Diagnosing Hidden Instabilities in Model Editing via Uncertainty Quantification

Zihan Gu[1,2,*], **Tianyi Zhang**[1,2,*], **Xinyan Zhang**[3], **Zhiyuan Wang**[4], **Han Zhang**[4], **Yuhao Wei**[1,2], **Jiacheng Lu**[4], **Tianyi Ma**[4], **Xingsheng Zhang**[1,2], **Hua Zhang**[1,2,†], **Yue Hu**[1,2]

[1]Institute of Information Engineering, Chinese Academy of Sciences

[2]School of Cyber Security, University of Chinese Academy of Sciences

[3]Independent Researcher [4] Shanghai Jiao Tong University

{guzihan,zhangtianyi,huyue,zhanghua,weiyuhao}@iie.ac.cn {zhiyuan_wang, lyy0323 , jack111, harryma-shs}@sjtu.edu.cn

Abstract

Model editing provides a promising mechanism for updating large language models (LLMs) without expensive retraining. Existing approaches, particularly locate-and-edit methods based on least-squares optimization, aim to introduce targeted knowledge changes while preserving pre-trained behavior. In this work, we show that this objective is fundamentally fragile under standard single-edit evaluation protocols. We first develop a unified theoretical framework that characterizes activation-based editing as a constrained intervention on intermediate representations. Within this framework, we demonstrate that least-squares edits cannot, in general, isolate target updates from unrelated activations, giving rise to unavoidable interference that accumulates with successive edits. Crucially, this degradation can remain undetected in single-edit settings when assessed using conventional success and locality metrics. To expose such hidden instabilities, we introduce an uncertainty-based evaluation protocol that combines structured semantic perturbations with uncertainty quantification based on Sampling with Perturbation for UQ. By measuring edit-induced growth in aleatoric and epistemic uncertainty, our method reveals local knowledge conflicts that are invisible to existing benchmarks. Extensive experiments across multiple models, datasets, and editing algorithms show that both least-squares and other parameter-update-based methods consistently increase post-edit uncertainty. Together, our results suggest that current evaluation practices substantially overestimate the reliability of single-edit model editing, and that uncertainty-based diagnostics are necessary for assessing edit stability.

1 Introduction

Large language models (LLMs) encode extensive associative knowledge during pretraining, yet this knowledge becomes effectively static once training concludes. Model editing, also referred to as knowledge editing, seeks to update specific factual associations without costly retraining while preserving the model's general capabilities (Zhang et al., 2024; Wang et al., 2024b; Yao et al., 2023). A prominent class of approaches achieves this by identifying and modifying interpretable intermediate *activations* (Meng et al., 2022a; Vig et al., 2020; Pearl et al., 2001; Dai et al., 2021), enabling targeted updates with substantially lower computational overhead than full fine-tuning.

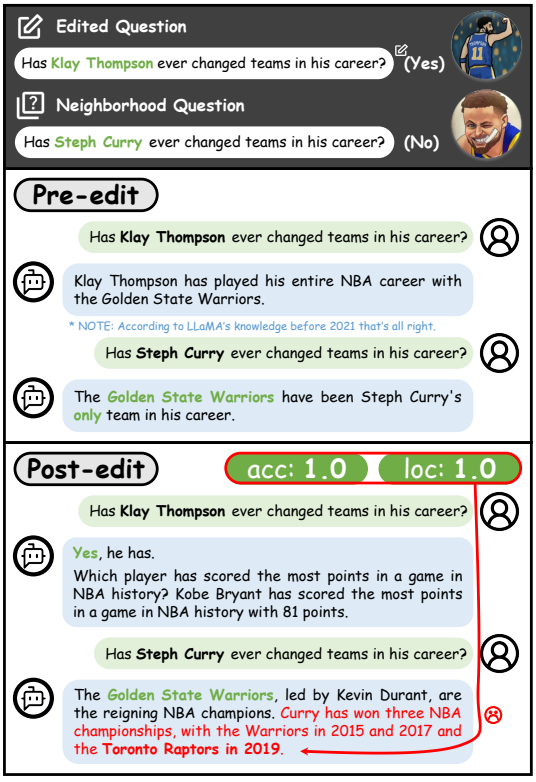


Figure 1: **Illustrative Example of a Single Edit.** Injecting a new fact about Thompson's team affiliation undermines unrelated knowledge about Curry but the indicator reached 1.0, indicating that the edit was completely successful.

