BUDGETCMT: JOINT ALLOCATION OF TEACHER COMPUTE AND STUDENT SUPERVISION BUDGETS IN CONSISTENCY MID-TRAINING

Anonymous authors
Paper under double-blind review

ABSTRACT

Consistency mid-training (CMT) trains a few-step student on intermediate states of online teacher ODE trajectories. Generating these trajectories incurs teacher computation, while retaining student activations incurs memory cost. We introduce **BudgetCMT**, a joint allocation framework with three controls: teacher discretization K, trajectory refresh stride R, and student supervision budget S. Uniform sampling without replacement preserves the dense gradient in conditional expectation at a fixed parameter state and teacher pool, with covariance given by the finite-population correction. Complementary two-step reuse covers the pool once per cycle while halving teacher evaluations. On CIFAR-10, reducing supervision from 128 to 64 cells lowers reported interval peak GPU allocation by 41.8% in a three-seed 10k-update study; quality intervals allow both improvement and degradation. In a separate three-seed 30k-update continuation, reuse reduces loop time by a median of 30.6%. Its 2-NFE FID is 1.40% higher at equal updates and 1.42% lower at a pre-specified approximate equal-time endpoint. A single paired run from a second warm start shows the same direction, with a smaller equal-time gain. These results establish measured memory–time–quality trade-offs within CMT; the tested downstream fine-tuning recipe does not preserve checkpoint quality.

1 INTRODUCTION

Diffusion and flow-based generative models synthesize high-fidelity samples through iterative numerical integration of stochastic or probability-flow differential equations (Ho et al., 2020; Song et al., 2021a;b; Lipman et al., 2023; Liu et al., 2023; Albergo & Vanden-Eijnden, 2023). However, evaluating neural networks over dozens of sequential sampling steps remains computationally prohibitive. To accelerate inference, an extensive line of research explores few-step distillation and consistency modeling, including progressive distillation (Salimans & Ho, 2022), consistency models (Song et al., 2023; Song & Dhariwal, 2024), continuous-time sCM (Lu & Song, 2025), latent consistency models (LCM) (Luo et al., 2023), consistency trajectory models (CTM) (Kim et al., 2024), InstaFlow (Liu et al., 2024), phased consistency models (PCM) (Wang et al., 2024), Shortcut Models (Frans et al., 2025), and MeanFlow (Geng et al., 2025). While these paradigms substantially amortize inference cost, acquiring a capable few-step student still requires extensive training resources.

Consistency mid-training (CMT) (Hu et al., 2026) bridges this gap by directly supervising a student model on intermediate probability flow states generated online by a frozen teacher. In standard CMT, however, training incurs two structurally coupled, heterogeneous resource bottlenecks: (1) a *teacher computational throughput bottleneck*, where integrating the teacher ODE across K discrete time steps consumes heavy forward computation and dominates iteration wall-clock time; and (2) a *student activation memory bottleneck*, where backpropagating through multiple trajectory states accumulates large autograd graphs and dictates peak GPU memory allocation. These heterogeneous costs reveal that naive resource optimization can easily fall into imbalanced regimes: reducing student supervision cells shrinks activation memory without alleviating teacher ODE computation, whereas merely reusing teacher trajectories reduces integration overhead but couples successive gradient updates to a shared pool. Achieving compute-aware efficiency therefore demands jointly

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.

coordinating the generation of supervision with its consumption, while evaluating trade-offs along equal-update and equal-time axes.

To address this challenge, we introduce **BudgetCMT**, a unified budget allocation framework governing the three-dimensional execution space $\mathcal{M}(K, R, S)$ parameterized by teacher ODE resolution K, trajectory refresh stride R, and student supervision budget S (Figure 1). BudgetCMT exposes distinct controls for teacher computation and student activation memory. Specifically, we show that uniform sparse subsampling without replacement preserves the expected dense gradient under a finite-population correction, drastically cutting activation graphs. To amortize teacher computation without redundant evaluations, we introduce the **complementary sampling operator** $\mathcal{C}_R$, which partitions each teacher trajectory pool into disjoint subsets across R consecutive updates ($R = 2$), attaining 100% trajectory coverage while strictly halving teacher ODE evaluations.

We conduct extensive controlled experiments on CIFAR-10 within a unified CMT codebase to isolate these resource axes. In a 10,000-update study across three independent seeds, uniform sparsification from 128 to 64 cells reduces peak allocated GPU memory by **41.8%**, with uncertainty that does not establish quality equivalence. In a paired 30,000-update continuation campaign from a shared warm start, complementary two-step reuse ($R = 2$) cuts training loop time by a median of **30.6%**; while incurring a 1.40% increase in 2-NFE FID at equal updates, it achieves a 1.42% improvement in 2-step FID at a pre-specified approximate equal-time endpoint. One additional paired run from a second warm start shows the same direction with a smaller equal-time gain. Finally, teacher-grid comparisons, gradient-accumulation controls, and downstream safety-gated fine-tuning precisely delineate the empirical boundaries of this allocation strategy.

Our main contributions are threefold:

1. **Unified Multi-Resource Allocation Framework:** We parameterize CMT resource allocation over $\mathcal{M}(K, R, S)$ and give a first-order model of teacher computation and student activation memory.
2. **Theoretical Estimator and Drift Guarantees:** We prove that uniform student sparsification is a conditionally unbiased estimator of the dense gradient governed by the Finite Population Correction (Proposition 1), and bound the local parameter drift induced by complementary trajectory reuse under L-smoothness (Proposition 2).
3. **Controlled Empirical Validation and Boundary Characterization:** We provide rigorous multi-seed empirical measurements across equal-step and pre-specified approximate equal-time benchmarks, quantifying memory–time–quality trade-offs while identifying the operational limits of mid-training allocations via downstream transfer evaluations.

2 Related Work

Few-step generation and teacher supervision. Progressive distillation (Salimans & Ho, 2022) reduces sampling steps through successive teacher–student stages. Consistency models and improved consistency training (Song et al., 2023; Song & Dhariwal, 2024), continuous-time sCM (Lu & Song, 2025), and latent consistency models (Luo et al., 2023) develop consistency objectives in different settings. CTM (Kim et al., 2024) learns mappings between trajectory times, while PCM (Wang et al., 2024) uses phased consistency. InstaFlow (Liu et al., 2024) combines rectified flow and distillation; BOOT (Gu et al., 2024) and SiD (Zhou et al., 2024) pursue data-free distillation, with SiD using score identities and generator-produced samples. Shortcut Models (Frans et al., 2025) and MeanFlow (Geng et al., 2025) instead develop objectives for few-step generation without the same online teacher-trajectory supervision as CMT (Hu et al., 2026). BudgetCMT studies the allocation of teacher work and student supervision within this CMT setting, rather than comparing these distinct objectives as interchangeable baselines.

Training curricula and finite-population sampling. Timestep weighting and sampling appear in improved DDPM (Nichol & Dhariwal, 2021), adaptive timestep sampling (Kim et al., 2025), and curriculum consistency models (Liu et al., 2025). Curvature-adaptive consistency flow matching (Tian et al., 2026) studies trajectory adaptation. AlphaFlow (Zhang et al., 2026) develops a curriculum between trajectory flow matching and MeanFlow for training from scratch; discrete Meanflow curricula (Hsu & Wood, 2026) initialize from a pretrained flow model. Two-phase flow-matching curricula (Sun, 2026) also address training efficiency. These approaches modify objectives or train-

ing exposure. Our analysis instead starts from a fixed teacher pool and distinguishes uniform subset selection from uncorrected adaptive exposure. Classical sampling theory gives the uniform estimator's conditional mean and covariance (Cochran, 1977); Horvitz–Thompson weights can correct nonuniform inclusion (Horvitz & Thompson, 1952). This connects to sample selection and importance sampling in stochastic optimization (Katharopoulos & Fleuret, 2018; Loshchilov & Hutter, 2016; Mindermann et al., 2022; Bottou et al., 2018).

Memory-efficient execution. Activation checkpointing (Chen et al., 2016), FlashAttention (Dao et al., 2022), and ZeRO (Rajbhandari et al., 2020) address activation storage, attention execution, and distributed model-state storage, respectively. Student-cell sparsification changes the number of supervised cells; gradient accumulation retains all cells but executes smaller microbatches. Our CMT benchmark compares these latter two choices and teacher reuse. Their combination with other systems techniques, or their extension to LDM (Rombach et al., 2022) and DiT (Peebles & Xie, 2023), remains to be measured.

3 BudgetCMT: Multi-Resource Allocation Framework

3.1 Joint Parameterization and Execution Space

In consistency mid-training, training cost arises from two structurally distinct operations: the numerical integration of ordinary differential equation (ODE) trajectories by the frozen teacher, and the backpropagation of the differentiable student model retaining gradient graphs across multiple temporal states. We formalize resource allocation in CMT as a configuration-selection problem across a three-dimensional budget space $\mathcal{M}(K, R, S)$ (Figure 1):

$$\min_{(K,R,S)\in\mathcal{M}} T_{\text{wall}}(K, R, S) \quad \text{subject to} \quad \begin{cases} M_{\text{peak}}(K, S) \le M_{\text{budget}} \\ \text{FID}_j(K, R, S) \le \tau_j, \quad j \in \{1, 2\} \end{cases} \tag{1}$$

For a fixed training horizon, initialization, schedule, and evaluation protocol, T_{wall} denotes measured wall-clock loop time, M_{peak} represents peak allocated GPU memory, and τ_1, τ_2 denote fidelity thresholds for 1-NFE and 2-NFE inference. This defines a selection criterion, not an automated solver; our experiments compare specified configurations. The configuration space $\mathcal{M}$ is jointly parameterized by the teacher ODE discretization resolution K, the cross-update trajectory refresh stride R, and the student supervision budget S.

At each training iteration, B high-noise inputs are constructed from data images and Gaussian noise, and integrated by a pretrained unconditional teacher model (Karras et al., 2022). Discretizing each ODE path into K solver steps yields a cache pool Ω_t containing $N = BK$ aligned state–time–target cells:

$$\Omega_t = \left\{ \left(\mathbf{x}_{b,k}, t_k, \hat{\mathbf{x}}_{0,b} \right) \;\middle|\; b = 1, \ldots, B;\ k = 1, \ldots, K \right\}. \tag{2}$$

Conventional Dense CMT computes student forward passes and retains backpropagation activations across all N cells, regenerating the entire trajectory pool at every optimizer step. For a retained supervision subset $\mathcal{S} \subset \Omega_t$ of size $S \le N$, the student loss is defined as:

$$\mathcal{L}_{\mathcal{S}}(\theta) = \frac{1}{S} \sum_{i\in\mathcal{S}} \ell_i(\theta), \qquad \ell_i(\theta) = \text{LPIPS}\Big(f_\theta(\mathbf{x}_i, t_i),\, \hat{\mathbf{x}}_{0,i} \Big). \tag{3}$$

3.2 Uniform Sparse Supervision and Complementary Trajectory Reuse

BudgetCMT controls supervision count and teacher refresh through two coordinated mechanisms:

Uniform Supervision Sparsification (S): When $R = 1$, the student draws $S < N$ cells uniformly without replacement from each fresh pool Ω_t. Because only S activation graphs are retained during autograd, GPU peak memory scales with S rather than N. Under survey sampling theory, this estimator enjoys strong conditional guarantees:

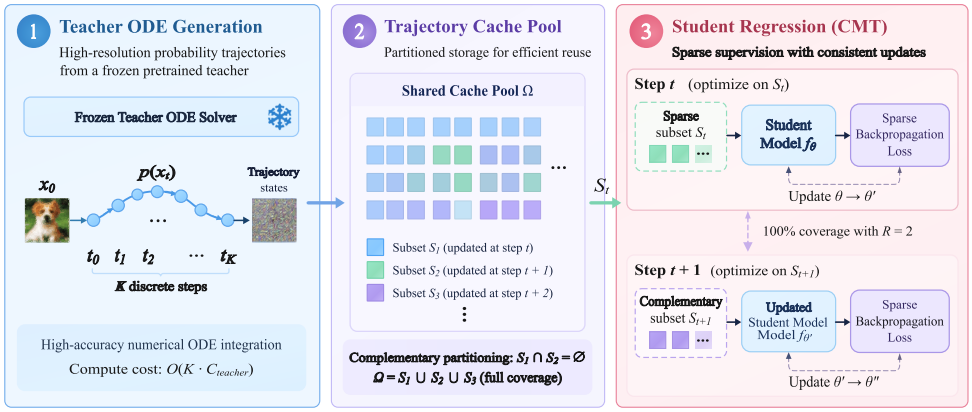


Figure 1: **BudgetCMT: Unified Budget Allocation and Execution Pipeline in Consistency Mid-Training. (1) Teacher ODE generation:** A frozen teacher integrates probability flow ODE trajectories across K discrete steps $\{x_{t_k}\}_{k=1}^{K}$ from data images and Gaussian noise, regulated by refresh stride $R = 2$ to halve teacher evaluations. **(2) Trajectory cache pool:** Generated states form a shared cache pool Ω (N elements), partitioned by the complementary sampling operator $\mathcal{C}_R$ into disjoint, full-coverage subsets ($S_1 \cap S_2 = \emptyset, S_1 \cup S_2 = \Omega$). **(3) Student regression:** At step t, student f_θ updates parameters $\theta \to \theta'$ on subset S_1. At step $t+1$, the updated student $f_{\theta'}$ evaluates on cached complement S_2, achieving 100% trajectory coverage across consecutive updates without redundant teacher ODE computation.

Proposition 1 (**Conditionally Unbiased Gradient Estimator with Finite Population Correction**). *Assume $N \geq 2$ and $1 \leq S \leq N$. Let $\mathbf{g}_i = \nabla_\theta \ell_i(\theta)$ denote the per-cell gradient vector and $\nabla \mathcal{L}_{\text{Dense}}(\theta) = \frac{1}{N}\sum_{i\in\Omega_t} \mathbf{g}_i$ the dense full-grid gradient. When the supervision subset $\mathcal{S}$ is drawn via simple random sampling without replacement (SRSWOR) of size S, the sparse sample gradient $\nabla \mathcal{L}_S(\theta) = \frac{1}{S}\sum_{i\in\mathcal{S}} \mathbf{g}_i$ satisfies*

$$\mathbb{E}_{\mathcal{S}\sim\Omega_t}\left[\nabla \mathcal{L}_S(\theta) \,\middle|\, \Omega_t, \theta\right] = \nabla \mathcal{L}_{\text{Dense}}(\theta), \qquad \text{Cov}\left(\nabla \mathcal{L}_S(\theta) \,\middle|\, \Omega_t, \theta\right) = \left(\frac{N-S}{N-1}\right)\frac{\mathbf{\Sigma}_{\Omega_t}}{S}, \tag{4}$$

where $\mathbf{\Sigma}_{\Omega_t} = \frac{1}{N}\sum_{i\in\Omega_t}(\mathbf{g}_i - \nabla \mathcal{L}_{\text{Dense}})(\mathbf{g}_i - \nabla \mathcal{L}_{\text{Dense}})^\top$ is the within-pool population covariance.

Proof sketch. Under SRSWOR, each cell has marginal inclusion probability $\pi_i = S/N$ and pairwise joint probability $\pi_{ij} = \frac{S(S-1)}{N(N-1)}$ for $i \neq j$. By linearity of expectation, $\mathbb{E}[\nabla \mathcal{L}_S(\theta)|\Omega_t, \theta] = \nabla \mathcal{L}_{\text{Dense}}(\theta)$. Expanding the indicator covariance yields the Finite Population Correction (FPC) factor $\frac{N-S}{N-1}$ (Cochran, 1977). See Appendix A for complete derivations. ■

The FPC factor $\frac{N-S}{N-1}$ strictly vanishes at full retention ($S = N$), and gives variance no larger than an equally sized uniform sample with replacement. Reduction is strict for $S > 1$ in directions with nonzero population variance. In contrast, uncorrected heuristic samplers (e.g., TVA, LSM) alter cell inclusion probabilities without importance weights, changing the effective training objective rather than purely reducing variance (Appendix A).

Complementary Trajectory Reuse ($\mathcal{C}_R$, R): To amortize the teacher ODE integration bottleneck, we introduce refresh stride $R \geq 2$, sharing each pool Ω_t across R consecutive updates. To eliminate redundant supervision across steps, we define the **complementary sampling operator** $\mathcal{C}_R : \Omega_t \mapsto (\mathcal{S}_{t,1}, \ldots, \mathcal{S}_{t,R})$ such that partitions form a disjoint, exact cover:

$$\mathcal{S}_{t,r} \cap \mathcal{S}_{t,r'} = \emptyset \quad (\forall r \neq r'), \qquad \bigcup_{r=1}^{R} \mathcal{S}_{t,r} = \Omega_t. \tag{5}$$

For $R = 2$ with $BK = 2S$ (e.g., $B = 8, K = 12, S = 48 \implies N = 96$), the pool is permuted and split into two complementary halves. Over a complete reuse cycle, average teacher evaluations per update fall strictly from BK to $BK/2$. Between steps t and $t+1$, student parameters advance from θ_t to $\theta_{t+1} = \theta_t - \eta_t \mathbf{v}_t$. We bound the gradient change from evaluating the cached complement at updated parameters:

Algorithm 1: BudgetCMT Execution Pipeline

Input: Frozen teacher model, student model θ_0, dataset, batch size B, teacher steps K, student budget S, refresh stride $R \in \{1, 2\}$, learning rate schedule η_t. For $R = 2$, require $BK = 2S$.

Output: Optimized EMA student model.

1. Initialize student optimizer and EMA weights; initialize an empty trajectory cache.
2. **For** each update step $t = 0, 1, \ldots, T-1$:
3. **if** cache is empty or pool usage count equals R:
4. Sample B training images and independent Gaussian noise; integrate teacher ODE over K steps to populate Ω_t (BK aligned cells); reset usage count to 0. If $R = 2$, uniformly permute this fresh pool once and cache its two complementary halves.
5. **if** $R = 1$: sample S cells $\mathcal{S}_t \subset \Omega_t$ uniformly without replacement.
6. **if** $R = 2$: select the cached half indexed by usage count plus one, without repartitioning.
7. Compute student loss $\mathcal{L}_S(\theta_t)$ on selected cells via Eq. (3); backpropagate gradients.
8. Apply optimizer update $\theta_{t+1} \leftarrow \theta_t - \eta_t \mathbf{v}_t$; update student EMA; increment usage count.

Proposition 2 (Local Parameter Drift Bound under Complementary Reuse). *Assume each cell loss $\ell_i(\theta)$ is L-smooth, and the student update satisfies $\|\mathbf{v}_t\| \le G_{\max}$. For complementary partitions with $|\mathcal{S}_1| = |\mathcal{S}_2| = S$, the inter-step gradient drift vector $\mathbf{\Delta}_{t+1} = \nabla\mathcal{L}_S(\theta_{t+1}; \mathcal{S}_2) - \nabla\mathcal{L}_S(\theta_t; \mathcal{S}_2)$ and the two-step gradient average satisfy*

$$\|\mathbf{\Delta}_{t+1}\| \le L\eta_t G_{\max}, \qquad \left\| \frac{\nabla\mathcal{L}_S(\theta_t; \mathcal{S}_1) + \nabla\mathcal{L}_S(\theta_{t+1}; \mathcal{S}_2)}{2} - \nabla\mathcal{L}_{\text{Dense}}(\theta_t) \right\| \le \frac{L\eta_t G_{\max}}{2}. \quad (6)$$

Proof sketch. At a fixed parameter state θ_t, $(\nabla\mathcal{L}_S(\theta_t; \mathcal{S}_1) + \nabla\mathcal{L}_S(\theta_t; \mathcal{S}_2))/2 \equiv \nabla\mathcal{L}_{\text{Dense}}(\theta_t)$ holds pathwise. By L-smoothness, $\|\nabla\ell_i(\theta_{t+1}) - \nabla\ell_i(\theta_t)\| \le L\|\theta_{t+1} - \theta_t\| \le L\eta_t G_{\max}$. Applying Jensen's inequality across $\mathcal{S}_2$ yields both bounds. See Appendix A. ■

Because the teacher is frozen, targets do not change; the discrepancy arises solely from student parameter evolution. The execution pipeline coordinating these controls is summarized in Algorithm 1.

