EvoEdit: Evolving Null-space Alignment for Robust and Efficient Knowledge Editing

Sicheng Lyu[1,2,3*] **Yu Gu**[1*] **Xinyu Wang**[1,3] **Jerry Huang**[2,4]
Sitao Luan[2,4] **Yufei Cui**[1,2] **Xiao-Wen Chang**[1] **Peng Lu**[4†]
[1]McGill University [2]Mila - Quebec AI Institute [3]SimpleWay.AI [4]Université de Montréal
sicheng.lyu@mail.mcgill.ca yu.gu4@mail.mcgill.ca, peng.lu@umontreal.ca

Abstract

Ensuring that Large Language Models (LLMs) remain useful throughout their lifetimes requires frequent updates to maintain factual accuracy, yet standard "locate-then-edit" methods suffer from catastrophic interference in sequential settings, where successive updates degrade previously integrated knowledge. In this work, we propose EvoEdit, a framework that achieves stable, large-scale sequential editing through *sequential null-space alignment*. Unlike previous works, EvoEdit dynamically projects knowledge edits into the null space of both original and previously modified knowledge, with theoretical guarantees of output invariance across extended edit sequences. This approach balances model adaptability with knowledge preservation, preventing the cumulative interference that commonly leads to model collapse with previous methods. Empirical evaluations across multiple LLMs demonstrates state-of-the-art performance at moderate scales (e.g., 2K edits) and provides substantial improvements in stability and retention at large scales of up to 10K edits. Furthermore, the method achieves up to a *3.5× speedup* over existing baselines, establishing EvoEdit as a principled, efficient, and scalable solution for reliable knowledge management in real-world LLM deployments. Code is available at here.

1 Introduction

Large language models (LLMs) have demonstrated a remarkable ability to store and recall vast amounts of knowledge during pre-training (OpenAI, 2023; Anthropic, 2023; Dubey et al., 2024; Yang et al., 2024a; DeepSeek-AI et al., 2025; Comanici et al., 2025; Yang et al., 2025a), enabling them to utilize this knowledge for downstream tasks. However, powerful as they may be, becoming outdated remains a concern, with their static knowledge leading to eventual hallucinations or factual errors over time (Cao et al., 2021; Mitchell et al., 2022). While re-training with updated data offers a straightforward solution, the computational expensive and risks of overfitting or catastrophic forgetting (Kirkpatrick et al., 2017) prohibits this as an effective approach. Alternatively, *model editing* has emerged as a targeted updating procedure of specific factoids without additional training (Wang et al., 2025b; Gupta et al., 2024).

Most editing methods follow a *locate-then-edit paradigm* (Meng et al., 2022, 2023), which first identifies a small set of influential parameters within the model and then applies a perturbation to integrate new knowledge. While effective at isolated edits, recent studies (Yang et al., 2025b; Wang et al., 2025a) reveal that these approaches suffer from catastrophic interference when deployed in sequential editing scenarios, where multiple updates are applied over time (Yang et al., 2024b). As new perturbations accumulate, previous edits are also disrupted, leading to greater difficulty in preserving knowledge and more severe issues such as model collapse. Thus a critical barrier remains when deploying model editing in realistic settings; continual updates are essential for keeping LLMs reliable and up-to-date. Many methods fail to capture this real-world complexity and are only evaluated in synthetic frameworks (Yao et al., 2023; Wang et al., 2025b). Thus under more realistic conditions which better mimic the ever-changing world (Wang et al., 2025b), performance of existing methods crater, revealing that current approaches degrade catastrophically after only a few hundred edits and fail to scale to the real-world.

This paper addresses the challenge of *scalable sequential factual knowledge editing* in large language models. We propose EvoEdit, a sequential null-space alignment framework designed for high-volume updates to a model's stored facts. Empirical evaluations across multiple benchmarks show that EvoEdit matches state-of-the-art editing per-

[*]Equal contribution.
[†]Corresponding author.

Findings of the Association for Computational Linguistics: ACL 2026, pages 1520–1540

formance while significantly improving knowledge retention and achieving up to a *3.5× speedup*. We summarize our key contributions as follows:

- We propose a robust and efficient subspace alignment method that mitigates null-space drift caused by sequential knowledge updates.
- We provide theoretical guarantees of output invariance across extended edit sequences through a provable null-space projection mechanism.
- We develop a numerically stable and computationally efficient implementation leveraging the Woodbury matrix identity for scalable updates.

2 Related Work

2.1 Evaluation of Model Editing

Current evaluations of model editing primarily focus on assessing both the effectiveness of edits and their adverse impacts on overall model performance. Effectiveness is typically measured along three key dimensions: 1) *Reliability* — the success rate of applying the intended knowledge edits; 2) *Generalization* — the ability of the edited knowledge to remain consistent under paraphrased or semantically similar inputs; 3) *Locality* — the extent to which the edit influences unrelated/peripheral knowledge. These metrics collectively capture whether an edit operates within its intended scope (Meng et al., 2022; Wang et al., 2024; Yao et al., 2023). Beyond these, recent work has also examined the fine-grained effects of post-editing, such as mitigating biases or preventing the injection of harmful information into large language models (LLMs) (Chen et al., 2024b,a).

2.2 Taxonomy of Model Editing

The current model editing approaches can be categorized as either introducing an auxiliary mechanism or updating (a subset of) the model's parameters to store new knowledge (Wang et al., 2025b). This paper focuses on the latter and follows the locate–then–edit paradigm proposed by ROME (Meng et al., 2022). MEMIT (Meng et al., 2023) extends this to support batched edits. Other methods adopt meta-learning, exemplified by MEND (Mitchell et al., 2022) and KE (Cao et al., 2021), which learn to predict parameter updates.

3 Preliminaries

Model editing (Meng et al., 2022, 2023; Fang et al., 2025) seeks to update factual knowledge encoded in a frozen language model through one-shot or batched parameter updates. Facts are commonly represented as triples (s, r, o), where s denotes a subject, r a relation, and o an object. For example, an arbitrary fact can be stored as

$$(s, r, o) = (\text{“CEO of Tesla”},\ \text{“is”},\ \text{“Tom Zhu”})\,.$$

Given a prompt containing the pair (s, r), the model is expected to generate the token(s) corresponding to o. If this fact is edited, the same prompt should instead produce a new target object $\tilde{o}$, e.g.,

$$(s, r, \tilde{o}) = (\text{“CEO of Tesla”},\ \text{“is”},\ \text{“Elon Musk”})\,.$$

Ideally, the change induced from o to $\tilde{o}$ for a specific fact should not impact un-related or ancillary facts, *i.e.,* when updating (s_1, r_1, o_1) to $(s_1, r_1, \tilde{o}_1)$ (s_2, r_2, o_2) should not be influenced if the relationship between these two facts is weak.

3.1 Feed-Forward Blocks as Associative Memory

We consider auto-regressive LLMs, where the next token x_t is predicted from its preceding context. Let $\boldsymbol{h}^{(l-1)}$ denote the hidden state input to layer l of x_t. The feed-forward network (FFN) computes

$$\underbrace{\boldsymbol{m}^{(l)}}_{\boldsymbol{v}} = \boldsymbol{W}_{\text{out}}^{(l)} \underbrace{\sigma\Big(\boldsymbol{W}_{\text{in}}^{(l)}\left(\boldsymbol{h}^{(l-1)} + \boldsymbol{a}^{(l)}\right)\Big)}_{\boldsymbol{k}},$$

where $\boldsymbol{a}^{(l)}$ is the output of the attention block, $\boldsymbol{W}_{\text{in}}^{(l)}$ and $\boldsymbol{W}_{\text{out}}^{(l)}$ are the learnable projection matrices, and σ is the activation function. This transformation can be interpreted as an associative memory (Geva et al., 2021; Elhage et al., 2022). The nonlinear projection $\sigma\left(\boldsymbol{W}_{\text{in}}^{(l)}\left(\boldsymbol{h}^{(l-1)} + \boldsymbol{a}^{(l)}\right)\right)$ generates a key representation $\boldsymbol{k}$ that encodes the input context, while the output projection $\boldsymbol{W}_{\text{out}}^{(l)}$ maps this key to a corresponding value $\boldsymbol{v}$. When editing knowledge, each update can be represented as a key–value pair, where $\boldsymbol{k}$ encodes (s, r) and $\boldsymbol{v}$ encodes the new target object $\tilde{o}$. This associative interpretation provides a principled rationale for knowledge editing: modifying either the key space or the output projection directly alters how facts are retrieved and expressed. In locate-then-edit frameworks, for instance, the goal is to update $\boldsymbol{W}_{\text{out}}^{l}$ to reflect the revised associations. For notational simplicity, we omit layer-specific subscripts and superscripts of $\boldsymbol{W}_{\text{out}}^{l}$ in what follows.

3.2 Model Editing in LLMs

In the locate-then-edit paradigm, model parameters $\boldsymbol{W}$ are perturbed by a small update $\boldsymbol{\Delta}$ to reflect the knowledge update. Specifically, given a knowledge triple $(s, r, \tilde{o})$, the modified weights $\boldsymbol{W}+\boldsymbol{\Delta}$ are expected to encode the corresponding new key–value association, ensuring $(\boldsymbol{W}+\boldsymbol{\Delta})\boldsymbol{k} = \boldsymbol{v}$ ideally. The central challenge lies in determining an optimal $\boldsymbol{\Delta}$: one that effectively injects or replaces the target knowledge while minimizing unintended interference with knowledge that should be preserved (Meng et al., 2023). This can be formulated as the following optimization problem.

$$\min_{\boldsymbol{\Delta}} (\underbrace{\|(\boldsymbol{W}+\boldsymbol{\Delta})\boldsymbol{K}_1 - \boldsymbol{V}_1\|^2}_{\text{(1) Ensure effective edits}} + \underbrace{\|(\boldsymbol{W}+\boldsymbol{\Delta})\boldsymbol{K}_0 - \boldsymbol{V}_0\|^2}_{\text{(2) Preserve model knowledge}}), \quad (1)$$

where $\boldsymbol{W} \in \mathbb{R}^{d_v \times d_K}$, $\boldsymbol{K}_1 \in \mathbb{R}^{d_K \times u}$ and $\boldsymbol{V}_1 \in \mathbb{R}^{d_v \times u}$ denote the matrices by collecting the $\boldsymbol{k}$ and $\boldsymbol{v}$ of renewing knowledge, $\boldsymbol{K}_0 \in \mathbb{R}^{d_K \times N}$ and $\boldsymbol{V}_0 \in \mathbb{R}^{d_v \times N}$ denote the matrices by collecting the $\boldsymbol{k}$ and $\boldsymbol{v}$ of the knowledge we would like to preserve. $\boldsymbol{K}_0$ is usually estimated by utilizing sufficient text input (Meng et al., 2023), e.g., 100K (s, r, o) triplets from Wikipedia are chosen randomly to encode $\boldsymbol{K}_0$.Joint optimization reveals a fundamental preservation-injection dilemma: parameter updates for new facts often disrupt retained knowledge. To resolve this, recent null-space projection methods have emerged to mitigate such interference.

3.3 Null-space of Preserved Knowledge

Null-space Projection. Given two matrices $\boldsymbol{A}$ and $\boldsymbol{B}$, if $\boldsymbol{B}^\top \boldsymbol{A} = \boldsymbol{0}$, then by definition the columns of $\boldsymbol{B}$ lie in the left null space of $\boldsymbol{A}$ (or equivalently the (right) null-space of $\boldsymbol{A}^\top$). In the context of model editing, consider an update $\boldsymbol{\Delta}$ projected onto the left null space of $\boldsymbol{K}_0$, i.e., $\boldsymbol{\Delta}\boldsymbol{P}\boldsymbol{K}_0 = \boldsymbol{0}$. Under this constraint, we obtain

$$(\boldsymbol{W}+\boldsymbol{\Delta}\boldsymbol{P})\boldsymbol{K}_0 = \boldsymbol{W}\boldsymbol{K}_0 = \boldsymbol{V}_0. \quad (2)$$

This ensures that the update $\boldsymbol{\Delta}\boldsymbol{P}$ leaves the existing key–value mappings $\boldsymbol{K}_0 \mapsto \boldsymbol{V}_0$ unchanged. Thus, if the null-space projector $\boldsymbol{P}$ of the historical knowledge matrix $\boldsymbol{K}_0$ is available, any update can be projected through $\boldsymbol{P}$ to ensure that $\boldsymbol{\Delta}\boldsymbol{P}$ preserves the existing mappings, thereby safeguarding preserved knowledge from interference.

Null-space Projector Estimation. Directly computing the null-space projector $\boldsymbol{P}$ of the historical knowledge key matrix $\boldsymbol{K}_0$ is intractable, since $\boldsymbol{K}_0$ typically possesses a growing number of columns as knowledge editing progresses and its column vectors are rarely orthonormal to each other. To address this, we exploit the fact that $\boldsymbol{K}_0$ and its non-central covariance matrix $\boldsymbol{K}_0\boldsymbol{K}_0^\top$ share a left null space, i.e.,Null($\boldsymbol{K}_0^T$) = Null($\boldsymbol{K}_0\boldsymbol{K}_0^T$). Thus the null space can be estimated via $\boldsymbol{K}_0\boldsymbol{K}_0^\top$, avoiding materializing the dense matrix $\boldsymbol{K}_0$ and reducing memory cost because the row dimension is fixed and typically much smaller than the column dimension (Wang et al., 2021; Fang et al., 2025; Sun et al., 2025). One way of obtaining the (approximate) left null-space basis is first to perform singular value decomposition (SVD) on the covariance matrix:

$$\begin{aligned} &\{\boldsymbol{U}, \boldsymbol{\Lambda}, \boldsymbol{U}^\top\} = \mathrm{SVD}\left(\boldsymbol{K}_0\boldsymbol{K}_0^\top\right), \\ &\bar{\boldsymbol{U}} = \boldsymbol{U}_{[:, i:d_K]}, \;\; \boldsymbol{\Lambda}_{[i]} < \tau \leq \boldsymbol{\Lambda}_{[i-1]}, \end{aligned} \quad (3)$$

where τ is a threshold (e.g., 10^{-2}), and $\bar{\boldsymbol{U}}$ is formed by the singular vectors associated with singular values smaller than τ. The threshold is necessary because singular values are rarely exactly zero in practice. The resulting (approximate) null-space orthogonal projector is then defined as

$$\boldsymbol{P} = \bar{\boldsymbol{U}}\bar{\boldsymbol{U}}^\top. \quad (4)$$

4 Methodology

We begin by examining how sequential knowledge updates induce null-space drift, a key challenge that undermines the stability of continual editing. To address this, we propose `EvoEdit`—a novel framework that incrementally aligns the null space through a stable, synergistic mechanism, ensuring that each new edit integrates cleanly without degrading prior updates.

