COLORS: CONTINUAL LEARNING WITH CONSOLIDATED ORTHOGONAL LORA SUBSPACES

Anonymous authors
Paper under double-blind review

ABSTRACT

We introduce Consolidated Orthogonal LoRA Subspaces (COLORS), a continual learning framework that enables foundational models to learn an expanding set of tasks while consolidating shared skills, with minimal parameter growth and without forgetting previously learned tasks. COLORS makes no assumptions about task ordering and provably preserves previously learned tasks exactly. Under a cheaply verifiable incoherence condition between tasks, we further show that each new task can be learned to near-oracle quality. Our approach addresses a growing challenge created by the widespread use of low-rank adapters (LoRAs): while LoRAs make it easy to adapt foundation models to individual tasks, they provide no principled mechanism for organizing or consolidating knowledge across tasks. Existing continual learning methods either repeatedly update a shared subspace, causing *task drift*, or add new parameters for each new task, which increases model size and training costs as more tasks are learned. COLORS instead organizes LoRA updates into orthogonal subspaces that preserve previously learned task solutions while consolidating reusable structures within a compact parameter budget, $3.5\times$ smaller than separate adapters. Across challenging image-to-image transformation task sets, including image restoration, we demonstrate that COLORS continually adapts multiple foundational-model backbones while often outperforming state-of-the-art continual learning methods in reconstruction quality.

1 INTRODUCTION AND BACKGROUND

Continual learning aims to adapt a model to new tasks without *catastrophically forgetting* those learned before (McCloskey & Cohen, 1989). Unlike conventional finetuning, where learning a new task can interfere with and overwrite previously learned capabilities, continual learning must acquire new capabilities while retaining old ones. Here, we consider a particularly strict setting where new tasks arrive sequentially, and once a task has been learned, its training data is no longer available for retraining or supervision. This setting arises naturally when retaining past data is limited by privacy, storage, or operational constraints, for example, in healthcare applications.

Existing continual learning methods address this problem through data replay (Rebuffi et al., 2017; Shin et al., 2017), regularization (Kirkpatrick et al., 2017; Li & Hoiem, 2017), gradient projection (Saha et al., 2021; Farajtabar et al., 2020), or parameter isolation (Rusu et al., 2016). Replay requires retaining past information; Regularization and gradient projection provide only approximate retention of previous tasks; and parameter isolation prevents interference between tasks but limits knowledge sharing and causes parameter counts to grow with the number of tasks. This tension between knowledge sharing, interference, and parameter growth has become particularly important with the rise of foundation models (Brown et al., 2020; Rombach et al., 2022; Esser et al., 2024; Brooks et al., 2024) and low-rank adapters (LoRAs) (Hu et al., 2022).[1]

LoRAs enable efficient task-specific adaptation using small low-rank updates but inherit the same knowledge sharing–interference tradeoff. Separate LoRAs prevent interference but cannot share knowledge and grow with the number of tasks, while shared LoRA spaces risk interfering with previous tasks. Existing continual LoRA methods either provide only approximate protection (Wang et al., 2023; Liang & Li, 2024; Yu et al., 2025) or repeatedly update shared subspaces, causing

[1] A detailed discussion of prior work is in the supplement.

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.

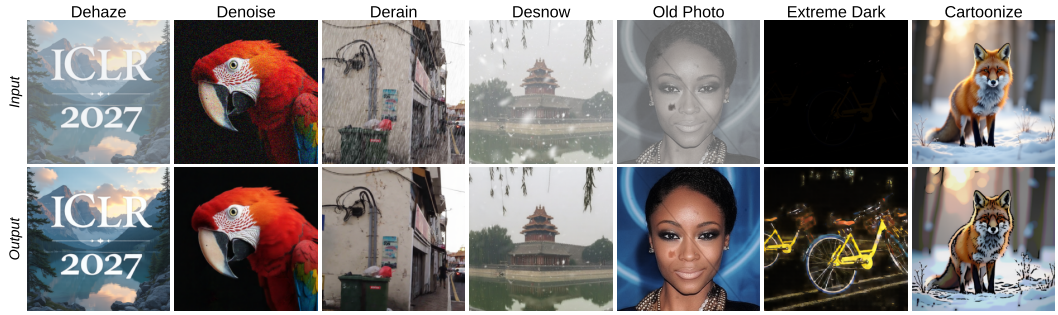

Figure 1: *Teaser*. We present COLORS, a continual learning framework for low-rank adaptors used in finetuning foundational models with provably zero-forgetting and flat training cost across increasing tasks. Our method is evaluated upon a wide-range of image-to-image benchmarks such as image restoration tasks and also image stylization tasks such as *cartoonization*.

previous tasks to drift (Kaushik et al., 2026). What is missing is a way to continually *consolidate* knowledge across LoRAs while guaranteeing that previously learned tasks remain unchanged.

We introduce **Consolidated Orthogonal LoRA Subspaces (COLORS)**, a continual learning framework that *reuses what is shared across tasks while keeping what is new orthogonal to what has already been learned*. COLORS represents each task using a shared basis of orthonormal LoRA directions and a small task-specific residual. As new tasks arrive, *rank consolidation* promotes recurring residual structure into the shared basis while keeping the remaining residual orthogonal to previously learned directions. Once learned, existing basis directions are never modified. Instead, new directions are appended to the basis, and previous tasks are re-expressed exactly in the new expanded basis representation. This guarantees non-forgetting of previously learned tasks, independent of the task number or order.

We make consolidation efficient by using a rank-space factorization of the Gram–Schmidt operator, avoiding repeated orthogonalization in the full parameter space. We further show that new-task capacity depends on its overlap with directions committed to previous tasks: under a verifiable incoherence condition, each task can approach the quality of an independently trained oracle adapter. We estimate this condition before training using inexpensive probe adapters. Following prior adapter-based continual learning methods (Kaushik et al., 2026; Wang et al., 2023; Liang & Li, 2024), we evaluate COLORS on image tasks, where ground-truth and oracle models enable direct measurement of forgetting and new-task performance. Our results demonstrate that across 16 image-restoration tasks, COLORS achieves within 0.15 dB of a jointly trained oracle (24.12 vs. 24.27 dB) with zero forgetting while maintaining a nearly constant training cost with growing tasks (3.2–3.7 ms per layer from 2 to 32 stored tasks).

2 METHOD: COLORS

We consider tasks $t = 1, 2, \ldots$ that arrive sequentially and share a frozen pretrained backbone network. Each task is learned from its own data as a low-rank adapter D_t, without access to data from previous tasks. COLORS has three main components. First, it represents each adapter using a common basis and a small task-specific residual. Second, it trains each new residual orthogonal to previously learned directions to prevent interference. Third, it consolidates recurring residual structure into the common basis without changing any previously learned adapter. This allows the representation to share structure across tasks while preserving old tasks exactly.

2.1 SHARED-RESIDUAL TASK REPRESENTATION

LoRA adapters across related tasks often share directions in weight space Kaushik et al. (2026). Motivated by this, instead of storing each adapter separately, we store shared directions once in a common basis, while each task stores only its coefficients over this common basis and a private residual.

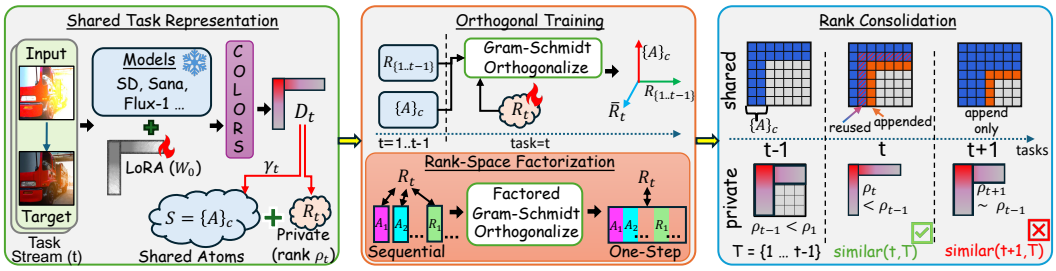

Figure 2: *Method Overview*. COLORS operates in three main stages. *(i)* A shared task representation comprising of orthonormal components and a private task representation to store task specific learnable directions. *(ii)* We employ orthogonalality via the Gram-Schmidt process embedded during training time through an efficient rank-space factorization. *(iii)* We perform an offline data free rank-consolidation after each task is learned to consolidate shared directions from private directions for each task.

Consider a target layer with frozen weights $W_0 \in \mathbb{R}^{o\times i}$, with i input and o output dimensions. All target layers are handled independently in the same way. After learning tasks $1 \dots t-1$, we maintain a shared basis $\mathcal{S} = \{A_1, \dots, A_c\}$ of c *immutable* orthonormal rank-one directions, which we call *atoms*:

$$A_j = u_j v_j^\top, \quad \langle A_j,\, A_k \rangle_F = \delta_{jk}, \tag{1}$$

where $u_j \in \mathbb{R}^o$ and $v_j \in \mathbb{R}^i$ are unit vectors, $\langle A,\, B \rangle_F = \mathrm{tr}(A^\top B)$ is the Frobenius inner product, and δ_{jk} is the Kronecker delta. Each atom $A_j \in \mathbb{R}^{o\times i}$ therefore represents a rank-one direction in the layer's weight space. [2]

We represent each task adapter D_t by coefficients $\gamma_t \in \mathbb{R}^c$ over the shared basis and a private residual $R_t \in \mathbb{R}^{o\times i}$:

$$D_t \;=\; \underbrace{\textstyle\sum_{j=1}^{c} \gamma_{t,j} A_j}_{S\gamma_t \text{ (shared)}} + R_t, \qquad \langle R_t,\, A_j \rangle_F = 0 \;\; \forall j, \qquad \langle R_t,\, R_s \rangle_F = 0 \;\; \forall s < t, \tag{2}$$

where $S\gamma_t$ denotes the shared directions. Each task reuses shared directions through γ_t, while R_t stores directions not represented by the shared basis or previous residuals. This reconstructs D_t up to the storage tolerance $\sqrt{\varepsilon} \approx 3\times10^{-6}$.

For efficiency, we store residuals in low-rank form, $R_t = B_t C_t$, with $B_t \in \mathbb{R}^{o\times\rho_t}$ and $C_t \in \mathbb{R}^{\rho_t\times i}$. The residual rank ρ_t is adapted as the shared basis grows (Sec. 2.3). Therefore, for an input $x \in \mathbb{R}^i$:

$$y = W_0 x + \alpha_t \big(\underbrace{U\big(\gamma_t \odot (Vx)\big)}_{S\gamma_t x \text{ (shared)}} + B_t (C_t\, x) \big), \tag{3}$$

where $U = [u_1, \dots, u_c] \in \mathbb{R}^{o\times c}$, $V = [v_1, \dots, v_c]^\top \in \mathbb{R}^{c\times i}$, and $\odot$ denote the Hadamard product. This evaluates $y = W_0 x + \alpha_t D_t x$ without forming D_t, and α_t controls adapter strength at inference; $\alpha_t = 1$ during training.

2.2 Orthogonal Residual Learning

When a new task t arrives, the shared atoms are frozen, and we train only its coefficients $\gamma_t \in \mathbb{R}^c$ and rank-ρ_t residual factors B_t and C_t. To restrict the residual to new directions, we project it against the $m = c + t - 1$ previously stored directions $\mathcal{E} = \{A_1..A_c, R_1..R_{t-1}\}$. Let $e_1, \dots, e_m$ be their Gram–Schmidt orthogonalization. During each forward pass, we project the residual as:

$$\widetilde{R}_t \;=\; B_t C_t - \sum_{j=1}^{m} \frac{\langle B_t C_t,\, e_j \rangle_F}{\|e_j\|_F^2}\, e_j, \tag{4}$$

This enforces orthogonality by construction rather than through a soft penalty (Wang et al., 2023).

Rank-space factorization. Directly evaluating Eq. (4) is expensive because each $e_j \in \mathbb{R}^{o\times i}$. Instead, all references are stored in low-rank form $D_j = P_j Q_j$, with rank $r_j \ll \min(o, i)$, allowing their Frobenius inner products to be computed as:

$$\langle P_j Q_j,\, P_k Q_k \rangle_F = \mathrm{tr}\big(Q_j\, Q_k^\top P_k^\top P_j\ \big) = \mathbf{1}^\top \big[(P_j^\top P_k) \odot (Q_j Q_k^\top) \big] \mathbf{1}. \tag{5}$$

[2] The atom count c is independent of the task count t. The basis is initialized from task 1 and grows by at most one atom per task, up to $c_{\max}$ (Sec. 2.3).

Before each new task, we run Gram–Schmidt once on the cached $m \times m$ reference Gram matrix, producing a lower-triangular matrix $G \in \mathbb{R}^{m \times m}$ whose row g_j expresses e_j in the original references. At each training step, we compute only the m live inner products $z_j = \langle B_t C_t,\, D_j \rangle_F$ using Eq. (5), and apply the projected residual directly in factored form:

$$\widetilde{R}_t\, x = B_t C_t\, x + \sum_{j=1}^{m} \kappa_j\, P_j(Q_j x), \qquad \kappa^\top = -p^\top G, \;\; p_j = \frac{g_j^\top z}{\|e_j\|_F^2}. \tag{6}$$

Here, p gives the normalized overlaps with the orthogonalized directions, and G maps them to coefficients κ over the stored low-rank references. Thus, Eq. (6) implements Eq. (4) without forming dense $o \times i$ projection matrices.

For a fixed rank ρ_t, the per-step cost is $O\big(\rho_t\,(o+i)\sum_{j=1}^{m} r_j\big)$. Consolidation (Sec. 2.3) moves the recurring adapter structure into the shared basis and reduces later residual ranks toward a smaller floor $\rho_{\min}$, keeping this cost nearly flat in practice. Gradients flow only through B_t, C_t, and γ_t.

2.3 Shared-Basis and Rank Consolidation

The initial basis is learned from task 1, which cannot reveal structure that recurs later in the stream. We therefore identify recurring structure in subsequent residuals and promote it to the shared basis, allowing future tasks to reuse it through γ_t with lower residual rank.

We initialize the basis using the top-c_0 singular pairs of task 1's adapter, storing the corresponding singular values as its coefficients. After each subsequent task t, we compute the mean residual $\bar{R} = \frac{1}{t}\sum_{s \le t} R_s$ and project its existing left atom directions $u_1 \dots u_c$ out of $\bar{R}$ via Gram–Schmidt. The leading singular pair (u, v) of the projected residual defines a candidate atom $uv^\top$ orthogonal to the existing basis. We measure how much residual energy it explains as:

$$e \;=\; \frac{\sum_{s \le t} \langle R_s,\, uv^\top \rangle_F^2}{\sum_{s \le t} \|R_s\|_F^2}. \tag{7}$$

Here, e is the fraction of the residuals' total energy (squared norm) captured by projection onto the candidate direction. If $e \ge \tau$ for a fixed threshold τ and atom count $c < c_{\max}$, we *append* the candidate to the shared basis as atom A_{c+1} and move its component from each residual into the shared coefficients:

$$\gamma_{s,c+1} \leftarrow \langle R_s,\, A_{c+1} \rangle_F\,, \qquad R_s \leftarrow R_s - \gamma_{s,c+1} A_{c+1}. \tag{8}$$

This update makes every residual orthogonal to the newly added atom while leaving its stored adapter *unchanged*. Already existing atoms are never modified.

As consolidation captures more shared structure, we adapt the residual rank using its singular-value spectrum. Let $r_\eta(R)$ be the smallest rank that retains a fraction η of R's total squared singular-value energy. We set the rank for the next task as:

$$\rho_{t+1} = \mathrm{median}_{\text{layers}}\; r_\eta(R_t), \;\; \text{clamped to } [\rho_{\min}, \rho_t], \tag{9}$$

where $\rho_{\min}$ is the rank floor. Thus, the residual rank can decrease as more reusable structure is consolidated into the shared basis. In our experiments, atoms promoted to the shared basis explain $25\% - 99\%$ of subsequent residual energy, while the residual rank ρ_t drops from 16 to 4 by the third task and remains at this floor, reducing subsequent storage and training cost.

Proposition 1 (**Zero forgetting by construction**). *Let $\widehat{D}_t = S\gamma_t + R_t$ denote task t's stored representation at any later point in the stream, and let D_t be its adapter as originally learned. Then $\left\|\widehat{D}_t - D_t\right\|_F \le (1+\nu_t)\sqrt{\varepsilon}\,\|D_t\|_F$, where ε is the storage truncation budget (10^{-11} in all experiments) and ν_t is the number of atoms appended after task t. This bound is independent of the number of subsequent tasks, their arrival order, and consolidations.*

Consolidation may repeatedly change the stored coordinates (γ_t, R_t), but the reconstructed adapter $\widehat{D}_t$ remains unchanged. The proof is provided in the supplement.

