OSD: ORTHOGONAL SPECTRAL DESCENT FOR CONTINUAL LEARNING IN LARGE LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Continual fine-tuning enables large language models (LLMs) to adapt to evolving tasks and domains, but sequential adaptation often leads to catastrophic forgetting of previously acquired capabilities. Projection-based continual learning methods mitigate this interference by constraining updates away from directions associated with past tasks, but typically select feasible updates under Euclidean/Frobenius geometry. In parallel, matrix-aware optimizers such as Muon suggest an alternative: selecting matrix-valued updates through spectral-norm-constrained steepest descent. We introduce Orthogonal Spectral Descent (OSD), a practical continual-learning instantiation of spectral steepest descent that combines protected task subspaces with efficient Muon-compatible constrained updates. OSD formulates each matrix update as a spectral-norm-bounded descent problem subject to linear non-interference constraints and solves it through lightweight warm-started dual corrections and Newton–Schulz matrix-sign iterations, avoiding an exact SVD at every optimizer step. Across standard continual-learning, TRACE, and heterogeneous domain-adaptation benchmarks, OSD consistently improves the stability–plasticity tradeoff across encoder–decoder and decoder-only LLMs. On the Standard CL benchmark, OSD raises average accuracy from 28.5% with sequential fine-tuning to **78.9%** (**2.8**×), while outperforming the strongest protected-subspace baseline by **3.0%** points. Controlled ablations further show that OSD improves average accuracy by **4.1%** points over its Frobenius-geometry counterpart while reducing training time by **14.3%**, supporting the complementary roles of protected task subspaces and spectral update geometry. These results establish update geometry as an important design dimension for continual LLM fine-tuning.

1 INTRODUCTION

Large-scale neural networks, including Large Language Models (LLMs), are increasingly deployed in settings that require sequential adaptation across diverse domains. A central challenge in such continual or post-training regimes is catastrophic forgetting Kirkpatrick et al. (2017); Goodfellow et al. (2013): when models are fine-tuned on new tasks, their performance on previously learned tasks can degrade rapidly. This phenomenon has been extensively documented in the literature on neural networks and, more recently, in large-scale models such as large language models (LLMs).

Early empirical and theoretical studies demonstrate that standard gradient-based training does not preserve previously acquired knowledge in the absence of explicit mechanisms Kirkpatrick et al. (2017); Shin et al. (2017); Yoon et al. (2018). To address this issue, the field of continual learning (also known as lifelong learning) studies how to train models on a sequence of tasks while retaining performance on past tasks. Foundational works establish core problem formulations and evaluation protocols for this setting Li & Hoiem (2017); Kirkpatrick et al. (2017); Lopez-Paz & Ranzato (2017). Formally, continual learning considers a sequence of tasks, where the learner observes data from each task in turn and must update its parameters using only *limited or no access to past data*, while maintaining strong performance across all previously encountered tasks. The central objective is to balance *plasticity* (the ability to learn new tasks) and *stability* (the ability to retain prior knowledge), a tradeoff that lies at the heart of continual learning research.

A prominent class of continual-learning methods mitigates forgetting by protecting directions associated with past tasks. GEM/A-GEM, OGD, O-LoRA, and OSFT (*Sculpting Subspaces*) instantiate

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.

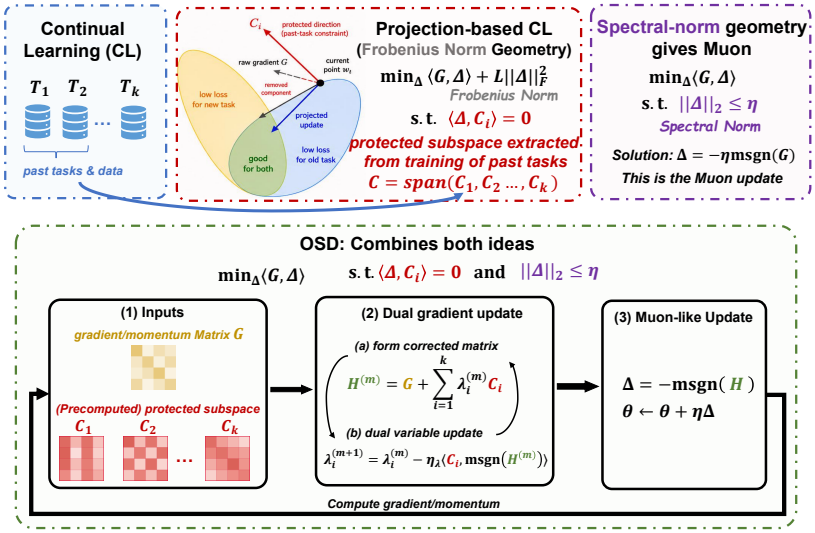

Figure 1: Overview of OSD. **Top left:** projection-based continual learning protects past-task directions through linear non-interference constraints and typically selects the feasible update using Euclidean/Frobenius geometry. **Top right:** Muon can be interpreted as spectral-norm-bounded steepest descent for matrix-valued parameters. **Bottom:** OSD combines the same protected-subspace principle with spectral steepest descent. Warm-started dual corrections modify the momentum matrix H, after which a Muon-compatible matrix-sign direction is approximated with Newton–Schulz iterations.

this idea with gradient, memory, or representation subspaces Lopez-Paz & Ranzato (2017); Chaudhry et al. (2018); Farajtabar et al. (2020a); Xiong & Xie (2026); Nayak et al. (2025). These methods specify *where* a new update should not act; their feasible update is typically selected using Euclidean/Frobenius geometry.

Muon Jordan; Liu et al. (2025) motivates a complementary choice for matrix-valued parameters: spectral steepest descent, whose idealized direction is the polar factor of the momentum matrix. For a corrected matrix $H = U\Sigma V^\top$, the spectral direction is proportional to $-UV^\top$, whereas the Frobenius steepest direction is proportional to $-H/\|H\|_F$. Protected-subspace constraints do not by themselves control how concentrated an update can be in the remaining feasible space, while an operator-norm bound controls worst-direction feature perturbation,

$$\|\Delta W h\|_2 \le \|\Delta W\|_2 \, \|h\|_2. \tag{1}$$

This suggests that protected directions and update geometry can play complementary roles in the stability–plasticity tradeoff.

We therefore introduce *OSD*[1], a practical continual-learning instantiation of constrained spectral steepest descent. OSD jointly enforces an operator-norm trust region and past-task non-interference constraints. It is not obtained by simply applying Muon before or after a Euclidean projection: the constraint multipliers modify the matrix whose polar factor defines the spectral descent direction. We derive a low-dimensional dual formulation and approximate it with warm-started dual iterations and Newton–Schulz matrix-sign updates, avoiding a full SVD at every optimizer step.

Our contributions are as follows:

- **Spectral steepest descent for continual LLM adaptation.** We formulate continual fine-tuning as spectral-norm-bounded steepest descent subject to linear non-interference constraints derived from protected past-task directions. This provides a direct way to combine projection-based continual learning with the update geometry of matrix-aware optimizers.
- **An efficient Muon-compatible constrained update.** We develop OSD, a practical solver for the resulting constrained problem based on its dual formulation. OSD performs lightweight

[1]Anonymous code repository: `https://anonymous.4open.science/r/OSD-BA06`

dual corrections together with Newton–Schulz approximations of the matrix-sign direction, enabling constrained spectral updates without requiring an exact SVD or expensive projection in the full parameter space.

- **Evaluation of the stability–plasticity tradeoff in LLM continual fine-tuning.** We evaluate OSD across standard continual-learning benchmarks, the 15-task benchmark, TRACE, and domain-specific sequential fine-tuning settings with both encoder–decoder and decoder-only architectures. Across the evaluated settings, OSD improves retention and overall continual-learning performance over sequential fine-tuning and competitive orthogonal-gradient baselines while maintaining strong adaptation to newly introduced tasks.

2 Related Work

Continual Learning in Large Language Models. Continual learning studies how to adapt models to sequential tasks while retaining prior knowledge Parisi et al. (2019); De Lange et al. (2021); Wang et al. (2024). Classical methods include regularization-based approaches Kirkpatrick et al. (2017); Zenke et al. (2017); Aljundi et al. (2018); Li & Hoiem (2017), replay-based approaches Lopez-Paz & Ranzato (2017); Chaudhry et al. (2018), and prompt- or parameter-efficient methods Wang et al. (2022b;a); Razdaibiedina et al. (2023); Qin & Joty (2021); Hu et al. (2022). In LLMs, sequential instruction tuning can degrade broad capabilities such as reasoning and instruction following, motivating benchmarks such as TRACE and long multi-task streams Wang et al. (2023b); Razdaibiedina et al. (2023); Wang et al. (2023a); Nayak et al. (2025).

Projection-based Continual Learning. Projection-based methods reduce forgetting by constraining update directions to avoid interference with previous tasks. GEM and A-GEM impose episodic-memory gradient constraints Lopez-Paz & Ranzato (2017); Chaudhry et al. (2018), while OGD projects new gradients away from subspaces important for prior tasks Farajtabar et al. (2020a). Recent LLM methods extend this principle to low-rank or full-parameter adaptation, including O-LoRA and Orthogonal Subspace Fine-Tuning (OSFT; *Sculpting Subspaces*) Wang et al. (2023a); Nayak et al. (2025). OSD reuses the protected-subspace principle but changes the geometry used to select a feasible matrix update.

Spectral-Norm-Aware Optimization and Muon. Muon is a recent optimizer for matrix-valued neural-network parameters that orthogonalizes momentum updates using Newton–Schulz iterations Jordan. Empirical and theoretical studies interpret Muon through operator-norm or spectral-norm geometry Liu et al. (2025); **?**); Chen et al. (2025). Spectral information has also been used directly in continual learning: *Learning Continually by Spectral Regularization* regularizes singular values of the *weights* Lewandowski et al. (2025). OSD instead constrains the geometry of the *update* ΔW and couples it to explicit protected-task constraints; our contribution is the joint constrained formulation, its Muon-compatible dual/polar solver, and its evaluation in continual LLM fine-tuning.

3 Preliminaries

We briefly review the two ingredients underlying our method: projection-based continual learning and Muon-style spectral-norm-aware optimization.

3.1 Projection-Based Continual Learning

Projection-based continual learning methods mitigate catastrophic forgetting by restricting new updates to directions that minimally interfere with previous tasks. Representative examples include Orthogonal Gradient Descent (OGD) Farajtabar et al. (2020b), which projects gradients away from past-task directions, and episodic-memory methods such as GEM Lopez-Paz & Ranzato (2017) and A-GEM Chaudhry et al. (2018), which enforce non-increasing loss on stored previous-task samples. Recent LLM methods further extend this principle to LoRA or full fine-tuning by constraining updates away from subspaces associated with pretrained or previous-task representations Zhang et al. (2025); Tekmen et al. (2025).