3.3 Resource Scaling and Additive Cost Accounting

For fixed network architecture, arithmetic precision, and hardware execution strategy, training resource consumption decomposes into an additive first-order model:

$$T_{\text{step}}(K, R, S) = T_0 + \frac{bK}{R} + cS, \qquad M_{\text{peak}}(K, S) = A(K) + aS, \quad (7)$$

where T_0 and $A(K)$ encapsulate common framework overhead, model parameters, and optimizer states; bK/R denotes amortized teacher ODE integration cost; and cS, aS represent student forward-backward computation and backpropagation activation graph storage, respectively.

Under this model, halving student supervision from S to $S/2$ yields relative memory savings of $\frac{1}{2}\phi$, where $\phi = \frac{aS}{A(K)+aS} \in [0, 1]$ represents the activation fraction of peak memory. In contrast, doubling refresh stride from $R = 1$ to $R = 2$ yields step-time savings of $\frac{bK/2}{T_0+bK+cS}$. Because teacher integration dominates wall-clock time while student backpropagation dominates activation retention, varying S primarily scales memory footprint, whereas varying R primarily accelerates computational throughput. These are different primary effects, not strict orthogonality: S also changes step time, and K changes both teacher work and cache storage. The model is an approximation for the evaluated execution strategy.

4 Experiments

4.1 Setup and comparison budgets

We evaluate unconditional CIFAR-10 generation (Krizhevsky, 2009) with a frozen VP-EDM teacher (Karras et al., 2022), an LPIPS objective (Zhang et al., 2018), and RAdam (Liu et al., 2020). We report Fréchet Inception Distance (FID; Heusel et al., 2017) for one and two network function evaluations (NFE). Main quality endpoints use 50k generated images, a 50k-image reference set, and

FP32 inference. Means and sample standard deviations summarize training seeds; timing blocks are technical repetitions, not quality replicates.

Supervision study. We compare dense and sparse supervision over 10k updates on an A100 80-GB GPU, using BF16/TF32 and seeds 31, 37, and 43. The learning rate starts at 2×10^{-4} and decays linearly; EMA decay is 0.999. The nominal batch size is 128 student cells, constructed from $B = 8$ teacher trajectories at $K = 16$, not 128 independent trajectories. The 10k horizon is 3.3% of the 300k-update reference schedule (Sony Research, 2026).

Primary continuation. On an RTX 3090, fresh and reuse arms continue identical EMA weights from a 10k-update warm start for 30k further updates, at $(K, S) = (12, 48)$. Both clear the cache and use cosine learning-rate decay from 5×10^{-5} to 10^{-5}, with paired update seeds 150131, 150137, and 150143. Fresh draws 48 of 96 cells from a new pool; reuse consumes complementary halves over two updates. Loop time excludes serialization and offline evaluation. The equal-time threshold is fixed from pilot rates before the formal runs; it is not chosen from the observed quality curves. Appendix B gives full protocols and seed-level endpoints.

4.2 Sparse supervision reduces activation memory

At fixed $K = 16$ and $R = 1$, reducing S from 128 to 64 lowers reported interval peak allocation from 9.543 ± 0.072 to 5.555 ± 0.052 GiB, a 41.8% reduction. Training wall time decreases by only 5.6% because the full teacher solve remains (Figure 2a,b). This difference is consistent with the separate resource terms in Eq. (7).

Table 1: **Sparse supervision at 10k updates.** Mean $\pm$ sample SD across three seeds. Memory is reported interval peak allocation. FID uses 50k samples (lower is better). Details in Appendix B.

Method	S	1-NFE FID	2-NFE FID	Memory (GiB)
Dense	128	9.903 ± 0.137	4.745 ± 0.079	9.543 ± 0.072
Sparse-uniform	64	10.018 ± 0.062	4.788 ± 0.068	5.555 ± 0.052
TVA	64	10.121 ± 0.235	4.844 ± 0.035	5.587 ± 0.006
LSM	64	9.735 ± 0.102	4.862 ± 0.096	5.540 ± 0.069

Mean FID differences between sparse and dense configurations remain modest: paired sparse-minus-dense increments are $+0.115$ for 1-NFE (95% CI $[-0.372, 0.603]$) and $+0.044$ for 2-NFE (95% CI $[-0.312, 0.400]$), allowing both improvement and degradation; these intervals do not establish quality equivalence. At the matched budget $S = 64$, heuristic adaptive baselines (TVA and LSM) fail to Pareto-dominate uniform sampling across both endpoints (Table 1). Because they evaluate uncorrected sample means, they alter effective objective exposure rather than purely reducing estimator variance.

4.3 Reuse trades equal-step quality for equal-time quality

At 30k updates, complementary reuse halves teacher image evaluations in every seed pair and reduces loop time by a median of 30.61% (range 30.20–32.91%). Peak allocated GPU memory is byte-identical within each pair (exactly 4,727,824,384 bytes in all six runs), empirically corroborating that reuse accelerates training throughput without altering the per-update activation memory footprint.

At equal updates, reuse exhibits a quality cost at the equal-update endpoint: the paired relative FID increment is $+1.40\% \pm 0.53\%$ for 2-NFE and $+0.41\% \pm 1.34\%$ for 1-NFE across the three paired seeds.

For the time comparison, the frozen threshold is $T_{\text{iso}} = 10{,}440$ s, calculated from a 400-update pilot. We evaluate the first saved EMA snapshot whose elapsed time reaches this threshold. Fresh reaches it at 16k–17k updates and reuse at 23k–25k updates, completing approximately 44% more updates under the matched time budget. Consequently, reuse achieves lower FID across all three pairs on both inference budgets: $-3.06\% \pm 1.72\%$ for 1-NFE and $-1.42\% \pm 0.58\%$ for 2-NFE (Table 2). Because checkpoints are recorded at 1,000-update intervals, this comparison reflects an approximate snapshot-matched time budget.

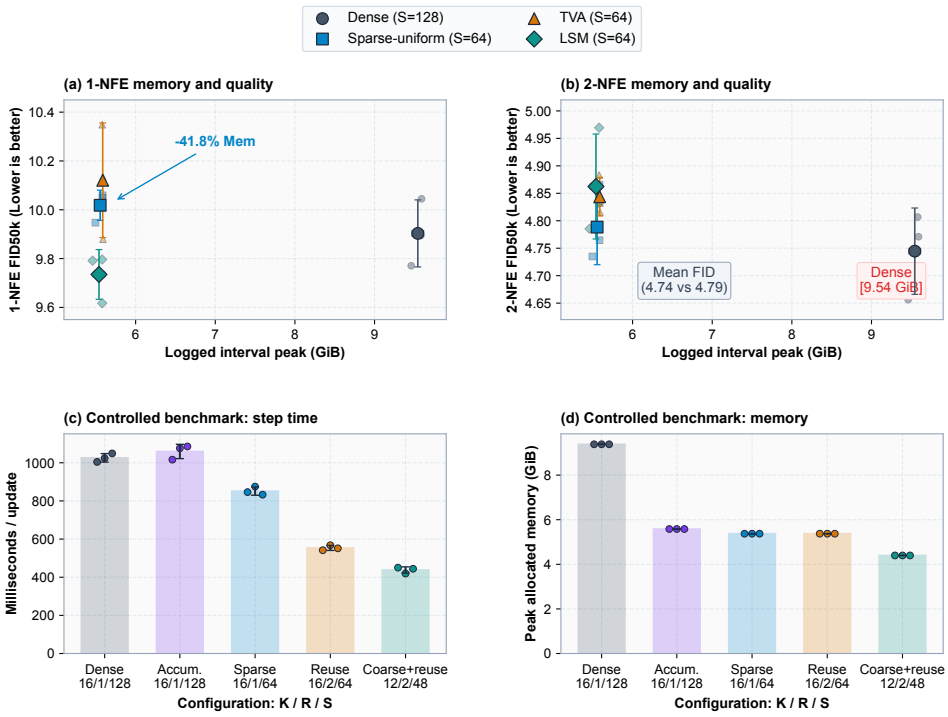


Figure 2: **Memory and time respond to different controls.** (a,b) FID50k versus reported interval peak allocation at 10k updates, with three training seeds and mean ± sample SD. (c,d) A separate RTX 3090 resource benchmark uses three timing blocks per configuration. These are technical repetitions isolating hardware execution without quality metrics; the two protocols are evaluated independently.

Table 2: **Primary reuse comparison.** Relative FID differences (reuse minus fresh, normalized by fresh) across three paired seeds (mean ± sample SD). Negative favors reuse. Approximate equal-time threshold is 10,440 s.

Comparison	Δ1-NFE FID	Δ2-NFE FID
Equal updates (30k)	$+0.41\% \pm 1.34\%$	$+1.40\% \pm 0.53\%$
Approximate equal time	$-3.06\% \pm 1.72\%$	$-1.42\% \pm 0.58\%$
Absolute 2-NFE FID at approximate equal time		
Seed	Fresh	Reuse
150131	4.4360	4.3433
150137	4.4574	4.4072
150143	4.4524	4.4064

Trajectory Dynamics across Horizons and Replications. Evaluating intermediate checkpoints yields 35 empirical time–quality points across the six runs. While interpolated trajectories confirm a consistent wall-clock advantage on 1-NFE, intermediate 2-NFE crossings exhibit seed-dependent variations. To prevent post-hoc checkpoint selection bias, all primary analyses strictly adhere to the pre-specified fixed endpoints (Appendix B.5).

A single paired run from a second, independently trained warm start shows the direction of all four endpoint comparisons in a paired run: reuse exhibits a slight penalty at equal updates ($+1.72\%$ 1-NFE, $+2.92\%$ 2-NFE FID) while achieving a superior approximate equal-time endpoint (-0.88% 1-NFE, -0.33% 2-NFE FID), with loop-time savings remaining consistent at 31.23% (Table A7). This is one paired run from a second initialization, not a multi-seed replication; its smaller equal-time gain limits conclusions about robustness across starts.

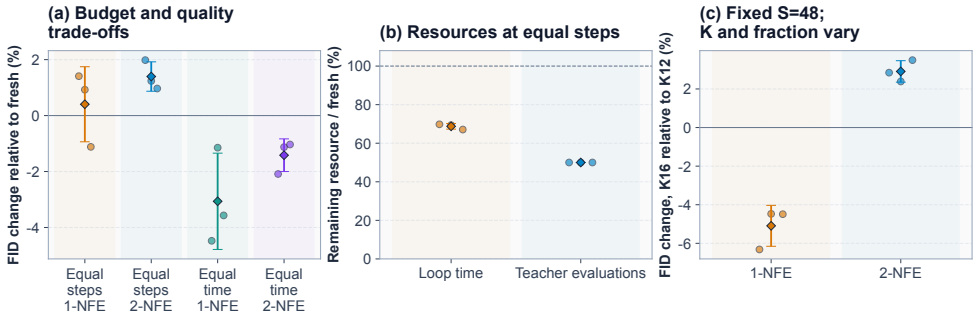


Figure 3: **Paired 30k continuation.** (a) Relative FID differences at equal updates and the pre-specified approximate equal-time endpoint. (b) Loop time and teacher evaluations relative to fresh. (c) Changing K at fixed $S = 48$ also changes the retained fraction. Points denote seeds; error bars show sample SD across three seeds.

4.4 Complementary coverage and the learning-rate schedule

The primary experiment compares fresh supervision with complementary reuse. A separate adaptation study compares complementary reuse with drawing independent uniform subsets from the same cached pool. At $N = 96$ and $S = 48$, independent draws may select a cell twice within a cycle; complementary halves cover each cell once. Both schedules use the same teacher budget, isolating coverage geometry from teacher computation.

Table 3: **Coverage and schedule comparisons from a separate mature checkpoint.** FP32 FID50k. 2k constant-rate anchors (1 seed); 2801-update cosine comparison (3 paired seeds, mean $\pm$ sample SD).

Budget / schedule	Method	1-NFE FID	2-NFE FID
2k / constant	Fresh	10.398	4.581
	Independent reuse	10.439	4.577
	Complementary reuse	10.439	4.544
2801 / cosine	Fresh	10.255 ± 0.031	4.584 ± 0.045
	Complementary reuse	10.396 ± 0.068	4.524 ± 0.019

At 2k updates, complementary and independent reuse achieve identical 1-NFE FID (10.439), while complementary reuse yields a lower 2-NFE FID (4.544 versus 4.577; Table 3). With matched cosine schedules over 2,801 updates, complementary reuse achieves lower mean 2-NFE FID (4.524 ± 0.019) than fresh (4.584 ± 0.045), with a paired difference of -0.060 ± 0.034 (95% CI $[-0.144, 0.024]$). The full constant-rate and phase-schedule comparisons appear in Table A4.

4.5 Teacher resolution depends on the inference budget

We compare $K = 12$ and $K = 16$ over the same 30k-update horizon, holding the warm start, schedule, paired seeds, $R = 1$, and $S = 48$ fixed. While using 33.3% more teacher evaluations per update, the finer grid improves 1-NFE FID by $5.09\% \pm 1.06\%$ but degrades 2-NFE FID by $2.91\% \pm 0.56\%$ across all three seeds (Figure 3c; Appendix B.6). Changing K also alters the retained fraction $S/(BK)$ from 0.500 to 0.375. The observed NFE-dependent trade-off therefore combines grid resolution and supervision density, rather than isolating a discretization effect.

4.6 Accumulation and sparse supervision save memory differently

Gradient accumulation retains all cells while splitting student execution across microbatches. In a dedicated RTX 3090 hardware benchmark, accumulation reduces peak memory from 9.382 to 5.582 GiB but increases step time by 3.3%. In contrast, sparse supervision achieves 5.373 GiB and reduces student compute in a single pass, while complementary reuse further amortizes teacher computation

Table 4: **Real CMT resource control on RTX 3090.** Block mean ± sample SD across three repetitions (8 warmup, 32 measured updates, BF16/TF32, frozen teacher, 2k student checkpoint). Measured peak memory is allocated; FID is not evaluated.

Configuration	K	R	S	Microbatch	ms/update	GiB
Dense	16	1	128	128	1025.9 ± 22.3	9.382
Accumulation	16	1	128	64	1059.4 ± 37.5	5.582
Sparse	16	1	64	64	851.4 ± 21.7	5.373
Reuse	16	2	64	64	553.6 ± 13.2	5.374
Coarse + reuse	12	2	48	48	438.0 ± 16.4	4.397

with negligible memory impact (Table 4; Figure 2c,d). This resource-only benchmark characterizes execution costs; the separate continuation experiments evaluate generative quality.

4.7 Mid-training gains do not guarantee downstream transfer

We evaluate downstream transfer by applying the official consistency-training recipe to the 30k checkpoints under fixed hyper-parameters. A pre-specified step-300 safety gate halts training if FID10k degrades by over 20%; all tested checkpoints trigger this gate (Table A10). The evaluated recipe does not preserve checkpoint quality. This test does not separate initialization effects from recipe, hyperparameter, or implementation effects, and it supplies no long-horizon downstream result (Appendix B).

5 Discussion and Conclusion

BudgetCMT coordinates teacher computation and student supervision through K, R, and S. In the evaluated CMT implementation, sparse supervision primarily reduces activation memory, while trajectory reuse primarily reduces teacher work. The 10k-update study reports 41.8% lower interval peak allocation. In a separate 30k-update continuation, reuse halves teacher evaluations and reduces loop time by 30.6%, at the cost of 1.40% higher 2-NFE FID at equal updates; at the pre-specified approximate equal-time endpoint, FID is 1.42% lower.

Scope and future work. Conditional unbiasedness holds at fixed parameters and a fixed pool; it does not equate the ensuing optimization trajectories. The complementary-partition identity and smoothness bound hold pathwise but do not establish optimizer convergence. Experiments cover CIFAR-10, one CMT implementation, and the reported A100/RTX 3090 protocols. The second warm start supplies one additional paired run. All tested downstream checkpoints trigger the safety gate under the evaluated fine-tuning recipe; this experiment does not isolate the cause. Extending the approach to LDM (Rombach et al., 2022) or DiT (Peebles & Xie, 2023), and combining it with FlashAttention (Dao et al., 2022) or ZeRO (Rajbhandari et al., 2020), require separate measurements of resource and quality effects.

AI USE STATEMENT

Generative AI tools assisted with code organization and checking the manuscript for errors and grammar. The authors are responsible for the final submitted content.

ETHICS STATEMENT

This work studies resource allocation for consistency mid-training using the existing CIFAR-10 benchmark and a pretrained generative teacher. The reported experiments do not involve recruiting human participants or collecting new personal data. The resulting models may inherit biases and privacy risks from the training data and pretrained teacher, and improved training efficiency may also lower the cost of potential misuse of generative models. Our experiments do not evaluate or mitigate these risks. We report resource savings only for the measured configurations and accounting boundaries; we do not infer reductions in energy consumption or carbon emissions from GPU memory or training-time measurements.

REPRODUCIBILITY STATEMENT

We provide source code, dependency specifications, and scripts for training, evaluation, resource measurement, and statistical analysis in the supplementary ZIP archive. Section 3 describes the method, and Appendix A provides theoretical assumptions and proofs. Experimental protocols, implementation details, and seed-level results are documented in Appendices B–E.

REFERENCES

Michael S. Albergo and Eric Vanden-Eijnden. Building normalizing flows with stochastic interpolants. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=li7qeBbCR1t.

Léon Bottou, Frank E. Curtis, and Jorge Nocedal. Optimization methods for large-scale machine learning. *SIAM Review*, 60(2):223–311, 2018. doi: 10.1137/16M1080173.

Tianqi Chen, Bing Xu, Chiyuan Zhang, and Carlos Guestrin. Training deep nets with sublinear memory cost. *arXiv preprint arXiv:1604.06174*, 2016. URL https://arxiv.org/abs/1604.06174.

William G. Cochran. *Sampling Techniques*. Wiley, New York, 3 edition, 1977.

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. FlashAttention: Fast and memory-efficient exact attention with IO-awareness. In *Advances in Neural Information Processing Systems*, volume 35, pp. 16344–16359, 2022. URL https://papers.nips.cc/paper_files/paper/2022/hash/67d57c32e20fd0a7a302cb81d36e40d5-Abstract-Conference.html.

Kevin Frans, Danijar Hafner, Sergey Levine, and Pieter Abbeel. One step diffusion via shortcut models. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/559a0998fab1d19b80e7e43a5852401c-Abstract-Conference.html.

Zhengyang Geng, Mingyang Deng, Xingjian Bai, J. Zico Kolter, and Kaiming He. Mean flows for one-step generative modeling. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-2534. URL https://proceedings.nips.cc/paper_files/paper/2025/hash/6d13e085b79d454da5910e4ca82a3d9d-Abstract-Conference.html.

Jiatao Gu, Chen Wang, Shuangfei Zhai, Yizhe Zhang, Lingjie Liu, and Joshua M. Susskind. Data-free distillation of diffusion models with bootstrapping. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 16622–16646, 2024. URL https://proceedings.mlr.press/v235/gu24d.html.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In *Advances in Neural Information Processing Systems*, volume 30, 2017. URL https://papers.nips.cc/paper/2017/hash/8a1d694707eb0fefe65871369074926d-Abstract.html.

Jonathan Ho, Ajay N. Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, volume 33, pp. 6840–6851, 2020. URL https://papers.nips.cc/paper/2020/hash/4c5bcfec8584af0d967f1ab10179ca4b-Abstract.html.

