GAUGE-INDUCED TELEPORTATION FOR PARAMETER-EFFICIENT LLM UNLEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Parameter-efficient LLM unlearning has emerged as an important research direction that aims to efficiently remove specific knowledge while preserving the model's remaining general capabilities. In this paper, we first observe that existing methods typically reach the empirical Pareto frontier prematurely, with limited further improvement even when optimization is prolonged. We therefore turn to an additional degree of freedom in optimization, i.e., gauge transformation, which is largely overlooked but can induce alternative optimization trajectories under the same LoRA function. However, gauge selection itself presents a temporal dilemma: selecting a gauge too early incurs substantial long-horizon risk, whereas selecting it too late suffers from history-induced incompatibility with the accumulated optimization state. To address this issue, we first introduce an orbit-marginalized training strategy before gauge selection to improve compatibility across gauge choices. We then employ an H-step rollout proxy to identify a favorable gauge for subsequent standard optimization. Furthermore, we adopt a reference-relative objective stabilization mechanism to prevent excessive optimization of the forgetting objective. Extensive experiments across standard LLM unlearning benchmarks and diverse base models demonstrate that **G**auge-**I**nduced **T**eleportation (`GIT`) consistently pushes beyond the empirical limits of existing approaches by non-negligible margins, establishing new competitive performance under prevailing evaluation metrics.

1 INTRODUCTION

Large language models (LLMs) have advanced rapidly in recent years, relying on increasingly large and diverse training corpora (Achiam et al., 2023). Under this paradigm, concerns regarding outdated, private, or otherwise undesirable knowledge have become increasingly prominent, creating a growing need to selectively remove specific knowledge from LLMs. Following the development of machine unlearning (Geng et al., 2025), various LLM unlearning approaches have been proposed (Qiu et al., 2025), with substantial efforts devoted to designing forgetting and retention objectives (Zhang et al., 2024; Fan et al., 2026; Maini et al., 2024), mitigating conflicts between them (Zhou et al., 2026a), and training schedule (Wang et al., 2025; Yang et al., 2025b). More recently, several studies (Cha et al., 2025; Liu et al., 2026) have adopted Low-Rank Adaptation (LoRA) (Hu et al., 2021) for unlearning to improve parameter efficiency while reducing damage to the model's general capabilities.

Recent studies have increasingly formulated LLM unlearning around the trade-off between forgetting efficacy and retained utility, with several methods explicitly evaluating or advancing the empirical forget–retain Pareto frontier (Zhang et al., 2026; Xiao et al., 2026a; Entesari et al., 2026). In this paper, focusing on parameter-efficient LLM unlearning, we first empir-

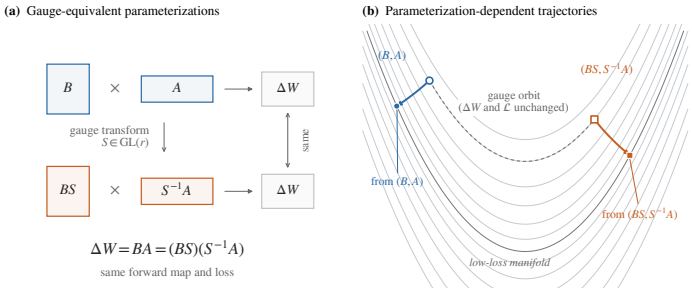

Figure 1: Illustration of gauge transformation (teleportation): current predictions and subsequent updates.

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.

ically verify this phenomenon across different approaches and further reveal that they reach this frontier *prematurely*, resulting in suboptimal forgetting–retention trade-offs (See Sec. 4). We partially attribute this limitation to a largely overlooked factor: the fixed gauge transformation adopted in LoRA parameterization. As illustrated in Figure 1, LoRA admits a family of gauge-equivalent parameterizations, i.e., $\Delta W = BA = (BS)(S^{-1}A)$, which represent the same forward mapping and loss but can induce distinct optimization trajectories Yen et al. (2025). Standard LoRA training updates its factors without explicitly comparing equivalent representations, thereby restricting the optimization trajectory to a particular gauge choice.

Building on this observation, we further demonstrate empirically that alternative gauges can induce substantially more favorable optimization trajectories, providing an opportunity to push beyond the prematurely reached empirical Pareto frontier. The central challenge, however, is *when and how* to select a favorable gauge. Our theoretical and empirical analyses reveal a temporal dilemma: selecting a gauge too early incurs substantial long-horizon risk because short-horizon evaluations may poorly reflect its eventual optimization quality, whereas selecting it too late suffers from history-induced incompatibility with the accumulated optimization state. This motivates a two-stage **G**auge-**I**nduced **T**eleportation (`GIT`) paradigm: (1) *orbit marginalization* with gauge sampling, which avoids premature commitment to a particular gauge and preserves compatibility across candidate gauges during early-stage optimization; and (2) *delayed gauge fixing* with an H-step rollout proxy, which identifies a favorable gauge for subsequent optimization. Moreover, we observe that gauge-induced teleportation can accelerate progress on the forgetting objective, potentially resulting in excessive forgetting updates at the expense of retention. We therefore introduce a reference-relative objective stabilization mechanism, inspired by NPO (Zhang et al., 2024), to bound such excessive updates and maintain a favorable forgetting–retention trade-off.

In a nutshell, our contributions can be summarized as three-fold:

- We identify that existing parameter-efficient LLM unlearning approaches reach their empirical forgetting–retention Pareto frontiers prematurely, and reveal LoRA gauge as an overlooked optimization degree of freedom that can induce alternative and more favorable optimization trajectories. We further uncover a temporal dilemma in gauge selection, where early selection suffers from long-horizon risk while late selection suffers from history-induced incompatibility.
- We propose `GIT`, a gauge-induced teleportation framework that combines early-stage orbit marginalization for preserving gauge compatibility with delayed H-step gauge fixing for selecting a favorable optimization trajectory. We further introduce a reference-relative objective stabilization mechanism to prevent excessive optimization of the forgetting objective.
- We conduct extensive experiments across standard LLM unlearning benchmarks, diverse base models, and mainstream unlearning approaches. `GIT` largely improves forgetting–retention trade-offs over strong baselines by considerable margins, establishing new competitive performance under prevailing evaluation metrics.

2 Related Work

LLM Unlearning. LLM unlearning aims to remove designated data or knowledge from a trained model while preserving retained utility (Yao & Xu, 2024). Early optimization-based methods directly reverse learning: gradient ascent maximizes the loss on forget data, while GradDiff (Maini et al., 2024) additionally minimizes the retain loss. To mitigate the resulting instability and excessive forgetting, subsequent methods regulate the forgetting objective or learning signals. NPO (Zhang et al., 2024) formulates forgetting as negative preference optimization, while SimNPO (Fan et al., 2026) removes its dependence on a frozen reference model. WGA (Wang et al., 2025) and SatImp (Yang et al., 2025b) reweight token-level forgetting signals based on confidence, saturation, and importance, whereas CE-U (Yang, 2025) and UNDIAL (Dong et al., 2025) modify target distributions or logits for more stable forgetting. RMU (Li et al., 2024) instead redirects internal representations of forget examples. More recently, LoRA-based methods improve parameter efficiency by restricting unlearning to low-rank adapters: LUNE (Liu et al., 2026) learns negative-only LoRA updates to manipulate targeted representations, while ReGLU (Xiao et al., 2026b) combines representation-guided initialization with retain-orthogonal regularization to identify selective unlearning subspaces. These studies highlight both the efficiency of low-rank unlearning and the importance of controlling its update

subspace to balance forgetting and retention. Different from these approaches, GIT exploits the gauge freedom inherent in LoRA parameterization as an additional optimization degree of freedom, enabling alternative optimization trajectories without increasing the expressive capacity of the adapter.

Symmetry Teleportation. Symmetry teleportation searches parameterizations with identical or approximately identical losses but different optimization geometries, and can be broadly categorized by whether loss invariance is exact. *Exact-invariant approaches* enforce $\mathcal{L}(g\cdot\theta) = \mathcal{L}(\theta)$ by construction. Neural Teleportation derived function-preserving transformations from quiver representation theory (Armenta et al., 2023), while Zhao et al. (Zhao et al., 2022a) formalized teleportation over loss-invariant group orbits to accelerate optimization. Subsequent studies connected such symmetries to flat minima and conserved gradient-flow quantities (Zhao et al., 2022b), established broader convergence and generalization properties (Zhao et al., 2024), generalized teleportation to exact level-set constraints without explicit group actions (Mishkin et al., 2024), and characterized symmetry-induced paths between minima (Zhao et al., 2025). Tran et al. (Tran et al.) further derived architecture-specific symmetries for Transformers. Since exact invariance often requires known architecture-dependent symmetries, *approximate-invariant approaches* instead adopt data-dependent or penalty-based constraints. Wu et al. (Wu et al., 2026) preserve sampled outputs through input-null-space projection, while COST (Zhou et al., 2026b) learns LoRA perturbations that approximately preserve task losses while reducing multi-task conflicts. BOOST (Yang et al., 2025a) extends this paradigm to long-tailed recognition by seeking approximately loss-preserving regions with weaker head–tail conflicts. These relaxations improve scalability, but operate near a loss level set rather than on an exactly invariant symmetry orbit. In contrast, GIT leverages the exact gauge equivalence inherent in LoRA parameterization to realize loss-invariant teleportation for LLM unlearning, a perspective that remains largely unexplored in this domain.

Discussion with counterparts. To the best of our knowledge, LoL (Putterman et al., 2025), LoRA-RITE Yen et al. (2025), and PolarGradM (Lau & Su, 2026) are works most closely related to ours, as all exploit the gauge structure of LoRA for optimization. However, LoL, LoRA-RITE, and PolarGradM primarily seek gauge-invariant architectures or optimization trajectories, thereby eliminating the influence of gauge transformations on optimization. In contrast, GIT actively exploits such gauge transformations as productive degrees of freedom, jointly optimizing over gauge-equivalent trajectories in the early stage and subsequently selecting a desirable fixed gauge for late-stage optimization.

3 PRELIMINARY

LLM Unlearning. Let π_{θ_0} be a model trained or fine-tuned on $\mathcal{D} = \mathcal{D}_f \cup \mathcal{D}_r$, where $\mathcal{D}_f$ and $\mathcal{D}_r$ are the forget and retain sets. Unlearning seeks a model that removes the influence of $\mathcal{D}_f$ while preserving useful behavior on $\mathcal{D}_r$. An ideal reference is $\pi_{\theta^\star_{-f}}$, retrained without $\mathcal{D}_f$, although such retraining is typically prohibitively expensive (Maini et al., 2024; Zhang et al., 2024).

We keep the backbone of π_{θ_0} frozen and write θ for the trainable LoRA factors; dependence on the frozen backbone is implicit in the losses below. With both component losses expressed under a minimization convention, the unlearning problem has two objectives:

$$\min_{\theta} \big(\mathcal{L}_f(\theta; \mathcal{D}_f), \mathcal{L}_r(\theta; \mathcal{D}_r, \theta_0)\big).$$

Here, minimizing $\mathcal{L}_f$ promotes forgetting, while minimizing $\mathcal{L}_r$ preserves retained behavior. For minibatches $\xi = (\xi_f, \xi_r)$, we write the scalarized training objective as

$$\mathcal{L}(\theta; \xi) = \lambda_f \mathcal{L}_f(\theta; \xi_f) + \lambda_r \mathcal{L}_r(\theta; \xi_r), \tag{1}$$

where $\lambda_f, \lambda_r \geq 0$ control the trade-off. Different unlearning methods may instantiate the component losses differently; our gauge procedure does not require a particular form of $\mathcal{L}_f$ (Fan et al., 2026; Dorna et al., 2026). During optimization, $z_t = (\theta_t, \omega_t)$ denotes the trainable LoRA factors together with their optimizer state.

