GRADIENT AS BEHAVIOR FINGERPRINT: STABLE SEQUENTIAL MODEL EDITING VIA GRADIENT DRIFT REGULARIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language models (LLMs) have achieved remarkable success across a wide range of tasks, but outdated or incorrect knowledge may persist in their parameters, leading to hallucinations and factual errors. Model editing aims to address this issue by enabling targeted updates to specific knowledge. However, in sequential editing settings, repeated updates can interfere with one another, resulting in accumulated instability and degradation of previously edited knowledge. In this paper, we revisit sequential editing from a gradient-based perspective. We use the gradient of the target log-likelihood with respect to the editing residual as a target-conditioned descriptor of first-order local perturbation response. We establish a Taylor bound showing that the gradient discrepancy between two state-specific reference centers controls the difference between their centered local responses up to a second-order remainder. Building on this diagnostic, we propose Gradient Drift Regularization (GDR). For each historical edit, GDR stores a gradient anchor at its insertion-time reference center and later computes a query gradient from the current model state when that edit is selected. GDR uses the discrepancy between these gradients to construct a fixed, dimension-wise quadratic penalty on the current editing residual, discouraging updates along coordinates with larger observed historical discrepancies. To improve efficiency, we further introduce a preconditioned key-space selection mechanism that identifies historical edits potentially affected by the current update, limiting query-gradient evaluation to a top-k subset. Experiments on LLaMA-3.1, GPT-2-XL, and Qwen-2.5 show that GDR preserves overall model performance and improves the retention of historical edits over long editing sequences. The results also reveal a strong empirical association between lower gradient drift and better historical retention.

1 INTRODUCTION

Large Language Models (LLMs) have achieved remarkable progress in natural language understanding and generation (Brown et al., 2020; Liu et al., 2024), but may retain outdated or incorrect information that leads to hallucinations and factual errors (Petroni et al., 2019; Roberts et al., 2020; Zhang et al., 2025). Model editing addresses this problem by updating specific knowledge without retraining the entire model (Cao et al., 2024; Yao et al., 2023; Zhang et al., 2024; Mazzia et al., 2024). Existing methods either modify a subset of model parameters (Meng et al., 2023a;b; Jiang et al., 2025) or introduce external modules while preserving the original weights (Huang et al., 2023; Yu et al., 2024; Hartvigsen et al., 2023; Zheng et al., 2023). However, in sequential editing, successive updates may interfere with previously edited or unrelated knowledge, resulting in accumulated degradation and, in severe cases, model collapse (Li et al., 2023; Gupta et al., 2024; Gu et al., 2024).

Recent approaches improve sequential-editing stability through different forms of update control. AlphaEdit constrains updates using a null-space projection (Fang et al., 2024), whereas HSE combines structured replay with Fisher-based regularization (Fang et al., 2025). Nevertheless, preserving early edits over long update sequences remains challenging. This motivates a complementary question: can we identify an observable signal that characterizes how the response associated with a historical edit changes as subsequent edits are applied?

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.

We study this question using the gradient of the target log-likelihood with respect to the editing residual. This edit-space gradient provides a target-conditioned descriptor of the first-order perturbation response around a specified evaluation state, where "local" refers to a neighborhood in residual coordinates rather than semantic proximity between facts. For two model states, we define gradient drift as the discrepancy between their edit-space gradients evaluated at their respective reference centers. We establish a Taylor bound relating this discrepancy to differences between the corresponding centered local responses, up to a second-order remainder. The bound motivates gradient drift as a first-order diagnostic of local-response change, without assuming that a single gradient completely characterizes model behavior or guarantees knowledge retention.

Building on this diagnostic, we propose **Gradient Drift Regularization (GDR)**. For each historical edit, GDR stores a **gradient anchor** at its insertion-time reference center. When that edit is later selected, a **query gradient** is computed under the current model state. Rather than directly aligning these gradients or minimizing their discrepancy, GDR uses the discrepancy as a weighting signal to construct a fixed, dimension-wise quadratic penalty on the current editing residual. This penalty discourages updates along coordinates associated with larger historical gradient drift. To limit query-gradient computation, we further introduce a **preconditioned key-space selection mechanism** that restricts evaluation to a top-k subset of potentially affected historical edits. This bounds the number of historical gradient evaluations per step. Figure 1 presents an overview of the framework.

In summary, our contributions are as follows: (1) We introduce edit-space gradient drift as a target-conditioned, first-order diagnostic of local-response change and establish a Taylor bound relating gradient drift between state-specific reference centers to centered local-response differences. (2) We propose GDR, which converts anchor–query gradient discrepancies into a drift-weighted quadratic penalty, together with a preconditioned key-space top-k mechanism for limiting historical query-gradient evaluation. (3) Experiments across multiple LLMs and benchmarks show that GDR improves historical-edit retention while maintaining overall editing performance, and reveal a strong empirical association between lower gradient drift and better retention over long editing sequences.

2 Related Work

Model editing paradigms. Model editing methods are commonly divided into parameter-preserving and parameter-modifying approaches. Parameter-preserving methods keep the original model weights fixed and introduce auxiliary modules or external memory to store edited knowledge, including T-Patcher (Huang et al., 2023), GRACE (Hartvigsen et al., 2023), and WISE (Wang et al., 2024a). Parameter-modifying methods instead update a subset of model parameters through meta-learning or locate-then-edit mechanisms. MEND (Mitchell et al., 2021) learns an update rule from editing gradients, while ROME (Meng et al., 2023a) and MEMIT (Meng et al., 2023b) modify feed-forward layers viewed as key–value memories (Geva et al., 2021; 2022; Dai et al., 2022). AnyEdit (Jiang et al., 2025) further generalizes parameter-modifying editing to more flexible knowledge formats. GDR belongs to the parameter-modifying paradigm and builds on the locate-then-edit framework.

Stability under sequential editing. Repeated parameter updates may accumulate interference and degrade previously edited knowledge. Existing approaches address this problem from different perspectives. PRUNE (Ma et al., 2024) regularizes the geometry of the update matrix, while AlphaEdit (Fang et al., 2024) projects updates onto a null space constructed to reduce interference with preserved knowledge. HSE (Fang et al., 2025) combines structured replay with Fisher-based parameter regularization, following the broader idea of protecting important parameters during continual learning (Kirkpatrick et al., 2017). These methods primarily constrain update geometry, estimate parameter importance, or revisit historical information. In contrast, GDR measures changes in the target-conditioned response of historical edits and uses the resulting gradient drift to weight a residual-space regularizer.

3 Preliminaries of LLM parameter editing

Problem formulation. Knowledge editing aims to update a specific factual association in a pre-trained language model without retraining the whole model. Following prior work, we denote an edit

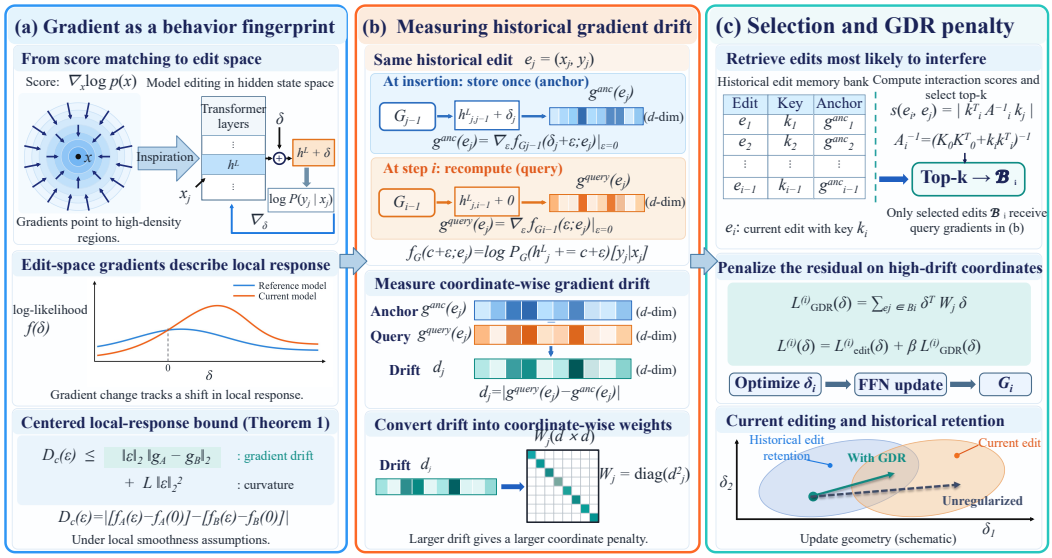

Figure 1: **Overview of GDR.** (a) Motivated by the local-gradient perspective of score matching, we use the edit-space gradient as a first-order descriptor of how the target log-likelihood responds to residual perturbations. Theorem 1 relates gradient drift to changes between centered local responses. (b) For each historical edit, GDR stores a gradient anchor at insertion time and recomputes a query gradient under the current model when that edit is selected. Their coordinate-wise discrepancy captures how the historical edit's local response has changed and determines the diagonal regularization weights. (c) A covariance-preconditioned top-k rule selects the historical edits most likely to interact with the current update. GDR then penalizes the current editing residual more strongly on coordinates with larger historical gradient drift, balancing the incorporation of the current edit with the retention of earlier edits.

as $e_i = (s_i, r_i, o_i, o_i^*)$, where s_i is the subject, r_i is the relation, o_i is the original object, and o_i^* is the desired target. Each edit induces an input–output pair (x_i, y_i), where $x_i = p(s_i, r_i)$ is a templated prompt and $y_i = o_i^*$ is the target completion. Given a sequence of edits $E = \{e_1, \ldots, e_T\}$, sequential editing aims to incorporate new knowledge while preserving previously edited and unrelated model behaviors.

Representation-level editing target. We adopt a MEMIT-style locate-and-edit framework(Meng et al., 2023b), which updates a small set of Transformer FFN layers $\mathcal{R} = \{\ell_1, \ldots, \ell_R\}$. Let $L = \max(\mathcal{R})$ denote the target layer where the edited knowledge is expected to be fully expressed. For the i-th edit, let h_i^L be the hidden representation at layer L and the last subject token under the pre-edit model G_{i-1}. Instead of directly optimizing all model parameters, the method first learns a representation-space perturbation δ_i and defines the target representation as

$$z_i = h_i^L + \delta_i. \tag{1}$$

The perturbation is optimized to make the model predict the new target y_i:

$$\mathcal{L}^{(i)}_{\text{edit}}(\delta) = \frac{1}{P} \sum_{n=1}^{P} -\log \mathbb{P}_{G_{i-1}(h_i^L \mathrel{+}= \delta)} \left[y_i \mid x^n_{\text{pref}} \oplus x_i \right], \tag{2}$$

where $G_{i-1}(h_i^L \mathrel{+}= \delta)$ denotes the forward execution of G_{i-1} with δ injected into the hidden state at layer L and the last subject token. We denote the optimized perturbation by δ_i.

From representation target to FFN key-value update. The optimized target z_i is then converted into localized parameter updates on the selected FFN layers. Following the key-value memory view of Transformer FFNs, each edit is associated with a key at an edited FFN layer. For notational simplicity, we omit the layer superscript when the edited layer is clear from context. For an input

prompt x_i, let $h^{\ell-1}$ denote the hidden representation before the FFN module of layer ℓ at the last subject token. The corresponding FFN key and value are

$$k_i = \sigma\big(W_{\text{in}}\gamma(h^{\ell-1})\big), \qquad v_i = W_{\text{out}}k_i, \tag{3}$$

where W_{in} and W_{out} are the FFN up-projection and down-projection matrices, respectively, $\gamma(\cdot)$ is layer normalization, and $\sigma(\cdot)$ is the FFN activation function. Thus, k_i is the output of the FFN up-projection and the input to the down-projection matrix W_{out}, which is the matrix modified during editing. Here, k_i determines how the edit accesses the FFN memory, while v_i is the value retrieved from the current down-projection matrix.

To write the desired representation change into the model, the residual $z_i - h_i^L$ is distributed over the edited layers, with $r_i = (z_i - h_i^L)/(L - \ell + 1)$ for the edited layer ℓ, and transformed into a target value $v_i^* = v_i + r_i$ for each edited layer. The FFN down-projection matrix is then updated so that the edited key maps to the target value:

$$\widetilde{W}_{\text{out}} = W_{\text{out}} + \Delta, \qquad \widetilde{W}_{\text{out}}k_i = v_i^*. \tag{4}$$

In the single-sample editing case, the closed-form update can be written as

$$\Delta = (v_i^* - W_{\text{out}}k_i)\, k_i^\top \left(K_0K_0^\top + k_ik_i^\top\right)^{-1}, \tag{5}$$

where K_0 denotes the background keys used to preserve existing key-value associations. The detailed update process is provided in Appendix B.1. After applying these updates to all selected layers $\ell \in \mathcal{R}$, the model state becomes G_i.

4 Why gradient drift matters in sequential editing

In sequential model editing, each new update can alter how previously inserted knowledge responds to subsequent perturbations. We characterize this change through **edit-space gradient drift**, which describes the variation in the target-conditioned first-order response of a historical edit across model states. We then relate gradient drift to differences between centered local responses through a Taylor bound.

Edit-space gradient. In the sequential editing framework introduced in Section 3, each update is parameterized by a learnable representation-space residual $\delta \in \mathbb{R}^d$. For an editing sample $e_i = (x_i, y_i)$, let h_i^L denote its target-layer representation under the model state G_{i-1}. At a residual center c, we define the edit-space gradient as

$$g^{(i)}(e_i; c) \;\triangleq\; \nabla_\epsilon \log \mathbb{P}_{G_{i-1}\left(h_i^L \mathrel{+}= c+\epsilon\right)}[y_i \mid x_i]\Big|_{\epsilon=0}. \tag{6}$$

When the residual center is clear from context, we omit c and write $g^{(i)}(e_i)$. This gradient describes the first-order variation of the target log-likelihood under a small displacement around $h_i^L + c$, where "local" refers to a neighborhood in the representation-space residual coordinates. This perspective is conceptually motivated by score matching and score-based modeling, where log-density gradients characterize first-order local variation (Hyvärinen & Dayan, 2005; Song & Ermon, 2019; Song & Kingma, 2021). Analogously, tracking the target-conditioned edit-space gradient across model states provides a signal of local-response change, leading to the following result.