A representative Euclidean formulation is

$$\min_{\Delta} \langle G, \Delta \rangle_F + \frac{L}{2}\|\Delta\|_F^2 \quad \text{s.t.} \quad \langle C_i, \Delta \rangle_F = 0,\ i = 1, \ldots, k, \tag{2}$$

where $L > 0$ and $\langle A, B \rangle_F = \mathrm{tr}(A^\top B)$. If $\mathcal{C}$ denotes the protected span, its solution is

$$\Delta^\star = -\frac{1}{L}\mathrm{Proj}_{\mathcal{C}^\perp}(G) = -\frac{1}{L}\big(G - \mathrm{Proj}_{\mathcal{C}}(G)\big).$$

Thus, Euclidean projection removes the component of G in the protected subspace; OSD will retain the same type of linear constraint while changing the steepest-descent geometry.

3.2 The Muon Optimizer and Spectral-Norm Geometry

Muon (Momentum Orthogonalized by Newton–Schulz) is designed for matrix-shaped neural-network parameters Jordan; Liu et al. (2025). Given a first-moment matrix $G = U\Sigma V^\top$, its idealized orthogonalized direction is the polar factor,

$$\Delta^\star = -\eta UV^\top. \tag{3}$$

Rather than computing an explicit SVD at every step, Muon approximates the polar factor using a small number of Newton–Schulz iterations. For exposition, a classical cubic iteration on a suitably normalized input X is

$$f(X) = \frac{3}{2}X - \frac{1}{2}XX^\top X, \qquad UV^\top \approx f^{\circ K}(X),$$

with K small in practice. Our implementation uses the same matrix-sign approximation in both the Muon-only and OSD arms, so the controlled comparison changes the protected-subspace constraint rather than the orthogonalization routine.

From an optimization perspective, Muon's update can be interpreted as solving, for each matrix block, a linear minimization problem over the spectral-norm unit ball (Bernstein & Newhouse; Bernstein (2025)). Given a gradient-like matrix $G \in \mathbb{R}^{m\times n}$, consider:

$$\min_{\Delta\in\mathbb{R}^{m\times n}} \ \langle G, \Delta \rangle \qquad \text{s.t.}\ \|\Delta\|_2 \le \eta. \tag{4}$$

where $\|\cdot\|_2$ is the induced *matrix 2-norm (spectral norm)*. One can show that the optimal solution is given by exactly (3). Under this lens, Muon can be understood as performing steepest descent under the induced 2-norm geometry.

4 Methodology

The preliminaries above reveal a natural way to combine projection-based continual learning with Muon-style optimization. Projection-based methods protect past-task knowledge by enforcing orthogonality to constraint directions, whereas Muon chooses matrix updates through a spectral-norm-constrained steepest descent step. We therefore formulate continual learning updates directly under spectral-norm geometry, imposing orthogonality constraints from past tasks while selecting the best current-task descent direction within the spectral-norm ball. This leads to a Muon-compatible analogue of classical OGD-style updates. In particular, we propose to formulate continual learning updates for matrix parameters as the solution of a spectral-norm–aware constrained optimization problem of the form

$$\begin{aligned} &\min_{\Delta\in\mathbb{R}^{m\times n}} \quad \langle G, \Delta \rangle_F \\ &\text{s.t.} \quad \|\Delta\|_2 \le \eta, \quad \langle C_i, \Delta \rangle_F = 0, \quad i = 1, \ldots, k, \end{aligned} \tag{5}$$

where G denotes the current task gradient (or momentum) and $\{C_i\}_{i=1}^k$ encode constraint directions associated with previously learned tasks (e.g., gradients on episodic memory or protected subspaces). This formulation selects a spectral steepest-descent direction while satisfying the same linear non-interference constraints used to protect past-task directions. The protected subspace provides the stability mechanism; the spectral trust region changes how plasticity is allocated within the remaining feasible directions.

The constrained problem has a convenient dual characterization even though the optimal multipliers do not generally admit a closed form.

Proposition 1 (Dual characterization of OSD). *Let $\eta > 0$ and define $H(\lambda) = G + \sum_{i=1}^{k} \lambda_i C_i$. Problem* (5) *satisfies strong duality, and its dual is*

$$\min_{\lambda \in \mathbb{R}^k} \|H(\lambda)\|_\star , \tag{6}$$

where $\|\cdot\|_\star$ is the nuclear norm. A primal optimum can be recovered as $\Delta^\star = -\eta S^\star$ for a compatible $S^\star \in \partial\|H(\lambda^\star)\|_\star$ satisfying the KKT equality conditions. When $H(\lambda^\star)$ has full active rank, $S^\star = UV^\top = \mathtt{msgn}(H(\lambda^\star))$ for an SVD $H = U\Sigma V^\top$.

Proof sketch. The Lagrangian is $\mathcal{L}(\Delta, \lambda) = \langle H(\lambda), \Delta\rangle_F$. The support function of the spectral-norm ball gives $\inf_{\|\Delta\|_2 \le \eta} \langle H, \Delta\rangle_F = -\eta\|H\|_\star$, yielding (6). Strong duality follows from Slater's condition because $\Delta = 0$ satisfies the equality constraints and lies strictly inside the spectral ball. The KKT conditions then identify a nuclear-norm subgradient compatible with the equality constraints; in the full-rank case this is the polar factor $UV^\top$.

The dual objective is convex but nonsmooth. Using a valid nuclear-norm subgradient $S(H) \in \partial\|H\|_\star$, its ith dual component is $\langle C_i, S(H)\rangle_F$. We therefore use warm-started subgradient iterations

$$\lambda_i^{(m+1)} = \lambda_i^{(m)} - \eta_\lambda \left\langle C_i, S\Big(H(\lambda^{(m)})\Big)\right\rangle_F , \tag{7}$$

and approximate $S(H)$ with the same Newton–Schulz matrix-sign operation used by Muon. In practice, we warm-start the dual variables from the previous optimizer step, take only $T_{\text{in}} \in \{1, 2\}$ dual corrections, and approximate the polar/nuclear-norm subgradient with Newton–Schulz iterations. The structured low-rank implementation used in our experiments is given below.

Remark 1 (Modularity). *The proposed framework is modular and can be viewed as a drop-in replacement for the update step in a wide class of projection-based continual learning methods. In particular, any method that specifies a collection of constraint directions $\{C_i\}$—whether derived from gradients, memory buffers, or structured subspaces—can be directly integrated into our formulation by replacing the standard Euclidean projection step with the spectral-norm-constrained update in* (5)*. This makes the update geometry modular with respect to the protected-subspace estimator: the same constraint construction can be paired with either Euclidean or spectral steepest-descent updates.*

4.1 Low-rank constraints from singular vectors

We instantiate (5) with the same singular-vector protected-subspace construction used by OSFT/S-culpting Subspaces Nayak et al. (2025), so that the principal methodological change is the feasible-update geometry rather than the protected-subspace estimator. Let $U \in \mathbb{R}^{m\times k}$ and $V \in \mathbb{R}^{n\times k}$ denote the top-k left and right singular vectors obtained from the protected model/task statistics. In our implementation these bases are computed once at the beginning of a continual-learning stage and held fixed within that stage, so the SVD cost is amortized rather than incurred at every optimizer step. In this setting, the constraints in (5) take the form

$$U^\top \Delta V = 0,$$

which enforces orthogonality of Δ to the bilinear subspace spanned by $\{u_a v_b^\top\}_{a,b=1}^{k}$. Equivalently, this corresponds to choosing $C_i = C_{ab} = u_a v_b^\top$ in (5), so that the general constraint $\langle C_i, \Delta\rangle = 0$ reduces to $\langle C_{ab}, \Delta\rangle = 0$ for all $(a, b) \in [k] \times [k]$. Under this parameterization, the dual variables can be organized into a matrix $\Lambda \in \mathbb{R}^{k\times k}$, yielding the structured correction

$$\textstyle\sum_{a,b} \Lambda_{ab} C_{ab} = U\Lambda V^\top.$$

Substituting this structure into (5) leads to a specialized instance of the OSD update, where the dual optimization is carried out over a k^2-dimensional space and all operations can be implemented using low-rank matrix products. The resulting algorithm (Algorithm 1) is thus a direct instantiation of the general framework with a structured choice of $\{C_i\}$, enabling more efficient computation while preserving the same constrained formulation.

In this paper, we focus on this specific instantiation for both practical and conceptual reasons. First, the resulting constraint subspace admits a structured low-rank representation, reducing the dimensionality of the dual problem and enabling efficient matrix–matrix implementations. Second, the formulation operates in the full-parameter fine-tuning regime, where updates are applied directly to matrix-valued

Algorithm 1: OSD with Low-Rank Constraints

Input: Current model parameters θ_0; protected task/model statistics; rank k; step size η; dual step size η_Λ; inner iterations T_{in}; mini-batch sampler $\mathcal{B}(\cdot)$.

Output: Continually trained parameters θ.

Step 0 (Precompute constraints). Obtain $U \in \mathbb{R}^{m\times k}$ and $V \in \mathbb{R}^{n\times k}$ as the top-k left/right singular vectors of the protected statistics for the current stage, then hold them fixed within the stage. Initialize $\Lambda_{-1} = \mathbf{0} \in \mathbb{R}^{k\times k}$ and $G_{-1} = 0$.

for $t = 0, \ldots, T-1$ **do**

Sample a mini-batch $\mathcal{S}_t \sim \mathcal{B}(\cdot)$ and compute momentum $G_t \leftarrow \beta G_{t-1} + \nabla_\theta \ell(\theta_t; \mathcal{S}_t)$.

Inner loop (dual descent).

Warm-start $\Lambda^{(0)} \leftarrow \Lambda_{t-1}$.

for $m = 0, \ldots, T_{\text{in}} - 1$ **do**

Form the shifted matrix $H^{(m)} \leftarrow G_t + U\,\Lambda^{(m)} V^\top$.

Compute approximate matrix sign via Newton–Schulz $S^{(m)} \leftarrow \texttt{NS5}(H^{(m)})$.

Dual update: $\Lambda^{(m+1)} \leftarrow \Lambda^{(m)} - \eta_\Lambda\, U^\top S^{(m)} V$.

Set $\Lambda_t \leftarrow \Lambda^{(T_{\text{in}})}$ and $H_t \leftarrow G_t + U\Lambda_t V^\top$.