3 When Can Orthogonal Learning Succeed? A Capacity Analysis

Zero forgetting imposes a strict constraint: new tasks cannot use residual directions already committed to prior tasks. Therefore, the central question is: *Does enough capacity remain to learn new tasks?* We outline the core guarantee below and defer the full analysis, including the capacity bounds, lower bounds under task overlap, and formal proofs, to Appendix D.

3.1 Learnability and the Incoherence Condition

The success of orthogonal consolidation depends on a geometric property of the task sequence. Let D_t^* be the optimal, unconstrained adapter for task t. Its *private component* E_t is the portion of these optimal weights that falls outside the shared atom span $\mathcal{S}$.[3]

A task sequence is pairwise μ-*incoherent* if the normalized private components of any two tasks (for stored tasks, their learned residuals R_s) have an absolute cosine similarity of at most μ. Intuitively, tasks may share substantial structure through the common basis, while their remaining private components overlap little. Highly coherent tasks (large μ) compete for the same private directions, causing capacity loss (Theorem B, Appendix D). Conversely, incoherent tasks (small μ) have little private overlap, and this loss remains bounded.

Theorem A (**Learning under incoherence**). *Assume a bounded, L-Lipschitz loss, normalized inputs and bounded adapters. Let $\hat{h}$ be the empirical risk minimizer over our constrained class using S_t samples, and h^* the unconstrained optimum. Let $\mathcal{Q}_t = \operatorname{span}\{R_s\}_{s<t}$ be the span blocked by earlier residuals, $K_t = \dim(\mathcal{S} \oplus \mathcal{Q}_t)$, and G_t the Gram matrix of the normalized earlier residuals. If the task family is μ-incoherent and the eigenvalue $\lambda_{\min}(G_t) > 0$, then with high probability:*

$$\mathcal{R}(\hat{h}) - \mathcal{R}(h^*) \;\le\; \underbrace{\tilde{\mathcal{O}}\Big(\sqrt{\tfrac{c+\rho_t(o+i)}{S_t}}\Big)}_{\textit{Estimation}} \;+\; \underbrace{L\Big(\tfrac{\mu\sqrt{K_t}}{\sqrt{\lambda_{\min}(G_t)}}\,\|E_t\|_F + \operatorname{tail}_{\rho_t}\Big)}_{\textit{Approximation}},$$

where $\operatorname{tail}_{\rho_t}$ is the error of the best rank-ρ_t approximation of the part of D_t^* outside $\mathcal{S} \oplus \mathcal{Q}_t$.

This bound yields two practical insights. The **estimation term** dictates sample complexity: fitting only c coefficients and a rank-ρ_t residual requires far fewer samples than a full-rank update. The **approximation term** is the price of the orthogonality constraint: the ideal private update E_t pointing into directions blocked by earlier tasks remains strictly out of reach. Incoherence keeps this blocked portion small, and a well-conditioned G_t prevents compounding errors across tasks. (empirically, $\lambda_{\min}(G_t) \ge 0.34$ in our measured task streams).

This mirrors the ERM analysis of Share (Kaushik et al., 2026) under similar assumption sets, but with a dual philosophy: their similarity condition requires tasks to be highly alike so one basis serves all, whereas our incoherence condition requires private parts to be distinct so every residual stays reachable.

3.2 Probing Task Sets Before Training

The theoretical bound relies on the true incoherence μ, which is technically unknown before training. To estimate it, we introduce *pre-flight probes*: before fully training task t, we train a throwaway rank-4 probe adapter for just 300 steps ($\sim$55s per GPU).

Pairwise coherence among these probes serves as our score. This relies on *probe-faithfulness*, the geometric property that early training directions reliably predict final converged geometry, which we prove for the linearized regime (Prop. S3) and validate empirically in Sec. E.4. This score directly steers compute allocation. A low probe score indicates new private structure, justifying full-rank training. A high probe score warns that private directions are already taken: overlapping with *locked* residuals provably forfeits capacity (Theorem B), while shared atom mass remains easily reachable. Such highly coherent tasks can therefore train safely at the rank floor $\rho_{\min}$ or reuse a stored adapter. Indeed, empirically two of the three real flagged tasks in Sec. E.4 converged at $\rho_{\min}$ while adding a single new atom. Sec. E.4 validates this across seeds and architectural backbones.

4 Experiments

4.1 Implementation

Objective. Our per-task loss must map an input image to a paired target rather than generate from noise, since sampling from noise would re-synthesize content and discard the input. We therefore

[3]Note that E_t is the oracle's theoretical private component, whereas R_t is the private residual learned under our orthogonality constraint.

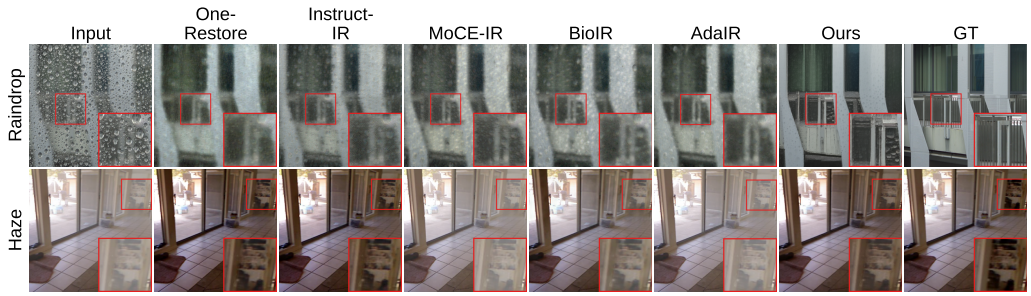

Figure 3: *Qualitative Results on Restoration.* Our method *(continual)* was compared with SOTA all-in-one restoration methods learned under the *easier* joint learning setup for a fair comparison.

adopt the deterministic image-to-image bridge of Liu et al. (2023). With paired latents $(x^{\text{in}}, x^{\text{tgt}})$, a mixing time $s \sim \mathcal{U}[0,1]$ and $x_s = (1-s)\,x^{\text{tgt}} + s\,x^{\text{in}}$, the backbone f_θ trains with the loss

$$\mathcal{L}_t = \mathbb{E}\left\| f_\theta(x_s,\, s,\, \tau_t) - (x^{\text{in}} - x^{\text{tgt}}) \right\|_2^2, \tag{10}$$

where τ_t is a per-task text prompt. Inference starts at the input latent and integrates the learned direction toward the target in 16 Euler steps. Every method trains this loss on identical data, so all differences are attributable solely to adapters the continual learning method used.

Training. Adaptation should leave the backbone's interfaces intact while giving each task room to learn. Our low-rank adapters therefore target every linear layer of the self-attention blocks, while cross-attention stays untouched to preserve the text interface for task selection. All methods use the AdamW optimizer Loshchilov & Hutter (2019) at learning rate 10^{-4} with gradient clipping 1.0, batch size 1 at an image resolution of 512 trained for 15k to 40k steps per task over backbone. Our configuration sets $c_0 = \rho_1 \in \{16, 32\}$ with the backbone width, $c_{\max} \in \{24, 48\}$ accordingly, and τ=0.1, η=0.99, $\rho_{\min}$=4 everywhere.

Models. We evaluate our continual learning method not on a single model but on different families of foundational generative image backbones such as Sana-1600M, Sana-4.8B Xie et al. (2025a), SD 1.5, SD3.5-Medium Esser et al. (2024) and FLUX.1-dev Black Forest Labs (2024) in latent space and PixelGen Ma et al. (2026) in pixel space, spanning 0.9B to 12B parameters. We also evaluate SANA-video for continual learning on videos.

4.2 Image-to-Image Benchmarks

We evaluate our method on a diverse set of task streams. We therefore assemble 16 atomic restoration tasks following the task suite of UnSCAR (Mandal et al., 2026), and the tasks differ widely in nature. Furthermore, we also evaluate on diverse tasks beyond image restoration such as stylization. We also expand our evaluation suite to video restoration.

Table 1: Ours (continual) against five all-in-one architectures retrained *jointly* on the same data, their non-continual upper bound (per-task PSNR, dB, over all 8 tasks; additional metrics such as LPIPS/SSIM in the appendix). The final column is the mean over the 8 tasks, jointly trained for them and continual for ours.

method	haze	rain	snow	drop	shadow	noise	deblur	expo.	mean
OneRestore	**28.0**	30.2	26.4	18.1	21.5	**30.2**	28.5	21.2	**25.5**
InstructIR	26.7	30.3	26.6	15.9	17.8	25.8	29.1	19.9	24.0
MoCE-IR	26.4	31.2	26.5	17.1	17.2	25.8	29.2	17.6	23.9
BioIR	26.1	31.0	26.5	17.2	16.6	23.9	29.7	17.5	23.6
AdaIR	26.9	**31.3**	**27.2**	16.8	17.3	27.0	**30.1**	18.3	24.4
ours (continual)	26.7	28.7	26.5	**19.0**	**28.2**	26.7	27.0	**21.4**	25.0

All-in-one Restoration Table 1 compares our approach against unified architectures trained jointly on all data upfront. These baselines maintain a small edge on pure reconstruction tasks while we dominate global corrections like shadow removal and raindrop removal.

Table 2: Continual learning on Sana Xie et al. (2025a) as the stream grows from 4 to 16 tasks, reported as mean PSNR (dB) over the tasks seen so far. Baseline methods use official implementation. Ours and the independent adapters use only the current task adapter. Prefix-sequential GS activates adapters $1:k$ for task k. O-LoRA (Wang et al., 2023) and InfLoRA (Liang & Li, 2024) keep all adapters active. PCA-recompute denotes Share (Kaushik et al., 2026). The joint oracle trains on all tasks at once. The bottom rows repeat the stream on SD3.5-M.

Method	4 tasks	8 tasks	12 tasks	16 tasks
Ours	21.85	**21.80**	**22.12**	21.05
Independent adapters	21.73	21.63	21.97	20.66
Prefix-sequential GS	21.66	21.67	22.01	20.85
PCA-recompute	**21.88**	21.71	22.05	**21.14**
O-LoRA	17.70	14.91	14.88	9.80
InfLoRA	16.87	14.89	15.13	9.89
Naive sequential	17.26	17.26	12.96	8.02
Joint oracle	21.65	20.97	21.20	20.06
Degraded input	17.28	19.85	21.82	18.67
SD3.5-M flagship stream:				
Ours	26.09	25.67	25.52	24.12
Naive sequential	18.24	17.37	11.48	6.98

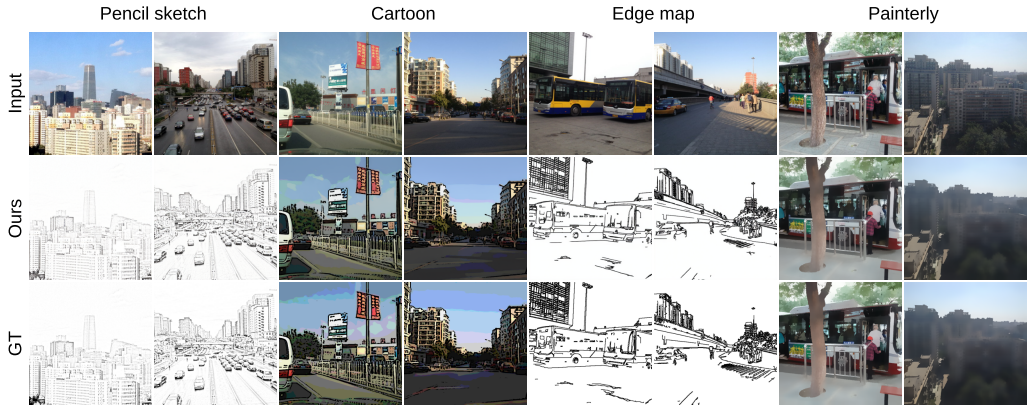

Figure 4: *Generalization to diverse tasks.* Our method, continually trained on eight restoration and four style tasks, generalizes to diverse image stylization tasks. Compared to SOTA continual learning methods Kaushik et al. (2026), COLORS shows no task drift (Fig. 5) and requires substantially less training cost (Fig. 6).

Table 2 tracks quality as the stream grows from 4 to 16 tasks. Methods that deploy every stored adapter at once, and naive sequential fine-tuning, fall to 8 to 10 dB by 16 tasks. Ours stays level at 21 to 22 dB and ends above both the independent adapters and the Sana joint oracle, which sees all data up front. The rows that match our quality each pay for it elsewhere. Independent adapters share nothing and store $3.5\times$ more parameters, the prefix-sequential chain slows with every task, and PCA-recompute gets there by rewriting tasks it has already stored (Fig. 5.d). Ours is the only row that keeps top quality with no forgetting, flat training cost and base-model inference. On SD3.5 the same recipe reaches 24.12 dB, within 0.15 dB of its joint oracle.

Diverse Tasks and Domain Generalization: Our method is not limited to restoration. We add four style tasks after the eight restoration tasks: cartoonization, pencil sketch, edge map, and painterly. Each trains to 20.7–32.5 dB (Fig. 4), and every earlier restoration adapter stays identical. We also apply COLORS to video, learning five real video benchmarks in sequence with temporally consistent outputs (see Appendix).

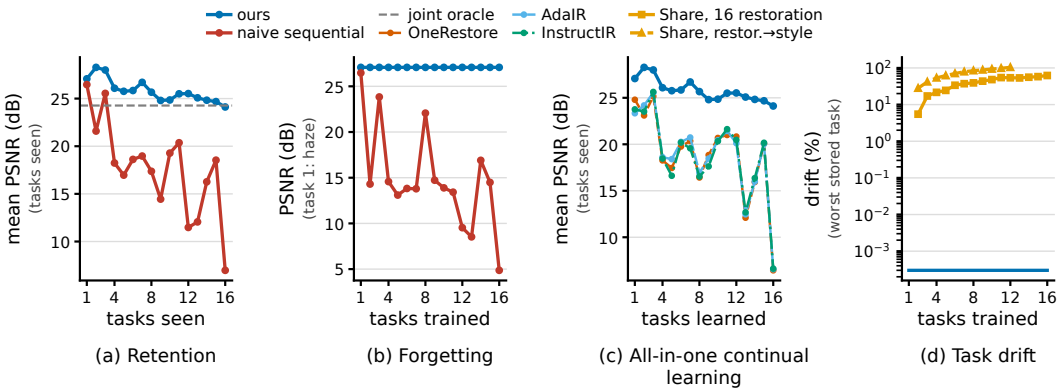

Figure 5: *Retention on the 16-task stream.* (a) Mean PSNR over tasks seen (SD3.5). (b) First-task PSNR after each later task. (c) All-in-one models trained sequentially. (d) Largest change of any stored task since storage. Ours stays near flat in every panel.

Table 3: Consolidating 16 public FLUX.1-dev adapters at increasing storage budgets. Top: final mean adapter storage error (%) Bottom: adapter drift after storage (%) The one-shot optimum is the best error reachable if all 16 adapters were known in advance and compressed together. Exact counts are in the supplement.

	storage budget				
method	0.18G	0.34G	0.77G	1.06G	1.5G
stored-adapter error (%):					
one-shot optimum	22.2	13.6	5.2	3.3	1.8
Share	54.8	34.1	18.1	13.9	9.5
ours	**34.4**	**22.7**	**6.4**	**4.7**	**2.2**
drift (%):					
Share	52.0	43.1	28.3	21.4	13.4
ours	**0**	**0**	**0**	**0**	**0**

4.3 Task Retention and Performance Drift

Figure 5 shows the true cost of sequential learning. Naive sequential training overwrites what it learned before, so the first task falls from 26.5 dB to 4.9 dB by the end of the stream (b). All-in-one models fare no better (c). All five forget in the same pattern, which points to the methods being sensitive to the task order rather than the model architecture as the driver. Share keeps quality longer but silently rewrites its stored tasks, which drift by up to 104% (d). Our method stays flat by construction. Every task keeps its arrival quality, the mean tracks the joint oracle within 0.2 dB (a), and stored adapters never move beyond numerical precision.

4.4 Consolidating Pretrained Adapters

Consolidation also works on adapters we did not train. We consolidate 16 public FLUX.1-dev generation adapters from HuggingFace one at a time, with no training. Table 3 sweeps the storage budget. At full budget we come within 0.4 points of the one-shot optimum, which sees every adapter up front, at 2.2% error and zero drift. Share instead degrades adapters after storing them, because each basis update rewrites earlier weights. At its default budget it ends at 54.8% error, of which 52.0% appeared after storage. At our budget Share reaches 9.5%, yet its first adapter still decays from 0.5% to 35.3%. We reach 6.4% with half that budget. Figure 7 shows the effect on generation, and the remaining adapters and exact parameter counts are in the supplement (Sec. E.8).

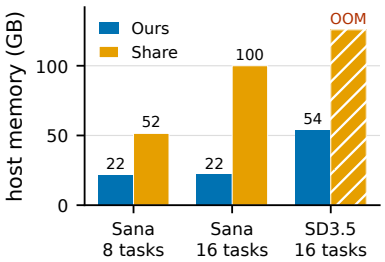

Figure 6: Peak host memory in training. Share runs out of memory (OOM) on SD3.5.