Pareto Concept. For the two minimization objectives in Eqn. 1, a feasible solution θ' Pareto dominates θ if $\mathcal{L}_j(\theta') \leq \mathcal{L}_j(\theta)$ for $j \in \{f, r\}$, with at least one strict inequality. A solution is Pareto

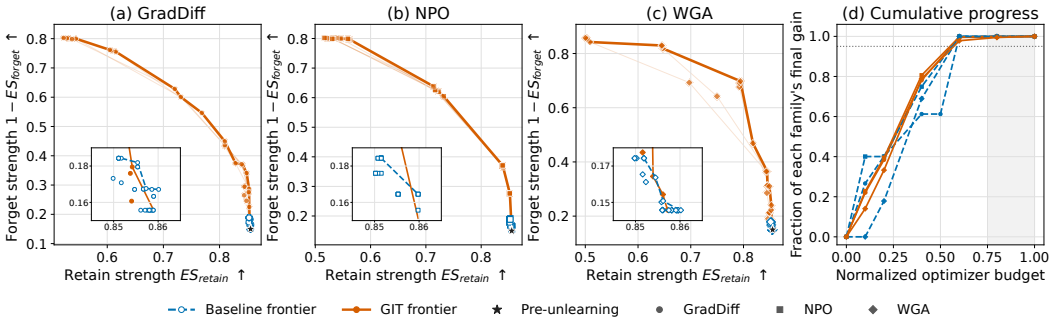

Figure 2: Empirical Identification of premature saturation for fix-gauge LLM unlearning approaches (GradDiff, NPO, and WGA). Please refer to Appendix Sec. A.5 for experiment setup.

optimal if no feasible solution dominates it. The set of such solutions is the *Pareto set*; its image in objective space is the *Pareto frontier*. Our experiments assess a distinct, evaluation-space notion. Let

$$\mathbf{e}(\theta) = \big(1 - \mathrm{ES}_{\mathrm{forget}}(\theta), \mathrm{ES}_{\mathrm{retain}}(\theta)\big) \tag{2}$$

collect forget and retain strength (Dorna et al., 2026), both oriented so that larger is better. Given the finite set $\mathcal{C}$ of evaluated runs and checkpoints, its *empirical Pareto frontier* contains the points $\mathbf{e}(\theta)$, $\theta \in \mathcal{C}$, that are not dominated coordinate-wise by another point in $\mathcal{C}$. This frontier is conditional on the tested methods, hyperparameters, and compute budgets; it need not coincide with the Pareto frontier of the training losses or with a globally optimal frontier.

4 Motivation and Observation

4.1 Observations on the LLM Unlearning Saturation

As our initial motivation, we empirically observe that existing LLM unlearning methods reach their empirical Pareto frontiers prematurely, with the corresponding results presented in Figure 2. Specifically, Figure 2 (a)–(c) depict the empirical Pareto frontiers of GradDiff, NPO, and WGA, respectively. As further quantified in Figure 2 (d), these methods have already achieved nearly all of their eventual Pareto-frontier improvement by approximately 60% of the training process, leaving the remaining 40% with only marginal gains. This premature saturation motivates us to investigate whether the empirical Pareto frontier can be further advanced without introducing additional trainable parameters. Under the LoRA parameterization, we identify a key yet largely overlooked factor: the fixed gauge transformation. Specifically, LoRA admits a family of gauge-equivalent parameterizations, $\Delta W = BA = (BS)(S^{-1}A)$, which represent the same model update but can induce distinct optimization trajectories. Conventional LoRA implicitly commits to a fixed gauge throughout training, potentially restricting its optimization trajectory and contributing to the observed premature Pareto saturation. As comparison in advance, our `GIT` reaches operating points outside parts of the baseline envelope, suggesting that the saturation can be alleviated by changing the optimization geometry rather than merely extending a baseline trajectory.

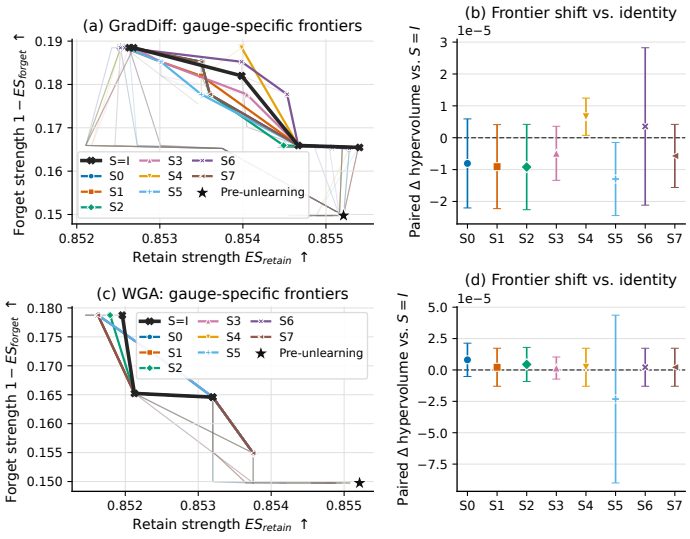

Figure 3: Frontier and HV of different gauge initialization ($S*$). Refer to Appendix Sec. A.6 for more details.

4.2 GAUGE-INDUCED OPTIMIZATION TRAJECTORIES

To demonstrate that gauge choice is a key factor affecting LoRA optimization in LLM unlearning, we further conduct experiments with diverse gauge transformation initializations. To reduce the influence of randomness, all results are averaged over three random seeds, with the corresponding results presented in Figure 3. As shown, different gauge choices lead to substantially different optimization trajectories and heterogeneous frontier shifts. More importantly, several alternative gauges consistently yield more favorable empirical Pareto frontiers than the conventional identity gauge. These results indicate that the gauge choice is not merely a redundant reparameterization from an optimization perspective; instead, selecting an appropriate gauge can steer training toward more favorable forgetting–retention trade-offs and potentially push beyond the empirical Pareto frontier reached under the default gauge, as demonstrated in Figure 2.

4.3 WHY NOT SIMPLY SELECT A GAUGE?

Selection at the beginning of training has a horizon problem. A zero-step function-based score cannot distinguish gauge-equivalent representatives; a short rollout reveals immediate optimizer behavior but leaves most of training unobserved. Delaying selection shortens this unobserved horizon, yet creates another difficulty: the incoming LoRA factors and Adam moments have been shaped by the preceding update geometry. Teleporting to a new gauge preserves the current function, but does not recreate a candidate-aligned optimization history.

Orbit marginalization we adopted (see Sec. 5.1) avoids committing every prefix update to the identity geometry. To characterize one such update, fix the same pre-step state z and mini-batch for all branches. Let $X_z(S)$ be the post-update state obtained under gauge S and mapped back to common coordinates, including both LoRA factors and Adam moments. For $S \sim \mu$, define

$$m_z = \mathbb{E}_S X_z(S), \qquad V_z = \mathbb{E}_S \|X_z(S) - m_z\|^2, \qquad \Delta_z = \|X_z(I) - m_z\|^2,$$

using a fixed weighted Euclidean norm on the augmented state. For independent samples $S_1, \ldots, S_M \sim \mu$, let $\bar{X}_{z,M} = M^{-1} \sum_{j=1}^{M} X_z(S_j)$.

Theorem 1 (One-step incoming-state compatibility)**.** *For an independent candidate $T \sim \mu$,*

$$\mathbb{E}_T \|X_z(I) - X_z(T)\|^2 = V_z + \Delta_z, \tag{3}$$

$$\mathbb{E}_{S_{1:M},T} \|\bar{X}_{z,M} - X_z(T)\|^2 = V_z + \frac{V_z}{M}. \tag{4}$$

Thus, orbit averaging has lower expected augmented-state mismatch at this anchor whenever $\Delta_z > V_z/M$.

Please refer to Appendix Sec. C for proof. The theorem is local to one transition from a shared state. It explains why averaging mapped-back factor updates and Adam moments can provide a less identity-specialized incoming state; it neither guarantees a smaller mismatch for every candidate nor establishes an ordering of complete training trajectories. In particular, an augmented-state distance does not automatically order the nonlinear response of Adam. We therefore measure that response directly.

Let $u_I = X_z(I)$, $u_M = \bar{X}_{z,M}$, and $u_S = X_z(S)$. After transporting an incoming state u into candidate gauge S, let $D_S(u)$ denote the effective LoRA-weight change from one matched Adam step. We define

$$C_{\mathrm{dyn}}(u, S) = \frac{\|D_S(u) - D_S(u_S)\|_F^2}{\|D_S(u_S)\|_F^2 + \epsilon}. \tag{5}$$

This quantity measures candidate-conditioned effective-update mismatch; it is not the augmented-state distance in Theorem 1. Figure 4 tests the proposed links at a matched late anchor. Orbit-MC (orbit marginalization) incoming states exhibit substantially smaller C_{dyn} than identity incoming states (Figure 4 (a)). Under matched candidate gauges and finite tail budgets, they also exhibit lower tail loss oscillation (Figure 4 (b)). Finally, 7 of 8 (88%) candidates obtain a higher final score [1] with

[1] We define $q_S(u_*) = \frac{1-\mathrm{ES}_{\mathrm{forget}}(\theta^{S,*}) + \mathrm{ES}_{\mathrm{retain}}(\theta^{S,*}) + \mathrm{MU}(\theta^{S,*})}{3}$, qualifying the overall unlearning performance with the combination of ES.Forget, ES.Retain, and Utility.

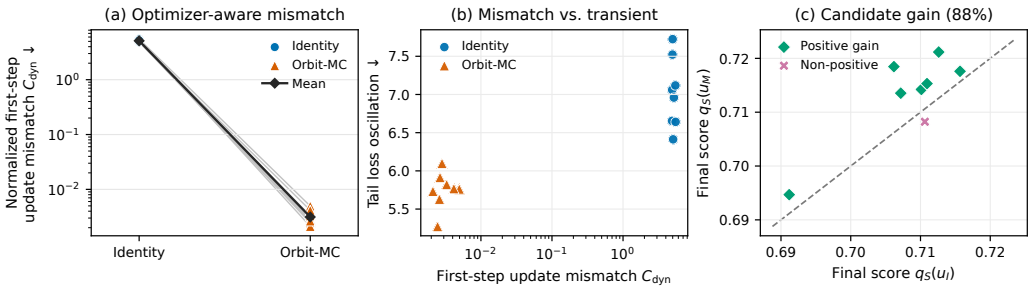

Figure 4: Orbit averaging improves candidate-gauge transitions. At a matched anchor, (a) orbit-MC incoming updates have lower optimizer-aware mismatch; (b) lower mismatch is accompanied by less tail oscillation; (c) orbit-MC achieves a higher final score for 7 of 8 candidate gauges.

orbit-MC rather than identity incoming updates (Figure 4 (c)). These are complementary observations: the state-level theorem motivates the intervention, while the matched experiment tests its realized optimizer and endpoint effects.

5 Principal Design

In this section, we introduce the overall design of GIT. As illustrated in Figure 5, unlike conventional fixed-gauge optimization, GIT consists of two main stages: *orbit marginalization* for early-stage gauge-compatible training, followed by *delayed gauge fixing*, where an H-step rollout proxy is used to select a favorable gauge for subsequent optimization. Finally, we introduce a saturation-controlled forgetting mechanism to prevent excessive optimization of the forgetting objective.