Primal update (Muon-style step). $\Delta_t \leftarrow -\eta\,\texttt{NS5}(H_t), \quad \theta_{t+1} \leftarrow \theta_t + \Delta_t$.

weights and therefore align naturally with the matrix-centric geometry underlying Muon. In contrast, parameter-efficient fine-tuning (PEFT) methods such as LoRA restrict updates to low-rank adapter modules, extending it to these regimes would require reinterpreting the spectral-norm constraint under the induced low-rank parameterization. At the same time, as noted in Remark 1, the proposed framework is modular and accommodates arbitrary choices of constraint directions $\{C_i\}$. Exploring alternative constructions of $\{C_i\}$ within the spectral-norm formulation, as well as extending the method to LoRA and other PEFT settings, remains an interesting direction for future work.

5 Experimental Results

We evaluate OSD on both standard continual learning benchmarks and domain-specialized LLM adaptation settings. Our goal is to test whether the proposed spectral-norm-aware constrained update improves the stability–plasticity trade-off across different task lengths, domains, and model architectures. We compare against representative continual learning baselines, including recent protected-subspace methods such as O-LoRA and OSFT (Sculpting Subspaces). Unless otherwise specified, we follow standard task orders from prior work and report results averaged over multiple runs. Additional implementation details and hyperparameters are provided in Appendix C.

5.1 Benchmarks and Evaluation Protocol

We evaluate OSD on benchmarks with different task lengths, domain shifts, and model capabilities:

- **Standard Continual Learning Benchmark:** Following the task orders used in our experiments, the standard setting contains DBpedia, Amazon Reviews, Yahoo Answers, and AG News Zhang et al. (2015).
- **Extended Continual Learning Benchmark (15 Tasks):** This benchmark extends the standard setting to a longer task stream Razdaibiedina et al. (2023). It includes the four standard tasks above plus Yelp, four GLUE datasets (MNLI, QQP, RTE, SST-2) Wang et al. (2018), five SuperGLUE datasets (WiC, CB, COPA, MultiRC, BoolQ) Wang et al. (2019), and IMDB.
- **TRACE Benchmark:** TRACE evaluates continual learning for instruction-tuned LLMs across diverse skills, including multilingual understanding, domain knowledge, arithmetic reasoning, and coding Wang et al. (2023b).
- **Domain-Specific Sequential Learning (Coding, Math, Medical):** To test specialized-domain adaptation, we train sequentially on BigCodeBench Zhuo et al. (2024), GSM8K Cobbe et al. (2021), and HuatuoGPT-o1 Chen et al..

Models and Protocol. We evaluate both encoder-decoder and decoder-only architectures, including T5-Large, Qwen2.5 (1.5B, 3B, 7B), LLaMA2 (7B), LLaMA3.2 (1B, 3B). We follow standard continual learning protocols and average results over three independent runs with randomly permuted task sequences when applicable. The definition of average accuracy (AA) and backward transfer (BT) see C.2. Additional implementation details are provided in Appendices C and hyperparameter ablations are provided in Appendix G.

5.2 Baseline Methods

We compare OSD with representative methods and distinguish full-parameter controls from PEFT methods because they do not have the same trainable-parameter budget.

- **Full-parameter controls: SeqFT/SeqSFT** uses AdamW without anti-forgetting constraints. For the component analysis we additionally use **Muon-only** (spectral update, no protected-subspace constraint) and **Frob-OGD** (the same protected-subspace constraint with a Frobenius steepest-descent update). The Frobenius arm is rescaled layer-wise to match $\|\Delta W\|_F$ in the corresponding spectral arm.
- **Projection/PEFT methods:** SeqLoRA, IncLoRA, O-LoRA, and OSFT (*Sculpting Subspaces*) represent low-rank or protected-subspace continual adaptation.
- **Other CL baselines:** Replay, EWC Kirkpatrick et al. (2017), LwF, L2P Wang et al. (2022b), LFPT5, and ProgPrompt Razdaibiedina et al. (2023) cover replay-, regularization-, and prompt-based approaches.
- **Model-merging baselines:** SLERP Jang et al. (2024) sequentially interpolates task models, while TIES-Merging Yadav et al. (2023) trims task-vector updates, resolves sign conflicts, and merges sign-consistent components.
- **Offline/non-continual references:** PerTaskFT trains a separate model per task and MTL trains jointly on all tasks. We report them as references rather than literal upper bounds.

The PEFT comparisons reflect end-task performance across practical continual-learning strategies; the full-parameter Muon-only/Frob-OGD/OSD controls are the comparisons used to isolate update geometry and protected-subspace effects.

5.3 Main Result

Table 1: Comparison of Average Accuracy (%) on standard and 15-task continual learning benchmarks using T5-Large. Our OSD results are averaged over three seeds for each order. For each column among continual learning methods, the best performance is highlighted in **bold**, and the second-best is underlined.

Method	Standard CL Benchmark				15-Task CL Benchmark			
	Order-1	Order-2	Order-3	Avg	Order-4	Order-5	Order-6	Avg
SeqFT	18.9	24.9	41.7	28.5	7.4	7.4	7.5	7.4
SeqLoRA	44.6	32.7	53.7	43.7	2.3	0.6	1.9	1.6
IncLoRA	66.0	64.9	68.3	66.4	63.3	58.5	61.7	61.2
Replay	55.2	56.9	61.3	57.8	55.0	54.6	53.1	54.2
EWC	48.7	47.7	54.5	50.3	45.3	44.5	45.6	45.1
LwF	54.4	53.1	49.6	52.3	50.1	43.1	47.4	46.9
L2P	60.3	61.7	61.1	60.7	57.5	53.8	56.9	56.1
LFPT5	67.6	72.6	77.9	72.7	70.4	68.2	69.1	69.2
O-LoRA	75.4	75.7	76.3	75.8	**72.3**	64.8	71.6	69.6
OSFT	75.3	74.0	**78.4**	75.9	71.6	69.6	72.7	71.3
OSD (Ours)	**78.9**$_{\pm 0.8}$	**79.4**$_{\pm 0.7}$	78.3$_{\pm 1.0}$	**78.9**$_{\pm 1.0}$	72.2$_{\pm 0.8}$	**70.3**$_{\pm 0.7}$	**73.5**$_{\pm 0.4}$	**72.0**$_{\pm 1.5}$
SLERP Jang et al. (2024)	40.5	43.0	45.8	43.1	2.4	1.5	2.7	2.2
TIES Yadav et al. (2023)	35.0	38.5	37.8	37.1	7.8	7.1	5.8	6.9
ProgPrompt	75.2	75.0	75.1	75.1	78.0	77.7	77.9	77.9
PerTaskFT	70.0	70.0	70.0	70.0	78.1	78.1	78.1	78.1
MTL (Offline Ref.)	80.0	80.0	80.0	80.0	76.5	76.5	76.5	76.5

Table 1 compares OSD with representative continual learning baselines on the T5-Large continual learning benchmarks. We consider both the standard benchmark and the large 15-task benchmark,

following the same experimental protocol as O-LoRA. On the standard benchmark, OSD achieves the best average accuracy among the compared continual learning methods, reaching 78.9% and improving over O-LoRA (75.8%) and OSFT (75.9%). On the large benchmark, OSD remains highly competitive under a longer and more heterogeneous task stream, achieving 72.0% average accuracy and improving over O-LoRA (69.6%) and OSFT (71.3%). Across the six T5 task orders, OSD is consistently competitive with the strongest protected-subspace baselines, with the clearest gains on the standard four-task setting.

Table 2: TRACE benchmark performance with the LLaMA-2-7B-Chat backbone model.

Method	**LLaMA-2-7B-Chat**	
	AA (%)	**BT (%)**
SeqFT	23.0	-8.3
LoRASeqFT	9.2	-24.6
O-LoRA	41.3	-6.2
OSFT	48.4	-7.1
Ours	**49.4**$_{\pm 0.3}$	**-4.7**$_{\pm 1.2}$
PerTaskFT	57.6	N/A
MTL	52.3	N/A

Table 2 evaluates OSD on the TRACE benchmark using the decoder-only LLaMA-2-7B-Chat backbone. TRACE contains a diverse sequence of instruction-following tasks, making it substantially different from the T5-Large classification-style benchmarks. OSD achieves the best performance among all compared methods, with 49.4% AA and -4.7% BT representing the highest final average accuracy and lowest forgetting rate.

Table 3 further evaluates OSD on a domain-specific curriculum, Coding $\rightarrow$ Math $\rightarrow$ Medical, using Qwen2.5-1.5B-Instruct and Qwen2.5-3B-Instruct. This benchmark tests whether the method can preserve earlier domain skills while adapting to substantially different reasoning and output formats. OSD achieves the highest average accuracy at every stage for both model sizes. In particular, after the final Medical stage, OSD reaches 29.8% AA on Qwen2.5-1.5B-Instruct and 43.8% AA on Qwen2.5-3B-Instruct, outperforming Sculpting Subspaces and O-LoRA. These results show that OSD remains effective in heterogeneous domain-adaptation scenarios where tasks differ substantially in knowledge domain, reasoning style, and output format. We also observe that Math performance can increase after the Medical stage. Because the same qualitative effect appears in control optimizers, we treat this as an observed cross-domain improvement rather than OSD-specific positive backward transfer and rely on the full per-domain trajectory when assessing retention.

Overall, OSD performs consistently across encoder–decoder and decoder-only backbones. The component analysis below is used to separate the contribution of spectral update geometry from that of protected-subspace constraints, rather than attributing the main-table gains to either component in isolation.

5.4 Component and Update-Spectrum Analysis

To disentangle the effects of update geometry and protected-subspace constraints, we compare OSD with Muon-only, Frob-only, and Frob-OGD on the Standard CL Benchmark. The Frobenius variants use matched update energy, while Frob-OGD additionally applies the same protected-subspace mechanism used by OSD.

Table 4 shows that OSD achieves the best accuracy while remaining computationally efficient. OSD reaches $78.9 \pm 1.0\%$ AA, exceeding Frob-only ($76.44 \pm 1.03\%$), Frob-OGD ($74.84 \pm 2.44\%$), and Muon-only ($50.72 \pm 5.30\%$). Notably, OSD is also faster than both Frobenius variants, requiring 26.54 minutes compared with 30.59 minutes for Frob-only and 30.98 minutes for Frob-OGD. This yields approximately 13.2% and 14.3% lower training time, respectively, while achieving higher AA. All Frobenius baselines are tuned under the same computational resource setting as OSD; the corresponding learning-rate search and selected configurations are reported in Appendix G.1.

Table 3: Continual learning on Coding → Math → Medical. Values are test accuracy (%); AA averages observed tasks. Best results are **bold**, second-best are underlined.