Theorem 1 (Gradient drift bounds centered local-response drift). Let G_A and G_B be two model states obtained at different stages of sequential editing, and let $e_j = (x_j, y_j)$ be a historical edit. Let $h_{j,A}^L$ and $h_{j,B}^L$ denote the representations of e_j produced by the two model states, and let c_A and c_B denote their respective residual centers. Define the corresponding local response functions on a common displacement ϵ as

$$f_A(\epsilon) \triangleq \log \mathbb{P}_{G_A\left(h_{j,A}^L \mathrel{+}= c_A+\epsilon\right)}[y_j \mid x_j], \qquad f_B(\epsilon) \triangleq \log \mathbb{P}_{G_B\left(h_{j,B}^L \mathrel{+}= c_B+\epsilon\right)}[y_j \mid x_j]. \tag{7}$$

Their edit-space gradients at the respective reference centers are

$$g_A(e_j) \;\triangleq\; \nabla_\epsilon f_A(\epsilon)\big|_{\epsilon=0}, \qquad g_B(e_j) \;\triangleq\; \nabla_\epsilon f_B(\epsilon)\big|_{\epsilon=0}. \tag{8}$$

Assume that f_A and f_B are twice continuously differentiable on a convex neighborhood $\mathcal{U}$ of $\epsilon = 0$, and that the spectral norms of their Hessians are uniformly bounded by a constant $L > 0$ on $\mathcal{U}$. Then, for any $\epsilon \in \mathcal{U}$,

$$\Big|\big(f_A(\epsilon) - f_A(0)\big) - \big(f_B(\epsilon) - f_B(0)\big)\Big| \le \|\epsilon\|_2 \cdot \big\|g_A(e_j) - g_B(e_j)\big\|_2 + L\|\epsilon\|_2^2. \tag{9}$$

A complete proof is provided in Appendix B.3.

Implication for sequential editing. Theorem 1 relates the difference between two centered local responses to the gradient drift $\|g_A(e_j) - g_B(e_j)\|_2$ and a second-order curvature term $L\|\epsilon\|_2^2$. Centering each response at its state-specific reference point allows the comparison to incorporate changes in the model state, historical representation, and residual center.

The bound motivates gradient drift as a first-order signal of how the local response associated with a historical edit evolves during sequential editing. Because the gradient is taken with respect to the target log-likelihood, it connects edit-space directions to the functional response of the historical target.

5 Our method

Building on Theorem 1, which relates gradient drift to differences between centered local perturbation responses around state-specific reference centers, we next instantiate this signal for sequential editing. Specifically, we evaluate the edit-space gradient in Eq. 6 at two different model states and residual centers: an anchor state recorded when a historical edit is inserted, and a query state evaluated under the current model.

5.1 Gradient signals for sequential editing

In sequential editing, let the i-th edit correspond to $e_i = (x_i, y_i)$, and denote the set of historical edits by $\mathcal{M}_{i-1} = \{e_j\}_{j=1}^{i-1}$. For each historical edit e_j, we use two gradient signals: (i) a **gradient anchor**, computed once when e_j is inserted, and (ii) a **query gradient**, recomputed under the current model state when e_j is selected at a later editing step.

Definition 1 (Gradient Anchor). For a historical edit e_j, let $h^L_{j,j-1}$ denote its target-layer representation produced by the model state G_{j-1} when the edit is originally inserted. After obtaining the optimized editing residual δ_j from Eq. 2, we define the gradient anchor as

$$g^{\text{anc}}(e_j) = \nabla_\epsilon \log \mathbb{P}_{G_{j-1}\left(h^L_{j,j-1} \mathrel{+}= \delta_j + \epsilon\right)}\big[y_j \mid x_j\big]\Big|_{\epsilon=0}. \tag{10}$$

The anchor is evaluated around the insertion-time reference center δ_j and records the target-conditioned first-order response of e_j when the edit is constructed. It is computed once after δ_j is obtained and retained as a fixed reference for subsequent editing steps.

Definition 2 (Query Gradient). At the i-th editing step, let $h^L_{j,i-1}$ denote the target-layer representation of the historical edit e_j recomputed under the current model state G_{i-1}. We define its query gradient as

$$g^{\text{query}}(e_j) = \nabla_\epsilon \log \mathbb{P}_{G_{i-1}\left(h^L_{j,i-1} \mathrel{+}= \epsilon\right)}\big[y_j \mid x_j\big]\Big|_{\epsilon=0}. \tag{11}$$

The query gradient measures the current first-order sensitivity of the historical target at the unperturbed residual center. It is recomputed only when e_j is selected at the current editing step, as described in Section 5.2. Together, the anchor and query gradients instantiate the state-specific quantities in Theorem 1, and their discrepancy measures the corresponding first-order response drift.

Choice of response signal. Prior work on continual learning and sequential editing suggests two natural signals for preserving historical behavior: changes in internal representations and Fisher-based estimates of coordinate importance (Kirkpatrick et al., 2017; Fang et al., 2025). Representation drift measures changes in hidden representations, while Fisher-style weighting uses gradient

magnitude at a single model state to identify important coordinates. Since our objective is to track how the target-conditioned response of a historical edit evolves across model states, we use gradient drift as the weighting signal in GDR. Controlled comparisons with representation-drift and Fisher-style variants are reported in Appendix D.3.

5.2 TOP-K SELECTION VIA KEY-SPACE SIMILARITY

As the number of edits grows, computing $g^{\text{query}}(e_j)$ for all historical edits becomes increasingly expensive. Therefore, at the i-th editing step, we select only a small subset of historical edits that are most likely to be affected by the current edit. We use the FFN key representation defined in Eq. 3 as a lightweight proxy for interference. Since the edited down-projection matrix is accessed through the key k, the effect of a current update on a historical edit can be estimated by how strongly the historical key responds to the update direction induced by the current key. The detailed derivation is provided in Appendix B.2.

Importance score. For the current edit e_i and a historical edit $e_j \in \mathcal{M}_{i-1}$, we define the importance score as

$$s(e_i, e_j) = \left| k_i^\top A_i^{-1} k_j \right|, \qquad A_i^{-1} = \left(K_0 K_0^\top + k_i k_i^\top \right)^{-1}. \tag{12}$$

Here, K_0 denotes the background-key matrix used in the closed-form parameter update. For each historical edit e_j, its target-layer key k_j is computed when the edit is inserted and reused in subsequent selection steps. The score estimates the potential interaction between the current edit and a historical edit under the covariance-preconditioned key geometry. A larger value indicates that the current update has greater potential to affect e_j.

Top-k selection. The top-k module involves two decisions: how many historical edits to evaluate at each step and which historical edits to select. We set the selection budget to $k = 20$ based on the selection-size analysis in Appendix D.2. With this budget fixed, we compare uniform random sampling, recency-based selection, cosine key similarity, and our covariance-preconditioned key score. The controlled comparison in Appendix D.2 shows that the preconditioned score provides the strongest overall editing and historical-retention performance. Specifically, we select the k historical edits with the highest preconditioned scores:

$$\mathcal{B}_i \;=\; \mathrm{TopK}(\mathcal{M}_{i-1},\, k;\, s), \tag{13}$$

where s is defined in Eq. 12. When $|\mathcal{M}_{i-1}| \le k$, we use the complete history and set $\mathcal{B}_i = \mathcal{M}_{i-1}$. Query gradients are recomputed only for the selected $\min(k, |\mathcal{M}_{i-1}|)$ historical edits. The preconditioned selection rule scores all historical keys but restricts the more expensive query-gradient computation to the selected subset. The corresponding costs are analyzed in Appendix D.6.

5.3 GRADIENT DRIFT REGULARIZATION

Gradient Drift Regularization (GDR) converts the gradient drift of selected historical edits into dimension-wise weights for regularizing the current editing residual.

At the i-th editing step, given the selected historical subset $\mathcal{B}_i$ from Section 5.2, we compute the element-wise discrepancy between the query gradient and the stored anchor for each $e_j \in \mathcal{B}_i$:

$$d_j \triangleq \left| g^{\text{query}}(e_j) - g^{\text{anc}}(e_j) \right|. \tag{14}$$

We use this discrepancy to construct the GDR loss:

$$\mathcal{L}_{\text{GDR}}^{(i)}(\delta) = \sum_{e_j \in \mathcal{B}_i} \delta^\top W_j \delta, \qquad W_j \triangleq \mathrm{diag}(d_j^2). \tag{15}$$

Larger gradient discrepancies therefore impose stronger penalties on the corresponding editing coordinates. The quantities d_j and W_j are computed before optimizing the current residual and remain fixed throughout the optimization.

Combining GDR with the editing objective in Eq. 2, the total loss at the i-th step is

$$\mathcal{L}^{(i)}(\delta) = \mathcal{L}_{\text{edit}}^{(i)}(\delta) + \beta \mathcal{L}_{\text{GDR}}^{(i)}(\delta), \tag{16}$$

where $\beta > 0$ balances the current edit objective and the preservation of historical responses. The optimal residual δ_i is obtained by minimizing $\mathcal{L}^{(i)}$, after which the model is updated using the closed-form parameter update described in Appendix B.1. The overall procedure is summarized in Algorithm 1.

6 EXPERIMENTS

In this section, we conduct an empirical evaluation of the proposed method and aim to answer the following questions:

- **RQ1:** How does our method perform on knowledge editing benchmarks under the sequential editing setting?
- **RQ2:** Compared to existing baselines, can our method effectively control gradient drift to mitigate forgetting of historical edits?
- **RQ3:** To what extent can sequential editing preserve the model's reasoning, generation, and language-understanding capabilities?

6.1 EXPERIMENTAL SETUP

Base LLMs and baseline methods. We conduct experiments on three models: LLama-3.1-8B-Instruct (8B), Qwen-2.5-7B-Instruct (7B), and GPT2-XL (1.5B). We compare our method against a range of model editing baselines, including FT-L(Zhu et al., 2020), MEND(Mitchell et al., 2021), as well as several locate-then-edit approaches and their variants, such as MEMIT(Meng et al., 2023b), ROME(Meng et al., 2023a), PRUNE(Ma et al., 2024), AlphaEdit(Fang et al., 2024), and HSE(Fang et al., 2025). For baselines and implementation details for each model, see Appendix C.2 and C.3.

Datasets and evaluation metrics. We evaluate GDR on two widely used model-editing benchmarks, CounterFact (Meng et al., 2023b) and ZsRE (Levy et al., 2017). Following prior work (Meng et al., 2023b), we measure **Efficacy (rewrite success)**, **Generalization (paraphrase success)**, **Specificity (neighborhood success)**, **Fluency (generation entropy)**, and **Consistency (reference score)**. To assess whether sequential editing preserves capabilities beyond factual editing, we additionally evaluate the post-edited models on three complementary downstream benchmarks. GSM8K (Cobbe et al., 2021) evaluates mathematical reasoning using the solve rate of final answers. SAMSum (Gliwa et al., 2019) evaluates abstractive summarization. RTE (Bentivogli et al., 2009) evaluates natural-language inference. We further evaluate compositional reasoning and long-form generation on MQuAKE (Zhong et al., 2023) and LEME (Rosati et al., 2024), respectively. Detailed dataset descriptions and metric definitions are provided in Appendix C.1, while the complete MQuAKE and LEME results are reported in Appendix D.5.

6.2 PERFORMANCE ON SEQUENTIAL EDITING (RQ1)

To evaluate performance under sequential model editing, we conduct experiments on LLaMA-3.1-8B-Instruct, Qwen-2.5-7B-Instruct, and GPT-2-XL. For each model, 1,000 samples are edited sequentially with one sample per editing step. Table 1 reports the results on the two instruction-tuned models, while the complete GPT-2-XL results are provided in Appendix D.1.

Overall, GDR achieves strong performance across models and evaluation metrics. In particular, it improves editing efficacy (Eff.) and generalization (Gen.), showing that the newly inserted knowledge can be recalled from both the original prompts and their paraphrases. Meanwhile, GDR maintains competitive specificity (Spe.), fluency (Flu.), and consistency (Consis.), resulting in a favorable balance between incorporating new knowledge and preserving broader model behavior.The same trend is observed on GPT-2-XL. The improvements remain consistent across the three model families, indicating that the proposed drift-dependent regularization is not tied to a particular architecture.Computational and memory costs are analyzed in Appendix D.6.

Table 1: Comparison of GDR with existing methods after 1,000 sequential edits on LLaMA-3.1-8B-Instruct and Qwen-2.5-7B-Instruct. *Eff.*, *Gen.*, *Spe.*, *Flu.*, and *Consis.* denote Efficacy, Generalization, Specificity, Fluency, and Consistency, respectively. Each entry reports the sample mean ± standard error of the mean across evaluation instances. The best mean is highlighted in bold and the second-best is underlined.