5.1 Orbit Marginalization Training

Assume the baseline unlearning objective described in Eqn. 1 as: $\mathcal{L}(\theta;\xi) = \lambda_f \mathcal{L}_f(\theta;\xi_f) + \lambda_r \mathcal{L}_r(\theta;\xi_r)$, where both component losses follow a minimization convention. The method does not require a particular definition of $\mathcal{L}_f$. Let $z_t = (\theta_t, \omega_t)$ denote the trainable parameters and optimizer state. Importantly, the object being marginalized is an optimizer transition, rather than the current loss, which is already constant along a gauge orbit.

For a sampled gauge S, define

$$\Phi_t^S(z_t;\xi_t) = \mathcal{T}_S^{-1}\left[\mathcal{U}_t\left(\mathcal{T}_S z_t;\xi_t\right)\right], \tag{6}$$

where $\mathcal{U}_t$ performs one optimizer update and $\mathcal{T}_S$ specifies the corresponding transformation of both parameters and optimizer state. The inverse transformation returns the updated state to the common storage coordinates. During the first K optimizer steps, we independently sample M gauges at each step and apply

$$S_{t,1},\ldots,S_{t,M} \overset{\text{i.i.d.}}{\sim} \mu_b, \qquad z_{t+1} = \frac{1}{M}\sum_{m=1}^{M} \Phi_t^{S_{t,m}}(z_t;\xi_t). \tag{7}$$

All branches start from the same pre-step state and use the same mini-batch realization. Thus, Eqn. 7 averages mapped-back factor updates and optimizer moments, not a sequence of M optimizer steps. For prefix training, we use positive diagonal gauges,

$$S = \mathrm{diag}(e^{u_1},\ldots,e^{u_r}), \qquad u_j \sim \mathrm{Uniform}[-b,b], \tag{8}$$

independently across rank channels and LoRA blocks. This bounded distribution controls numerical conditioning while providing a tractable family of alternative update geometries.

5.2 Delayed Gauge Fixing: H-Step Rollout as Proxy

After orbit-marginalized prefix training, we select a gauge at the anchor state z_K. Since gauge transformations preserve the current model function, evaluating the current loss cannot distinguish

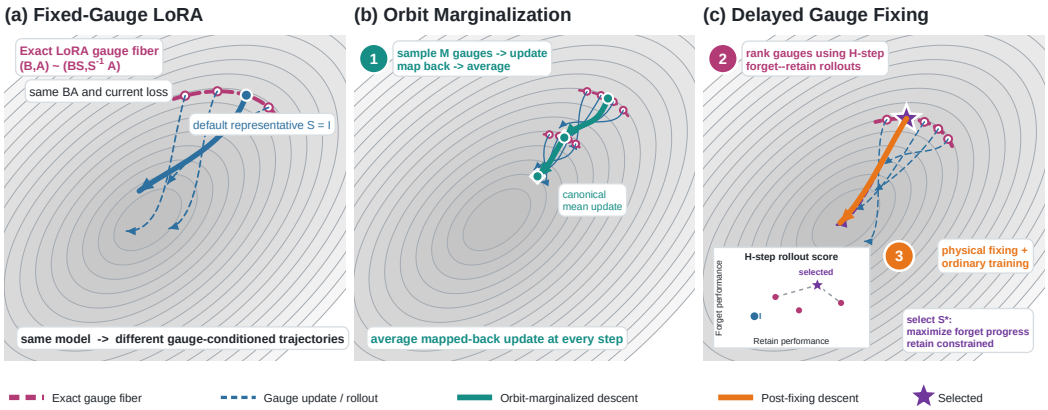

Figure 5: The Comparison between fixed-gauge LoRA and GIT. (a) is the fixed-gauge LoRA, which initialize the transformation S as I.

candidate representatives. Instead, we assess how each candidate affects subsequent optimization. Let $\mathcal{S}$ denote the candidate set and let $\mathcal{I}_S(z_K)$ denote the anchor after applying candidate S, We define the H-step rollout proxy as

$$\widehat{Q}_H(S; z_K) = -\widehat{\mathcal{J}}\left(\theta^S_{K+H}; \mathcal{D}_{\text{sel}}\right), \qquad z^S_{K+H} = \mathcal{U}_{K:K+H-1}\left(\mathcal{I}_S(z_K)\right), \tag{9}$$

where $\mathcal{U}_{K:K+H-1}$ executes H optimizer steps and $\widehat{\mathcal{J}}$ evaluates the resulting model on a fixed selection bank D_{sel} [2]. Without objective stabilization, $\mathcal{J}$ is the baseline's combined unlearning objective; Section 5.3 specifies its stabilized counterpart. We select

$$S^\star \in \arg\max_{S\in\mathcal{S}} \widehat{Q}_H(S; z_K). \tag{10}$$

After scoring, temporary rollout updates are discarded. We apply the selected intervention at the anchor and continue ordinary training. Unlike current-loss or one-step gradient statistics, this proxy observes a finite segment of the candidate's actual optimizer-dependent trajectory. It therefore incorporates interactions across successive updates, while remaining cheaper than evaluating complete continuations.

5.3 Saturation-Controlled Forgetting (SCF)

Averaging alternative update geometries does not itself prevent excessive forgetting. In particular, an unbounded forgetting objective may continue suppressing forget-set likelihood after further suppression ceases to improve the desired unlearning behavior (empirically verified in Appendix Sec. B.3). We therefore optionally introduce a reference-relative saturation controller. For forget example i, let $C_i(\theta)$ be the summed answer-token NLL, n_i its number of valid tokens, and C_i^{ref} the corresponding NLL under the frozen pre-unlearning reference. Define

$$d_i(\theta) = \frac{C_i(\theta) - C_i^{\text{ref}}}{n_i}. \tag{11}$$

The controller targets an increase of $\tau > 0$ nats per answer token, with a fixed relative transition width $w = 0.1\tau$:

$$g_i(\theta) = \sigma\left(\frac{\tau - d_i(\theta)}{w}\right), \qquad \bar{g}(\theta) = \frac{\sum_i n_i g_i(\theta)}{\sum_i n_i}.$$

Training uses

$$\mathcal{L}_{\text{stab}} = \lambda_f \,\text{sg}[\bar{g}(\theta)]\, \mathcal{L}_f^{\text{native}} + \lambda_r \mathcal{L}_r, \tag{12}$$

[2] $\mathcal{D}_{\text{sel}} = (\mathcal{D}_{f,\text{sel}}, \mathcal{D}_{r,\text{sel}})$ is a small, fixed selection bank shared by all candidates. It is used only to score the endpoint of each temporary rollout and does not supply the rollout updates. Final test metrics are evaluated on a separate test set and are never used for gauge selection.

where sg denotes stop-gradient. This preserves the baseline's native forgetting direction within each batch while attenuating its contribution as the reference-relative NLL increase reaches the target. For candidate evaluation, we use a smooth capped negative-NLL score,

$$\phi_i(\theta) = -\frac{C_i^{\text{ref}}}{n_i} - \tau + w \log\left[1 + \exp\left(\frac{\tau - d_i(\theta)}{w}\right)\right], \tag{13}$$

and replace the forget component of $\mathcal{J}$ by its token-weighted mean. This provides a bounded-below, sign-consistent forgetting criterion for rollout evaluation.

5.4 Theoretical Analysis

Theorem 1 concerns the state entering candidate-conditioned optimization. We next examine when the measured first-step response in Eqn. 5 can be informative about a finite tail.

For a candidate S, let $W_h(u, S)$ be the effective LoRA weights after h candidate-conditioned optimizer steps from incoming state u, where $h = 0, \ldots, H$. The compared tails use the same mini-batches and learning-rate schedule. Let $\ell_t(u, S)$ be the loss evaluated before tail step t, and define loss oscillation as variation beyond net change:

$$O_H(u, S) = \sum_{t=1}^{H-1} |\ell_{t+1}(u, S) - \ell_t(u, S)| - |\ell_H(u, S) - \ell_1(u, S)|. \tag{14}$$

Write $r_S^2 = \|D_S(u_S)\|_F^2 + \epsilon$, so that the measured first-step update difference is exactly $r_S\sqrt{C_{\text{dyn}}(u, S)}$.

A first-step effective update cannot, by itself, determine an entire AdamW trajectory: differences hidden in optimizer moments may emerge later. Accordingly, the next proposition is conditional on a finite-horizon response property. For the incoming branches compared in our experiment, suppose

$$\|W_h(u, S) - W_h(u_S, S)\|_F \le a_{S,h}\, r_S \sqrt{C_{\text{dyn}}(u, S)}, \qquad \begin{matrix} h = 0, \ldots, H, \\ u \in \{u_I, u_M\}, \end{matrix} \tag{15}$$

for nonnegative sensitivity coefficients $a_{S,h}$. This condition does not require contraction or a linear optimizer; it explicitly states when the first-step diagnostic remains informative over the finite tail.

Proposition 1 (Conditional finite-tail stability)**.** *Suppose Eq. 15 holds and each matched-step loss is locally L_ℓ-Lipschitz in effective LoRA weights. Then, for $u \in \{u_I, u_M\}$,*

$$|O_H(u, S) - O_H(u_S, S)| \le 2L_\ell r_S \left(\sum_{h=0}^{H-1} a_{S,h}\right) \sqrt{C_{\text{dyn}}(u, S)}. \tag{16}$$

The proposition shows that a smaller measured C_{dyn} tightens an upper bound on deviation from the candidate-aligned tail. It does *not* imply that the orbit-MC tail must have lower realized oscillation than the identity tail: two upper bounds alone cannot order their underlying quantities. Figure 4 (b) tests that actual ordering, while Figure 4 (c) tests whether the smoother observed transitions coincide with positive orbit-versus-identity finite-horizon score gains. Refer to Appendix Sec. C.2 for proof.

6 Experiments

In this section, following the setting of SatImp (Yang et al., 2025b), We conduct experiments on the TOFU (Maini et al., 2024) and MUSE (Shi et al., 2025) benchmarks. All results are carried out based on the implementation of open-sourced Open-Unlearning code (Dorna et al., 2026), and more implementation details please refer to the Appendix Sec. A. Specifically, on the TOFU benchmark, in addition to the ES score for retain (ES Re.) and unlearn (ES Un.) tasks, we also report the results of Forget Quality (FQ) and Model Utility (MU). We expect both FQ and MU to achieve high values. However, as discussed in existing literature (Wang et al., 2025), FQ is not entirely suitable for LLM unlearning tasks. The FQ value only reflects the difference between the outputs of current model and the gold-standard model, while numerous studies indicate that even with a low FQ value, the model can still exhibit excellent unlearning performance. More results, including additional evaluation across benchmarks, as well as ablation study, please refer to Appendix Sec. B.

Table 1: Comparison between unlearning approaches on TOFU. ↑ / ↓ indicate larger / smaller values are preferable. The top two results are in **bold** font for each unlearning setup. ▲ indicates our GIT augmented one outperforms its baseline.