Stage	Method	Qwen2.5-1.5B				Qwen2.5-3B			
		Code	Math	Med.	AA	Code	Math	Med.	AA
A	SeqSFT	17.1	–	–	17.1	31.1	–	–	31.1
	MuonSFT	17.4	–	–	17.4	32.8	–	–	32.8
	O-LoRA	17.9	–	–	17.9	32.7	–	–	32.7
	OSFT	17.1	–	–	17.1	32.7	–	–	32.7
	OSD	**18.4**$_{\pm 0.2}$	–	–	**18.4**$_{\pm 0.2}$	**32.9**$_{\pm 0.4}$	–	–	**32.9**$_{\pm 0.4}$
B	SeqSFT	14.1	46.5	–	30.3	18.2	60.6	–	39.4
	MuonSFT	17.1	52.0	–	34.6	18.7	61.3	–	40.0
	O-LoRA	17.6	51.8	–	34.7	18.7	**67.0**	–	42.9
	OSFT	13.8	47.0	–	30.4	20.8	62.6	–	41.7
	OSD	**17.8**$_{\pm 0.6}$	**52.5**$_{\pm 0.7}$	–	**35.2**$_{\pm 0.5}$	**29.7**$_{\pm 0.4}$	64.6$_{\pm 0.2}$	–	**47.2**$_{\pm 0.2}$
C	SeqSFT	**15.0**	46.4	22.0	27.8	19.1	62.8	28.0	36.6
	MuonSFT	14.7	46.6	20.2	27.1	18.7	62.5	28.3	36.5
	O-LoRA	14.8	46.4	20.8	27.3	16.2	63.0	25.2	34.8
	OSFT	14.8	47.4	22.6	28.3	32.1	63.0	27.6	40.9
	OSD	14.9$_{\pm 0.2}$	**51.0**$_{\pm 0.3}$	**23.4**$_{\pm 0.8}$	**29.8**$_{\pm 0.3}$	**32.9**$_{\pm 0.4}$	**69.8**$_{\pm 0.7}$	**28.8**$_{\pm 0.8}$	**43.8**$_{\pm 0.4}$

Table 4: Component ablation on the Standard CL Benchmark. AA is reported as mean±std (%), and training time is the average end-to-end training time in minutes.

Method	AA (%, mean±std)	Train Time (min)
Muon-only	50.72 ± 5.30	$\mathbf{25.12}$
Frob-only	76.44 ± 1.03	30.59
Frob-OGD	74.84 ± 2.44	30.98
OSD	$\mathbf{78.9 \pm 1.0}$	26.54

To further examine how spectral geometry changes the update, we analyze matched-energy updates during Stage B. At $\|\Delta W\|_F \approx 2.5 \times 10^{-4}$, the spectral arm has a $3.2\times$ smaller $\sigma_{\max}$ (4.1×10^{-5} versus 1.3×10^{-4}), while its stable rank increases from 4.4 to 40 and its effective rank from 366 to 553. These results suggest that spectral geometry redistributes comparable update energy across a broader set of singular directions, providing mechanistic evidence for a less concentrated update. Full spectrum statistics are reported in Appendix F.

6 Conclusion

We introduced OSD, a practical continual-learning instantiation of spectral steepest descent for LLM fine-tuning. OSD retains protected task-subspace constraints while replacing the Euclidean feasible update with a spectral-norm-bounded matrix update, implemented through a low-dimensional dual correction and Newton–Schulz matrix-sign approximation. Across T5, TRACE, and heterogeneous domain streams, OSD improves the balance between retaining previously acquired capabilities and adapting to new tasks. Controlled Muon-only and Frob-OGD comparisons indicate that neither spectral optimization nor protected-subspace constraints alone explain the full final result, while matched-energy spectrum measurements provide supporting evidence for a less concentrated update. These findings suggest that update geometry is a useful design dimension for protected-subspace continual learning, complementary to the choice of which task directions are preserved.

AI use statement

We used large language models as writing and coding assistants during the preparation of this manuscript. Specifically, LLMs were used to improve language clarity and presentation and to assist with coding and debugging for numerical experiments. All AI-assisted material was reviewed and verified by the authors. All technical ideas, theoretical results, experimental design, and final

interpretations were developed, verified, and approved by the authors, who take responsibility for the final content of this work.

Reproducibility statement

We provide the benchmark definitions, task sequences, model backbones, evaluation metrics, implementation details, optimization settings, hyperparameters, ablations, and detailed per-task results in the appendix. These details are intended to support reproduction of the experimental results reported in the main paper.

References

Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 139–154, 2018.

Jeremy Bernstein. Deriving muon, 2025. URL https://jeremybernste.in/writing/deriving-muon.

Jeremy Bernstein and Laker Newhouse. Modular duality in deep learning. In *Forty-second International Conference on Machine Learning*.

Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with a-gem. *arXiv preprint arXiv:1812.00420*, 2018.

Junying Chen, Zhenyang Cai, Ke Ji, Xidong Wang, Wanlong Liu, Rongsheng Wang, Jianye Hou, and Benyou Wang. Huatuogpt-o1, towards medical complex reasoning with llms, 2024. *URL https://arxiv. org/abs/2412.18925*.

Lizhang Chen, Jonathan Li, and Qiang Liu. Muon optimizes under spectral norm constraints. *arXiv preprint arXiv:2506.15054*, 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.

Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Aleš Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE transactions on pattern analysis and machine intelligence*, 44(7):3366–3385, 2021.

Mehrdad Farajtabar, Navid Azizan, Alex Mott, and Ang Li. Orthogonal gradient descent for continual learning. In *International conference on artificial intelligence and statistics*, pp. 3762–3773. PMLR, 2020a.

Mehrdad Farajtabar, Navid Azizan, Alex Mott, and Ang Li. Orthogonal gradient descent for continual learning. In Silvia Chiappa and Roberto Calandra (eds.), *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pp. 3762–3773. PMLR, 26–28 Aug 2020b. URL https://proceedings.mlr.press/v108/farajtabar20a.html.

Ian J Goodfellow, Mehdi Mirza, Da Xiao, Aaron Courville, and Yoshua Bengio. An empirical investigation of catastrophic forgetting in gradient-based neural networks. *arXiv preprint arXiv:1312.6211*, 2013.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Liang Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *Iclr*, 1(2):3, 2022.

Young Kyun Jang, Dat Huynh, Ashish Shah, Wen-Kai Chen, and Ser-Nam Lim. Spherical linear interpolation and text-anchoring for zero-shot composed image retrieval. In *European conference on computer vision*, pp. 239–254. Springer, 2024.

Keller Jordan. Muon: An optimizer for hidden layers in neural networks. URL https://kellerjordan.github.io/posts/muon/. Blog post.

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.

Alex Lewandowski, Michał Bortkiewicz, Saurabh Kumar, Andras Gyorgy, Dale Schuurmans, Mateusz Ostaszewski, and Marlos C Machado. Learning continually by spectral regularization. In *International Conference on Learning Representations*, volume 2025, pp. 34504–34530, 2025.

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

Jingyuan Liu, Jianlin Su, Xingcheng Yao, Zhejun Jiang, Guokun Lai, Yulun Du, Yidao Qin, Weixin Xu, Enzhe Lu, Junjie Yan, et al. Muon is scalable for llm training. *arXiv preprint arXiv:2502.16982*, 2025.

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

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Nikhil Shivakumar Nayak, Krishnateja Killamsetty, Ligong Han, Abhishek Bhandwaldar, Prateek Chanda, Kai Xu, Hao Wang, Aldo Pareja, Oleg Silkin, Mustafa Eyceoz, et al. Sculpting subspaces: Constrained full fine-tuning in llms for continual learning. *arXiv preprint arXiv:2504.07097*, 2025.

German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural networks*, 113:54–71, 2019.

Chengwei Qin and Shafiq Joty. Lfpt5: A unified framework for lifelong few-shot language learning based on prompt tuning of t5. *arXiv preprint arXiv:2110.07298*, 2021.

Anastasia Razdaibiedina, Yuning Mao, Rui Hou, Madian Khabsa, Mike Lewis, and Amjad Almahairi. Progressive prompts: Continual learning for language models. *arXiv preprint arXiv:2301.12314*, 2023.

Gobinda Saha, Isha Garg, and Kaushik Roy. Gradient projection memory for continual learning. *arXiv preprint arXiv:2103.09762*, 2021.

Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. *Advances in neural information processing systems*, 30, 2017.

Brian Tekmen, Jason Yin, and Qianqian Tong. Gem-style constraints for peft with dual gradient projection in lora. In *2025 IEEE International Conference on Data Mining Workshops (ICDMW)*, pp. 2736–2743. IEEE, 2025.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP workshop BlackboxNLP: Analyzing and interpreting neural networks for NLP*, pp. 353–355, 2018.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. Superglue: A stickier benchmark for general-purpose language understanding systems. *Advances in neural information processing systems*, 32, 2019.

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

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.

Zifeng Wang, Zizhao Zhang, Sayna Ebrahimi, Ruoxi Sun, Han Zhang, Chen-Yu Lee, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, et al. Dualprompt: Complementary prompting for rehearsal-free continual learning. In *European conference on computer vision*, pp. 631–648. Springer, 2022a.

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

Yifeng Xiong and Xiaohui Xie. Oplora: Orthogonal projection lora prevents catastrophic forgetting during parameter-efficient fine-tuning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 34088–34096, 2026.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin A Raffel, and Mohit Bansal. Ties-merging: Resolving interference when merging models. *Advances in neural information processing systems*, 36:7093–7115, 2023.

Jaehong Yoon, Eunho Yang, Jeongtae Lee, and Sung Ju Hwang. Lifelong learning with dynamically expandable networks. In *6th International Conference on Learning Representations, ICLR 2018*, 2018.

Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *International conference on machine learning*, pp. 3987–3995. Pmlr, 2017.

Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28, 2015.

Zhixin Zhang, Zeming Wei, and Meng Sun. Dynamic orthogonal continual fine-tuning for mitigating catastrophic forgettings, 2025. URL `https://arxiv.org/abs/2509.23893`.

Terry Yue Zhuo, Minh Chien Vu, Jenny Chim, Han Hu, Wenhao Yu, Ratnadira Widyasari, Imam Nur Bani Yusuf, Haolan Zhan, Junda He, Indraneil Paul, et al. Bigcodebench: Benchmarking code generation with diverse function calls and complex instructions. *arXiv preprint arXiv:2406.15877*, 2024.

A Related Work (Detailed)

Continual Learning in Large Language Models. Continual learning (CL) aims to enable models to acquire a sequence of new tasks while preserving performance on previously learned ones Parisi et al. (2019); De Lange et al. (2021); Wang et al. (2024). Classical CL methods are commonly grouped into regularization-based, replay-based, and architecture- or prompt-based approaches. Regularization-based methods, such as EWC Kirkpatrick et al. (2017), Synaptic Intelligence Zenke et al. (2017), MAS Aljundi et al. (2018), and Learning without Forgetting Li & Hoiem (2017), discourage updates that strongly perturb parameters or predictions important to earlier tasks. Replay-based methods, such as GEM and A-GEM Lopez-Paz & Ranzato (2017); Chaudhry et al. (2018), mitigate forgetting by storing or replaying examples from previous tasks, but this can introduce memory, privacy, or data-governance concerns when applied to large-scale language models. Prompt- and parameter-efficient methods, including L2P Wang et al. (2022b), DualPrompt Wang et al. (2022a), Progressive Prompts Razdaibiedina et al. (2023), LFPT5 Qin & Joty (2021), and LoRA-based adaptation Hu et al. (2022), reduce the number of trainable parameters by adapting small prompt vectors or low-rank modules while keeping most pretrained weights fixed.