4.1 Null-Space Drift and Interference

During sequential editing, the goal is to determine a series of perturbations $\{\boldsymbol{\Delta}_1, \ldots, \boldsymbol{\Delta}_t\}$ corresponding to knowledge updates $\{\boldsymbol{K}_1, \ldots, \boldsymbol{K}_t\}$. Two challenges exist: not only must the initial preserved knowledge $\boldsymbol{K}_0$ remain intact, but the newly injected knowledge should also remain robust against contamination from subsequent updates. Prior work, including `AlphaEdit` (Fang et al., 2025) and `LangEdit` (Sun et al., 2025), demonstrates the benefits of performing updates within the null space of preserved knowledge. `AlphaEdit`, however, employs a fixed null-space projector, ignoring the

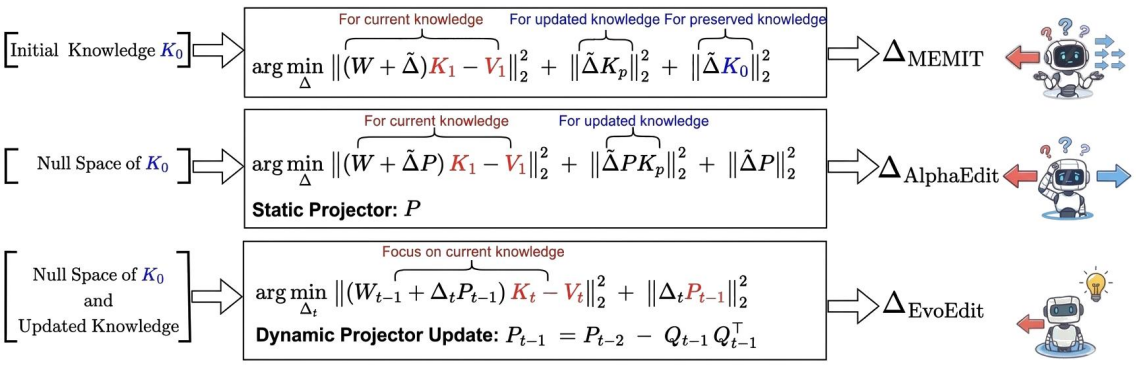

Figure 1: Evolution of Model Editing Objectives. Comparison between `MEMIT` (top), `AlphaEdit` with a static projector (middle), and the proposed `EvoEdit` (bottom), which utilizes a Dynamic Projector Update to sequentially refine knowledge while preserving the null space of previous states.

drift induced by sequential updates. `LangEdit` mitigates this by recomputing the null space after each edit, but its reliance on the SVD of the non-centered covariance matrix $\widehat{\boldsymbol{K}}_t\widehat{\boldsymbol{K}}_t^\top$, where $\widehat{\boldsymbol{K}}_t = [\boldsymbol{K}_0, \ldots, \boldsymbol{K}_t]$, however, because $\widehat{\boldsymbol{K}}$ is rank-deficient, Performing Singular Value Decomposition (SVD) on this ill-conditioned matrix yields highly unstable results, as the disproportionately wide range of magnitudes causes small singular values to be lost to floating-point roundoff errors, while the corresponding singular vectors become significantly distorted and cease to accurately span the intended subspace.

4.2 Efficient Null-space Alignment

We now describe sequential editing as an optimization problem. Given the model weights $\boldsymbol{W}_{t-1}$ and the collection of all initial and updated key-value pairs $\left(\widehat{K}_t, \widehat{V}_t\right)$, our goal is to determine the optimal perturbation $\hat{\Delta}$ to inject $(\boldsymbol{K}_t, \boldsymbol{V}_t)$ by solving the following problem at time step t:

$$\begin{aligned}\min_{\boldsymbol{\Delta}_t} &\left\|\left(\boldsymbol{W}_{t-1}+\hat{\boldsymbol{\Delta}}\right)\boldsymbol{K}_t - \boldsymbol{V}_t\right\|^2 \\ &+ \left\|\left(\boldsymbol{W}_{t-1}+\hat{\boldsymbol{\Delta}}\right)\widehat{\boldsymbol{K}}_{t-1} - \widehat{\boldsymbol{V}}_{t-1}\right\|^2,\end{aligned} \tag{5}$$

where $\widehat{\boldsymbol{V}}_{t-1} = [\boldsymbol{V}_0, \ldots, \boldsymbol{V}_{t-1}]$ denotes the collection of all initial and updated values up to step $t-1$. To address this, we require the null-space projector of $\widehat{\boldsymbol{K}}_{t-1}$. Instead of recomputing it from $\widehat{\boldsymbol{K}}_{t-1}\widehat{\boldsymbol{K}}_{t-1}^\top$, we introduce a sequential alignment approach that updates the null-space projection based on the new key matrix $\boldsymbol{K}_{t-1}$.

Denote by $\boldsymbol{P}_{t-2} \in \mathbb{R}^{d_K \times d_K}$ the orthogonal projector onto the left null space of $\widehat{\boldsymbol{K}}_{t-2}$. To capture

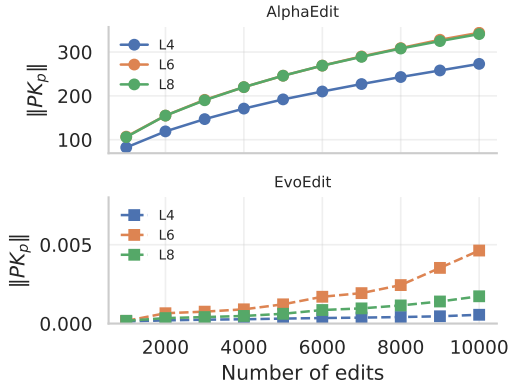

Figure 2: Quantitative Analysis of Null-space Drift. The growth of $\|PK_p\|_F$ in `AlphaEdit` (solid) across different layers indicates significant drift as the number of edits increases, which forces a trade-off between new knowledge acquisition and interference suppression. `EvoEdit` (dashed) eliminates this drift by dynamically adapting the projector, maintaining stable geometry even under extended sequential editing (up to 10k edits).

the portion of this null space that aligns with the basis directions induced by $\boldsymbol{K}_{t-1}$, we perform a singular value decomposition (SVD) w.r.t. $\boldsymbol{P}_{t-2}\boldsymbol{K}_{t-1}$, and select singular vectors with singular values greater than a pre-defined threshold τ:

$$\begin{aligned}\{\boldsymbol{U}, \boldsymbol{\Sigma}, \boldsymbol{U}^\top\} &= \mathrm{SVD}(\boldsymbol{P}_{t-2}\boldsymbol{K}_{t-1}), \\ \boldsymbol{Q}_{t-1} &= \boldsymbol{U}_{[:,:i]}, \quad \Sigma_{[i]} > \tau > \Sigma_{[i-1]}.\end{aligned} \tag{6}$$

Because $\boldsymbol{K}_{t-1}$ has far fewer columns than $\boldsymbol{K}_0$ or $\widehat{\boldsymbol{K}}_{t-2}$, this SVD step is both more efficient and numerically stable than recomputing the full covariance matrix. The updated projector is then obtained via the deflation step

$$\boldsymbol{P}_{t-1} = \boldsymbol{P}_{t-2} - \boldsymbol{Q}_{t-1}\boldsymbol{Q}_{t-1}^\top. \tag{7}$$

where $\boldsymbol{P}_0$ is the null-space projector w.r.t. the initial preserved knowledge $\boldsymbol{K}_0$. $\boldsymbol{Q}_{t-1}$ indicates the directions related to the newly updated knowledge $\boldsymbol{K}_{t-1}$. $\boldsymbol{P}_{t-1}$ is the aligned projector that satisfies $\boldsymbol{P}_{t-1}\widehat{\boldsymbol{K}}_{t-1} = \boldsymbol{0}$.

4.3 Sequential Editing via aligned Null-space Projector

Next, we present how to leverage the aligned projector to solve the sequential editing problem. We input $\hat{\boldsymbol{\Delta}}$ with $\boldsymbol{\Delta}\boldsymbol{P}_{t-1}$ to Equation (5):

$$\begin{aligned}\min_{\boldsymbol{\Delta}_t} &\|(\boldsymbol{W}_{t-1}+\boldsymbol{\Delta}_t\boldsymbol{P}_{t-1})\,\boldsymbol{K}_t-\boldsymbol{V}_t\|^2 \\ &+\left\|(\boldsymbol{W}_{t-1}+\boldsymbol{\Delta}_t\boldsymbol{P}_{t-1})\widehat{\boldsymbol{K}}_{t-1}-\widehat{\boldsymbol{V}}_{t-1}\right\|^2,\end{aligned} \tag{8}$$

Since the projector $\boldsymbol{P}_{t-1}$ guarantees that $\boldsymbol{\Delta}_t\boldsymbol{P}_{t-1}\widehat{\boldsymbol{K}}_{t-1} = \boldsymbol{0}$, the second term in Equation (8) becomes independent of $\boldsymbol{\Delta}_t$ and can therefore be omitted. To further stabilize convergence, we introduce a regularization term $\|\boldsymbol{\Delta}_t\boldsymbol{P}_{t-1}\|^2$, yielding the final optimization problem:

$$\min_{\boldsymbol{\Delta}_t}\|(\boldsymbol{W}_{t-1}+\boldsymbol{\Delta}_t\boldsymbol{P}_{t-1})\,\boldsymbol{K}_t-\boldsymbol{V}_t\|^2+\|\boldsymbol{\Delta}_t\boldsymbol{P}_{t-1}\|^2. \tag{9}$$

Defining the residual of the current edit as $\boldsymbol{R}_t = \boldsymbol{V}_t - \boldsymbol{W}_{t-1}\boldsymbol{K}_t$ for notational simplicity, we solve the optimization problem using the normal equations (Lang, 2012; Strang, 2022), yielding

$$\boldsymbol{\Delta}_t\boldsymbol{P}_{t-1} = \boldsymbol{R}_t\boldsymbol{K}_t^\top\boldsymbol{P}_{t-1}\left(\boldsymbol{K}_t\boldsymbol{K}_t^\top\boldsymbol{P}_{t-1}+\boldsymbol{I}\right)^{-1}. \tag{10}$$

Direct computation of Equation (10) requires $O(d_K^3)$ flops due to the matrix inversion; thus, we leverage the Woodbury matrix identity (Guttman, 1946; Woodbury, 1950; Hager, 1989; Higham, 2002) for a more efficient formulation. Full derivations are provided in Appendix A.4.

$$\boldsymbol{\Delta}_{\texttt{EvoEdit}} = \boldsymbol{R}_t\left(\boldsymbol{K}_t^\top\boldsymbol{P}_{t-1}\boldsymbol{K}_t+\boldsymbol{I}_r\right)^{-1}\boldsymbol{K}_t^\top\boldsymbol{P}_{t-1}. \tag{11}$$

Algorithm 2 provides a numerically stable computation method for Equation (11). We provide the complete algorithm of `EvoEdit` in Algorithm 1.

Complexity analysis. The per-edit complexity of `EvoEdit` is $O(d_K(rn+n^2)+n^3)$, where $n = \max(n_t, n_{t-1})$. By representing the orthogonal projector in low-rank form as $\boldsymbol{P} = I_{d_K} - \boldsymbol{Q}\boldsymbol{Q}^\top$ with $\boldsymbol{Q}\in\mathbb{R}^{d_K\times r}$, the method eliminates the $O(d_K^3)$ bottleneck typical of standard null-space approaches. Through SVD-based alignment and Woodbury-style rearrangements, the hidden dimension d_K appears only linearly in the total cost. Since the cubic dependency is confined strictly to the edit size n_t—which is typically several orders of magnitude smaller than d_K—`EvoEdit` enables highly efficient updates without the need to materialize or invert dense $d_K\times d_K$ matrices.

Null-space drift comparison. In `AlphaEdit`, the regularization term $\|PK_p\|_F$ grows with the number of edits, increasingly constraining new updates and forcing a trade-off between fitting $(W+\tilde{\Delta}P)K_1 \approx V_1$ and suppressing interference. `EvoEdit` avoids this bottleneck by dynamically adapting the projector. As shown in Figure 2, the Frobenius norm of PK_p in `AlphaEdit` becomes several orders of magnitude larger than in `EvoEdit` across layers, indicating significant null-space drift, whereas `EvoEdit` maintains stable geometry under extended edits.