Method	Model	CounterFact					ZsRE		
		Eff.↑	Gen.↑	Spe.↑	Flu.↑	Consis.↑	Eff.↑	Gen.↑	Spe.↑
Pre-edited		$7.85_{\pm 0.32}$	$10.58_{\pm 0.35}$	$89.48_{\pm 0.29}$	$635.23_{\pm 0.74}$	$24.14_{\pm 0.31}$	$36.99_{\pm 0.38}$	$36.34_{\pm 0.37}$	$31.89_{\pm 0.34}$
FT-L	LLaMA-3.1-8B-Instruct	$83.55_{\pm 0.54}$	$72.00_{\pm 0.61}$	$38.03_{\pm 0.68}$	$410.91_{\pm 1.21}$	$35.68_{\pm 0.42}$	$18.96_{\pm 0.57}$	$10.90_{\pm 0.49}$	$2.58_{\pm 0.31}$
MEND		$51.29_{\pm 0.73}$	$50.81_{\pm 0.69}$	$48.31_{\pm 0.71}$	$401.05_{\pm 1.46}$	$0.81_{\pm 0.17}$	$0.72_{\pm 0.06}$	$0.84_{\pm 0.07}$	$0.39_{\pm 0.04}$
MEMIT		$50.90_{\pm 0.76}$	$50.62_{\pm 0.72}$	$49.11_{\pm 0.69}$	$443.66_{\pm 1.53}$	$4.72_{\pm 0.24}$	$5.08_{\pm 0.19}$	$4.93_{\pm 0.18}$	$4.04_{\pm 0.16}$
ROME		$68.60_{\pm 0.65}$	$51.45_{\pm 0.72}$	$46.12_{\pm 0.70}$	$523.58_{\pm 1.18}$	$1.78_{\pm 0.13}$	$5.40_{\pm 0.20}$	$5.30_{\pm 0.19}$	$1.45_{\pm 0.10}$
PRUNE		$50.80_{\pm 0.74}$	$50.95_{\pm 0.68}$	$49.07_{\pm 0.66}$	$443.81_{\pm 1.49}$	$4.96_{\pm 0.25}$	$5.31_{\pm 0.21}$	$5.02_{\pm 0.20}$	$4.10_{\pm 0.17}$
HSE		$92.30_{\pm 0.42}$	$85.80_{\pm 0.51}$	**68.75**$_{\pm 0.48}$	$618.27_{\pm 0.86}$	$32.25_{\pm 0.37}$	$91.02_{\pm 0.39}$	$83.08_{\pm 0.46}$	$29.16_{\pm 0.34}$
AlphaEdit		$93.60_{\pm 0.36}$	$78.15_{\pm 0.47}$	$62.03_{\pm 0.52}$	$613.54_{\pm 0.79}$	$35.12_{\pm 0.39}$	$92.26_{\pm 0.34}$	$84.17_{\pm 0.43}$	$31.42_{\pm 0.36}$
GDR(Ours)		**97.50**$_{\pm 0.28}$	**93.95**$_{\pm 0.34}$	$66.39_{\pm 0.49}$	**627.88**$_{\pm 0.76}$	**40.38**$_{\pm 0.31}$	**96.81**$_{\pm 0.27}$	**89.11**$_{\pm 0.38}$	**33.16**$_{\pm 0.35}$
Pre-edited		$8.22_{\pm 0.35}$	$13.56_{\pm 0.43}$	$83.11_{\pm 0.31}$	$627.81_{\pm 0.81}$	$26.74_{\pm 0.38}$	$34.32_{\pm 0.41}$	$29.79_{\pm 0.39}$	$27.42_{\pm 0.37}$
FT-L	Qwen-2.5-7B-Instruct	$79.70_{\pm 0.58}$	$60.05_{\pm 0.67}$	$44.63_{\pm 0.65}$	$419.24_{\pm 1.34}$	$2.41_{\pm 0.19}$	$14.76_{\pm 0.55}$	$13.99_{\pm 0.52}$	$4.84_{\pm 0.29}$
MEND		$48.62_{\pm 0.75}$	$51.71_{\pm 0.72}$	$49.35_{\pm 0.70}$	$417.12_{\pm 1.42}$	$1.32_{\pm 0.16}$	$0.63_{\pm 0.05}$	$0.76_{\pm 0.06}$	$0.34_{\pm 0.04}$
MEMIT		$46.30_{\pm 0.78}$	$47.05_{\pm 0.75}$	$53.09_{\pm 0.73}$	$319.15_{\pm 1.61}$	$0.08_{\pm 0.06}$	$33.89_{\pm 0.46}$	$31.91_{\pm 0.43}$	$2.37_{\pm 0.18}$
ROME		$74.00_{\pm 0.61}$	$65.50_{\pm 0.68}$	$50.79_{\pm 0.72}$	$441.68_{\pm 1.25}$	$7.54_{\pm 0.22}$	$27.77_{\pm 0.49}$	$28.21_{\pm 0.48}$	$4.58_{\pm 0.24}$
PRUNE		$50.60_{\pm 0.74}$	$51.05_{\pm 0.69}$	$49.08_{\pm 0.68}$	$514.47_{\pm 1.19}$	$4.33_{\pm 0.26}$	$37.77_{\pm 0.53}$	$28.47_{\pm 0.49}$	$4.73_{\pm 0.25}$
HSE		$69.20_{\pm 0.66}$	$43.50_{\pm 0.72}$	$74.79_{\pm 0.54}$	$603.12_{\pm 0.93}$	$26.30_{\pm 0.41}$	$73.32_{\pm 0.48}$	$65.30_{\pm 0.51}$	$33.85_{\pm 0.37}$
AlphaEdit		$96.31_{\pm 0.31}$	$93.12_{\pm 0.38}$	$71.51_{\pm 0.49}$	$613.12_{\pm 0.82}$	$35.61_{\pm 0.36}$	$87.68_{\pm 0.43}$	$82.41_{\pm 0.47}$	$42.54_{\pm 0.32}$
GDR(Ours)		**99.80**$_{\pm 0.14}$	**96.95**$_{\pm 0.27}$	**77.88**$_{\pm 0.42}$	**625.74**$_{\pm 0.77}$	**39.06**$_{\pm 0.33}$	**96.19**$_{\pm 0.29}$	**93.26**$_{\pm 0.35}$	**44.38**$_{\pm 0.30}$

6.3 Gradient drift and historical retention (RQ2)

To evaluate historical edit retention and its empirical relationship with gradient drift, we conduct single-sample sequential editing experiments on the ZsRE dataset. After the T-th edit, we select the earliest 20% of the edited samples and measure their prediction accuracy, which we denote as *historical efficacy*. On the same subset, we compute the gradient drift magnitude by taking the ℓ_2 norm of d_j in Eq. 14 for each historical edit and averaging the resulting values over the subset. Since $d_j = |g^{\text{query}}(e_j) - g^{\text{anc}}(e_j)|$, its norm $\|d_j\|_2$ corresponds to the gradient discrepancy in Theorem 1. The results are shown in Fig. 2.

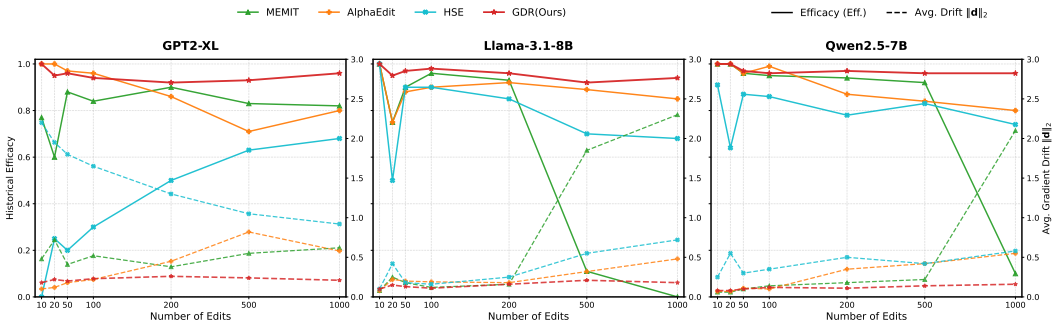

Figure 2: Historical efficacy and gradient drift during sequential editing. Solid lines show historical efficacy, defined as the prediction accuracy on the earliest 20% of edited samples after editing step T. Dashed lines show the average gradient drift magnitude, computed by averaging the per-sample ℓ_2 norm $\|d_j\|_2$ over the same historical subset. Here, d_j is the element-wise discrepancy between the current query gradient and the insertion-time gradient anchor.

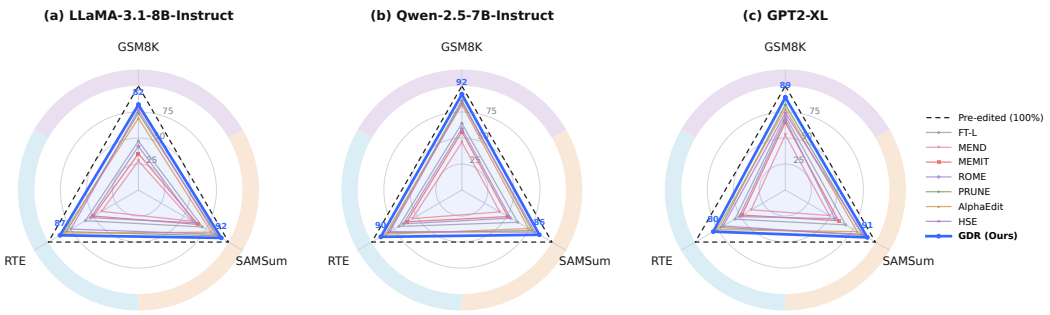

Figure 3: **General-capability retention after 1,000 sequential edits.** Panels (a), (b), and (c) report results on LLaMA-3.1-8B-Instruct, Qwen-2.5-7B-Instruct, and GPT2-XL, respectively. The three axes correspond to GSM8K, SAMSum, and RTE. Each curve represents an editing method, and the dashed reference triangle denotes 100% of the corresponding pre-edited performance. A profile closer to the reference triangle indicates better preservation of general capabilities. The radar plots report normalized retention; raw task-specific scores and evaluation details are provided in the appendix.

As the number of edits increases, GDR consistently achieves the highest historical efficacy among the compared methods, indicating better empirical retention of early inserted knowledge. AlphaEdit and HSE exhibit more pronounced degradation over long editing sequences, particularly near $T = 1000$. We also observe a consistent inverse association between gradient drift and historical efficacy: larger average drift tends to coincide with lower historical efficacy, whereas GDR maintains both lower drift and higher retention.

This empirical pattern complements Theorem 1. The theorem characterizes changes in centered local responses through gradient drift, while historical efficacy measures whether the corresponding edited predictions remain available over a long editing sequence. Together, the theoretical relation and retention results support the use of gradient drift as a target-conditioned signal for sequential editing.For clarity, Fig. 2 includes only the main representative methods. Full comparisons with all baselines and additional analyses of gradient drift are provided in Appendix D.7.

6.4 Preservation of General Capabilities After Sequential Editing (RQ3)

To assess general-capability retention beyond factual editing, we evaluate the models after 1,000 sequential edits on GSM8K, SAMSum, and RTE, using solve rate, average ROUGE-1/2/L F_1, and classification accuracy, respectively. Since these metrics have different scales, we report normalized retention as $R_{m,q} = 100 \times S^{(1000)}_{m,q} / S_{\mathrm{pre},q}$, where $S^{(1000)}_{m,q}$ and $S_{\mathrm{pre},q}$ are the post-edited and pre-edited scores. Thus, 100% denotes complete retention of the original performance.

As shown in Fig. 3, GDR maintains the strongest and most consistent capability retention across the three model families, with its profiles remaining closest to the 100% reference on reasoning, summarization, and natural-language inference. These results indicate that GDR improves historical-edit retention without substantially degrading the evaluated downstream capabilities. Raw scores and evaluation details are provided in Appendix D.4.

7 Conclusion and Limitations

We introduced Gradient Drift Regularization for stable sequential model editing. GDR measures changes between insertion-time and current edit-space gradients and uses them to construct a dimension-wise penalty on subsequent updates. Combined with preconditioned top-k selection, GDR improves historical retention while maintaining editing and downstream performance across multiple model families. The current implementation still requires storing historical keys and gradient anchors and scoring the growing key memory; reducing these costs is an important direction for future work.

AI USE STATEMENT

In this work, we used generative AI tools, including OpenAI Codex, to assist with literature discovery, language editing, LaTeX formatting, figure prototyping, code drafting and debugging, and feedback on the presentation of the theoretical analysis and experimental design. All AI-assisted suggestions were reviewed by the authors. Citations were checked against their original sources, mathematical derivations and proofs were independently verified, and AI-assisted code was inspected and tested before use. All empirical results reported in the paper were obtained from experiments executed and validated by the authors. AI-generated mock values and preliminary visualizations were used only for internal layout prototyping and are not reported as empirical evidence. Generative AI did not make final methodological decisions or determine which results were reported. The authors take full responsibility for the content, claims, code, and artifacts included in this submission.

ETHICS STATEMENT

This work uses publicly available pretrained models and established model-editing benchmarks and does not involve human subjects, private data, or new collection of personally identifiable information. Improving the stability of sequential editing may help maintain previously corrected knowledge while incorporating later updates, making language models easier to maintain and audit. However, model-editing techniques can also be misused to insert false, biased, misleading, or harmful information into deployed models. Moreover, successful retention of an edit does not establish that the edited information is truthful, fair, or safe. Practical deployment should therefore include restricted edit permissions, provenance and change logging, independent validation of proposed edits, monitoring of unintended behavioral changes, and mechanisms for rollback. Applications in high-stakes domains should additionally require review by qualified domain experts. Our experiments evaluate technical editing reliability and should not be interpreted as providing a general guarantee of model safety.

REPRODUCIBILITY STATEMENT

The definitions of the gradient anchor, query gradient, covariance-preconditioned top-k selection, and GDR objective are provided in Sections 5.1–5.3. Appendix A provides end-to-end pseudocode for the sequential editing procedure, while Appendix B.1 details the representation optimization and closed-form parameter update on which GDR is implemented. The assumptions and complete proof of Theorem 1 are given in Appendix B.3. Appendix C.1 specifies the datasets, evaluation splits, and metric definitions, and Appendix C.2 describes the compared methods. Appendix C.3 reports the computing hardware, edited layers, optimization steps, learning rates, regularization coefficients, background-covariance construction, and the definition of the reported instance-level standard errors. Additional controlled results are provided for the top-k budget and selection strategy in Appendix D.2, the response signals and regularization variants in Appendix D.3, the raw general-capability scores in Appendix D.4, and the computational and memory costs in Appendix D.6. All base models and evaluation benchmarks used in this work are publicly available and cited in the paper.

REFERENCES

Luisa Bentivogli, Peter Clark, Ido Dagan, and Danilo Giampiccolo. The fifth pascal recognizing textual entailment challenge. *TAC*, 7(8):1, 2009.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Boxi Cao, Hongyu Lin, Xianpei Han, and Le Sun. The life cycle of knowledge in big language models: A survey. *Machine Intelligence Research*, 21(2):217–238, 2024.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems, 2021. URL https://arxiv.org/abs/2110.14168.

Damai Dai, Li Dong, Yaru Hao, Zhifang Sui, Baobao Chang, and Furu Wei. Knowledge neurons in pretrained transformers. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8493–8502, 2022.

Rick Durrett. *Probability: theory and examples*, volume 49. Cambridge university press, 2019.

Junfeng Fang, Houcheng Jiang, Kun Wang, Yunshan Ma, Shi Jie, Xiang Wang, Xiangnan He, and Tat-Seng Chua. Alphaedit: Null-space constrained knowledge editing for language models. *arXiv preprint arXiv:2410.02355*, 2024.

Quntian Fang, Zhen Huang, Zhiliang Tian, Minghao Hu, Dongsheng Li, Yiping Yao, Xinyue Fang, Menglong Lu, and Guotong Geng. Hippocampal-like sequential editing for continual knowledge updates in large language models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer feed-forward layers are key-value memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 5484–5495, 2021.

Mor Geva, Avi Caciularu, Kevin Wang, and Yoav Goldberg. Transformer feed-forward layers build predictions by promoting concepts in the vocabulary space. In *Proceedings of the 2022 conference on empirical methods in natural language processing*, pp. 30–45, 2022.

Bogdan Gliwa, Iwona Mochol, Maciej Biesek, and Aleksander Wawer. Samsum corpus: A human-annotated dialogue dataset for abstractive summarization. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pp. 70–79. Association for Computational Linguistics, 2019. doi: 10.18653/v1/d19-5409. URL http://dx.doi.org/10.18653/v1/D19-5409.

Jia-Chen Gu, Hao-Xiang Xu, Jun-Yu Ma, Pan Lu, Zhen-Hua Ling, Kai-Wei Chang, and Nanyun Peng. Model editing harms general abilities of large language models: Regularization to the rescue. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 16801–16819, 2024.

Akshat Gupta, Anurag Rao, and Gopala Anumanchipalli. Model editing at scale leads to gradual and catastrophic forgetting. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 15202–15232, 2024.

Tom Hartvigsen, Swami Sankaranarayanan, Hamid Palangi, Yoon Kim, and Marzyeh Ghassemi. Aging with grace: Lifelong model editing with discrete key-value adaptors. *Advances in Neural Information Processing Systems*, 36:47934–47959, 2023.

Zeyu Huang, Yikang Shen, Xiaofeng Zhang, Jie Zhou, Wenge Rong, and Zhang Xiong. Transformer-patcher: One mistake worth one neuron. *arXiv preprint arXiv:2301.09785*, 2023.