Method	Forget-1%				Forget-5%				Forget-10%			
	ES Re.↑	ES Un.↓	FQ↑	MU↑	ES Re.↑	ES Un.↓	FQ↑	MU↑	ES Re.↑	ES Un.↓	FQ↑	MU↑
Phi-1.5												
Original	0.8565	0.8502	0.0013	0.3910	0.8565	0.8477	3.0e-13	0.3910	0.8565	0.8566	1.9e-22	0.3910
GA	0.8527	0.6861	0.0030	0.3970	0.4513	0.3643	9.6e-10	0.4040	0.0434	0.0396	**2.6e-5**	0.0090
CEU	**0.8578**	0.8231	0.0030	**0.4062**	0.1150	0.0921	3.6e-9	0.3600	0.0365	**0.0340**	8.5e-19	0
SatImp	0.8551	0.8607	0.0013	0.3911	0.8559	0.8431	3.0e-13	0.3958	0.8703	0.8517	9.5e-24	0.3973
GradDiff	0.8555	0.6886	0.0013	0.3946	0.7479	0.5464	5.9e-14	0.3979	0.2786	0.1979	5.6e-20	0.3775
GradDiff-GIT	0.7181	0.1063▲	0.0541▲	0.3743	0.8068▲	0.0982▲	**3.0e-5**▲	**0.4161**▲	0.8894▲	**0.0340**▲	1.5e-198	0.3866▲
UNDIAL	0.4826	0.4097	1.8e-14	0.3904	0.4487	0.3306	1.9e-9	0.4011	0.8585	0.8402	**0.0030**	0.4046
UNDIAL-GIT	0.5438▲	**0.0341**▲	0.0541▲	0.3889	0.2594	**0.0356**▲	4.6e-7▲	0.3962	0.1381	0.0348▲	1.8e-14	0.3735
NPO	0.8533	0.6700	0.0013	0.3960	0.4795	0.3895	9.6e-10	0.4069	0.1825	0.1456	1.1e-9	0.3532
NPO-GIT	0.7565	0.1093▲	**0.0971**▲	0.3900	0.8126▲	0.0911▲	**3.0e-3**▲	0.4095▲	0.8856▲	0.0992▲	4.3e-10	0.3977▲
SimNPO	0.8547	0.8502	0.0013	0.3919	0.8567	0.8266	3.0e-13	0.3951	0.8707	0.8328	2.1e-24	0.3972
SimNPO-GIT	0.8492	0.4163▲	0.0013	**0.3972**▲	**0.8853**▲	0.3899▲	6.6e-12▲	0.4039▲	**0.9246**▲	0.3295▲	2.0e-23▲	**0.4068**▲
WGA	**0.8555**	0.8018	0.0013	0.3941	0.8539	0.6802	3.0e-13	0.3901	0.8481	0.5333	9.0e-23	0.3988
WGA-GIT	0.8354	**0.0934**▲	**0.0971**▲	0.3907	**0.8749**▲	**0.0748**▲	2.4e-6▲	**0.4112**▲	**0.9160**▲	0.0388▲	2.2e-11▲	**0.4144**▲
LLaMA2-7B												
Original	0.9885	1.0000	0.0030	0.6275	0.9885	0.9808	5.9e-14	0.6275	0.9885	0.9820	2.0e-25	0.6275
GA	0.9883	0.9649	0.0030	0.6280	0.9745	0.8889	5.9e-14	0.6284	0.1043	0.0954	**3.1e-7**	0.4111
CEU	0.9882	1.0000	1.3e-3	0.6312	0.7237	0.6570	6.5e-13	**0.6497**	0.0290	**0.0269**	2.2e-17	8.1e-4
SatImp	**0.9888**	1.0000	1.3e-03	0.6257	0.9891	0.9772	2.6e-14	0.6215	**0.9907**	0.9801	8.1e-27	0.6151
GradDiff	0.9887	0.9649	0.0030	0.6262	0.9823	0.8754	1.3e-13	0.6233	0.6854	0.5449	1.9e-22	0.6020
GradDiff-GIT	0.6623	0.1583▲	**0.0286**▲	0.5970	0.8934	**0.0468**▲	9.5e-79	0.6328▲	0.9783▲	**0.0295**▲	5.2e-197	**0.6328**▲
UNDIAL	0.9878	1.0000	0.0013	**0.6320**	0.8194	0.7389	1.3e-13	**0.6498**	0.5502	0.4970	2.0e-25	**0.6419**
UNDIAL-GIT	0.6729	**0.1103**▲	0.0030▲	**0.6570**▲	0.3764	0.0554▲	1.4e-12▲	0.6315	0.2755	0.0463▲	3.9e-22▲	0.6152
NPO	0.9881	0.9649	0.0013	0.6278	0.9707	0.8914	2.6e-14	0.6305	0.2433	0.2104	5.1e-16	0.5287
NPO-GIT	0.6183	**0.1330**▲	**0.0541**▲	0.5976	0.8534	0.0887▲	**0.1779**▲	0.6331▲	0.9822▲	0.0502▲	**0.1111**▲	0.6198▲
SimNPO	**0.9889**	1.0000	0.0030	0.6256	0.9888	0.9683	2.6e-14	0.6207	0.9904	0.9665	8.1e-27	0.6164
SimNPO-GIT	0.9793	0.7288▲	0.0030	0.6158	0.9843	0.6538▲	3.1e-12▲	0.6072	0.9854	0.4928▲	3.4e-21▲	0.6101
WGA	0.9885	0.9737	0.0030	0.6265	**0.9891**	0.8966	3.0e-13	0.6222	0.9735	0.7231	3.9e-22	0.6206
WGA-GIT	0.9470	0.1451▲	0.0030	0.6217	**0.9899**▲	**0.0315**▲	**2.9e-11**▲	0.6210	**0.9927**▲	0.0311▲	9.5e-16▲	0.6072

6.1 Overall Results

Table 1 compares GIT-augmented objectives with their corresponding baselines across two model scales and three forget-set splits. Overall, GIT most consistently improves the forgetting–retention trade-off for the moderate and large removal requests. On Phi-1.5 Forget-5, augmenting NPO reduces unlearn extraction strength from 0.3895 to 0.0911, while increasing retain extraction strength from 0.4795 to 0.8126, forget quality from 9.6×10^{-10} to 3.0×10^{-3}, and model utility from 0.4069 to 0.4095. The improvement remains pronounced on Forget-10: WGA-GIT reduces unlearn extraction strength from 0.5333 to 0.0388, while improving retain extraction strength from 0.8481 to 0.9160 and model utility from 0.3988 to 0.4144. These results suggest that the gains are not merely obtained by enforcing stronger forgetting, but often coincide with better retention and utility.

The same pattern transfers to LLaMA2-7B. In particular, on Forget-10, NPO-GIT improves retain extraction strength from 0.2433 to 0.9822, reduces unlearn extraction strength from 0.2104 to 0.0502, and raises forget quality from 5.1×10^{-16} to 0.1111, while recovering model utility from 0.5287 to 0.6198. WGA-GIT also reduces unlearn extraction strength from 0.8966 to 0.0315 on Forget-5 while preserving retain extraction strength (0.9891 versus 0.9899). Similar improvements across GradDiff, NPO, SimNPO, and WGA indicate that the proposed training procedure is not tied to a particular unlearning objective or model scale.

The gains are nevertheless not uniform across all metrics. On the smaller Forget-1% split, stronger forgetting occasionally comes with reduced retain strength or model utility, where several baselines already perform strongly and leave less room for joint improvement. Certain aggressive configurations also obtain low forget quality despite favorable extraction-strength values, as observed for GradDiff-GIT on Phi-1.5 Forget-10. This also highlights the importance of evaluating unlearning with multiple complementary metrics. Thus, GIT should not be interpreted as strictly dominating its baseline in every coordinate. Rather, it substantially expands the attainable forgetting–retention trade-offs, with its largest and most consistent advantages appearing for the more challenging Forget-5% and Forget-10% settings.

AI USE STATEMENT

In this work, we used generative AI tools to assist with searching for and reviewing relevant literature, polishing and rewriting author-written text, and checking the correctness of author-developed theoretical proofs, as well as some baseline implementations. All identified literature was verified against the original sources, and all AI-assisted edits and proof checks were independently reviewed and validated by the authors. Generative AI tools were not used for any other purpose. We take full responsibility for the final content of this work.

REPRODUCIBILITY STATEMENT

We have provided sufficient details for reproducibility in the Appendix Sec. A, including general experiment setup, configurations, and method-specific settings.

REFERENCES

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Marco Armenta, Thierry Judge, Nathan Painchaud, Youssef Skandarani, Carl Lemaire, Gabriel Gibeau Sanchez, Philippe Spino, and Pierre-Marc Jodoin. Neural teleportation. *Mathematics*, 11 (2):480, 2023.

Sungmin Cha, Sungjun Cho, Dasol Hwang, and Moontae Lee. Towards robust and parameter-efficient knowledge unlearning for llms. In *International Conference on Learning Representations*, volume 2025, pp. 19276–19298, 2025.

Yijiang River Dong, Hongzhou Lin, Mikhail Belkin, Ramon Huerta, and Ivan Vulić. Undial: Self-distillation with adjusted logits for robust unlearning in large language models. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 8827–8840, 2025.

Vineeth Dorna, Anmol Mekala, Wenlong Zhao, Andrew McCallum, Zico Kolter, Zachary Lipton, and Pratyush Maini. Openunlearning: Accelerating llm unlearning via unified benchmarking of methods and metrics. *Advances in Neural Information Processing Systems*, 38, 2026.

Taha Entesari, Arman Hatami, Rinat Khaziev, Anil Ramakrishna, and Mahyar Fazlyab. Constrained entropic unlearning: A primal-dual framework for large language models. *Advances in Neural Information Processing Systems*, 38:166435–166474, 2026.

Chongyu Fan, Jiancheng Liu, Licong Lin, Jinghan Jia, Ruiqi Zhang, Song Mei, and Sijia Liu. Simplicity prevails: Rethinking negative preference optimization for llm unlearning. *Advances in Neural Information Processing Systems*, 38:1540–1567, 2026.

Jiahui Geng, Qing Li, Herbert Woisetschlaeger, Zongxiong Chen, Fengyu Cai, Yuxia Wang, Preslav Nakov, Hans-Arno Jacobsen, and Fakhri Karray. A comprehensive survey of machine unlearning techniques for large language models. *arXiv preprint arXiv:2503.01854*, 2025.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

Tim Tsz-Kit Lau and Weijie Su. Symmetry-compatible principle for optimizer design: Embeddings, lm heads, swiglu mlps, and moe routers. *arXiv preprint arXiv:2605.18106*, 2026.

Nathaniel Li, Alexander Pan, Anjali Gopal, Summer Yue, Daniel Berrios, Alice Gatti, Justin D Li, Ann-Kathrin Dombrowski, Shashwat Goel, Long Phan, et al. The wmdp benchmark: Measuring and reducing malicious use with unlearning. *arXiv preprint arXiv:2403.03218*, 2024.

Yezi Liu, Hanning Chen, Wenjun Huang, Yang Ni, and Mohsen Imani. Lune: Efficient llm unlearning via lora fine-tuning with negative examples. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 16560–16576, 2026.

Pratyush Maini, Zhili Feng, Avi Schwarzschild, Zachary C Lipton, and J Zico Kolter. Tofu: A task of fictitious unlearning for llms. *arXiv preprint arXiv:2401.06121*, 2024.

Aaron Mishkin, Alberto Bietti, and Robert M Gower. Level set teleportation: An optimization perspective. *arXiv preprint arXiv:2403.03362*, 2024.

Theo Putterman, Derek Lim, Yoav Gelberg, Michael M Bronstein, Stefanie Jegelka, and Haggai Maron. Gl equivariant metanetworks for learning on low rank weight spaces. In *The Fourth Learning on Graphs Conference*, 2025.

Ruichen Qiu, Jiajun Tan, Jiayue Pu, Honglin Wang, Xiao-Shan Gao, and Fei Sun. A survey on unlearning in large language models. *arXiv preprint arXiv:2510.25117*, 2025.

Weijia Shi, Jaechan Lee, Yangsibo Huang, Sadhika Malladi, Jieyu Zhao, Ari Holtzman, Daogao Liu, Luke Zettlemoyer, Noah Smith, and Chiyuan Zhang. Muse: Machine unlearning six-way evaluation for language models. In *International Conference on Learning Representations*, volume 2025, pp. 27797–27818, 2025.

Viet-Hoang Tran, Tan Lai Ngoc, Khanh Vinh Bui, Van-Hoan Trinh, Thieu Vo, and Tan Minh Nguyen. Accelerating transformer training: Architectural symmetry, positional encoding, and teleportation.