Algorithm 1 `EvoEdit`: Sequential Editing via Evolving Null-space Alignment

Require: Initial weights $\boldsymbol{W}_0$ and projector $\boldsymbol{P}_0$, Data sequence $\{(s_1,r_1,o_1),\ldots,(s_T,r_T,o_T)\}$

for $t = 1$ to T **do**
  Extract $(\boldsymbol{K}_t, \boldsymbol{V}_t)$ with (s_t, r_t, o_t).
  if $t > 1$ **then**
    Project $\boldsymbol{Z} \leftarrow \boldsymbol{P}_{t-2}\boldsymbol{K}_{t-1}$
    Decompose $(\boldsymbol{U}_{t-1},\boldsymbol{\Sigma},\boldsymbol{V}_{t-1}) \leftarrow \mathrm{SVD}(\boldsymbol{Z})$
    Extract singular vectors with large singular values $\boldsymbol{Q}_{t-1} = \boldsymbol{U}_{t-1[:,:d]}$
    Update $\boldsymbol{P}_{t-1} \leftarrow \boldsymbol{P}_{t-2} - \boldsymbol{Q}_{t-1}\boldsymbol{Q}_{t-1}^\top$
  else
    $\boldsymbol{P}_{t-1} \leftarrow \boldsymbol{P}_0$
  end if
  Compute residual: $\boldsymbol{R}_t \leftarrow \boldsymbol{V}_t - \boldsymbol{W}_{t-1}\boldsymbol{K}_t$
  Solve closed-form update (Equation (11)):

$$\boldsymbol{\Delta}_t\boldsymbol{P}_{t-1} = \boldsymbol{R}_t\left(\boldsymbol{I}+\boldsymbol{K}_t^\top\boldsymbol{P}_{t-1}\boldsymbol{K}_t\right)^{-1}\boldsymbol{K}_t^\top\boldsymbol{P}_{t-1}$$

  Update weights: $\boldsymbol{W}_t \leftarrow \boldsymbol{W}_{t-1} + \boldsymbol{\Delta}_t\boldsymbol{P}_{t-1}$
end for

Ensure: Updated model $\boldsymbol{W}_T$

4.4 Theoretical Analysis

We analyze the *projector-alignment mechanism* and show that the iterate $\boldsymbol{P}_{t-1}$ serves as (or closely approximates) the orthogonal projector onto the null space of the accumulated knowledge matrix $\widehat{\boldsymbol{K}}_{t-1} = [\boldsymbol{K}_0, \boldsymbol{K}_1, \ldots, \boldsymbol{K}_{t-1}] \in \mathbb{R}^{d\times r_{t-1}}$. In the exact (non-truncated) setting, the null space of

$\boldsymbol{P}_{t-1}$ coincides with the column space of $\widehat{\boldsymbol{K}}_{t-1}$:

$$\mathrm{Null}(\boldsymbol{P}_{t-1}) = \mathrm{Range}(\widehat{\boldsymbol{K}}_{t-1}),$$

implying that $\boldsymbol{P}_{t-1}$ projects onto the subspace orthogonal to all previously edited knowledge. Under the practical truncation scheme motivated by numerical stability, this equivalence holds approximately, and the deviation is provably bounded (see Theorem 4.2). All proofs are provided in Appendix A. First, define the step-j projected keys

$$\boldsymbol{R}_j \coloneqq \boldsymbol{P}_{j-1}\boldsymbol{K}_j.$$

Theorem 4.1 (Exact equivalence without truncation)**.** *Suppose that at each step j, the update uses $\boldsymbol{Q}_j$ as any orthonormal basis of* $\mathrm{Range}(\boldsymbol{R}_j) = \mathrm{Range}(\boldsymbol{P}_{j-1}\boldsymbol{K}_j)$*, i.e.,* no truncation is applied and all nonzero left singular directions of $\boldsymbol{R}_j$ are retained. *Then, for all $t \geq 0$,*

$$\mathrm{Null}(\boldsymbol{P}_{t-1}) = \mathrm{Range}\left(\widehat{\boldsymbol{K}}_{t-1}\right).$$

For more general cases, we also provide the theoretical bound of the deviation between the truncated projector and the ideal projector.

Theorem 4.2 (Global error bound with truncation)**.** *For any $t \geq 1$, let $\widetilde{\boldsymbol{P}}_t$ be the projector obtained by applying truncation thresholds $\{\tau_j\}_{j=1}^{t}$. Then the cumulative deviation satisfies*

$$\left\|\widetilde{\boldsymbol{P}}_t - \boldsymbol{P}_t^{\star}\right\|_2 \leq \min\left\{1,\ \sum_{j=1}^{t} \frac{\|\boldsymbol{E}_j\|_2}{\sigma_{q_j} - \sigma_{q_j+1}} + \max_j \frac{\|\boldsymbol{\Sigma}_{2,j}\|_2}{\left\|\boldsymbol{R}_j^{*}\right\|_2}\right\},$$

where $\boldsymbol{\Sigma}_{2,j}$ denotes the truncated singular values at step j, and q_j is the largest index with $\sigma_{q_j} \geq \tau_j$.

Corollary 4.3 (Interference bound)**.** *Define $\boldsymbol{C}_t \coloneqq \left[\boldsymbol{R}_1^{\star},\ \boldsymbol{R}_2^{\star},\ \ldots,\ \boldsymbol{R}_t^{\star}\right] \in \mathbb{R}^{d\times M}$. Let $\boldsymbol{\Delta}$ be any future edit with $\|\boldsymbol{\Delta}\|_2 \leq \Gamma$, where $\Gamma > 0$. Then*

$$\left\|\boldsymbol{\Delta}\widetilde{\boldsymbol{P}}_t\boldsymbol{x}\right\| \leq \Gamma \left\|\widetilde{\boldsymbol{P}}_t - \boldsymbol{P}_t^{\star}\right\|_2 \|\boldsymbol{x}\| \ \ \forall\, \boldsymbol{x} \in \mathrm{span}(\boldsymbol{C}_t),$$

so the worst-case interference is controlled by the cumulative projector approximation error.

Remark 4.4 (Numerical truncation). In finite precision, one can stabilize the update toward the "ideal" SVD by discarding tiny singular values due to data noise or rounding errors from forming $\boldsymbol{Q}_j$. After truncation, $\boldsymbol{P}_{t-1}$ remains a controlled approximation of the ideal projector, with deviation governed by the spectral gap at the truncation index and the discarded tail energy (truncated small singular values) (Wedin, 1972; Davis and Kahan, 1970).

5 Experiments and Results

In this section, we evaluate our approach against several representative model editing methods. The comparison covers both sequential editing performance and the general editing capability of large language models (LLMs). To further validate the design of `EvoEdit`, we also conduct a comprehensive ablation study and efficiency analysis.

5.1 Experimental Setup

Datasets and Backbone LLMs. We perform experiments on two widely used benchmarks: the CounterFact (Meng et al., 2022) and ZsRE datasets (Levy et al., 2017). To assess model-agnostic applicability, we evaluate on four LLM families of different scales: `Llama-3` (3B and 8B) (Dubey et al., 2024), Qwen2.5 (7B) (Yang et al., 2024a), GPTJ-6B (Wang and Komatsuzaki, 2021) and GPT2-XL (Radford et al., 2019).

Baseline Methods and Evaluation Metrics. We compare our approach with several established model editing methods, including `ROME` (Meng et al., 2022), `MEMIT` (Meng et al., 2023), and `AlphaEdit` (Fang et al., 2025). Following prior work, we adopt five standard metrics to assess performance: *Efficacy*, *Generalization*, *Specificity*, *Fluency*, and *Consistency*.

5.2 Factual Editing Results

Table 1 presents a comprehensive comparison of our proposed `EvoEdit` with prior model editing methods across multiple language models and two benchmark datasets—CounterFact and ZsRE after **2K** edits. `EvoEdit` consistently achieves the highest or second-highest performance across nearly all evaluation metrics, including *Efficacy*, *Generalization*, *Specificity*, *Fluency*, and *Consistency*. For the CounterFact benchmark, `EvoEdit` notably surpasses `AlphaEdit` and other strong baselines such as `MEMIT` and `ROME`, achieving higher rewrite success without sacrificing fluency and consistency.

On ZsRE, `EvoEdit` further exhibits remarkable generalization and specificity, indicating robust transfer of the edited knowledge across diverse contexts. Across all five backbone models, including `Llama-3-8B`, `Qwen2.5-7B-Instruct`, `Llama-3.2-3B`, `GPT2-XL`, and `GPT-J-6B`, our method maintains top-tier efficacy and consistency, underscoring its scalability and reliability in sequential editing scenarios. Overall, these results

Method	Model	COUNTERFACT					ZSRE		
		Eff.↑	Gen.↑	Spe.↑	Flu.↑	Consis.↑	Eff.↑	Gen.↑	Spe.↑
FT	Llama-3-8B	83.33 ±0.37	67.79 ±0.40	46.63 ±0.37	233.72 ±0.22	8.77 ±0.05	30.48 ±0.26	30.22 ±0.32	15.49 ±0.17
MEND		63.24 ±0.31	61.17 ±0.36	45.37 ±0.38	372.16 ±0.80	4.21 ±0.05	0.91 ±0.05	1.09 ±0.05	0.53 ±0.02
InstructEdit		66.58 ±0.24	64.18 ±0.35	47.14 ±0.37	443.85 ±0.78	7.28 ±0.04	1.58 ±0.04	1.36 ±0.08	1.01 ±0.05
ROME		64.40 ±0.41	61.42 ±0.42	49.44 ±0.38	449.06 ±0.26	3.31 ±0.02	2.01 ±0.07	1.80 ±0.07	0.69 ±0.03
MEMIT		65.65 ±0.47	64.65 ±0.42	51.56 ±0.38	437.43 ±1.67	6.58 ±0.11	34.62 ±0.36	31.28 ±0.34	18.49 ±0.19
PRUNE		68.25 ±0.46	64.75 ±0.41	49.82 ±0.36	418.03 ±1.52	5.90 ±0.10	24.77 ±0.27	23.87 ±0.27	20.69 ±0.23
RECT		66.05 ±0.47	63.62 ±0.43	61.41 ±0.37	526.62 ±0.44	20.54 ±0.09	86.05 ±0.23	80.54 ±0.27	31.67 ±0.22
AlphaEdit		98.90 ±0.10	94.22 ±0.19	67.88 ±0.29	622.49 ±0.16	32.4 ±0.11	94.47 ±0.13	91.13 ±0.19	**32.55 ±0.22**
EvoEdit (Ours)		**99.67 ±0.08**	**94.93 ±0.27**	**69.99 ±0.59**	**623.09 ±0.98**	**32.64 ±0.34**	**95.74 ±0.03**	**92.13 ±0.23**	32.41 ±0.30
ROME	Qwen-7B	68.65 ±1.09	62.44 ±0.86	51.35 ±0.94	539.77 ±26.09	4.26 ±1.20	18.93 ±2.46	17.20 ±3.06	7.56 ±1.63
MEMIT		65.65 ±0.47	64.65 ±0.42	51.56 ±0.38	437.43 ±1.67	6.58 ±0.11	34.62 ±0.36	31.28 ±0.34	18.49 ±0.19
AlphaEdit		**99.57 ±0.10**	79.81 ±1.05	82.65 ±0.13	626.80 ±0.33	30.98 ±0.19	**99.74 ±0.14**	91.58 ±0.44	41.58 ±0.34
EvoEdit (Ours)		99.50 ±0.20	**82.50 ±1.55**	**82.73 ±0.08**	626.80 ±0.23	**31.22 ±0.09**	99.54 ±0.17	**91.67 ±0.72**	**41.97 ±1.17**
ROME	Llama-3-3B	69.88 ±2.07	63.79 ±1.67	48.88 ±0.65	**656.96 ±3.04**	1.65 ±0.75	2.25 ±0.80	2.03 ±0.70	0.05 ±0.08
MEMIT		76.03 ±0.86	77.40 ±1.12	60.70 ±0.32	576.23 ±8.92	20.76 ±0.37	0.00 ±0.00	0.00 ±0.00	1.59 ±1.56
AlphaEdit		99.42 ±0.19	96.69 ±0.19	64.71 ±0.26	624.84 ±6.66	32.92 ±1.17	94.44 ±0.10	89.87 ±0.31	**30.09 ±0.13**
EvoEdit (Ours)		**99.77 ±0.08**	**97.27 ±0.08**	**71.32 ±0.50**	630.71 ±2.28	**34.10 ±0.64**	**94.97 ±0.10**	**89.92 ±0.49**	29.26 ±0.22
ROME	GPT2-XL	54.60 ±0.48	51.18 ±0.40	52.68 ±0.33	366.13 ±1.40	0.72 ±0.02	47.50 ±0.43	43.56 ±0.42	14.27 ±0.19
MEMIT		94.70 ±0.22	85.82 ±0.28	60.50 ±0.32	477.26 ±0.54	22.72 ±0.15	79.17 ±0.32	71.44 ±0.36	26.42 ±0.25
AlphaEdit		**99.50 ±0.24**	**93.95 ±0.34**	66.39 ±0.31	**597.88 ±0.18**	**39.38 ±0.15**	**94.81 ±0.30**	86.11 ±0.29	25.88 ±0.21
EvoEdit (Ours)		99.32 ±0.16	92.96 ±0.43	**66.53 ±0.12**	592.04 ±1.12	38.04 ±0.22	94.51 ±1.93	**87.70 ±2.47**	**25.89 ±0.07**
ROME	GPT-J-6B	57.50 ±0.48	54.20 ±0.40	52.05 ±0.31	589.42 ±0.08	3.22 ±0.02	56.42 ±0.42	54.65 ±0.42	9.86 ±0.16
MEMIT		98.55 ±0.11	95.50 ±0.16	63.64 ±0.31	546.28 ±0.88	34.89 ±0.15	94.91 ±0.16	90.22 ±0.23	**30.39 ±0.27**
AlphaEdit		99.75 ±0.08	**96.38 ±0.23**	**75.48 ±0.21**	618.50 ±0.17	**42.08 ±0.15**	**99.79 ±0.14**	96.00 ±0.22	28.29 ±0.25
EvoEdit (Ours)		99.75 ±0.00	95.94 ±0.28	75.21 ±0.53	**619.12 ±1.23**	41.42 ±0.27	98.69 ±0.21	**96.46 ±0.27**	28.62 ±0.21

Table 1: Comparison of EvoEdit with existing methods on the sequential model editing task. We evaluate across five dimensions: *Eff.* (Efficacy), *Gen.* (Generalization), *Spe.* (Specificity), *Flu.* (Fluency), and *Consis.* (Consistency), capturing both the accuracy of edits and their quality in natural language generation. The best-performing results are highlighted in bold, while the second-best results are underlined, facilitating a clear visual comparison of relative performance across methods.

10K Edits	Eff.↑	Gen.↑	Spe.↑	Flu.↑	Consis.↑
MEMIT	49.73	49.24	51.54	389.31	3.45
ROME	47.57	48.45	52.52	465.02	1.83
GRACE	7.83	10.21	**89.20**	568.49	3.61
FT	94.04	84.12	38.15	401.57	21.35
AlphaEdit	66.78	58.27	51.79	489.91	4.59
EvoEdit	**98.29**	**91.21**	63.91	**613.88**	**33.22**

Table 2: Performance comparison on the CounterFact benchmark using Llama-3-8B under 10,000 sequential factual edits. Bold denotes the best result and underline denotes the second best. EvoEdit achieves the strongest performance on four of five metrics.

validate that EvoEdit provides more accurate, consistent, and generalizable edits than existing approaches, establishing a new state of the art in sequential knowledge editing.