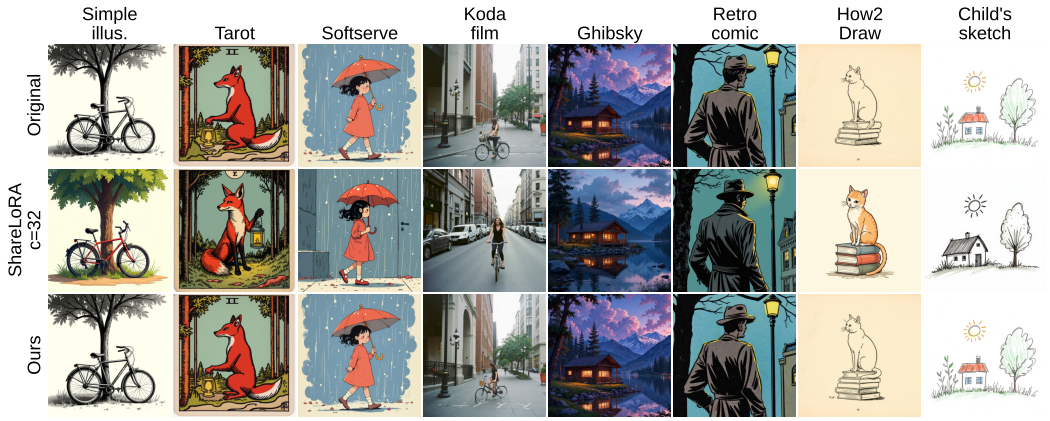

Figure 7: *Consolidating public FLUX.1-dev adapters.* Each column shows one of the 16 HuggingFace adapters, rendered with the same prompt and seed from the original adapter (top) and from its consolidated version after all 16 are stored. Share Kaushik et al. (2026) at its default budget (c=32) alters colors, objects and style. Ours reproduces the original.

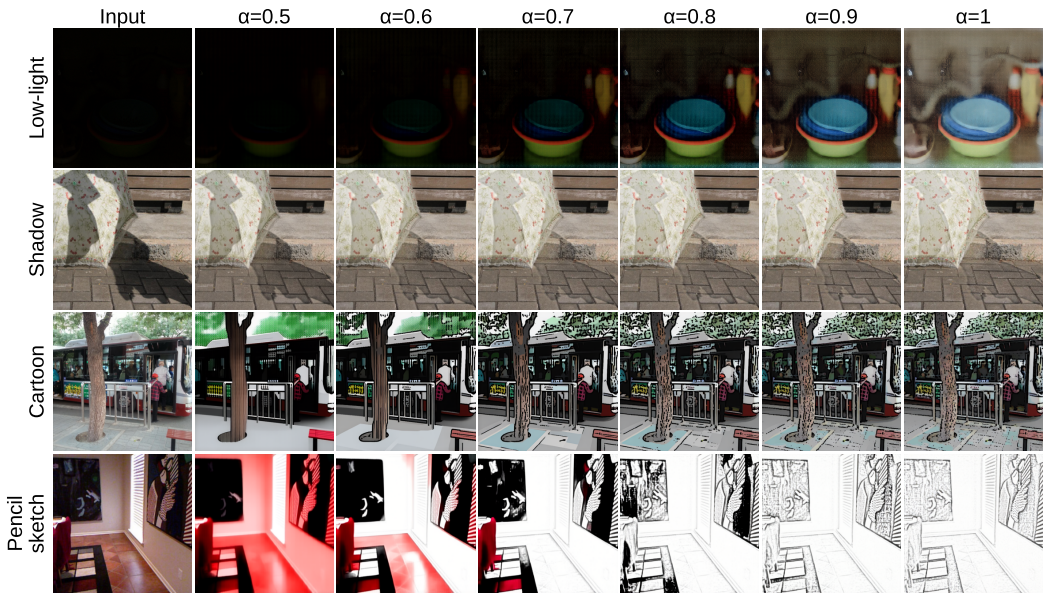

Figure 8: *Per-task strength control on SD3.5.* Each row scales one task's stored adapter by α, from half strength to the full trained adapter at α=1. Restoration (low-light) and stylization (pencil sketch) strengthen smoothly as α grows.

5 Discussion and Limitations

We provide additional analyses in the supplement, including full validation of the pre-flight probes (Sec. E.4), ablations on adaptive rank, seeds, and task order (Sec. E.3), and an analysis of per-task restoration strength control (Sec. E.5). Each task's restoration strength can also be dialed (zero-shot) at inference by scaling its adapter with α (Fig. 8 and Sec. E.5). The main limitation of our approach follows directly from the exact non-forgetting. A new task can freely reuse shared atoms but cannot access directions stored exclusively in an earlier task's private residual. As a result, a near-duplicate of an earlier task may lose part of its achievable gain. However, we find this case to be rare in practice: none of the 120 natural task pairs we evaluate shows duplicate-level overlap, and the pre-flight probe identifies such cases before training (Table 13). Another limitation is that the task must be specified at inference, through the prompt or the dial. Our model does not yet detect the degradation from the input image itself.

Ethics Statement

Our method supports image restoration and editing with large generative models, including tasks such as face restoration and image completion. These capabilities can also be used to modify image content. We conduct all experiments using public datasets and publicly available adapters for research purposes and introduce no new data about individuals. Our method does not add capabilities beyond those of the underlying models. It provides a more efficient way to learn and retain multiple existing tasks without forgetting.

Reproducibility Statement

We provide backbone, training, and implementation details in the supplement, together with complete proofs of our theoretical results. All reported numbers are obtained from measured runs, and we plan to release the full code and data needed to reproduce our results. The supplementary code already reproduces the consolidation experiment in Table 3 end to end without training. It downloads the 16 FLUX.1-dev adapters from HuggingFace, consolidates them using COLORS and computes the one-shot optimum. A single script reproduces all errors, drift, and parameter counts in Table 3 on one GPU in about 30 minutes, enabling direct verification of our storage and compression results.

AI Use Statement

We used large language models for programming assistance, exploratory analyses, proofreading, and as an aid in mathematical proving. While we established the theoretical direction and overarching proof strategies, LLMs were used to help identify specific algebraic expressions and bounds. We made all research decisions, including the method and experimental design, and carefully reviewed and verified all generated code, mathematical steps, and analyses. All research results and claims are verified, and we take full responsibility for the paper's content.

References

Mahmoud Afifi, Konstantinos G Derpanis, Björn Ommer, and Michael S Brown. Learning multi-scale photo exposure correction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2021.

Eirikur Agustsson and Radu Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, July 2017a.

Eirikur Agustsson and Radu Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *CVPR Workshops*, 2017b.

Michael S Albergo, Nicholas M Boffi, and Eric Vanden-Eijnden. Stochastic interpolants: A unifying framework for flows and diffusions. *Journal of Machine Learning Research*, 26(209):1–80, 2025.

Alif Ashrafee and Bartosz Krawczyk. Restoring without forgetting: Continual learning across image degradations. *arXiv preprint arXiv:2608.23799*, 2026.

Black Forest Labs. FLUX. `https://github.com/black-forest-labs/flux`, 2024.

Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Leo Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, et al. Video generation models as world simulators. *OpenAI Blog*, 1(8):1, 2024.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, 2020.

Jianrui Cai, Hui Zeng, Hongwei Yong, Zisheng Cao, and Lei Zhang. Toward real-world single image super-resolution: A new benchmark and a new model. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 3086–3095, 2019.

Emmanuel J. Candès and Yaniv Plan. Tight oracle inequalities for low-rank matrix recovery from a minimal number of noisy random measurements. *IEEE Transactions on Information Theory*, 57 (4):2342–2359, 2011.

Chen Chen, Qifeng Chen, Jia Xu, and Vladlen Koltun. Learning to see in the dark. In *CVPR*, 2018.

Haoyu Chen, Jingjing Ren, Jinjin Gu, Hongtao Wu, Xuequan Lu, Haoming Cai, and Lei Zhu. Snow removal in video: A new dataset and a novel method. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 13165–13176. ieee, 2023.

Hongming Chen, Xiang Chen, Chen Wu, Zhuoran Zheng, Jinshan Pan, and Xianping Fu. Towards ultra-high-definition image deraining: A benchmark and an efficient method. *IEEE Transactions on Multimedia*, 2026.

Junyu Chen, Han Cai, Junsong Chen, Enze Xie, Shang Yang, Haotian Tang, Muyang Li, Yao Lu, and Song Han. Deep compression autoencoder for efficient high-resolution diffusion models. In *ICLR*, 2025.

De Cheng, Yanling Ji, Dong Gong, Yan Li, Nannan Wang, Junwei Han, and Dingwen Zhang. Continual all-in-one adverse weather removal with knowledge replay on a unified network structure. *IEEE Transactions on Multimedia*, 26:8184–8196, 2024.

Marcos V Conde, Gregor Geigle, and Radu Timofte. InstructIR: High-quality image restoration following human instructions. In *European Conference on Computer Vision*, 2024.

Carl Eckart and Gale Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218, 1936.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. *arXiv preprint arXiv:2403.03206*, 2024.

Mehrdad Farajtabar, Navid Azizan, Alex Mott, and Ang Li. Orthogonal gradient descent for continual learning. In *International Conference on Artificial Intelligence and Statistics*, 2020.

Laniqng Guo, Chong Wang, Yufei Wang, Siyu Huang, Wenhan Yang, Alex C Kot, and Bihan Wen. Single-image shadow removal using deep learning: A comprehensive survey. *arXiv preprint arXiv:2407.08865*, 2024a.

Yu Guo, Yuan Gao, Yuxu Lu, Huilin Zhu, Ryan Wen Liu, and Shengfeng He. OneRestore: A universal restoration framework for composite degradation. In *European Conference on Computer Vision*, 2024b.

Qingbo Hao. MLRN, 2023. URL `https://doi.org`.

Roger A Horn and Charles R Johnson. *Matrix Analysis*. Cambridge University Press, 2012.

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.

Rakib Hyder, Ken Shao, Boyu Hou, Panos Markopoulos, Ashley Prater-Bennette, and M Salman Asif. Incremental task learning with incremental rank updates. In *European Conference on Computer Vision*, 2022.

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

Yeying Jin, Xin Li, Jiadong Wang, Yan Zhang, and Malu Zhang. Raindrop clarity: A dual-focused dataset for day and night raindrop removal. In *European Conference on Computer Vision*, pp. 1–17. Springer, 2024.

Kotha Kartheek, Lingamaneni Gnanesh Chowdary, and Snehasis Mukherjee. Continual learning-based unified model for unpaired image restoration tasks. *arXiv preprint arXiv:2507.19184*, 2025.

Prakhar Kaushik, Ankit Vaidya, Shravan Chaudhari, Rama Chellappa, and Alan Yuille. Shared LoRA subspaces for almost strict continual learning. *arXiv preprint arXiv:2602.06043*, 2026.

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.

Boyi Li, Wenqi Ren, Dengpan Fu, Dacheng Tao, Dan Feng, Wenjun Zeng, and Zhangyang Wang. Benchmarking single-image dehazing and beyond. *IEEE TIP*, 2019.

Boyun Li, Xiao Liu, Peng Hu, Zhongqin Wu, Jiancheng Lv, and Xi Peng. All-in-one image restoration for unknown corruption. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

Ruoteng Li, Robby T Tan, and Loong-Fah Cheong. All in one bad weather removal using architectural search. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12):2935–2947, 2017.

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

Guan-Horng Liu, Arash Vahdat, De-An Huang, Evangelos A Theodorou, Weili Nie, and Anima Anandkumar. I^2SB: Image-to-image Schrödinger bridge. In *International Conference on Machine Learning*, 2023.

Jianzhao Liu, Jianxin Lin, Xin Li, Wei Zhou, Sen Liu, and Zhibo Chen. LIRA: Lifelong image restoration from unknown blended distortions. In *European Conference on Computer Vision*, 2020.

Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*, 2019.

Ziwei Luo, Fredrik K Gustafsson, Zheng Zhao, Jens Sjölund, and Thomas Schön. Controlling vision-language models for multi-task image restoration. In *International Conference on Learning Representations*, pp. 16226–16246, 2024.

Zehong Ma, Ruihan Xu, and Shiliang Zhang. PixelGen: Improving pixel diffusion with perceptual loss. *arXiv preprint arXiv:2602.02493*, 2026.

Debabrata Mandal, Soumitri Chattopadhyay, Guansen Tong, and Praneeth Chakravarthula. UniCoRN: Latent diffusion-based unified controllable image restoration network across multiple degradations. *arXiv preprint arXiv:2503.15868*, 2025.

Debabrata Mandal, Soumitri Chattopadhyay, Yujie Wang, Marc Niethammer, and Praneeth Chakravarthula. UnSCAR: Universal, scalable, controllable, and adaptable image restoration. *arXiv preprint arXiv:2603.07406*, 2026.

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

Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of Learning and Motivation*, volume 24, pp. 109–165. Elsevier, 1989.

Leon Mirsky. Symmetric gauge functions and unitarily invariant norms. *The Quarterly Journal of Mathematics*, 11(1):50–59, 1960.

Yurii Nesterov. *Lectures on Convex Optimization*. Springer, 2018.

Ozan Özdenizci and Robert Legenstein. Restoring vision in adverse weather conditions with patch-based denoising diffusion models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(8):10346–10357, 2023.

Vaishnav Potlapalli, Syed Waqas Zamir, Salman Khan, and Fahad Shahbaz Khan. PromptIR: Prompting for all-in-one blind image restoration. In *Advances in Neural Information Processing Systems*, 2023.

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

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.

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

Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. In *Advances in Neural Information Processing Systems*, 2017.

James Seale Smith, Yen-Chang Hsu, Lingyu Zhang, Ting Hua, Zsolt Kira, Yilin Shen, and Hongxia Jin. Continual diffusion: Continual customization of text-to-image diffusion with C-LoRA. *arXiv preprint arXiv:2304.06027*, 2023.

George Stoica, Daniel Bolya, Jakob Bjorner, Pratik Ramesh, Taylor Hearn, and Judy Hoffman. ZipIt! merging models from different tasks without training. In *International Conference on Learning Representations*, 2024.

Jeya Maria Jose Valanarasu, Rajeev Yasarla, and Vishal M Patel. TransWeather: Transformer-based restoration of images degraded by adverse weather conditions. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

Roman Vershynin. *High-Dimensional Probability*. Cambridge University Press, 2018.

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

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, 2022.

Chen Wei, Wenjing Wang, Wenhan Yang, and Jiaying Liu. Deep retinex decomposition for low-light enhancement. *arXiv preprint arXiv:1808.04560*, 2018.

Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, and Ludwig Schmidt. Model soups: Averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International Conference on Machine Learning*, 2022.

Chen Wu, Pu Wang, and Zhuoran Zheng. Re-examine all-in-one image restoration: A catastrophic forgetting perspective. *Pattern Recognition Letters*, 197:161–167, 2025.

Enze Xie, Junsong Chen, Junyu Chen, Han Cai, Haotian Tang, Yujun Lin, Zhekai Zhang, Muyang Li, Ligeng Zhu, Yao Lu, and Song Han. SANA: Efficient high-resolution image synthesis with linear diffusion transformers. In *ICLR*, 2025a.

Enze Xie, Junsong Chen, Yuyang Zhao, Jincheng Yu, Ligeng Zhu, et al. SANA 1.5: Efficient scaling of training-time and inference-time compute in linear diffusion transformer. In *ICML*, 2025b.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin Raffel, and Mohit Bansal. TIES-merging: Resolving interference when merging models. In *Advances in Neural Information Processing Systems*, 2023.

Enneng Yang, Zhenyi Wang, Li Shen, Shiwei Liu, Guibing Guo, Xingwei Wang, and Dacheng Tao. AdaMerging: Adaptive model merging for multi-task learning. In *International Conference on Learning Representations*, 2024.

Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. Language models are super mario: Absorbing abilities from homologous models as a free lunch. In *International Conference on Machine Learning*, 2024.

Xiaobing Yu, Jin Yang, Xiao Wu, Peijie Qiu, and Xiaofeng Liu. FM-LoRA: Factorized low-rank meta-prompting for continual learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 6399–6408. IEEE, 2025.

Eduard Zamfir, Zongwei Wu, Nancy Mehta, Yuedong Tan, Danda Pani Paudel, Yulun Zhang, and Radu Timofte. Complexity experts are task-discriminative learners for any image restoration. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025.

Kaihao Zhang, Tao Wang, Wenhan Luo, Boheng Chen, Wenqi Ren, Bjorn Stenger, Wei Liu, Hongdong Li, and Yang Ming-Hsuan. Mc-blur: A comprehensive benchmark for image deblurring. In *IEEE Transactions on Circuits and Systems for Video Technology*, 2023.

Linqi Zhou, Aaron Lou, Samar Khanna, and Stefano Ermon. Denoising diffusion bridge models. In *The Twelfth International Conference on Learning Representations*, 2024.

