absorbed into a frozen residual matrix, such that the trainable component and the residual together reconstruct the original weight at initialization. PiSSA uses the same rank, scaling factor, and target modules as the other LoRA-based baselines, but disables adapter dropout.

MiLoRA. Minor Singular Component Low-Rank Adaptation (MiLoRA) (Wang et al., 2025a) also decomposes each pretrained weight matrix through singular value decomposition. In contrast to PiSSA, MiLoRA uses the r smallest singular values and their corresponding left and right singular vectors to initialize the trainable low-rank factors. The remaining principal singular components are kept frozen, allowing adaptation to occur within the less dominant singular subspace. MiLoRA uses the same rank, scaling factor, and target modules as SeqLoRA and PiSSA, and uses the same adapter dropout of 0.1 as SeqLoRA.

C.1.4 Full and Constrained Fine-Tuning Methods

Sequential Full-Parameter Fine-Tuning. Sequential full-parameter fine-tuning (SeqFT) updates all model parameters on each task in sequence without explicit regularization, replay, or subspace constraint.

Orthogonal Subspace Fine-Tuning. Orthogonal Subspace Fine-Tuning (OSFT) (Nayak et al., 2026) constrains full-parameter updates through layer-wise singular subspaces. Before learning each task, OSFT decomposes every adapted weight matrix as $W^{(l)} = U^{(l)}\Sigma^{(l)}V^{(l)\top}$. It then estimates the importance $I^{(l)}$ of layer l from the average cosine similarity between that layer's input and output activations on a probing corpus. Layers with higher importance scores are assigned larger protected subspaces. The fraction of singular directions retained at layer l is determined by

$$r^{(l)} = \mathrm{mrr} + I^{(l)} \left(\mathrm{trr} - \mathrm{mrr}\right), \tag{36}$$

where the normalized importance score $I^{(l)}$ controls the allocation between the minimum retention ratio mrr and the target retention ratio trr. We follow the original configuration and set $\mathrm{mrr} = 0.1$ and $\mathrm{trr} = 0.8$.

Based on $r^{(l)}$, OSFT selects the singular vectors associated with the largest singular values to form the protected left and right subspaces, denoted by $U^{(l)}_{\mathrm{high}}$ and $V^{(l)}_{\mathrm{high}}$, respectively. Given the current gradient $G^{(l)}$, OSFT removes the component aligned with both protected subspaces:

$$G^{(l)}_{\mathrm{proj}} = G^{(l)} - U^{(l)}_{\mathrm{high}} U^{(l)\top}_{\mathrm{high}} G^{(l)} V^{(l)}_{\mathrm{high}} V^{(l)\top}_{\mathrm{high}}. \tag{37}$$

The projected gradient $G^{(l)}_{\mathrm{proj}}$ is then used to update the full weight matrix.

The original OSFT protocol estimates layer importance using samples from the preceding task. This is incompatible with our setting, in which historical task data cannot be revisited and the original pretraining data are unavailable before the first continual-learning task. We therefore use the same fixed, unlabeled general-domain reference corpus employed by SVC as the probing corpus for OSFT. All other OSFT configurations follow the original method.

Table 4: Effect of target-module coverage on Llama3-8B under Order 1. The default PiSSA and MiLoRA results and the SVC result are reproduced from Table 1. Higher is better for all metrics.

Method	Target modules	HumanEval	GSM8K	OP	BWT
PiSSA	q_proj, v_proj	28.05	39.42	47.81	−2.47
PiSSA	All linear	10.37	3.56	39.80	−4.19
MiLoRA	q_proj, v_proj	29.27	45.49	47.90	−1.55
MiLoRA	All linear	24.39	41.55	48.46	−1.65
SVC	All linear	**30.49**	**51.55**	**49.62**	**−0.94**

Additional implementation details. Unless otherwise specified above, all methods follow the common optimization and training settings described in Section 4.1. SVC is applied to all linear modules in the model. Its trainable singular-vector channels are selected adaptively for each layer and task, without a manually specified global selection ratio.

C.2 Task Orders

We evaluate every method under three task orders to account for the sensitivity of continual-learning performance to task ordering. Each order contains all eight TRACE tasks, and the same orders are used for every model and baseline.

Order 1: C-STANCE → FOMC → MeetingBank → ScienceQA → NumGLUE-cm → NumGLUE-ds → 20Minuten → Py150.

Order 2: NumGLUE-cm → NumGLUE-ds → FOMC → 20Minuten → C-STANCE → MeetingBank → ScienceQA → Py150.

Order 3: MeetingBank → FOMC → 20Minuten → C-STANCE → ScienceQA → NumGLUE-cm → NumGLUE-ds → Py150.

C.3 Motivation Experiment Details

Figure 1 compares LoRA and SVF under independent adaptation to the eight TRACE tasks. Each run starts from the same Llama3-8B pretrained checkpoint and uses the task data and splits described in Section 4.1 and Appendix B.1. LoRA follows the configuration in Appendix C.1.3, with rank $r = 8$, scaling factor $\alpha = 32$, dropout 0.1, and adapters on q_proj and v_proj. SVF updates all singular values in every linear layer while keeping the corresponding left and right singular vectors fixed. Both methods use the training settings in Section 4.1: Adam, a batch size of 64, five epochs, no warmup, and a learning rate of 10^{-4}. Figure 1(a) reports GSM8K Pass@1 before and after each independent adaptation. Figure 1(b) reports performance on each task's test set using the metrics defined in Section 4.1.

D Effect of Expanding PEFT Target Modules

To examine whether SVC's advantage can be explained solely by its broader module coverage, we expanded the target modules of PiSSA and MiLoRA from q_proj and v_proj to all linear modules. We evaluated both methods on Llama3-8B under Order 1, keeping their other experimental settings unchanged. Table 4 compares these runs with the corresponding main results. Expanding the target modules reduced HumanEval and GSM8K performance for both methods. PiSSA also showed lower OP and BWT. MiLoRA's OP increased slightly, from 47.90 to 48.46, while its BWT decreased from -1.55 to -1.65. SVC outperformed both expanded baselines on all four metrics. These results indicate that broader module coverage alone does not account for SVC's advantage in this setting.