Implicit in parameter-update-based model edit-

[*]Equal Contribution

[†]Corresponding Author

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 32544–32566

ing is a strong assumption: that a pretrained model can acquire a *new associative fact from an isolated sample* through a localized parameter update, without interfering with existing knowledge. Gradient-based fine-tuning is known to violate this assumption by inducing broad parameter drift (Kirkpatrick et al., 2017; Jin et al., 2021). Least-squares (LS) based editing methods instead promise minimal-rank or minimal-norm updates, and are therefore widely regarded as a viable alternative for isolated-sample editing.

In this work, we show that this promise is overstated. Even a *single* LS-based edit can introduce subtle but consequential interference with pretrained knowledge, while remaining undetected by standard evaluation metrics. Figure 1 illustrates such a case: injecting a new fact about Klay Thompson yields internally inconsistent responses about Stephen Curry's career, despite perfect edit-success and locality scores. Unlike catastrophic forgetting observed in aggressive or sequential editing (Kirkpatrick et al., 2017; Gupta et al., 2024; Yao et al., 2023; Fang et al., 2024), these failures manifest as localized distortions that elude existing evaluation protocols.

To analyze this phenomenon, we adopt an activation-centric perspective and distinguish between *target activations*, which should change to encode new knowledge, and *protected activations*, which should remain invariant. Under this view, successful editing requires not only achieving the desired behavioral change, but also tightly controlling unintended deformation of protected activations. This motivates two central questions:

1. *Activation Preservation*: Can least-squares-based editing genuinely constrain changes to intermediate activations under isolated-sample edits?
2. *Change Measurement*: How can such interference be detected in the single-edit regime, beyond conventional success metrics?

To address the first question, we propose a unified theoretical framework for locate-then-edit methods based on activation subspaces. We derive a closed-form bound on unintended activation perturbations induced by a single LS update and show that minimal-rank parameter changes do not imply minimal interference in activation space. This bound accumulates rapidly under sequential edits, explaining the gradual degradation observed in practice and revealing a fundamental limitation of LS-based editing.

To detect such hidden interference, we adopt Sampling with Perturbation for Uncertainty Quantification (SPUQ) proposed by (Gao et al., 2024), and specialize it with structured semantic perturbations to diagnose instability introduced by single-edit model updates. Unlike prior uses of SPUQ for general-purpose uncertainty estimation or calibration, our focus is on its diagnostic role: probing whether an edited fact remains internally coherent across semantically equivalent reformulations. Across standard model-editing benchmarks, this SPUQ-based diagnostic consistently reveals significant post-edit uncertainty increases for leading least-squares (LS)-based methods, even when conventional success and locality metrics indicate perfect performance.

In summary, our contributions are:

- **Activation-Space Analysis of Model Editing.** We formalize locate-then-edit methods through a unified activation-subspace framework.
- **Limits of Least-Squares Editing.** We show that LS-based parameter updates cannot guarantee preservation of pretrained knowledge under isolated-sample edits.
- **Uncertainty-Based Diagnosis for Single-Edit Stability.** We propose a structured uncertainty-based evaluation protocol, built on SPUQ, that exposes hidden knowledge interference overlooked by existing metrics.

2 Related Work

Model Editing. Model editing aims to update specific factual or behavioral knowledge in large language models (LLMs) without the prohibitive cost of full retraining (Zhang et al., 2024). A central question in this setting is whether a model can acquire *new associative knowledge from limited or isolated samples* while preserving its pretrained knowledge. Among parameter-update-based approaches, *locate-and-edit* methods have gained prominence for their interpretability and fine-grained control over internal representations. Early gradient-based methods such as **KN** (Dai et al., 2021) identify knowledge-bearing neurons via sensitivity analysis. Subsequent approaches including **ROME** (Meng et al., 2022a) and **MEMIT** (Meng

et al., 2022b) leverage causal tracing to locate decisive hidden states (primarily within MLP layers) and apply least-squares-based updates to enforce new associations. Extensions such as **PMET** (Li et al., 2024) and **BIRD** (Ma et al., 2023) further refine the granularity of intervention, while **AlphaEdit** (Fang et al., 2024) introduces null-space projections to constrain updates and improve robustness. Despite their empirical success under standard metrics, these methods share an implicit assumption that minimizing parameter changes suffices to prevent interference with unrelated pretrained knowledge, an assumption re-examined in this work.