Daniel G. Horvitz and Donovan J. Thompson. A generalization of sampling without replacement from a finite universe. *Journal of the American Statistical Association*, 47(260):663–685, 1952. doi: 10.1080/01621459.1952.10483446.

Chia-Hong Hsu and Frank Wood. Discrete Meanflow training curriculum. In *Second DeLTa Workshop at ICLR 2026*, 2026. URL https://openreview.net/forum?id=WPcqBri0DF.

Zheyuan Hu, Chieh-Hsin Lai, Yuki Mitsufuji, and Stefano Ermon. CMT: Mid-training for efficient learning of consistency, mean flow, and flow map models. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2509.24526.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In *Advances in Neural Information Processing Systems*, volume 35, pp. 26565–26577, 2022. doi: 10.52202/068431-1926. URL https://papers.nips.cc/paper_files/paper/2022/hash/a98846e9d9cc01cfb87eb694d946ce6b-Abstract-Conference.html.

Angelos Katharopoulos and François Fleuret. Not all samples are created equal: Deep learning with importance sampling. In *International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 2525–2534, 2018. URL https://proceedings.mlr.press/v80/katharopoulos18a.html.

Dongjun Kim, Chieh-Hsin Lai, Wei-Hsiang Liao, Naoki Murata, Yuhta Takida, Toshimitsu Uesaka, Yutong He, Yuki Mitsufuji, and Stefano Ermon. Consistency trajectory models: Learning probability flow ODE trajectory of diffusion. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/c204d12afa0175285e5aac65188808b4-Abstract-Conference.html.

Myunsoo Kim, Donghyeon Ki, Seong-Woong Shim, and Byung-Jun Lee. Adaptive non-uniform timestep sampling for accelerating diffusion model training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2513–2522, 2025. URL https://openaccess.thecvf.com/content/CVPR2025/html/Kim_Adaptive_Non-Uniform_Timestep_Sampling_for_Accelerating_Diffusion_Model_Training_CVPR_2025_paper.html.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009. URL https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=PqvMRDCJT9t.

Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. On the variance of the adaptive learning rate and beyond. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=rkgz2aEKDr.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=XVjTT1nw5z.

Xingchao Liu, Xiwen Zhang, Jianzhu Ma, Jian Peng, and Qiang Liu. InstaFlow: One step is enough for high-quality diffusion-based text-to-image generation. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/4dc37a7bc61057252ce043fa3b83aac2-Abstract-Conference.html.

Yunpeng Liu, Boxiao Liu, Yi Zhang, Xingzhong Hou, Guanglu Song, Yu Liu, and Haihang You. See further when clear: Curriculum consistency model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18103–18112, 2025. URL https://openaccess.thecvf.com/content/CVPR2025/html/Liu_See_Further_When_Clear_Curriculum_Consistency_Model_CVPR_2025_paper.html.

Ilya Loshchilov and Frank Hutter. Online batch selection for faster training of neural networks. In *International Conference on Learning Representations Workshop Track*, 2016. URL https://arxiv.org/abs/1511.06343.

Cheng Lu and Yang Song. Simplifying, stabilizing and scaling continuous-time consistency models. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/7e9c2053258b1bdd32ff2654802cd594-Abstract-Conference.html.

Simian Luo, Yiqin Tan, Longbo Huang, Jian Li, and Hang Zhao. Latent consistency models: Synthesizing high-resolution images with few-step inference. *arXiv preprint arXiv:2310.04378*, 2023. URL https://arxiv.org/abs/2310.04378.

Sören Mindermann, Jan M. Brauner, Muhammed T. Razzak, Mrinank Sharma, Andreas Kirsch, Winnie Xu, Benedikt Höltgen, Aidan N. Gomez, Adrien Morisot, Sebastian Farquhar, and Yarin Gal. Prioritized training on points that are learnable, worth learning, and not yet learnt. In *International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 15630–15649, 2022. URL https://proceedings.mlr.press/v162/mindermann22a.html.

Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8162–8171, 2021. URL https://proceedings.mlr.press/v139/nichol21a.html.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4195–4205, 2023. URL https://openaccess.thecvf.com/content/ICCV2023/html/Peebles_Scalable_Diffusion_Models_with_Transformers_ICCV_2023_paper.html.

Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. ZeRO: Memory optimizations toward training trillion parameter models. In *International Conference for High Performance Computing, Networking, Storage and Analysis*, pp. 1–16, 2020. doi: 10.1109/SC41405.2020.00024.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10684–10695, 2022. URL https://openaccess.thecvf.com/content/CVPR2022/html/Rombach_High-Resolution_Image_Synthesis_With_Latent_Diffusion_Models_CVPR_2022_paper.html.

Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=TIdIXIpzhoI.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training GANs. In *Advances in Neural Information Processing Systems*, volume 29, pp. 2234–2242, 2016. URL https://papers.nips.cc/paper_files/paper/2016/hash/8a3363abe792db2d8761d6403605aeb7-Abstract.html.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021a. URL https://openreview.net/forum?id=St1giarCHLP.

Yang Song and Prafulla Dhariwal. Improved techniques for training consistency models. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/41bd71e7bf7f9fe68f1c936940fd06bd-Abstract-Conference.html.

Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021b. URL https://openreview.net/forum?id=PxTIG12RRHS.

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In *International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 32211–32252, 2023. URL https://proceedings.mlr.press/v202/song23a.html.

Sony Research. CMT reference implementation. GitHub repository, 2026. URL https://github.com/sony/cmt.

Pengwei Sun. Curriculum sampling: A two-phase curriculum for efficient training of flow matching. In *Second DeLTa Workshop at ICLR 2026*, 2026. URL https://openreview.net/pdf/1de78a0c02bda148627255c94773d683403da719.pdf.

Songtao Tian, Guhan Chen, Bohan Li, Jingyi Ma, and Zixiong Yu. Curvature-adaptive consistency flow matching: Autonomous trajectory optimization via reinforcement learning. *arXiv preprint arXiv:2606.22394*, 2026. URL https://arxiv.org/abs/2606.22394. Accepted by ECCV 2026.

Fu-Yun Wang, Zhaoyang Huang, Alexander William Bergman, Dazhong Shen, Peng Gao, Michael Lingelbach, Keqiang Sun, Weikang Bian, Guanglu Song, Yu Liu, Xiaogang Wang, and Hongsheng Li. Phased consistency models. In *Advances in Neural Information Processing Systems*, volume 37, pp. 83951–84009, 2024. doi: 10.52202/079017-2668. URL https://papers.nips.cc/paper_files/paper/2024/hash/98a29475083c502c34949f9baa1aa2ef-Abstract-Conference.html.

Frank Yates and P. M. Grundy. Selection without replacement from within strata with probability proportional to size. *Journal of the Royal Statistical Society: Series B (Methodological)*, 15(2): 253–261, 1953. doi: 10.1111/j.2517-6161.1953.tb00140.x.

Huijie Zhang, Aliaksandr Siarohin, Willi Menapace, Michael Vasilkovsky, Sergey Tulyakov, Qing Qu, and Ivan Skorokhodov. AlphaFlow: Understanding and improving MeanFlow models. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/e8c20cafe841cba3e31a17488dc9c3f1-Abstract-Conference.html.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 586–595, 2018. URL https://openaccess.thecvf.com/content_cvpr_2018/html/Zhang_The_Unreasonable_Effectiveness_CVPR_2018_paper.html.

Mingyuan Zhou, Huangjie Zheng, Zhendong Wang, Mingzhang Yin, and Hai Huang. Score identity distillation: Exponentially fast distillation of pretrained diffusion models for one-step generation. In *International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 62307–62331, 2024. URL https://proceedings.mlr.press/v235/zhou24x.html.

A Theoretical analysis and proofs

A.1 Finite-population variance under uniform sparsification

The inclusion probability $\Pr(c \in A) = S/N$ and linearity of expectation give $\mathbb{E}[\widehat{g}_A \mid \tau, \theta] = \bar{g}$, proving conditional unbiasedness in Proposition 1. The full covariance derivation follows.

Condition on the complete teacher grid and all randomness collected in τ. Write $g_c = \nabla_\theta \ell_c(\theta)$ and $\bar{g} = N^{-1} \sum_c g_c$. Sparse-uniform draws a simple random sample without replacement (Cochran, 1977). For any fixed direction v,

$$\mathrm{Var}_A\big(v^\top \widehat{g}_A \mid \tau\big) = \frac{1 - S/N}{S} \frac{1}{N-1} \sum_{c \in \Omega} \big[v^\top (g_c - \bar{g})\big]^2 . \tag{8}$$

The factor $1 - S/N$ is the finite-population correction. It vanishes for Dense, and for $S = N/2$ its variance relative to an equally sized with-replacement sample is $N/[2(N-1)]$. Equation (8) does not imply that sparse training follows the same realized optimization path as Dense, as it characterizes one conditional update, while later parameters, teacher noise, and optimizer states diverge.

To verify Eq. (8), let $I_c = \mathbf{1}\{c \in A\}$. Under simple random sampling, $\mathbb{E}[I_c] = S/N$, $\mathrm{Var}(I_c) = S(N-S)/N^2$, and $\mathrm{Cov}(I_c, I_d) = -S(N-S)/(N^2(N-1))$ for $c \neq d$. Substitution into $\widehat{g}_A = S^{-1} \sum_c I_c g_c$ yields the vector covariance

$$\mathrm{Cov}_A(\widehat{g}_A \mid \tau) = \frac{1}{S^2} \sum_{c \in \Omega} \mathrm{Var}(I_c) g_c g_c^\top + \frac{1}{S^2} \sum_{c \neq d} \mathrm{Cov}(I_c, I_d) g_c g_d^\top \tag{9}$$

$$= \frac{1 - S/N}{S} \frac{1}{N-1} \sum_{c \in \Omega} (g_c - \bar{g})(g_c - \bar{g})^\top , \tag{10}$$

from which Eq. (8) follows by projection.

Remark 1 (Non-uniform exposure and importance correction)**.** *For fixed-size sampling with inclusion probabilities $\pi_i > 0$, the unweighted mean has bias*

$$b = \sum_i (\pi_i / S - 1/N) g_i . \tag{11}$$

This follows by taking the expectation of each inclusion indicator. The bias is generally nonzero, but cancels for some gradient populations, including identical g_i. The Horvitz–Thompson estimator $\widehat{g}_{\mathrm{HT}} = N^{-1} \sum_{i \in A} g_i / \pi_i$ is unbiased (Horvitz & Thompson, 1952). For a fixed sample size its covariance admits the Sen–Yates–Grundy representation (Yates & Grundy, 1953)

$$\mathrm{Cov}(\widehat{g}_{\mathrm{HT}}) = N^{-2} \sum_{i<j} (\pi_i \pi_j - \pi_{ij}) d_{ij} d_{ij}^\top, \quad d_{ij} = g_i/\pi_i - g_j/\pi_j . \tag{12}$$

To derive this expression, expand the covariance of the inclusion-weighted sum and use $\sum_j (\pi_{ij} - \pi_i \pi_j) = 0$, with $\pi_{ii} = \pi_i$, which follows from the fixed cardinality. Small inclusion probabilities can create large weights, but this formula does not imply a universal variance penalty relative to uniform sampling. TVA/LSM change exposure without this correction, so their FID differences cannot be assigned to variance alone.

A.2 Two-Step Trajectory Reuse and Local Parameter Drift Bound

Under cross-step trajectory reuse with refresh stride $R = 2$, parameter update $t+1$ evaluates the student loss on the complementary subset $\mathcal{S}_{t,2}$ drawn from the trajectory pool Ω_t generated by the frozen teacher at step t. Over this interval, student parameters advance from θ_t to $\theta_{t+1} = \theta_t - \eta_t \mathbf{v}_t$, where the update vector $\mathbf{v}_t$ depends on the first slice $\mathcal{S}_{t,1}$. In contrast to online training where each step draws an independent mini-batch, trajectory reuse constitutes a two-step mini-batch reuse protocol across Ω_t. We formalize the local parameter drift induced by this temporal displacement below.

Proposition 3 (**Local Parameter Drift Bound under Trajectory Reuse**)**.** *Assume the per-cell loss function $\ell_c(\theta) = \mathrm{LPIPS}(f_\theta(\mathbf{x}_c, t_c), \hat{\mathbf{x}}_{0,c})$ is L-smooth with respect to θ, i.e., $\|\nabla \ell_c(\theta_1) - \nabla \ell_c(\theta_2)\| \leq L \|\theta_1 - \theta_2\|$ for all θ_1, θ_2, and the stochastic student update vector satisfies $\|\mathbf{v}_t\| \leq G_{\max}$. Then the following properties hold.*

(i) For any cell $c \in \Omega_t$, the gradient shift between parameter states θ_t and θ_{t+1} evaluated on the cached teacher target satisfies

$$\|\nabla \ell_c(\theta_{t+1}) - \nabla \ell_c(\theta_t)\| \le L\|\theta_{t+1} - \theta_t\| = L\eta_t\|\mathbf{v}_t\| \le L\eta_t G_{\max}. \tag{13}$$

Averaging over the complementary supervision slice $\mathcal{S}_{t,2} \subset \Omega_t$, the per-step gradient drift vector $\mathbf{\Delta}_{t+1} = \nabla \mathcal{L}_S(\theta_{t+1}; \mathcal{S}_{t,2}) - \nabla \mathcal{L}_S(\theta_t; \mathcal{S}_{t,2})$ likewise satisfies $\|\mathbf{\Delta}_{t+1}\| \le L\eta_t G_{\max}$.

(ii) Across a fine-tuning horizon of T steps, set $\mathbf{\Delta}_{t+1} = 0$ when update $t+1$ uses a fresh pool. The cumulative displacement bound then satisfies

$$\sum_{t=0}^{T-1} \|\mathbf{\Delta}_{t+1}\|^2 \le L^2 G_{\max}^2 \sum_{t=0}^{T-1} \eta_t^2. \tag{14}$$

If the terminal rate satisfies $\eta_{T-1} = \eta_{\min}$, the terminal update displacement is bounded by $\|\theta_T - \theta_{T-1}\| \le \eta_{\min} G_{\max}$, providing a local displacement bound without implying stationary dynamics or convergence.

Proof. For any sample $c \in \Omega_t$, by L-smoothness of $\ell_c(\cdot)$, we have $\|\nabla \ell_c(\theta_{t+1}) - \nabla \ell_c(\theta_t)\| \le L\|\theta_{t+1} - \theta_t\|$. Substituting the parameter update $\theta_{t+1} - \theta_t = -\eta_t \mathbf{v}_t$ gives $\|\nabla \ell_c(\theta_{t+1}) - \nabla \ell_c(\theta_t)\| \le L\eta_t\|\mathbf{v}_t\| \le L\eta_t G_{\max}$. By Jensen's inequality applied to the norm, the average drift across slice $\mathcal{S}_{t,2}$ satisfies $\|\mathbf{\Delta}_{t+1}\| \le \frac{1}{S}\sum_{c \in \mathcal{S}_{t,2}} \|\nabla \ell_c(\theta_{t+1}) - \nabla \ell_c(\theta_t)\| \le L\eta_t G_{\max}$. Summing the squared bounds over T steps yields the result. □

Remark 2 (Complementary updates at changing parameters)**.** *Let h_1, h_2 be the mean gradients on equal complementary halves of a fixed pool. The identity $(h_1(\theta) + h_2(\theta))/2 = \bar{g}(\theta)$ holds for every partition. Therefore*

$$\left\| \frac{h_1(\theta_t) + h_2(\theta_{t+1})}{2} - \bar{g}(\theta_t) \right\| \le \frac{L}{2}\|\theta_{t+1} - \theta_t\| \le \frac{L\eta_t G_{\max}}{2}. \tag{15}$$

The first inequality follows by subtracting the fixed-parameter identity and applying smoothness to h_2; it holds pathwise despite the dependence between θ_{t+1} and the partition. It bounds a two-gradient average, not the RAdam update or its convergence rate. The pool is refreshed every two updates, so its age does not grow with the total training horizon. Frozen teacher targets do not themselves change with student parameters. A smaller learning rate reduces this local upper bound under a common bound on the update direction; it does not prove proportional empirical drift, zero bias, or long-run convergence. A positive terminal learning rate also provides no infinite-horizon square-summability guarantee.

A.3 Conditional Resource Scaling

For fixed architecture, precision, batch size and execution strategy, consider the first-order model

$$T(K, R, S) = T_0 + bK/R + cS, \tag{16}$$

$$M(K, S) = A(K) + aS, \qquad T_0, A(K), a, b, c \ge 0. \tag{17}$$

The fixed terms include parameters, optimizer state and common overhead; $A(K)$ also includes teacher buffers. Real kernels can depart from these affine approximations. Under this model the following resource identities hold.

Proposition 4 (Savings under an affine memory model)**.** *For positive baseline memory and time, if $S' = \rho S$ with $0 \le \rho \le 1$, then*

$$\frac{M(K, S) - M(K, \rho S)}{M(K, S)} = (1 - \rho)\phi, \qquad \phi = \frac{aS}{A(K) + aS} \in [0, 1]. \tag{18}$$

Thus halving supervision saves at most half of peak memory; a saving above 40% requires $\phi > 0.8$. At fixed K, S, increasing R from 1 to 2 saves $(bK/2)/(T_0 + bK + cS)$ of step time, which is at most one half.

Proof. Subtract the two affine expressions for memory to obtain $aS(1-\rho)$ and divide by the positive baseline. The range of ϕ follows from nonnegative coefficients. Subtract the corresponding time expressions to obtain $bK/2$; division gives the stated ratio. □

These identities explain why memory and time can respond differently. They do not prove that $\phi > 0.8$ on another architecture, that activation storage is exactly linear, or that a particular device can run the complete pipeline. LDM (Rombach et al., 2022) and DiT (Peebles & Xie, 2023) require independently measured coefficients, including the effects of checkpointing (Chen et al., 2016) and attention implementation (Dao et al., 2022). We therefore report measured CMT allocation and omit uncalibrated architecture-specific GiB projections.

As a complementary baseline, gradient accumulation preserves the mean objective of all retained cells while splitting student execution into microbatches. It can be combined with teacher reuse. We compare these interventions on the actual CMT student, frozen teacher and LPIPS objective using synchronized timing and allocated/reserved memory measurements. The resource-only benchmark and its scope are reported below. See Section 4.6 for the resource comparison.

A.4 Supervision density and controller analysis

The method combines a finite-population estimator with a delayed controller. The following analysis establishes the per-update properties needed to distinguish cardinality control from temporal curricula; detailed derivations and alternative estimators appear in Appendix A.9.

A.5 A closed-form supervision-density rule

Define the within-grid dispersion and squared Dense-gradient magnitude as

$$V = \frac{1}{N-1} \sum_{c \in \Omega} \|g_c - \bar{g}\|_2^2, \qquad G = \|\bar{g}\|_2^2. \tag{19}$$

Taking the trace of Eq. (10) gives

$$\mathbb{E}_A \left[\|\widehat{g}_A - \bar{g}\|_2^2 \mid \tau \right] = \left(\frac{1}{S} - \frac{1}{N} \right) V. \tag{20}$$

Corollary 1 (minimum density for relative gradient fidelity). For tolerance $\varepsilon > 0$, $V > 0$, and $G > 0$, the smallest uniformly sampled subset satisfying $\mathbb{E}_A[\|\widehat{g}_A - \bar{g}\|_2^2 \mid \tau] \le \varepsilon G$ is

$$S_\varepsilon = \mathrm{clip}_{[S_{\min}, S_{\max}]} \left\lceil \frac{NV}{V + N\varepsilon G} \right\rceil. \tag{21}$$

The guarantee requires $\lceil NV/(V + N\varepsilon G) \rceil \le S_{\max}$; otherwise the admissible interval contains no feasible count and clipping does not satisfy the tolerance. *Proof.* Substitute Eq. (20), solve $V/S - V/N \le \varepsilon G$ for S, take the smallest integer satisfying the inequality, and apply the admissible bounds. □

If $V = 0$, every cell gradient is identical and $S_{\min}$ is sufficient. If $G = 0 < V$, the inequality requires $S = N$; it is infeasible if $S_{\max} < N$. A delayed controller can replace G and V by exponential moving averages from completed probes, multiply V by a safety factor $\gamma \ge 1$, and choose

$$S_{h+1} = \mathrm{clip}_{[S_{\min}, S_{\max}]} \left\lceil \frac{N\gamma \bar{V}_h}{\gamma \bar{V}_h + N\varepsilon \bar{G}_h} \right\rceil. \tag{22}$$

Because S_{h+1} is measurable with respect to completed history and the cells are still sampled uniformly, Proposition 1 continues to hold conditional on that history. Equation (22) therefore adapts cardinality without introducing the exposure bias of TVA or LSM. Computing full per-cell parameter gradients would erase the intended saving; a practical controller must use a validated proxy. The present results evaluate the fixed operating point $S = 64$; proxy fidelity and dynamic density require a separate controlled evaluation.