Aapo Hyvärinen and Peter Dayan. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6(4), 2005.

Houcheng Jiang, Junfeng Fang, Ningyu Zhang, Guojun Ma, Mingyang Wan, Xiang Wang, Xiangnan He, and Tat-seng Chua. Anyedit: Edit any knowledge encoded in language models. *arXiv preprint arXiv:2502.05628*, 2025.

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

Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. Zero-shot relation extraction via reading comprehension. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pp. 333–342, 2017.

Zherui Li, Houcheng Jiang, Hao Chen, Baolong Bi, Zhenhong Zhou, Fei Sun, Junfeng Fang, and Xiang Wang. Reinforced lifelong editing for language models. *arXiv preprint arXiv:2502.05759*, 2025.

Zhoubo Li, Ningyu Zhang, Yunzhi Yao, Mengru Wang, Xi Chen, and Huajun Chen. Unveiling the pitfalls of knowledge editing for large language models. *arXiv preprint arXiv:2310.02129*, 2023.

Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pp. 74–81, Barcelona, Spain, July 2004. Association for Computational Linguistics. URL https://aclanthology.org/W04-1013/.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.

Jun-Yu Ma, Hong Wang, Hao-Xiang Xu, Zhen-Hua Ling, and Jia-Chen Gu. Perturbation-restrained sequential model editing. *arXiv preprint arXiv:2405.16821*, 2024.

Vittorio Mazzia, Alessandro Pedrani, Andrea Caciolai, Kay Rottmann, and Davide Bernardi. A survey on knowledge editing of neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 36(7):11759–11775, 2024.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in gpt, 2023a. URL https://arxiv.org/abs/2202.05262.

Kevin Meng, Arnab Sen Sharma, Alex Andonian, Yonatan Belinkov, and David Bau. Mass-editing memory in a transformer, 2023b. URL https://arxiv.org/abs/2210.07229.

Eric Mitchell, Charles Lin, Antoine Bosselut, Chelsea Finn, and Christopher D Manning. Fast model editing at scale. *arXiv preprint arXiv:2110.11309*, 2021.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. Language models as knowledge bases? In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pp. 2463–2473, 2019.

Adam Roberts, Colin Raffel, and Noam Shazeer. How much knowledge can you pack into the parameters of a language model? In *Proceedings of the 2020 conference on empirical methods in natural language processing (EMNLP)*, pp. 5418–5426, 2020.

Domenic Rosati, Robie Gonzales, Jinkun Chen, Xuemin Yu, Yahya Kayani, Frank Rudzicz, and Hassan Sajjad. Long-form evaluation of model editing. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 3749–3780, 2024.

Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32, 2019.

Yang Song and Diederik P Kingma. How to train your energy-based models. *arXiv preprint arXiv:2101.03288*, 2021.

Gilbert Strang. *Introduction to linear algebra*. SIAM, 2022.

Martin J Wainwright. *High-dimensional statistics: A non-asymptotic viewpoint*, volume 48. Cambridge university press, 2019.

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

Peng Wang, Zexi Li, Ningyu Zhang, Ziwen Xu, Yunzhi Yao, Yong Jiang, Pengjun Xie, Fei Huang, and Huajun Chen. Wise: Rethinking the knowledge memory for lifelong model editing of large language models. *Advances in Neural Information Processing Systems*, 37:53764–53797, 2024a.

Song Wang, Yaochen Zhu, Haochen Liu, Zaiyi Zheng, Chen Chen, and Jundong Li. Knowledge editing for large language models: A survey. *ACM Computing Surveys*, 57(3):1–37, 2024b.

Wanli Yang, Fei Sun, Xinyu Ma, Xun Liu, Dawei Yin, and Xueqi Cheng. The butterfly effect of model editing: Few edits can trigger large language models collapse. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 5419–5437, 2024a.

Wanli Yang, Fei Sun, Jiajun Tan, Xinyu Ma, Du Su, Dawei Yin, and Huawei Shen. The fall of rome: Understanding the collapse of llms in model editing. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 4079–4087, 2024b.

Yunzhi Yao, Peng Wang, Bozhong Tian, Siyuan Cheng, Zhoubo Li, Shumin Deng, Huajun Chen, and Ningyu Zhang. Editing large language models: Problems, methods, and opportunities. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 10222–10240, 2023.

Lang Yu, Qin Chen, Jie Zhou, and Liang He. Melo: Enhancing model editing with neuron-indexed dynamic lora. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 19449–19457, 2024.

Ningyu Zhang, Yunzhi Yao, Bozhong Tian, Peng Wang, Shumin Deng, Mengru Wang, Zekun Xi, Shengyu Mao, Jintian Zhang, Yuansheng Ni, et al. A comprehensive study of knowledge editing for large language models. *arXiv preprint arXiv:2401.01286*, 2024.

Yue Zhang, Yafu Li, Leyang Cui, Deng Cai, Lemao Liu, Tingchen Fu, Xinting Huang, Enbo Zhao, Yu Zhang, Yulong Chen, et al. Siren's song in the ai ocean: A survey on hallucination in large language models. *Computational Linguistics*, 51(4):1373–1418, 2025.

Ce Zheng, Lei Li, Qingxiu Dong, Yuxuan Fan, Zhiyong Wu, Jingjing Xu, and Baobao Chang. Can we edit factual knowledge by in-context learning? In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 4862–4876, 2023.

Zexuan Zhong, Zhengxuan Wu, Christopher D Manning, Christopher Potts, and Danqi Chen. Mquake: Assessing knowledge editing in language models via multi-hop questions. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 15686–15702, 2023.

Chen Zhu, Ankit Singh Rawat, Manzil Zaheer, Srinadh Bhojanapalli, Daliang Li, Felix Yu, and Sanjiv Kumar. Modifying memories in transformer models. *arXiv preprint arXiv:2012.00363*, 2020.

TECHNICAL APPENDICES AND SUPPLEMENTARY MATERIAL

In the Appendix, we provide additional methodological details, experimental setups, complete results, proofs, and discussions of societal impacts, as outlined below:

- Appendix A: Pseudocode for Gradient Drift Regularization.
- Appendix B: Implementation details of current model editing and related proofs.
- Appendix C: Experimental setups.
- Appendix D: More experimental results.

A PSEUDOCODE FOR GRADIENT DRIFT REGULARIZATION

Algorithm 1 summarizes the overall procedure of GDR in the single-sample sequential editing setting. We use $k = 20$ for the historical subset size. When the number of historical edits is no larger than k, all historical edits are used for regularization; otherwise, $\mathcal{B}_i$ is selected by the preconditioned key-space score in Section 5.2.

Algorithm 1 Sequential editing with Gradient Drift Regularization (GDR)

Require: Initial model G_0; raw edit sequence $\widetilde{E} = \{\widetilde{e}_i = (s_i, r_i, o_i, o_i^*)\}_{i=1}^{T}$; prompt template $p(\cdot)$; prefix distribution $\mathcal{U}_{\text{prefix}}$; edited layers $\mathcal{R}$; background keys K_0; hyperparameters β and k
Ensure: Edited model G_T

Initialize historical record bank $\mathcal{H}_0 \leftarrow \emptyset$
for $i = 1$ to T **do**
  Construct $x_i = p(s_i, r_i)$ and $y_i = o_i^*$; set $e_i = (x_i, y_i)$
  Sample $\{x_{\text{pref}}^n\}_{n=1}^{P} \sim \mathcal{U}_{\text{prefix}}$
  Compute the target-layer representation h_i^L and selection key k_i under G_{i-1} using Eq. 3
  Let $\mathcal{M}_{i-1} \leftarrow \{e_j : (e_j, k_j, g^{\text{anc}}(e_j)) \in \mathcal{H}_{i-1}\}$
  if $|\mathcal{M}_{i-1}| \le k$ **then**
    $\mathcal{B}_i \leftarrow \mathcal{M}_{i-1}$
  else
    Compute $A_i^{-1} \leftarrow (K_0 K_0^\top + k_i k_i^\top)^{-1}$
    for each record $(e_j, k_j, g^{\text{anc}}(e_j)) \in \mathcal{H}_{i-1}$ **do**
      Compute $s(e_i, e_j) \leftarrow |k_i^\top A_i^{-1} k_j|$
    end for
    $\mathcal{B}_i \leftarrow \text{TopK}(\mathcal{M}_{i-1}, k; s)$
  end if
  for each selected edit $e_j = (x_j, y_j) \in \mathcal{B}_i$ **do**
    Recompute $h_{j,i-1}^L$ under the current model G_{i-1}
    Compute $g^{\text{query}}(e_j)$ using Eq. 11
    Retrieve $g^{\text{anc}}(e_j)$ from the corresponding record in $\mathcal{H}_{i-1}$
    Set $d_j \leftarrow |g^{\text{query}}(e_j) - g^{\text{anc}}(e_j)|$ and $W_j \leftarrow \text{diag}(d_j^2)$
  end for
  Obtain $\delta_i \leftarrow \arg\min_\delta \mathcal{L}^{(i)}(\delta)$ using Eq. 16
  Compute the insertion-time anchor $g^{\text{anc}}(e_i)$ at $(G_{i-1}, h_i^L, \delta_i)$ using Eq. 10
  Set $z_i \leftarrow h_i^L + \delta_i$
  for each edited layer $\ell \in \mathcal{R}$ **do**
    Compute the layer-specific key k_i^ℓ
    Compute the closed-form update Δ_i^ℓ as described in Appendix B.1
    $W_{\text{out}}^\ell \leftarrow W_{\text{out}}^\ell + \Delta_i^\ell$
  end for
  Obtain G_i after applying all layer updates
  Append the new record: $\mathcal{H}_i \leftarrow \mathcal{H}_{i-1} \cup \{(e_i, k_i, g^{\text{anc}}(e_i))\}$
end for
return G_T

B IMPLEMENTATION DETAILS OF CURRENT MODEL EDITING AND RELATED PROOFS

B.1 IMPLEMENTATION DETAILS OF LOCATE-AND-EDIT METHODS

In this section, we provide the detailed procedure for converting the representation-space target in Section 3 into localized parameter updates. We mainly follow the locate-and-edit frameworkWang et al. (2024b); Yang et al. (2024a).

Given the i-th edit $e_i = (s_i, r_i, o_i, o_i^*)$, we denote its prompt and target completion as (x_i, y_i), where $x_i = p(s_i, r_i)$ and $y_i = o_i^*$Yang et al. (2024b); Li et al. (2025). The model before applying the i-th edit is denoted as G_{i-1}, and the edited model after incorporating this edit is denoted as G_i. MEMIT first constructs a target representation z_i in the hidden space and then converts this representation-level target into parameter updates on selected FFN layers.

Locating editing layers. Locate-and-edit methods are based on the observation that factual associations are largely stored in the feed-forward network (FFN) modules of Transformer layers. Following prior workMeng et al. (2023a;b), a set of editing layers is selected and denoted as

$$\mathcal{R} = \{\ell_1, \ldots, \ell_R\}. \tag{17}$$

We use $L = \max(\mathcal{R})$ to denote the target layer where the edited knowledge is expected to be fully expressed. For the i-th edit, let h_i^L be the hidden representation at layer L and the last subject token under the pre-edit model G_{i-1}.

Constructing the target representation. As introduced in Section 3, MEMIT first optimizes a representation-space perturbation δ_i such that injecting it into h_i^L makes the model predict the new target y_iMeng et al. (2023a). The editing loss is

$$\mathcal{L}_{\text{edit}}^{(i)}(\delta) = \frac{1}{P} \sum_{n=1}^{P} -\log \mathbb{P}_{G_{i-1}(h_i^L \mathrel{+}= \delta)} \left[y_i \mid x_{\text{pref}}^n \oplus x_i \right], \tag{18}$$

where $G_{i-1}(h_i^L \mathrel{+}= \delta)$ denotes the forward execution of G_{i-1} with δ injected into the hidden state h_i^L. The optimized perturbation is denoted as

$$\delta_i = \arg\min_{\delta} \mathcal{L}_{\text{edit}}^{(i)}(\delta), \tag{19}$$

and the target representation is defined as

$$z_i = h_i^L + \delta_i. \tag{20}$$

Layer-wise residual allocation. MEMIT distributes the target change from h_i^L to z_i across the selected editing layers $\mathcal{R}$. In the following derivation, we focus on an arbitrary edited layer $\ell \in \mathcal{R}$ and omit the layer superscript for notational simplicity. For this layer, the residual that needs to be written is usually defined as

$$r_i = \frac{z_i - h_i^L}{L - \ell + 1}, \tag{21}$$

where h_i^L is computed under the current partially edited model during the MEMIT update procedure. This residual allocation spreads the desired representation change over multiple layers, rather than forcing a single layer to absorb the entire edit.

Key-value representation at an editing layer. For the selected layer ℓ, the FFN module can be interpreted as a key-value memory. Let $h_i^{\ell-1}$ denote the hidden representation before layer ℓ for the edit prompt x_i. The key corresponding to the i-th edit is computed as

$$k_i = \frac{1}{P} \sum_{n=1}^{P} \sigma \left(W_{\text{in}} \gamma \left(h_{i,n}^{\ell-1} \right) \right), \tag{22}$$

where $h_{i,n}^{\ell-1}$ is the layer-$(\ell-1)$ hidden representation under the n-th prompt variant $x_{\text{pref}}^n \oplus x_i$, $\gamma(\cdot)$ denotes layer normalization, and $\sigma(\cdot)$ is the FFN activation function. The corresponding value produced by the FFN down-projection is

$$v_i = W_{\text{out}} k_i. \tag{23}$$

To realize the desired residual r_i, the target value at this layer is defined as

$$v_i^* = v_i + r_i. \tag{24}$$

Closed-form parameter update. Given a set of new edit-induced key-value pairs at the current edited layer, MEMIT updates the FFN down-projection matrix W_{out}. Let (K_0, V_0) denote background key-value pairs sampled from a Wikipedia corpus to approximate pre-existing associations in the pre-trained model, and let (K_1, V_1^*) denote the new key-value pairs induced by the requested edits:

$$K_0 = [k_1, \ldots, k_n], \qquad V_0 = [v_1, \ldots, v_n], \tag{25}$$

$$K_1 = [k_{n+1}, \ldots, k_{n+u}], \qquad V_1^* = [v_{n+1}^*, \ldots, v_{n+u}^*], \tag{26}$$

where n is the number of background associations and u is the number of requested edits written at this step. In the single-sample sequential setting used in this paper, we typically have $u = 1$.