Qizhou Wang, Jin Zhou, Saebyeol Shin, Bo Han, Kilian Weinberger, et al. Rethinking llm unlearning objectives: A gradient perspective and go beyond. In *International Conference on Learning Representations*, volume 2025, pp. 61897–61931, 2025.

Zihao Wu, Juncheng Dong, Ahmed Aloui, and Vahid Tarokh. Teleportation with null space gradient projection for optimization acceleration. In *International Conference on Pattern Recognition*, pp. 262–276. Springer, 2026.

Zeguan Xiao, Siqing Li, Yong Wang, Xuetao Wei, Jian Yang, Yun Chen, and Guanhua Chen. Modeling llm unlearning as an asymmetric two-task learning problem. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 19467–19477, 2026a.

Zeguan Xiao, Lang Mo, Yun Chen, Lei Yang, Jiehui Zhao, Lili Yang, and Guanhua Chen. Representation-guided parameter-efficient llm unlearning. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 14602–14616, 2026b.

Bo Yang. Ce-u: Cross entropy unlearning. *arXiv preprint arXiv:2503.01224*, 2025.

Mianzimei Yang, Zhipeng Zhou, Jin Zhang, Yuanhao Pu, Hong Xie, and Defu Lian. Conflict-buffering optimization by symmetry teleportation for deep long-tailed recognition. In *Proceedings of the 33rd ACM International Conference on Multimedia*, pp. 1366–1375, 2025a.

Puning Yang, Qizhou Wang, Zhuo Huang, Tongliang Liu, Chengqi Zhang, and Bo Han. Exploring criteria of loss reweighting to enhance llm unlearning. *arXiv preprint arXiv:2505.11953*, 2025b.

Yuanshun Yao and Xiaojun Xu. Large language model unlearning. *Advances in Neural Information Processing Systems*, 37:105425–105475, 2024.

Jui-Nan Yen, Si Si, Zhao Meng, Felix Yu, Venkata Sai Surya Subramanyam Duvvuri, Inderjit Dhillon, Cho-Jui Hsieh, and Sanjiv Kumar. Lora done rite: Robust invariant transformation equilibration for lora optimization. In *International Conference on Learning Representations*, volume 2025, pp. 75815–75834, 2025.

Chenlong Zhang, Zhuoran Jin, Hongbang Yuan, Jiaheng Wei, Tong Zhou, Kang Liu, Jun Zhao, and Yubo Chen. Rule: Reinforcement unlearning achieves forget-retain pareto optimality. *Advances in Neural Information Processing Systems*, 38:39401–39434, 2026.

Ruiqi Zhang, Licong Lin, Yu Bai, and Song Mei. Negative preference optimization: From catastrophic collapse to effective unlearning. *arXiv preprint arXiv:2404.05868*, 2024.

Bo Zhao, Nima Dehmamy, Robin Walters, and Rose Yu. Symmetry teleportation for accelerated optimization. *Advances in neural information processing systems*, 35:16679–16690, 2022a.

Bo Zhao, Iordan Ganev, Robin Walters, Rose Yu, and Nima Dehmamy. Symmetries, flat minima, and the conserved quantities of gradient flow. *arXiv preprint arXiv:2210.17216*, 2022b.

Bo Zhao, Robert M Gower, Robin Walters, and Rose Yu. Improving convergence and generalization using parameter symmetries. In *International Conference on Learning Representations*, volume 2024, pp. 55008–55035, 2024.

Bo Zhao, Nima Dehmamy, Robin Walters, and Rose Yu. Understanding mode connectivity via parameter space symmetry. *arXiv preprint arXiv:2505.23681*, 2025.

Shiji Zhou, Tianbai Yu, Zhi Zhang, Heng Chang, Xiao Zhou, Dong Wu, and Han Zhao. Efficient utility-preserving machine unlearning with implicit gradient surgery. *Advances in Neural Information Processing Systems*, 38:4531–4569, 2026a.

Zhipeng Zhou, Ziqiao Meng, Pengcheng Wu, Peilin Zhao, and Chunyan Miao. Continual optimization with symmetry teleportation for multi-task learning. *Advances in Neural Information Processing Systems*, 38:171209–171230, 2026b.

Appendix for "Gauge-Deduced Teleportation for Parameter-Efficient LLM Unlearning"

A Benchmarks and Experiment Setup **15**

A.1 Benchmarks and Evaluation Metrics 15

A.2 General Experiment Setup 16

A.3 `GIT` Configuration 17

A.4 Baselines and Method-Specific Settings 18

A.5 Experiment Setup of Frontier Construction in Figure 2 18

A.6 Experiment Setup of Gauge Initialization in Figure 3 19

B Additional Experimental Results **19**

B.1 More Evaluation 19

B.2 Ablation Study 19

B.3 Excessive Forgetting Loss 20

C Theoretical Analysis **22**

C.1 Proof of Theorem 1 22

C.2 Proof of Proposition 1 22

C.3 A Conditional Endpoint-Score Consequence 22

C.4 Why the Response Condition Is Explicit 23

D Algorithm **23**

E Limitations **23**

F Conclusion **23**

A BENCHMARKS AND EXPERIMENT SETUP

A.1 BENCHMARKS AND EVALUATION METRICS

We evaluate GIT on two complementary language-model unlearning benchmarks: TOFU (Maini et al., 2024), which evaluates targeted knowledge removal in synthetic question–answer data, and MUSE (Shi et al., 2025), which evaluates unlearning on larger-scale real-world text corpora.

A.1.1 TOFU

Dataset. TOFU contains 4,000 question–answer pairs associated with 200 fictitious authors, with 20 question–answer pairs per author. Following the standard protocol, we consider Forget-1%, Forget-5%, and Forget-10%, corresponding to 2, 10, and 20 forgotten authors, respectively. The remaining authors constitute the retain set. Evaluation is conducted on the forget, retain, real-author, and world-fact subsets.

We evaluate Phi-1.5 and LLaMA-2-7B. For each architecture, unlearning starts from the checkpoint fine-tuned on the complete TOFU training set. A separately trained retain-only model is used as the reference for Forget Quality and other reference-relative metrics.

Extraction Strength. Extraction Strength measures how much of a target answer remains recoverable when progressively longer answer prefixes are supplied. For a prompt–answer pair (x, y), let

$$k_\theta(x, y) = \min \{k : f_\theta(y_{<k} \mid x) = y_{\geq k}\}. \tag{17}$$

We define

$$\mathrm{ES}_\theta(x, y) = 1 - \frac{k_\theta(x, y)}{|y|}, \tag{18}$$

and report its mean over the corresponding evaluation subset. Lower forget-set Extraction Strength $\mathrm{ES}_{\mathrm{Un}}$ indicates stronger removal, whereas higher retain-set Extraction Strength $\mathrm{ES}_{\mathrm{Re}}$ indicates better retention.

Forget Quality. Forget Quality (FQ) compares the truth-ratio distribution of the unlearned model on forgotten examples with that of the retain-only reference model. Following TOFU, we report the p-value of a two-sample Kolmogorov–Smirnov test. A higher value indicates that the unlearned model more closely resembles a model that was never trained on the forgotten data.

Model Utility. Model Utility (MU) measures performance outside the forget set. It is the harmonic mean of nine quantities: answer probability, truth ratio, and ROUGE recall on the retain, real-author, and world-fact subsets. Higher MU indicates better preservation of general model capability.

Empirical Pareto frontier. We represent each evaluated model using

$$(1 - \mathrm{ES}_{\mathrm{Un}}, \mathrm{ES}_{\mathrm{Re}}), \tag{19}$$

where larger values are preferred along both axes. An evaluated checkpoint is dominated if another checkpoint is no worse in either coordinate and strictly better in at least one. The resulting frontier is empirical because it is defined only over the evaluated methods and hyperparameter configurations.

A.1.2 MUSE

Dataset. MUSE (Shi et al., 2025) evaluates unlearning on two real-world textual corpora: NEWS and BOOKS. Each corpus provides a forget set $\mathcal{D}_f$, a retain set $\mathcal{D}_r$, an in-distribution holdout set $\mathcal{D}_h$, verbatim text for continuation-based evaluation, and question–answer pairs for knowledge-based evaluation.

NEWS consists of BBC news articles collected after August 2023. Its forget, retain, and holdout sets contain disjoint articles. The forget and retain corpora contain approximately 0.8M and 1.6M tokens, respectively. We use the official MUSE-News target checkpoint based on LLaMA-2-7B.

BOOKS uses the Harry Potter book series as the forget corpus and Harry Potter FanWiki articles as the retain corpus. The forget and retain corpora contain approximately 1.1M and 0.5M tokens, respectively. This setting tests whether the model can remove the original copyrighted text while preserving related domain knowledge. We use the official MUSE-Books target checkpoint based on ICLM-7B.

MUSE defines six evaluation dimensions. Following the commonly used primary evaluation protocol, we report its first four metrics: Verbatim Memorization, Knowledge Memorization, Privacy Leakage, and Utility Preservation. We do not use the scalability and sequential-sustainability protocols in the main comparison.

Verbatim Memorization. For a sequence x and a prefix length ℓ, the model is prompted with $x_{:\ell}$ and asked to generate its continuation. Verbatim Memorization is

$$\mathrm{VerbMem}(\theta, \mathcal{D}_f) = \frac{1}{|\mathcal{D}_f^{\mathrm{verb}}|} \sum_{x \in \mathcal{D}_f^{\mathrm{verb}}} \mathrm{ROUGE\text{-}L}\left(f_\theta(x_{:\ell}), x_{\ell:}\right). \tag{20}$$

Lower VerbMem indicates that the model is less able to reproduce forgotten text verbatim.

Knowledge Memorization. Let $\mathcal{D}_f^{\mathrm{QA}}$ denote question–answer pairs derived from the forget corpus. Knowledge Memorization is defined as

$$\mathrm{KnowMem}(\theta, \mathcal{D}_f) = \frac{1}{|\mathcal{D}_f^{\mathrm{QA}}|} \sum_{(q,a) \in \mathcal{D}_f^{\mathrm{QA}}} \mathrm{ROUGE\text{-}L}\left(f_\theta(q), a\right). \tag{21}$$

Lower forget-set KnowMem indicates stronger removal of knowledge associated with the forgotten corpus.

Privacy Leakage. MUSE measures membership leakage using the Min-$K\%$ probability attack. Let

$$\mathrm{AUC}\left(\theta; \mathcal{D}_f, \mathcal{D}_h\right) \tag{22}$$

denote the attack AUC for distinguishing the forget set from the in-distribution holdout set. Privacy Leakage is computed relative to the retain-only model θ_{-f}:

$$\mathrm{PrivLeak}(\theta) = \frac{\mathrm{AUC}(\theta; \mathcal{D}_f, \mathcal{D}_h) - \mathrm{AUC}(\theta_{-f}; \mathcal{D}_f, \mathcal{D}_h)}{\mathrm{AUC}(\theta_{-f}; \mathcal{D}_f, \mathcal{D}_h)}. \tag{23}$$

A value close to zero is preferred. Both large positive and large negative values may indicate privacy leakage caused by under-unlearning or over-unlearning, respectively.

Utility Preservation. Utility Preservation evaluates knowledge retention on question–answer pairs derived from the retain corpus:

$$\mathrm{UtilPres}(\theta, \mathcal{D}_r) = \frac{1}{|\mathcal{D}_r^{\mathrm{QA}}|} \sum_{(q,a) \in \mathcal{D}_r^{\mathrm{QA}}} \mathrm{ROUGE\text{-}L}\left(f_\theta(q), a\right). \tag{24}$$

Higher UtilPres indicates better preservation of knowledge that is not targeted for removal.