We further scale our evaluation to **10K** edits on Llama-3-8B using a batch size of 100 (Table 2). Under this large-scale editing regime, EvoEdit achieves the strongest performance across all five metrics, surpassing both parametric and non-parametric baselines by substantial margins. These results demonstrate that EvoEdit maintains high effectiveness, generalization, specificity, fluency, and consistency even as the number of edits grows, highlighting its robustness and stability in extended editing scenarios.

5.3 General Capability Tests

Setup. To evaluate the impact of sequential editing on foundational model performance, we monitor F1 scores across four representative tasks from the GLUE benchmark: SST (sentiment classification), MRPC (paraphrase detection), MMLU (broad knowledge reasoning), and NLI (natural language inference). These benchmarks serve as a vital proxy for the model's linguistic integrity and reasoning stability throughout the editing process.

Performance. As illustrated in Figure 3, EvoEdit exhibits superior robustness in preserving general capabilities compared to all baseline methods. While ROME and MEMIT suffer from a catastrophic performance collapse—with F1 scores dropping to baseline levels as early as 400 and 800 edits, respectively—EvoEdit maintains

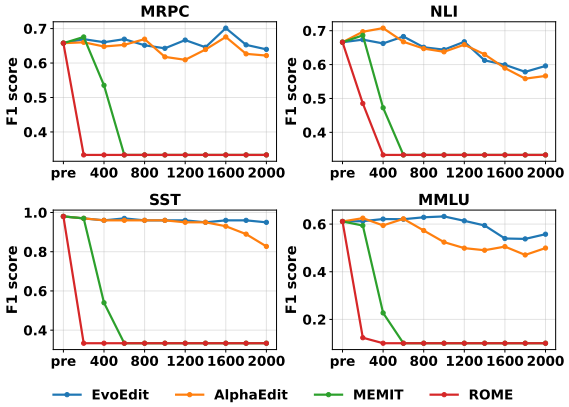

Figure 3: Comparison of EvoEdit with three other baselines (ROME, MEMIT, AlphaEdit) on Llama-3-8B across four evaluation benchmarks (MRPC, NLI, SST, and MMLU). The experiments are conducted with a batch size of 1 over a total of 2000 sequential edits. The x-axis represents the cumulative number of edits applied to the model, while the y-axis indicates the corresponding F1 score, reflecting the effectiveness of each method in preserving and updating model knowledge over successive edits.

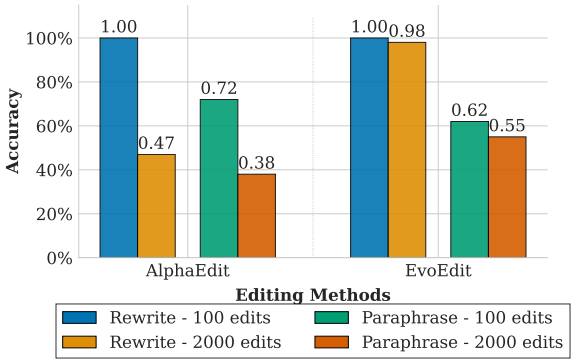

Figure 4: Rewrite and Paraphrase accuracy (%) on the *first 100* edited facts under two editing methods (AlphaEdit and EvoEdit). We report performance at two stages: immediately after the first 100 edits (*100 edits*) and after *2000* total edits. For each stage, bars show accuracy on the *Rewrite* and *Paraphrase* probes. We use a batch size of 1. Evaluation is conducted using Llama-3-8B on COUNTERFACT.

high, stable scores throughout the entire 2,000-edit sequence. Even compared to AlphaEdit, which demonstrates a noticeable degradation on SST and MRPC as edits progress, EvoEdit demonstrates a remarkably stable trajectory, particularly evident in the SST and MMLU panels. Ultimately, even at the 2,000-edit mark, EvoEdit preserves a significant portion of its pre-edit performance, confirming that its dynamic projector effectively mitigates the interference that typically cripples standard sequential editing paradigms.

6 Analysis and Discussion

6.1 Robust Analysis

Accuracy Drop of Earlier Edits. We investigate how well earlier edits are preserved when subsequent edits are added. Specifically, we record the rewrite accuracy and paraphrase accuracy of the first 100 edits at two checkpoints: after 100 (left) and 2000 (right) edits. Figure 4 shows a substantial performance drop for these early edits in both rewrite and paraphrase accuracy (a decrease of 53% in rewrite accuracy and 34% in paraphrase accuracy), whereas the drop under EvoEdit is much smaller than under AlphaEdit (a decrease of 2% in rewrite accuracy and 7% in paraphrase accuracy). This serves as an indication that the null-space projector updates are effective in future edits to not interfere with previous ones, suggesting that EvoEdit can better preserve earlier edits.

6.2 Efficiency Analysis

Speed-up. We evaluate editing efficiency of EvoEdit against AlphaEdit on an NVIDIA H100 (80GB) using a fixed sequence of 500 edits to ensure a fair comparison. As summarized in Table 3, EvoEdit substantially reduces per-edit latency and delivers significant speed-ups across different batch sizes (BS). We specifically benchmark two critical components: the time required to compute the update matrix (*Solve*) and the time required to recompute the projection matrix at each step using the update (*Proj*). In detail, the $O(d^3)$ solver used by AlphaEdit is replaced in EvoEdit by a smaller k_t-sized inner system, minimizing the *Solve* cost. While AlphaEdit does not update projection matrices, EvoEdit still achieves substantial gains in total wall-clock time, ranging from 1.98× to 3.53× faster depending on the model and batch size.

Memory consumption. Across 1000 edits on Llama-3-8B, EvoEdit achieves markedly better memory efficiency than existing approaches, lowering peak allocated and peak reserved GPU memory by up to 14% and 15% (Table 4). Beyond the raw savings, this reduction translates into a consistently leaner execution profile, enabling smoother batched editing, reducing memory-related slowdowns, and improving overall system stability under sustained workloads. These results underscore that EvoEdit delivers high-quality edits while maintaining a more efficient memory footprint.

Model / BS	AlphaEdit (s)		EvoEdit (s)		
	Solve ↓	Total ↓	Solve ↓	Proj ↓	Total ↓
Llama-3-8B					
BS=1	1313.6	1313.6	1.4	661.0	662.4 (x1.98)
BS=10	224.5	224.5	0.3	141.7	142.0 (x1.58)
BS=100	22.0	22.0	0.1	8.8	8.9 (x2.47)
Qwen2.5-7B					
BS=1	3251.3	3251.3	2.3	1491.4	1493.8 (x2.18)
BS=10	461.6	461.6	0.4	169.0	169.4 (x2.72)
BS=100	39.9	39.9	0.1	11.2	11.3 (x3.53)

Table 3: Runtime on the MCF dataset across batch sizes (BS) for `Llama-3-8B` and `Qwen2.5-7B-Instruct` for 500 total edits. *Solve*: time to compute the update matrix; *Proj*: time to update the projector; *Total*: overall runtime in seconds (↓ better).

Method	Peak Alloc. (GB)	Peak Reserved (GB)
MEMIT	36.87	38.45
AlphaEdit	34.79	35.36
EvoEdit	**31.73** (**-14%**)	**32.74** (**-15%**)

Table 4: GPU memory usage over 1000 edits on `Llama-3-8B`. `EvoEdit` reduces peak allocated and reserved memory compared to prior methods.

7 Conclusion

We propose `EvoEdit`, a scalable model editing method that exploits the model's null space to enable continual edits. By addressing the limitations of fixed null-space projectors in prior work, `EvoEdit` mitigates interference. We further provide a numerically stable formulation that reduces editing complexity from cubic to quadratic in the input dimension. Experiments on modern LLMs such as `Llama` and `Qwen` validate our theory, showing improved editing performance across benchmarks while achieving substantial runtime speedups.

8 Limitations and Ethical Considerations

The primary limitation of our work remains the finite number of models and datasets on which our method is tested. Furthermore, there are specific scenarios where the tested datasets do not cover, for example, controlling for the relatedness between sequentially edited facts.

As our work only looks at sequential model editing, we do not foresee or anticipate any specific ethical considerations to be acknowledged. However, like all model-editing methods, there are scenarios where such techniques could be used to apply undesirable knowledge or traits within models, which can be worth future discussion. We only use AI for grammar checking and polish the writing.

References

Anthropic. 2023. Model card: Claude 3. [Online]. Available: https://www.anthropic.com/model-card-claude-3.

Nicola De Cao, Wilker Aziz, and Ivan Titov. 2021. Editing factual knowledge in language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 6491–6506. Association for Computational Linguistics.

Canyu Chen, Baixiang Huang, Zekun Li, Zhaorun Chen, Shiyang Lai, Xiongxiao Xu, Jia-Chen Gu, Jindong Gu, Huaxiu Yao, Chaowei Xiao, Xifeng Yan, William Yang Wang, Philip Torr, Dawn Song, and Kai Shu. 2024a. Can editing llms inject harm? *Preprint*, arXiv:2407.20224.

Ruizhe Chen, Yichen Li, Zikai Xiao, and Zuozhu Liu. 2024b. Large language model bias mitigation from the perspective of knowledge editing. In *ICLR 2024 Workshop on Secure and Trustworthy Large Language Models*.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit S. Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, Luke Marris, Sam Petulla, Colin Gaffney, Asaf Aharoni, Nathan Lintz, Tiago Cardal Pais, Henrik Jacobsson, Idan Szpektor, Nan-Jiang Jiang, and 81 others. 2025. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *CoRR*, arXiv:2507.06261.

Chandler Davis and William M. Kahan. 1970. The rotation of eigenvectors by a perturbation. III. *SIAM Journal on Numerical Analysis*, 7(1):1–46.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, and 81 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *CoRR*, arXiv:2501.12948.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, and 82 others. 2024. The llama 3 herd of models. *CoRR*, arXiv:2407.21783.

Nelson Elhage, Tristan Hume, Catherine Olsson, Neel Nanda, Tom Henighan, Scott Johnston, Sheer ElShowk, Nicholas Joseph, Nova DasSarma, Ben Mann, Danny Hernandez, Amanda Askell, Kamal Ndousse, Andy Jones, Dawn Drain, Anna Chen, Yuntao Bai, Deep Ganguli, Liane Lovitt,

and 14 others. 2022. Softmax linear units. *Transformer Circuits Thread*. Https://transformer-circuits.pub/2022/solu/index.html.

Junfeng Fang, Houcheng Jiang, Kun Wang, Yunshan Ma, Jie Shi, Xiang Wang, Xiangnan He, and Tat-Seng Chua. 2025. Alphaedit: Null-space constrained knowledge editing for language models. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. 2021. Transformer feed-forward layers are key-value memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 5484–5495. Association for Computational Linguistics.

Akshat Gupta, Anurag Rao, and Gopala Anumanchipalli. 2024. Model editing at scale leads to gradual and catastrophic forgetting. In *Findings of the Association for Computational Linguistics, ACL 2024, Bangkok, Thailand and virtual meeting, August 11-16, 2024*, pages 15202–15232. Association for Computational Linguistics.

Louis Guttman. 1946. Enlargement methods for computing the inverse matrix. *The annals of mathematical statistics*, pages 336–343.

William W Hager. 1989. Updating the inverse of a matrix. *SIAM review*, 31(2):221–239.

Nicholas J Higham. 2002. *Accuracy and stability of numerical algorithms*. SIAM.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526.

Serge Lang. 2012. *Introduction to linear algebra*. Springer Science & Business Media.

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

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2022. Locating and editing factual associations in GPT. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.

Kevin Meng, Arnab Sen Sharma, Alex J. Andonian, Yonatan Belinkov, and David Bau. 2023. Mass-editing memory in a transformer. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Eric Mitchell, Charles Lin, Antoine Bosselut, Chelsea Finn, and Christopher D. Manning. 2022. Fast model editing at scale. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

OpenAI. 2023. GPT-4 technical report. *CoRR*, arXiv:2303.08774.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

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

Wei Sun, Tingyu Qu, Mingxiao Li, Jesse Davis, and Marie-Francine Moens. 2025. Mitigating negative interference in multilingual knowledge editing through null-space constraints. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 8796–8810, Vienna, Austria. Association for Computational Linguistics.

Ben Wang and Aran Komatsuzaki. 2021. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. `https://github.com/kingoflolz/mesh-transformer-jax`.

Ke Wang, Yiming Qin, Nikolaos Dimitriadis, Alessandro Favero, and Pascal Frossard. 2025a. MEMOIR: lifelong model editing with minimal overwrite and informed retention for llms. *CoRR*, abs/2506.07899.

Peng Wang, Ningyu Zhang, Bozhong Tian, Zekun Xi, Yunzhi Yao, Ziwen Xu, Mengru Wang, Shengyu Mao, Xiaohan Wang, Siyuan Cheng, Kangwei Liu, Yuansheng Ni, Guozhou Zheng, and Huajun Chen. 2024. EasyEdit: An easy-to-use knowledge editing framework for large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pages 82–93, Bangkok, Thailand. Association for Computational Linguistics.

Shipeng Wang, Xiaorong Li, Jian Sun, and Zongben Xu. 2021. Training networks in null space of feature covariance for continual learning. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pages 184–193. Computer Vision Foundation / IEEE.

Song Wang, Yaochen Zhu, Haochen Liu, Zaiyi Zheng, Chen Chen, and Jundong Li. 2025b. Knowledge editing for large language models: A survey. *ACM Comput. Surv.*, 57(3):59:1–59:37.

Per-Åke Wedin. 1972. Perturbation bounds in connection with singular value decomposition. *BIT Numerical Mathematics*, 12(1):99–111.

Max A Woodbury. 1950. *Inverting modified matrices*. Department of Statistics, Princeton University.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 40 others. 2025a. Qwen3 technical report. *CoRR*, arXiv:2505.09388.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, and 22 others. 2024a. Qwen2.5 technical report. *CoRR*, arXiv:2412.15115.