Recent work has shown that CL for LLMs is especially challenging because sequential instruction tuning can degrade not only previous task accuracy but also general reasoning, instruction-following, and safety behavior. TRACE Wang et al. (2023b) provides a comprehensive benchmark for LLM continual learning, covering domain-specific tasks, multilingual ability, code generation, and mathematical reasoning in a unified evaluation format. Other language-model CL studies evaluate both

standard continual-learning task sequences and longer multi-task sequences to measure the stability-plasticity trade-off under more realistic task streams Razdaibiedina et al. (2023); Wang et al. (2023a); Nayak et al. (2025). These benchmarks motivate update mechanisms that can learn new task-specific information while explicitly controlling interference with previously acquired capabilities.

Projection-based Continual Learning. Projection-based CL methods address catastrophic forgetting by modifying the update direction rather than only adding scalar penalties to the loss. GEM Lopez-Paz & Ranzato (2017) and A-GEM Chaudhry et al. (2018) project or approximate-project gradients so that the loss on stored previous-task samples does not increase. Orthogonal Gradient Descent (OGD) Farajtabar et al. (2020a) instead projects the gradient for a new task onto a subspace that is approximately orthogonal to directions affecting previous-task outputs. Related orthogonal-subspace methods further enforce non-interference by constructing task-specific subspaces or decomposing representation spaces Saha et al. (2021); Wang et al. (2023a). Compared with replay-based approaches, these methods reduce or avoid dependence on storing previous training data, but maintaining and projecting against historical gradient or representation subspaces can become expensive for modern LLMs.

For language-model continual learning, O-LoRA adapts the projection principle to low-rank adaptation Wang et al. (2023a). Instead of updating the full model, O-LoRA assigns different tasks to low-rank subspaces and encourages these subspaces to remain mutually orthogonal. This design reduces task interference with only marginal additional parameters and avoids storing previous task data. However, because O-LoRA operates within low-rank adapters, its expressivity may be limited when full-model adaptation is needed. More recently, *Sculpting Subspaces* introduces Orthogonal Subspace Fine-Tuning (OSFT), a constrained full fine-tuning strategy for LLM continual learning Nayak et al. (2025). These methods use singular-value-based subspace analysis to identify task-relevant directions and constrain future updates away from critical directions associated with prior tasks. This line of work shows that geometric constraints and orthogonal update rules are effective for mitigating forgetting, but existing methods often rely on Euclidean projection or SVD-based subspace construction. In contrast, OSD keeps the protected-subspace mechanism but changes the geometry used to select feasible matrix updates.

Spectral-Norm-Aware Optimization and Muon. Standard LLM training and fine-tuning commonly use AdamW Loshchilov & Hutter (2017), which applies coordinate-wise adaptive scaling with decoupled weight decay. Muon, short for MomentUm Orthogonalized by Newton–Schulz, is a recent optimizer designed for matrix-valued neural-network parameters Jordan. Unlike AdamW, Muon first forms a momentum update and then approximately orthogonalizes the matrix update using Newton–Schulz iterations, producing update directions close to the polar factor of the momentum matrix. Recent empirical work shows that Muon can scale to large language model training and may improve training efficiency compared with AdamW in large-scale pretraining regimes Liu et al. (2025). From a theoretical perspective, modular duality and related analyses interpret Muon-style updates through operator-norm-aware geometry, while recent work further formalizes Muon as implicitly optimizing under spectral-norm constraints **?**Chen et al. (2025). These views support the idea that matrix-valued updates should not always be treated as flat Euclidean vectors.

Spectral quantities have also been used directly for continual learning. *Learning Continually by Spectral Regularization* regularizes the singular values of the model weights to preserve trainability **?**. OSD acts on a different object: it constrains the geometry of the update ΔW itself and couples that update to explicit past-task non-interference constraints rather than adding a spectral penalty on W.

Beyond pretraining, Muon-style orthogonalized updates have also motivated work on parameter-efficient adaptation. For example, recent LoRA-oriented extensions formulate low-rank adaptation from a geometric or Riemannian perspective and use Muon-inspired orthogonalization to reduce parametrization imbalance in low-rank factors. These developments suggest that orthogonalized matrix updates are useful not only for optimization speed but also for controlling the geometry of learned parameter changes. However, existing Muon work primarily focuses on pretraining efficiency, optimization theory, or low-rank adaptation, rather than continual learning. Our work connects Muon-style orthogonalized updates with projection-based continual learning: by applying operator-norm-aware updates under orthogonal subspace constraints, we aim to reduce catastrophic forgetting while retaining the plasticity needed for sequential LLM fine-tuning.

B Limitations and Future Work

Our study focuses on representative continual-learning settings rather than exhaustive deployment streams. OSD is currently a full-parameter method for selected matrix-valued weights, so comparisons with PEFT methods such as O-LoRA do not control the number of trainable parameters; we therefore interpret them as practical end-task comparisons rather than equal-budget optimizer comparisons. In the decoder-only domain experiments, OSD targets two-dimensional MLP projection matrices while non-targeted parameters are updated with AdamW. Extending the spectral geometry to LoRA factors or arbitrary higher-order tensors requires choosing an induced weight-space or tensor-native norm and is left to future work. Finally, our mechanism analysis provides empirical support rather than a theorem that spectral geometry improves retention in every setting; longer task streams and convergence analysis remain open directions.

C Numerical Details

C.1 Task Sequences and Model Backbones

We evaluate OSD under four continual supervised fine-tuning protocols: the standard continual learning benchmark, the 15-task long-sequence benchmark, the TRACE aligned-LLM benchmark, and our domain-specific curriculum. The full task orders are summarized in Table 5.

For Orders 1–6, we use T5-Large as the encoder–decoder backbone. For TRACE, we use LLaMA-2-7B-Chat as the decoder-only aligned LLM backbone. For the domain-specific curriculum, we evaluate decoder-only instruction models, including Qwen2.5-Instruct and LLaMA3.2-Instruct families. Unless otherwise specified, each task or stage is initialized from the checkpoint produced by the previous task or stage.

Table 5: Task orders used in the continual learning benchmarks. Orders 1–3 correspond to the standard benchmark, Orders 4–6 correspond to the 15-task long-sequence benchmark, TRACE corresponds to the aligned-LLM continual learning benchmark, and Domain-specific corresponds to our Coding–Math–Medical curriculum.

Order / Protocol	Task sequence
Order-1	DBpedia → Amazon → Yahoo → AGNews
Order-2	DBpedia → Amazon → AGNews → Yahoo
Order-3	Yahoo → Amazon → AGNews → DBpedia
Order-4	MNLI → CB → WiC → COPA → QQP → BoolQA → RTE → IMDB → Yelp → Amazon → SST-2 → DBpedia → AGNews → MultiRC → Yahoo
Order-5	MultiRC → BoolQA → WiC → MNLI → CB → COPA → QQP → RTE → IMDB → SST-2 → DBpedia → AGNews → Yelp → Amazon → Yahoo
Order-6	Yelp → Amazon → MNLI → CB → COPA → QQP → RTE → IMDB → SST-2 → DBpedia → AGNews → Yahoo → MultiRC → BoolQA → WiC
TRACE	C-STANCE → FOMC → MeetingBank → Py150 → ScienceQA → NumGLUE-cm → NumGLUE-ds → 20Minuten
Domain-specific	Stage A: Coding (BigCodeBench) → Stage B: Math (GSM8K) → Stage C: Medical (HuatuoGPT-o1)

C.2 Metrics.

Average Accuracy (AA). Our primary metric for evaluating overall performance is Average Accuracy (AA). This metric captures the model's final proficiency across all tasks once the entire continual learning sequence is complete. Formally, let $a_{T,i}$ denote the test accuracy on task i after the model has finished training on the final task T. The Average Accuracy is defined as:

$$\mathrm{AA} = \frac{1}{T}\sum_{i=1}^{T} a_{T,i}. \tag{8}$$

Backward Transfer (BT). To explicitly quantify the extent of catastrophic forgetting, we report Backward Transfer (BT). This metric measures the influence that learning subsequent tasks has on

the retention of previously acquired skills. Let $a_{i,i}$ denote the test accuracy on task i immediately after it is initially learned. The Backward Transfer after training on all T tasks is calculated as:

$$\mathrm{BT} = \frac{1}{T-1} \sum_{i=1}^{T-1} (a_{T,i} - a_{i,i}). \tag{9}$$

A negative BT indicates that the model has forgotten prior knowledge during the sequential training stages. Conversely, a BT close to zero, or a positive value, implies that prior capabilities have been successfully protected or even positively reinforced.

C.3 Dataset Details

We evaluate OSD on both established continual-learning benchmarks and a domain-specific sequential instruction-tuning curriculum. The established benchmarks include the standard 15-Task Continual Learning Benchmark and TRACE, which measure long-horizon retention under heterogeneous task streams. In addition, we construct a three-stage domain sequence consisting of coding, mathematical reasoning, and medical reasoning to evaluate catastrophic forgetting in practical LLM adaptation scenarios.

Standard and 15-Task Continual Learning Benchmarks. The standard continual learning benchmark consists of text classification tasks drawn from DBpedia, Amazon Reviews, Yahoo Answers, and AG News. The 15-task long-sequence benchmark extends this setting to a more diverse task stream. Following prior work Razdaibiedina et al. (2023), it includes five text classification datasets, AG News, Amazon Reviews, Yelp Reviews, DBpedia, and Yahoo Answers; four GLUE datasets, MNLI, QQP, RTE, and SST-2 Wang et al. (2018); five SuperGLUE datasets, WiC, CB, COPA, MultiRC, and BoolQ Wang et al. (2019); and the IMDB sentiment classification dataset. These tasks cover topic classification, sentiment analysis, natural language inference, paraphrase detection, semantic similarity, commonsense reasoning, reading comprehension, and yes/no question answering.

These benchmarks are useful for continual learning because they evaluate both short-term adaptation and long-term retention over heterogeneous task sequences. In our experiments, the model learns each task sequentially without access to future-task data. After training on the full task stream, we report AA across all tasks and BT to quantify forgetting on previously learned tasks.