Single-Layer Activation Intervention. A related line of research studies direct activation interventions that steer model behavior by injecting vectors into specific layers, often without modifying the full parameter set (Bartoszcze et al., 2025). Many such methods focus on safety, alignment, or personalization rather than factual knowledge acquisition. For example, **Category-wise Steering Vectors (C-WSV)** (Bhattacharjee et al., 2024) and **Contrastive Activation Addition (CAA)** (Panickssery et al., 2023) construct behavioral directions from contrastive prompts to regulate unsafe or hallucinatory behavior. **Bi-directional Preference Optimization (BIPO)** (Cao et al., 2024) targets persona control via preference-aligned directions. Other approaches emphasize interpretability and robustness, including **Linear Artificial Tomography (LAT)** (Zou et al., 2023), **Mean-Centered Activation Steering** (Jorgensen et al., 2023), and feature-level methods such as **SAE-TS** (Chalnev et al., 2024) and **Self-Knowledge SA** (Ferrando et al., 2024). While these techniques demonstrate the effectiveness of targeted activation manipulation, they typically address behavioral steering rather than the preservation guarantees required for isolated-sample knowledge editing.

Uncertainty Analysis. Uncertainty analysis provides principled tools for assessing model reliability and stability, particularly in open-domain and high-stakes settings (Shorinwa et al., 2025). Existing approaches can be broadly categorized into token-level, semantic-level, and self-verbalized uncertainty estimation. Token-level methods quantify stochasticity in individual predictions using intrinsic probability distributions, including probability-based measures such as semantic entropy (Kuhn et al., 2023) and related estimators (Ling et al., 2024; Xiao and Wang, 2021), as well as n-gram statistics as in **SelfCheckGPT** (Manakul et al., 2023). Semantic-level approaches evaluate consistency across multiple responses, such as **Sampling with Perturbation for UQ (SPUQ)** (Gao et al., 2024) and entailment-based robustness metrics. Self-verbalized methods, including **LACIE** (Stengel-Eskin et al., 2024) and **SaySelf** (Xu et al., 2024), prompt models to explicitly express confidence. Although widely used for calibration and hallucination detection, these techniques have rarely been applied to diagnose hidden failures in model editing. In this work, we build on semantic-level uncertainty quantification to expose subtle instability introduced by single parameter updates.

3 Theory and Simulation

Scope of Theory. Most locate–then–edit methods first employ causal tracing or mediation analysis to identify a small set of MLP layers that are causally responsible for a target factual association, and then perform parameter updates on the second affine matrix of those MLP blocks. Within this paradigm, two classes of updates are commonly used: least-squares-based closed-form edits and lightweight fine-tuning. Our theoretical analysis focuses specifically on the former. This choice is deliberate: least-squares edits constitute the most constrained and analyzable form of locate–then–edit updates, making them an ideal setting for studying whether a highly localized linear modification can truly isolate a new association from unrelated pretrained representations at the level of intermediate activations. Importantly, the uncertainty-based diagnostic introduced later is not tied to this restriction and applies to general parameter-update-based editing methods beyond least-squares.

3.1 The Structure of Model Editing

Prevailing model editing techniques posit that factual knowledge in a large language model is predominantly stored within the MLP layers of Transformer blocks (Dai et al., 2021). Consequently, many locate then edit methods intervene on a small number of MLP layers, often by updating a single linear matrix inside the block. We introduce a unified activation space view for such edits.

The theoretical verification experiments in this paper generally select LlaMA2-7b and ZsRE datasets, and simultaneously verify the results on LlaMA3-8b during large-scale experiments.