Wanli Yang, Fei Sun, Xinyu Ma, Xun Liu, Dawei Yin, and Xueqi Cheng. 2024b. The butterfly effect of model editing: Few edits can trigger large language models collapse. In *Findings of the Association for Computational Linguistics, ACL 2024, Bangkok, Thailand and virtual meeting, August 11-16, 2024*, pages 5419–5437. Association for Computational Linguistics.

Wanli Yang, Fei Sun, Jiajun Tan, Xinyu Ma, Qi Cao, Dawei Yin, Huawei Shen, and Xueqi Cheng. 2025b. The mirage of model editing: Revisiting evaluation in the wild. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, pages 15336–15354. Association for Computational Linguistics.

Yunzhi Yao, Peng Wang, Bozhong Tian, Siyuan Cheng, Zhoubo Li, Shumin Deng, Huajun Chen, and Ningyu Zhang. 2023. Editing large language models: Problems, methods, and opportunities. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 10222–10240. Association for Computational Linguistics.

A Proofs

A.1 Proof of Theorem 4.1 (Exact equivalence without truncation)

Proposition A.1. *Let $\boldsymbol{K}_0 \in \mathbb{R}^{d\times m}$ and define*

$$\boldsymbol{P}_0 = \boldsymbol{I}_d - \boldsymbol{K}_0 \left(\boldsymbol{K}_0^\top \boldsymbol{K}_0\right)^{-1} \boldsymbol{K}_0^\top.$$

Then $\boldsymbol{P}_0$ is the orthogonal projector onto $\mathrm{Range}(\boldsymbol{K}_0)^\perp$ and $\mathrm{Range}(\boldsymbol{P}_0) = \mathrm{Range}(\boldsymbol{K}_0)^\perp$.

Proof. Set $\boldsymbol{M} = (\boldsymbol{K}_0^\top \boldsymbol{K}_0)^{-1}$. Since $\boldsymbol{K}_0^\top \boldsymbol{K}_0$ is symmetric, so is $\boldsymbol{M}$. Then it follows,

$$\left(\boldsymbol{K}_0 \boldsymbol{M} \boldsymbol{K}_0^\top\right)^2 = \boldsymbol{K}_0 \boldsymbol{M} \left(\boldsymbol{K}_0^\top \boldsymbol{K}_0\right) \boldsymbol{M} \boldsymbol{K}_0^\top = \boldsymbol{K}_0 \boldsymbol{M} \boldsymbol{K}_0^\top,$$

and $\left(\boldsymbol{K}_0 \boldsymbol{M} \boldsymbol{K}_0^\top\right)^\top = \boldsymbol{K}_0 \boldsymbol{M} \boldsymbol{K}_0^\top$, so $\boldsymbol{K}_0 \boldsymbol{M} \boldsymbol{K}_0^\top$ is the orthogonal projector onto $\mathrm{Range}(\boldsymbol{K}_0)$. Therefore $\boldsymbol{P}_0 = \boldsymbol{I}_d - \boldsymbol{K}_0 \boldsymbol{M} \boldsymbol{K}_0^\top$ is symmetric and idempotent.
To identify its range, note that for any $\boldsymbol{x}$,

$$\boldsymbol{K}_0^\top \boldsymbol{P}_0 \boldsymbol{x} = \boldsymbol{K}_0^\top \boldsymbol{x} - (\boldsymbol{K}_0^\top \boldsymbol{K}_0) \boldsymbol{M} \boldsymbol{K}_0^\top \boldsymbol{x} = \boldsymbol{K}_0^\top \boldsymbol{x} - \boldsymbol{I}_d (\boldsymbol{K}_0^\top \boldsymbol{x}) = 0,$$

so $\mathrm{Range}(\boldsymbol{P}_0) \subseteq \mathrm{Null}\left(\boldsymbol{K}_0^\top\right)$. Conversely, if $\boldsymbol{y} \in \mathrm{Null}\left(\boldsymbol{K}_0^\top\right)$ then $\boldsymbol{P}_0 \boldsymbol{y} = \boldsymbol{y}$, hence $\boldsymbol{y} \in \mathrm{Range}(\boldsymbol{P}_0)$. Thus

$$\mathrm{Range}(\boldsymbol{P}_0) = \mathrm{Null}\left(\boldsymbol{K}_0^\top\right) = \mathrm{Range}(\boldsymbol{K}_0)^\perp.$$

□

Proposition A.2. *If $\boldsymbol{P}_{j-1}$ is idempotent and $\boldsymbol{Q}_j$ is an orthonormal basis of $\mathrm{Range}(\boldsymbol{P}_{j-1}\boldsymbol{K}_j)$, then $\boldsymbol{P}_{j-1}\boldsymbol{Q}_j = \boldsymbol{Q}_j$ and $\boldsymbol{Q}_j^\top \boldsymbol{P}_{j-1} = \boldsymbol{Q}_j^\top$.*

Proof. Each column $\boldsymbol{q}$ of $\boldsymbol{Q}_j$ lies in $\mathrm{Range}(\boldsymbol{R}_j) \subseteq \mathrm{range}(\boldsymbol{P}_{j-1})$. Hence $\boldsymbol{q} = \boldsymbol{P}_{j-1}\boldsymbol{z}$ for some $\boldsymbol{z}$. Then

$$\boldsymbol{P}_{j-1}\boldsymbol{q} = \boldsymbol{P}_{j-1}(\boldsymbol{P}_{j-1}\boldsymbol{z}) = \boldsymbol{P}_{j-1}^2\boldsymbol{z} = \boldsymbol{P}_{j-1}\boldsymbol{z} = \boldsymbol{q},$$

since $\boldsymbol{P}_{j-1}$ is idempotent. This proves $\boldsymbol{P}_{j-1}\boldsymbol{Q}_j = \boldsymbol{Q}_j$. The second identity follows by transposition. □

Proposition A.3. *Let $\boldsymbol{P}_{j-1}$ and $\boldsymbol{Q}_j$ be defined as in Proposition A.2. Then $\boldsymbol{P}_j \triangleq \boldsymbol{P}_{j-1} - \boldsymbol{Q}_j\boldsymbol{Q}_j^\top$ is also an orthogonal projector.*

Proof. It is obvious that $\boldsymbol{P}_j$ is symmetric. We just need to show it is idempotent.

$$\begin{aligned}\boldsymbol{P}_j^2 &= \left(\boldsymbol{P}_{j-1} - \boldsymbol{Q}_j\boldsymbol{Q}_j^\top\right)^2 \\ &= \boldsymbol{P}_{j-1}^2 - \boldsymbol{P}_{j-1}\boldsymbol{Q}_j\boldsymbol{Q}_j^\top - \boldsymbol{Q}_j\boldsymbol{Q}_j^\top\boldsymbol{P}_{j-1} + \boldsymbol{Q}_j\boldsymbol{Q}_j^\top\boldsymbol{Q}_j\boldsymbol{Q}_j^\top.\end{aligned}$$

Using $\boldsymbol{P}_{j-1}^2 = \boldsymbol{P}_{j-1}$, $\boldsymbol{P}_{j-1}\boldsymbol{Q}_j = \boldsymbol{Q}_j$, $\boldsymbol{Q}_j^\top\boldsymbol{P}_{j-1} = \boldsymbol{Q}_j^\top$ (by Proposition A.2.), and $\boldsymbol{Q}_j^\top\boldsymbol{Q}_j = \boldsymbol{I}$, we obtain

$$\boldsymbol{P}_j^2 = \boldsymbol{P}_{j-1} - \boldsymbol{Q}_j\boldsymbol{Q}_j^\top = \boldsymbol{P}_j.$$

Thus $\boldsymbol{P}_j$ is symmetric and idempotent, hence an orthogonal projector. □

Lemma A.4 (Projector difference for nested subspaces)**.** *Let $S, U \subseteq \mathbb{R}^d$ be linear subspaces with $\boldsymbol{U} \subseteq S$. Let $\boldsymbol{P}_S$ and $\boldsymbol{P}_U$ be the orthogonal projectors onto S and $\boldsymbol{U}$, respectively. Then*

$$\boldsymbol{P} \coloneqq \boldsymbol{P}_S - \boldsymbol{P}_U$$

is the orthogonal projector onto the subspace $S \cap \boldsymbol{U}^\perp$; in particular,

$$\mathrm{range}(\boldsymbol{P}) = S \cap \boldsymbol{U}^\perp \quad \textit{and} \quad \mathrm{Null}(\boldsymbol{P}) = \boldsymbol{U} \oplus S^\perp.$$

Proof. Since $\boldsymbol{U} \subseteq S$, $\boldsymbol{P}_S\boldsymbol{P}_U = \boldsymbol{P}_U\boldsymbol{P}_S = \boldsymbol{P}_U$. Hence

$$\boldsymbol{P}^\top = (\boldsymbol{P}_S - \boldsymbol{P}_U)^\top = \boldsymbol{P}_S - \boldsymbol{P}_U = \boldsymbol{P}, \qquad \boldsymbol{P}^2 = (\boldsymbol{P}_S - \boldsymbol{P}_U)^2 = \boldsymbol{P}_S - \boldsymbol{P}_U.$$

Thus $\boldsymbol{P}$ is an orthogonal projector. For any x, write $x = s + s_\perp$ with $s = \boldsymbol{P}_S x \in S$ and $s_\perp \in S^\perp$, and then decompose $s = \boldsymbol{U} + w$ with $\boldsymbol{U} = \boldsymbol{P}_U s \in \boldsymbol{U}$ and $w \in S \cap \boldsymbol{U}^\perp$. We get

$$\boldsymbol{P}x = (\boldsymbol{P}_S - \boldsymbol{P}_U)(s + s_\perp) = s - u = w \in S \cap \boldsymbol{U}^\perp,$$

so $\mathrm{range}(\boldsymbol{P}) \subseteq S \cap \boldsymbol{U}^\perp$. Conversely, for any $w \in S \cap \boldsymbol{U}^\perp$, $(\boldsymbol{P}_S - \boldsymbol{P}_U)w = w$, hence $w \in \mathrm{range}(\boldsymbol{P})$ and $\mathrm{range}(\boldsymbol{P}) = S \cap \boldsymbol{U}^\perp$. Taking orthogonal complements yields $\mathrm{Null}(\boldsymbol{P}) = \boldsymbol{U} \oplus S^\perp$. □

Corollary A.5 (Instantiation for our update)**.** *Let* $S \coloneqq \mathrm{range}(\boldsymbol{P}_{t-1})$, $\boldsymbol{U} \coloneqq \mathrm{span}(\boldsymbol{R}_t) = \mathrm{Range}(\boldsymbol{Q}_t)$ *with* $\boldsymbol{R}_t = \boldsymbol{P}_{t-1}\boldsymbol{K}_t$ *and* $\mathrm{Range}(\boldsymbol{Q}_t) = \mathrm{Range}(\boldsymbol{R}_t) \subseteq S$*. Then, for* $\boldsymbol{P}_t \coloneqq \boldsymbol{P}_{t-1} - \boldsymbol{Q}_t\boldsymbol{Q}_t^\top$*,*

$$\mathrm{range}(\boldsymbol{P}_t)) = \mathrm{range}(\boldsymbol{P}_{t-1}) \cap \mathrm{span}(\boldsymbol{R}_t)^\perp\,, \qquad \mathrm{Null}(\boldsymbol{P}_t)) = \mathrm{Null}(\boldsymbol{P}_{t-1}) \,\oplus\, \mathrm{span}(\boldsymbol{R}_t)\,.$$

Proof. Apply Lemma A.4 with $\boldsymbol{P}_S = \boldsymbol{P}_{t-1}$ and $\boldsymbol{P}_U = \boldsymbol{Q}_t\boldsymbol{Q}_t^\top$. □

Theorem A.6 (Exact equivalence without truncation)**.** *If no truncation is applied when forming* $\boldsymbol{Q}_j$*, then for all* $t \geq 0$*,*

$$\mathrm{Null}(\boldsymbol{P}_t) = \mathrm{span}(\boldsymbol{K}_0, \ldots, \boldsymbol{K}_t)\,, \qquad \boldsymbol{P}_t = \boldsymbol{P}_t^\star.$$

Proof. Base case ($t = 0$). By Proposition A.1, $\boldsymbol{P}_0$ is the orthogonal projector onto $\mathrm{span}(\boldsymbol{K}_0)^\perp$, hence $\mathrm{Null}(\boldsymbol{P}_0) = \mathrm{span}(\boldsymbol{K}_0)$.