A.2 GENERAL EXPERIMENT SETUP

Implementation. Our implementation is built upon Open-Unlearning (Dorna et al., 2026). We adapt its training and evaluation pipeline to LoRA-based unlearning and implement orbit marginalization, delayed gauge selection, and saturation-controlled forgetting on top of the same codebase. All baselines and `GIT`-augmented methods use the same Open-Unlearning data and evaluation implementations.

Experiments are implemented with PyTorch 2.4.1, Transformers 4.51.3, Datasets 3.0.1, Accelerate 0.34.2, DeepSpeed 0.15.4, and PEFT. All experiments are conducted on NVIDIA A100 GPUs with 80GB of memory.

LoRA parameterization. The pretrained backbone is frozen, and unlearning updates only the LoRA parameters. For all models and benchmarks, we use LoRA rank $r = 32$, scaling parameter $\alpha_{\mathrm{LoRA}} = 64$, and zero LoRA dropout. LoRA modules are inserted into the supported attention and feed-forward projection layers. A baseline and its `GIT`-augmented counterpart always use the same LoRA parameterization.

TOFU training. We use AdamW with weight decay 0.01. The learning rate is 2×10^{-5} for Phi-1.5 and 1×10^{-5} for LLaMA-2-7B. Unlearning is performed for 10 epochs with an effective batch size of 16. The first epoch is used for learning-rate warmup, followed by linear decay. Final TOFU evaluation uses a batch size of 32.

MUSE training. For MUSE-News and MUSE-Books, we initialize from the official target checkpoints provided by MUSE. Each model is unlearned for 10 epochs using AdamW and a constant learning rate of 1×10^{-5}. We use a per-device batch size of 4 and eight gradient-accumulation steps, resulting in an effective batch size of 32.

We report evaluation of the final unlearned checkpoint. Knowledge-memorization and utility-preservation evaluations use a batch size of 16, whereas verbatim-memorization and Min-$K\%$ membership evaluations use a batch size of 8. The corresponding official retain-only checkpoint is used to compute PrivLeak.

Matched comparisons. Within each baseline–`GIT` pair, we match the pretrained checkpoint, LoRA initialization, data order, mini-batch realization, native objective, learning-rate schedule, and total optimizer-step budget. Final benchmark metrics are evaluated only after training and are never available to the gauge selector.

A.3 `GIT` Configuration

Orbit-marginalized prefix. For both TOFU and MUSE, we apply orbit marginalization during the first $K = \lceil 0.6T \rceil$ optimizer steps and use the per-step Monte Carlo estimator in Eqn. 7 with $M = 4$.

We use positive diagonal gauges

$$S = \mathrm{diag}(e^{u_1}, \ldots, e^{u_r}), \qquad u_j \sim \mathrm{Uniform}[-5, 5], \tag{25}$$

sampled independently across LoRA rank channels and adapted layers. All Monte Carlo branches start from the same incoming parameter and optimizer state and use the same mini-batch. After one branch-specific optimizer update, each branch is mapped back to the common storage coordinates.

We average the resulting LoRA-factor updates together with the corresponding Adam first- and second-moment states. Therefore, $M = 4$ represents four Monte Carlo samples of the same optimizer transition rather than four sequential optimizer steps. The nominal training-step budget remains identical to that of the corresponding baseline.

Delayed gauge fixing. At the end of the orbit-marginalized prefix, we evaluate candidate gauges using the H-step rollout proxy in Eqn. 9. The default rollout horizon is $H = 5$ optimizer steps. Proxy scores are evaluated using eight mini-batches from a fixed selection bank. The selection bank is shared across candidate gauges, while final test metrics remain inaccessible to the selector.

All candidates use identical rollout mini-batches and learning-rate schedules. After candidate scoring, the temporary rollout states are discarded. The selected gauge is then applied once to the anchor state, and ordinary training continues in the selected representative for the remaining $T - K$ steps. The selected transformation is not repeatedly reapplied at every tail step.

Candidate construction uses 20 projected outer iterations, a maximum absolute log-scale of 1.5, an outer search radius of 2.0, and five backtracking scales. We use a trust-region threshold of 10^{-4}, a retain-damage tolerance of 5×10^{-4}, and a balance regularization coefficient of 0.01.

Saturation-controlled forgetting. Unless explicitly stated otherwise, the complete GIT method includes the saturation controller described in Eqn. 12. We use a reference-relative target of $\tau = 1.0$ nat per answer token and set the transition width to $w = 0.1\tau$. The controller rescales only the native

forgetting component; it does not change the direction or formulation of the underlying unlearning loss.

During gauge selection, the forget component of the rollout score is replaced by the smooth capped criterion in Eqn. 13. This prevents a candidate from being preferred merely because it continues to increase forget-set negative log-likelihood after the desired forgetting level has already been reached.

A.4 Baselines and Method-Specific Settings

We compare GIT against representative objective-based unlearning methods. For every underlying method, its baseline and GIT-augmented versions use the same native objective and method-specific coefficients. GIT changes the optimization history and introduces delayed gauge selection, but does not replace the underlying unlearning objective.

Gradient Ascent. Gradient Ascent (GA) minimizes the negative forget-set negative log-likelihood without an explicit retain objective.

Cross-Entropy Unlearning. Cross-Entropy Unlearning (CE-U) (Yang, 2025) suppresses the target answer using a cross-entropy-based forgetting objective while excluding the first answer token from the forget loss.

Gradient Difference. GradDiff (Maini et al., 2024) combines gradient ascent on the forget set with conventional likelihood minimization on the retain set. We assign equal weights to the forget and retain components.

Negative Preference Optimization. NPO (Zhang et al., 2024) suppresses forgotten responses relative to the original model. We use inverse-temperature parameter $\beta = 0.1$ and equal coefficients for the NPO and retain objectives.

Simplified NPO. SimNPO (Fan et al., 2026) removes the reference-model term from NPO. We use $\beta = 4.5$, forget coefficient 1.0, and retain coefficient 0.125.

Weighted Gradient Ascent. WGA (Wang et al., 2025) reweights token-level forgetting gradients according to their current likelihood. We use $\beta = 1.0$ and equal coefficients for the forget and retain terms.

UNDIAL. UNDIAL (Dong et al., 2025) constructs a modified target distribution for forgotten examples. We use smoothing parameter $\beta = 10$ and do not add a separate retain-loss term.

Saturation Importance. SatImp (Yang et al., 2025b) attenuates token-level forgetting gradients as the corresponding knowledge becomes sufficiently suppressed. We use $\beta_1 = 5$, $\beta_2 = 1$, forget coefficient 1.0, and retain coefficient 0.1.

GIT-augmented methods. We apply GIT to GradDiff, UNDIAL, NPO, SimNPO, and WGA. For an underlying method $\mathcal{B}$, the corresponding $\mathcal{B}$-GIT variant preserves the native objective, LoRA parameterization, training data, and total optimizer budget of $\mathcal{B}$. It additionally uses the orbit-marginalized prefix, delayed H-step gauge selection, and saturation controller described above. This paired construction isolates the effect of GIT from changes in the underlying unlearning objective.

A.5 Experiment Setup of Frontier Construction in Figure 2

We evaluate frontier saturation on TOFU Forget-1% using Phi-1.5 and three representative unlearning objectives: GradDiff, NPO, and WGA. For each method, we sweep the forget-to-retain scalarization ratio $\rho = \gamma/\alpha \in \{0.1, 0.3, 1, 3, 10\}$, using $\gamma = \rho/(1+\rho)$ and $\alpha = 1/(1+\rho)$, and evaluate intermediate checkpoints throughout training. All configurations use the same learning rate, LoRA configuration, optimizer budget, and three random seeds. Baseline and GIT runs are matched under the same hyperparameter grid. Each checkpoint is represented by retain strength ES_{retain} and forget strength $1 - ES_{\text{forget}}$, with both dimensions being higher-is-better. We first average configurations

with identical hyperparameters and training progress across seeds, and then retain the non-dominated points separately for each method and family to construct the conditional empirical Pareto frontier. To quantify saturation, we additionally compute the cumulative two-dimensional hypervolume using all checkpoints available up to each normalized optimizer budget.

A.6 Experiment Setup of Gauge Initialization in Figure 3

We study whether functionally equivalent LoRA gauges induce different optimization trajectories on TOFU Forget-1% with Phi-1.5, using GradDiff and WGA as representative unlearning objectives. Starting from the same LoRA initialization, we compare the identity representative $S = I$ with eight fixed positive-diagonal gauges. For each LoRA block, the diagonal log-scales are sampled from $[-0.5, 0.5]$ and centered to satisfy $\det(S) = 1$. Each gauge is applied once before optimizer construction as $B \leftarrow BS$ and $A \leftarrow S^{-1}A$, preserving the effective update BA and hence the initial model function. Across gauges, we hold the base model, LoRA initialization, minibatch order, optimizer, learning rate, scalarization ratio, and training budget fixed. We sweep $\rho \in \{0.1, 1, 10\}$ with three paired training seeds and evaluate checkpoints throughout 20 training epochs. For each gauge, seed-averaged checkpoints are represented by retain strength ES_{retain} and forget strength $1 - ES_{\text{forget}}$, from which we construct a gauge-specific non-dominated frontier. To compare frontier quality, we compute the two-dimensional hypervolume on an identical paired checkpoint grid and report its seed-wise difference relative to $S = I$; error bars denote the sample standard deviation across training seeds. Gauge sampling seeds are independent of training seeds and are not treated as additional statistical replicates.

B Additional Experimental Results

B.1 More Evaluation

TOFU. We additional evaluate baselines with three models (LLaMA-3.2-1B/3B, and LLaMA-3.1-8B) on TOFU. As presented in Table 3, `GIT` substantially reduces ES Un. in nearly every setting, while frequently improving FQ and maintaining competitive model utility. The gains are particularly clear on Forget-10%, where `GIT` can improve all four metrics simultaneously. For example, with Llama-3.2-3B, NPO-`GIT` reduces ES Un. from 0.2885 to 0.0893, while increasing ES Re. from 0.3167 to 0.8653 and MU from 0.5679 to 0.6682. Similarly, on Llama-3.1-8B, WGA-`GIT` improves ES Re. from 0.9126 to 0.9324, ES Un. from 0.8037 to 0.1104, FQ from 8.1×10^{-27} to 2.6×10^{-8}, and MU from 0.6600 to 0.6760. Although stronger forgetting occasionally trades off some retained extraction on smaller forget splits, the overall results show that `GIT` consistently moves the underlying methods toward more favorable forgetting–utility trade-offs across model scales.

MUSE. We also conduct experiments on MUSE, and present results in Table 2. As shown, `GIT` also provides complementary improvements on the MUSE Books and News domains, demonstrating that its benefits extend beyond TOFU. Every augmented baseline improves over its original counterpart on at least one principal dimension. Notably, WGA-`GIT` improves all four News metrics, reducing VerbMem from 59.7 to 56.3 and KnowMem from 63.3 to 56.9, moving PrivLeak from -99.8 to -50.7, and increasing UtilPrest from 51.2 to 53.3. GradDiff-`GIT` also moves News PrivLeak from -99.8 to 2.94 while substantially reducing both memorization metrics, and improves Books utility from 5.5 to 55.1. These method- and domain-dependent gains support our central conclusion that `GIT` exposes optimization trajectories unavailable to a fixed-gauge baseline, thereby expanding the attainable trade-off rather than merely strengthening a single unlearning objective.