TRACE Benchmark. TRACE is a continual-learning benchmark designed for instruction-tuned large language models Wang et al. (2023b). Unlike traditional classification-focused continual-learning benchmarks, TRACE converts diverse datasets into a unified instruction-following format and evaluates whether an LLM can retain general capabilities while sequentially adapting to new tasks. The benchmark contains eight tasks spanning domain-specific knowledge, multilingual understanding, arithmetic reasoning, code generation, and general instruction-following ability.

TRACE provides a more realistic evaluation setting for LLM continual learning because sequential instruction tuning may degrade broad model capabilities, not only classification accuracy. For example, adapting the model to a new domain may harm its previous ability to solve mathematical problems, generate code, or follow multilingual instructions. Therefore, TRACE directly tests the stability-plasticity trade-off in aligned LLMs. In our experiments, we follow the TRACE evaluation protocol and report AA and BT using the LLaMA-2-7B-Chat backbone. We compare OSD with sequential fine-tuning and representative orthogonal-gradient baselines, including O-LoRA and Sculpting Subspaces.

Coding Dataset: BigCodeBench. For the coding stage in our domain-specific continual-learning sequence, we use BigCodeBench Zhuo et al. (2024). BigCodeBench is a code-generation benchmark designed to evaluate practical programming ability beyond simple function-completion problems. Compared with earlier coding benchmarks such as HumanEval, BigCodeBench contains more realistic programming tasks that require understanding natural-language specifications, using external Python libraries, composing multiple operations, and generating executable code.

In our experiments, BigCodeBench is used as the first stage of the coding–mathematics–medical continual-learning sequence. We use the instruction-style setting for supervised fine-tuning and evaluation. To construct the training split, we deterministically sample 800 coding problems for

supervised fine-tuning and reserve the remaining problems for held-out evaluation. Coding performance is evaluated using pass@1 under the official BigCodeBench execution-based protocol, where generated programs are executed against hidden unit tests. This stage tests whether the model can acquire executable code-generation ability and whether this ability is retained after later mathematical and medical adaptation.

Mathematics Dataset: GSM8K. For the mathematical reasoning stage, we use GSM8K Cobbe et al. (2021), a benchmark of grade-school math word problems. Each problem requires multi-step arithmetic reasoning and produces a final numerical answer. GSM8K is widely used for evaluating reasoning ability in language models because correct answers usually require decomposing a natural-language problem into intermediate reasoning steps rather than relying on surface-level pattern matching.

In our continual-learning sequence, GSM8K is used after the coding stage. The model is trained on the GSM8K training split, with optional subsampling controlled by the experimental configuration. Evaluation is performed on the official test split. We use exact-match numerical accuracy as the main metric after extracting the final answer from the model response. This stage evaluates whether the model can learn structured mathematical reasoning while preserving coding ability acquired in the previous stage.

Medical Reasoning Dataset: HuatuoGPT-o1. For the medical reasoning stage, we use HuatuoGPT-o1 Chen et al.. HuatuoGPT-o1 provides medical reasoning data for supervised fine-tuning and open-ended verifiable medical questions for evaluation. This dataset is substantially different from the coding and mathematics datasets because it requires domain-specific biomedical knowledge, careful reasoning, and medically appropriate answer generation.

In training, we use the supervised fine-tuning data from `FreedomIntelligence/medical-o1-reasoning-SFT`. Our preprocessing pipeline applies an English-only filter and uses answer-span supervision, so that the model is optimized on the final medically relevant answer. For evaluation, we use the verifiable-problem set from `FreedomIntelligence/medical-o1-verifiable-problem` and sample 1,000 English questions. Since many medical answers are open-ended and cannot be reliably evaluated by exact string matching, we report verifier-based `judge_accuracy` using a dedicated medical verifier model. This final stage is intentionally knowledge-intensive and serves as a strong test of catastrophic forgetting: after adapting to medical reasoning, the model is re-evaluated on the earlier coding and mathematics tasks to measure retention.

Three-Stage Domain Continual-Learning Sequence. In addition to the standard 15-Task and TRACE benchmarks, we evaluate OSD on a practical three-stage sequence: Coding $\rightarrow$ Mathematics $\rightarrow$ Medical. This sequence is designed to reflect realistic LLM adaptation, where a model may be sequentially specialized to different domains with very different output formats and reasoning requirements. The coding stage emphasizes executable program synthesis, the mathematics stage emphasizes symbolic and arithmetic reasoning, and the medical stage emphasizes knowledge-intensive open-ended reasoning. This setting allows us to evaluate both plasticity on the current domain and retention of previous capabilities after each stage.

D Standard and TRACE Benchmark Implementation

For the standard and TRACE continual learning experiments, including the standard short-sequence benchmark, the 15-task long-sequence benchmark, and the TRACE benchmark, we replicate the task sequences, prompts, and dataset configurations established in the original works (Wang et al., 2023a;b).For the standard continual learning benchmarks, we use T5-Large as the encoder–decoder backbone and follow the O-LoRA benchmark protocol. Orders 1–3 correspond to the standard benchmark, while Orders 4–6 correspond to the 15-task long-sequence benchmark. For the TRACE benchmark, we use LLaMA-2-7B-Chat as the decoder-only backbone.

Computational Resources. All OSD runs for T5-Large are executed on a single H100 80GB GPU using standard PyTorch training with bf16 precision and gradient checkpointing. We use a per-device batch size of 8 with gradient accumulation of 8, resulting in an effective batch size of 64. All TRACE

experiments are conducted on a server equipped with 8 NVIDIA H100 GPUs using DeepSpeed ZeRO Stage 2 with gradient checkpointing enabled. We use a per-GPU micro-batch size of 1 with gradient accumulation steps of 1, resulting in an effective training batch size of 8.

Unless otherwise specified, the main reported metric is AA, computed by evaluating all tasks in the stream after training on the final task. For both T5 and TRACE benchmarks, we report the mean over three independent random seeds unless otherwise noted. For T5 evaluation, we use a maximum source length of 512 and a maximum generation length of 50. For TRACE evaluation, we use a maximum prompt length of 1024 and a maximum answer length of 512. The full task sequences are listed in Table 5, and the main training hyperparameters are summarized in Table 6.

Table 6: Final hyperparameter choices for OSD on the Standard and TRACE continual learning benchmarks.

Hyperparameter	**Standard Benchmark**	**15-Task Benchmark**	**TRACE**
LR	1×10^{-3}	1×10^{-3}	1×10^{-5}
MUON_LR	5×10^{-3}	5×10^{-3}	1×10^{-5}
MUON_T	5	5	5
MUON_ETA_DUAL	1×10^{-4}	3×10^{-4}	1×10^{-5}
Max protected rank	64	48	32
SVD_NITER	4	4	4

E Domain-Specific Benchmark Implementation

E.1 Training Details

For the coding domain, we use the fixed 800-problem training subset of BigCodeBench constructed as described in Appendix C, with the remaining problems reserved for held-out evaluation. For mathematical reasoning, the model is trained on the GSM8K training set. Depending on the experimental setting, this data is optionally subsampled, bounded by the `num_train_examples` parameter. For the medical domain, we train on the HuatuoGPT-o1 SFT set. Our training pipeline applies an English-only filter, yielding approximately 20,000 training examples. We enforce answer-span supervision and explicitly remove any examples lacking supervised tokens to prevent non-finite loss issues during optimization.

E.2 Evaluation Details

We assess model plasticity and retention using domain-specific protocols and metrics. **Coding:** We adhere to the official BigCodeBench remote evaluation protocol. Specifically, we generate a single model completion per task (the pass@1 setting) and format the predictions into a JSON file. These are submitted to the official evaluator endpoint for execution-based scoring, and we report the resulting pass@1 accuracy. **Mathematics:** Performance on GSM8K is evaluated on the official `test` split, with the primary metric being exact-match numeric accuracy. **Medical Reasoning:** From the HuatuoGPT-o1 verifiable problem set, we randomly sample 1,000 questions for evaluation. Since medical responses are open-ended, we evaluate correctness using the pretrained (`medical_o1_verifier_3B`) verifier. The verifier takes the generated model response and reference answer as input and outputs a binary correctness prediction. We report the fraction of responses classified as correct as `judge_accuracy`.

E.3 Optimization and Hyperparameters

Stage A (coding script defaults). `max_length=2048`, `batch_size=4`, `grad_accum=4`, `epochs=3`, `lr=5e-6`, `weight_decay=0.01`, `warmup_ratio=0.03`, `max_steps=500`, `max_grad_norm=1.0`, seed 42.

```
You are a helpful math assistant.  Provide the final numeric answer
on the LAST line in the exact form:  #### <number>
Solve the following problem.  Output the final numeric answer on a
new line as:  #### <number>
[Math Question]
{question}
```

Figure 2: Prompt template used for zero-shot mathematical evaluation on the GSM8K dataset.

```
You are an expert Python programmer.  Please solve the following
task.
You may briefly explain your logic, but you MUST provide your final
solution inside a single Markdown code block (“‘python ...  “‘).
Here is the task:
{task_prompt}
```

Figure 3: Prompt template used for zero-shot coding evaluation on the BigCodeBench dataset.

```
You are an expert medical evaluator assessing whether a model’s
response correctly answers a medical question.  Your task is to
compare the model’s response to the reference answer and determine
if the model’s response is:
1.  CORRECT: The response contains the key medical information
from the reference answer, even if phrased differently or includes
additional correct medical details.
2.  INCORRECT: The response is medically wrong, misses the main
point, or provides incorrect medical information.
Focus on medical accuracy and completeness, not on writing style or
verbosity.
[Medical Question]
{question}
[Reference Answer]
{reference_answer}
[Model Response]
{model_response}
Evaluate the model’s response.  Output ONLY one of:  "CORRECT" or
"INCORRECT".
```

Figure 4: Input template used by the medical$_o1_verifier_3B model for verifier-based evaluation on the HuatuoGPT-o1 medical dataset.$

Stage B (math script defaults). `max_length=512`, `batch_size=1`, `grad_accum=8`, `epochs=3`, `lr=1e-5`, `weight_decay=0.01`, `warmup_ratio=0.03`, `max_steps=1800`, `max_grad_norm=1.0`, seed 42.

Stage C (medical script defaults). Script defaults are `max_length=1536`, `batch_size=1`, `grad_accum=8`, `epochs=1`, `lr=2e-5`, `max_steps=1200`. For stronger adaptation, we use extended budget runs (e.g., `epochs=3`, `max_steps=-1`, tuned warmup/LR).

OSD settings. For the decoder-only domain experiments, Algorithm 1 is applied to the two-dimensional MLP projection weights (`gate_proj`, `up_proj`, and `down_proj`); non-targeted parameters are updated with AdamW. Key OSD knobs are `muon_k`, `muon_T`, `muon_eta`, `muon_eta_dual`, the matrix-sign method (`ns` or `svd`), and optional warm-starting of dual variables.