Induction step. Assume $\mathrm{Null}(\boldsymbol{P}_{t-1}) = \mathrm{span}(\boldsymbol{K}_0, \ldots, \boldsymbol{K}_{t-1})$ and assume $\boldsymbol{P}_{t-1}$ is an orthogonal projector. Write

$$\boldsymbol{K}_t = (\boldsymbol{I} - \boldsymbol{P}_{t-1})\,\boldsymbol{K}_t + \boldsymbol{P}_{t-1}\boldsymbol{K}_t \;=:\; \boldsymbol{B}_t + \boldsymbol{R}_t,$$

so that $\mathrm{Range}(\boldsymbol{B}_t) \subseteq \mathrm{Null}(\boldsymbol{P}_{t-1})$ and $\mathrm{Range}(\boldsymbol{R}_t) \subseteq \mathrm{range}(\boldsymbol{P}_{t-1})$. Since $\mathrm{Null}(\boldsymbol{P}_{t-1}) \perp \mathrm{range}(\boldsymbol{P}_{t-1})$, we have the orthogonal direct sum

$$\begin{aligned}\mathrm{span}(\boldsymbol{K}_0, \ldots, \boldsymbol{K}_t) &= \mathrm{span}(\boldsymbol{K}_0, \ldots, \boldsymbol{K}_{t-1},\, \boldsymbol{B}_t + \boldsymbol{R}_t) \\ &= \mathrm{span}(\boldsymbol{K}_0, \ldots, \boldsymbol{K}_{t-1}) \,\oplus\, \mathrm{span}(\boldsymbol{R}_t) \\ &= \mathrm{Null}(\boldsymbol{P}_{t-1}) \,\oplus\, \mathrm{span}(\boldsymbol{R}_t)\,.\end{aligned}$$

On the other hand, since both $\boldsymbol{P}_{t-1}$ and $\boldsymbol{Q}_t\boldsymbol{Q}_t^\top$ are orthogonal projectors, by Proposition A.3 and Lemma A.4, $\boldsymbol{P}_t = \boldsymbol{P}_{t-1} - \boldsymbol{Q}_t\boldsymbol{Q}_t^\top$ is also an orthogonal projector with $\mathrm{Range}(\boldsymbol{Q}_t) = \mathrm{span}(\boldsymbol{R}_t) \subseteq \mathrm{range}(\boldsymbol{P}_{t-1})$. Hence, by Corollary A.5,

$$\begin{aligned}\mathrm{Null}(\boldsymbol{P}_t) &= \mathrm{Null}(\boldsymbol{P}_{t-1}) \,\oplus\, \mathrm{span}(\boldsymbol{R}_t) \\ &= \mathrm{span}(\boldsymbol{K}_0, \ldots, \boldsymbol{K}_t)\,.\end{aligned}$$

Finally, since $\boldsymbol{P}_t$ is an orthogonal projector and its null space equals $\mathrm{span}(\boldsymbol{K}_0, \ldots, \boldsymbol{K}_t)$, it must coincide with the unique projector onto $\mathrm{span}(\boldsymbol{K}_0, \ldots, \boldsymbol{K}_t)^\perp$, i.e., $\boldsymbol{P}_t = \boldsymbol{P}_t^\star$. □

A.2 Proof of Theorem 4.2 (Global error bound with truncation)

We first give a lemma that separates the retained-part (geometric) deviation from the discarded-part (tail-energy) error. This avoids comparing subspaces of different dimensions by only applying $\sin\Theta$ to q-dimensional subspaces.

Notation A superscript * denotes the *ideal, no-truncation* quantity. In particular, $\boldsymbol{P}_{j-1}^*$ is the orthogonal projector onto $\mathrm{span}(\boldsymbol{K}_0, \ldots, \boldsymbol{K}_{j-1})^\perp$, and $\boldsymbol{R}_j^* = \boldsymbol{P}_{j-1}^*\boldsymbol{K}_j$. By contrast, $\boldsymbol{P}_{j-1}$ is the algorithmic projector after $j-1$ steps with truncation, and $\boldsymbol{R}_j = \boldsymbol{P}_{j-1}\boldsymbol{K}_j = \boldsymbol{R}_j^* + \boldsymbol{E}_j$ with $\boldsymbol{E}_j = \left(\boldsymbol{P}_{j-1} - \boldsymbol{P}_{j-1}^*\right)\boldsymbol{K}_j$. For two subspaces $\mathcal{U}, \mathcal{V} \subset \mathbb{R}^d$ with orthonormal bases U, V and orthogonal projectors $P_\mathcal{U} = UU^\top$, $P_\mathcal{V} = VV^\top$,

let $\Theta(\mathcal{U},\mathcal{V}) = (\theta_1,\dots,\theta_q)$ denote the vector of principal angles (with $q = \min\{\dim\mathcal{U}, \dim\mathcal{V}\}$). We write

$$\|\sin\Theta(\mathcal{U},\mathcal{V})\| \;:=\; \max_i \sin\theta_i,$$

and recall the identity

$$\|P_{\mathcal{U}} - P_{\mathcal{V}}\|_2 \;=\; \|\sin\Theta(\mathcal{U},\mathcal{V})\|. \tag{12}$$

Lemma A.7. *(Refined, gap–tail form) Let $\boldsymbol{R}_j^* = \boldsymbol{P}_{j-1}^*\boldsymbol{K}_j$ with SVD $\boldsymbol{R}_j^* = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top$, where the nonzero singular values are $\sigma_1 \ge \cdots \ge \sigma_r > 0$ and $\boldsymbol{U} = [\boldsymbol{U}_1\ \boldsymbol{U}_2]$ with $\boldsymbol{U}_1 \in \mathbb{R}^{d\times q}$. Given a threshold $\tau_j > 0$, let $q = \max\{i : \sigma_i \ge \tau_j\}$, and construct $\widehat{\boldsymbol{U}}_1$ by applying the same truncation rule to $\boldsymbol{R}_j = \boldsymbol{P}_{j-1}\boldsymbol{K}_j = \boldsymbol{R}_j^* + \boldsymbol{E}_j$, where $\boldsymbol{E}_j = (\boldsymbol{P}_{j-1} - \boldsymbol{P}_{j-1}^*)\boldsymbol{K}_j$. Define the projectors $\boldsymbol{P}_1 = \boldsymbol{U}_1\boldsymbol{U}_1^\top$ and $\widehat{\boldsymbol{P}}_1 = \widehat{\boldsymbol{U}}_1\widehat{\boldsymbol{U}}_1^\top$, and set $\boldsymbol{P}^\star = \boldsymbol{U}\boldsymbol{U}^\top$.*

*(i) (**No truncation**) If $\tau_j \le \sigma_r$ then $q = r$, $\boldsymbol{U}_1 = \boldsymbol{U}$, and $\widehat{\boldsymbol{U}}_1$ has the same dimension; hence $\sin\Theta\left(\mathrm{span}(\boldsymbol{U}_1), \mathrm{span}\left(\widehat{\boldsymbol{U}}_1\right)\right) = 0$ whenever $\boldsymbol{E}_j = 0$, and in general*

$$\left\|\sin\Theta\left(\mathrm{span}(\boldsymbol{U}_1), \mathrm{span}\left(\widehat{\boldsymbol{U}}_1\right)\right)\right\| \;\le\; \frac{\|\boldsymbol{E}_j\|_2}{\sigma_r}.$$

*(ii) (**With truncation**) If $\tau_j \in (\sigma_{q+1}, \sigma_q)$ for some $q < r$, let the spectral gap be $\gamma_j = \sigma_q - \sigma_{q+1} > 0$. Then*

$$\left\|\sin\Theta\left(\mathrm{span}(\boldsymbol{U}_1), \mathrm{span}\left(\widehat{\boldsymbol{U}}_1\right)\right)\right\| \;\le\; \frac{\|\boldsymbol{E}_j\|_2}{\gamma_j}, \tag{13}$$

$$\left\|(\boldsymbol{P}_1 - \boldsymbol{P}^\star)\ \boldsymbol{R}_j^*\right\|_F^2 \;=\; \sum_{i>q}\sigma_i^2. \tag{14}$$

Proof. The bound Equation (13) is a standard Davis–Kahan–Wedin type result for q-dimensional left singular subspaces under additive perturbation E_j, with denominator given by the gap between the q-th and $(q{+}1)$-th singular values of $\boldsymbol{R}_j^*$. For Equation (14), write

$$(\boldsymbol{P}_1 - \boldsymbol{P}^\star)\,\boldsymbol{R}_j^* = (\boldsymbol{U}_1\boldsymbol{U}_1^\top - \boldsymbol{U}\boldsymbol{U}^\top)\,\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top = -\boldsymbol{U}_2\boldsymbol{\Sigma}_2\boldsymbol{V}^\top,$$

where $\boldsymbol{\Sigma}_2 = \mathrm{diag}(\sigma_{q+1},\dots,\sigma_r)$. Taking Frobenius norms yields $\left\|(\boldsymbol{P}_1 - \boldsymbol{P}^\star)\ \boldsymbol{R}_j^*\right\|_F^2 = \|\boldsymbol{\Sigma}_2\|_F^2 = \sum_{i>q}\sigma_i^2$. □

We next prove the theorem by using the lemma. We bound the one-step *action* of the projector error on the signal $\boldsymbol{R}_j^*$. Using $\widetilde{\boldsymbol{P}}_j = \boldsymbol{P}_{j-1} - \widehat{\boldsymbol{P}}_1$ and $\boldsymbol{P}_j^* = \boldsymbol{P}_{j-1}^* - \boldsymbol{P}^*$, we have the exact decomposition

$$\widetilde{\boldsymbol{P}}_j - \boldsymbol{P}_j^* = (\boldsymbol{P}_{j-1} - \boldsymbol{P}_{j-1}^*) \;+\; (\boldsymbol{P}^* - \boldsymbol{P}_1) \;+\; \left(\boldsymbol{P}_1 - \widehat{\boldsymbol{P}}_1\right).$$

Hence, by the triangle inequality and sub-multiplicativity of operator/Frobenius norms,

$$\left\|\left(\widetilde{\boldsymbol{P}}_j - \boldsymbol{P}_j^*\right)\ \boldsymbol{R}_j^*\right\|_F \;\le\; \underbrace{\left\|(\boldsymbol{P}_{j-1} - \boldsymbol{P}_{j-1}^*)\ \boldsymbol{R}_j^*\right\|_F}_{\text{carry-over}} + \underbrace{\left\|(\boldsymbol{P}^* - \boldsymbol{P}_1)\ \boldsymbol{R}_j^*\right\|_F}_{\text{tail}} + \underbrace{\left\|\left(\boldsymbol{P}_1 - \widehat{\boldsymbol{P}}_1\right)\ \boldsymbol{R}_j^*\right\|_F}_{\text{geometry}}. \tag{15}$$

For the last two terms, by equation (12) and Equation (13),

$$\|(\boldsymbol{P}_1 - \widehat{\boldsymbol{P}}_1)\,\boldsymbol{R}_j^*\|_F \;\le\; \|\boldsymbol{P}_1 - \widehat{\boldsymbol{P}}_1\|_2\,\|\boldsymbol{R}_j^*\|_F \;=\; \|\sin\Theta(\mathrm{span}(\boldsymbol{U}_1), \mathrm{span}(\widehat{\boldsymbol{U}}_1))\|\,\|\boldsymbol{R}_j^*\|_F \;\le\; \frac{\|\boldsymbol{E}_j\|_2}{\gamma_j}\,\|\boldsymbol{R}_j^*\|_F.$$

Therefore,

$$\left\|\left(\widetilde{\boldsymbol{P}}_j - \boldsymbol{P}_j^*\right)\ \boldsymbol{R}_j^*\right\|_F \;\le\; \left\|(\boldsymbol{P}_{j-1} - \boldsymbol{P}_{j-1}^*)\ \boldsymbol{R}_j^*\right\|_F \;+\; \frac{\|\boldsymbol{E}_j\|_2}{\gamma_j}\,\left\|\boldsymbol{R}_j^*\right\|_F \;+\; \|\boldsymbol{\Sigma}_2\|_F. \tag{16}$$

Absorbing the carry-over term. By the induction hypothesis at step $j-1$ we already have $\|\boldsymbol{P}_{j-1} - \boldsymbol{P}^*_{j-1}\|_2 \le B_{j-1}$, where B_{j-1} denotes the cumulative gap–tail bound up to step $j-1$. Hence

$$\left\|\left(\boldsymbol{P}_{j-1} - \boldsymbol{P}^*_{j-1}\right)\, \boldsymbol{R}^*_j\right\|_F \;\le\; \|\boldsymbol{P}_{j-1} - \boldsymbol{P}^*_{j-1}\|_2\, \|\boldsymbol{R}^*_j\|_F \;\le\; B_{j-1}\, \|\boldsymbol{R}^*_j\|_F.$$

where $B_{j-1} = \frac{1}{\sigma_{\min}\left(\boldsymbol{C}_{j-1}\right)} \sum_{i=1}^{j-1} \left(\frac{\|\boldsymbol{E}_i\|_2}{\gamma_i}\, \|\boldsymbol{R}^*_i\|_F \;+\; \|\boldsymbol{\Sigma}_{2,i}\|_F \right)$. By induction on j (unwinding the same bound for steps $1, \dots, j-1$), this carry-over factor is controlled by the already accumulated gap–tail terms up to step $j-1$. For exposition we subsume it into the geometric factor and keep the displayed right-hand side in the simple "gap–tail" form used below; the full three-term recurrence Equation (16) only strengthens the final bound.

Using the above, we obtain the one-step *gap–tail* bound

$$\left\|\left(\widetilde{\boldsymbol{P}}_j - \boldsymbol{P}^*_j\right)\, \boldsymbol{R}^*_j\right\|_F \;\le\; \frac{\|\boldsymbol{E}_j\|_2}{\gamma_j}\, \left\|\boldsymbol{R}^*_j\right\|_F \;+\; \|\boldsymbol{\Sigma}_2\|_F\,. \tag{17}$$

We now pass to the global bound. Let $\boldsymbol{C}_t = [\boldsymbol{R}^*_1, \dots, \boldsymbol{R}^*_t] \in \mathbb{R}^{d \times M}$, and recall $\mathcal{C}_t = \operatorname{span}(\boldsymbol{C}_t)$. Define the *restricted operator norm* on $\mathcal{C}_t$:

$$\|\boldsymbol{A}\|^{(\mathcal{C}_t)}_{2\to 2} \coloneqq \sup_{x \in \mathcal{C}_t,\, x \neq 0} \frac{\|\boldsymbol{A}\boldsymbol{x}\|_2}{\|\boldsymbol{x}\|_2}.$$

Using induction on t and the fact that at step j the update only acts through the rank-q_j projector change on $\boldsymbol{R}^*_j$, we obtain

$$\begin{aligned}\left\|\left(\widetilde{\boldsymbol{P}}_t - \boldsymbol{P}^*_t\right)\, \boldsymbol{C}_t\right\|_F \;&\le\; \sum_{j=1}^{t} \left\|\left(\widetilde{\boldsymbol{P}}_j - \boldsymbol{P}^*_j\right)\, \boldsymbol{R}^*_j\right\|_F \\ &\le\; \sum_{j=1}^{t} \left(\frac{\|\boldsymbol{E}_j\|_2}{\gamma_j}\, \left\|\boldsymbol{R}^*_j\right\|_F \;+\; \|\boldsymbol{\Sigma}_{2,j}\|_F \right), \end{aligned} \tag{18}$$

where $\boldsymbol{\Sigma}_{2,j}$ is the discarded block at step j.