The updated matrix $\widetilde{W}_{\text{out}}$ is obtained by solving the following regularized least-squares problem:

$$\widetilde{W}_{\text{out}} = \arg\min_{\widehat{W}} \left(\left\| \widehat{W} K_0 - V_0 \right\|_F^2 + \left\| \widehat{W} K_1 - V_1^* \right\|_F^2 \right). \tag{27}$$

Since the original matrix W_{out} already satisfies the background associations approximately, the update can be expressed as

$$\widetilde{W}_{\text{out}} = W_{\text{out}} + \Delta. \tag{28}$$

Applying the normal equationStrang (2022) gives the MEMIT closed-form update:

$$\Delta = (V_1^* - W_{\text{out}} K_1) \, K_1^\top \left(K_0 K_0^\top + K_1 K_1^\top \right)^{-1}. \tag{29}$$

Therefore, the updated FFN down-projection becomes

$$\widetilde{W}_{\text{out}} = W_{\text{out}} + (V_1^* - W_{\text{out}} K_1) \, K_1^\top \left(K_0 K_0^\top + K_1 K_1^\top \right)^{-1}. \tag{30}$$

The term $K_0 K_0^\top$ acts as a covariance-based regularizer that discourages updates along directions frequently activated by background knowledge, while $K_1 K_1^\top$ encodes the directions associated with the requested edits. After applying Eq. 30 to each selected layer $\ell \in \mathcal{R}$, the resulting model becomes G_i.

B.2 Derivation of the causes of the key-space interference

We provide a simple derivation to show how a newly written edit can affect the output of another input through the FFN value projection. For simplicity, we focus on the target layer L and omit the layer superscript. We consider the single-sample sequential editing setting, where one edit is written at each step.

Let W denote the FFN value projection matrix before applying an edit. Suppose an edit with key k and target value v^* is written into the model. Following the regularized least-squares update in Appendix B.1, the updated matrix can be written as

$$W' = W + \Delta, \tag{31}$$

where, in the single-edit case,

$$\Delta = (v^* - Wk) \, k^\top \left(K_0 K_0^\top + k k^\top \right)^{-1}. \tag{32}$$

Here, K_0 denotes the background keys used to regularize the update, and $v^* - Wk$ is the value residual that the edit attempts to write.

Now consider a new input whose FFN key is k^*. After the edit has been applied, the value produced by the updated projection matrix is

$$W'k^* = (W + \Delta)\, k^* = Wk^* + \Delta k^*. \tag{33}$$

Substituting Eq. 32 into Eq. 33, we obtain

$$W'k^* = Wk^* + (v^* - Wk)\, k^\top \left(K_0 K_0^\top + kk^\top\right)^{-1} k^*. \tag{34}$$

The first term Wk^* is the original value produced for the new input before the edit. The second term

$$\underbrace{(v^* - Wk)\, k^\top \left(K_0 K_0^\top + kk^\top\right)^{-1} k^*}_{\text{interference induced by the edit}} \tag{35}$$

is the additional value shift caused by the previous edit. Therefore, this term characterizes how the edit written for key k affects another input with key k^*.

Let

$$\alpha(k, k^*) = k^\top \left(K_0 K_0^\top + kk^\top\right)^{-1} k^*. \tag{36}$$

Then Eq. 35 can be rewritten as

$$\Delta k^* = \alpha(k, k^*)\, (v^* - Wk)\,. \tag{37}$$

This shows that the interference on k^* is controlled by two factors: the value residual $v^* - Wk$, which determines what is written into the model, and the scalar coefficient $\alpha(k, k^*)$, which measures how strongly the new input key k^* overlaps with the edited key direction after covariance preconditioning.

Therefore, the edit produces no additional first-order value shift on k^* when

$$\alpha(k, k^*) = k^\top \left(K_0 K_0^\top + kk^\top\right)^{-1} k^* = 0. \tag{38}$$

In this case,

$$\Delta k^* = 0, \qquad W'k^* = Wk^*, \tag{39}$$

which means that the edit does not change the value retrieved by the new input at this layer.

This analysis provides a basis for evaluating the degree of interference between two edits, which we use in the subsequent section for selecting the most relevant historical edits in our Top-k selection mechanism. We define an importance score based on the preconditioned interaction between the current edit's key k_i and the historical edit's key k_j, as shown in Eq. 12. This allows us to prioritize edits that are most likely to be affected by the current update.

B.3 Proof of Theorem 1: Gradient drift bounds centered local-response drift

We restate the theorem using the two state-specific evaluation centers.

Setup. Let G_A and G_B be two model states, and let $e_j = (x_j, y_j)$ be a historical edit. Let $h_{j,A}^L$ and $h_{j,B}^L$ denote the representations of e_j produced by the two model states, and let c_A and c_B be their respective residual centers. Define

$$f_A(\epsilon) \triangleq \log \mathbb{P}_{G_A\left(h_{j,A}^L \mathrel{+}= c_A + \epsilon\right)}[y_j \mid x_j], \qquad f_B(\epsilon) \triangleq \log \mathbb{P}_{G_B\left(h_{j,B}^L \mathrel{+}= c_B + \epsilon\right)}[y_j \mid x_j]. \tag{40}$$

Their gradients at the respective reference centers are

$$g_A(e_j) = \nabla_\epsilon f_A(\epsilon)\big|_{\epsilon=0}, \qquad g_B(e_j) = \nabla_\epsilon f_B(\epsilon)\big|_{\epsilon=0}. \tag{41}$$

Assume that f_A and f_B are twice continuously differentiable on a convex neighborhood $\mathcal{U}$ of $\epsilon = 0$, and that their Hessians satisfy

$$\left\|\nabla_\epsilon^2 f_A(\epsilon)\right\|_2 \le L, \qquad \left\|\nabla_\epsilon^2 f_B(\epsilon)\right\|_2 \le L, \qquad \forall \epsilon \in \mathcal{U}, \tag{42}$$

where $\|\cdot\|_2$ denotes the spectral norm for matrices.

We want to show that, for any $\epsilon \in \mathcal{U}$,

$$\begin{aligned} &\Big|\big(f_A(\epsilon) - f_A(0)\big) - \big(f_B(\epsilon) - f_B(0)\big)\Big| \\ &\qquad \le \|\epsilon\|_2 \cdot \left\|g_A(e_j) - g_B(e_j)\right\|_2 + L\|\epsilon\|_2^2. \end{aligned} \tag{43}$$

Step 1: Taylor expansion around each reference center. Fix $\epsilon \in \mathcal{U}$. Since f_A is twice continuously differentiable on $\mathcal{U}$, Taylor's theorem Wainwright (2019) with the integral form of the remainder gives

$$f_A(\epsilon) = f_A(0) + \langle g_A(e_j), \epsilon \rangle + R_A(\epsilon), \tag{44}$$

where

$$R_A(\epsilon) = \int_0^1 (1-t)\epsilon^\top \nabla_\epsilon^2 f_A(t\epsilon)\epsilon \, dt. \tag{45}$$

Using the spectral-norm bound in Eq. 42, we obtain

$$\begin{aligned} |R_A(\epsilon)| &\le \int_0^1 (1-t)\big\|\nabla_\epsilon^2 f_A(t\epsilon)\big\|_2 \|\epsilon\|_2^2 \, dt \\ &\le L\|\epsilon\|_2^2 \int_0^1 (1-t)\, dt = \frac{L}{2}\|\epsilon\|_2^2. \end{aligned} \tag{46}$$

By the same argument applied to f_B,

$$f_B(\epsilon) = f_B(0) + \langle g_B(e_j), \epsilon \rangle + R_B(\epsilon), \qquad |R_B(\epsilon)| \le \frac{L}{2}\|\epsilon\|_2^2. \tag{47}$$

Step 2: Subtract the centered expansions. Subtracting the baseline value from each expansion and then subtracting Eq. 47 from Eq. 44 gives

$$\begin{aligned} &\big(f_A(\epsilon) - f_A(0)\big) - \big(f_B(\epsilon) - f_B(0)\big) \\ &\quad = \langle g_A(e_j) - g_B(e_j), \epsilon \rangle + R_A(\epsilon) - R_B(\epsilon). \end{aligned} \tag{48}$$

Taking absolute values and applying the triangle inequality,

$$\begin{aligned} &\Big|\big(f_A(\epsilon) - f_A(0)\big) - \big(f_B(\epsilon) - f_B(0)\big)\Big| \\ &\quad \le |\langle g_A(e_j) - g_B(e_j), \epsilon \rangle| + |R_A(\epsilon)| + |R_B(\epsilon)|. \end{aligned} \tag{49}$$

Step 3: Bound the first-order term. By the Cauchy–Schwarz inequality Durrett (2019),

$$|\langle g_A(e_j) - g_B(e_j), \epsilon \rangle| \le \big\|g_A(e_j) - g_B(e_j)\big\|_2 \cdot \|\epsilon\|_2. \tag{50}$$

Step 4: Combine the bounds. Substituting Eq. 46, the corresponding bound on $|R_B(\epsilon)|$ in Eq. 47, and Eq. 50 into Eq. 49 yields

$$\begin{aligned} &\Big|\big(f_A(\epsilon) - f_A(0)\big) - \big(f_B(\epsilon) - f_B(0)\big)\Big| \\ &\quad \le \|\epsilon\|_2 \cdot \big\|g_A(e_j) - g_B(e_j)\big\|_2 + \frac{L}{2}\|\epsilon\|_2^2 + \frac{L}{2}\|\epsilon\|_2^2. \end{aligned} \tag{51}$$

Collecting the two remainder terms gives

$$\begin{aligned} &\Big|\big(f_A(\epsilon) - f_A(0)\big) - \big(f_B(\epsilon) - f_B(0)\big)\Big| \\ &\quad \le \|\epsilon\|_2 \cdot \big\|g_A(e_j) - g_B(e_j)\big\|_2 + L\|\epsilon\|_2^2, \end{aligned} \tag{52}$$

which is exactly Eq. 43 and is identical to the main-text bound in Eq. 9.

Remark. The bound compares changes relative to the two state-specific evaluation centers. Consequently, the baseline likelihood difference between the two states does not appear in the bound. The first-order term depends on the gradient drift $\|g_A(e_j) - g_B(e_j)\|_2$, while the remaining error is second order in $\|\epsilon\|_2$. This result characterizes differences between centered local perturbation responses; it does not bound the absolute prediction difference $|f_A(0) - f_B(0)|$ or by itself guarantee the retention of a historical edit.

C EXPERIMENTAL SETUPS

C.1 DATASETS AND EVALUATION METRICS

We evaluate our method on standard benchmarks for model editing, covering factual editing, reasoning, and general language understanding.

CounterFactMeng et al. (2023a) is a widely used benchmark for factual model editing, formulated as a cloze-style task. Each sample corresponds to a counterfactual triple (s_i, r_i, o_i^*) that contradicts real-world knowledge, making it challenging for pre-trained models. For each edit, the dataset provides paraphrased prompts to evaluate generalization and neighborhood prompts constructed by replacing the subject to assess specificity.

ZsRELevy et al. (2017) is a question-answering dataset adapted for model editing. Each sample consists of a subject and a target answer, along with paraphrased questions generated via back-translation for evaluating generalization. The dataset also includes unrelated natural questions to measure specificity.

MQuAKEZhong et al. (2023) evaluates multi-hop reasoning after editing. It focuses on whether factual updates propagate correctly through chains of reasoning, testing the consistency of entailed knowledge beyond single-hop recall.

LEMERosati et al. (2024) extends evaluation to long-form generation, highlighting challenges such as factual drift, internal consistency, and lexical coherence. It reveals that short-form editing metrics may not correlate with long-form generation quality.

GSM8K (Cobbe et al., 2021) evaluates mathematical reasoning using grade-school word problems. We evaluate on the official test split, which contains 1,319 examples. Performance is measured by the solve rate, i.e., the percentage of examples for which the extracted final numerical answer matches the reference answer.

SAMSum (Gliwa et al., 2019) evaluates abstractive dialogue summarization. We use its official test split containing 819 dialogues and measure performance using the average of ROUGE-1, ROUGE-2, and ROUGE-L F_1 scores (Lin, 2004):

$$S_{\mathrm{SAMSum}} = \frac{1}{3}\left(\mathrm{R1}_{F_1} + \mathrm{R2}_{F_1} + \mathrm{RL}_{F_1}\right). \tag{53}$$

RTE (Bentivogli et al., 2009) evaluates natural-language inference as a binary classification problem. Because the labels of the official test split are not publicly available, we evaluate on the 277-example validation split distributed with GLUE (Wang et al., 2018). Performance is measured using classification accuracy.

CounterFact Metrics. Since CounterFact involves counterfactual targets, evaluation compares the probabilities of the edited target o_i^* and the original fact o_i.

- **Efficacy (Eff.)**: proportion of edits where the model assigns higher probability to the target o_i^* than the original o_i:

$$\mathbb{E}_i\big[P_{f_i}(o_i^* \mid (s_i, r_i)) > P_{f_i}(o_i \mid (s_i, r_i))\big]. \tag{54}$$

- **Generalization (Gen.)**: proportion of paraphrased prompts where the model prefers o_i^*:

$$\mathbb{E}_i[\mathbb{E}_{p\in\text{para}(s_i,r_i)}[P_{f_i}(o_i^* \mid p) > P_{f_i}(o_i \mid p)]]. \tag{55}$$

- **Specificity (Spe.)**: proportion of neighborhood prompts where the model preserves the original fact o_i:

$$\mathbb{E}_i[\mathbb{E}_{p\in\text{neigh}(s_i,r_i)}[P_{f_i}(o_i \mid p) > P_{f_i}(o_i^* \mid p)]]. \tag{56}$$

- **Fluency (Flu.)**: measures repetition using n-gram entropy:

$$-\frac{2}{3}\sum_k g_2(k)\log g_2(k) - \frac{4}{3}\sum_k g_3(k)\log g_3(k). \tag{57}$$

- **Consistency (Consis.)**: cosine similarity between TF-IDF vectors of generated text and reference Wikipedia text:

$$\cos\big(\text{TF-IDF}(f_i(s_i)), \text{TF-IDF}(\text{Wiki}(o_i^*))\big). \tag{58}$$

ZsRE Metrics. ZsRE evaluates top-1 accuracy of generated answers:

- **Efficacy (Eff.)**: accuracy on original prompts:

$$\mathbb{E}_i\big[o_i^* = \arg\max_o P_{f_i}(o \mid (s_i, r_i))\big]. \tag{59}$$

- **Generalization (Gen.)**: accuracy on paraphrased prompts:

$$\mathbb{E}_i[\mathbb{E}_{p\in\text{para}(s_i,r_i)}[o_i^* = \arg\max_o P_{f_i}(o \mid p)]]. \tag{60}$$

- **Specificity (Spe.)**: accuracy on neighborhood prompts:

$$\mathbb{E}_i[\mathbb{E}_{p\in\text{neigh}(s_i,r_i)}[o_i^* = \arg\max_o P_{f_i}(o \mid p)]]. \tag{61}$$

C.2 BASELINES

We consider a set of representative methods for model editing, including parameter-modifying approaches FT-LZhu et al. (2020), MENDMitchell et al. (2021), ROMEMeng et al. (2023a), MEMITMeng et al. (2023b), PRUNEMa et al. (2024), HSEFang et al. (2025), and AlphaEditFang et al. (2024). Since our proposed method also belongs to the parameter-modifying paradigm, we focus our comparison on this category. Detailed implementation settings follow the official configurations provided in their respective papers.