E.4 Extended Main Results

In our extended numerical study, we evaluate the generalizability of OSD across additional architectures and scales, specifically examining the Qwen2.5-7B, LLaMA3.2-1B-instruct, and LLaMA3.2-3B-instruct models on the domain-specific Coding $\rightarrow$ Math $\rightarrow$ Medical sequence. The results, detailed in Tables 7, 8, and 9, confirm that the advantages of spectral-norm-aware orthogonal projection extend reliably across different LLM backbones.

Performance on the LLaMA3.2 Family. Tables 8 and 9 demonstrate that OSD consistently outperforms the baselines on both the 1B and 3B LLaMA3.2 architectures. For the 1B model, OSD achieves the highest Average Accuracy (AA) at every stage, culminating in a final Stage C AA of 26.2%, outperforming SeqSFT (24.2%) and Sculpting Subspaces (23.9%). The advantage is even more pronounced in the 3B model, where OSD sweeps all stages. Notably, OSD exhibits exceptional knowledge retention: after training on the final Medical domain, the 3B model retains a Math accuracy of 55.1%, suffering minimal catastrophic forgetting compared to its peak Stage B Math score of 56.0%. In contrast, baselines like O-LoRA and SeqSFT experience sharper drops in previously acquired skills.

Scaling to Qwen2.5-7B. Table 7 extends our evaluation to the larger Qwen2.5-7B-Instruct model, incorporating Gradient Replay to further anchor prior knowledge. While O-LoRA shows strong competitive performance at Stage B (edging out OSD in AA by 0.8%), OSD dominates the final curriculum evaluation at Stage C. At the conclusion of the three-stage sequence, OSD achieves the highest accuracy across all three individual domains—Coding (38.1%), Math (75.2%), and Medical (35.6%)—yielding the highest overall AA of 49.6%.

Overall, these extended results reinforce our primary findings. Integrating the Muon optimizer's operator-norm geometry with projection-based constraints provides a highly robust mechanism for balancing plasticity on new domains while safeguarding previously acquired reasoning capabilities.

Computational Resources All experiments are conducted on a single NVIDIA GH200 Grace–Hopper Superchip (96 GB HBM3, 900 W TDP) per training run. Multi-run repeats and the four optimizer conditions (AdamW, O-LoRA, Sculpting-Subspace, OSD) are scheduled in parallel across separate GH200 nodes when available; each individual run is single-GPU and does not require model- or pipeline-parallelism. CPU-side data loading uses the node's Grace CPU with 64-bit unified memory.

F Controlled Component, Spectrum, and Cost Analysis

Controlled geometry/protection comparison. Table 4 reports the main-paper comparison using the same Coding $\rightarrow$ Math $\rightarrow$ Medical curriculum. The Frobenius arm is rescaled layer-wise to match $\|\Delta W\|_F$ in the corresponding spectral arm, controlling for total update energy.

Matched-energy update spectrum. Table 10 summarizes the Stage B update-spectrum statistics on the targeted MLP layers. At matched Frobenius energy, the spectral arm has a substantially smaller

Table 7: Continual learning performance of the Qwen2.5-7B-instruct architecture evaluated across a sequential three-stage curriculum (Coding → Math → Medical). All metrics represent testing accuracy percentages (%). For each metric within a specific evaluation stage, the best performance is highlighted in **bold**, and the second-best is underlined.

Stage	Optimizer	Coding	Math	Medical	AA
A (Coding)	SeqSFT	37.8	-	-	37.8
	O-LoRA	37.9	-	-	37.9
	OSFT	38.8	-	-	38.8
	OSD	**40.8**	-	-	**40.8**
B (Math)	SeqSFT	**38.8**	72.0	-	55.4
	O-LoRA	**38.8**	**75.0**	-	**56.9**
	OSFT	38.6	72.4	-	55.5
	OSD	38.2	74.0	-	56.1
C (Medical)	SeqSFT	36.1	71.0	34.4	47.2
	O-LoRA	37.5	74.0	35.0	48.8
	OSFT	37.9	73.2	35.2	48.8
	OSD	**38.1**	**75.2**	**35.6**	**49.6**

Table 8: Comparative performance of LLaMA3.2-1B-instruct across sequential training stages (Coding → Math → Medical). Average Accuracy (AA) measures the mean performance across all three tasks at each stage. Bold values indicate the best performance, while underlined values denote the runner-up.

Stage	Optimizer	Coding	Math	Medical	AA
A (Coding)	SeqSFT	16.7	-	-	16.7
	O-LoRA	14.7	-	-	14.7
	OSFT	18.7	-	-	18.7
	OSD	**20.8**$_{\pm 1.1}$	-	-	**20.8**$_{\pm 1.1}$
B (Math)	SeqSFT	**21.7**	34.6	-	28.2
	O-LoRA	18.7	**35.8**	-	27.3
	OSFT	20.7	35.6	-	28.2
	OSD	**21.7**$_{\pm 1.2}$	35.6$_{\pm 1.4}$	-	**28.7**$_{\pm 0.9}$
C (Medical)	SeqSFT	16.7	33.8	22.2	24.2
	O-LoRA	11.7	31.6	22.0	21.8
	OSFT	17.6	32.2	22.0	23.9
	OSD	**19.7**$_{\pm 0.8}$	**35.5**$_{\pm 1.4}$	**23.5**$_{\pm 0.7}$	**26.2**$_{\pm 0.6}$

Table 9: Continual learning performance of the LLaMA3.2-3B-instruct architecture evaluated across a sequential three-stage curriculum (Coding → Math → Medical). All metrics represent testing accuracy percentages (%). For each metric within a specific evaluation stage, the best performance is highlighted in **bold**, and the second-best is underlined.

Stage	Optimizer	Coding (800 Q)	Math (GSM8K)	Medical (Huatuo)	AA
A (Coding)	SeqSFT	24.4	-	-	24.4
	O-LoRA	25.8	-	-	25.8
	OSFT	24.7	-	-	24.7
	OSD	**28.3**$_{\pm 1.6}$	-	-	**28.3**$_{\pm 1.6}$
B (Math)	SeqSFT	20.0	53.0	-	36.5
	O-LoRA	21.4	**55.0**	-	38.2
	OSFT	23.2	**55.0**	-	39.1
	OSD	**26.4**$_{\pm 0.8}$	**56.0**$_{\pm 1.9}$	-	**41.2**$_{\pm 1.0}$
C (Medical)	SeqSFT	17.0	50.0	31.8	32.9
	O-LoRA	18.2	52.4	31.8	34.1
	OSFT	20.1	53.8	31.8	35.2
	OSD	**25.1**$_{\pm 0.5}$	**55.1**$_{\pm 1.5}$	**32.9**$_{\pm 1.2}$	**37.7**$_{\pm 0.7}$

largest singular value together with markedly larger effective and stable ranks. These measurements support the interpretation that spectral geometry redistributes the update budget across more singular directions rather than merely reducing its total magnitude.

Table 10: Update-spectrum statistics of ΔW on targeted MLP layers during Stage B. Frobenius energy $\|\Delta W\|_F$ is matched by construction between the spectral and Frobenius arms.

Geometry	$\sigma_{\max}$	**Eff. rank**	**Stable rank**	$\|\Delta W\|_F$
Spectral arm	$\sim 4.1 \times 10^{-5}$	~ 553	~ 40	$\sim 2.5 \times 10^{-4}$
Frobenius arm (matched energy)	$\sim 1.3 \times 10^{-4}$	~ 366	~ 4.4	$\sim 2.5 \times 10^{-4}$

G Hyperparameter Sensitivity

To characterize sensitivity to the core OSD hyperparameters, we vary optimization and protected-subspace settings while keeping the remaining configuration fixed. Because our evaluation spans diverse domains, we analyze these factors across two distinct settings: (1) the standard and 15-task NLP benchmarks using the encoder-decoder T5-Large, and (2) the highly specialized Coding $\rightarrow$ Math $\rightarrow$ Medical sequence using the decoder-only Qwen2.5-1.5B-instruct architecture. Unless otherwise stated, the primary evaluation metric is the final Average Accuracy (AA).

G.1 Learning Rate Allocation and Plasticity

OSD relies on a dual-branch update mechanism, necessitating a careful balance between the matrix-level plasticity (controlled by the Muon step size) and the scalar/vector updates (controlled by the AdamW base learning rate).

Muon / Primal Step Size (`MUON_LR` / `muon_eta`). The Muon step size directly controls matrix-level plasticity for the targeted 2D parameters. Table 11 shows that increasing `MUON_LR` substantially improves performance on the T5 standard benchmark, raising the average accuracy from 74.3% to 77.7%. On the 15-task benchmark, 5×10^{-3} emerges as the optimal value. Conversely, for the Qwen2.5 architecture (Figure 5), the scaling factor for the orthogonalized momentum (`muon_eta`) exhibits a slow degradation if overly aggressive. A conservative value of 10^{-6} provides the highest stability for LLMs, ensuring unconstrained updates do not distort the established geometry.

Table 11: Effect of Muon step size. For the standard benchmark, we report the average over Orders 1–3. For the large benchmark, we report Large Order-4. All other hyperparameters are fixed within each block.

Setting	**`MUON_LR`**	**AA (%)**
Standard benchmark		
Standard	1×10^{-3}	74.3
Standard	2×10^{-3}	76.5
Standard	5×10^{-3}	**77.7**
15-task benchmark, Order-4		
Large	4×10^{-3}	67.8
Large	5×10^{-3}	**72.2**
Large	6×10^{-3}	70.5

AdamW Base Learning Rate (`LR` / `lr`). Reducing the AdamW branch learning rate is not a viable substitute for explicit subspace protection. As shown in Table 13, suppressing `LR` to 3×10^{-5} on T5 causes severe underfitting on the current task (e.g., Amazon collapses). A similar high sensitivity is observed in Qwen2.5 (Figure 5), where AA monotonically increases as `lr` approaches 5×10^{-5}. This confirms that OSD requires sufficient baseline plasticity combined with strictly enforced orthogonal corrections.

Table 12: Learning-rate sensitivity of the Frobenius baselines on the Standard CL Benchmark (Order 1, Seed 1; development AA). Best result for each method is shown in **bold**.

Method	Matrix LR	Dev. AA (%)
Frob-only	3×10^{-5}	**76.59**
	1×10^{-4}	75.85
	3×10^{-4}	74.91
	1×10^{-3}	34.94
Frob-OGD	3×10^{-4}	**77.20**
	1×10^{-3}	39.63
	3×10^{-3}	23.82
	5×10^{-3}	6.19

Learning-Rate Sensitivity of Frobenius Baselines We further examine the sensitivity of the Frobenius-geometry controls to the matrix learning rate on the Standard CL Benchmark. Table 12 reports development-set AA for Order 1, Seed 1. Both Frob-only and Frob-OGD are highly sensitive to the learning rate: overly large step sizes lead to substantial degradation, whereas smaller learning rates yield considerably stronger performance. Frob-only performs best at 3×10^{-5} with 76.59% AA, while Frob-OGD achieves its best result of 77.20% at 3×10^{-4}. We therefore use the tuned learning rates for the corresponding Frobenius baselines in the main component analysis.