A Appendix Organization

We organize the appendix into 5 sections:

1. **Related work**. We review prior work on all-in-one image restoration, continual learning with low-rank adapters, model merging and shared adapter subspaces, and explain how our approach differs from each.
2. **Implementation Details:** We list the backbones and the training settings for every run. We also describe the data splits and the evaluation protocol.
3. **Supporting Proofs:** We prove every proposition and theorem from the main paper. We also list the standard results that these proofs use.
4. **Additional Experiments:** We show more results for all sixteen restoration tasks and for the video restoration task stream. We show empirical validation for the pre-flight task probes. We also show various design ablations and measure it's effect on the performance and compute/storage costs.
5. **Interpreting the Adapters:** We show by visualizing attention maps how much of each task lives in the shared atoms and how much lives in its own residual. We then map where each part acts in the image.

B Related Work

Continual learning with low-rank adaptation. The central challenge in continual learning is to learn new tasks without changing the previously learned ones. Existing methods reduce interference through regularization (Kirkpatrick et al., 2017; Li & Hoiem, 2017), replay (Rebuffi et al., 2017), or task-specific prompts (Wang et al., 2022). However, these methods either retain information from previous tasks or provide only approximate protection against forgetting. Adapters like LoRA enables parameter-efficient continual adaptation through low-rank updates (Hu et al., 2022), but suffers a similar tradeoff between sharing knowledge across tasks and isolating tasks to prevent interference. Methods such as O-LoRA and InfLoRA constrain new updates relative to previous ones (Wang et al., 2023; Liang & Li, 2024), while C-LoRA and FM-LoRA use regularization to

reduce interference during diffusion customization (Smith et al., 2023; Yu et al., 2025). These constraints remain approximate and can allow previous tasks to drift. Strict rank-increment methods avoid this by isolating task updates but sacrifice sharing (Hyder et al., 2022). The Share framework instead constructs a common subspace through repeated PCA computation and truncated projections (Kaushik et al., 2026), but recomputing the subspace can create compounding lossy errors and change the representation of previous tasks. Overall, existing methods do not simultaneously provide *knowledge sharing and exact non-forgetting*. COLORS addresses this gap using an append-only shared basis that consolidates new knowledge along orthogonal LoRA subspaces without modifying previously learned tasks.

Knowledge consolidation across tasks. A related problem is how to combine knowledge learned independently across multiple tasks. Model merging aims to address this by combining independently fine-tuned models into a single checkpoint without accessing their original training data. Weight averaging (Wortsman et al., 2022) and Fisher-weighted averaging (Matena & Raffel, 2022) directly combine model parameters, while task arithmetic treats fine-tuning updates as vectors that can be added to a common base model (Ilharco et al., 2023). Because directly adding the weights causes interference, later methods remove or resolve conflicting components (Yadav et al., 2023), sparsify task updates (Yu et al., 2024), align internal representations (Stoica et al., 2024), or optimize merging coefficients (Yang et al., 2024). However, these methods resolve interference *after* independently training the tasks. For continual learning, we instead need a representation that prevents such interference as each task arrives while still extracting structure that can be shared. COLORS performs this consolidation during adaptation by separating shared directions from orthogonal task-specific components, without changing previously learned task functions. Our consolidation enforces strict orthogonality during adaptation, while our adapters are orthogonal at training time, ensuring that the interference between tasks vanishes completely.

Continual learning for image restoration. Image restoration exposes the tradeoff between knowledge sharing and task isolation directly because a single model must handle many different degradations. All-in-one methods share a restoration backbone and distinguish tasks using encoders (Li et al., 2020), queries (Valanarasu et al., 2022), contrastive embeddings (Li et al., 2022; Luo et al., 2024), prompts (Potlapalli et al., 2023), language instructions (Conde et al., 2024), or expert routing (Zamfir et al., 2025). Diffusion-based restoration methods use similar shared formulations (Özdenizci & Legenstein, 2023). However, these methods generally assume that all tasks are available during training and, therefore, sequentially adding new tasks changes shared parameters and causes catastrophic forgetting (Wu et al., 2025). Continual restoration methods address this using expert routing (Liu et al., 2020), replay (Cheng et al., 2024), regularization (Kartheek et al., 2025), multi-head adapters (Mandal et al., 2025), or isolated adaptation paths (Ashrafee & Krawczyk, 2026). However, they suffer from the same tradeoff in which sharing parameters risks interference, while isolating tasks causes parameters to grow linearly with tasks. COLORS breaks this tradeoff for the first time by sharing consolidated directions across tasks while keeping new task-specific information orthogonal to previously learned functions. Our exact shared subspace with orthogonal private components provides sublinear parameter scaling and composable multi-task control with guaranteed zero forgetting.

C Implementation Details

Our full method described in Sec. 2 is presented in Algorithm 1.

C.1 Backbones and adapted layers

Every backbone, autoencoder and text encoder is frozen, and adapters are placed only on attention projections. In SD 1.5 and Sana, self-attention and text cross-attention are separate, so only self-attention (`attn1`: query, key, value and output projections) is adapted and cross-attention is left untouched. This covers the 16 transformer blocks of the SD 1.5 UNet (6 down, 1 middle, 9 up), all 20 blocks of Sana-1600M and all 60 of Sana-4.8B. In SD3.5-M and FLUX.1-dev, image and text tokens are processed in two streams that attend to each other jointly, and no separate cross-attention exists. The query, key, value and output projections of both streams are adapted in every joint block (24 in SD3.5-M, 19 double blocks in FLUX.1-dev), except the text output projection of the last

Algorithm 1 Continual learning with append-only consolidation (one layer; all layers independent)

Require: frozen W_0; tasks $\mathcal{T}_1, \mathcal{T}_2, \ldots$ arriving sequentially; $c_0, c_{\max}, \tau, \eta, \rho_{\min}$

$\mathcal{S} \leftarrow \varnothing$; $\rho_1 \leftarrow c_0$
for $t = 1, 2, \ldots$ **do**
  if $t = 1$ **then**
    train plain LoRA D_1 (rank ρ_1) on task 1; $\mathcal{S} \leftarrow$ top-c_0 SVD atoms of D_1; store (γ_1, R_1)
  else
    train γ_t on frozen $\mathcal{S}$ + rank-ρ_t residual, GS-orthogonal to $\mathcal{S} \cup \{R_{<t}\}$ in rank space ▷ Eqs. (4)–(6)
    store (γ_t, R_t) exactly (SVD factors, tail energy $\leq \varepsilon$)
  end if
  $\bar{R}_\perp \leftarrow \mathrm{GS}(\bar{R} \perp \mathcal{S})$; extract leading singular pair (u, v)
  if (u, v) explains $e \geq \tau$ **and** $|\mathcal{S}| < c_{\max}$ **then**
    append $uv^\top$ to $\mathcal{S}$; re-project all R_s ▷ exact, Eq. (8)
  end if
  $\rho_{t+1} \leftarrow$ adaptive rank ▷ Eq. (9)
end for

SD3.5-M block, which does not exist. The image-only self-attention (`attn2`) in the first 13 SD3.5-M blocks is adapted the same way. In the 38 single blocks of FLUX.1-dev, attention is computed over the concatenated image and text tokens, and the query, key and value projections are adapted. In PixelGen, image tokens attend to keys and values from both the image and the text. Its fused image query, key and value projection, its text key and value projection and its output projection are adapted in all 16 blocks. Table 4 gives the resulting adapter budgets and training cost for each backbone. The video backbone is described in its own section.

Table 4: Backbones, adapter budget and training cost of the 16-task streams: SD 1.5 (Rombach et al., 2022), Sana-1600M (Xie et al., 2025a) with the DC-AE autoencoder (Chen et al., 2025), SD3.5-M (Esser et al., 2024), Sana-4.8B (Xie et al., 2025b), FLUX.1-dev (Black Forest Labs, 2024) and PixelGen (Ma et al., 2026). "Per rank" counts the parameters that one unit of adapter rank adds over all adapted layers. r_0 is the first-task rank, c_0 the initial shared basis and $c_{\max}$ its cap. Hours are wall clock for the whole stream on one A100 80GB and GB is peak memory, including sample grids, consolidation and checkpointing.

backbone	params	space	layers	per rank	$r_0/c_0/c_{\max}$	steps	hours	GB
SD 1.5	0.9B	4-ch VAE	64	0.10M	16/16/24	25k	36	57.1
Sana-1600M	1.6B	DC-AE 32×	80	0.36M	16/16/24	25k	45	49.0
SD3.5-M	2.5B	16-ch VAE	243	0.75M	32/32/48	40k	166	54.6
Sana-4.8B	4.8B	DC-AE 32×	240	1.08M	32/32/48	30k	192	32.2
FLUX.1-dev	12B	16-ch VAE	266	1.63M	16/16/24	20k	150	44.0
PixelGen	1.1B	pixel	48	0.22M	16/16/24	15k	11*	11.5

* Stages 9 to 16 ran on an RTX 6000. The 4-, 8-, 12- and 16-task Sana streams of main-paper Table 2 (main) train 20k steps per task.

The first task trains at rank r_0 and consolidation absorbs it whole, so $c_0 = r_0$. Later tasks shrink to the rank their residual spectrum needs, down to $\rho_{\min} = 4$. A late rank-4 task on SD3.5 thus trains 3.0M parameters, against 23.9M for a rank-32 LoRA on the same layers.

C.2 Data

Table 5 lists the 16 tasks used in the evaluation. Each source gets a deterministic hash split of 95% train and 5% validation. Synthetic pairs degrade a clean image at load time with fresh randomness every epoch. The loader takes the largest square crop of each pair, random in training and centered in validation, and resizes it to 512^2. Degraded and clean images share the crop. Every backbone and every baseline sees the same pairs.

All streams use a fixed order, set before the first run. The shorter streams of main-paper Table 2 are its subset of length 4, 8 and 12. Each task has a set of real prompts, embedded once per backbone so

Table 5: The 16 tasks in stream order. "Val" counts full-validation images. Synthetic degradations: Gaussian ($\sigma = 0.2 \cdot 255$) or Poisson ($\lambda \sim \mathcal{U}[0, 15]$) noise, JPEG quality in $[30, 99]$, grayscale, and old photos (sepia or desaturation blended with MLRN scratch and stain textures).

#	task	sources	kind	content	train	val
1	haze	RESIDE Li et al. (2019) (SOTS, OTS) CDD-11 Guo et al. (2024b)	real	indoor, outdoor	70.6k	100
2	rain streaks	4K-Rain13k Chen et al. (2026) CDD-11 Guo et al. (2024b)	real	outdoor	13.5k	100
3	snow	SRRS-2021 Chen et al. (2023) CDD-11 Guo et al. (2024b)	real	outdoor	22.8k	100
4	raindrops	Raindrop Clarity Jin et al. (2024) day and night	real	outdoor	12.7k	100
5	shadow	SRD Guo et al. (2024a)	real	natural scenes	2.9k	100
6	denoise	DIV2K Agustsson & Timofte (2017b)	synth.	natural scenes	0.76k	37
7	motion blur	RHM Zhang et al. (2023)	real	dynamic scenes	83.1k	100
8	over-exposure	exposure corr. Afifi et al. (2021) LDR–HDR	real	natural scenes	7.2k	100
9	defocus	LSD Zhang et al. (2023)	real	natural scenes	5.3k	100
10	JPEG	Flickr2K Agustsson & Timofte (2017a)	synth.	natural scenes	0.95k	54
11	real SR	RealSR V3 ($2\times, 3\times, 4\times$) Cai et al. (2019)	real	natural scenes	1.5k	71
12	colorization	CelebA Liu et al. (2015) DIV2K Agustsson & Timofte (2017b) Flickr2K Agustsson & Timofte (2017a)	synth.	faces, scenes	2.4k	100
13	low-light	LOL Wei et al. (2018) CDD-11 Guo et al. (2024b)	real	indoor, outdoor	1.6k	86
14	old photo	CelebA Liu et al. (2015) DIV2K Agustsson & Timofte (2017b) MLRN textures Hao (2023)	synth.	faces, scenes	3.9k	100
15	face inpainting	CelebA Liu et al. (2015)	real	faces	28.4k	100
16	extreme dark	SID Sony Chen et al. (2018)	real	raw captures	2.3k	100

that no text encoder runs in training. Training samples a prompt uniformly and evaluation uses one fixed prompt per task.

C.3 The bridge objective

Let x^{cl} and x^{de} be the latents of a clean and a degraded image. We draw $s \sim \mathcal{U}[0, 1]$ and train

$$x_s = (1 - s)\, x^{\mathrm{cl}} + s\, x^{\mathrm{de}}, \qquad \mathcal{L} = \mathbb{E} \left\| f_\theta(x_s,\, s,\, \tau_t) - \left(x^{\mathrm{de}} - x^{\mathrm{cl}}\right) \right\|_2^2, \tag{11}$$

where τ_t is the prompt of task t and the backbone receives $1000s$ as its timestep. Inference starts at the degraded latent and takes 16 Euler steps to $s = 0$ without noise. This is paired bridge matching in its deterministic limit, the $\sigma \to 0$ case of a Brownian bridge between fixed endpoints Liu et al. (2023); Zhou et al. (2024); Albergo et al. (2025).



Figure 9: Flow matching from noise against our bridge, same FLUX.1-dev backbone and validation images. Left: haze. Right: low-light. The figure shows the failure mode of flow matching.

Why not standard flow matching. Our first adapters generated the clean latent from Gaussian noise, conditioned on the degraded latent. They must rebuild every pixel, so outputs wash out, invent texture or stay dark, and PSNR stays near or below the input (Fig. 9). The bridge starts from the degraded image and learns only the short path to the clean one. The scene always stays in place and overall PSNR increases by 14 and 18 dB on these images.

C.4 SETTINGS AND EVALUATION

Table 6 provides hyper-parameter details used for training all backbones and baseline methods.

Table 6: Settings shared by all methods and backbones unless noted.

optimizer	AdamW, lr 10^{-4}, clip 1.0	$r_0 = c_0$	16 (32 on SD3.5-M, Sana-4.8B)
batch	1 pair at 512^2	$c_{\max}$	24 (48 on the same two)
precision	bf16, fp32 master weights	τ	0.1, ≤ 1 new atom per task
consolidation	fp64	η	0.99
seed	0 (1, 2 for the seed study)	$\rho_{\min}$	4
inference	16 Euler steps, $\alpha_t = 1$	steps	Table 4

C.5 BASELINES

Table 7: The methods compared in main-paper Table 2. All of them train the bridge objective (Eq. 11) on the same data in the same task order. Baselines train 15k steps per task on Sana and 40k on SD3.5-M. Our step counts are listed in Table 4. Each method is run the way it was designed, as described below. Prefix-sequential GS and naive sequential results at k tasks are read from the 16-task run after task k. The joint oracle and PCA-recompute are retrained for each stream length.

Method	How it works and how it is run
Ours	Each task is stored as coefficients on the shared atoms plus its own residual. At inference only that task's adapter is merged into the frozen weights.
Independent adapters	A separate rank-16 LoRA is trained from the base model for every task. Nothing is shared, and each task runs with its own adapter.
Prefix-sequential GS	Each task gets its own adapter, made orthogonal to all earlier adapters by Gram–Schmidt, with no shared atoms. Because it was trained on top of them, task k only works with adapters 1 to k switched on. Run as a single model with every adapter on, it falls to 16.87, 15.41, 13.31 and 8.19 dB at 4, 8, 12 and 16 tasks.
PCA-recompute	Our reimplementation of Share Kaushik et al. (2026). The shared basis is rebuilt by PCA after every task, so earlier tasks are re-expressed in the new basis and drift.
O-LoRA Wang et al. (2023)	Sequential rank-16 adapters. Earlier adapters are frozen and stay on, and a penalty ($\lambda = 0.5$) pushes each new adapter towards orthogonality.
InfLoRA Liang & Li (2024)	Each new down-projection is fixed orthogonal to the earlier ones and only the up-projection is trained. All adapters stay on.
Naive sequential	A single adapter is fine-tuned on one task after another.
Joint oracle	A single adapter is trained on all tasks at once with the same total budget. It is not continual and serves as a reference.
Degraded input	The degraded image itself, scored against the ground truth.

Retrained all-in-one models. OneRestore, InstructIR, MoCE-IR, BioIR and AdaIR are trained from scratch on our data with their own architecture, loss and crop size, using AdamW at 0.9×10^{-4}, weight decay 10^{-4}, gradient accumulation 4 and clipping 0.9. The sequential version is trained with 15k steps per task and the joint version the same total on all tasks mixed. Conditioned models receive oracle per-task CLIP text embeddings from our prompt set, which favors them. After every stage we evaluate all seen tasks on 50 unseen images.

Verification gates. Zero forgetting requires that every task is stored exactly and that no later task can change it. Stored residuals are truncated SVDs kept in floating point, so three properties are checked in full 64-bit precision after every consolidation stage: exact re-expression of the new task, orthonormality of the atoms, and orthogonality of every residual to the atoms. Exact re-expression requires the new task, rebuilt from its coefficients and residual, to match the trained adapter to a relative error below 10^{-5}, and the run is stopped otherwise. Orthonormality requires the inner products of the atoms to deviate from the identity by less than 10^{-6}. Orthogonality requires the projection of each residual onto the atoms, relative to the task's update, to stay below 10^{-5}. The measured worst values over all 14 runs are 3.1×10^{-6}, 2.5×10^{-8} and 3.5×10^{-6}, and every run passes all three checks at every stage.