- **FT-L** performs standard fine-tuning on the identified editing layers, typically with weight decay to limit excessive parameter updates. It directly optimizes the model parameters using the editing objective, making it a simple but strong baseline. However, because it lacks an explicit mechanism for preserving unrelated or previously edited knowledge, it is prone to interference in sequential editing.
- **MEND** learns a gradient-based editing network that transforms raw gradients into parameter updates, enabling efficient and scalable model editing. Instead of applying gradients directly to the model, MEND predicts low-rank parameter modifications conditioned on the edit example. This allows fast test-time editing, but its performance can be sensitive to the distribution of edits encountered during training.
- **ROME** adopts a locate-then-edit strategy, identifying critical layers responsible for factual recall and applying rank-one updates to modify specific knowledge. It interprets feed-forward modules as key-value memories and edits factual associations by updating the value projection at selected layers. ROME is effective for single factual edits, but its rank-one update design can struggle when many edits are applied sequentially.
- **MEMIT** extends ROME to support batch editing by distributing updates across multiple layers, allowing simultaneous insertion of multiple knowledge facts. It solves a least-squares-style update that maps a set of keys to desired values while considering background covariance statistics. Although MEMIT improves scalability over ROME, repeated sequential applications can still accumulate interference and degrade earlier edits.
- **PRUNE** constrains parameter updates by controlling the condition number of the update matrix Δ, thereby stabilizing edits and reducing interference. The method aims to avoid overly sharp or ill-conditioned perturbations that may harm model behavior. By regularizing the update structure, PRUNE improves stability compared with naive fine-tuning, but it does not explicitly preserve the gradient behavior of historical edits.
- **HSE** introduces a replay-based mechanism combined with Fisher regularization to preserve previously edited knowledge during sequential updates. It draws inspiration from hippocampal memory consolidation, where replayed historical samples are used to reduce forgetting. While effective in some long-sequence settings, its stability can depend heavily on the quality and frequency of replayed samples.

- **AlphaEdit** projects the update matrix onto a null space that preserves existing knowledge, ensuring that new edits minimally interfere with prior edits. It constrains updates to directions that are less likely to affect previously stored knowledge representations. This projection-based strategy improves locality and stability, but its effectiveness may weaken when the accumulated editing trajectory becomes long and complex.

C.3 Implementation details

We conduct all experiments on a single A800 80GB GPU. Each model sequentially edits 1,000 data instances in approximately 7 hours. For the sampling of (K_0, V_0) in Appendix B.1, we use a corpus of 100,000 samples extracted from Wikipedia. For the hyperparameter settings and implementation details of baseline methods, we follow the configurations used in MEMITMeng et al. (2023b) and AlphaEditFang et al. (2024). The implementation details of our proposed method are summarized as follows.

- **LLaMA-3.1-8B-Instruct.** We apply editing to layers $[4, 5, 6, 7, 8]$. The update norm of δ is constrained to 0.75 times the norm of the original output representation to ensure controlled modifications. The iterative optimization of δ is capped at a maximum of 25 steps, with a learning rate of 1×10^{-1}. To balance the retention of previous knowledge and the incorporation of new information, we set the balance coefficient β in Eq. 16 to 7×10^{-5} for both two datasets. In addition, the coefficient λ_C for C_0 is set to 15000.
- **Qwen-2.5-7B-Instruct.** We apply editing to layers $[13, 14, 15, 16, 17]$. The update norm of δ is constrained to 0.75 times the norm of the original output representation. For dataset-specific hyperparameters, we use different configurations for ZsRE and CounterFact due to their different editing formats and optimization difficulty. On ZsRE, we optimize δ for at most 15 steps with a learning rate of 5×10^{-1}, and set the balance coefficient β in Eq. 16 to 3.5×10^{-4}. On CounterFact, we optimize δ for at most 25 steps with a learning rate of 1×10^{-1}, and set the balance coefficient β in Eq. 16 to 5×10^{-5}. In addition, the coefficient λ_C for C_0 is set to 20000.
- **GPT2-XL.** We apply editing to layers $[4, 5, 6, 7, 8]$. The update norm of δ is constrained to 0.75 times the norm of the original output representation. The iterative optimization of δ is capped at a maximum of 15 steps, with a learning rate of 5×10^{-1}. To balance the retention of previous knowledge and the incorporation of new information, we set the balance coefficient β in Eq. 16 to 2.5×10^{-4} for ZsRE and 5×10^{-4} for CounterFact. In addition, the coefficient λ_C for C_0 is set to 20000.

Statistical reporting. For each method, model, and dataset, we compute the corresponding metric at the evaluation-instance level and report the sample mean together with its standard error. For a metric with per-instance values $\{z_n\}_{n=1}^{N}$, the standard error is computed as $\mathrm{SE} = s_z/\sqrt{N}$, where s_z is the sample standard deviation. For binary success metrics, this is equivalent to the binomial standard error $\sqrt{\hat{p}(1-\hat{p})/N}$.

D More experimental results

D.1 Sequential editing results on GPT-2-XL

To complement the results on the instruction-tuned models in Table 1, we report the complete GPT-2-XL results under the same 1,000-edit setting. As shown in Table 2, GDR achieves the best result on seven of the eight metrics and remains competitive in CounterFact specificity, showing that its improvements also extend to a different model architecture.

D.2 Analysis of top-k selection

Selection-size analysis. We conduct a hyperparameter analysis to determine the number of historical edits k used in the top-k selection module. The goal of this study is to examine the trade-off between sequential editing performance and computational cost. We evaluate different values of k on the CounterFact dataset under the sequential editing setting, where the model is edited

Table 2: Comparison of sequential editing methods on GPT-2-XL after 1,000 edits. Each entry reports the sample mean ± standard error of the mean across evaluation instances. The best mean is highlighted in bold and the second-best is underlined.

Method	CounterFact					ZsRE		
	Eff.↑	Gen.↑	Spe.↑	Flu.↑	Consis.↑	Eff.↑	Gen.↑	Spe.↑
Pre-edited	$22.23_{\pm 0.62}$	$24.34_{\pm 0.68}$	$78.53_{\pm 0.44}$	$626.64_{\pm 0.92}$	$31.88_{\pm 0.47}$	$22.19_{\pm 0.51}$	$31.30_{\pm 0.46}$	$24.15_{\pm 0.39}$
FT-L	$71.80_{\pm 0.68}$	$49.22_{\pm 0.75}$	$47.44_{\pm 0.72}$	$283.01_{\pm 1.54}$	$1.83_{\pm 0.14}$	$10.54_{\pm 0.43}$	$7.70_{\pm 0.39}$	$0.27_{\pm 0.12}$
MEND	$49.10_{\pm 0.77}$	$49.48_{\pm 0.74}$	$49.59_{\pm 0.75}$	$502.68_{\pm 1.32}$	$29.17_{\pm 0.46}$	$0.31_{\pm 0.09}$	$0.38_{\pm 0.10}$	$1.80_{\pm 0.08}$
MEMIT	$\underline{95.50_{\pm 0.39}}$	$85.05_{\pm 0.52}$	$63.60_{\pm 0.61}$	$486.40_{\pm 1.16}$	$21.95_{\pm 0.43}$	$87.54_{\pm 0.44}$	$71.83_{\pm 0.51}$	$\underline{25.55_{\pm 0.37}}$
ROME	$53.80_{\pm 0.76}$	$55.75_{\pm 0.73}$	$52.51_{\pm 0.70}$	$388.74_{\pm 1.48}$	$1.60_{\pm 0.12}$	$65.69_{\pm 0.56}$	$59.29_{\pm 0.53}$	$21.45_{\pm 0.35}$
PRUNE	$93.47_{\pm 0.43}$	$85.10_{\pm 0.51}$	$56.12_{\pm 0.65}$	$470.90_{\pm 1.22}$	$15.64_{\pm 0.36}$	$53.22_{\pm 0.58}$	$50.38_{\pm 0.55}$	$19.53_{\pm 0.39}$
HSE	$92.90_{\pm 0.46}$	$83.40_{\pm 0.55}$	$\mathbf{75.07}_{\pm 0.50}$	$\underline{596.44_{\pm 0.98}}$	$\underline{36.86_{\pm 0.40}}$	$88.20_{\pm 0.45}$	$74.82_{\pm 0.52}$	$24.48_{\pm 0.38}$
AlphaEdit	$95.15_{\pm 0.38}$	$\underline{91.55_{\pm 0.46}}$	$\underline{65.13_{\pm 0.57}}$	$501.65_{\pm 1.07}$	$25.13_{\pm 0.42}$	$\underline{91.18_{\pm 0.41}}$	$\underline{78.08_{\pm 0.48}}$	$24.85_{\pm 0.36}$
GDR (Ours)	$\mathbf{97.70}_{\pm 0.32}$	$\mathbf{93.35}_{\pm 0.39}$	$63.16_{\pm 0.61}$	$\mathbf{610.88}_{\pm 0.91}$	$\mathbf{39.38}_{\pm 0.35}$	$\mathbf{95.81}_{\pm 0.36}$	$\mathbf{88.92}_{\pm 0.43}$	$\mathbf{26.07}_{\pm 0.37}$

continuously over a stream of knowledge updates. Unless otherwise specified, we use LLaMA-3.1-8B-Instruct as the backbone model and keep all other hyperparameters fixed. We vary k in $\{0, 5, 10, 15, 20, 25, 30, 35\}$, where $k = 0$ removes the historical gradient-anchor constraint and serves as the efficiency-oriented baseline.

For each value of k, we report three metrics: (i) *Edit Score*, which measures the overall sequential editing quality; (ii) *Time / Edit*, which measures the average wall-clock time required for each edit; and (iii) *Relative Cost*, which is normalized by the cost of $k = 0$. The results are summarized in Tab. 3, and the corresponding performance–efficiency trade-off is visualized in Fig. 4.

Table 3: Hyperparameter analysis of the top-k selection size. A larger k introduces more historical constraints but also increases the editing cost. We choose $k = 20$ as it achieves a strong editing score while maintaining moderate computational overhead.

k	Edit Score ↑	Time / Edit (s) ↓	Relative Cost ↓
0	86.6	9.84	1.00×
5	87.8	12.92	1.31×
10	88.3	15.00	1.52×
15	87.7	18.10	1.84×
20	**90.4**	**20.18**	**2.05×**
25	91.6	25.31	2.57×
30	89.8	35.58	3.62×
35	86.9	46.35	4.71×

Results. As shown in Tab. 3, increasing k generally improves the editing score at the beginning, indicating that incorporating historical gradient anchors helps preserve previously edited knowledge. When k increases from 0 to 20, the edit score improves from 86.6 to 90.4, while the relative cost increases from $1.00\times$ to $2.05\times$. This suggests that a moderate number of historical constraints can effectively reduce interference during sequential editing.

Although $k = 25$ achieves the highest edit score of 91.6, it requires a noticeably higher cost of $2.57\times$. When k is further increased to 30 or 35, the computational cost grows rapidly, while the edit score decreases to 89.8 and 86.9, respectively. This shows that using too many historical constraints may introduce redundant or weakly relevant samples, which increases optimization difficulty without consistently improving editing quality. Therefore, we set $k = 20$ in the main experiments, as it provides a good balance between sequential editing performance and computational efficiency.

Selection-strategy ablation. The preceding analysis determines the number of historical edits evaluated at each editing step. We next fix the selection budget to $k = 20$ and compare four strategies

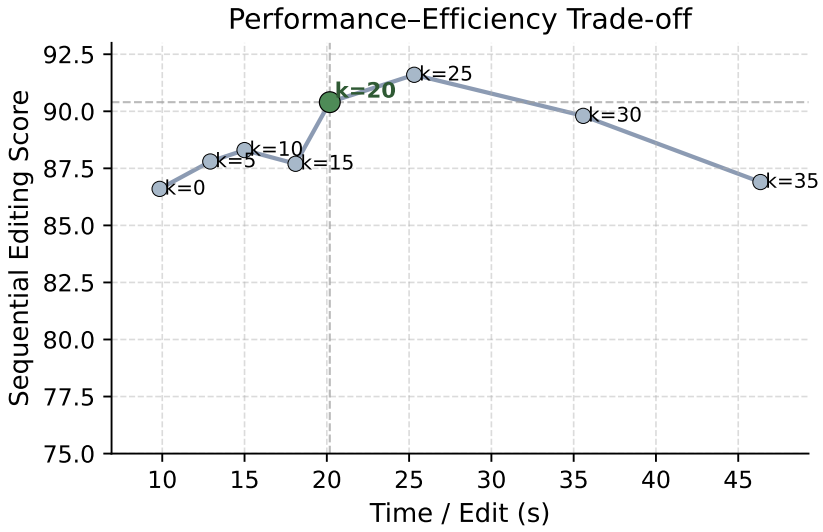

Figure 4: Performance–efficiency trade-off under different top-k values. The horizontal axis denotes the average time per edit, and the vertical axis denotes the sequential editing score. The dashed lines mark the result of $k = 20$, which provides a favorable balance between performance and computational efficiency.

Table 4: Ablation of historical-edit selection strategies on LLaMA-3.1-8B-Instruct after 1,000 sequential edits. The selection budget is fixed to $k = 20$, and all variants share the same editing and optimization settings. *Hist.* denotes historical efficacy on the earliest 20% of edited samples.

Selection strategy	CounterFact				ZsRE			
	Eff.↑	Gen.↑	Spe.↑	Hist.↑	Eff.↑	Gen.↑	Spe.↑	Hist.↑
Uniform random	82.64	75.31	55.82	59.47	74.28	65.73	22.06	51.94
Recency	86.91	80.42	58.37	66.83	80.75	72.48	24.61	60.27
Cosine key similarity	91.36	86.24	62.07	77.58	88.94	81.06	28.42	74.79
Preconditioned key score (Ours)	**97.50**	**93.95**	**66.39**	**89.62**	**96.81**	**89.11**	**33.16**	**88.37**

for determining which historical edits are included in the selected subset. All variants use the same model, edit sequence, GDR objective, query-gradient computation, and optimization settings; they differ only in the construction of $\mathcal{B}_i$.

Uniform random selection samples k historical edits without replacement. Recency-based selection chooses the k most recently inserted edits. Cosine selection ranks historical edits according to

$$s_{\cos}(e_i, e_j) = \frac{|k_i^\top k_j|}{\|k_i\|_2 \|k_j\|_2}, \tag{62}$$

whereas preconditioned selection uses the score in Eq. 12. When $|\mathcal{M}_{i-1}| \leq k$, all strategies use the complete history.