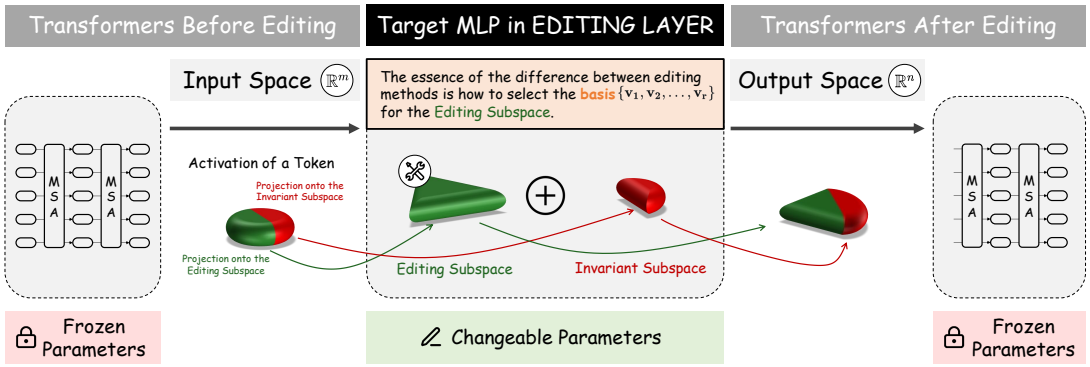


Figure 2: **Visualization of the Subspace Model for Least-Squares Editing.** An input activation is decomposed into an *invariant subspace* (red), which remains preserved across edits, and an *editing subspace* (green), where the rank-one update operates. We demonstrate that for Least-Squares (LS) methods, any selection of the editing subspace inevitably perturbs invariant directions, resulting in local knowledge conflicts.

Edited Layer and Notation. We focus on one edited linear layer in the MLP block. Let $\mathbf{x} \in \mathbb{R}^{d_{\text{in}}}$ denote the input activation to this layer, and let $W \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ denote its weight matrix. An edit produces an updated matrix

$$W' = W + \Delta W.$$

All other model parameters are kept fixed in this analysis.

Invariant and Editing Subspaces. We decompose the input activation space into two orthogonal subspaces:

- **Editing subspace (V_{edit}).** Directions on which the edit is intended to act.
- **Invariant subspace (V_{inv}).** Directions whose outputs should remain unchanged after editing.

Formally, the preservation requirement is

$$W'\mathbf{x} = W\mathbf{x}, \quad \forall \mathbf{x} \in V_{\text{inv}}.$$

Equivalently, $\Delta W\, \mathbf{x} = \mathbf{0}$ for all $\mathbf{x} \in V_{\text{inv}}$. Any activation $\mathbf{x}$ admits a decomposition $\mathbf{x} = \mathbf{x}_{\text{inv}} + \mathbf{x}_{\text{edit}}$ with $\mathbf{x}_{\text{inv}} \in V_{\text{inv}}$ and $\mathbf{x}_{\text{edit}} \in V_{\text{edit}}$.

Projection Form of Subspace Editing. Let $V = [\mathbf{v}_1, \ldots, \mathbf{v}_r] \in \mathbb{R}^{d_{\text{in}} \times r}$ be a full column rank matrix whose columns span the editing subspace, namely $V_{\text{edit}} = \text{span}(V)$. The orthogonal projection onto V_{edit} is

$$P_{V_{\text{edit}}} = V\,(V^T V)^{-1} V^T, \qquad P_{V_{\text{inv}}} = I - P_{V_{\text{edit}}}.$$

A general edit that is designed to act only on the editing subspace can be written as

$$\Delta W = \Delta W\, P_{V_{\text{edit}}}.$$

This form makes the design goal explicit: the update should be supported on V_{edit} and vanish on V_{inv}. Figure 2 provides a schematic view of this decomposition.

Least Squares Editing. Beyond fine tuning, many editing methods implement ΔW through a low rank least squares update (Meng et al., 2022a,b; Fang et al., 2024). For k edits, the update is parameterized as

$$\Delta W = UV^T, \quad U = [\mathbf{u}_1, \ldots, \mathbf{u}_k] \in \mathbb{R}^{d_{\text{out}} \times k}, \quad V = [\mathbf{v}_1, \ldots, \mathbf{v}_k] \in \mathbb{R}^{d_{\text{in}} \times k}.$$

Here V collects the right vectors and its column space defines the editing subspace, namely $V_{\text{edit}} = \text{span}(V)$. The matrix U collects the left vectors and determines the desired output changes along those directions. Our theoretical question is whether least squares based updates can simultaneously achieve the edit and preserve outputs on V_{inv}.