C.6 COMPUTE AND STORAGE

Figure 10 looks at what our design costs during training, at inference and in storage.

Training. Keeping a new task orthogonal to everything already stored costs the same whether we have stored two tasks or thirty-two. We compute it in the small space spanned by the adapters' ranks instead of the full weight space, so it takes 3.2 to 3.7 ms per layer from 2 up to 32 stored tasks, while the naive recursive version already needs 295 ms at 12 (panel a). Concretely, with m stored references of rank r on an $o \times i$ layer and T tokens, we cache an $m \times m$ Gram matrix once per stage, and each training step then costs $O(mr^2(o+i)) + O(T(o+i)mr)$ operations, without ever forming an $o \times i$ matrix. Over a whole stream our time per stage stays flat, while prefix-sequential GS slows down fivefold because it has to run every earlier adapter in the forward pass (panel b). Shrinking the rank helps as well: on FLUX.1-dev a training step stays near 1.0 s for the whole stream, against 2.1 s by task 16 if every task keeps its full rank (Note: Inference time reported also include pass through a massive diffusion backbone each time.).

Inference. Panel c compares four ways of running one restoration task on SD3.5.

- *Merged* is how we deploy. The task's update is added into the frozen weights once, so sampling costs exactly as much as the original model, 0.94 s for 16 steps.
- *Unmerged* runs our shared and private parts as two extra low-rank branches instead. It gives the same output and takes 1.52 s.
- *LoRA* is a single ordinary rank-32 adapter kept as a branch, at 1.26 s.
- *All on* keeps all 16 stored adapters active at once, which is how O-LoRA, InfLoRA and prefix-sequential GS have to run. It takes 5.08 s, over five times slower than ours.

Storage. Consolidation also keeps the model small (panel d). The first task is absorbed into 32 shared atoms per layer, and every later task stores only its coefficients on those atoms plus a residual whose rank shrinks from 32 to 4. After 16 SD3.5 tasks this comes to 108M parameters, 30M for the shared atoms and 78M for the residuals. Sixteen separate rank-32 LoRAs would need 382M and sixteen full fine-tunes 9.2B, so we store the stream $3.5\times$ more compactly than independent adapters and $85\times$ more compactly than full fine-tuning, and the gap grows with every task. Share, in contrast, grows in memory as it rebuilds its basis and runs out of host memory on the 16-task SD3.5 stream (Figure 6). One practical detail: finished stages must move their cached Gram matrices off the GPU, otherwise a 16-task run fills 73 GB of caches by stage 10.

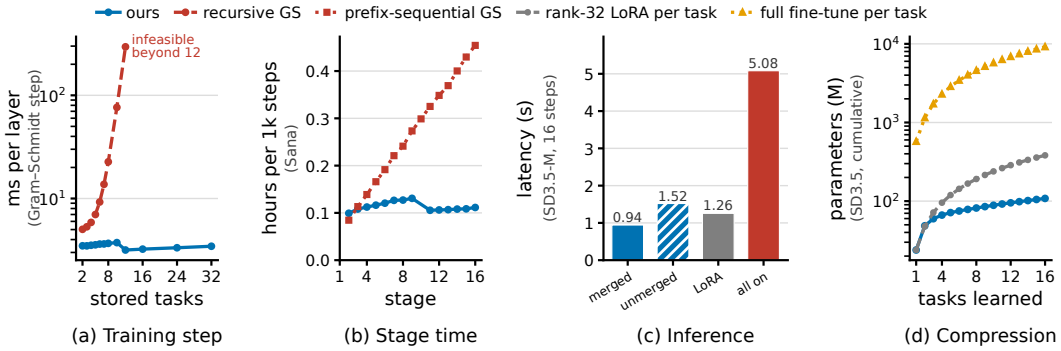

Figure 10: Cost of our design. (a) Time of the orthogonalization step per layer as more tasks are stored. (b) Training time per 1k steps at each stage on Sana. (c) Time to restore one image on SD3.5 in the four settings described in the text. (d) Parameters stored as the SD3.5 stream grows.

D Formal Proofs and Theoretical Setup

This section gives the setup and proofs for Section 4 of the main paper, stating each standard result briefly where it is used.

D.1 Setup and Assumptions

We formalize continual learning as the adaptation of a pre-trained generative backbone through sequential low-rank updates. We analyze a single target layer with a frozen base weight matrix $W_0 \in \mathbb{R}^{o\times i}$, holding all other layers fixed; the same argument applies to each adapted layer in turn. Adapters are $o \times i$ matrices, compared with the Frobenius inner product $\langle X,\, Y\rangle_F = \operatorname{tr}(X^\top Y)$ and its norm $\|X\|_F$. We write $\|\cdot\|_2$ for the Euclidean norm of vectors, $P_{\mathcal{V}}$ for the orthogonal projection onto a linear subspace $\mathcal{V} \subseteq \mathbb{R}^{o\times i}$, $T_\rho(M)$ for the best rank-ρ approximation of a matrix M with singular values $\sigma_1(M) \geq \sigma_2(M) \geq \ldots$, and $\operatorname{tail}_\rho(M) = \big(\sum_{j>\rho} \sigma_j(M)^2\big)^{1/2}$ for what that approximation leaves out.

Suppose the network has processed $t-1$ tasks. Its memory consists of c orthonormal atoms spanning the shared subspace $\mathcal{S} = \operatorname{span}\{A_1, \ldots, A_c\}$ and the stored residuals spanning $\mathcal{Q}_t = \operatorname{span}\{R_s\}_{s<t}$. Writing $S\gamma = \sum_j \gamma_j A_j$, task s is stored as $S\gamma_s + R_s$. Each residual is made orthogonal to the atoms and to the earlier residuals when it is formed. Later atom appends subtract $\langle R,\, A\rangle_F\, A$ from every stored residual, which keeps each residual orthogonal to the atoms but lets residuals of different tasks overlap slightly, so they need not stay mutually orthogonal. The storage truncation below perturbs orthogonality to the atoms only at the storage floor (measured leakage at most 3.5×10^{-6}), and the capacity analysis treats it as exact. A new adapter may use the atoms but must avoid the stored residuals, so its private part lives in the *free complement* $\mathcal{W}_t = (\mathcal{S} \oplus \mathcal{Q}_t)^\perp$. Since $\mathcal{Q}_t$ is spanned by $t-1$ matrices, $K_t = \dim \mathcal{Q}_t \leq t-1$.

Task t provides S_t i.i.d. samples from its data distribution $\mathcal{D}_t$. We evaluate an adapter D by its *population risk* $\mathcal{R}(D) = \mathbb{E}_{\mathcal{D}_t}\, \ell\big((W_0{+}D)x, y\big)$, the expected loss over the true data distribution, and train on its finite-sample counterpart, the empirical risk $\widehat{\mathcal{R}}(D)$. We write h_D for the predictor $x \mapsto (W_0 + D)x$ and $\mathcal{R}(h_D) = \mathcal{R}(D)$. The trainable parameters are the coefficients $\gamma \in \mathbb{R}^c$ on the atoms and a rank-ρ_t factor d, the product of the two LoRA matrices, which the Gram–Schmidt step projects onto the free complement before it enters the forward pass. Given a norm bound $b > 0$, continual learning is empirical risk minimization over the constrained hypothesis class

$$\mathcal{H}_t = \Big\{\, h_D,\;\; D = S\gamma + P_{\mathcal{W}_t}(d) \;:\; \operatorname{rank}(d) \leq \rho_t,\;\; \|\gamma\|_2^2 + \|d\|_F^2 \leq b^2 \,\Big\}. \tag{12}$$

The rank limit applies to the trained factor d; the stored residual $P_{\mathcal{W}_t}(d)$ can have higher rank. Because $S\gamma \perp P_{\mathcal{W}_t}(d)$, the atoms are orthonormal and projections do not increase norms, every adapter in $\mathcal{H}_t$ has $\|D\|_F \leq b$.

Assumption A1 (Regularity)**.** *The loss $\ell(\cdot, y)$ is L-Lipschitz in the network prediction and takes values in $[0, \ell_{\max}]$, and inputs satisfy $\|x\|_2 \leq 1$.*

This makes the risk Lipschitz in the adapter. For any adapters D, D',

$$\big|\ell\big((W_0{+}D)x, y\big) - \ell\big((W_0{+}D')x, y\big)\big| \leq L\,\|(D - D')x\|_2 \leq L\,\|D - D'\|_F\,, \tag{13}$$

because the spectral norm is at most the Frobenius norm. Taking expectations, $|\mathcal{R}(D) - \mathcal{R}(D')| \leq L\,\|D - D'\|_F$, and the same holds for $\widehat{\mathcal{R}}$.

Assumption A2 (Oracle adapter)**.** *Task t has an oracle adapter D_t^* with $\|D_t^*\|_F \leq b$ and $\nabla\mathcal{R}(D_t^*) = 0$, inducing the predictor h^*. Its private component is $E_t = D_t^* - P_{\mathcal{S}}(D_t^*)$, everything outside the atom span, including any part that falls into $\mathcal{Q}_t$. Its overlap with the stored residuals is $\beta_t = \|P_{\mathcal{Q}_t} D_t^*\|_F \,/\, \|D_t^*\|_F$.*

Assumption A3 (Restricted strong convexity)**.** *The population risk is λ-strongly convex on the subspace $\mathcal{S} \oplus \mathcal{W}_t \oplus \operatorname{span}\{D_t^*\}$, which contains every feasible adapter and the oracle: for D in it, $\mathcal{R}(D) - \mathcal{R}(D_t^*) \geq \frac{\lambda}{2}\,\|D - D_t^*\|_F^2$. For our quadratic bridge objective, λ is a measurable property of each layer's data rather than a modeling assumption (median 4.5×10^{-3} across layers), and the bounds below hold per layer with that layer's λ.*

This is the quadratic lower bound of a strongly convex function (Nesterov, 2018, Thm. 2.1.8), whose linear term vanishes because $\nabla\mathcal{R}(D_t^*) = 0$.

D.2 Error Analysis: Exactness and Zero Forgetting

The Eckart–Young–Mirsky theorem (Eckart & Young, 1936; Mirsky, 1960) states that $T_\rho(M)$ is the closest matrix of rank at most ρ to M in Frobenius norm, at distance exactly $\mathrm{tail}_\rho(M)$.

Proposition 1 (Zero forgetting). *Let D_t be task t's adapter as trained and $\widehat{D}_t^{(n)}$ its stored representation after task $n \geq t$. If the storage truncation discards at most a fraction ε of the residual's energy and ν_t atoms are appended after task t, then $\left\|\widehat{D}_t^{(n)} - D_t\right\|_F \leq (1+\nu_t)\sqrt{\varepsilon}\,\|D_t\|_F$ for every $n \geq t$.*

Proof of Proposition 1. At storage time, task t's residual R_t is factorized by a truncated SVD that discards a tail with at most a fraction ε of $\|R_t\|_F^2$. By Eckart–Young–Mirsky this changes the residual by at most $\sqrt{\varepsilon}\,\|R_t\|_F$, and $\|R_t\|_F \leq \|D_t\|_F$ because R_t is orthogonal to $S\gamma_t$. So the first stored adapter is within $\sqrt{\varepsilon}\,\|D_t\|_F$ of D_t.

Afterwards, task t's state changes only when consolidation appends a new unit atom A. The update is $R \mapsto R - \langle R,\, A\rangle_F\, A$ and $\gamma \mapsto (\gamma, \langle R,\, A\rangle_F)$, so the atom takes over exactly the component of R along A and the reconstructed adapter does not move:

$$S'\gamma' + R' = S\gamma + \langle R,\, A\rangle_F\, A + (R - \langle R,\, A\rangle_F\, A) = S\gamma + R.$$

The new residual is then stored again with the same truncation budget, which moves the adapter by at most $\sqrt{\varepsilon}$ times the residual's norm. Both the projection and the truncation shrink the residual, so its norm stays at most $\|R_t\|_F \leq \|D_t\|_F$, and each append adds at most $\sqrt{\varepsilon}\,\|D_t\|_F$ to the error. After ν_t appends the error is at most $(1+\nu_t)\sqrt{\varepsilon}\,\|D_t\|_F$. Since at most $c_{\max} - c_0$ atoms are ever appended, the bound holds for every later task, independent of stream length and arrival order. With $\varepsilon = 10^{-11}$ and $c_{\max} - c_0 \leq 16$ in all our runs it stays below 6×10^{-5}, and the measured error is at most 3.1×10^{-6}. □

D.3 Capacity and Learnability Proofs

Proposition S1 (Capacity bounds). *Every adapter in $\mathcal{H}_t$ is at distance at least $\|P_{\mathcal{Q}_t} D_t^*\|_F$ from the oracle, and some adapter in $\mathcal{H}_t$ is at distance at most $\left(\|P_{\mathcal{Q}_t} D_t^*\|_F^2 + \mathrm{tail}_{\rho_t}(P_{\mathcal{W}_t} D_t^*)^2\right)^{1/2}$.*

Proof of Proposition S1. The subspaces $\mathcal{S}$, $\mathcal{Q}_t$ and $\mathcal{W}_t$ are mutually orthogonal and together span $\mathbb{R}^{o\times i}$, so the oracle splits as $D_t^* = D_S + D_Q + D_W$ with $D_S = P_{\mathcal{S}} D_t^*$, $D_Q = P_{\mathcal{Q}_t} D_t^*$ and $D_W = P_{\mathcal{W}_t} D_t^*$.

For the upper bound we build one feasible adapter. We pick γ with $S\gamma = D_S$, possible because the atoms are an orthonormal basis of $\mathcal{S}$, and the factor $d = T_{\rho_t}(D_W)$. This candidate lies in $\mathcal{H}_t$: d has rank at most ρ_t, and $\|\gamma\|_2^2 + \|d\|_F^2 \leq \|D_S\|_F^2 + \|D_W\|_F^2 \leq \|D_t^*\|_F^2 \leq b^2$, using that truncating singular values cannot increase the norm and Assumption A2. Its $\mathcal{S}$ part is exact and it has no $\mathcal{Q}_t$ part, so

$$\|D - D_t^*\|_F^2 = \|D_Q\|_F^2 + \left\|P_{\mathcal{W}_t}\left(D_W - T_{\rho_t}(D_W)\right)\right\|_F^2 \leq \|D_Q\|_F^2 + \mathrm{tail}_{\rho_t}(D_W)^2,$$

where the equality uses $D_W \in \mathcal{W}_t$ and the inequality that projections do not increase norms, followed by Eckart–Young–Mirsky.

For the lower bound, every adapter in $\mathcal{H}_t$ lies in $\mathcal{S} \oplus \mathcal{W}_t$ and has no component in $\mathcal{Q}_t$, so $\|D - D_t^*\|_F \geq \|P_{\mathcal{Q}_t}(D - D_t^*)\|_F = \|D_Q\|_F$. □

Theorem B (Capacity loss under coherence). *Under Assumption A3, every $h \in \mathcal{H}_t$ satisfies $\mathcal{R}(h) - \mathcal{R}(h^*) \geq \frac{\lambda}{2}\,\beta_t^2\,\|D_t^*\|_F^2$.*

Proof of Theorem B. Let $h = h_D$. By the lower bound of Proposition S1, $\|D - D_t^*\|_F \geq \|P_{\mathcal{Q}_t} D_t^*\|_F = \beta_t\,\|D_t^*\|_F$. Both D and D_t^* lie in the subspace of Assumption A3 and $\nabla\mathcal{R}(D_t^*) = 0$ by Assumption A2, so $\mathcal{R}(h) - \mathcal{R}(h^*) \geq \frac{\lambda}{2}\,\|D - D_t^*\|_F^2 \geq \frac{\lambda}{2}\beta_t^2\,\|D_t^*\|_F^2$. □

We now bound the overlap when tasks are incoherent. Let $\widehat{R}_s = R_s / \|R_s\|_F$. We say the stream is *$\mu$-incoherent at task t* if $|\left\langle \widehat{R}_s,\, \widehat{R}_{s'} \right\rangle_F| \le \mu$ for all $s \neq s' < t$ and $|\left\langle \widehat{R}_s,\, E_t \right\rangle_F| \le \mu \|E_t\|_F$ for all $s < t$. This is the incoherence of the main paper stated for the stored residuals, which are exactly the directions the constraint locks. Since residuals are orthogonal to the atoms, $\left\langle \widehat{R}_s,\, D_t^* \right\rangle_F = \left\langle \widehat{R}_s,\, E_t \right\rangle_F$. The proof uses Gershgorin's circle theorem (Horn & Johnson, 2012, Thm. 6.1.1): every eigenvalue of a square matrix G lies within $\sum_{k \neq j} |G_{jk}|$ of some diagonal entry G_{jj}.