Results. Table 4 shows that random and recency-based selection provide weaker historical retention because they do not account for the interaction between the current edit and historical keys. Cosine similarity improves the selection quality, while the covariance-preconditioned score achieves the strongest overall balance between current-edit performance and historical retention. We therefore use the preconditioned score in the final GDR formulation.

D.3 Ablation of regularization strategies and GDR design choices

We conduct two complementary ablation studies. First, we compare GDR with standard regularization strategies over different editing lengths to evaluate long-horizon stability. Second, we perform

Table 5: Ablation study on regularization strategies for sequential editing. Results are reported on both ZsRE and CounterFact datasets. We evaluate Efficacy (Eff.), Generalization (Gen.), and Specificity (Spe.) across different editing lengths T. Our method (GDR) consistently achieves more stable performance, especially in long-horizon settings. The best results are highlighted in bold.

ZsRE

Method	$T=10$			$T=100$			$T=500$			$T=1000$			$T=1500$			$T=2000$		
	Eff.	Gen.	Spe.	Eff.	Gen.	Spe.	Eff.	Gen.	Spe.	Eff.	Gen.	Spe.	Eff.	Gen.	Spe.	Eff.	Gen.	Spe.
MEMIT	98.15	92.41	28.36	90.18	87.63	30.52	11.47	10.36	11.28	5.08	4.93	4.04	0.00	0.00	0.00	0.00	0.00	0.00
MEMIT + Fisher	98.36	86.27	39.18	95.42	**90.58**	36.47	14.63	15.72	18.55	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MEMIT + ℓ_2	90.58	70.36	38.47	77.28	72.41	**39.63**	21.52	17.38	17.41	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
GDR (Ours)	**99.27**	**93.39**	**39.36**	**97.26**	90.25	34.75	**96.98**	**88.16**	**32.58**	**96.81**	**89.11**	**33.16**	**95.32**	**85.23**	**30.25**	**93.25**	**82.36**	**25.63**

CounterFact

Method	$T=10$			$T=100$			$T=500$			$T=1000$			$T=1500$			$T=2000$		
	Eff.	Gen.	Spe.	Eff.	Gen.	Spe.	Eff.	Gen.	Spe.	Eff.	Gen.	Spe.	Eff.	Gen.	Spe.	Eff.	Gen.	Spe.
MEMIT	99.00	95.32	87.14	98.10	90.27	87.63	50.18	51.47	49.22	50.90	50.62	49.11	49.36	47.52	48.27	47.41	46.58	38.11
MEMIT + Fisher	98.66	93.28	80.19	94.34	85.67	87.21	52.43	53.18	48.36	54.29	53.74	46.58	51.63	46.72	41.38	47.55	40.63	37.24
MEMIT + ℓ_2	98.23	90.41	79.26	90.58	85.32	86.74	50.27	51.64	47.18	50.46	49.37	48.55	49.28	47.63	50.41	47.72	40.58	28.36
GDR (Ours)	**99.13**	**95.36**	**88.36**	**98.17**	**92.56**	**88.65**	**96.34**	**90.15**	**70.15**	**97.50**	**93.95**	**66.39**	**93.12**	**90.25**	**63.21**	**90.78**	**87.64**	**59.36**

a controlled ablation of the response signal and weighting function used by GDR. The latter keeps the editing framework and quadratic regularization form fixed while varying only the construction of the coordinate-wise weights.

Experimental setup. We conduct experiments on the ZsRE and CounterFact datasets using LLaMA-3.1-8B-Instruct. For the long-horizon comparison, we evaluate performance at editing lengths $T \in \{10, 100, 500, 1000, 1500, 2000\}$. For the controlled design ablation, we report results after $T = 1000$ sequential edits. All controlled variants use the same edit sequence, top-k selection budget, residual optimization procedure, and closed-form parameter update.

We report Efficacy (Eff.), Generalization (Gen.), and Specificity (Spe.). For the controlled design ablation, we additionally report historical efficacy (Hist.), defined as the prediction accuracy on the earliest 20% of edited samples after $T = 1000$.

Long-horizon comparison of regularization strategies. We compare four variants within a unified editing framework. **MEMIT** serves as the base method without additional constraints. We further consider two commonly used regularization strategies: **MEMIT + Fisher**, which incorporates Fisher-based constraintsFang et al. (2025), and **MEMIT +** ℓ_2, which applies standard parameter-level ℓ_2 regularization. Finally, we evaluate **GDR (Ours)**, which introduces our proposed gradient-based regularization mechanism. All methods use the same backbone model, edit sequence, and editing budget. The regularization hyperparameters are selected under the same validation protocol and then fixed for the test sequence.

Results. The results are summarized in Tab. 5. We observe that the vanilla MEMIT rapidly degrades as the number of edits increases, particularly in the generalization and specificity metrics. Adding ℓ_2 regularization provides limited improvement, indicating that naive parameter-level constraints are insufficient to stabilize sequential updates. The Fisher-based method improves early-stage performance but still suffers from noticeable degradation when T becomes large.

In contrast, our method consistently maintains strong performance across all editing lengths. Notably, at $T = 1000$ and beyond, our approach achieves significantly higher generalization and specificity scores, demonstrating superior robustness under long editing sequences. Even at $T = 2000$, where most baselines collapse, our method preserves stable behavior with only minor performance decay.

These results show that incorporating GDR into the MEMIT editing framework substantially improves long-horizon stability compared with unregularized, ℓ_2-regularized, and parameter-level

Table 6: Controlled ablation of response signals and weighting functions on LLaMA-3.1-8B-Instruct after 1,000 sequential edits. All variants share the same edit sequence and optimization settings, while the regularized variants use the same top-k selection and quadratic penalty form. *Hist.* denotes historical efficacy on the earliest 20% of edited samples.

Variant	CounterFact				ZsRE			
	Eff.↑	Gen.↑	Spe.↑	Hist.↑	Eff.↑	Gen.↑	Spe.↑	Hist.↑
No regularization	50.90	50.62	49.11	28.40	5.08	4.93	4.04	3.15
Representation drift	79.84	73.26	58.47	62.15	72.46	65.81	26.72	58.44
Anchor-gradient Fisher	68.72	64.18	54.63	49.36	56.38	49.27	21.84	42.16
Gradient drift (d_j)	94.38	90.52	63.84	84.77	93.72	86.54	31.48	83.05
Gradient drift (d_j^2, GDR)	**97.50**	**93.95**	**66.39**	**89.62**	**96.81**	**89.11**	**33.16**	**88.37**

Fisher-regularized alternatives. We next isolate the contributions of the response signal and weighting function through a controlled design ablation.

Controlled ablation of response signals and weighting functions. This experiment isolates two design choices in GDR: the signal used to characterize historically sensitive editing coordinates and the transformation used to convert this signal into regularization weights. All variants use the same selected historical subsets $\mathcal{B}_i$ and the same quadratic penalty form, differing only in the construction of W_j. Implementation details, including regularization-coefficient selection, are provided in Appendix C.3.

For the representation-drift variant, we define

$$r_j \triangleq \left| h_{j,i-1}^L - \left(h_{j,j-1}^L + \delta_j \right) \right|, \qquad W_j^{\mathrm{rep}} \triangleq \mathrm{diag}(r_j^2). \tag{63}$$

This variant constructs the coordinate weights from the geometric displacement between the insertion-time and current representation centers.

For the anchor-gradient Fisher variant, we use

$$W_j^{\mathrm{F}} \triangleq \mathrm{diag}\left(\left(g^{\mathrm{anc}}(e_j) \right)^2 \right). \tag{64}$$

This provides a diagonal single-sample empirical-Fisher proxy in the same editing coordinates as GDR.

For gradient drift, we compare linear and squared weighting:

$$W_j^{\mathrm{linear}} \triangleq \mathrm{diag}(d_j), \qquad W_j^{\mathrm{squared}} \triangleq \mathrm{diag}(d_j^2), \tag{65}$$

where d_j is defined in Eq. 14. The squared variant is the final GDR formulation. All vector squares are applied element-wise.

Results. As shown in Table 6, representation-drift and anchor-gradient Fisher weighting improve over the unregularized baseline but remain less effective than gradient-drift weighting, especially in generalization and historical efficacy. This indicates that changes in target-conditioned sensitivity provide a more informative signal than representation displacement or importance measured at a single reference state. Both gradient-drift variants perform strongly, with squared weighting achieving the best overall efficacy–retention trade-off. We therefore use $W_j = \mathrm{diag}(d_j^2)$ in the final GDR formulation.

D.4 Detailed results for general-capability retention

Table 7 reports the raw task-specific scores underlying the normalized retention profiles in Fig. 3. The three task columns retain their original metrics: solve rate for GSM8K, the average of ROUGE-1, ROUGE-2, and ROUGE-L F_1 scores for SAMSum, and classification accuracy for RTE.

Let $\mathcal{Q}$ denote the three general-capability tasks: GSM8K, SAMSum, and RTE. We compute the average normalized retention as

$$\overline{R}_m = \frac{1}{|\mathcal{Q}|} \sum_{q \in \mathcal{Q}} \frac{S_{m,q}^{(1000)}}{S_{\mathrm{pre},q}}, \tag{66}$$

Table 7: General-capability performance after 1,000 sequential edits. GSM8K is reported using solve rate, SAMSum using average ROUGE, and RTE using classification accuracy. All three raw task scores are expressed as percentages. *Avg. Ret.* denotes the arithmetic mean of the three task-wise normalized retention scores defined in Eq. 66.

Model	Method	GSM8K Solve rate	SAMSum Avg. ROUGE	RTE Accuracy	Avg. Ret. (%)
LLaMA-3.1-8B-Instruct	Pre-edited	84.20	32.70	78.30	100.00
	FT-L	39.57	23.22	46.20	59.00
	MEND	23.58	19.62	32.10	43.00
	MEMIT	29.47	21.58	39.15	50.33
	ROME	35.36	20.60	41.50	52.67
	PRUNE	62.31	28.78	62.64	80.67
	AlphaEdit	58.10	26.49	64.99	77.67
	HSE	64.83	27.80	58.73	79.00
	GDR (Ours)	**69.04**	**30.08**	**68.12**	**87.00**
Qwen-2.5-7B-Instruct	Pre-edited	91.00	33.50	81.20	100.00
	FT-L	58.24	20.77	56.84	65.33
	MEND	41.86	14.07	44.66	47.67
	MEMIT	50.96	17.09	49.53	56.00
	ROME	53.69	18.09	52.78	59.33
	PRUNE	78.26	26.13	66.58	82.00
	AlphaEdit	74.62	24.79	69.02	80.33
	HSE	76.44	27.14	64.96	81.67
	GDR (Ours)	**83.72**	**28.81**	**73.08**	**89.33**
GPT2-XL	Pre-edited	32.50	22.80	55.60	100.00
	FT-L	23.07	15.28	31.14	64.67
	MEND	17.23	11.17	21.13	46.67
	MEMIT	21.12	13.45	26.69	57.33
	ROME	22.10	12.54	28.36	58.00
	PRUNE	26.65	19.38	40.03	79.67
	AlphaEdit	25.35	18.24	42.26	78.00
	HSE	24.05	19.84	38.36	76.67
	GDR (Ours)	**28.93**	**20.75**	**44.48**	**86.67**

where $S_{m,q}^{(1000)}$ is the raw score of method m after 1,000 sequential edits and $S_{\mathrm{pre},q}$ is the corresponding score of the pre-edited model.

The results show that GDR achieves the highest raw score on all three downstream tasks across the three model families. Its advantage is also reflected in the average normalized retention, indicating more consistent preservation across reasoning, summarization, and natural-language inference. PRUNE, AlphaEdit, and HSE retain substantial portions of the pre-edited capabilities, but their relative performance varies more across tasks and model families.

D.5 Evaluation on expanding benchmark: LEME and MQuAKE

To further evaluate the effectiveness of our method in more challenging reasoning scenarios, we conduct additional experiments on two benchmarks: LongformEvaluation (LEME)Rosati et al. (2024) and MQuAKEZhong et al. (2023). These benchmarks are designed to probe the impact of model editing on compositional reasoning and long-form generation quality.Specifically, LEME focuses on long-form generative tasks and evaluates multiple aspects of output quality, including edit success, factual correctness, and internal consistency. In contrast, MQuAKE emphasizes multi-hop reasoning and measures whether the model can consistently propagate newly edited knowledge through chains of dependent facts.The results are summarized in Tab. 8.

Overall, our method demonstrates strong performance across both benchmarks. On MQuAKE, GDR achieves higher scores on multi-hop reasoning tasks, indicating that it better preserves the logical consistency of updated knowledge across multiple reasoning steps. On LEME, our method maintains competitive or superior performance across edit accuracy, factual correctness, and internal consistency, suggesting that it effectively balances knowledge updating with the preservation of global coherence.

Table 8: Performance of GDR on MQuAKE and LEME datasets for multi-Hop reasoning and long-form editing. Metrics include Bert-score, Rouge-L, Edit Consistency, Factual Consistency, and Internal Consistency. The best results are highlighted in bold.

Model	Method	MQuAKE		LEME		
		Bert-score↑	Rouge-L↑	Edit↑	Factual↑	Internal↑
LLaMA-3.1-8B-Instruct	MEMIT	$75.31_{\pm 0.11}$	$22.73_{\pm 0.18}$	$2.86_{\pm 0.20}$	$2.74_{\pm 0.18}$	$4.56_{\pm 0.32}$
	HSE	$\underline{78.42_{\pm 0.17}}$	$\underline{84.17_{\pm 0.24}}$	$3.73_{\pm 0.26}$	$4.02_{\pm 0.29}$	$\underline{6.35_{\pm 0.40}}$
	AlphaEdit	$69.85_{\pm 0.15}$	$23.04_{\pm 0.23}$	$\underline{3.92_{\pm 0.24}}$	$\underline{4.18_{\pm 0.27}}$	$6.21_{\pm 0.38}$
	GDR (Ours)	$\mathbf{98.65}_{\pm 0.13}$	$\mathbf{96.17}_{\pm 0.21}$	$\mathbf{4.35}_{\pm 0.22}$	$\mathbf{4.61}_{\pm 0.25}$	$\mathbf{6.88}_{\pm 0.35}$
GPT2-XL	MEMIT	$73.54_{\pm 0.10}$	$26.84_{\pm 0.13}$	$2.20_{\pm 0.23}$	$2.48_{\pm 0.21}$	$4.02_{\pm 0.35}$
	HSE	$74.56_{\pm 0.20}$	$\underline{69.36_{\pm 0.25}}$	$3.28_{\pm 0.31}$	$\underline{3.52_{\pm 0.30}}$	$5.74_{\pm 0.43}$
	AlphaEdit	$\underline{78.15_{\pm 0.22}}$	$20.78_{\pm 0.27}$	$\underline{3.43_{\pm 0.34}}$	$3.31_{\pm 0.32}$	$\underline{5.93_{\pm 0.48}}$
	GDR (Ours)	$\mathbf{94.79}_{\pm 0.18}$	$\mathbf{91.46}_{\pm 0.24}$	$\mathbf{3.78}_{\pm 0.32}$	$\mathbf{3.82}_{\pm 0.29}$	$\mathbf{6.41}_{\pm 0.41}$

These results highlight that GDR not only improves editing accuracy but also enhances the model's ability to reason over edited knowledge and generate consistent long-form outputs.