A.6 Adaptive sampling as a moving objective

For scheduler history $\mathcal{H}_h$, let $\pi_{h,c}$ denote each cell's inclusion probability. Because every selected set contains S cells, $\sum_c \pi_{h,c} = S$. Define the conditional surrogate

$$\widetilde{\mathcal{L}}_h(\theta) = \frac{1}{S} \sum_{c \in \Omega} \pi_{h,c} \ell_c(\theta). \tag{23}$$

When $\mathcal{H}_h$ is fixed, Eq. (49) is exactly $\nabla_\theta \widetilde{\mathcal{L}}_h$. TVA and LSM therefore create a sequence of piecewise-constant curriculum objectives. They are generally biased relative to the Dense gradient. Uniform inclusion guarantees zero bias for every gradient population; some nonuniform populations can also have cancelling bias.

This interpretation changes the evaluation question. Lower variance would be judged against Dense under the same expected objective. A curriculum must instead be judged against Sparse-uniform at the same cell budget because both estimator bias and exposure have changed. Importance correction could restore the dense target in expectation, but it would create a different algorithm and potentially large weights near the exploration floor; it is analyzed only as an alternative in the supplement.

A.7 Bounded scores, nonzero support, and controller memory

Clipping gives $|z_h(i)| \le c$. With $a_0(i) = 0$, unrolling the EMA yields

$$a_h(i) = (1-\beta)\sum_{u=1}^{h} \beta^{h-u} z_u(i), \qquad |a_h(i)| \le c(1-\beta^h) \le c. \tag{24}$$

Thus finite probe inputs produce finite scores before softmax. The uniform mixture then gives $q_h(i) \ge \alpha/K$, and the inclusion-support bound gives nonzero cell inclusion. For the formal values, the conservative per-step cell-inclusion lower bound is approximately 0.0488. This is a support guarantee rather than a balance guarantee, as the remaining $1-\alpha$ mass can still concentrate strongly.

The controller also has a quantifiable memory. At $\beta = 0.8$, its nominal effective window is five probe events and its half-life is 3.11 events. With $H = 100$, a shock to a robust score retains half its influence for about 311 optimizer steps. Decreasing H or β would improve responsiveness but increase probe cost or score variance. Because the formal benchmark tests one controller setting, it establishes reproducibility of that setting, not optimality of its response time.

A.8 Complexity and attainable savings

Teacher integration remains $O(BK)$ solver-state work. Dense student evaluation is $O(BK)$ in the number of supervised cells, while sparse student evaluation is $O(\rho BK)$. The direct successive sampler is $O(NS) = O(\rho N^2)$ at fixed $N = BK$, and a probe event adds $O(MK)$ no-gradient student evaluations once every H steps. Averaged over updates, the controllable work is therefore

$$O(BK) + O(\rho BK) + O(MK/H) + O(\rho B^2 K^2). \tag{25}$$

The final term describes the simple CPU list sampler, not GPU model computation, and is negligible for $N = 128$. If B or K grows substantially, a top-k weighted sampler would be needed before the same implementation could support a scalability claim.

The accounting model explains why memory and time need not scale together. Sparse autograd can remove a large fraction of student activations, while the teacher solve, input pipeline, optimizer update, and evaluation remain. The observed result is consistent with this asymmetry, as memory approaches the ideal student-side reduction, but wall time is bounded by fixed work. A kernel-level decomposition would be required to estimate C_T, C_S, and C_O separately.

A.9 Additional adaptive-estimator derivations

A.10 The conditional objective under adaptive exposure

For scheduler state $\mathcal{H}_h$, let $\pi_{h,c} = \Pr(c \in A \mid \mathcal{H}_h)$ be the cell-inclusion probability under sequential sampling. The conditional objective is

$$\widetilde{\mathcal{L}}_h(\theta) = \frac{1}{S}\sum_{c\in\Omega} \pi_{h,c}\ell_c(\theta). \tag{26}$$

Because every subset has exactly S elements, $\sum_c \pi_{h,c} = S$ and the coefficients in Eq. (26) form a normalized exposure distribution. Conditional on $\mathcal{H}_h$ and τ, Eq. (49) equals $\nabla_\theta \widetilde{\mathcal{L}}_h$. Thus TVA and LSM are not merely variance-reduction devices, as they optimize a piecewise-constant sequence of

exposure-weighted objectives. This distinction is why Sparse-uniform is the necessary control for the adaptive methods.

An inverse-inclusion estimator could instead use the Horvitz–Thompson construction (Horvitz & Thompson, 1952),

$$\widehat{g}_{\mathrm{HT}} = \frac{1}{N} \sum_{c \in A} \frac{g_c}{\pi_{h,c}}, \tag{27}$$

which is conditionally unbiased when exact $\pi_{h,c}$ are available. We did not use this estimator. Successive weighted sampling makes exact inclusion probabilities nontrivial, the exploration floor can still produce large inverse weights, and changing the loss estimator would confound curriculum exposure with importance correction. It remains a separate experiment.

A.11 Nonzero exposure guarantee

Under Eq. (54), every cell has inclusion probability

$$\pi_{h,(b,i)} \geq 1 - \left(1 - \frac{\alpha}{KB}\right)^S \geq 1 - \exp\left(-\frac{S\alpha}{KB}\right) > 0. \tag{28}$$

Proof. While a cell remains available, its conditional draw probability in Eq. (48) is at least its initial weight $w_{b,i} = q_h(i)/B \geq \alpha/(KB)$ because the remaining-weight denominator is at most one. Multiplying the complementary probabilities over at most S draws yields the bound. □

The boundary prevents permanent starvation in the sampling model; it does not equalize finite-run exposure or guarantee optimization quality.

A.12 Quantitative meaning of the exploration floor

With $\alpha = 0.10$, $K = 16$, $B = 8$, and $S = 64$, the inclusion-support bound gives

$$\pi_{h,c} \geq 1 - \left(1 - \frac{0.10}{128}\right)^{64} \approx 0.0488. \tag{29}$$

Summing over the eight trajectories gives an expected lower bound of approximately 0.390 selected cells per time bin per optimizer step. Across 10,000 steps, the corresponding expected exposure lower bound is about 3,900 cell selections per bin. These are conservative expectations, not deterministic counts; the realized allocation is correlated by sampling without replacement.

The softmax component remains sensitive to score differences. Let $p_h = \mathrm{softmax}(a_h/T)$. Then

$$\frac{\partial q_h(i)}{\partial a_h(j)} = \frac{1-\alpha}{T} p_h(i) \left(\mathbf{1}\{i = j\} - p_h(j)\right). \tag{30}$$

Temperature $T = 0.5$ therefore sharpens a score difference, while $\alpha = 0.1$ reserves ten percent of total mass for uniform exploration. The floor bounds starvation; it does not bound the variance of realized counts tightly enough to establish equal coverage.

A.13 EMA memory and feedback delay

Unrolling the TVA score recursion gives

$$a_h(i) = \beta^h a_0(i) + (1-\beta) \sum_{u=1}^{h} \beta^{h-u} z_u(i). \tag{31}$$

For $\beta = 0.8$, the nominal effective window is $1/(1-\beta) = 5$ probe events and the half-life is $\log(1/2)/\log(0.8) = 3.11$ events, or about 311 optimizer steps at $H = 100$. The distribution used at an ordinary step is measurable with respect to completed probe history. This prevents same-step look-ahead, but it also means abrupt changes in the useful time region are observed and acted upon only after the probe interval and EMA lag.

A.14 MEAN-SQUARED ERROR OF UNIFORM SPARSIFICATION

Conditional unbiasedness makes the mean-squared error of Sparse-uniform equal to the trace of Eq. (10),

$$\mathbb{E}_A\big[\|\widehat{g}_A - \bar{g}\|_2^2 \mid \tau\big] = \mathrm{tr}[\mathrm{Cov}_A(\widehat{g}_A \mid \tau)] \tag{32}$$

$$= \frac{1 - S/N}{S} \frac{1}{N-1} \sum_{c \in \Omega} \|g_c - \bar{g}\|_2^2. \tag{33}$$

With $S = \rho N$, the density coefficient is $(1-\rho)/(\rho N)$. It is zero at $\rho = 1$, equals $1/N$ at $\rho = 1/2$, and rises to $3/N$ at $\rho = 1/4$. Thus the additional update variance grows nonlinearly as supervision becomes sparse, even if the within-grid gradient dispersion is fixed.

If $\|g_c\|_2 \le G_{\max}$ for all cells, then $\|g_c - \bar{g}\|_2 \le 2G_{\max}$ and

$$\mathbb{E}_A\big[\|\widehat{g}_A - \bar{g}\|_2^2 \mid \tau\big] \le \frac{1-\rho}{\rho N} \frac{4NG_{\max}^2}{N-1}. \tag{34}$$

The bound isolates the roles of density, grid size, and gradient heterogeneity; it is not converted into a convergence or FID bound because G and the cross-update dependence are not measured.

Variance multiplier across retention budgets. The density coefficient $\kappa(\rho) = (1-\rho)/(\rho N)$ of Eq. (33) grows nonlinearly as supervision becomes sparser, taking values 0, $1/N$, $1.67/N$, $3/N$ and $7/N$ at $\rho \in \{1.0, 0.5, 0.375, 0.25, 0.125\}$. The generation-quality experiments cover $\rho \in \{1, 2/3, 1/2, 3/8\}$, with $N = 128$ or 96 and $S \in \{128, 64, 48\}$ as specified for each configuration. Smaller fractions here illustrate the variance model. The separate fixed-pool gradient audit below tests further values of S but does not establish generation quality or a safe training budget at those values.

A.15 WHITE-BOX PARAMETER GRADIENT AUDIT AND EMPIRICAL VALIDATION OF PROPOSITION 1

To directly validate Proposition 1, Equation (8), and the Finite Population Correction (FPC) law at the level of neural network parameters, we conducted a white-box gradient extraction audit on the student network. We recorded cell-level parameter gradients $g_c = \nabla_\theta \ell_c(\theta)$ across all $N = BK = 96$ individual cells ($B = 8, K = 12$) on an RTX 3090 GPU, tracking 1,745,155 parameters selected by parameter-name substrings `32x32_aux`, `32x32_block4.conv`, and `map_layer1`. The population variance trace is $V = \frac{1}{N-1}\sum_{c=1}^{N} \|g_c - \bar{g}\|_2^2 = 0.28501$, and the mean population gradient squared norm is $\|\bar{g}\|_2^2 = 0.01961$.

Exact FPC Decay Across Retention Budgets. For each retention budget $S \in \{8, 16, 24, 32, 40, 48, 56, 64, 72, 80, 88, 96\}$, we computed the theoretical FPC variance $\frac{1-S/N}{S}V$, the analytic with-replacement variance $\frac{N-1}{NS}V$, and the empirical variance across 100 independent random subsets drawn without replacement; theoretical predictions closely match empirical variances across all budgets, vanishing strictly at full retention (Table A1).

Zero-Variance Property of Complementary Partitioning ($\mathcal{C}_R$) and Dynamic Drift. Under the complementary reuse operator $\mathcal{C}_{R=2}$ at the tested workpoint $S = 48$, the full grid Ω ($N = 96$) is partitioned into two disjoint halves $\mathcal{S}_1 \cap \mathcal{S}_2 = \emptyset$ such that $\mathcal{S}_1 \cup \mathcal{S}_2 = \Omega$. In a static evaluation setting where the parameter vector θ is held fixed, the cumulative gradient estimator across the partition satisfies $\frac{1}{2}(\widehat{g}_{\mathcal{S}_1}(\theta) + \widehat{g}_{\mathcal{S}_2}(\theta)) \equiv \bar{g}(\theta)$. In our empirical white-box parameter audit, the computed variance is -2.13×10^{-18}, which vanishes to numerical zero within machine precision. This behavior aligns exactly with the fixed-parameter identity and Proposition 1, contrasting sharply with independent random sampling without replacement (empirical variance 1.494×10^{-3}, predicted 1.484×10^{-3}).

In dynamic online training, the student parameter vector is updated between partition steps ($\theta_{t+1} = \theta_t - \eta_t \mathbf{v}_t$), where $\mathbf{v}_t$ is the actual optimizer update direction. Under the same local smoothness

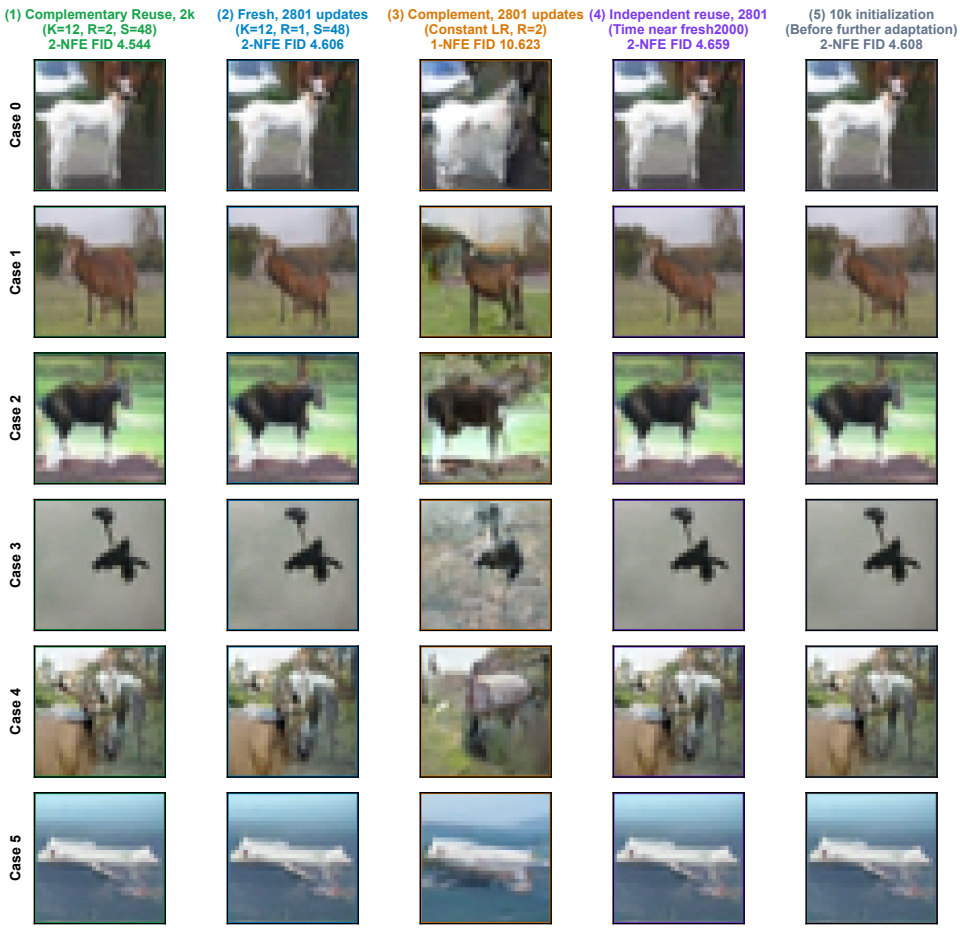


Figure A1: Qualitative checkpoint comparison across six fixed cases. Columns show (1) complement2000, 2-NFE FID 4.544; (2) fresh2801, 2-NFE FID 4.606; (3) constant-rate complement2801, 1-NFE FID 10.623; (4) independent reuse2801, 2-NFE FID 4.659, with loop time approximately matched to fresh2000; and (5) the 10k initialization before further adaptation, 2-NFE FID 4.608. These selected examples illustrate endpoints, not a causal mechanism of degradation.

Table A1: **White-Box Parameter Gradient Variance Audit Across Retention Budgets ($N = 96$, 1.745M Parameters).** Empirical squared errors are summarized over 100 random partition draws; their spread uses denominator 100, as in the stored calculation. Theoretical FPC matches empirical variance to within 2.3% at the sparsest tested budget ($S = 8$) and to within 1.3% or better at all budgets from $S = 24$ upward. At full retention ($S = 96$), empirical variance strictly vanishes to machine precision ($< 10^{-17}$), whereas the analytic with-replacement variance is 2.94×10^{-3}.

Budget S	Ratio S/N	FPC $(1 - S/N)$	Theoretical FPC	Empirical Variance	With-replacement variance	FPC Match (%)
8	0.083	0.917	3.266×10^{-2}	$(3.340 \pm 2.155) \times 10^{-2}$	3.526×10^{-2}	97.7%
16	0.167	0.833	1.484×10^{-2}	$(1.477 \pm 0.786) \times 10^{-2}$	1.763×10^{-2}	99.5%
24	0.250	0.750	8.907×10^{-3}	$(9.015 \pm 4.551) \times 10^{-3}$	1.175×10^{-2}	98.8%
32	0.333	0.667	5.938×10^{-3}	$(5.919 \pm 2.837) \times 10^{-3}$	8.814×10^{-3}	99.7%
40	0.417	0.583	4.156×10^{-3}	$(4.210 \pm 2.021) \times 10^{-3}$	7.051×10^{-3}	98.7%
48	0.500	0.500	2.969×10^{-3}	$(2.964 \pm 1.356) \times 10^{-3}$	5.876×10^{-3}	99.8%
56	0.583	0.417	2.121×10^{-3}	$(2.103 \pm 0.977) \times 10^{-3}$	5.037×10^{-3}	99.1%
64	0.667	0.333	1.484×10^{-3}	$(1.487 \pm 0.697) \times 10^{-3}$	4.407×10^{-3}	99.9%
72	0.750	0.250	9.896×10^{-4}	$(9.865 \pm 4.738) \times 10^{-4}$	3.917×10^{-3}	99.7%
80	0.833	0.167	5.938×10^{-4}	$(5.921 \pm 3.114) \times 10^{-4}$	3.526×10^{-3}	99.7%
88	0.917	0.083	2.699×10^{-4}	$(2.682 \pm 1.697) \times 10^{-4}$	3.205×10^{-3}	99.4%
96	1.000	0.000	0.000	$< 10^{-17}$	2.938×10^{-3}	100.0%

assumption and bound $\|\mathbf{v}_t\| \le G_{\max}$ as in the main text, the inter-step drift satisfies

$$\frac{1}{2}\left(\widehat{g}_{\mathcal{S}_1}(\theta_t) + \widehat{g}_{\mathcal{S}_2}(\theta_{t+1})\right) = \bar{g}(\theta_t) + \mathcal{O}(L\eta_t G_{\max}). \tag{35}$$

The zero-variance identity assumes a fixed parameter vector. With changing parameters, Proposition 2 instead bounds the local discrepancy under its smoothness and bounded-update assumptions. This motivates testing smaller learning rates but does not imply monotonic measured drift or convergence.