Proposition S2 (Overlap under incoherence). *Let G_t be the Gram matrix of the normalized stored residuals. If the stream is μ-incoherent at task t and $\lambda_{\min}(G_t) > 0$, then*

$$\|P_{\mathcal{Q}_t} D_t^*\|_F \le \frac{\mu\sqrt{K_t}}{\sqrt{\lambda_{\min}(G_t)}}\, \|E_t\|_F \le \frac{\mu\sqrt{K_t}}{\sqrt{1-\mu(K_t-1)}}\, \|E_t\|_F\,,$$

where the second inequality requires $\mu(K_t - 1) < 1$.

Proof of Proposition S2. Choose K_t linearly independent normalized residuals $u_1, \dots, u_{K_t}$ among the $\widehat{R}_s$ that span $\mathcal{Q}_t$, and let G_t be their Gram matrix, $(G_t)_{jk} = \langle u_j,\, u_k \rangle_F$. With $w_j = \langle u_j,\, D_t^* \rangle_F$, the projection onto the span of a non-orthonormal basis is $\|P_{\mathcal{Q}_t} D_t^*\|_F^2 = w^\top G_t^{-1} w$. Incoherence gives $|w_j| \le \mu \|E_t\|_F$, so $\|w\|_2^2 \le K_t \mu^2 \|E_t\|_F^2$ and

$$\|P_{\mathcal{Q}_t} D_t^*\|_F^2 \le \frac{\|w\|_2^2}{\lambda_{\min}(G_t)} \le \frac{K_t\, \mu^2\, \|E_t\|_F^2}{\lambda_{\min}(G_t)}.$$

For the second form, G_t has unit diagonal and off-diagonal entries at most μ in absolute value, so Gershgorin gives $\lambda_{\min}(G_t) \ge 1 - \mu(K_t - 1)$. Taking square roots gives both claims. □

The bound increases with K_t, so it also holds with K_t replaced by any larger count such as $\dim(\mathcal{S} \oplus \mathcal{Q}_t)$, the form used in the main paper. In our streams $\lambda_{\min}(G_t) \ge 0.34$ in every layer. Incoherence is also the typical case: a uniformly random direction in the $m = o \cdot i$ dimensions of a layer puts an expected fraction $K_t/m \le 1.5 \times 10^{-4}$ of its energy into $\mathcal{Q}_t$ in every layer we adapt (Vershynin, 2018, Ch. 5).

For the estimation error we use two standard tools. Hoeffding's inequality states that for a fixed adapter, with loss values in $[0, \ell_{\max}]$, $\Pr\left(|\mathcal{R}(D) - \widehat{\mathcal{R}}(D)| > \xi\right) \le 2\exp(-2S_t\xi^2/\ell_{\max}^2)$. To apply it to the whole class at once we cover it by finitely many points. Candès and Plan (Candès & Plan, 2011, Lemma 3.1) show that $o \times i$ matrices of rank at most ρ and unit Frobenius norm admit an η-net of at most $(9/\eta)^{(o+i+1)\rho}$ points, and the same volumetric argument covers those of norm at most b with $(9b/\eta)^{(o+i+1)\rho}$ points. A ball of radius b in $\mathbb{R}^c$ needs at most $(3b/\eta)^c$ points (Vershynin, 2018, Cor. 4.2.13).

Theorem A (Learning under incoherence). *Under Assumptions A1 and A2, let $\hat{h}$ minimize the empirical risk over $\mathcal{H}_t$ and let $p_t = c + (o + i + 1)\rho_t$. If the stream is μ-incoherent at task t and $\lambda_{\min}(G_t) > 0$, then with probability at least $1 - \delta$*

$$\mathcal{R}(\hat{h}) - \mathcal{R}(h^*) \le \underbrace{\frac{8Lb}{S_t} + 2\ell_{\max}\sqrt{\frac{p_t \ln(9S_t) + \ln(2/\delta)}{2S_t}}}_{\text{estimation}} + \underbrace{L\Big(\frac{\mu\sqrt{K_t}}{\sqrt{\lambda_{\min}(G_t)}}\, \|E_t\|_F + \operatorname{tail}_{\rho_t}(P_{\mathcal{W}_t} D_t^*)\Big)}_{\text{approximation}}.$$

Proof of Theorem A. Let $h_\circ \in \arg\min_{h \in \mathcal{H}_t} \mathcal{R}(h)$ be the best predictor in the class, which exists because the class is compact, and split the excess risk as $\mathcal{R}(\hat{h}) - \mathcal{R}(h^*) = [\mathcal{R}(\hat{h}) - \mathcal{R}(h_\circ)] + [\mathcal{R}(h_\circ) - \mathcal{R}(h^*)]$.

Estimation. We show that the empirical risk is close to the population risk for every adapter in the class at once, and then use that $\hat{h}$ minimizes the empirical risk. Let $\Delta = \sup_{h \in \mathcal{H}_t} |\mathcal{R}(h) - \widehat{\mathcal{R}}(h)|$.

To control Δ, we cover the class with finitely many adapters. Taking the η-nets above for γ and for d with $\eta = b/S_t$, and noting that $(\gamma, d) \mapsto S\gamma + P_{\mathcal{W}_t}(d)$ moves the adapter by at most $\|\gamma - \gamma'\|_2 + \|d - d'\|_F$, every adapter in $\mathcal{H}_t$ lies within 2η of one of $\mathcal{N} = (3S_t)^c (9S_t)^{(o+i+1)\rho_t}$ net adapters, so $\ln \mathcal{N} \le p_t \ln(9S_t)$. Hoeffding's inequality with a union bound over these $\mathcal{N}$ adapters, solved for the deviation at failure probability δ, shows that with probability at least $1 - \delta$ every net adapter has $|\mathcal{R} - \widehat{\mathcal{R}}| \le \xi = \ell_{\max}\sqrt{(\ln \mathcal{N} + \ln(2/\delta))/(2S_t)}$. Then

$$\begin{aligned}
\mathcal{R}(\hat{h}) - \mathcal{R}(h_\circ) &= [\mathcal{R}(\hat{h}) - \widehat{\mathcal{R}}(\hat{h})] + [\widehat{\mathcal{R}}(\hat{h}) - \widehat{\mathcal{R}}(h_\circ)] \\
&\quad + [\widehat{\mathcal{R}}(h_\circ) - \mathcal{R}(h_\circ)] \\
&\le 2\Delta && (\hat{h} \text{ minimizes } \widehat{\mathcal{R}}) \\
&\le 2(\xi + 4L\eta) && (\text{Eq. (13)}) \\
&\le \frac{8Lb}{S_t} + 2\ell_{\max}\sqrt{\frac{p_t \ln(9S_t) + \ln(2/\delta)}{2S_t}}. && (\text{value of } \eta, \mathcal{N})
\end{aligned}$$

The step using Eq. (13) holds because, for any adapter D in the class and its nearest net adapter D' (within 2η),

$|\mathcal{R}(D) - \widehat{\mathcal{R}}(D)| \le |\mathcal{R}(D) - \mathcal{R}(D')| + |\mathcal{R}(D') - \widehat{\mathcal{R}}(D')| + |\widehat{\mathcal{R}}(D') - \widehat{\mathcal{R}}(D)| \le 2L\eta + \xi + 2L\eta$,

where the outer two terms use Eq. (13) and the middle one the net bound, so $\Delta \le \xi + 4L\eta$.

For $S_t \ge 9$ we have $\ln(9S_t) \le 2 \ln S_t$ and $p_t \ln S_t \ge 1$, so $1/S_t \le \sqrt{p_t \ln S_t / S_t}$ and $\ln(2/\delta) \le \ln(2/\delta)\, p_t \ln S_t$. Together with $p_t = c + (o + i + 1)\rho_t \le 2\big(c + \rho_t(o + i)\big)$, this gives

$$\mathcal{R}(\hat{h}) - \mathcal{R}(h_\circ) \le C\sqrt{\frac{p_t \ln S_t}{S_t}} = \tilde{\mathcal{O}}\Big(\sqrt{\tfrac{c+\rho_t(o+i)}{S_t}}\Big), \qquad C = 8Lb + 2\ell_{\max}\sqrt{1 + \tfrac{1}{2}\ln(2/\delta)},$$

where $\tilde{\mathcal{O}}$ hides the constant C and the factor $\ln S_t$. This is the estimation term of the main paper: it scales with the parameters of a rank-ρ_t adapter plus c coefficients rather than the $o \cdot i$ parameters of unconstrained fine-tuning.

Approximation. Here we ask how close the best adapter in the class can get to the oracle. We compare $h_\circ$ with the candidate D of Proposition S1, whose error splits into a locked part and a truncation part, and use $\sqrt{x^2 + y^2} \le x + y$:

$$\begin{aligned}
\mathcal{R}(h_\circ) - \mathcal{R}(h^*) &\le \mathcal{R}(h_D) - \mathcal{R}(h^*) && (h_\circ \text{ is best in } \mathcal{H}_t) \\
&\le L\,\|D - D_t^*\|_F && (\text{Eq. (13)}) \\
&\le L\Big(\,\|P_{\mathcal{Q}_t} D_t^*\|_F + \mathrm{tail}_{\rho_t}(P_{\mathcal{W}_t} D_t^*)\Big) && (\text{Prop. S1}) \\
&\le L\Big(\frac{\mu\sqrt{K_t}}{\sqrt{\lambda_{\min}(G_t)}}\,\|E_t\|_F + \mathrm{tail}_{\rho_t}(P_{\mathcal{W}_t} D_t^*)\Big). && (\text{Prop. S2})
\end{aligned}$$

The first term is what the orthogonality constraint costs and vanishes as tasks become incoherent. The second is what the rank limit costs and vanishes once ρ_t covers the free part of the oracle. Adding the two bounds proves the theorem. □

D.4 Probe-Faithfulness in the Linearized Regime

The score reading compares short probe adapters instead of fully trained ones. Here we show that, in the linearized regime, the angle between two probes already tracks the angle between the fully trained adapters, and does so more closely the longer the probes train.

Proposition S3 (Probe-faithfulness). *Fix a layer and T tasks whose inputs share one distribution, so that the linearized objectives $\ell_t(v) = \mathbb{E}\|\Phi(x)\, v - y_t\|_2^2$ in $v = \mathrm{vec}(D)$ share the kernel $\Gamma = \mathbb{E}[\Phi^\top \Phi]$ and differ only in their targets. Let v_t^* solve $\Gamma v_t^* = \mathbb{E}[\Phi^\top y_t]$, suppose $\mathcal{V} = \mathrm{span}\{v_1^*, \dots, v_T^*\}$ is invariant under Γ with eigenvalues in $[\kappa_-, \kappa_+]$ there, $\kappa_- > 0$, and let $v_t(\tau)$ be the probe after training time τ by gradient flow from $v = 0$. With $f_\pm = 1 - e^{-\kappa_\pm \tau}$ and $r = f_+/f_- \ge 1$, for every pair of tasks*

$$\big|\cos\angle(v_s(\tau), v_t(\tau)) - \cos\angle(v_s^*, v_t^*)\big| \le 2\big(r^2 - 1\big),$$

and r decreases toward 1 as the probes train longer.

Proof of Proposition S3. Every probe is the same filter applied to its oracle. The gradient of ℓ_t is $2\Gamma(v - v_t^*)$, so gradient flow from 0 gives $v_t(\tau) = F\, v_t^*$ with $F = I - e^{-\Gamma\tau}$ (absorbing the factor 2 into τ). The filter F is the same for every task, and on $\mathcal{V}$ it is symmetric with eigenvalues in $[f_-, f_+]$. If all eigenvalues were equal ($r = 1$), F would only rescale and every angle would be preserved exactly. The bound measures how much the uneven stretching can change an angle.

How much the filter bends an angle. Let a, b be the unit vectors along v_s^* and v_t^*, and write $F^2/f_-^2 = I + H$, where H is symmetric with eigenvalues in $[0, \omega]$ and $\omega = r^2 - 1$. Then

$$\cos\angle(Fa, Fb) = \frac{\langle Fa, Fb\rangle}{\|Fa\|_2\|Fb\|_2} = \frac{a^\top b + a^\top H b}{\zeta},$$

where $\zeta = \sqrt{(1 + a^\top H a)(1 + b^\top H b)}$ lies in $[1, 1 + \omega]$. Subtracting $\cos\angle(a, b) = a^\top b$ gives

$$\big|\cos\angle(Fa, Fb) - a^\top b\big| = \frac{\big|a^\top b\,(1 - \zeta) + a^\top H b\big|}{\zeta} \le \omega + \omega = 2(r^2 - 1),$$

using $|a^\top b| \le 1$, $|1 - \zeta| \le \omega$, $|a^\top H b| \le \omega$ and $\zeta \ge 1$.

Longer probes are more faithful. Since $\frac{d}{d\tau}\ln(1 - e^{-\kappa\tau}) = \phi(\kappa\tau)/\tau$ with the decreasing function $\phi(x) = x/(e^x - 1)$, we get $\frac{d}{d\tau}\ln r = \big(\phi(\kappa_+\tau) - \phi(\kappa_-\tau)\big)/\tau \le 0$, so r does not increase and tends to 1. □

E More Experiments

This section adds perceptual metrics, every continual-learning baseline side by side, the ablations, the pre-flight task-probe validation, training dynamics, strength control and more examples.

E.1 Quality beyond PSNR

On the SD3.5 stream we score our model and the joint oracle on the same 40 validation images per task (Table 8). Ours matches the oracle on PSNR, SSIM and LPIPS, even though the oracle sees all 16 datasets at once and ours learns them one at a time. Table 9 extends this to every baseline on the 8-task Sana stream with six metrics. Ours, the independent adapters and PCA-recompute lead on every metric, ahead of the joint oracle, while the methods that run all adapters together fall far behind, so their collapse in Table 2 holds for perceptual metrics too. Tables 10 and 11 give per-task results for SD3.5 and for all six backbones.

Table 8: SD3.5, 16 tasks: ours against the joint oracle on the same 40 validation images per task, averaged over tasks.

	PSNR↑	SSIM↑	LPIPS↓
Degraded input	18.80	0.650	0.370
Joint oracle	24.16	**0.759**	0.178
Ours	**24.24**	0.756	**0.176**

Table 9: Sana, 8 tasks, full validation sets. Methods that keep all adapters on are marked with †.

	PSNR↑	SSIM↑	LPIPS↓	NIQE↓	BRISQUE↓	MUSIQ↑
Ours	21.75	0.624	0.211	**4.07**	17.0	60.9
Independent adapters	**21.89**	**0.629**	**0.207**	4.12	17.1	61.3
PCA-recompute	21.63	0.622	0.210	**4.07**	**16.8**	**61.8**
Joint oracle	20.96	0.614	0.230	4.22	18.4	60.8
Naive sequential	16.92	0.509	0.392	5.33	29.8	56.3
Prefix-sequential GS†	16.30	0.522	0.364	5.20	27.4	56.8
O-LoRA†	10.44	0.377	0.633	12.9	–	47.5
InfLoRA†	10.17	0.372	0.632	12.2	–	47.9

BRISQUE is undefined on some of the near-constant images that O-LoRA and InfLoRA produce.

Table 10: SD3.5, 16 tasks: our model per task on the full validation sets. FID is given for the four generative tasks only.

Task	n	PSNR↑	SSIM↑	LPIPS↓	NIQE↓	MUSIQ↑	FID↓
Haze removal	100	26.67	0.877	0.048	3.94	63.2	–
Rain streaks	100	28.71	0.786	0.093	3.91	68.6	–
Desnowing	100	26.45	0.884	0.104	4.30	58.1	–
Raindrop (day)	100	18.96	0.445	0.379	4.82	64.3	–
Shadow removal	100	28.20	0.848	0.094	4.31	62.4	–
Denoise	37	26.69	0.771	0.118	3.20	70.3	–
Motion deblur	100	26.97	0.770	0.132	5.56	42.9	–
Over-exposure	100	21.36	0.806	0.123	3.77	63.1	–
Defocus deblur	100	20.76	0.585	0.321	5.68	52.1	–
JPEG correction	54	26.52	0.774	0.087	3.24	71.6	–
Real super-resolution	71	30.11	0.881	0.083	5.83	64.0	–
Colorization	100	24.17	0.889	0.152	4.21	69.2	70.0
Low-light	86	22.85	0.744	0.154	3.61	64.2	–
Old photo	100	21.90	0.821	0.207	4.60	69.0	82.6
Face inpainting	100	24.11	0.863	0.155	5.64	52.3	60.2
Extreme dark relighting	100	17.77	0.471	0.534	6.61	38.2	279.7
Mean		24.51	0.763	0.174	4.57	60.8	–

Table 11: Per-task PSNR (dB) of our model on all six backbones after 16 tasks, grid protocol, tasks in stream order.