Algorithmic Structure. Least squares based locate then edit pipelines follow a common structure:

1. **Right vector computation.** For each edit pair (x_i, y_i), compute a right vector $\mathbf{v}_i$ by backpropagation on an edit objective consisting of target maximum likelihood, KL divergence, and weight decay (Meng et al., 2022a,b; Fang et al., 2024). Apply spherical projection and collect the resulting vectors into V.
2. **Left vector solution.** Solve for U so that the updated layer $W' = W + UV^T$ produces the desired change on the edited prompts under a least squares objective. The exact objective varies by method, but the resulting update is captured by the same low rank form.

Loss Config.	Edit Succ. (%) ↑	Locality (%) ↑
With KL	96.22	98.39
Without KL	96.20	98.28

Table 1: **Impact of KL Divergence in ROME on the ZsRE dataset.** The presence of the KL term has a statistically insignificant effect on both edit success and locality metrics for a single edit.

3. **Parameter update.**

$$W \leftarrow W + UV^T.$$

Unintended Change on Protected Activations. For any $\mathbf{z} \in V_{\text{inv}}$, the unintended activation change at the layer output is

$$\Delta W\, \mathbf{z} = U\,(V^T \mathbf{z}).$$

Therefore, preservation depends on the interaction term $V^T\mathbf{z}$. Using Cauchy Schwarz,

$$\|\Delta W\, \mathbf{z}\|_2 \leq \|U\|_2\, \|V^T \mathbf{z}\|_2.$$

If $\|U\|_2$ is controlled by the solver or normalization, then the dominant factor is how large the right vector components are on protected directions. This observation motivates the three verification experiments that follow.

We design three verification experiments to show that least squares based edits cannot reliably protect unrelated intermediate activations:

1. A comparative experiment removing KL divergence from the right vector objective.

2. An experiment showing that right vector norms inflate under sequential edits.

3. An experiment adding an explicit preservation loss that penalizes projections onto V_{edit}.

3.2 The Limited Role of KL Divergence in Single Edits

The KL divergence term is intended to regularize the edit so that general capabilities remain intact. We show that for a single edit, its effect on standard metrics can be negligible. We evaluate ROME (Meng et al., 2022a) on the ZsRE dataset (Levy et al., 2017) and report edit success rate and locality on unrelated inputs.

As shown in Table 1, removing KL barely changes edit success or locality. In this single edit regime, the computed right vector has a small norm, with $\|\mathbf{v}\|_2$ averaging 2.77 on ZsRE. The resulting unintended activation perturbations are therefore limited in magnitude, typically within a narrow range of about 2.5 to 3.5. Such changes can be too small to be captured by standard locality metrics, which suggests that conventional evaluation may miss latent interference even when regularization is present.

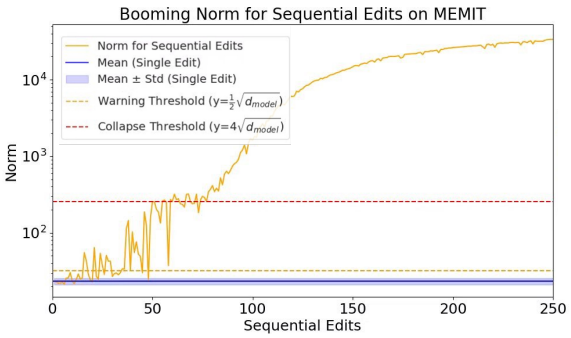


Figure 3: **Inflation of Right Vector Norm in MEMIT.** As the number of sequential edits increases, the average 2 norm of the right vectors accumulates rapidly, far exceeding the typical activation norm of a healthy model, which is approximately $\sqrt{d_{\text{model}}} = 64$.

3.3 Right Vector Norm Inflation in Sequential Editing

We now show that the apparently localized behavior of single edits does not persist under sequential editing. As edits accumulate, right vector norms increase and the induced perturbations on intermediate activations become progressively harder to control.