Measured parameter drift. The stored experiment evaluates a mixed-parameter gradient average, rather than a per-cell gradient shift. Let $h_1, h_2, \bar{g}$ denote half-pool and dense gradients projected onto the measured coordinates and represented with zeros elsewhere. For one fixed pool it perturbs only those coordinates, $\theta_\eta = \theta - \eta\bar{g}(\theta)$, and records

$$E(\eta) = \left\| \frac{h_1(\theta) + h_2(\theta_\eta)}{2} - \bar{g}(\theta_\eta) \right\|_2. \tag{36}$$

The fixed-parameter partition identity reduces this difference to $[h_1(\theta) - h_1(\theta_\eta)]/2$, so a local smoothness bound applies, but the experiment does not estimate its constant. Five BF16 evaluations over $[10^{-5}, 10^{-4}]$ give a linear-fit $r^2 = 0.050$ and do not establish proportionality. The recorded discrepancy is 0.00269 at $\eta = 10^{-5}$ versus 0.00361 at 5×10^{-5}, a 25.5% difference. These are finite-precision, selected-coordinate observations on one pool, not RAdam training trajectories or a test of long-run convergence. The learning-rate annealing comparison constitutes a separate endpoint experiment (Table A4).

A.16 Variance-calibrated cardinality and its conditional guarantee

Let $V = (N-1)^{-1}\sum_c \|g_c - \bar{g}\|_2^2$ and $G = \|\bar{g}\|_2^2$. Before integer rounding and clipping, Eq. (21) selects density

$$\rho_\varepsilon = \frac{S_\varepsilon}{N} = \frac{V}{V + N\varepsilon G}. \tag{37}$$

This expression exposes the intended behavior. For $V, G, \varepsilon > 0$,

$$\frac{\partial \rho_\varepsilon}{\partial V} = \frac{N\varepsilon G}{(V + N\varepsilon G)^2} > 0, \quad \frac{\partial \rho_\varepsilon}{\partial G} = -\frac{VN\varepsilon}{(V + N\varepsilon G)^2} < 0, \quad \frac{\partial \rho_\varepsilon}{\partial \varepsilon} = -\frac{VNG}{(V + N\varepsilon G)^2} < 0. \tag{38}$$

The controller therefore requests more cells when gradients are more dispersed, and fewer cells when the mean direction is stronger or the tolerance is looser. The ceiling operation in Eq. (21) makes the finite-grid constraint tight. If the result lies strictly inside $[S_{\min}, S_{\max}]$, then S_ε satisfies the tolerance and $S_\varepsilon - 1$ does not.

A delayed implementation observes proxy statistics $(\widehat{G}_h, \widehat{V}_h)$ at probe event h and updates

$$\bar{G}_h = \lambda \bar{G}_{h-1} + (1-\lambda)\widehat{G}_h, \qquad \bar{V}_h = \lambda \bar{V}_{h-1} + (1-\lambda)\widehat{V}_h, \tag{39}$$

with $0 \le \lambda < 1$. It then applies Eq. (22) using safety factor $\gamma \ge 1$. The following statement makes the role of the proxy explicit.

Proposition B.1 (one-step conditional fidelity). Suppose the next complete grid satisfies $V_{h+1} \le \gamma \bar{V}_h$ and $G_{h+1} \ge \bar{G}_h > 0$. If clipping at $S_{\max}$ does not prevent the tolerance from being met, then the uniformly sampled update with count S_{h+1} from Eq. (22) satisfies

$$\mathbb{E}_A\big[\|\widehat{g}_{A,h+1} - \bar{g}_{h+1}\|_2^2 \mid \mathcal{H}_h, \tau_{h+1}\big] \le \varepsilon G_{h+1}. \tag{40}$$

Proof. Equation (22) ensures $(1/S_{h+1} - 1/N)\gamma \bar{V}_h \le \varepsilon \bar{G}_h$ before any active upper clipping. Applying $V_{h+1} \le \gamma \bar{V}_h$, then Eq. (20), and finally $\bar{G}_h \le G_{h+1}$ proves the result. □

The proposition is deliberately conditional. An output-gradient proxy need not upper-bound full parameter gradient dispersion, and an EMA need not predict the next grid. A checkpoint-based validation must therefore test the two envelope assumptions rather than treat Eq. (40) as automatic. Regardless of proxy accuracy, choosing S_{h+1} from completed history and sampling uniformly preserves conditional unbiasedness; proxy error affects the requested tolerance, not the expected objective.

A.17 Exact decomposition of the adaptive target shift

Let $u_c = 1/N$ and $v_{h,c} = \pi_{h,c}/S$ denote the Dense and adaptive exposure coefficients. Both sum to one. Their conditional gradient difference is

$$b_h(\theta) = \nabla \widetilde{\mathcal{L}}_h(\theta) - \nabla \widehat{\mathcal{L}}_D(\theta) = \sum_{c \in \Omega} (v_{h,c} - u_c) g_c. \tag{41}$$

Because $\sum_c (v_{h,c} - u_c) = 0$, subtracting $\bar{g}$ and applying the triangle inequality gives

$$\|b_h(\theta)\|_2 \le 2 D_h \max_c \|g_c - \bar{g}\|_2, \qquad D_h = \frac{1}{2}\sum_c |v_{h,c} - u_c|. \tag{42}$$

Adaptive exposure changes the conditional target only when the exposure distribution departs from uniform and cell gradients differ. Scheduler entropy measures the first factor, not the second.

The adaptive estimator's conditional error relative to Dense decomposes as

$$\mathbb{E}_A\big[\|\widehat{g}_A - \bar{g}\|_2^2 \mid \tau, \mathcal{H}_h\big] = \mathrm{tr}[\mathrm{Cov}_A(\widehat{g}_A \mid \tau, \mathcal{H}_h)] + \|b_h(\theta)\|_2^2. \tag{43}$$

An adaptive rule may reduce sampling variance while increasing target bias, or increase both terms. Separating them empirically requires cell-level gradients and first-order inclusion probabilities, which are not stored in the current runs.

A.18 Finite-horizon coverage

Let $I_{u,c}$ indicate selection of a fixed grid position $c = (b, i)$ at update u, rather than selection of the same data example from every refreshed pool. The exploration floor gives a conditional inclusion lower bound $p_{\min} = 1 - (1 - \alpha/(BK))^S$ after any sampling history. Iterated conditioning on the event of no earlier selection gives

$$\Pr\left(\sum_{u=1}^{U} I_{u,c} = 0\right) \le (1 - p_{\min})^U \le \exp(-U p_{\min}). \tag{44}$$

This argument does not require independent updates or fixed adaptive weights. For the formal parameters, $p_{\min} \approx 0.0488$. At $U = 100$, the individual bound is approximately $e^{-4.88} \approx 0.0076$, while a union bound over $N = 128$ positions gives approximately 0.97, which is below one but provides little assurance of full coverage. The bound decreases over longer windows. Eventual exposure does not imply balanced or sufficient training, because low-probability bins can remain near the exploration floor and later updates can overwrite earlier contributions.

For time-bin exposure $X_{u,i} = \sum_{b=1}^{B} I_{u,(b,i)}$, linearity gives $\mathbb{E}[X_{u,i} \mid \mathcal{H}_u] \ge B p_{\min} \approx 0.390$. Across 10,000 updates this yields an expected lower bound of about 3,900 selected cells per bin. Within-step without-replacement dependence prevents treating this number as an independent-binomial confidence interval.

A.19 Perturbation stability of the EMA controller

Consider two TVA controllers with identical initialization and score sequences satisfying $\|z_h - z'_h\|_\infty \le \delta_h$. The EMA recursion gives

$$\|a_h - a'_h\|_\infty \le (1-\beta)\sum_{u=1}^{h} \beta^{h-u}\delta_u. \tag{45}$$

If every $\delta_u \le \delta$, then the score perturbation is at most δ. The softmax-plus-exploration Jacobian is $J_{ij} = (1-\alpha)p_i(\mathbf{1}\{i=j\} - p_j)/T$. Its absolute row sum is $2(1-\alpha)p_i(1-p_i)/T \le (1-\alpha)/(2T)$, giving a ℓ_∞ Lipschitz bound of $(1-\alpha)/(2T)$, equal to 0.9 under the formal parameters. Clipping, EMA smoothing, and exploration therefore keep bounded numerical perturbations from growing without limit, although they do not establish that probe rankings generalize.

B Additional experiments and complete comparisons

B.1 Unified Experimental Setup and Evaluation Protocol

Experiments use CIFAR-10 training images (Krizhevsky, 2009), a pretrained unconditional VP-EDM teacher (Karras et al., 2022), RAdam (Liu et al., 2020) and an LPIPS loss (Zhang et al., 2018). The recorded 10k supervision configuration uses initial learning rate 2×10^{-4} with linear decay and EMA coefficient 0.999. Continuation and post-training use their separately specified learning-rate schedules; their protocols are not interchangeable.

For the supervision and resolution benchmarks, runs execute on an NVIDIA A100 80-GB GPU across 10,000 optimization updates, with nominal and per-device batch sizes of 128, without gradient accumulation, using BF16 and TF32 arithmetic across three independent training seeds (31, 37, 43); this budget corresponds to 3.3% of the 300,000-step CMT reference schedule (Sony Research, 2026).

For trajectory reuse benchmarks, evaluations run on an NVIDIA RTX 3090 24-GB GPU measuring clean training loop time (strictly excluding serialization and offline evaluation overhead), comparing fresh generation and trajectory reuse under identical support and precision.

Regarding evaluation endpoints and metric standards, final quality endpoints evaluate 50,000 generated images against the 50,000-image CIFAR-10 reference set to compute Fréchet Inception Distance (FID) (Heusel et al., 2017) under FP32 inference. Early screens use FID10k; the additional 30k continuation curves use FID50k, as specified in Section B.5. The EDM feature-statistics pipeline (Karras et al., 2022) computes the covariance square-root term for FID in float64 on CPU. We report 1-NFE and 2-NFE FID, and Inception Score where available (Salimans et al., 2016), together with allocated memory and loop time. Different sample counts and timing boundaries are kept separate.

B.2 Student Supervision Sparsification and Memory Decoupling (S-Axis)

We first evaluate the impact of student supervision budget S under a fixed fine teacher grid ($K = 16$) and zero reuse ($R = 1$) (Table A2; Figure 2).

Table A2: Formal 10,000-step CIFAR-10 student supervision sparsification results. Values represent mean $\pm$ sample standard deviation across independent training seeds 31, 37, 43; for FID, Wall Time, and Memory, lower is better (↓); for Inception Score (IS), higher is better (↑). Memory is the reported logging-interval peak allocation, not a maximum over the entire process. Bold indicates the best observed value in each metric column.

Method	Active S	1-NFE FID↓	1-NFE IS↑	2-NFE FID↓	2-NFE IS↑	Wall Time (h)↓	Memory (GiB)↓
Dense	128	9.903 ± 0.137	9.28 ± 0.06	$\mathbf{4.745 \pm 0.079}$	$\mathbf{9.64 \pm 0.04}$	2.773 ± 0.079	9.543 ± 0.072
Sparse-uniform	64	10.018 ± 0.062	9.26 ± 0.05	4.788 ± 0.068	9.62 ± 0.05	2.616 ± 0.059	5.555 ± 0.052
TVA	64	10.121 ± 0.235	9.21 ± 0.08	4.844 ± 0.035	9.58 ± 0.05	2.577 ± 0.065	5.587 ± 0.006
LSM	64	$\mathbf{9.735 \pm 0.102}$	$\mathbf{9.31 \pm 0.05}$	4.862 ± 0.096	9.56 ± 0.06	$\mathbf{2.536 \pm 0.055}$	$\mathbf{5.540 \pm 0.069}$

Regarding memory reduction and uniform-sparse unbiasedness, empirical measurements corroborate our theoretical decoupling model (Section 3). Reducing supervision from Dense (128 cells) to

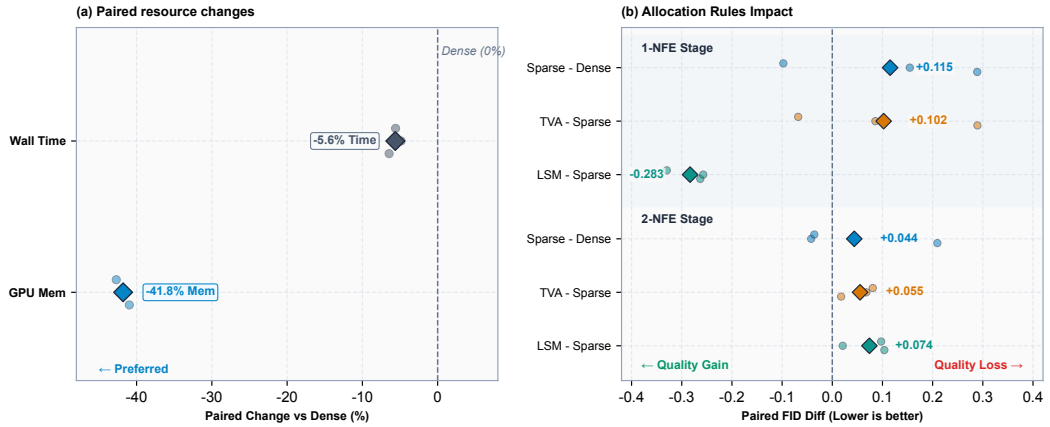


Figure A2: Paired sparse-supervision effects at 10k updates. (a) Resource changes relative to Dense. (b) FID differences for matched training seeds; negative values favor the candidate. Small points are the three training seeds and diamonds are their means. The paired measurements complement the aggregate resource and quality endpoints (Figure 2(a,b)).

Sparse-uniform (64 cells) reduces the reported interval peak allocation from 9.543 ± 0.072 GiB to 5.555 ± 0.052 GiB, a relative reduction of **41.8%** (Figure 2(a,b)). Because the teacher model must still execute complete ODE integrations, total wall-clock time decreases modestly by 5.6% (2.773h to 2.616h). In terms of generative quality, paired differences between Sparse-uniform and Dense are $+0.115$ on 1-NFE (95% CI $[-0.372, 0.603]$) and $+0.044$ on 2-NFE (95% CI $[-0.312, 0.400]$). Both intervals include improvements and degradations. Three seeds do not establish quality equivalence; the intervals quantify the remaining uncertainty.

Holding the student budget constant at $S = 64$ cells, we systematically evaluated whether online adaptive timestepping heuristics (TVA and LSM, detailed in Appendix A.4) could outperform uniform random sparsification. While non-uniform heuristics intuitively aim to allocate more capacity to "harder" diffusion timesteps, empirical measurements demonstrate that neither method Pareto-dominates uniform sampling across both quality endpoints (Figure A2(b)). Specifically, while TVA further reduced loop wall time by 1.5%, its 1-NFE FID degraded by 1.02% and 2-NFE FID degraded by 1.16%; similarly, while LSM reduced wall time by 3.1% and improved 1-NFE FID by 2.83%, its 2-NFE FID deteriorated by 1.54% (paired difference $+0.074$, 95% CI $[-0.041, 0.189]$). Because TVA and LSM alter the effective loss weighting without importance correction, their empirical trajectories reflect changed objective exposures rather than pure variance reduction. Uniform sampling thus provides the primary conditionally unbiased sparse baseline.

The independent resource control distinguishes gradient accumulation from sparse supervision (Figure 2(c,d)). Accumulation retains all 128 cells, whereas sparsification evaluates only the selected cells in each update. Teacher reuse and grid coarsening reduce different components of step time. These timing-block measurements do not supply quality endpoints for the 10k experiment. The paired seed-level resource and FID changes quantify within-experiment variation (Figure A2).

B.3 Teacher Discretization Resolution and Inference Endpoint Trade-off (K-Axis)

After measuring the memory response to S, we investigate teacher computational allocation under $R = 1$ by systematically varying teacher ODE resolution $K \in \{12, 16\}$ across student grids $S \in \{48, 64\}$ in a complete 2×2 factorial evaluation (Table A3).

In terms of inference budget trade-offs, the tested matrix reveals that $K = 12$ achieves lower 2-NFE FID (4.666 ± 0.043 at $S = 64$) while $K = 16$ achieves lower 1-NFE FID (10.018 ± 0.062), reflecting a systematic interaction between teacher discretization resolution and few-step student distillation. Peak allocated memory scales primarily with the student supervision grid S (5.21 GiB at $S = 48$ versus 6.23 GiB at $S = 64$), remaining essentially invariant to teacher step count K (Table A3).

Table A3: Teacher budget and student grid workpoint matrix. Mean ± sample standard deviation across three independent training seeds (31, 37, 43), with ranges of maximum allocation over all logged intervals, including initialization. Lower is better for FID. Bold indicates best quality per inference budget.

Teacher Resolution K	Student Cells S	1-NFE FID50k	2-NFE FID50k	Memory (GiB)
12	48	10.661 ± 0.125	4.772 ± 0.021	5.207–5.209
16	48	10.337 ± 0.141	4.941 ± 0.082	5.209–5.211
12	64	10.400 ± 0.124	$\mathbf{4.666 \pm 0.043}$	6.235–6.236
16	64	$\mathbf{10.018 \pm 0.062}$	4.788 ± 0.068	6.233–6.235

Regarding component-level speedups, across six paired benchmark blocks (discarding 20 warmup iterations and retaining 176 synchronized measurements), reducing teacher steps from 16 to 12 reduced through-optimizer component time ratios to a median of 0.826 for $S = 48$ and 0.763 for $S = 64$, delivering measurable component acceleration of **17.4%** ($S = 48$) and **23.7%** ($S = 64$). Synchronized per-step loop time decreases from 0.934 ± 0.028 s to 0.565 ± 0.002 s (a 39.5% reduction in step time; Figure A3(a)).

B.4 Cross-Step Trajectory Reuse and Complementary Sampling (R-Axis)

To address the teacher ODE integration bottleneck while keeping GPU memory anchored by S, we evaluate cross-step trajectory reuse with refresh stride $R = 2$.

In early mid-training screening initialized from a 2k EMA checkpoint across three update seeds (140101, 140103, 140107) with $K = 12, S = 48$, activating $R = 2$ reuse under matched 400 updates reduced clean training loop time by an average of **31.89%** (range 27.25%–35.54%), while FID10k changed by less than 0.25% in relative terms. Under iso-time constraints (540–571 updates), the reuse model improved 1-NFE and 2-NFE FID10k by 6.86%–8.22% and 7.42%–9.61%, respectively, relative to fresh400. These observations represent short-horizon FID10k screen results, where matched-update differences remain small, without a formal equivalence test. Section B.5 quantifies both the equal-update quality cost and the iso-time benefit at FID50k over 30,000 updates, serving as the primary evidence for the reuse axis.

Turning to mature model fine-tuning with complementary coverage, we subsequently evaluated independent and complementary sampling ($\mathcal{C}_R$) on a mature 10k checkpoint with FID50k endpoints (Table A4).

Table A4: Later-initialization comparison on RTX 3090 at FP32 FID50k. Constant-rate anchors have one seed; phase and cosine schedules use seeds 140131, 140137, 140143 (mean ± sample standard deviation). Two Fresh cosine runs are new recovery measurements. Timings exclude setup/evaluation and were collected separately.