Finally, for any $\boldsymbol{x} \in \mathcal{C}_t$ write $\boldsymbol{x} = \boldsymbol{C}_t \boldsymbol{\alpha}$ with $\boldsymbol{\alpha} \in \mathbb{R}^M$. Then

$$\left\|\left(\widetilde{\boldsymbol{P}}_t - \boldsymbol{P}^*_t\right)\, \boldsymbol{x}\right\|_2 \;\le\; \frac{\left\|\left(\widetilde{\boldsymbol{P}}_t - \boldsymbol{P}^*_t\right)\, \boldsymbol{C}_t\right\|_F}{\sigma_{\min}(\boldsymbol{C}_t)}\, \|\boldsymbol{x}\|_2\,,$$

where $\sigma_{\min}(\boldsymbol{C}_t)$ is the smallest nonzero singular value of $\boldsymbol{C}_t$. Hence

$$\|\widetilde{\boldsymbol{P}}_t - \boldsymbol{P}^*_t\|^{(\mathcal{C}_t)}_{2\to 2} \;\le\; \frac{1}{\sigma_{\min}(\boldsymbol{C}_t)} \sum_{j=1}^{t} \left(\frac{\|\boldsymbol{E}_j\|_2}{\gamma_j}\, \|\boldsymbol{R}^*_j\|_F \;+\; \|\boldsymbol{\Sigma}_{2,j}\|_F \right). \tag{19}$$

Since both $\widetilde{\boldsymbol{P}}_t$ and $\boldsymbol{P}^*_t$ are orthogonal projectors, their unrestricted spectral-norm difference is at most 1, so we may write the final global bound as

$$\|\widetilde{\boldsymbol{P}}_t - \boldsymbol{P}^*_t\|_2 \;\le\; \min\left\{ 1,\; \frac{1}{\sigma_{\min}(\boldsymbol{C}_t)} \sum_{j=1}^{t} \left(\frac{\|\boldsymbol{E}_j\|_2}{\gamma_j}\, \|\boldsymbol{R}^*_j\|_F + \|\boldsymbol{\Sigma}_{2,j}\|_F \right) \right\}.$$

This completes the proof of Theorem 4.2.

A.3 Proof of Corollary A.8 (Interference bound)

Notation recalled. For $t \geq 1$, define the (block) matrix

$$\boldsymbol{C}_t \;:=\; \big[\boldsymbol{R}_1^\star,\; \boldsymbol{R}_2^\star,\; \ldots,\; \boldsymbol{R}_t^\star\big] \in \mathbb{R}^{d\times M}, \quad \text{so that} \quad \operatorname{span}(\boldsymbol{C}_t) = \operatorname{span}\big(\boldsymbol{R}_1^\star, \ldots, \boldsymbol{R}_t^\star\big).$$

The ideal projector $\boldsymbol{P}_t^\star$ is the orthogonal projector onto the orthogonal complement of $\operatorname{span}(\boldsymbol{C}_t)$, i.e.,

$$\boldsymbol{P}_t^\star \;=\; \operatorname{span}(\boldsymbol{C}_t)^\perp \quad\Longrightarrow\quad \boldsymbol{P}_t^\star x \;=\; 0, \quad \forall\, x \in \operatorname{span}(\boldsymbol{C}_t).$$

Let $\widetilde{\boldsymbol{P}}_t$ be the truncated projector constructed in the main text. For matrices, $\|\cdot\|_2$ denotes the spectral norm; for vectors, $\|\cdot\|$ is the Euclidean norm.

Corollary A.8 (Interference bound)**.** *Let $\boldsymbol{\Delta} \in \mathbb{R}^{d\times d}$ be any (future) linear edit with $\|\boldsymbol{\Delta}\|_2 \leq \Gamma$. Then for every $\boldsymbol{x} \in \operatorname{span}(\boldsymbol{C}_t)$,*

$$\big\|\boldsymbol{\Delta}\,\widetilde{\boldsymbol{P}}_t\,\boldsymbol{x}\big\| \;\leq\; \Gamma\,\big\|\widetilde{\boldsymbol{P}}_t - \boldsymbol{P}_t^\star\big\|_2\,\|\boldsymbol{x}\|.$$

In particular, the worst-case interference on $\operatorname{span}(\boldsymbol{C}_t)$ is controlled by the projector approximation error $\|\widetilde{\boldsymbol{P}}_t - \boldsymbol{P}_t^\star\|_2$.

Proof. Fix any $\boldsymbol{x} \in \operatorname{span}(\boldsymbol{C}_t)$. By definition of $\boldsymbol{P}_t^\star$, we have $\boldsymbol{P}_t^\star \boldsymbol{x} = 0$. Therefore

$$\widetilde{\boldsymbol{P}}_t\,\boldsymbol{x} \;=\; \big(\widetilde{\boldsymbol{P}}_t - \boldsymbol{P}_t^\star\big)\boldsymbol{x}.$$

Applying $\boldsymbol{\Delta}$ and taking norms, we use submultiplicativity of the operator norm:

$$\big\|\boldsymbol{\Delta}\,\widetilde{\boldsymbol{P}}_t\,\boldsymbol{x}\big\| \;=\; \big\|\boldsymbol{\Delta}\,(\widetilde{\boldsymbol{P}}_t - \boldsymbol{P}_t^\star)\,\boldsymbol{x}\big\| \;\leq\; \|\boldsymbol{\Delta}\|_2\,\|\widetilde{\boldsymbol{P}}_t - \boldsymbol{P}_t^\star\|_2\,\|\boldsymbol{x}\| \;\leq\; \Gamma\,\|\widetilde{\boldsymbol{P}}_t - \boldsymbol{P}_t^\star\|_2\,\|\boldsymbol{x}\|.$$

This proves the claim. □

A.4 Proof of Equation (10) (Solution with heavy matrix inversion.)

Proof. Given the aligned projector matrix $\boldsymbol{P}_{t-1}$, since it is an orthogonal projection, we have $\boldsymbol{P} = \boldsymbol{P}^\top$ and $\boldsymbol{P}^2 = \boldsymbol{P}$. The sequential editing objective:

$$L = \big(\|(\boldsymbol{W}_{t-1} + \boldsymbol{\Delta}_t\boldsymbol{P}_{t-1})\boldsymbol{K}_t - \boldsymbol{V}_t\|^2 + \|\boldsymbol{\Delta}_t\boldsymbol{P}_{t-1}\|^2\big). \tag{20}$$

We define $\boldsymbol{R} = \boldsymbol{V}_t - \boldsymbol{W}_{t-1}\boldsymbol{K}_t$, setting the matrix derivative $\nabla_{\boldsymbol{\Delta}} L = 0$ yields:

$$(\boldsymbol{\Delta}_t\boldsymbol{P}_{t-1}\boldsymbol{K}_t - \boldsymbol{R})\boldsymbol{K}_t^\top\boldsymbol{P}_{t-1}^\top + \boldsymbol{\Delta}_t\boldsymbol{P}\boldsymbol{P}^\top = \boldsymbol{0}. \tag{21}$$

Factorize $\boldsymbol{\Delta}_t\boldsymbol{P}_{t-1}$ and use $\boldsymbol{P} = \boldsymbol{P}^\top$ and $\boldsymbol{P}^2 = \boldsymbol{P}$, we obtain:

$$\boldsymbol{\Delta}_t\boldsymbol{P}_{t-1}(\boldsymbol{K}_t\boldsymbol{K}_t^\top\boldsymbol{P}_{t-1} + \boldsymbol{I}) = \boldsymbol{R}\boldsymbol{K}_t^\top\boldsymbol{P}_{t-1}. \tag{22}$$

We then get the final closed-form solution if the inversion of the bracketed term.

$$\boldsymbol{\Delta}_{\texttt{EvoEdit}} = \boldsymbol{\Delta}_t\boldsymbol{P}_{t-1} = \boldsymbol{R}\boldsymbol{K}_t^\top\boldsymbol{P}_{t-1}\big(\boldsymbol{K}_t\boldsymbol{K}_t^\top\boldsymbol{P}_{t-1} + \boldsymbol{I}\big)^{-1}. \tag{23}$$

□

A.5 Proof of Equation (11) (Solution via Woodbury-identity-matrix.)

Proof. Let

$$\boldsymbol{K}_t \in \mathbb{R}^{d\times m}, \qquad \boldsymbol{P}_{t-1} \in \mathbb{R}^{d\times d}, \qquad \boldsymbol{R} \in \mathbb{R}^{d\times r}.$$

Start from the given expression

$$\boldsymbol{\Delta}_t\boldsymbol{P}_{t-1} = \boldsymbol{R}\,\boldsymbol{K}_t^\top\boldsymbol{P}_{t-1}\Big(\boldsymbol{K}_t\boldsymbol{K}_t^\top\boldsymbol{P}_{t-1} + I_d\Big)^{-1}.$$

We use the standard matrix identity (a rearrangement closely related to Woodbury):

$$\big(\boldsymbol{I}_d + \boldsymbol{A}\boldsymbol{B}\big)^{-1}\boldsymbol{A} = \boldsymbol{A}\big(\boldsymbol{I}_m + \boldsymbol{B}\boldsymbol{A}\big)^{-1} \qquad \text{for } \boldsymbol{A} \in \mathbb{R}^{d\times m},\ \boldsymbol{B} \in \mathbb{R}^{m\times d}, \tag{24}$$

which is verified by multiplying both sides on the left by $(\boldsymbol{I}_d + \boldsymbol{A}\boldsymbol{B})$ and on the right by $(\boldsymbol{I}_m + \boldsymbol{B}\boldsymbol{A})$ (or derived from the Woodbury identity).

Take

$$\boldsymbol{A} = \boldsymbol{K}_t, \qquad \boldsymbol{B} = \boldsymbol{K}_t^\top \boldsymbol{P}_{t-1}.$$

Then $\boldsymbol{A}\boldsymbol{B} = \boldsymbol{K}_t\boldsymbol{K}_t^\top \boldsymbol{P}_{t-1}$ and $\boldsymbol{B}\boldsymbol{A} = \boldsymbol{K}_t^\top \boldsymbol{P}_{t-1}\boldsymbol{K}_t$. Applying Equation (24) gives

$$\big(\boldsymbol{I}_d + \boldsymbol{K}_t\boldsymbol{K}_t^\top \boldsymbol{P}_{t-1}\big)^{-1}\boldsymbol{K}_t = \boldsymbol{K}_t\big(\boldsymbol{I}_m + \boldsymbol{K}_t^\top \boldsymbol{P}_{t-1}\boldsymbol{K}_t\big)^{-1}.$$

Multiply the last identity on the left by $\boldsymbol{K}_t^\top \boldsymbol{P}_{t-1}$ (or equivalently take transposes and rearrange) to obtain

$$\boldsymbol{K}_t^\top \boldsymbol{P}_{t-1}\big(\boldsymbol{I}_d + \boldsymbol{K}_t\boldsymbol{K}_t^\top \boldsymbol{P}_{t-1}\big)^{-1} = \big(\boldsymbol{I}_m + \boldsymbol{K}_t^\top \boldsymbol{P}_{t-1}\boldsymbol{K}_t\big)^{-1}\boldsymbol{K}_t^\top \boldsymbol{P}_{t-1}. \tag{25}$$

Substitute Equation (25) into the right-hand side of (1):

$$\begin{aligned}
\boldsymbol{\Delta}_t\boldsymbol{P}_{t-1} &= \boldsymbol{R}\left[\boldsymbol{K}_t^\top \boldsymbol{P}_{t-1}\big(\boldsymbol{I}_d + \boldsymbol{K}_t\boldsymbol{K}_t^\top \boldsymbol{P}_{t-1}\big)^{-1}\right] \\
&= \boldsymbol{R}\left[\big(\boldsymbol{I}_m + \boldsymbol{K}_t^\top \boldsymbol{P}_{t-1}\boldsymbol{K}_t\big)^{-1}\boldsymbol{K}_t^\top \boldsymbol{P}_{t-1}\right] \\
&= \boldsymbol{R}\,(\boldsymbol{K}_t^\top \boldsymbol{P}_{t-1}\boldsymbol{K}_t + \boldsymbol{I}_m)^{-1}\boldsymbol{K}_t^\top \boldsymbol{P}_{t-1},
\end{aligned}$$

which proves the claim. □

B Detailed Complexity Derivations

B.1 Preliminaries and Notation

Let d be the hidden size of the edited layer. At edit step t, denote the current key matrix $\boldsymbol{K}_t \in \mathbb{R}^{d\times k_t}$, the stacked preserved/previous keys $\boldsymbol{K}_p \in \mathbb{R}^{d\times m_p}$, and the residual R_t. We implement the projector in operator form $\boldsymbol{P} = I - \boldsymbol{Q}\boldsymbol{Q}^\top$ with $\boldsymbol{Q} \in \mathbb{R}^{d\times r}$ (rank $r \ll d$). All complexities below are per edit; we count only the dominant FLOPs and omit lower-order terms.

B.2 `AlphaEdit` — Full Derivation and Cost

`AlphaEdit` uses the projected closed form

$$\Delta_{\text{Alpha}} \;=\; \boldsymbol{R}_t\,\boldsymbol{K}_t^\top \boldsymbol{P}\,\Big(\boldsymbol{K}_p\boldsymbol{K}_p^\top \boldsymbol{P} \;+\; \boldsymbol{K}_t\boldsymbol{K}_t^\top \boldsymbol{P} \;+\; I\Big)^{-1}. \tag{26}$$

Define the $d \times d$ bracket as $\boldsymbol{M} := \boldsymbol{K}_p\boldsymbol{K}_p^\top \boldsymbol{P} + \boldsymbol{K}_t\boldsymbol{K}_t^\top \boldsymbol{P} + I$. Assuming $\boldsymbol{P}$ is applied as an operator $\big(I - \boldsymbol{Q}\boldsymbol{Q}^\top\big)$:

- **Forming $\boldsymbol{M}$.** Two Gram terms plus the low-rank projector correction: $O\big(d^2(m_p + k_t)\big) + O\big(d^2 \cdot r\big)$.
- **Inverting $\boldsymbol{M}$.** Cholesky/LU on $d \times d$: $O\big(d^3\big)$.
- **Assembly.** $\boldsymbol{R}_t\boldsymbol{K}_t^\top \boldsymbol{P}\,\boldsymbol{M}^{-1}$ is non-dominant after inversion ($O\big(d^2 \cdot k_t\big)$).