Task	SD1.5	Sana	SD3.5-M	Sana-4.8B	FLUX.1-dev	PixelGen
Haze removal	25.95	22.33	27.07	23.27	24.58	22.60
Rain streaks	27.24	24.83	29.48	25.63	29.10	29.51
Desnowing	24.98	22.66	27.44	22.78	25.76	23.51
Raindrop (day)	18.12	17.59	20.35	17.76	19.90	18.74
Shadow removal	24.04	22.38	24.48	22.16	25.31	23.37
Denoise	22.35	21.07	26.25	22.15	26.17	27.38
Motion deblur	27.92	26.05	31.87	26.23	31.35	27.80
Over-exposure	17.31	17.45	18.42	17.35	19.58	17.41
Defocus deblur	18.76	18.15	17.89	18.11	17.38	18.28
JPEG correction	21.98	20.59	25.30	20.74	24.66	26.46
Real super-resolution	27.46	26.78	31.97	27.29	31.14	32.88
Colorization	26.35	25.51	25.78	25.74	26.33	20.94
Low-light	7.73	18.45	19.76	19.87	18.43	17.26
Old photo	18.56	17.47	21.54	18.15	23.43	19.32
Face inpainting	20.20	19.25	22.63	21.80	21.65	21.60
Extreme dark relighting	12.27	16.09	15.67	17.12	17.91	18.05
Mean	21.33	21.04	24.12	21.63	23.92	22.82

E.2 CONTINUAL-LEARNING BASELINES

Figure 11 shows one image per task for every method on the 8-task Sana stream. O-LoRA and InfLoRA turn many images into flat color or noise, naive sequential training and prefix-sequential GS with all adapters on leave the degradation in place, and ours restores the scene as well as the oracles do. Figure 12 follows every task after every stage of the 16-task stream. Each of our columns stays at its arrival quality, while naive sequential training and all five all-in-one models lose earlier tasks as new ones arrive. Those models learn each new task well, at 19 to 38 dB, and reach 23.6 to 25.5 dB on the first 8 tasks when trained jointly, yet they end at 6.3 to 6.7 dB after the 16-task stream. Approximate orthogonality does not prevent this either. O-LoRA pushes the cosine between its adapters down to 4.3×10^{-4} and InfLoRA makes its down-projections orthogonal to 6.4×10^{-9}, yet both lose more than 6 dB when their adapters run together, because the interference lives in the full update, which ours keeps orthogonal to 9.5×10^{-9}. In standard continual-learning terms our backward transfer is exactly zero, and on SD3.5 our final accuracy of 24.12 dB comes within 0.5 dB of what naive sequential training reaches right after learning each task, while the naive learner itself ends at 6.98 dB.

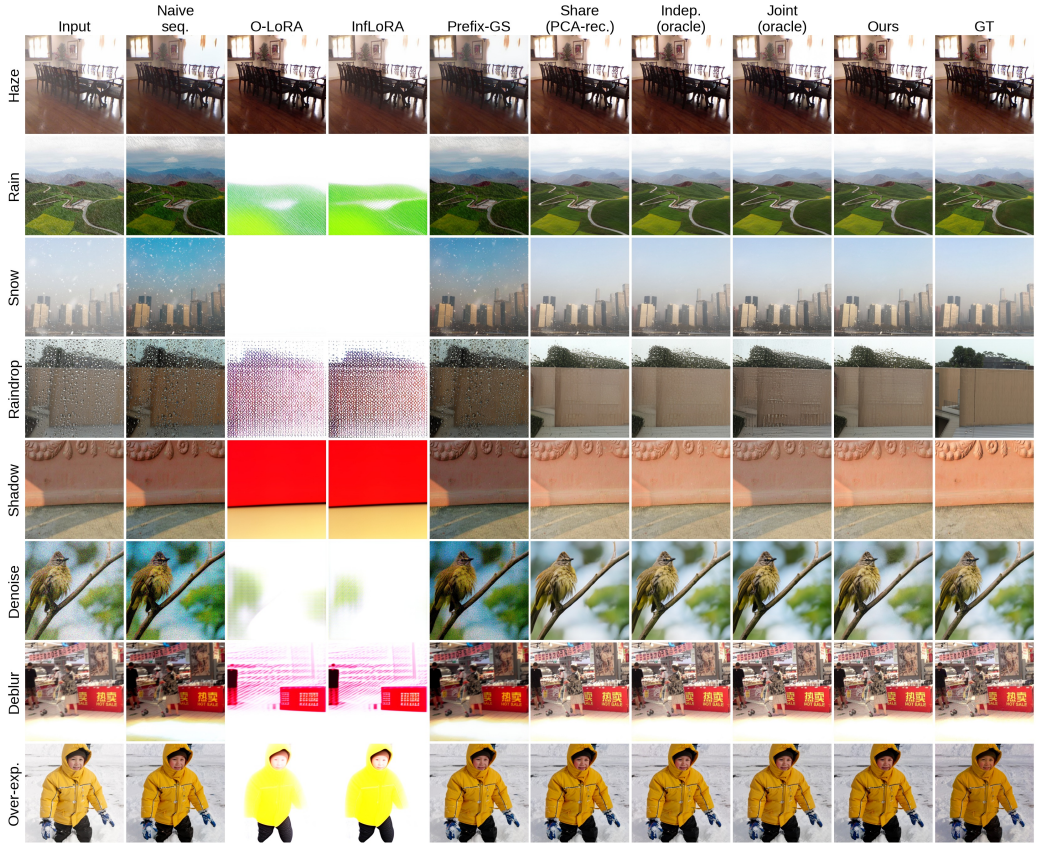

Figure 11: All continual-learning baselines on the 8-task Sana stream, one validation image per task. Prefix-sequential GS is shown with every adapter on.

E.3 ABLATIONS

Table 12 changes one design choice at a time on Sana. Three choices matter. Letting the rank shrink is essential: if every task keeps rank 16, early tasks use up the room that later tasks need, and the stream ends 1 dB below its input instead of 2.4 dB above it. Restricting adapters to self-attention also matters, since adding cross-attention drops the stream to 2 dB below its input. Finally, Gram–Schmidt is what keeps stored tasks apart. Without it the PSNR barely moves, but the residuals overlap the

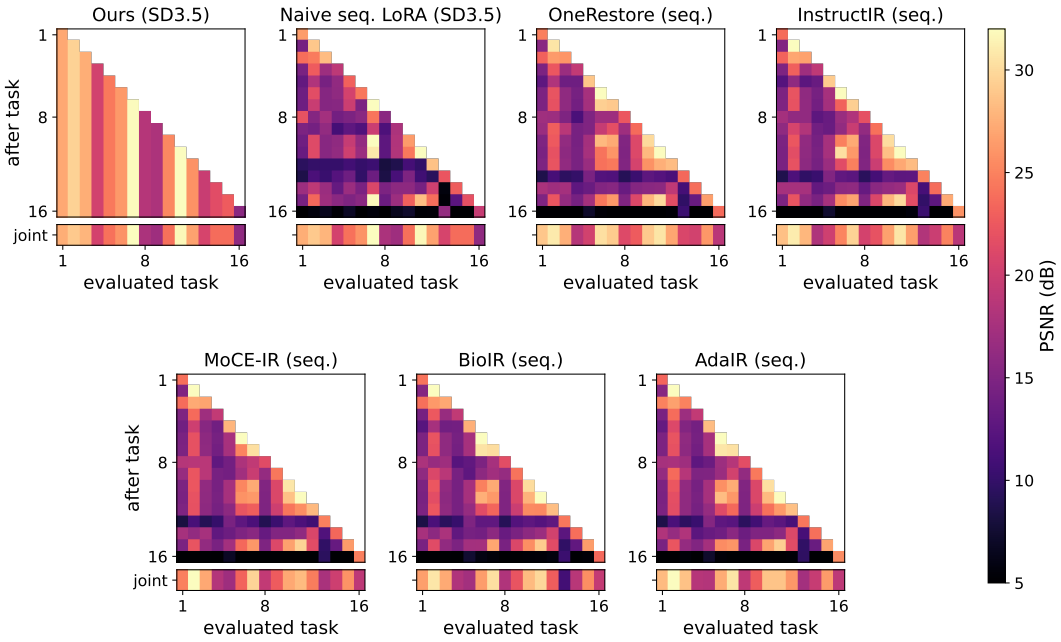

Figure 12: PSNR of every task (columns) after every training stage (rows) on the 16-task stream, with the joint oracle below each map.

shared atoms by 0.77 instead of 3×10^{-6}, and the zero-forgetting guarantee no longer holds. The remaining choices do not matter much. Doubling the shared atom cap from 24 to 48 changes nothing implying unused shared ranks remain available even at the end of the task stream. Training the 16-task stream in reverse order ends at the same 21.04 dB as the forward order. Across three random 16-task orderings used for training, our re-expression error stays at 3.0 to 3.1×10^{-6} every time, while PCA-recompute drifts by 37 to 39% on its worst task and naive sequential training ends anywhere between 14.4 and 17.1 dB.

Table 12: Ablations on Sana, grid protocol. Gain is PSNR over the degraded input of the same images.

Stream	Variant	PSNR	Gain	Residual overlap
16 tasks, 25k steps	Ours	21.04	+2.37	2.9×10^{-6}
	Fixed rank 16	17.40	−0.99	2.5×10^{-6}
8 tasks, 20k steps	Ours	21.95	+0.50	3.0×10^{-6}
	With cross-attention	18.58	−2.00	3.0×10^{-6}
16 tasks, 5k steps	Ours	20.01	+1.87	3.1×10^{-6}
	Atom cap 48	19.99	+1.70	–
	No Gram–Schmidt	19.98	+1.23	0.77

E.4 Validating the score reading

Before a new task is trained, we want to know whether it overlaps with tasks we have already stored, because a task that mostly repeats a stored one cannot gain much under our orthogonality constraint. The score reading answers this cheaply. It trains a small probe adapter for each task for a few hundred steps and scores how much the probes of two tasks overlap. We check two things: that this cheap score predicts the overlap of the fully trained adapters, and that its verdict holds up when we actually train the tasks.

The score is a reliable stand-in for full training. Across all 120 task pairs it ranks the fully trained overlaps with a Spearman correlation of 0.71 to 0.73, and 0.81 when the probes train for 3000 steps. It barely depends on the random seed, since scores from different seeds agree at 0.97 to 0.98. It also transfers across architectures: probes trained on SD 1.5 still rank Sana's overlaps at 0.71, so the score reflects the tasks rather than the network. Cheaper shortcuts do not work, as overlaps measured on CLIP or VAE features of task inputs (images) correlate at only 0.02 and -0.10.

Table 13 then checks the verdicts in practice. Each row trains the listed tasks in order and reports how the last task fares compared with training it alone. Every natural pair, chain and sibling passes, and keeps 99 to 149% of the gain it reaches on its own, so passing tasks lose nothing by being learned after others. The three deliberately duplicated tasks are all flagged before training, with scores of 0.74 to 0.78, well above the highest natural pair at 0.60. Once trained, each flagged task either loses part of its gain or is absorbed by the shared atoms with a single new atom, which is exactly what a task with nothing new to add should do. In short, the score reading separates new tasks from repeated ones before any training, with a wide margin between the two.

Table 13: Score-reading validation on Sana, 5k steps per stage. The last task of each stream is the one being tested. "Kept" is its gain over the degraded input relative to the same task trained alone.

Case	Stream	Score	Verdict	Outcome
Pair	haze → extreme dark	0.02	pass	115% kept
Pair	JPEG → colorization	0.10	pass	149% kept
Pair	rain → snow	0.19	pass	103% kept
Chain	face inpainting, haze → extreme dark	0.02	pass	111% kept
Chain	extreme dark, JPEG → colorization	0.10	pass	131% kept
Chain	extreme dark, rain → snow	0.19	pass	99% kept
Sibling	face inpainting, extreme dark → low-light	0.13	pass	107% kept
Duplicate	haze, rain → rain	**0.78**	flag	loses 44%
Duplicate	face inpainting, low-light → low-light	**0.75**	flag	absorbed, one new atom
Duplicate	extreme dark, face inpainting → face inpainting	**0.74**	flag	absorbed, one new atom

E.5 Training dynamics and strength control

Figure 13 shows how the model grows. The residual rank drops to 4 or 5 within three to seven tasks and ends at 4 on every backbone, the atoms grow steadily until they reach their cap, and the training loss settles within the first two thousand steps of every stage. Every stored task can also be scaled at inference by a strength α_t,

$$y = W_0 x + \alpha_t \big(S\gamma_t + R_t\big)x, \tag{14}$$

where $\alpha_t = 0$ gives the frozen backbone and $\alpha_t = 1$ the full task. Since stored adapters never change, turning one dial cannot affect any other task. On Sana the rain dial raises PSNR at every step on every image, from about 8 to about 25 dB, and the haze dial rises just as steadily with a Spearman correlation of 1.0. Intermediate strengths have no ground truth, so for all 16 tasks we use two heuristic checks. The LPIPS distance to the fully restored output shrinks at 98.7% of all dial steps, and the KID (Kernel Inception Distance) between our outputs and clean images falls from 0.063 at $\alpha = 0$ to 0.010 at $\alpha = 0.5$ and 0.001 at $\alpha = 1$, so the dial moves every image steadily towards the clean distribution.

Figure 14 shows the dial on the remaining tasks.

E.6 Video restoration

Our method carries over to video without any change to the code. We attach it to SANA-Video-2B, a video diffusion model, and train it on five standard video benchmarks one after another: REVIDE for dehazing, GoPro for deblurring, KITTI-snow for desnowing, DAVIS with strong Gaussian noise for denoising and RainMotion for deraining. Each task works on 17-frame clips at 480p, and just as on images, every stored task stays exact and the rank of each new task drops to 4 after the first two. On the test clips, dehazing lifts PSNR from 14.6 to 19.5 dB, denoising from 14.9 to 23.7 dB and deraining from 15.8 to 18.8 dB. Figure 15 follows one clip per task over time, and the restored frames stay consistent from one frame to the next.

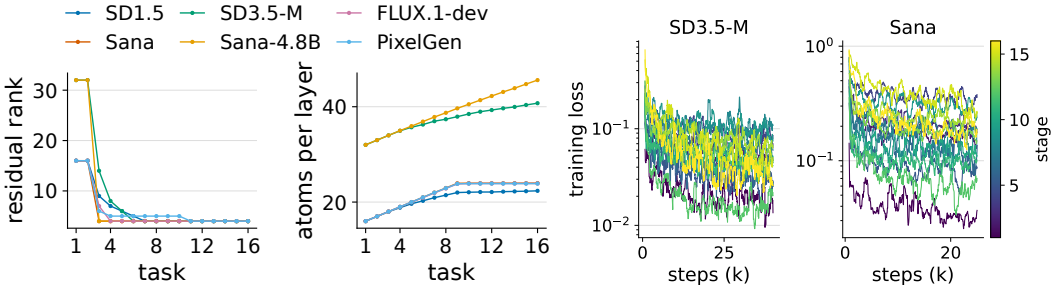

Figure 13: Left: residual rank and atoms per layer over the 16-task stream on all backbones. Right: training loss within each stage, colored by stage.

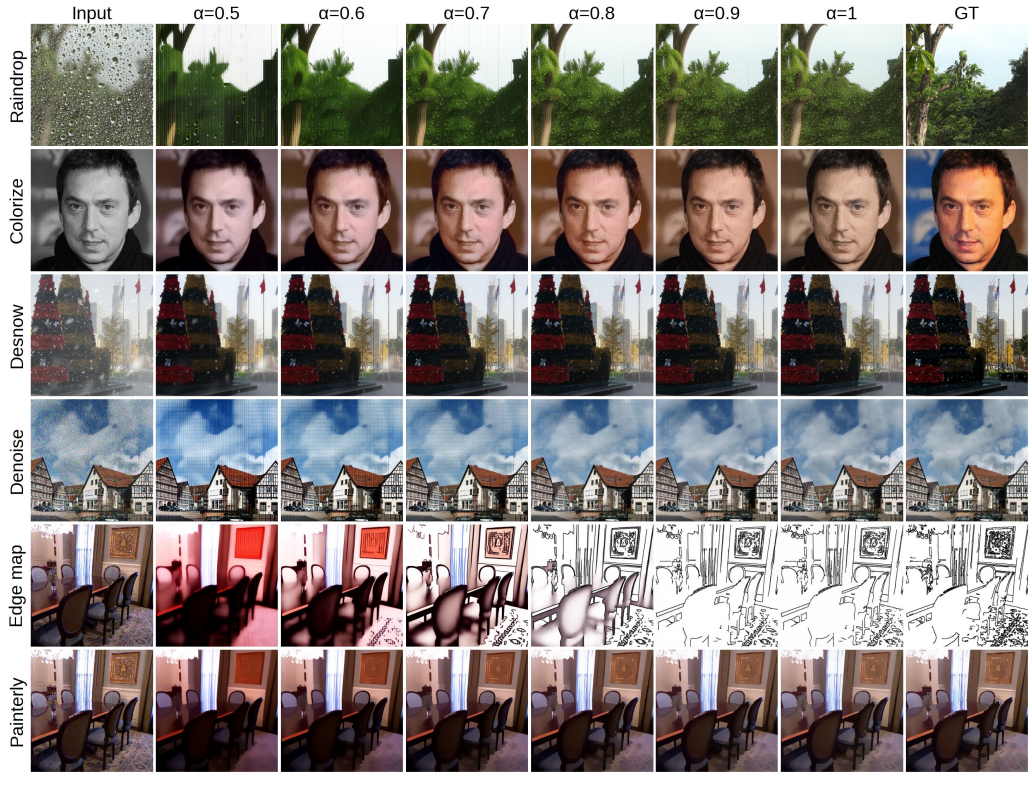

Figure 14: Strength control on the remaining tasks, from half strength to the full adapter at $\alpha = 1$. Desnowing and denoising already finish at about half strength.