B.2 Ablation Study

Table 4 shows that OMT alone achieves strong forgetting but severely damages retained capability, resulting in low ES_{Re} and MU. Adding the H-step proxy substantially improves FQ from 1.9×10^{-6} to 0.0541 and MU from 0.0525 to 0.2698, indicating more effective gauge selection. SCF further increases ES_{Re} to 0.7181 and MU to 0.3743, while preserving the improved FQ. This retention gain is accompanied by a higher ES_{Un}, consistent with SCF attenuating excessive forgetting rather than maximizing forget-loss suppression. Overall, the full method provides the most favorable forgetting–retention balance.

Table 2: Comparison between unlearning approaches on MUSE benchmark with LLaMA2-7B. ↑ / ↓ indicate larger / smaller values are preferable. → 0 indicate near 0 is preferable. ▲ indicates our GIT augmented one outperforms its baseline.

Method	Books				News			
	VerbMem↓	KnowMem↓	PrivLeak→ 0	UtilPres↑	VerbMem↓	KnowMem↓	PrivLeak→ 0	UtilPres↑
GA	0.0	0.0	-25.8	0.0	0.0	0.0	22.5	0.0
CEU	0.0	0.4	-40.4	0.2	0.0	0.0	11.2	0.0
SatImp	99.7	41.3	-57.6	66.6	57.1	64.7	-99.8	52.9
GradDiff	0.0	0.0	-31.2	5.5	55.6	66.6	-99.8	55.4
GradDiff-GIT	9.5	30.5	-59.1	55.1▲	16.8 ▲	53.2 ▲	2.94 ▲	49.0
UNDIAL	53.2	27.0	-57.9	46.8	24.9	14.2	-99.7	21.6
UNDIAL-GIT	56.4	40.3	-53.0▲	58.8▲	19.2▲	25.2	-99.7▲	30.3▲
NPO	93.0	36.5	-59.4	58.6	49.5	65.0	-99.8	52.9
NPO-GIT	57.2▲	36.0▲	-56.7▲	64.5▲	49.7	60.4▲	-99.7▲	47.0▲
WGA	0.0	0.0	-5.95	43.7	59.7	63.3	-99.8	51.2
WGA-GIT	0.0	33.3	-32.0	64.0▲	56.3▲	56.9▲	-50.7▲	53.3▲

Table 3: Additional comparison between unlearning approaches on TOFU. ↑ / ↓ indicate larger / smaller values are preferable. ▲ indicates our GIT augmented one outperforms its baseline.

Method	Forget-1%				Forget-5%				Forget-10%			
	ES Re.↑	ES Un.↓	FQ↑	MU↑	ES Re.↑	ES Un.↓	FQ↑	MU↑	ES Re.↑	ES Un.↓	FQ↑	MU↑
Llama-3.2-1B-Instruct												
Original	0.7327	0.7432	0.0068	0.5992	0.7327	0.7272	6.5e-13	0.5992	0.7327	0.7063	1.9e-22	0.5992
GA	0.7305	0.6992	0.0068	0.5991	0.6384	0.5919	1.3e-13	0.5880	0.2434	0.2113	3.2e-15	0.5160
CEU	0.7324	0.7027	0.0068	0.5997	0.5792	0.5680	5.9e-14	0.5898	0.0349	0.0327	1.8e-04	0
SatImp	0.7354	0.7581	0.0143	0.5981	0.7298	0.7279	3.1e-12	0.5989	0.7338	0.6946	8.1e-22	0.5982
GradDiff	0.7333	0.7135	0.0143	0.5974	0.7221	0.6607	3.1e-12	0.5965	0.5402	0.4526	8.1e-22	0.5844
GradDiff-GIT	0.4122	0.1332▲	0.0068	0.5666	0.5376	0.1114▲	1.2e-10▲	0.5871	0.7049▲	0.0353▲	1.5e-198	0.5940▲
UNDIAL	0.7319	0.7097	0.0068	0.6011	0.6710	0.6339	5.9e-14	0.5987	0.4025	0.3787	4.3e-23	0.5667
UNDIAL-GIT	0.4662	0.1757▲	0.0013	0.5758	0.3248	0.1700▲	5.9e-14	0.5582	0.2604	0.1271▲	2.8e-20▲	0.5314
NPO	0.7304	0.6767	0.0068	0.5985	0.6446	0.5926	3.0e-13	0.5890	0.2530	0.2163	1.7e-15	0.5197
NPO-GIT	0.2155	0.0900▲	0.1650▲	0.5160	0.3296	0.1026▲	0.0014▲	0.5429	0.6519▲	0.1271▲	0.0003▲	0.5920▲
SimNPO	0.7333	0.7544	0.0068	0.6009	0.7314	0.7210	6.6e-12	0.5995	0.7320	0.6917	1.7e-21	0.5979
SimNPO-GIT	0.7321	0.6426▲	0.0068	0.6007	0.7538▲	0.5880▲	6.6e-12	0.5977	0.7839▲	0.5109▲	5.6e-20▲	0.5950
WGA	0.7354	0.7289	0.0143	0.5979	0.7424	0.7117	1.4e-12	0.6008	0.7444	0.6696	1.9e-22	0.6018
WGA-GIT	0.6818	0.1636▲	0.0286▲	0.5917	0.6975	0.1177▲	2.4e-08▲	0.6018▲	0.6761	0.1636▲	0.0286▲	0.5939
Llama-3.2-3B-Instruct												
Original	0.9001	0.9202	0.0143	0.6664	0.9001	0.8870	2.6e-14	0.6664	0.9001	0.8904	8.1e-27	0.6664
GA	0.8980	0.8849	0.0143	0.6653	0.8100	0.7612	1.1e-14	0.6567	0.2549	0.2262	2.8e-16	0.5409
CEU	0.8996	0.9244	0.0068	0.6672	0.7725	0.7591	8.3e-16	0.6681	0.0547	0.0496	9.9e-11	0.1248
SatImp	0.9013	0.9202	0.0143	0.6653	0.8964	0.8803	1.3e-13	0.6616	0.9034	0.8754	4.4e-25	0.6608
GradDiff	0.9013	0.9202	0.0143	0.6654	0.9001	0.8380	2.6e-14	0.6644	0.8884	0.7966	9.0e-26	0.6653
GradDiff-GIT	0.3347	0.1558▲	0.0286▲	0.6116	0.6521	0.1852▲	5.6e-17	0.6715▲	0.7555	0.0531▲	3.4e-173	0.6641
UNDIAL	0.8983	0.9483	0.0068	0.6669	0.8201	0.7935	1.4e-16	0.6678	0.4249	0.3771	6.9e-28	0.6372
UNDIAL-GIT	0.5836	0.1852▲	0.0068	0.6634	0.3336	0.0692▲	3.4e-16▲	0.6529	0.2494	0.0579▲	1.8e-26▲	0.6336
NPO	0.8972	0.8849	0.0143	0.6659	0.8304	0.7754	1.1e-14	0.6590	0.3167	0.2885	2.2e-17	0.5679
NPO-GIT	0.2796	0.0909▲	0.0971▲	0.5757	0.5306	0.0952▲	0.0030▲	0.6241	0.8653▲	0.0893▲	1.7e-8▲	0.6682▲
SimNPO	0.9008	0.9202	0.0143	0.6644	0.8967	0.8732	3.0e-13	0.6612	0.9032	0.8694	4.1e-26	0.6610
SimNPO-GIT	0.8922	0.6422▲	0.0286▲	0.6616	0.9106▲	0.6715▲	1.3e-13	0.6578	0.9336▲	0.5739▲	4.4E-25▲	0.6566
WGA	0.9030	0.9202	0.0143	0.6653	0.9070	0.8602	4.7e-15	0.6645	0.9126	0.8037	8.1e-27	0.6660
WGA-GIT	0.7870	0.2085▲	0.0286▲	0.6581	0.8882	0.1228▲	0.0163▲	0.6596	0.9234▲	0.1104▲	2.6e-08▲	0.6760▲
Llama-3.1-8B-Instruct												
Original	0.9891	0.9768	0.0030	0.6276	0.9891	0.9720	1.4e-12	0.6276	0.9891	0.9789	8.1e-27	0.6276
GA	0.9889	0.9768	0.0068	0.6286	0.9883	0.9438	6.5e-13	0.6274	0.3655	0.3187	1.2e-17	0.5189
CEU	0.9889	0.9768	0.0030	0.6326	0.8623	0.7904	6.5e-13	0.6608	0.1485	0.1300	1.5e-16	0.4715
SatImp	0.9897	0.9768	0.0068	0.6255	0.9897	0.9726	3.0e-13	0.6221	0.9916	0.9793	1.3e-28	0.6234
GradDiff	0.9895	0.9768	0.0068	0.6267	0.9893	0.9396	6.5e-13	0.6233	0.9719	0.8943	3.6e-33	0.6142
GradDiff-GIT	0.5036	0.1630▲	0.0541▲	0.5915	0.7586	0.2426▲	6.5e-13	0.5954	0.8922	0.0517▲	7.1e-184	0.6229▲
UNDIAL	0.9889	0.9768	0.0030	0.6305	0.9452	0.9096	1.3e-13	0.6582	0.7681	0.7156	4.4e-25	0.6712
UNDIAL-GIT	0.8662	0.2574▲	0.0286▲	0.6910▲	0.5692	0.1134▲	6.5e-13▲	0.6714▲	0.4159	0.0800▲	3.6e-27	0.6423
NPO	0.9892	0.9768	0.0068	0.6291	0.9848	0.9334	1.3e-13	0.6286	0.5077	0.4631	1.4e-20	0.5872
NPO-GIT	0.4281	0.1374▲	0.4046▲	0.6297▲	0.9315	0.1056▲	0.1123▲	0.6235	0.9863▲	0.0886▲	3.1e-07▲	0.6041▲
SimNPO	0.9896	0.9768	0.0068	0.6261	0.9900	0.9694	3.0e-13	0.6230	0.9918	0.9719	2.4e-29	0.6222
SimNPO-GIT	0.9788	0.7300▲	0.0143▲	0.6259	0.9865	0.7258▲	3.1e-12▲	0.6150	0.9898	0.6147▲	5.6e-29▲	0.6063
WGA	0.9891	0.9768	0.0068	0.6265	0.9902	0.9407	6.5e-13	0.6266	0.9922	0.9301	1.6e-27	0.6302
WGA-GIT	0.9350	0.2045▲	0.0971▲	0.6223	0.9423	0.2061▲	6.9e-9▲	0.6351▲	0.9940▲	0.0969▲	6.0e-11▲	0.6196

B.3 Excessive Forgetting Loss

Orbit marginalization and delayed gauge fixing improve the optimization geometry, but they may also allow the native forgetting objective to be optimized substantially beyond the level required for

Table 4: Ablation study of GIT based on GradDiff with Phi-1.5 on TOFU Forget-1%. 'OMT' represents orbit marginalization training, Proxy represents H-step rollout proxy, and SCF represents saturation-controlled forgetting.