Table 13: Effect of the AdamW-branch learning rate on the standard benchmark (`MUON_LR`= 1×10^{-3}).

LR	AA (%)	Observation
3×10^{-5}	62.4	Too small; Amazon collapses to near-random performance.
3×10^{-4}	71.0	Better, but still below the default setting.
1×10^{-3}	**75.7**	Sufficient current-task plasticity.
2×10^{-3}	74.5	Increasing LR further does not improve performance.

G.2 Dual Correction Strength (`MUON_ETA_DUAL`)

The dual correction dictates how aggressively the optimizer enforces the protected-subspace constraint ($U^\top \Delta V \approx 0$). Table 14 demonstrates that this parameter has a strict sweet spot for long task sequences. On the 15-task T5 benchmark, a weak step (1×10^{-4}) fails to protect old-task directions, while a strong step (5×10^{-4}) over-constrains new-task plasticity. The intermediate 3×10^{-4} performs best. This behavior is mirrored perfectly in the Qwen2.5 experiments (Figure 5), where a distinct optimal peak is observed at 5×10^{-5}, representing the critical threshold for balancing constraint satisfaction with model plasticity.

Table 14: Effect of dual correction strength on the 15-task benchmark (Large Order-4).

MUON_ETA_DUAL	AA (%)
1×10^{-4}	67.3
3×10^{-4}	**72.2**
5×10^{-4}	66.5

G.3 Protected Subspace Capacity and Rank (`MUON_K`)

Table 15 and Figure 5 explore the allocation of protected rank (k). Both architectures reveal that protected-rank allocation is highly sensitive but non-monotonic. For T5, allocating a moderate protected rank ($0.08/\max 48$) yields the best trade-off, indicating the subspace must be large enough to retain structure but compact enough to permit future learning. For Qwen2.5, performance remains remarkably robust across a wide range ($k = 1$ to 20), with a localized optimal peak at $k = 3$. This

suggests that for highly specialized reasoning tasks, critical prior knowledge is heavily concentrated in a top-dominant, compact singular subspace.

Table 15: Exploratory analysis of protected-rank allocation on T5 Large Order-4.

Protected rank / capacity	**AA (%)**
0.10/max64/total768	68.1
0.05/max32/total1024	64.6
0.08/max48/total1024	**72.2**

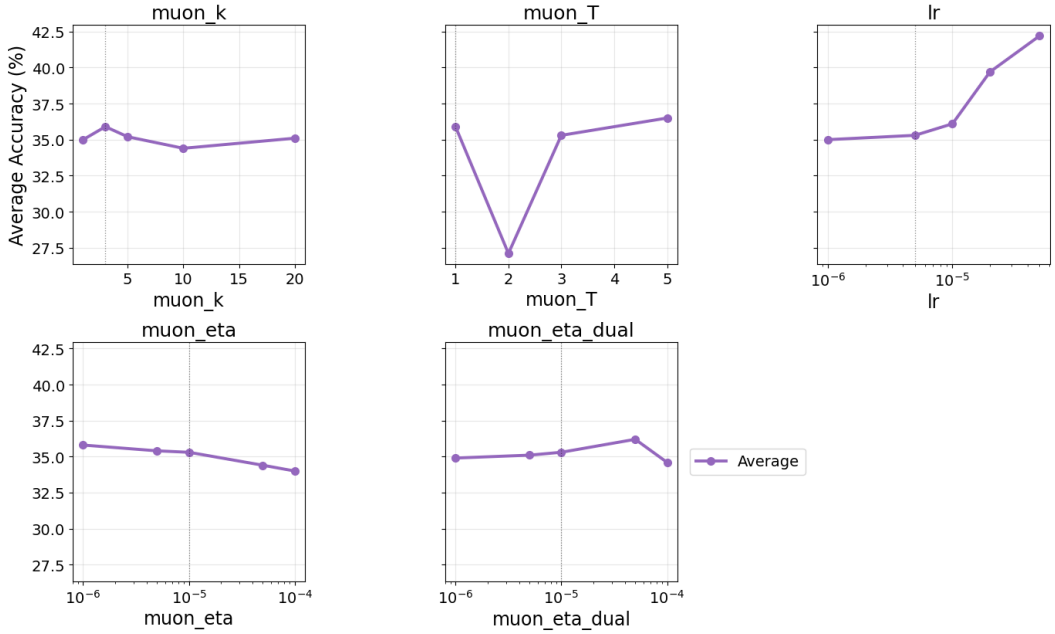

Figure 5: Ablation study of OSD hyperparameters on the Qwen2.5-1.5B architecture. The figure illustrates the impact of varying the protected subspace rank (`muon_k`), Newton-Schulz iterations (`muon_T`), base learning rate (`lr`), primal step size (`muon_eta`), and dual step size (`muon_eta_dual`) on the final average accuracy.

G.4 Orthogonalization Precision

The computational precision of the orthogonal projections is controlled by Randomized SVD iterations (`SVD_NITER`) in T5, and Newton-Schulz iterations (`muon_T`) in Qwen2.5. Table 16 shows that increasing `SVD_NITER` from 2 to 4 yields marginal gains. Similarly, Figure 5 highlights a non-linear response for `muon_T`, with strong performance at $T = 5$ and a dip at $T = 2$. Each Newton–Schulz iteration is dominated by matrix multiplications (roughly $\mathcal{O}(mn \min\{m, n\})$ for an $m \times n$ matrix, depending on orientation), so lower iteration counts reduce step time substantially. In our experiments, $T = 1$ and `SVD_NITER`=4 provide a favorable empirical accuracy–cost trade-off.

Table 16: Effect of randomized SVD iterations on the T5 standard benchmark. Fixed hyperparameters: LR=1×10^{-3}, MUON_LR=1×10^{-3}, MUON_ETA_DUAL=1×10^{-4}.

SVD_NITER	**Order-1**	**Order-2**	**Order-3**	**Avg**
2	75.7	72.9	74.2	74.3
4	75.5	73.7	73.6	74.3

H Detailed Per-Task Results

Table 17: Per-task performance on the TRACE benchmark with LLaMA-2-7B-Chat.

Task	After Task (%)	Final (%)
C-STANCE	$43.4_{\pm 1.3}$	$44.6_{\pm 1.4}$
FOMC	$71.6_{\pm 2.8}$	$58.9_{\pm 2.1}$
MeetingBank	$52.0_{\pm 0.2}$	$49.2_{\pm 0.9}$
Py150	$51.7_{\pm 0.8}$	$51.3_{\pm 1.0}$
ScienceQA	$84.6_{\pm 0.0}$	$70.0_{\pm 3.1}$
NumGLUE-cm	$27.2_{\pm 2.1}$	$25.1_{\pm 0.7}$
NumGLUE-ds	$57.0_{\pm 0.4}$	$55.6_{\pm 0.7}$
20Minuten	$40.7_{\pm 0.1}$	$40.7_{\pm 0.1}$
Average	$\mathbf{53.5}_{\pm 0.8}$	$\mathbf{49.4}_{\pm 0.3}$

Table 18: Order-1 per-task results (T5-Large): DBpedia → Amazon → Yahoo → AGNews.

Task	After Task	Final
DBpedia	$98.9_{\pm 0.0}$	$98.3_{\pm 0.4}$
Amazon	$59.9_{\pm 1.2}$	$55.3_{\pm 3.1}$
Yahoo	$73.8_{\pm 0.3}$	$71.6_{\pm 0.7}$
AGNews	$90.5_{\pm 0.1}$	$90.5_{\pm 0.1}$
AA: $78.9_{\pm 0.8}$	BT: $-2.5_{\pm 0.6}$	

Table 19: Order-2 per-task results (T5-Large): DBpedia → Amazon → AGNews → Yahoo.

Task	After Task	Final
DBpedia	$98.9_{\pm 0.0}$	$98.4_{\pm 0.2}$
Amazon	$59.9_{\pm 1.2}$	$57.7_{\pm 2.3}$
AGNews	$90.0_{\pm 0.3}$	$87.6_{\pm 0.4}$
Yahoo	$74.0_{\pm 1.0}$	$74.0_{\pm 1.0}$
AA: $79.4_{\pm 0.7}$	BT: $-1.7_{\pm 0.5}$	

Table 20: Order-3 per-task results (T5-Large): Yahoo → Amazon → AGNews → DBpedia.

Task	After Task	Final
Yahoo	$73.8_{\pm 0.5}$	$70.5_{\pm 1.1}$
Amazon	$60.2_{\pm 0.5}$	$55.7_{\pm 4.0}$
AGNews	$89.9_{\pm 0.4}$	$87.9_{\pm 1.0}$
DBpedia	$98.9_{\pm 0.1}$	$98.9_{\pm 0.1}$
AA: $78.3_{\pm 1.0}$	BT: $-3.3_{\pm 1.4}$	

Table 21: Order-4 per-task results (T5-Large): MNLI → CB → WiC → COPA → QQP → BoolQA → RTE → IMDB → Yelp → Amazon → SST-2 → DBpedia → AGNews → MultiRC → Yahoo.

Task	After Task	Final
MNLI	$84.9_{\pm 0.4}$	$47.2_{\pm 5.2}$
CB	$89.9_{\pm 1.7}$	$60.7_{\pm 2.9}$
WiC	$55.7_{\pm 4.3}$	$56.3_{\pm 1.6}$
COPA	$45.7_{\pm 1.7}$	$58.0_{\pm 4.5}$
QQP	$85.0_{\pm 2.4}$	$79.3_{\pm 4.8}$
BoolQA	$82.8_{\pm 0.3}$	$76.0_{\pm 2.5}$
RTE	$86.8_{\pm 1.2}$	$75.6_{\pm 5.5}$
IMDB	$93.9_{\pm 0.3}$	$94.2_{\pm 0.5}$
Yelp	$62.6_{\pm 2.1}$	$52.5_{\pm 2.5}$
Amazon	$60.3_{\pm 1.0}$	$51.7_{\pm 2.3}$
SST-2	$94.8_{\pm 0.4}$	$93.1_{\pm 1.7}$
DBpedia	$98.6_{\pm 0.4}$	$98.7_{\pm 0.0}$
AGNews	$90.1_{\pm 0.0}$	$89.2_{\pm 0.6}$
MultiRC	$73.6_{\pm 1.6}$	$76.0_{\pm 0.2}$
Yahoo	$74.3_{\pm 0.4}$	$74.3_{\pm 0.4}$
AA: $72.2_{\pm 0.8}$		BT: $-6.9_{\pm 1.0}$