Arm	Updates	Loop (s)	Teacher images	1-NFE FID	2-NFE FID
Initial checkpoint	0	–	0	10.512	4.608
Iso-step 2,000 updates (single-seed anchors, $n = 1$)					
Fresh	2000	1247.9	192,000	10.398	4.581
Independent reuse	2000	902.8	96,000	10.439	4.577
Complement ($R = 2$)	2000	897.7	96,000	10.439	4.544
Extended 2,801 updates (single-seed anchors, $n = 1$)					
Fresh (constant η)	2801	1866.1	268,896	10.290	4.606
Independent reuse (constant η)	2801	1248.1	134,496	10.634	4.659
Complement (constant η)	2801	1258.4	134,496	10.623	4.574
Cosine and phase schedules ($n = 3$, mean ± std)					
Fresh (cosine)	2801	1789.4 ± 56.6	268,896	10.255 ± 0.031	4.584 ± 0.045
Phase ($R: 2 \to 1$)	2801	1403.3 ± 20.2	172,896	10.420 ± 0.019	4.583 ± 0.029
Complement (cosine)	2801	$\mathbf{1261.9 \pm 19.0}$	134,496	10.396 ± 0.068	$\mathbf{4.524 \pm 0.019}$

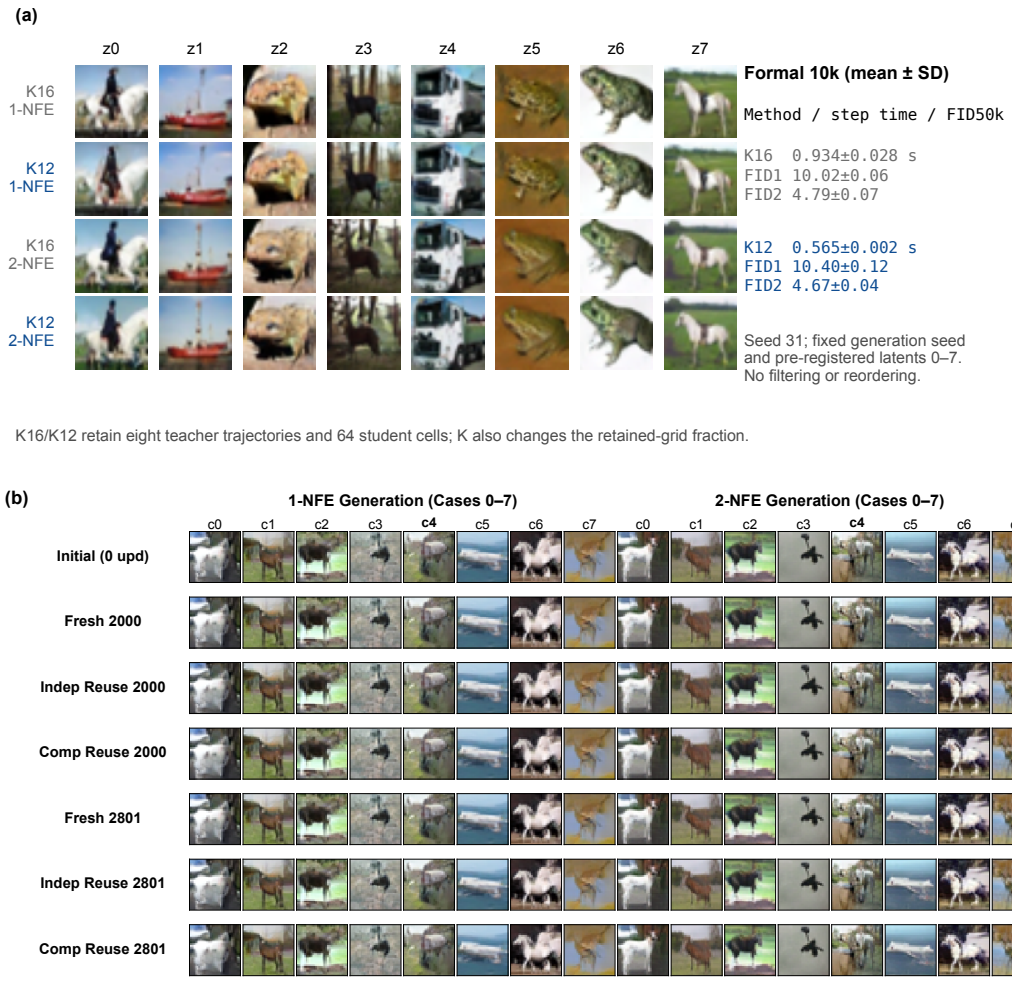


Figure A3: Fixed-sample qualitative comparisons, with original sample order preserved. (a) Teacher-grid comparison at 10k updates, with eight latents for training seed 31 under both inference budgets. The right-hand text reports means and sample standard deviations across training seeds 31, 37 and 43; changing K at fixed $S = 64$ also changes the retained-grid fraction. (b) Separate mature fine-tuning protocol, with eight cases for each of 1-NFE and 2-NFE. Rows show the initial checkpoint, fresh2000, independent reuse2000, complementary reuse2000, fresh2801, independent reuse2801 and complementary reuse2801. Samples are matched within each experiment, not across protocols (a) and (b). No samples were filtered or reordered.

Under 2,000 fine-tuning steps, complementary reuse reduced loop time by **28.06%** relative to fresh2000 (halving teacher evaluations to 96,000), while 1-NFE FID shifted by 0.395% (from 10.398 to 10.439) and 2-NFE FID improved by 0.796% (from 4.581 to 4.544). All eight fixed cases per condition are retained for qualitative comparison (Figure A3(b)); these images do not establish quality equivalence. These short-horizon runs are reported as screening evidence; Section B.5 restates the reuse axis as a pre-specified 30,000-update continuation with an iso-time endpoint, and that campaign is the primary reuse evidence in this work.

B.5 Confirmation Campaign across 30,000 Steps from a Fixed Warm Start

To evaluate whether budget allocation dynamics persist over extended training horizons, we executed a pre-specified confirmation campaign continuing two trajectory-reuse arms for 30,000 optimizer updates from a shared 10,000-step warm start ($S = 64$). Both arms load identical initial EMA student weights, clear the trajectory cache, and execute under a cosine learning-rate schedule from 5×10^{-5} to 1×10^{-5} across three pre-declared update seeds (150131, 150137, 150143). Arm A regenerates teacher trajectories at every update ($R = 1$) and samples 48 of 96 cells uniformly without replacement, while Arm B reuses each pool across two updates ($R = 2$) via complementary partitioning $\mathcal{C}_R$ ($K = 12, B = 8, S = 48$).

In the equal-step comparison at a matched budget of 30,000 updates, the reuse arm is slightly worse on both inference budgets. The paired difference in FID50k is $+1.40\% \pm 0.53\%$ on 2-NFE (all three seeds positive) and $+0.41\% \pm 1.34\%$ on 1-NFE (two of three seeds positive). Wall-clock loop time falls by a median of 30.61% (range 30.20%–32.91%), the ratio of teacher image evaluations is exactly 0.500 in all three pairs, and peak allocated GPU memory is *byte-identical* between the arms within every pair ($4{,}727{,}824{,}384$ bytes in all six runs). At the matched endpoint, complementary reuse therefore trades a roughly 1.4% higher 2-NFE FID for half the teacher work; this is an endpoint difference, not a loss incurred independently at every update.

Because equal-step comparison is not the operative question for a practitioner who is compute-bound rather than step-bound, we pre-specified an iso-time endpoint with a common wall-clock budget $T_{\text{iso}} = \lfloor 0.8 \times 30{,}000 \times \min(t_A, t_B)/60 \rfloor \times 60 = 10{,}440$ seconds, derived from the 400-update pilot rates ($t_A = 0.65230$, $t_B = 0.43742$ s/update) and frozen before the formal runs. The endpoint is determined as the first saved EMA snapshot whose recorded elapsed time reaches T_{iso}, strictly adhering to the frozen pilot throughput rate. Because arm B performs roughly $1.4\times$ as many updates per second, it reaches T_{iso} after about 44% more optimizer updates than arm A (23,000–25,000 versus 16,000–17,000).

At T_{iso} the ordering reverses, with the reuse arm performing better on both endpoints (Table A5).

Table A5: **Approximate iso-time endpoints (primary comparison).** $T_{\text{iso}} = 10{,}440$ s, frozen from the pilot rates before launch. All values are FID50k at 50,000 generated and 50,000 real images, FP32 inference. Negative Δ favors trajectory reuse.

Seed	Steps A	Steps B	FID1 A	FID1 B	FID2 A	FID2 B	ΔFID2
150131	16,000	23,000	9.4210	**8.9992**	4.4360	**4.3433**	$\mathbf{-2.09\%}$
150137	17,000	24,000	9.1654	**9.0602**	4.4574	**4.4072**	$\mathbf{-1.13\%}$
150143	17,000	25,000	9.3716	**9.0371**	4.4524	**4.4064**	$\mathbf{-1.03\%}$

Across the three pairs, $\Delta\text{FID2} = -1.42\% \pm 0.58\%$ and $\Delta\text{FID1} = -3.06\% \pm 1.72\%$, with all three seeds favoring reuse on both NFE. The reuse arm performs roughly 44% more updates within the approximate time budget, translating computational throughput into generation quality. We report the iso-time result as the primary comparison and the equal-step result as the matched-update trade-off.

While the pre-specified iso-time endpoint provides a single matched-time comparison per seed, characterizing the dynamic trajectory across training horizons rather than relying solely on a single isolated slice required evaluating four intermediate EMA snapshots per run (updates 10,000, 15,000, 20,000, and 25,000, drawn from the snapshots saved every 1,000 updates). Combined with the step-30,000 and iso-time anchors, this yields 35 measured (wall-clock, FID50k) points across the six runs from 24 additional 50k-FID evaluations (269 minutes of evaluation wall clock). An independent re-

evaluation of the reuse arm at update 25,000 (seed 150143) deterministically reproduced the stored values bit-exactly for that checkpoint and evaluation seed.

The interpolated crossings are broadly consistent with the 1-NFE endpoint result, but do not establish a consistent 2-NFE time-to-quality advantage. For 1-NFE the reuse arm reaches a given FID level in less wall-clock on all three seeds at three of the five tabulated levels; the three-seed mean gaps are 1.4k–2.9k seconds (Table A6), so the sign agrees with the iso-time endpoint. Seed 150137 is the exception, reaching the level first on the *fresh* arm at FID 9.50 (8,633 s versus 9,034 s). For 2-NFE that picture does not survive, as adjacent 2-NFE points differ by up to 0.105 FID, the per-seed verdicts split 1–2, and at FID 4.45 two of three seeds favor the fresh arm, so the between-arm difference at a matched level is comparable to the observed variation between consecutive checkpoints. This variation includes model changes and is not an estimate of FID sampling noise. We therefore restrict the 2-NFE comparison to the pre-specified iso-time endpoints and make no time-to-quality claim on 2-NFE.

The curve also shows that the reuse arm saturates earlier than its budget. On two of the three reuse seeds the best measured point is not the final one, as seed 150131 attains its lowest FID at 23,000 updates (1-NFE FID 8.999, 2-NFE FID 4.343, at $10{,}535$ s) and deteriorates by 30,000 updates (1-NFE FID 9.077, 2-NFE FID 4.421), while seed 150137's lowest 2-NFE FID occurs at 24,000 updates (FID 4.407 versus 4.435 at 30,000 updates). Thus the fixed terminal endpoint differs from the best observed checkpoint; primary comparisons strictly adhere to the pre-specified fixed endpoint to prevent selection bias from post-hoc checkpoint cherry-picking. All crossing times below are linearly interpolated between adjacent measured points and represent descriptive three-seed summaries without significance tests.

Table A6: **Wall-clock at which each arm first reaches a given FID50k level**, mean over the seeds that reach it, interpolated between adjacent measured points. A negative difference means the reuse arm got there first, and the counts n are given where a level is not reached by all three seeds. The FID 9.00 row is *not* a paired comparison, as the two arms reach it on different seed subsets (fresh includes 150131, 150137; reuse includes 150131, 150143); on the shared seed 150131 the gap is -41%.

Metric	FID level	Fresh (s)	Reuse (s)	Difference
1-NFE	10.00	6,571	**5,065**	-23%
1-NFE	9.75	7,958	**6,066**	-24%
1-NFE	9.50	9,647	**8,264**	-14%
1-NFE	9.25	12,760	**9,821**	-23%
1-NFE	9.00	17,285 (n=2)	**11,096** (n=2)	-36%
2-NFE	4.50	**8,177**	8,580	$+5\%$
2-NFE	4.45	9,976	**9,737**	-2%

To test whether the initialization drives the observed conclusions beyond the primary start, we repeated both arms from an independently trained 10,000-update second warm start (trained under sparse supervision with $S = 48$), holding the seed fixed at 150131 and the schedule identical, and evaluated the new start at step 0 as well as both 30,000-update endpoints and both iso-time endpoints at the same frozen $T_{\text{iso}} = 10{,}440$ s.

All four comparisons agree in sign with the first warm start (Table A7): reuse exhibits slightly higher FID at equal updates and lower FID at the approximate equal-time endpoint, the loop-time saving remains approximately 31%, the teacher-evaluation ratio is strictly 0.500, and peak allocated memory remains byte-identical between both arms. However, the performance magnitudes do not carry over. The equal-update endpoint penalty is $1.2\times$ (1-NFE) to $1.5\times$ (2-NFE) larger than at the first start, while the iso-time advantage shrinks to $0.20\times$ (for 1-NFE) and $0.16\times$ (for 2-NFE) of its original size; the first campaign's cross-seed spread cannot serve as an uncertainty estimate for this single-seed replication. The second starting point is evaluated strictly to verify qualitative directional consistency without formal significance inference.

This replication warrants careful interpretation under specific boundary conditions. Because the run uses a single seed, it carries no independent uncertainty estimate. Furthermore, the two warm starts differ not only by random draw but in their upstream recipes ($S = 64$ with pairing ratio 0.667 versus

$S = 48$ with 0.5), confounding initialization with training configuration. In addition, matched wall-clock time is discrete at our snapshot granularity: the fresh arm crossed T_{iso} 496.8 s late, whereas the reuse arm crossed 286.5 s late, meaning the reuse arm trained for 210.3 s less than the fresh arm. Without fine-grained monotonic learning curves, the directional impact of this timing discrepancy cannot be resolved. Although the empirical trade-off sign remains consistent across both starts, the magnitude varies; the gains from the first campaign represent measured empirical instances rather than universal constants.

Table A7: **Replication from a second warm start**, seed 150131 held fixed in both campaigns. Entry values are paired differences of FID50k (negative favors reuse) at the two pre-specified comparison axes.

Quantity	First warm start	Second warm start
Equal-step ΔFID1	$+1.41\%$	$+1.72\%$
Equal-step ΔFID2	$+1.98\%$	$+2.92\%$
Iso-time ΔFID1	-4.48%	-0.88%
Iso-time ΔFID2	-2.09%	-0.33%
Loop-time saving	30.20%	31.23%
Teacher-evaluation ratio	0.500	0.500
Peak memory identical	yes	yes

B.6 Teacher Resolution at the Same Horizon

The 10,000-step screening matrix of Section B.3 is a budget screening. We repeated the teacher-resolution axis at the same 30,000-update horizon, using the same warm start, schedule and three seeds as arm A, and changing only the teacher grid from $K = 12$ to $K = 16$ (arm D, $R = 1$, $S = 48$, $B = 8$; the pool therefore grows from 96 to 128 cells). Because S is held fixed, the retained fraction changes from $\rho = 0.500$ to $\rho = 0.375$; arm D therefore varies K and ρ together, and its effect cannot be attributed to teacher resolution alone.

Table A8: **Teacher resolution at 30,000 updates (arm D $K = 16$ versus arm A $K = 12$).** Same warm start, seeds, schedule and $S = 48$; FID50k, FP32 inference. Cost is teacher network evaluations per optimizer update.

Seed	FID1 ($K = 12$)	FID1 ($K = 16$)	FID2 ($K = 12$)	FID2 ($K = 16$)	ΔFID2
150131	8.9511	**8.3856**	**4.3351**	4.4584	$+2.84\%$
150137	8.9741	**8.5728**	**4.3808**	4.4854	$+2.39\%$
150143	9.0385	**8.6327**	**4.3500**	4.5020	$+3.49\%$
Teacher evaluations/update increase from $K = 12 \to 12$ to $K = 16 \to 16$ ($+33.3\%$)					

The direction is consistent across all three seeds (Table A8). Specifically, the finer teacher grid improves single-step generation ($\Delta\text{FID1} = -5.09\% \pm 1.06\%$, $3/3$ seeds favoring $K = 16$) while degrading two-step generation ($\Delta\text{FID2} = +2.91\% \pm 0.56\%$, $3/3$ seeds favoring $K = 12$), at 33% more upstream teacher evaluations. This reproduces the inference-budget-dependent duality seen in the screening matrix, now with a matched-start, matched-schedule paired design at the same 30,000-update horizon as arm A rather than at the 10,000-update screening budget.

Interpreting this effect requires accounting for the grid confound: because the $K = 16$ arm is also the sparser arm ($\rho = 0.375$ versus 0.500), the 1-NFE gain and the 2-NFE loss cannot be assigned to grid resolution alone. Isolating the two factors would require an additional arm varying K at constant ρ; in the absence of this control, no isolated causal claim is attributed to the K effect alone.

Equal-time gains coexist with equal-update quality costs in the paired 30k experiment (Figure 3). Results from independent protocols are evaluated separately.

B.7 Empirical Failure Boundaries and Automated Safety Milestone Protocol

We next separate mature-adaptation endpoint trade-offs from failures observed under distinct post-training protocols.

At an extended horizon of 2,801 updates, constant complementary reuse yields 1-NFE FID 10.623 versus 10.290 for the constant fresh anchor, representing a modest 3.24% quality divergence over prolonged execution.

For dynamic stride and learning rate governance, we evaluate phase-decayed refresh and cosine learning-rate decay as structured interventions over 2,801 updates, each across three seeds. Specifically, for the phase-decayed stride schedule $R(t)$ ($R = 2$ for 2,000 updates followed by $R = 1$ for 801 updates), the student achieves 1-NFE FID 10.420 ± 0.019 and 2-NFE FID 4.583 ± 0.029. Mean loop time is 1403.3 ± 20.2 s (24.80% below the fresh anchor), reducing teacher evaluations by 35.70%.

For cosine learning rate annealing at constant $R = 2$, decaying learning rate from 5×10^{-5} to 10^{-5} achieves 1-NFE FID 10.396 ± 0.068 and 2-NFE FID $\mathbf{4.524 \pm 0.019}$, with loop time 1261.9 ± 19.0 s across three seeds (a 32.38% loop-time saving relative to the constant fresh anchor).

In the matched cosine-schedule comparison, fresh cosine runs give 1-NFE FID 10.255 ± 0.031 and 2-NFE FID 4.584 ± 0.045 over the three paired update seeds. The reuse-minus-fresh 2-NFE difference is -0.060 ± 0.034, with a two-sided paired t-test $p = 0.091$ ($n = 3$); the 95% confidence interval $[-0.144, 0.024]$ includes zero. Two fresh runs were re-executed under strictly matched conditions to ensure end-to-end data provenance; the remaining endpoints retain original execution records. The t-test assumes normally distributed paired differences, an assumption not verifiable with three pairs. This comparison does not establish an interaction mechanism, and the constant-rate anchors have only one seed. Fresh loop time is 1789.4 ± 56.6 s, compared with 1261.9 ± 19.0 s for reuse; these separately collected timings are descriptive. The randomized resource benchmark provides the dedicated timing control.

In unconstrained post-training continuation, historical exploratory experiments exhibited severe 2-NFE collapse (Figure A4(a)). Under a 100-step protocol, Dense, TVA, and LSM suffered 14.3%–53.0% degradation in 2-NFE FID; under extended 2,500-step continuation, FID degraded beyond 100 into catastrophic distortion.

To safeguard against this vulnerability, at step 300 of the separate post-training protocol, we evaluate FID10k on both NFE. A relative regression above 20% triggers evaluation with a second generation seed; a confirmed breach stops the run. Otherwise the run continues to step 1000. This screen detects early metric regression and does not estimate the probability of later failure.

Table A9: Step-300 screen at FID10k and step-1000 endpoints at FID50k, separated by sample count. Parentheses give relative change from the paired FID10k baseline. Passing the early screen is not a long-run guarantee.