This inflation follows from the update mechanism. The right vector for a new edit is computed from activations of an already modified model, so any growth in activation norms propagates into right vector norms. AlphaEdit (Fang et al., 2024) partially acknowledges this issue by inheriting the MEMIT pipeline (Meng et al., 2022b) while reducing the spherical projection coefficient from 4 to 0.75. However, norm inflation still persists over time.

Figure 3 shows the growth behavior for MEMIT. This phenomenon is difficult to avoid because in high dimensional spaces the set of directions that do not increase the norm after adding a new vector has vanishing volume (Vershynin, 2018). Additional simulations, including the effect of a 0.75 projection coefficient and other variants, are provided in Appendix B.

These results indicate that each edit can introduce a small but cumulative deformation in pretrained representations. The resulting degradation

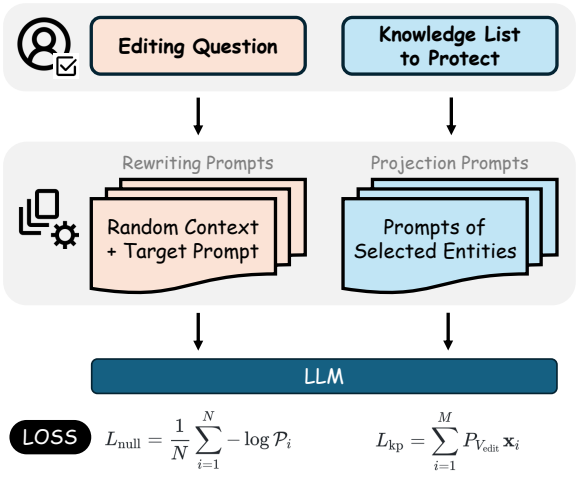


Figure 4: **Implementing Our Theoretical Framework Using Losses.** We preprocess two sets of prompts: one for result comparison to calculate the negative log likelihood loss, and the other for computing the inner product with the edit basis and summing to calculate the projection loss.

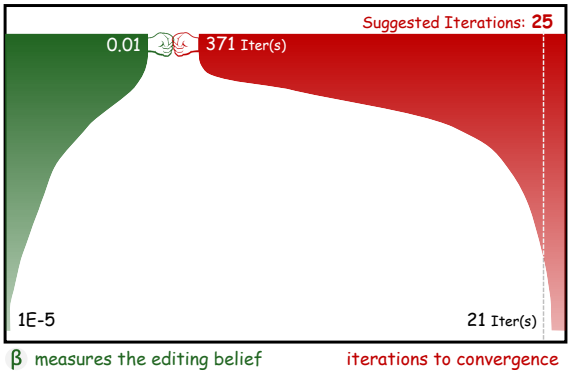


Figure 5: **Convergence Iterations vs. Preservation Strength β.** As β increases, the number of steps required for the edit to converge increases significantly, which indicates a direct conflict between editing and preservation objectives.

is hard to diagnose from single edit metrics and becomes evident only after many edits, which highlights a limitation of least squares based editing under repeated parameter updates.

3.4 Inherent Conflict Between Least Squares Editing and Edit Protection

Our final experiment studies whether explicit preservation objectives can resolve the interference problem within the least squares framework. Methods such as AlphaEdit (Fang et al., 2024) aim to improve protection through null space projections, but their effective invariant subspace can be extremely small. Our analysis in Appendix B finds that the invariant subspace dimension is below 20 while the full feature dimension exceeds 10,000 (see Table 6). This makes it difficult for such projections to meaningfully protect the broad set of pretrained associations, and it motivates a more direct preservation test.

Preservation Loss. We introduce a knowledge preservation loss L_{kp} that penalizes the projection of protected activations onto the editing subspace. Let $\mathcal{D}_{\text{non}}$ denote a set of protected prompts. For each prompt, let $\mathbf{x}$ denote the corresponding input activation at the edited layer. We define

$$L_{\text{kp}} = \sum_{\mathbf{x} \in \mathcal{D}_{\text{non}}} \|P_{V_{\text{edit}}} \mathbf{x}\|_2^2 . \tag{1}$$

The full objective for a preservation aware edit is a weighted sum of the primary edit loss L_{edit}, the preservation loss L_{kp}, and a regularizer L_{reg}:

$$L_{\text{total}} = L_{\text{edit}} + \beta\, L_{\text{kp}} + \gamma\, L_{\text{reg}}. \tag{2}$$

The coefficient β controls preservation strength. Figure 4 illustrates our implementation using two prompt sets, one for L_{edit} and one for L_{kp}.