OMT	Proxy	SCF	Forget-1%			
			ES Re.↑	ES Un.↓	FQ↑	MU↑
✓			0.0541	0.0304	1.9e-6	0.0525
✓	✓		0.0982	0.0346	0.0541	0.2698
✓	✓	✓	0.7181	0.1063	0.0541	0.3743

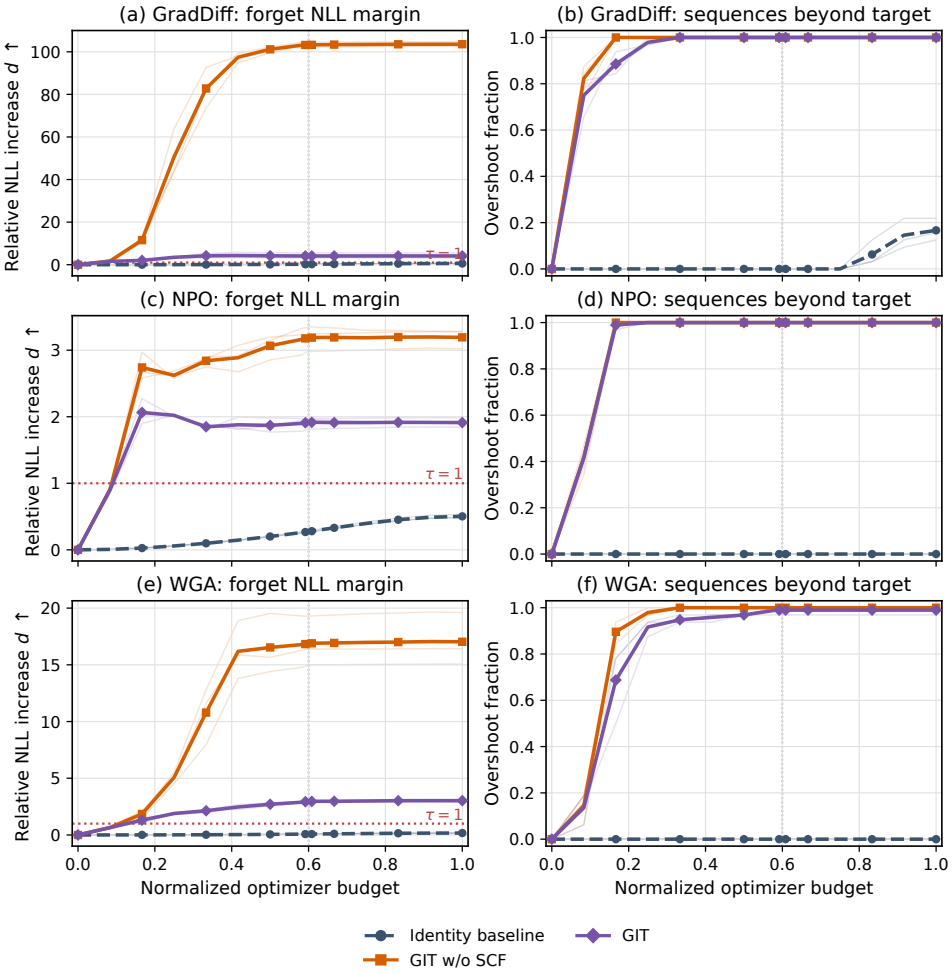

Figure 6: The results of overshoot forgetting. Panels (a,c,e) show the reference-relative forget NLL increase, and (b,d,f) show the fraction of sequences beyond $\tau = 1$. SCF substantially reduces post-target NLL growth, although it does not impose a hard per-sequence constraint.

effective unlearning. We therefore examine whether the saturation-controlled forgetting mechanism prevents such excessive optimization.

Figure 6 tracks the reference-relative forget NLL increase d throughout training. The dotted horizontal line denotes the saturation target $\tau = 1$. The left column reports the magnitude of d, while the right column reports the fraction of forget sequences satisfying $d_i > \tau$.

Without saturation control, GIT drives the forget loss far beyond the target for all three underlying objectives. At the end of training, the average NLL margin reaches approximately 104, 3.2, and

17 for GradDiff, NPO, and WGA, respectively. Adding saturation control reduces these values to approximately 4.0, 1.9, and 3.2. The reduction is particularly pronounced for GradDiff and WGA, for which the uncontrolled objective continues increasing long after nearly all forget sequences have crossed the target.

The overshoot-fraction plots provide an important qualification. For both `GIT` variants, most forget sequences eventually satisfy $d_i > \tau$. Therefore, the proposed controller should not be interpreted as a hard constraint that enforces $d_i \leq \tau$. Instead, once a sequence approaches the target, the sigmoid gate in Eqn. 12 progressively attenuates its contribution to the forgetting gradient. This substantially limits the *severity* of post-target optimization even when the target has already been crossed.

These results support the intended role of saturation-controlled forgetting: it preserves the strong forgetting dynamics enabled by `GIT` while preventing the native objective from assigning nearly uniform pressure to already forgotten examples. The controller consequently serves as a soft optimization safeguard rather than an exact stopping rule.

C Theoretical Analysis

C.1 Proof of Theorem 1

Write $X = X_z(T)$ and $m = m_z$. Since $\mathbb{E}_T(X - m) = 0$, expansion of the squared norm gives

$$\mathbb{E}_T\|X_z(I) - X\|^2 = \|X_z(I) - m\|^2 + \mathbb{E}_T\|X - m\|^2 \tag{26}$$

$$= \Delta_z + V_z. \tag{27}$$

The independent sample mean satisfies $\mathbb{E}\bar{X}_{z,M} = m$ and $\mathbb{E}\|\bar{X}_{z,M} - m\|^2 = V_z/M$. Because T is independent of $S_1, \ldots, S_M$, the cross term in the following expansion has zero expectation:

$$\mathbb{E}_{S_{1:M},T}\|\bar{X}_{z,M} - X_z(T)\|^2 = \mathbb{E}\|\bar{X}_{z,M} - m\|^2 + \mathbb{E}\|X_z(T) - m\|^2 \tag{28}$$

$$= \frac{V_z}{M} + V_z. \tag{29}$$

Subtracting the identities proves the result. The calculation holds for any fixed Hilbert-space norm used consistently across all mapped-back states. It is a same-anchor, one-step result; repeated prefix transitions do not follow from the identity without further analysis.

C.2 Proof of Proposition 1

Fix S and $u \in \{u_I, u_M\}$. Abbreviate $C = C_{\mathrm{dyn}}(u, S)$ and define $\delta_t = \ell_t(u, S) - \ell_t(u_S, S)$. The loss at tail step t is evaluated at W_{t-1}. Matched batches, the local Lipschitz condition, and Eqn. 15 give

$$|\delta_t| \leq L_\ell a_{S,t-1} r_S \sqrt{C}.$$

Using the reverse triangle inequality in Eqn. 14,

$$|O_H(u, S) - O_H(u_S, S)| \leq \sum_{t=1}^{H-1} |\delta_{t+1} - \delta_t| + |\delta_H - \delta_1| \tag{30}$$

$$\leq 2\sum_{t=1}^{H} |\delta_t| \tag{31}$$

$$\leq 2L_\ell r_S \left(\sum_{h=0}^{H-1} a_{S,h}\right) \sqrt{C}. \tag{32}$$

This proves Eqn. 16. Notice that it bounds each tail relative to the candidate-aligned reference; it does not compare the realized orbit-MC and identity oscillations directly.

C.3 A Conditional Endpoint-Score Consequence

For completeness, suppose a *continuous* endpoint functional Q is locally L_Q-Lipschitz in effective LoRA weights. Applying Eqn. 15 at $h = H$ yields

$$|Q(W_H(u, S)) - Q(W_H(u_S, S))| \leq L_Q a_{S,H} r_S \sqrt{C_{\mathrm{dyn}}(u, S)}. \tag{33}$$

For example, set

$$\delta_I^Q(S) = Q(W_H(u_S, S)) - Q(W_H(u_I, S)).$$

If

$$\delta_I^Q(S) > L_Q a_{S,H} r_S \sqrt{C_{\mathrm{dyn}}(u_M, S)},$$

then Eqn. 33 implies $Q(W_H(u_M, S)) > Q(W_H(u_I, S))$. A smaller orbit-MC mismatch makes this sufficient condition easier to satisfy, but does not ensure that it holds for any particular candidate.

The final metrics in Fig. 4 (c) may include discrete evaluation decisions and need not be Lipschitz in model weights. Their observed positive-gain fraction is therefore an empirical result, not a direct corollary of Eqn. 33.

C.4 Why the Response Condition Is Explicit

Eqn. 15 is not implied by a small C_{dyn} alone. Two incoming states can yield almost the same first effective LoRA update while retaining different Adam moments, which may produce different later updates. In that case the coefficients $a_{S,h}$ can be large and the finite-tail bound becomes uninformative. This distinction is precisely why the mechanism experiment reports all three quantities separately: first-step effective-update mismatch, realized tail oscillation, and final candidate-conditioned score.

D Algorithm

Algorithm 1 separates training into three stages. During the prefix, each Monte Carlo branch represents an alternative gauge-conditioned optimizer transition, and both the LoRA updates and Adam moments are averaged after being mapped to common coordinates. Candidate rollouts are temporary and do not consume the final training budget. After selection, the chosen gauge is applied exactly once at the anchor; subsequent tail updates continue normally in the selected representative without repeatedly applying the transformation.

E Limitations

Although `GIT` explores parameter-efficient LLM unlearning from a distinct optimization perspective and achieves promising performance, it still has several limitations. (1) `GIT` introduces additional computational overhead, mainly due to its more sophisticated training procedure and the additional rollout-based proxy evaluation. Developing more efficient alternatives remains an important direction for future work. (2) The current H-step rollout proxy is intuitive rather than specifically optimized for gauge selection. More principled proxies that better correlate with long-horizon optimization outcomes and final unlearning performance could further improve the effectiveness and efficiency of the framework. (3) `GIT` introduces several additional hyperparameters beyond those of the underlying baselines. Although our analyses suggest that these hyperparameters are relatively easy to tune and remain robust within reasonable ranges, they may still require further adjustment when transferred to substantially different models, datasets, or unlearning scenarios.

F Conclusion

We studied gauge freedom as an underexplored source of optimization diversity in LoRA-based LLM unlearning. Although gauge transformations preserve the current model function, they alter subsequent optimizer-dependent trajectories and can lead to substantially different forgetting–retention trade-offs. Motivated by the difficulty of selecting a reliable gauge from an identity-specialized optimization history, we introduced orbit-marginalized training, delayed gauge fixing through an H-step rollout proxy, and a saturation controller for the forgetting objective. Extensive experiments further show that the proposed framework improves several representative unlearning objectives across model scales and forget-set splits, particularly under larger removal requests. These results suggest that exploiting parameterization symmetry provides a promising direction for expanding the empirical Pareto frontier of efficient LLM unlearning.

Algorithm 1 Training Process of GIT

Input: pretrained model θ_0; forget and retain sets $(\mathcal{D}_f, \mathcal{D}_r)$; baseline unlearning objective $\mathcal{L}$;
Output: unlearned model θ_T
Initialize the trainable LoRA factors and Adam state as $z_0 = (\theta_0, \omega_0)$.
Use the stabilized objective in Eqn. 12 if saturation control is enabled; otherwise use the native baseline objective.
% Orbit-Marginalized Prefix
for $t = 0, \ldots, K-1$ **do**
  Sample the next forget–retain mini-batch.
  Sample M gauges independently from μ_b.
  for $m = 1, \ldots, M$ **do**
    Compute the M mapped-back gauge-conditioned transitions using Eqn. 6.
  end for
  Average the M mapped-back parameter and optimizer-state updates according to Eqn. 7 to obtain z_{t+1}.
end for
% Delayed Gauge Fixing
Construct the candidate gauges $\mathcal{S}$ at the anchor state z_K.
Fix a common rollout mini-batch stream and learning-rate schedule for all candidates.
for each candidate $S \in \mathcal{S}$ **do**
  Apply the candidate intervention to a temporary copy of z_K.
  Continue the temporary state for H matched optimizer steps.
  Evaluate its endpoint using the rollout proxy in Eqn. 9.
  Use the capped forget score in Eqn. 13 if saturation control is enabled.
  Record $\widehat{Q}_H(S; z_K)$ and discard the temporary rollout.
end for
Select $S^\star$ according to Eqn. 10.
Apply $S^\star$ once to the anchor parameters and optimizer state.
% Ordinary Tail Optimization
for $t = K, \ldots, T-1$ **do**
  Sample the next forget–retain mini-batch.
  Perform one native optimizer update using the selected representative and the chosen training objective.
end for
return θ_T