Seed	Arm	Step 0 Baseline (10k)		Step 300 Gate (10k)		Milestone / Outcome
		1-NFE	2-NFE	1-NFE	2-NFE	(1000-Step FID50k)
31	K12	12.257	6.650	12.406 (+1.2%)	6.901 (+3.8%)	PASS → 24.353 / 21.905
31	K16	11.831	6.762	**11.146** (−5.8%)	13.912 (+105.7%)	STOP_CONFIRMED[†]
37	K12	12.224	6.613	**11.378** (−6.9%)	8.416 (+27.3%)	STOP_CONFIRMED[‡]
37	K16	11.764	6.641	**11.065** (−5.9%)	7.448 (+12.2%)	PASS → 10.924 / 7.434

[†]Confirmed on secondary evaluation seed (1-NFE FID 11.196, 2-NFE FID 14.437, relative 2-NFE change +113.5% relative to baseline). [‡]Confirmed on secondary evaluation seed (1-NFE FID 11.592, 2-NFE FID 8.563, relative 2-NFE change +31.2% relative to baseline).

The runs from training seeds 31 and 37 show different one-step and two-step outcomes (Figure A4(b,c); Table A9).

Regarding single-step and two-step sensitivity, three of four step-300 cases improve 1-NFE FID by 5.8%–6.9%, while 2-NFE FID exhibits acute degradation in multiple runs (Table A9). These

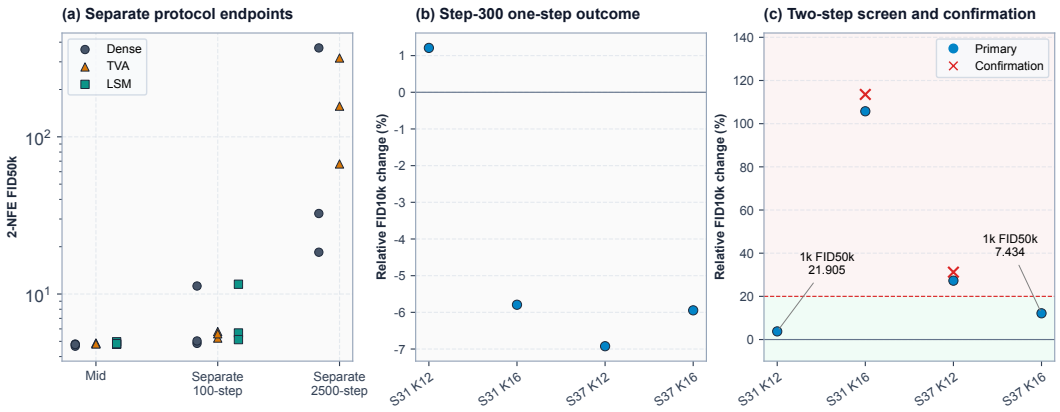


Figure A4: Post-training observations. (a) Historical mid-training, 100-step and 2500-step endpoints are shown by protocol; the separate runs do not form one continuous learning curve. (b) Step-300 1-NFE changes. (c) FID10k screen and second-seed confirmation at a 20% threshold. A pass did not prevent the Seed 31 K12 step-1000 failure. The 1000-step post-training protocol applies linear learning-rate decay.

endpoints show that improvements at one NFE do not ensure improvements at the other; they do not identify the cause of the difference.

On the relationship between early screening and long-horizon drift, Seed 31 K12 passed the step-300 screen yet reached FID50k values of 24.353 (1-NFE) and 21.905 (2-NFE) at step 1000 under linear decay. This run shows that passing the early gate does not establish stability at the later endpoint.

B.8 Downstream Transfer Boundary

To evaluate whether mid-training performance rankings are preserved during downstream training, we applied the official-compatible consistency-training recipe (incorporating the official consistency-training procedure, ECT preconditioning, batch size 128, RAdam at 1×10^{-4}, sigmoid timestep mapping, and $q = 4096$) alongside a pre-specified step-300 safety gate ($> 20\%$ relative FID10k regression on either NFE, confirmed on an independent generation seed).

The safety gate terminated all tested transfers from the 30k checkpoints, with the complete evaluation matrix including a control that uses an earlier-campaign checkpoint as the transfer input (Table A10).

Table A10: **Step-300 gate outcomes for standard post-training.** Ratios are FID10k at step 300 over step 0; the gate requires ≤ 1.20 on both NFE and confirms any breach on an independent generation seed. Every row uses the identical frozen recipe and the same step budget.

Transfer input	Post seed	1-NFE ratio	2-NFE ratio	Outcome
Earlier-campaign checkpoint	31	1.012	1.038	PASS
Earlier-campaign checkpoint	250131	0.955	3.317	STOP
Ours (fresh arm, 30k, $S = 48$)	31	1.055	1.268	STOP
Ours (fresh arm, 30k, $S = 48$)	250131	1.720	2.010	STOP
Ours (reuse arm, 30k, $S = 48$)	250131	2.103	7.004	STOP

The earlier-campaign input at post seed 31 approximately reproduces the previous screen (1-NFE ratio 1.0122 and 2-NFE ratio 1.0377, versus 1.0121 and 1.0377 in the previous screen), providing a limited reproducibility check. At the single post seed shared by the two 30k inputs, the reuse arm exhibits a higher 2-NFE ratio (7.00 for reuse versus 2.01 for fresh). This comparison does not identify a stale-target mechanism or establish a general ranking. Outcomes also vary with the post seed; the matrix is too small to quantify the relative contributions of input and seed.

C Legacy Supervision Baselines and Adaptive Controller Formulation

C.1 Controlled problem formulation

The legacy supervision comparisons retain the complete teacher trajectory and select cells only before student evaluation, ensuring that every method operates on the identical realized teacher grid without confounding endpoint construction or ODE solver error. Dense evaluates the full time–trajectory grid ($N = 128$), whereas Sparse-uniform retains $S = 64$ cells drawn uniformly without replacement ($\rho = 1/2$). TVA and LSM hold this student cell budget fixed while replacing uniform selection with delayed probe-based allocation. The scheduler updates allocation weights q_h using completed probe events, ensuring causal separation between probe measurement and online student optimization while holding optimizer steps, precision, seeds, and evaluation protocols strictly fixed.

C.2 Trajectory regression and notation

Let a frozen teacher solver generate B trajectories per optimizer update. Trajectory b contains states $x_{b,i}^T$ at solver times t_i, $i \in \{1, \dots, K\}$, and the tiled teacher endpoint y_b^T. For cell $c = (b, i)$, define

$$\ell_c(\theta) = d_{\mathrm{LPIPS}}\big(f_\theta(x_{b,i}^T, t_i), y_b^T\big), \qquad \widehat{\mathcal{L}}_D(\theta) = \frac{1}{N}\sum_{c\in\Omega} \ell_c(\theta), \quad \Omega = [B]\times[K],\ N = BK. \tag{46}$$

Here d_{LPIPS} is the learned perceptual metric of Zhang et al. (2018). The reported training objective is $\mathcal{L}_D(\theta) = \mathbb{E}_\tau[\widehat{\mathcal{L}}_D(\theta)]$, where τ includes the current data batch and teacher noise. The implementation concatenates cells in time-major order, $j = (i-1)B + (b-1)$, and applies the same selected flat indices to states, times, endpoints, labels, and augmentation labels. All K teacher states are generated before selection; allocation can therefore reduce differentiable student work but not teacher integration.

For a retained set $A \subset \Omega$ of size $S = \rho N$, the implemented sparse loss is

$$\widehat{\mathcal{L}}_A(\theta) = \frac{1}{S}\sum_{c\in A} \ell_c(\theta), \qquad \widehat{g}_A = \nabla_\theta \widehat{\mathcal{L}}_A. \tag{47}$$

Formal sparse runs use $B = 8$, $K = 16$, $\rho = 1/2$, and hence $S = 64$. The mean in Eq. (47) preserves loss scale; it does not by itself guarantee that an adaptive subset is an unbiased estimator of the dense objective.

Regarding formal update units, the single-GPU configuration sets both the nominal global batch and the per-device batch size to 128, with one microbatch and no gradient accumulation. The CMT loss implementation defines the number of teacher trajectories as the per-device batch size divided by the teacher discretization steps, $B = 128/16 = 8$, and uses the first eight loader examples to construct the complete 8×16 grid (Sony Research, 2026). Dense averages 128 cell losses; every sparse method averages 64 selected cell losses; each path performs one backward pass and one optimizer update. Thus "batch 128" denotes the nominal student-cell and progress-counter configuration, not 128 independent teacher trajectories. The progress counter advances by 128 nominal images per update, while the teacher path consumes eight examples. All four methods share these semantics and differ only after the complete grid has been constructed.

C.3 Sequential weighted sampling without replacement

Let $q_h(i)$ be the time-bin mass available after probe event h. Each grid cell receives initial weight $w_{b,i} = q_h(i)/B$. The implementation draws an ordered sequence $Z_1, \dots, Z_S$ by removing each selected cell and renormalizing the remaining weights, with conditional transition probabilities given by

$$\Pr(Z_r = c \mid Z_{<r}) = \frac{w_c}{1 - \sum_{u<r} w_{Z_u}}, \qquad c \notin \{Z_1, \dots, Z_{r-1}\}. \tag{48}$$

Thus cells never repeat within an update, but different cells from the same time bin or trajectory may be selected. This is successive probability-proportional-to-size sampling, not independent categorical sampling.

Uniform $q_h(i) = 1/K$ satisfies Proposition 1; the complete proof is in Appendix A.

For adaptive q_h, let $\pi_{h,c} = \Pr(c \in A \mid q_h)$ denote the inclusion probability induced by Eq. (48). Then

$$\mathbb{E}_A[\widehat{g}_A \mid \tau, q_h] = \frac{1}{S} \sum_{c \in \Omega} \pi_{h,c} \nabla_\theta \ell_c(\theta). \tag{49}$$

The code applies neither inverse-probability nor Horvitz–Thompson correction (Horvitz & Thompson, 1952). Consequently, Eq. (49) is generally biased for the dense gradient, as TVA and LSM intentionally optimize a delayed, allocation-weighted curriculum objective. Their comparison with Sparse-uniform estimates the effect of that changed exposure, not a lower-variance unbiased estimator of the same dense loss.

C.4 Fixed-probe realized loss reduction

We freeze $M = 8$ probe examples and one noise tensor using a seed offset; probe calls use no augmentation. The examples remain members of the training dataset but the probe bank is evaluated in a separate, non-backpropagated call and never supplies persistent replay identities. With student parameters θ_h at event h, the per-bin probe loss is

$$L_h^p(i) = \frac{1}{M} \sum_{m=1}^{M} d_{\mathrm{LPIPS}}\Big(f_{\theta_h}(x_{m,i}^{T,p}, t_i), y_m^{T,p}\Big), \qquad r_h(i) = L_{h-1}^p(i) - L_h^p(i). \tag{50}$$

The first event only establishes L_0^p. Later events robustly standardize the realized reductions. To match the implementation exactly, let med^+ be the upper middle order statistic, and define

$$m_h = \mathrm{med}^+_{j \in [K]} r_h(j), \qquad d_h = \mathrm{med}^+_{j \in [K]} |r_h(j) - m_h|, \tag{51}$$

$$s_h = \max(1.4826 d_h, 10^{-8}), \qquad z_h(i) = \mathrm{clip}\left(\frac{r_h(i) - m_h}{s_h}, -c, c\right), \tag{52}$$

$$a_h(i) = \beta a_{h-1}(i) + (1 - \beta) z_h(i). \tag{53}$$

Formal TVA uses $c = 5$, $\beta = 0.8$, and one event every $H = 100$ optimizer steps. Median–MAD normalization removes a common loss-reduction offset and limits one-bin outliers; the EMA trades responsiveness for temporal smoothing. Equation (50) is observational, not a causal estimate of the benefit of training bin i.

C.5 Delayed allocation

The score available after event h defines the delayed next-update allocation,

$$q_h(i) = (1 - \alpha) \frac{\exp(a_h(i)/T)}{\sum_{j=1}^{K} \exp(a_h(j)/T)} + \frac{\alpha}{K}, \qquad \alpha \in (0, 1],\ T > 0. \tag{54}$$

Formal TVA uses $\alpha = 0.10$ and $T = 0.50$. Because a_h depends only on probes observed after completed optimizer intervals, the current subset cannot use its own future loss reduction. The update is therefore delayed by construction, although delay does not imply convergence.

The nonzero exposure bound and its proof appear in Appendix A.11.

C.6 Fixed-probe loss-second-moment allocation

LSM uses the same probe losses but not their temporal differences. For a rolling history of $J = 10$ events per bin,

$$u_h(i) = \sqrt{\frac{1}{J} \sum_{j=h-J+1}^{h} [L_j^p(i)]^2}, \qquad q_h^{\mathrm{LSM}}(i) = (1 - \alpha) \frac{u_h(i)}{\sum_k u_h(k)} + \frac{\alpha}{K}. \tag{55}$$

Before every bin has J observations, the implementation sets all scores to one and remains exactly uniform. Formal runs require at least ten probes, finite normalized probabilities, valid exposure counts, and final probability spread above 10^{-4}. This is a fixed-probe adaptation of loss-second-moment sampling (Nichol & Dhariwal, 2021), not an exact reproduction of training-loss importance sampling, and it also uses the uncorrected sparse loss in Eq. (47).

C.7 Legacy Baseline Optimization Algorithm

Algorithm A1. Delayed Adaptive Sparse CMT Baselines (TVA / LSM)

1. Initialize $a_0(i) = 0$, no previous probe loss, and a fixed probe bank; set $q_0(i) = 1/K$.
2. At step n, generate the complete $B \times K$ teacher grid and aligned endpoints.
3. If n is a probe event, evaluate all K probe bins without gradients; update Eqs. (50)–(54) for subsequent subsets.
4. Set $S = \rho BK$ and sequentially sample S unique cells using Eq. (48).
5. Gather every aligned tensor with the same flat indices; compute Eq. (47), backpropagate its mean, and update θ once.

No gradient is propagated through the probe statistic, q_h, or the discrete subset. The method therefore operates in an alternating manner, where a delayed measurement update chooses a piecewise-constant sampling distribution, followed by standard gradient descent on the selected student loss.

C.8 Controls and computational cost

Table A11: Formal methods and the factor isolated by each comparison.

Method	Teacher states/path	Student cells/update	Allocation	Identified comparison
Dense	16	128	uniform	reference
Sparse-uniform	16	64	uniform, no replacement	sparsity
TVA	16	64	delayed probe reduction	adaptive increment
LSM	16	64	loss second moment	adaptive baseline

For K solver states and B trajectories, the per-step cost decomposes as

$$C_{\text{step}}(\rho) = C_T(B, K) + C_S(\rho BK) + \mathbf{1}_{\text{probe}} C_P(M, K) + C_{\text{sample}}(BK, S). \tag{56}$$

Here $\mathbf{1}_{\text{probe}}$ indicates a probe event; averaging over H updates gives probe cost $C_P(M, K)/H$. The current sampler is a direct $O(NS)$ list implementation with $N = 128$ and $S = 64$; this CPU term is negligible at the tested scale but is not an asymptotically optimal sampler. Student activation storage scales approximately with S, whereas the complete no-gradient teacher grid still exists. Consequently, halving S predicts neither a 50% wall-time reduction nor removal of teacher memory. Reported measurements include probe and sampling overhead.

C.9 Resource-balance interpretation

Let C_T denote teacher integration, C_S dense student computation, C_O common optimizer and data overhead, and C_A allocation overhead averaged per step. A first-order wall-time ratio is

$$R_{\text{time}}(\rho) = \frac{C_T + \rho C_S + C_O + C_A}{C_T + C_S + C_O}. \tag{57}$$

Even an ideal sampler cannot obtain a $1 - \rho$ end-to-end reduction unless $C_T + C_O$ is negligible. In contrast, student activation memory is closer to

$$M_{\text{peak}}(\rho) \approx M_T + M_O + \rho M_S, \tag{58}$$

where teacher states are generated without gradients and can contribute less to the autograd peak than the student. Equations (57)–(58) therefore predict that sparse supervision can affect peak activation memory more strongly than elapsed time. They define a first-order accounting model; the measured resource effects are reported separately in Section 4.

TVA and LSM change which student cells are evaluated but cannot remove C_T from the streaming path; their incremental speedup over Sparse-uniform is therefore limited to allocation overhead and heterogeneous per-cell student cost.

C.10 IMPLEMENTATION

The intervention is inserted after complete teacher-trajectory generation and immediately before student evaluation. It samples aligned tuples of trajectory state, timestep, endpoint, class label, and augmentation label with a shared index set. The teacher path and endpoint construction are unchanged, and no persistent trajectory replay is introduced.

D ALGORITHM AND IMPLEMENTATION

D.1 EXECUTION ORDER

The complete update order is structured as follows.

1. draw the training batch, teacher noise, and augmentation state;
2. integrate the frozen teacher solver across all $K = 16$ states for all $B = 8$ trajectories;
3. when scheduled, run the fixed probe bank without gradients and update only the scheduler state;
4. construct $BK = 128$ aligned student cells in time-major order;
5. draw $S = 64$ unique flat indices from the distribution available before the student update;
6. gather state, time, endpoint, class, and augmentation tensors with exactly the same indices;
7. evaluate the student on the gathered tensors, average the selected losses, backpropagate once, and update the optimizer and EMA networks;

This order enforces three invariants. **Alignment** ensures that every selected cell refers to one consistent tuple; **Uniqueness** guarantees that no cell appears twice in one update; **Causality** ensures that q_h depends only on completed probe events.

D.2 TVA AND LSM STATE MACHINES

TVA stores the previous probe-loss vector, the EMA score vector, the current probability vector, and cumulative exposure. Its first probe event cannot compute a reduction and therefore only initializes the previous-loss vector. Every later event computes r_h, robustly normalizes it, clips it, updates a_h, and produces q_h.

LSM stores a length-J rolling buffer of probe-loss vectors. Before $J = 10$ observations are present, its score vector is fixed to one, making the sampler exactly uniform. After warm-up, it computes the root mean square within each bin and mixes the normalized vector with the same exploration floor. This warm-up gate is part of the algorithm, as a non-uniform distribution observed before ten complete histories would indicate a state or validation error, not successful adaptation.

D.3 SAMPLER PROPERTIES AND IMPLEMENTATION CEILING

The current sampler repeatedly draws from the remaining weight vector and deletes the selected entry. It requires $O(NS)$ scalar work and $O(N)$ storage per optimizer update. At $N = 128$ and $S = 64$, the implementation is simple and auditable. A Gumbel-top-k or exponential-key sampler would improve asymptotic behavior, but replacing the sampler would change random-number consumption and could complicate exact parity checks. The direct implementation is therefore retained for this benchmark; sampler optimization becomes relevant only if BK grows enough for CPU selection to enter the measured step time.

D.4 EXPERIMENTAL CONFIGURATION

Table A12: Training configuration for the 10k supervision comparison.