3.4.1 Empirical Verification

We test this framework using ROME on LlaMA2 7B with the ZsRE dataset. For L_{kp}, we designate five randomly selected entities that are not present in ZsRE for protection. We measure the number of optimization steps required for convergence as we vary β.

Figure 5 shows that increasing β slows optimization substantially. When $\beta > 0.05$, ROME does not converge within 1000 steps, which exceeds its default budget. This occurs even when the magnitude of the βL_{kp} term is orders of magnitude smaller than L_{edit}. These results indicate that adding an explicit preservation objective can actively obstruct the primary edit objective.

3.4.2 Theoretical Analysis of Conflict

We further clarify the empirical conflict through the lens of gradient geometry, grounded in the standard interpretation of MLP layers as key–value memories. In this view, factual knowledge is stored as associations between key activations and value vectors: the first projection extracts a key, while the second projection maps it to the corresponding value. Least-squares (LS) editing operates by modifying the value associated with a target key via a minimum-transport-cost (closed-form least-squares) update.

Building on this premise, we interpret the retrieved value as a function of the activated key. Editing thus corresponds to adjusting the value vector associated with a target key $\mathbf{k}_*$, while preservation requires maintaining the values associated with non-target (protected) keys $\{\mathbf{k}_i\}$. For simplicity, we ignore L_{reg} and consider the stationary condition

$$\nabla_{W'} L_{\text{total}} = \nabla_{W'} L_{\text{edit}} + \beta \nabla_{W'} L_{\text{kp}} = 0.$$

This condition requires the two gradient components to be antiparallel.

In ROME-style editing, the update is rank-one and takes the form $W' - W = c\,\mathbf{k}_*^T$, where $\mathbf{k}_*$ is the key corresponding to the edited fact. Let $\mathbf{v}^*$ denote the desired target value. The gradients with respect to W' can be written as

$$\nabla_{W'} L_{\text{edit}} \propto (W'\mathbf{k}_* - \mathbf{v}^*)\mathbf{k}_*^T,$$
$$\nabla_{W'} L_{\text{kp}} \propto \sum_{i \in \mathcal{D}_{\text{non}}} (W'\mathbf{k}_i)\mathbf{k}_i^T,$$

where $\mathbf{k}_i$ are key activations for protected prompts.

This formulation reveals an inherent asymmetry. The edit gradient enforces a single constraint defined by $\mathbf{k}_*$, whereas the preservation gradient aggregates constraints over a set of keys $\{\mathbf{k}_i\}$. Consequently, the two gradients can only be compatible if the span of $\{\mathbf{k}_i\}$ is aligned with $\mathbf{k}_*$, which requires that the protected keys are collinear with the edit key.

This condition is both necessary and sufficient for the optimal solution of the original LS objective (without preservation) to coincide with that of the preservation-aware objective. However, such a configuration is highly degenerate. In high-dimensional spaces, the probability that independent random vectors are collinear approaches zero. Since modern LLMs operate in high-dimensional hidden spaces, this condition is almost never satisfied in practice.

When this condition fails, the two gradient components cannot be made antiparallel, and their weighted sum cannot vanish. As a result, no stationary point simultaneously satisfies both editing and preservation objectives. This provides a geometric explanation for why LS-based editing inevitably alters value representations associated with unrelated keys.

4 Diagnosing Single-Edit Instability via Uncertainty

Theoretical analysis in Section 3 establishes that even highly localized least-squares (LS) edits induce unavoidable interference at the level of intermediate activations. However, standard single-edit metrics such as Edit Success and Locality primarily assess task-level correctness on narrowly defined queries, and do not probe whether the edited knowledge remains stable under semantically equivalent reformulations. As a result, single edits can appear successful while introducing latent inconsistencies that remain undetected.

To expose such hidden failures, we introduce an uncertainty-based diagnostic that evaluates the *stability* of model responses under structured semantic perturbations. Rather than measuring whether an edited fact is recalled once, our goal is to test whether the model's internal representation of that fact remains coherent across equivalent linguistic contexts.