D.6 Computational and memory overhead

We quantify the additional computation and storage introduced by GDR. Figure 5(a) reports the average wall-clock time per edit for AlphaEdit, HSE, and GDR on three models. The additional computation of GDR mainly comes from historical-key scoring and query-gradient evaluation for the selected historical edits.

At editing step i, let $N_i = |\mathcal{M}_{i-1}|$ denote the number of historical edits. GDR evaluates the importance score in Eq. 12 for all N_i historical keys. After the preconditioned current-key vector has been formed, the exact key-scoring cost is linear in N_i. The top-k procedure then selects $\min(k, N_i)$ edits, and query gradients are evaluated only for this subset. For a fixed k, the number of historical query-gradient evaluations per editing step is therefore at most k, whereas exact key scoring continues to grow with N_i.

As specified in Algorithm 1, each historical record contains the edit $e_j = (x_j, y_j)$, its selection key k_j, and its gradient anchor $g^{\text{anc}}(e_j)$. Let d_k and d_g denote the dimensions of the selection key and gradient anchor, respectively. Excluding the variable-length input–target text, the tensor storage for N_i historical records is $\mathcal{O}(N_i(d_k + d_g))$. The historical-record storage therefore grows linearly with the editing sequence.

Figure 5(b) decomposes the measured per-edit runtime of GDR on LLaMA-3.1-8B. Here, base editing includes optimizing the residual for the current edit and applying the closed-form parameter update. Query-gradient evaluation includes recomputing the current representations and gradients for the selected historical edits. Key selection includes scoring the historical keys and selecting the top-k records. The query-gradient component plateaus after the history exceeds the top-k budget because at most $k = 20$ query gradients are evaluated at each step. In contrast, the key-selection component continues to grow with the number of historical records.

D.7 Gradient drift analysis

To further investigate model stability during sequential editing, we analyze gradient drift from both qualitative and quantitative perspectives.

Specifically, we study how gradients of historical samples deviate after long editing sequences, and how such deviations relate to model behavior across different evaluation metrics.Overall, these results provide direct empirical evidence that GDR successfully suppresses gradient drift and maintains local stability in the parameter space, which aligns with the improved retention performance observed in Section 6.3.

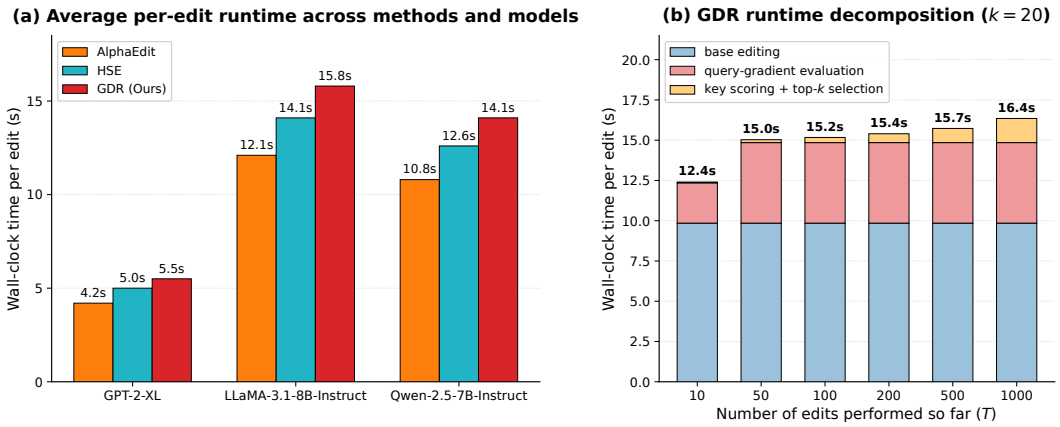

Figure 5: Computational overhead analysis. (a) Average wall-clock time per edit for AlphaEdit, HSE, and GDR across three models. (b) Per-edit runtime decomposition of GDR on LLaMA-3.1-8B at different editing-history lengths. Base editing includes current-edit optimization and the closed-form parameter update. With $k = 20$, the query-gradient component plateaus once the history exceeds the selection budget, whereas historical-key scoring and top-k selection continue to grow with the number of stored edits.

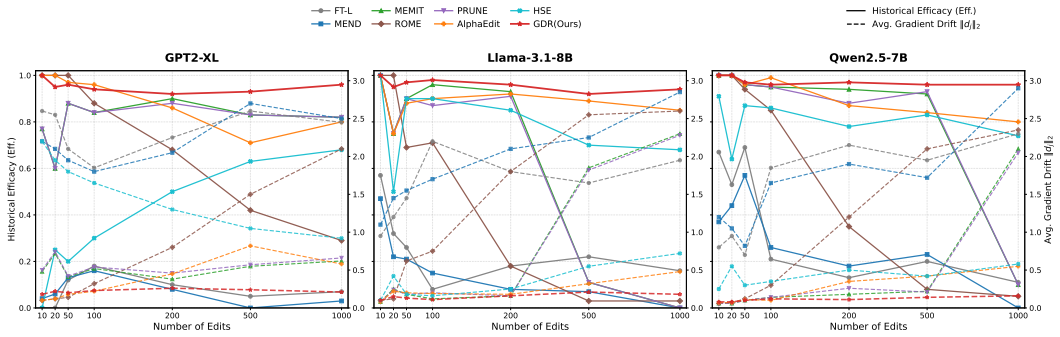

Figure 6: Full comparison of historical efficacy and gradient drift with all baselines. Solid lines denote historical efficacy (Eff.), measured on the earliest 20% of edited samples after each editing step T. Dashed lines denote the average gradient drift magnitude, measured by $\|d_j\|_2$ on the same subset. Larger gradient drift generally correlates with lower historical efficacy, while GDR consistently maintains both high historical efficacy and low drift across the three models.

D.7.1 Full results on historical efficacy and gradient drift

To avoid clutter in the main text, Fig. 2 only reports the main representative methods. Fig. 6 presents the corresponding full comparison with all baselines. The overall trend remains consistent: GDR preserves the highest historical efficacy and simultaneously maintains the smallest gradient drift across most checkpoints and models. Moreover, the full set of baselines further reveals a strong negative correlation between average gradient drift and historical efficacy, providing additional empirical support for our theoretical analysis.

D.7.2 Visualization of edit-space gradient drift

To further analyze model stability, we visualize the edit-space gradient drift of historical samples after sequential editing. We conduct experiments on the Qwen-2.5-7B-Instruct model using the ZsRE dataset under the standard sequential editing setting with $T = 1000$ edits.

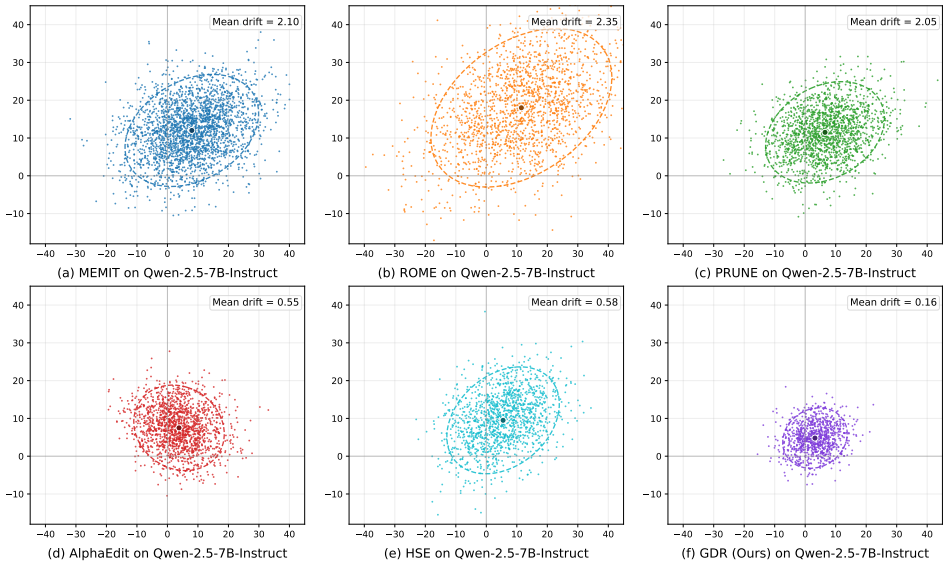

Figure 7: Visualization of edit-space gradient drift after $T = 1000$ sequential edits on Qwen-2.5-7B-Instruct. Each point represents a historical edit, obtained by projecting its gradient drift vector Δg_j into two dimensions. The value reported in each panel is the average drift magnitude, computed as the mean of the per-sample ℓ_2 norm $\|d_j\|_2$ over the same historical subset. More compact distributions near the origin indicate better preservation of local gradient behavior.

Experimental setup. Following the setup in Section 6.3, after completing $T = 1000$ sequential edits, we select historical samples from the earliest 20% of edited knowledge. For each historical edit $e_j = (x_j, y_j)$, we compute the gradient difference between its query gradient and the anchor gradient stored at the time of editing:

$$\Delta g_j = g^{\text{query}}(e_j) - g^{\text{anc}}(e_j).$$

The drift magnitude of this sample is measured by the ℓ_2 norm $\|d_j\|_2$, where $d_j = |\Delta g_j|$. The average drift reported in each panel is computed by averaging $\|d_j\|_2$ over the selected historical samples, which is the same metric used in Section 6.3.

To visualize the distribution, we project the high-dimensional signed drift vectors Δg_j into a two-dimensional space. Each point corresponds to one historical edit, and its position reflects the projected direction and magnitude of its gradient drift. The origin represents zero drift; therefore, points closer to the origin indicate smaller changes in local gradient behavior.

Results. The visualization results are shown in Fig. 7. We compare several representative methods, including MEMIT, ROME, PRUNE, AlphaEdit, HSE, and GDR.

As shown in Fig. 7, baseline methods exhibit more dispersed distributions, with many historical samples located farther from the zero-drift origin. This indicates that their local edit-space gradients have changed substantially after long sequential editing. In contrast, GDR produces the most compact distribution near the origin and achieves the smallest average drift magnitude, showing that it better preserves the local gradient behavior of historical edits.

D.7.3 Quantitative analysis of metric-specific gradient drift

To further examine whether controlling gradient drift also stabilizes other aspects of model behavior, we quantify gradient drift under different evaluation dimensions. While GDR is optimized using the editing objective, we evaluate whether the resulting stability extends to efficacy, generalization, and locality.

Metric-specific gradient drift. For each historical edit e_j, we compute metric-specific query and anchor gradients using the same input-target pairs as the corresponding evaluation metric.

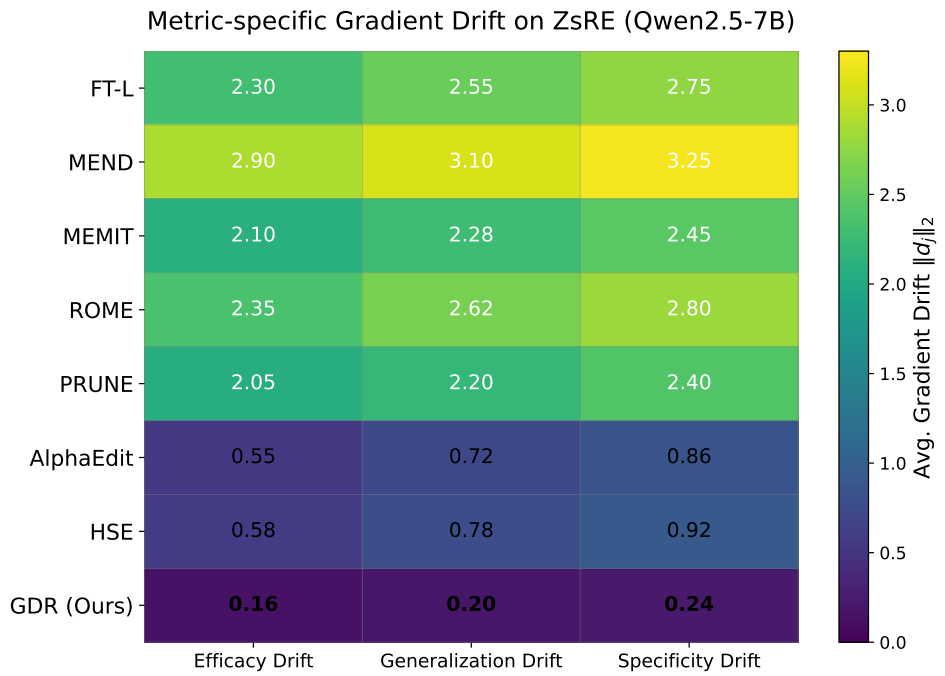

Figure 8: Metric-specific gradient drift after $T = 1000$ sequential edits on Qwen-2.5-7B-Instruct. Each entry reports the average per-sample ℓ_2 gradient drift under the corresponding evaluation dimension. Lower values indicate better preservation of historical behavior.

Specifically, efficacy drift is computed on the original edit prompt and target, generalization drift is computed on paraphrased prompts with the edited target, and specificity drift is computed on locality samples where the original model behavior should be preserved. For each metric $m \in \{\mathrm{Eff}, \mathrm{Gen}, \mathrm{Spe}\}$, we define

$$D_j^{(m)} = \|g_m^{\mathrm{query}}(e_j) - g_m^{\mathrm{anc}}(e_j)\|_2 , \tag{67}$$

where both gradients are computed with respect to the editing variable δ. We report the average value of $D_j^{(m)}$ over the selected historical samples. Therefore, lower values indicate that the corresponding aspect of historical behavior is better preserved.

Experimental setup. We conduct this analysis on Qwen-2.5-7B-Instruct using the ZsRE dataset after $T = 1000$ sequential edits. We use the same historical subset as in Section 6.3. To keep the scale consistent with Fig. 2, the efficacy drift values correspond to the average ℓ_2 drift magnitude reported for Qwen-2.5-7B at $T = 1000$.

Results. As shown in Fig. 8, GDR achieves the lowest gradient drift across all three evaluation dimensions. Compared with AlphaEdit, GDR reduces the drift to only 29.1%, 27.8%, and 27.9% on efficacy, generalization, and specificity , respectively; compared with HSE, the corresponding ratios are 27.6%, 25.6%, and 26.1%. This indicates that GDR preserves a substantially more stable edit-space gradient structure, and that the benefit extends beyond the directly optimized efficacy objective to generalization and specificity behavior.These results provide additional evidence that gradient drift is closely related to the broader behavioral stability of sequentially edited models.