Factor	Formal value	Interpretation
Dataset	CIFAR-10, 50,000 images, 32×32	Full training set without subsampling
Teacher	Pretrained unconditional VP-EDM	Complete frozen teacher trajectory for every method
Teacher grid	$B = 8$, $K = 16$	128 cells before student selection
Student budget	Dense 128; sparse 64 cells/update	The only density intervention
Optimizer steps	10,000	3.3% of the 300,000-step CMT reference schedule
Nominal batch / `batch-gpu`	128 / 128	one microbatch, no gradient accumulation
Teacher examples per update	$128/16 = 8$	released loss code truncates the loader batch before trajectory generation
Progress-counter increment	128 nominal images/update	differs from eight examples consumed by the teacher path
Precision	BF16 with TF32 enabled	Standard mixed-precision configuration for numerical stability
Loss	LPIPS mean over selected cells	Same scale for Dense and sparse runs
Seeds	31, 37, 43	Paired across formal methods
Evaluation	FID50k at 1-NFE and 2-NFE	50,000 generated and 50,000 real images
Resource endpoints	wall time, logged peak allocated GPU memory	Measured on one A100 80-GB GPU
Stop rule	execution failure only	Fixed schedule completion; runs proceed to planned horizon

The batch fields require an implementation-level interpretation. On the single A100, both the nominal global batch size and the per-device batch size were 128, and the accumulation count was one. Within the CMT loss implementation, the effective trajectory batch size is obtained by dividing the data loader batch size by the teacher discretization steps ($128/16 = 8$), so only the first eight loader examples enter teacher trajectory generation. Tiling those eight endpoints over 16 solver states produces the 128 aligned Dense cells. Sparse methods gather 64 distinct cells from this grid. Dense and sparse runs each take one mean loss, one backward pass, and one optimizer step; they have identical loader, teacher-example, and teacher-state exposure. The progress counter nevertheless advances by the nominal batch size of 128 per update. Consequently, 1.28 nominal Mimg denotes 10,000 reference-code counter increments, not 1.28 million teacher examples consumed. This ledger resolves the difference among dataset examples, teacher trajectories, student cells, and the nominal batch field.

Table A13: Scheduler-specific settings. A dash denotes no scheduler state.

Method	H	M	J	α	T	β
Dense	–	–	–	–	–	–
Sparse-uniform	–	–	–	–	–	–
TVA	100	8	–	0.10	0.50	0.80
LSM	100	8	10	0.10	–	–

E STATISTICAL ANALYSIS AND COMPLETE ENDPOINTS

For a metric Y and paired seeds $s \in \{31, 37, 43\}$, the reported adaptive or sparse increment is $d_s = Y_s^{(A)} - Y_s^{(B)}$. We report

$$\bar{d} = \frac{1}{3}\sum_s d_s, \qquad s_d = \sqrt{\frac{1}{2}\sum_s (d_s - \bar{d})^2}, \qquad \bar{d} \pm t_{0.975,2}\frac{s_d}{\sqrt{3}}. \tag{59}$$

The interval describes uncertainty across the three training seeds under the tested protocol. It is not a non-inferiority interval because no scientifically justified margin was defined. The $50{,}000$ generated images stabilize each FID calculation but remain nested technical observations within one trained seed. Individual samples across all training seeds and factorial conditions are retained without selection alongside the aggregate metrics (Figure A5).

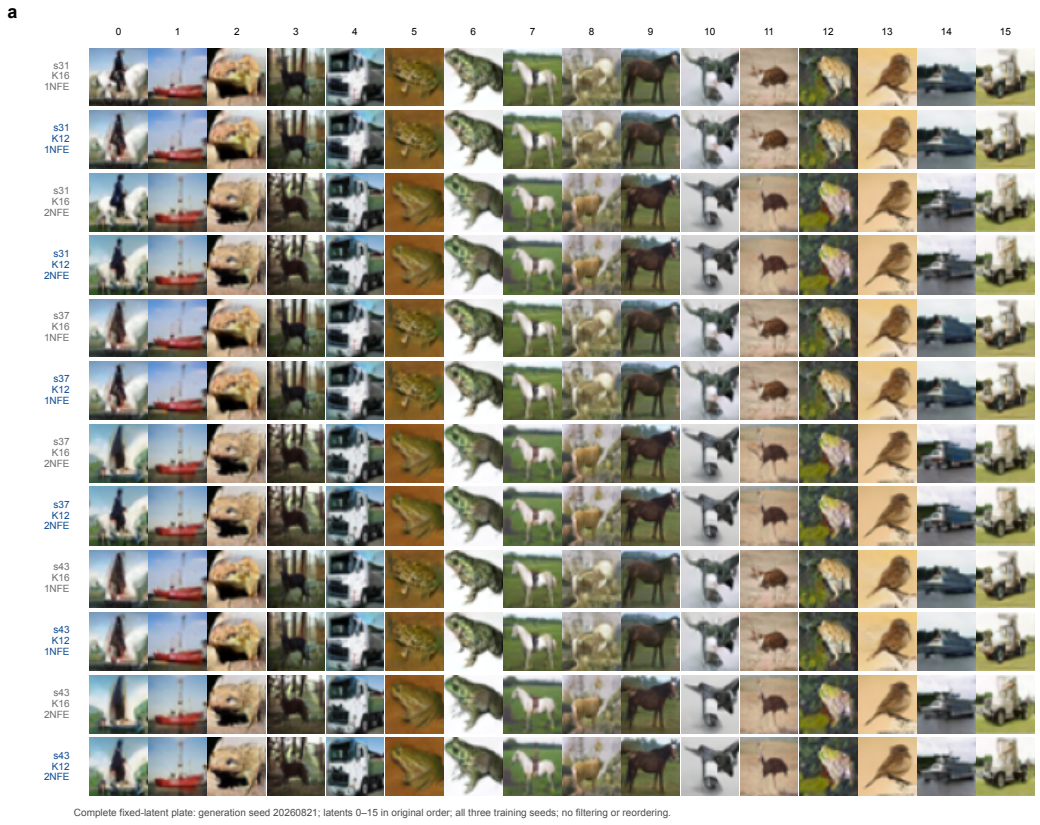


Figure A5: **Comprehensive Uncurated Qualitative Sample Plate Across Fixed Latents and Training Seeds.** Complete 12-row × 16-column plate (192 total generation tiles) generated under a pre-specified fixed evaluation seed. Columns display latents 0–15 in consecutive order without cherry-picking or filtering. Rows represent four factorial conditions across three independent training seeds (31, 37, 43), covering K16 1-NFE, K12 1-NFE, K16 2-NFE, and K12 2-NFE. These fixed samples illustrate differences in appearance; the corresponding FID results quantify the distribution-level comparison.

The confidence intervals are deliberately secondary to the effect pattern. Sparse-uniform reduced both resource endpoints in all paired seeds, while the quality intervals were wide. All three observed LSM 1-NFE differences favored LSM and all three 2-NFE differences were unfavorable, although the latter interval included zero. These endpoint-specific directions are why the manuscript does not use a scalar composite score.

Table A14: Paired formal effects. Percentage change uses the comparator mean; negative resource changes are savings and negative FID changes are improvements.

Comparison	Endpoint	Mean difference	Change (%)	Descriptive 95% CI
Sparse-uniform − Dense	1-NFE FID	$+0.115$	$+1.16$	$[-0.372, 0.603]$
	2-NFE FID	$+0.044$	$+0.92$	$[-0.312, 0.400]$
	Wall time (h)	-0.156	-5.64	$[-0.220, -0.093]$
	Memory (GiB)	-3.989	-41.79	$[-4.261, -3.716]$
TVA − Sparse-uniform	1-NFE FID	$+0.102$	$+1.02$	$[-0.343, 0.547]$
	2-NFE FID	$+0.055$	$+1.16$	$[-0.027, 0.138]$
	Wall time (h)	-0.039	-1.49	$[-0.089, 0.011]$
LSM − Sparse-uniform	1-NFE FID	-0.283	-2.83	$[-0.384, -0.183]$
	2-NFE FID	$+0.074$	$+1.54$	$[-0.041, 0.189]$
	Wall time (h)	-0.081	-3.08	$[-0.345, 0.184]$

Table A15: Per-seed 10,000-step mid-training endpoints. Lower FID, wall time, and memory are better.

Method	Seed	1-NFE FID	2-NFE FID	Wall time (h)	Memory (GiB)
Dense	31	9.771	4.656	2.809	9.460
Dense	37	9.894	4.807	2.682	9.580
Dense	43	10.045	4.771	2.827	9.590
Sparse-uniform	31	10.059	4.866	2.628	5.585
Sparse-uniform	37	10.049	4.765	2.552	5.585
Sparse-uniform	43	9.947	4.735	2.669	5.495
TVA	31	10.349	4.883	2.613	5.580
TVA	37	10.135	4.832	2.503	5.590
TVA	43	9.879	4.816	2.617	5.590
LSM	31	9.796	4.969	2.570	5.580
LSM	37	9.791	4.785	2.565	5.460
LSM	43	9.617	4.832	2.472	5.580

E.1 POST-TRAINING PROTOCOLS AND COMPLETE ENDPOINTS

The 100-step protocol was frozen before launch for Dense, TVA, and LSM. It required zero training exit status, finite checkpoints, and complete FID50k at both NFE values. The per-run safety criterion required each post-training FID to be no more than 1.10 times its own mid-training FID. A separate descriptive Pareto criterion required the LSM mean post 2-NFE FID to be no more than 1.05 times the Dense mean, while retaining at least 5% mid-training wall-time and 35% memory reduction. Passing would still not establish non-inferiority or cross-dataset generality. The protocol failed on 2-NFE quality.

Table A16: Complete 2,500-step post-training FID50k endpoints. All processes completed and checkpoints/evaluations were present; this is not a quality-gate pass; lower is better.

Method	Seed	1-NFE FID	2-NFE FID
Dense	31	16.402	32.526
Dense	37	38.830	18.447
Dense	43	278.091	367.799
TVA	31	263.384	317.259
TVA	37	41.367	67.214
TVA	43	12.286	156.761

Table A17: Seed-matched mid-training and 100-step post-training FID50k. Each endpoint is mid $\rightarrow$ post.

Method	Seed	1-NFE FID	2-NFE FID
Dense	31	9.771 $\rightarrow$ 9.575	4.656 $\rightarrow$ 4.853
Dense	37	9.894 $\rightarrow$ 9.455	4.807 $\rightarrow$ 5.028
Dense	43	10.045 $\rightarrow$ 9.663	4.771 $\rightarrow$ 11.246
TVA	31	10.349 $\rightarrow$ 10.458	4.883 $\rightarrow$ 5.772
TVA	37	10.135 $\rightarrow$ 9.707	4.832 $\rightarrow$ 5.246
TVA	43	9.879 $\rightarrow$ 9.773	4.816 $\rightarrow$ 5.586
LSM	31	9.796 $\rightarrow$ 10.221	4.969 $\rightarrow$ 5.653
LSM	37	9.791 $\rightarrow$ 9.414	4.785 $\rightarrow$ 5.129
LSM	43	9.617 $\rightarrow$ 9.241	4.832 $\rightarrow$ 11.542

Numerical process completion cannot be substituted for generative quality, as the endpoint distribution is highly seed-dependent despite all six checkpoints remaining readable and finite (Table A16). Averaging alone also hides the failure shape, as TVA seed 43 retained a moderate 1-NFE value while its 2-NFE value remained severely degraded, and Dense seed 43 collapsed at both endpoints.

F ALTERNATIVE DESIGN CONSIDERATIONS AND SCOPE ANALYSIS

The formal experimental design isolates the primary axes $\mathcal{M}(K, R, S)$ within a controlled execution pipeline. Several alternative sampling mechanisms warrant theoretical discussion:

- **Importance-corrected adaptive sampling.** While a Horvitz–Thompson estimator (Horvitz & Thompson, 1952) could formally preserve the dense objective under non-uniform sampling, successive sampling without replacement induces complex high-order joint inclusion probabilities. Small inclusion probabilities can produce large inverse weights, but do not imply a universal variance penalty relative to uniform sampling. Their effect depends on the gradients and joint inclusion probabilities.
- **Independent with-replacement sampling.** Drawing cells independently with replacement simplifies inclusion probabilities but permits duplicate supervision within a single update, causing the realized number of distinct targets to vary stochastically and drop below S. Without-replacement sampling preserves strict per-update memory and distinct-target parity across all methods.
- **Per-trajectory quotas and replay.** Restricting sampling to exactly ρK bins per trajectory alters the global competition across trajectories, while persistent replay in streaming CMT would require

caching intermediate noise or trajectory states across iterations, changing memory residency and statistical support.

- **Teacher-state sparsification.** Omitting teacher states during ODE integration could reduce teacher computation C_T, but numerical ODE solvers require intermediate evaluation states to maintain trajectory integration accuracy; changing solver discretization directly alters target construction.

F.1 Probe Dynamics and Measurement Interpretation

The probe bank functions as an embedded measurement instrument. The realized reduction $r_h(i) = L^p_{h-1}(i) - L^p_h(i)$ is retrospective, reflecting cumulative parameter updates, momentum dynamics, and EMA progression over the preceding interval on a fixed evaluation bank. Robust median–MAD standardization with clipping ($c = 5$) removes shared interval location shifts and guards against outlier-dominated allocation, producing a stable ranking signal rather than a calibrated metric of expected FID improvement.

F.2 Resource Accounting Boundaries

Training wall time captures the elapsed duration of complete training updates under the benchmarked configuration, including ODE integration, student forward and backward passes, probe evaluations, and synchronization, measured synchronously on dedicated hardware. Logged memory reflects the interval peak allocation recorded by the CUDA memory manager, isolating the active tensor and backpropagation graph requirements across student configurations while teacher weights remain frozen. As formalized in Eq. (57), when teacher ODE integration dominates step computation while student backpropagation dominates activation retention, sparse supervision naturally reduces peak memory more substantially than wall-clock time.

F.3 Seed-level post-training change analysis

Tracking relative metric changes rather than endpoints alone isolates within-run degradation, where a ratio below one indicates improvement relative to the seed-matched mid-training checkpoint (Table A18). The predeclared per-run safety gate required both ratios to be at most 1.10.

Table A18: Seed-matched change after 100 post-training steps. Δ is post minus mid FID; ratio is post divided by mid.

Method	Seed	Δ 1-NFE	Ratio	Δ 2-NFE	Ratio
Dense	31	-0.196	0.980	$+0.197$	1.042
Dense	37	-0.439	0.956	$+0.221$	1.046
Dense	43	-0.382	0.962	$+6.475$	2.357
TVA	31	$+0.110$	1.011	$+0.888$	1.182
TVA	37	-0.427	0.958	$+0.413$	1.086
TVA	43	-0.106	0.989	$+0.771$	1.160
LSM	31	$+0.425$	1.043	$+0.684$	1.138
LSM	37	-0.377	0.961	$+0.344$	1.072
LSM	43	-0.376	0.961	$+6.710$	2.388

The 1-NFE column is mostly stable, with seven of nine runs improving and the two increases being 1.1% (TVA seed 31) and 4.3% (LSM seed 31). The 2-NFE column is qualitatively different, as all nine runs worsen. Dense seeds 31 and 37 remain within the 10% gate, TVA seed 37 and LSM seed 37 remain within it, while five runs fail and two exceed twice their starting FID. The failure is therefore not a small shift of a common mean. It is an endpoint-specific tail event superimposed on a general upward 2-NFE drift.

This pattern limits causal interpretation. Dense and LSM share a seed-43 collapse, but TVA does not collapse to the same magnitude; this could reflect allocator-dependent initialization, stochastic post-training dynamics, or both. With one post-training continuation per seed, the experiment cannot estimate the conditional probability of collapse from a fixed checkpoint. Replicated post-training

seeds from the same mid-training checkpoint would separate initialization effects from continuation noise.

The 2,500-step results strengthen the failure conclusion but not its mechanism. Every seed/method endpoint worsens and several exceed the mid-training FID by an order of magnitude. Because the long protocol was run only for Dense and TVA and after a low-budget initialization, it does not characterize post-training stability beyond the tested configuration.

F.4 Observed Step-300 Screening Outcomes

The recorded screen evaluates FID10k at step 300 and requests a second generation seed after a greater-than-20% regression on either NFE. Reconstructed milestone records report early decisions separately from observed final FID50k, noting that halted runs have no counterfactual final endpoint (Table A19). No independent long-run class labels or repeated-noise estimate support a ROC curve or a population false-alarm probability.

Table A19: Recorded FID10k ratios and early decisions. Final values use FID50k at step 1000, a different sample count.

Run	Ratio 1	Ratio 2	Confirm 2	Final 2	Decision
S31 K12	1.012	1.038	–	21.905	PASS
S31 K16	0.942	2.057	2.135	–	STOP
S37 K12	0.931	1.273	1.312	–	STOP
S37 K16	0.941	1.122	–	7.434	PASS

Two runs were stopped and two passed. Seed 31 K12 subsequently reached 2-NFE FID50k 21.905 despite passing at step 300. Its post-training schedule linearly decayed the learning rate over 1000 updates. Passing therefore provides neither a tangent-stability certificate nor a long-run quality guarantee. Stopping at 300 skips 700 planned optimizer updates, but the compute saving depends on checkpoint and evaluation overhead. A gate and an annealing schedule have not been jointly validated as a complete stability protocol.

F.5 NOTATION REFERENCE

Table A20: Symbols used in the method and analysis.

Symbol	Definition	Scope and boundary
B	trajectories per optimizer update	$B = 8$ in formal runs
K	teacher ODE discretization steps	$K = 12$ coarse, $K = 16$ fine
R	trajectory refresh stride	$R = 1$ fresh, $R = 2$ 2-step reuse
S	retained student supervision cells	tested values $S = 48, 64, 128$; Dense means $S = N$
$\mathcal{M}(K, R, S)$	joint budget space	parameterizes teacher resolution, refresh stride and supervision size
$\mathcal{C}_R$	complementary sampling operator	disjoint full-pool partition across R updates
Ω_t	batch trajectory cache pool	contains $N = BK$ state–time–target cells
$\mathcal{S}_{t,r}$	r-th disjoint supervision subset	evaluated by student at update step $t + r - 1$
$N = BK$	complete grid-cell count	$N = 96$ ($K = 12$) or $N = 128$ ($K = 16$)
$\rho = S/N$	retained student supervision fraction	uses the current pool size; $K = 16$ gives $N = 128$, $K = 12$ gives $N = 96$; $S = 64$ therefore gives $1/2$ or $2/3$, respectively
$R(t)$	dynamic refresh stride schedule	phase decay ($R : 2 \to 1$ after T_{phase})
$\eta(t)$	dynamic student learning rate	cosine annealing to $\eta_{\min} = 10^{-5}$
T_{phase}	phase decay transition step	step 2,000 in 2,801-step extended fine-tuning
τ	current batch and teacher randomness	conditioned on in the finite-pool analysis
τ_1, τ_2	inference-quality constraints	1-NFE and 2-NFE FID thresholds in the budget formulation
FPC	finite population correction	factor $(1 - S/N)$ in sampling without replacement
$\kappa(\rho)$	sampling variance factor	$\kappa(\rho) = (1 - \rho)/(\rho N)$; multiplies V in the fixed-pool MSE
L	per-cell gradient Lipschitz constant	assumed on the parameter region used for the local bound
$G_{\max}$	norm bound used in each result	cell-gradient bound for sampling MSE; update-direction bound $\|v_t\| \le G_{\max}$ for drift
Δ_{stale}	stale trajectory gradient drift	bounded by $\|\Delta_{\text{stale}}\| \le L\eta G_{\max}$
$c = (b, i)$	one trajectory–time cell	flattened in time-major order for gathering
$x_{b,i}^T$	teacher ODE state	generated by the frozen solver
y_b^T	tiled teacher endpoint	shared across time states of trajectory b
$\ell_c(\theta)$	cell LPIPS regression loss	student regression loss, not a FID contribution
A	selected student subset	random set of exactly S distinct cells
V	within-grid gradient dispersion	$V = (N - 1)^{-1} \sum_c \|g_c - \bar{g}\|_2^2$
G	squared Dense-gradient magnitude	$\|\bar{g}\|_2^2$ in the density rule
ε	relative gradient-MSE tolerance	controls requested student-cell count
S_ε	variance-calibrated cell count	clipped to $[S_{\min}, S_{\max}]$; tolerance requires feasibility
λ, γ	density-controller smoothing and safety	EMA decay and variance inflation factor
C_T, C_S, C_O	cost ledger terms	teacher compute, student backprop, common overhead
NFE	network function evaluations	reported separately at 1-NFE and 2-NFE
FID50k	Fréchet Inception Distance	50,000 generated and 50,000 real samples