We also trained a second video model on DAVIS clips with synthetic haze, low light and noise. Across its 12 validation clips, PSNR rises from 12.1 to 18.3 dB for haze, from 8.0 to 18.8 dB for low light and from 20.9 to 24.1 dB for noise (Figure 16). The videos in the supplementary material show both models on longer stretches, with the input, our result and the ground truth side by side.

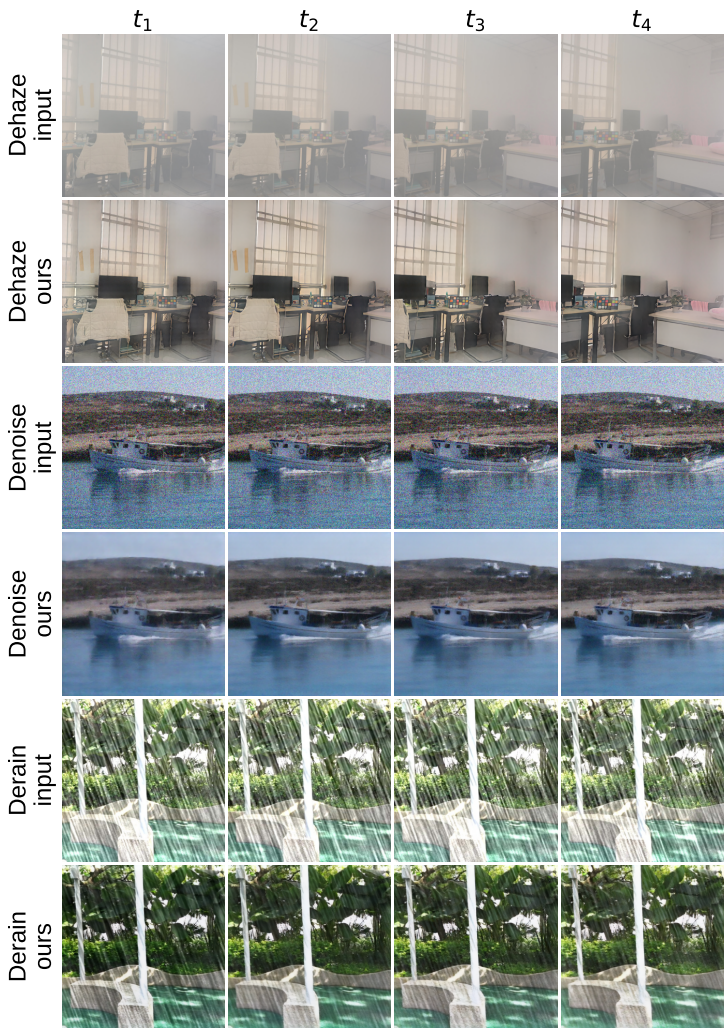

Figure 15: Video restoration on real benchmarks with SANA-Video-2B: dehazing on REVIDE, denoising on DAVIS and deraining on RainMotion. Each pair of rows shows four evenly spaced frames of one test clip.

E.7 More examples

Figure 17 compares our SD3.5 model with the joint oracle on the first eight tasks, and Figure 18 shows the stream that moves from restoration to stylization. Our styles follow their targets and the earlier restoration tasks keep their quality, while PCA-recompute rebuilds its basis for every new style.

E.8 Consolidating public adapters

The 16 public FLUX.1-dev adapters are consolidated one at a time in a fixed order with no training. The one-shot optimum knows all adapters up front and projects every adapter onto the best shared



Figure 16: Video restoration with synthetic degradations on DAVIS validation clips. Each pair of rows shows four evenly spaced frames of one clip.

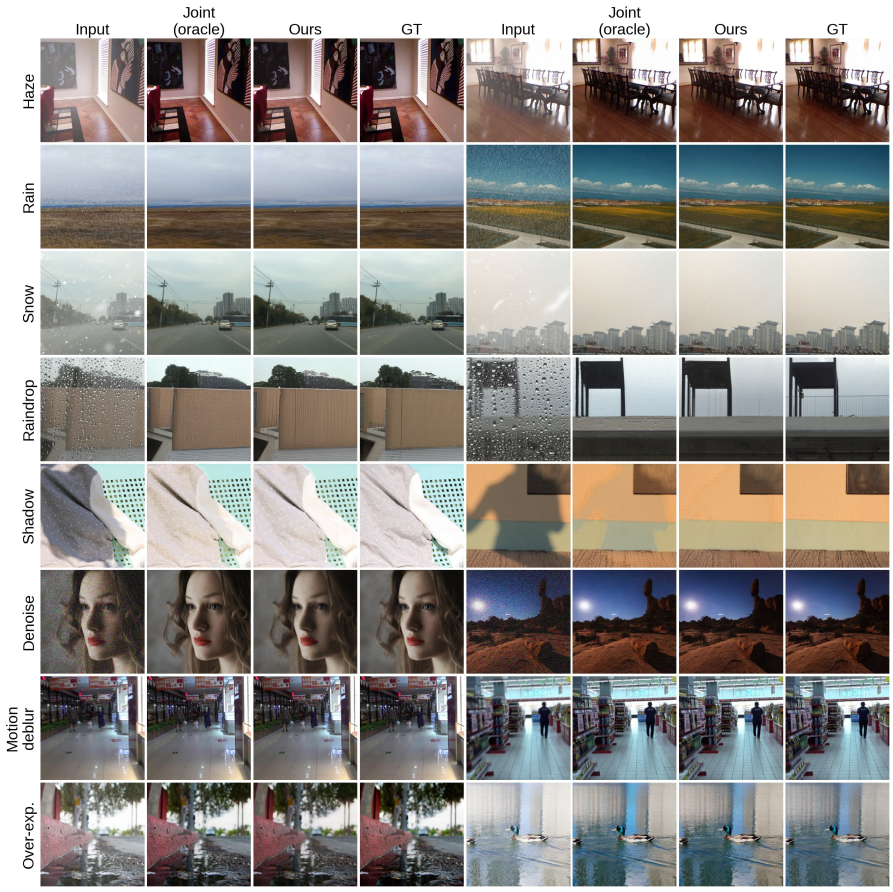

Figure 17: SD3.5, first eight tasks of the 16-task stream: input, joint oracle, ours and ground truth, two images per task.

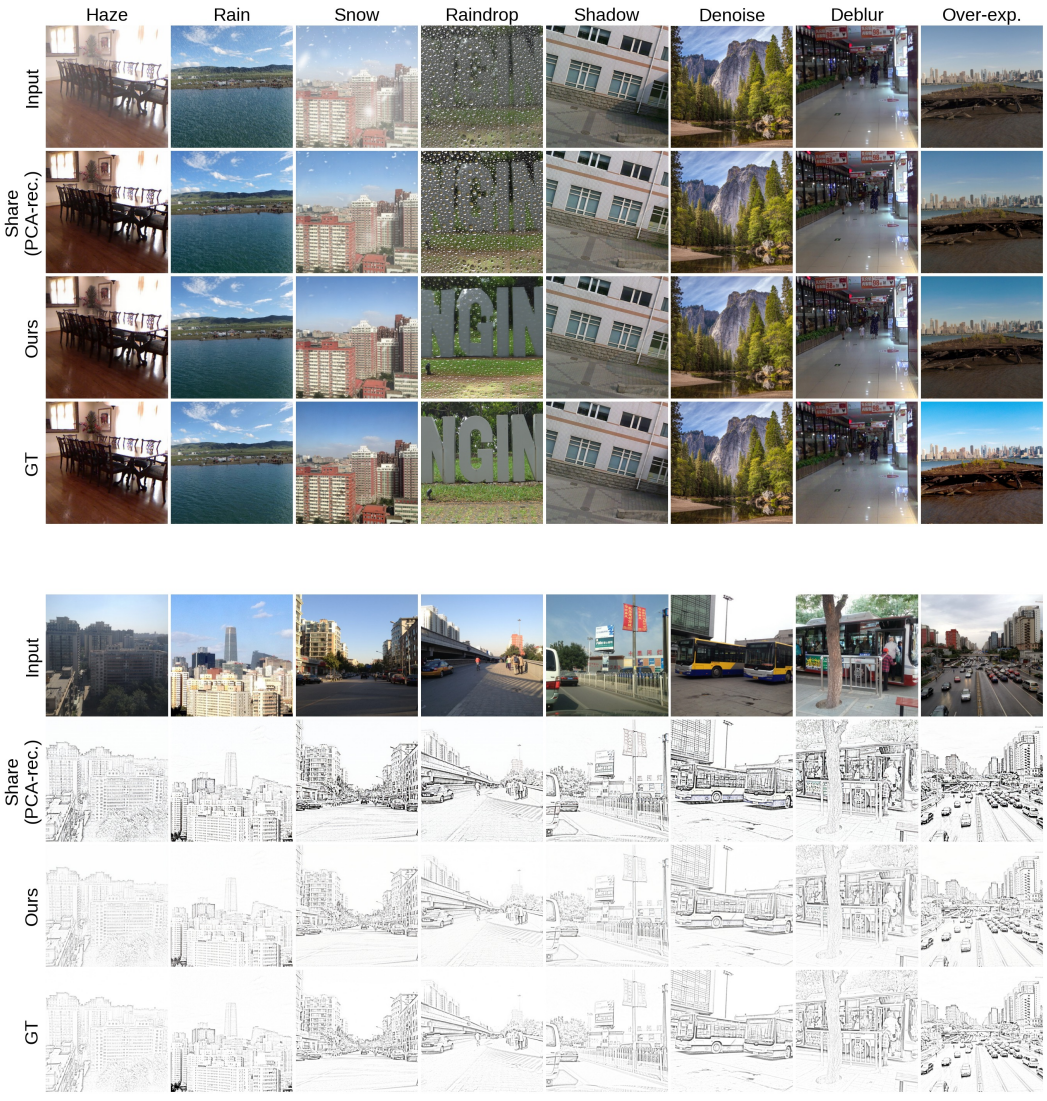

Figure 18: Restoration to stylization stream on SD3.5. Top: the eight restoration tasks after all style tasks are learned. Bottom: pencil sketch on eight scenes. Rows show input, PCA-recompute, ours and ground truth.

subspace of each layer, which is the lowest error a shared basis can reach at a given budget. Table 14 lists the exact parameter counts behind each budget of Table 3, and Figure 19 shows the eight adapters not in the main paper.

Table 14: Exact storage behind main Table 3 (M parameters) and mean end-of-stream error (%) over all 16 adapters.

budget column	0.18G	0.34G	0.77G	1.06G	1.5G
Share, components c	32	58	130	180	256
Share, parameters	189	343	768	1064	1513
ours, parameters	163	326	769	1064	1423
ours, error	34.4	22.7	6.4	4.7	2.2

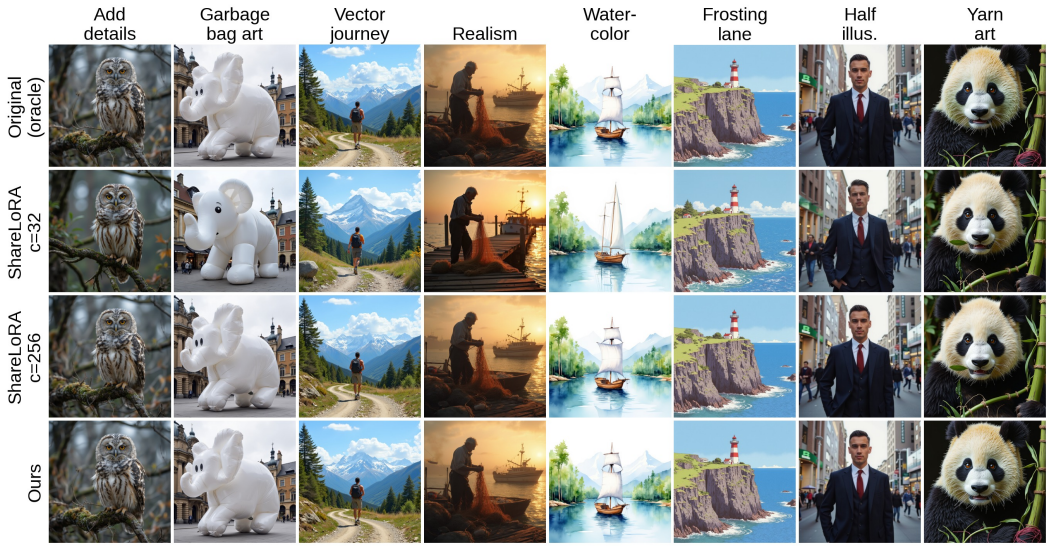

Figure 19: The remaining eight public adapters, same layout as the main-paper figure: original adapter, Share at its default budget, and ours after all 16 are stored.

F Adaptor Analysis

Each stored task has two parts, its coefficients on the shared atoms and its private residual. To see what each part does, we record how strongly each one responds at every image location, measured as the size of its output at the attention layers of the SD3.5 model and averaged over the sampling steps (Figure 20). The two parts look at different things. The shared part follows the structure of the scene, such as the letters behind the raindrops, the outlines of the buildings behind the snow and the bowls in the dark kitchen. The private part spreads over the regions that are specific to the degradation, such as the sky that snow and haze wash out, the whole area affected by the shadow and the droplet texture on the glass. The residual therefore carries what is specific to the degradation, while the shared part responds to the image content itself.

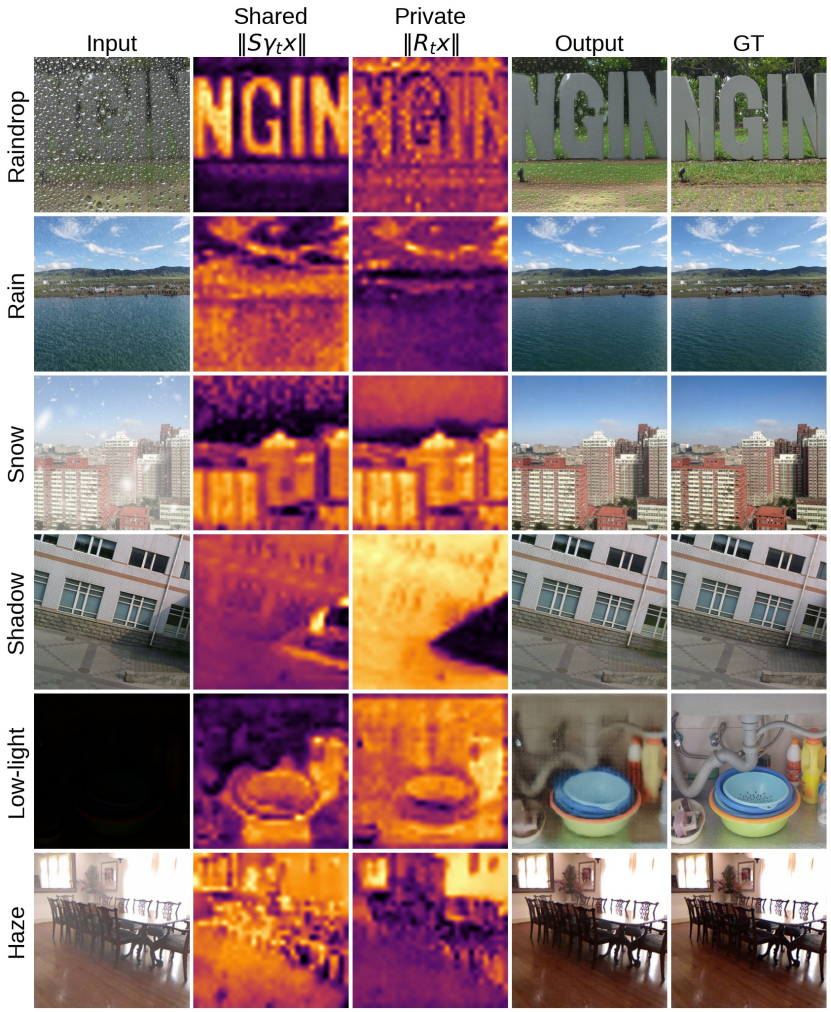

Figure 20: Response of the shared part $\|S\gamma_t x\|$ and the private part $\|R_t x\|$ of each task at every image location, SD3.5 after 16 tasks. Brighter means a stronger response.