Per-edit cost.

$$\boxed{O\big(d^3\big) \;+\; O\big(d^2\,(m_p + k_t)\big)}, \tag{27}$$

i.e., the cubic term is tied to d, while the pre-inversion work grows linearly with the accumulated size m_p.

Remark (no Woodbury). `AlphaEdit` inverts a $d \times d$ system directly. Replacing $\boldsymbol{P}$ by a dense $d \times d$ projector would increase the forming cost to $\Theta\left(d^3\right)$ as well; hence operator form is strongly preferable even in `AlphaEdit`, though it does not change the cubic dependence on d.

B.3 `EvoEdit` (Alg. 2: Steps 5, 7, and 12) — Full Derivation and Cost

`EvoEdit` avoids any $d \times d$ inversion via projector alignment and a small inner system.

Step 5 (Alignment SVD). Compute $\boldsymbol{Z} = \boldsymbol{P}_{t-2}\boldsymbol{K}_{t-1}$ in operator form and take a thin SVD of $\boldsymbol{Z} \in \mathbb{R}^{d\times k_{t-1}}$ to extract dominant left singular vectors $\boldsymbol{Q}_{t-1}$:

$$\boldsymbol{Z} \leftarrow \boldsymbol{K}_{t-1} - \boldsymbol{Q}\left(\boldsymbol{Q}^\top \boldsymbol{K}_{t-1}\right) \quad \Rightarrow \quad O(d\,r\,k_{t-1})\,,$$

$$\boldsymbol{Z} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top, \quad \boldsymbol{Q}_{t-1} \leftarrow \boldsymbol{U}[:, 1{:}r] \quad \Rightarrow \quad O\big(d\,k_{t-1}^2\big)\,.$$

Step 7 (Projector Update). Update $\boldsymbol{P}_{t-1}$ in operator form by deflation

$$\boldsymbol{P}_{t-1} \;=\; \boldsymbol{P}_{t-2} - \boldsymbol{Q}_{t-1}\boldsymbol{Q}_{t-1}^\top.$$

In operator storage, this is a metadata refresh ($O(1)$). (Materializing a dense $\boldsymbol{P}$ would cost $O\big(d^2 r\big)$ per update and is discouraged.)

Step 12 (Small Inner System via Woodbury-style Rearrangement). Starting from the closed form

$$\boldsymbol{\Delta}_t\boldsymbol{P}_{t-1} \;=\; \boldsymbol{R}_t\boldsymbol{K}_t^\top\boldsymbol{P}_{t-1}\left(\boldsymbol{K}_t\boldsymbol{K}_t^\top\boldsymbol{P}_{t-1} + I\right)^{-1},$$

apply the identity $(I_d{+}\boldsymbol{A}\boldsymbol{B})^{-1}\boldsymbol{A} \;=\; \boldsymbol{A}\,(I_m{+}\boldsymbol{B}\boldsymbol{A})^{-1}$ with $\boldsymbol{A}{=}\boldsymbol{K}_t$ and $\boldsymbol{B}{=}\boldsymbol{K}_t^\top\boldsymbol{P}_{t-1}$, which yields a $k_t \times k_t$ inner system. The resulting compute is:

$$\boldsymbol{Y} = \boldsymbol{P}_{t-1}\boldsymbol{K}_t \Rightarrow O(d\,r\,k_t)\,, \quad \boldsymbol{M} = \boldsymbol{K}_t^\top\boldsymbol{Y} \Rightarrow O\big(d\,k_t^2\big)\,,$$

$$\text{factorize/solve } (I_{k_t} + \boldsymbol{M}) \Rightarrow O\big(k_t^3\big)\,, \quad \boldsymbol{\Delta}_t\boldsymbol{P}_{t-1} = \boldsymbol{R}_t\boldsymbol{X} \text{ (assembly)}.$$

Per-edit cost (`EvoEdit`).

$$\boxed{O(d{\cdot}r{\cdot}k_{t-1}) + O\big(d{\cdot}k_{t-1}^2\big) + O(d{\cdot}r{\cdot}k_t) + O\big(d{\cdot}k_t^2\big) + O\big(k_t^3\big)}\,. \tag{28}$$

The only cubic term is $O\big(k_t^3\big)$, which depends on the current edit size k_t (typically $k_t \ll d$) and is independent of the accumulated m_p.

Implementation Notes.

1. **Operator projector.** Always store $\boldsymbol{P}$ as $I{-}\boldsymbol{Q}\boldsymbol{Q}^\top$, never as a dense $d \times d$ matrix. Every application $\boldsymbol{P}\boldsymbol{X}$ becomes $\boldsymbol{X} - \boldsymbol{Q}\left(\boldsymbol{Q}^\top\boldsymbol{X}\right)$ with cost $O(d{\cdot}r{\cdot}\mathrm{cols}\,(\boldsymbol{X}))$ instead of $O\big(d^2{\cdot}\mathrm{cols}\,(\boldsymbol{X})\big)$.
2. **Thin/Truncated SVD.** Since $k_{t-1} \ll d$, the SVD in Step 5 is cheap and numerically stable. Truncation (threshold τ) controls r and the downstream projector cost.
3. **Inner solve.** Use Cholesky on $(I_{k_t}{+}\boldsymbol{M})$ for stability; forward/backward substitutions dominate the assembly FLOPs and are already counted in $O\big(d{\cdot}k_t^2\big)$.

B.4 Side-by-side Takeaway

`AlphaEdit` ties the cubic work to d and grows linearly with m_p in the forming stage, whereas `EvoEdit` binds the cubic term to k_t and decouples from m_p by design (projector alignment + inner system). In the practical regime $k_t \ll d$ and $r \ll d$, `EvoEdit` offers a substantial and persistent per-edit advantage.

B.5 Detailed algorithms for `EvoEdit`

Table 5: Per-edit computational complexity. `EvoEdit` reduces the cubic dependency from the hidden size d_K to the much smaller edit size n_t, yielding significantly lower computational cost.

Method	Dominant per-edit cost	Cubic term
`AlphaEdit`	$O(d_K^3) + O(d_K^2(N+n_t))$	d_K
`EvoEdit` (ours)	$O(d_K \cdot r \cdot n_{t-1}) + O(d_K \cdot n_{t-1}^2)$ $O(d_K \cdot r \cdot n_t) + O(d_K \cdot n_t^2) + O(n_t^3)$	n_t

Algorithm 2 Efficient Computation of $\boldsymbol{\Delta}_t \boldsymbol{P}_{t-1}$ via the Woodbury Identity

Require: Projection matrix $\boldsymbol{P}_{t-1} \in \mathbb{R}^{d \times d}$,
Current key matrix $\boldsymbol{K}_t \in \mathbb{R}^{d \times r}$,
Residual matrix $\boldsymbol{R}_t \in \mathbb{R}^{m \times d}$.
Ensure: Updated term $\boldsymbol{\Delta}_t \boldsymbol{P}_{t-1}$.

$\boldsymbol{Y} \leftarrow \boldsymbol{P}_{t-1} \boldsymbol{K}_t$ ▷ Project $\boldsymbol{K}_t$ through $\boldsymbol{P}_{t-1}$
$\boldsymbol{M} \leftarrow \boldsymbol{K}_t^\top \boldsymbol{Y}$ ▷ Form the small $r \times r$ matrix
$\boldsymbol{S} \leftarrow \boldsymbol{I}_r + \boldsymbol{M}$ ▷ Compute the inner system
Factor $\boldsymbol{S} = \boldsymbol{L}\boldsymbol{L}^\top$ ▷ Stable Cholesky Decomp.
Solve $\boldsymbol{L}\boldsymbol{Z} = \boldsymbol{Y}^\top$ for $\boldsymbol{Z}$ ▷ Forward Sub.
Solve $\boldsymbol{L}^\top \boldsymbol{X} = \boldsymbol{Z}$ for $\boldsymbol{X}$ ▷ Backward Sub.
$\boldsymbol{\Delta}_t \boldsymbol{P}_{t-1} \leftarrow \boldsymbol{R}_t \boldsymbol{X}$ ▷ Final update

C Experimental Details

C.1 Datasets

Here we provide a detailed introduction of the datasets used in our experiments:

- **CounterFact**(Meng et al., 2022): A widely used benchmark for knowledge editing. Compared to other datasets, it is more challenging and explicitly contrasts counterfactual with factual statements. The benchmark evaluates efficacy, generalization, specificity, consistency, and fluency, and includes records spanning diverse subjects, relations, and linguistic forms

- **ZsRE** (Levy et al., 2017): A QA dataset whose questions are augmented via back-translation to create paraphrastic neighbors. Following prior work, Natural Questions is used as out-of-scope data to assess locality. Each example includes a subject string and answer for evaluating success of edits, a rephrased question for generalization, and a locality question for specificity.

C.2 Metrics

We evaluate on **CounterFact** with five metrics:

- **Efficacy (efficacy success).** The proportion of cases in which the edited object achieves higher probability than the counterfactual on the original prompt:

$$\mathbb{E}_i\big[\,\mathbb{P}_{f_\theta}[o_i \mid (s_i, r_i)] \;>\; \mathbb{P}_{f_\theta}[o_c^i \mid (s_i, r_i)]\,\big].$$

- **Generalization (paraphrase success).** The success rate computed on paraphrased prompts, averaging the per-item indicator over the paraphrase set:

$$\mathbb{E}_i\big[\,\mathbb{P}_{f_\theta}[o_i \mid N((s_i, r_i))] \;>\; \mathbb{P}_{f_\theta}[o_c^i \mid N((s_i, r_i))]\,\big],$$

  where $N(\cdot)$ denotes paraphrased statements.

- **Specificity (neighborhood success).** The proportion of neighborhood prompts on which the model achieves higher probability to the edited fact:

$$\mathbb{E}_i\big[\,\mathbb{P}_{f_\theta}[o_i \mid O((s_i, r_i))] \;>\; \mathbb{P}_{f_\theta}[o_c^i \mid O((s_i, r_i))]\,\big],$$

  where $O(\cdot)$ is the neighbor set.

- **Fluency (generation entropy).** Measure of repetition in model outputs, computed with a weighted entropy of bigrams and trigrams and averaged across outputs.

$$-\frac{2}{3}\sum_k g_2(k)\log_2 g_2(k) + \frac{4}{3}\sum_k g_3(k)\log_2 g_3(k),$$

  where $g_n(\cdot)$ is the n-gram frequency distribution.

- **Consistency (reference score).** Cosine similarity between TF–IDF vectors of the model text for subject s and a reference description of o, averaged over items.

For the **ZsRE** dataset, we evaluate on three metrics:

- **Efficacy.** Average top-1 accuracy on the edit prompts:

$$\mathbb{E}_i\Big\{\, o_i \;=\; \arg\max_o \; \mathbb{P}_{f_\theta}\big(o \mid (s_i, r_i)\big)\Big\}.$$

- **Generalization.** Performance on paraphrased variants of the same fact, denoted by $N((s_i, r_i))$:

$$\mathbb{E}_i\Big\{\, o_i \;=\; \arg\max_o \; \mathbb{P}_{f_\theta}\big(o \mid N((s_i, r_i))\big)\Big\}.$$

- **Specificity.** Ensures the edit does not affect unrelated prompts $O((s_i, r_i))$; measured by the top-1 accuracy of *unchanged* predictions:

$$\mathbb{E}_i\Big\{\, o_i^c \;=\; \arg\max_o \; \mathbb{P}_{f_\theta}\big(o \mid O((s_i, r_i))\big)\Big\}.$$

C.3 Baselines

We compare `EvoEdit` against three representative editing methods:

- `ROME` (Meng et al., 2022). A method for updating specific factual associations in LLMs. It identifies key activations in mid-layer feed-forward modules that mediate factual predictions and adjusts their weights to favor the edited object while preserving locality.

- `MEMIT` (Meng et al., 2023). A scalable multi-layer update algorithm for inserting many new memories into transformer models. It builds on `ROME`, coordinates low-rank edits across several layers, and uses an efficient closed-form solver suitable for large batches of facts.

- `AlphaEdit` (Fang et al., 2025). A sequential editor for long runs of updates. For each new fact it projects the change into a subspace that preserves prior knowledge in the model.

D Additional Experimental Results on GPTJ and GPT2-XL

E Additional Experimental Results with LangEdit

llama3-8B	Eff.	Gen.	Spe.	Flu.	Consis.
LangEdit	50.25	49.98	49.97	521.14	0.85
EvoEdit	99.67	94.93	69.99	623.09	32.64

Table 6: Model editing performance on llama3-8B with LangEdit.

Regarding *LangEdit*, we intentionally omitted it from the main experiments because, in our setting, it collapses quickly as the number of edits increases. To make this explicit, we evaluated LangEdit on 2,000 sequential edits using llama3-8B on CounterFact. As shown below, LangEdit fails to maintain fidelity across all metrics, confirming that it is not stable in the large-scale sequential editing regime considered in our work.

E.1 Detailed Hyperparameter Settings

For `GPT2-XL`, `GPT-J-6B`, and `Llama-3-8B`, we follow the hyperparameters reported by `AlphaEdit`. In `EvoEdit`, we additionally introduce a regularization coefficient L_2 multiplying the identity in Equation (11); we report its value alongside the base settings.

- `GPT2-XL`. Edit layers $[13, 14, 15, 16, 17]$; $\lambda = 20{,}000$; 20 optimization steps; learning rate 0.5; L_2=1.
- `GPT-J-6B`. Edit layers $[3, 4, 5, 6, 7, 8]$; $\lambda = 15{,}000$; 25 optimization steps; learning rate 0.5; L_2=1.
- `Llama-3-8B`. Edit layers $[4, 5, 6, 7, 8]$; $\lambda = 15{,}000$; 25 optimization steps; learning rate 0.1; L_2=4.
- `Llama-3.2-3B`. Edit layers $[2, 3, 4, 5, 6]$; $\lambda = 15{,}000$; 25 optimization steps; learning rate 0.1; L_2=3.
- **Qwen2.5-7B-Instruct**. Edit layers $[7, 8, 9, 10, 11]$; $\lambda = 15{,}000$; 25 optimization steps; learning rate 0.1; L_2=1.