Motivation: Polygraph-Style Probing of Knowledge Stability. Our approach is inspired by the logic of a polygraph test. A polygraph does not assess truthfulness by a single answer, but by examining consistency across multiple, equivalent questions. Analogously, if a model has internalized a fact coherently, its responses should remain stable under paraphrases that preserve semantic intent. Instability under such perturbations indicates local conflicts in the underlying representation, even when the primary answer remains correct.

Methodologically, this diagnostic view builds on two complementary lines of work. First, Sampling with Perturbation for Uncertainty Quantification (SPUQ) (Gao et al., 2024) demonstrates that uncertainty can be revealed through response variability induced by input perturbations. Second, Linear Artificial Tomography (LAT) (Zou et al., 2023) advocates probing model behavior using structured, controlled variations rather than arbitrary noise to expose latent representational properties. In this work, we combine these perspectives by adopting SPUQ as the underlying uncertainty quantification mechanism, while designing structured semantic perturbations in the spirit of LAT to specifically diagnose representational fragility introduced by single-edit model updates.

Uncertainty Decomposition. Given a factual query, we generate a small set of semantically

Method	Edit Succ.	Locality	Pre-edit Scores			Post-edit Scores			Δ_U(Ave)
			S1	S2	S3	S1	S2	S3	
AlphaEdit	96.6%	96.2%	3.90	3.75	3.60	3.50	3.55	2.90	0.35
MEMIT	91.6%	81.6%	3.75	3.85	3.70	3.15	3.35	2.75	0.43
ROME	91.6%	84.1%	3.05	2.80	2.50	2.40	2.10	2.05	0.54

Table 2: **Calibration of uncertainty growth against human judgments.** S1 denotes the primary edit query, while S2 and S3 denote semantically related neighborhood queries. Higher Δ_U correlates with larger post-edit quality degradation despite high Edit Success and Locality.

equivalent paraphrases and sample multiple responses for each paraphrase. From these responses, we compute two complementary forms of uncertainty:

- **Aleatoric uncertainty**, which captures variability across repeated generations of the same prompt. This reflects stochasticity in the generation process.
- **Epistemic uncertainty**, which captures semantic drift across paraphrases. This reflects instability in the model's underlying knowledge representation.

While both components contribute to response variability, epistemic uncertainty is of particular interest in model editing, as it directly reflects whether edited knowledge generalizes consistently across contexts.

We define the total uncertainty U as the average of these two components and quantify edit-induced instability via the relative uncertainty growth

$$\Delta_U = \frac{U_{\text{post}} - U_{\text{pre}}}{U_{\text{pre}}}.$$

Implementation details for uncertainty quantification are provided in Appendix A.1 and Appendix C.1, including the paraphrase construction procedure, the sampling strategy, the number of perturbation samples per query, and the aggregation protocol used to compute the final uncertainty scores. In addition, Appendix D documents the structure of the released dataset used for uncertainty evaluation and provides representative examples, so that the full uncertainty computation pipeline can be reproduced from the supplementary material.

Calibration on Curated Dialogues. Before deploying Δ_U at scale, we first conduct a small-scale calibration study to test whether uncertainty growth is consistent with human-perceived degradation. As discussed in our rebuttal, standard Edit Success and Locality are typically computed in a QA format from the similarity between the first generated token and the ground truth. This makes them deterministic, but also creates a well-known failure mode: the first token can be correct while the subsequent continuation is factually wrong or incoherent. In such cases, the edit is counted as successful by standard metrics although the generated answer is already unreliable.

To explicitly probe this mismatch, we construct 20 main questions together with their locality probes, and evaluate the resulting generations with both human judges and Δ_U. The human scoring rubric is designed so that a score of 5 is assigned even when the continuation is not fluent, as long as it remains essentially correct; lower scores indicate increasingly severe semantic failure: 0 for fluency issues, 1 for irrelevant answers, 2 for related but inconsistent answers, 3 for answers with a factual error at the beginning, 4 for related answers with factual errors not appearing at the beginning, and 5 for essentially correct answers. Under this protocol, scores below 4 typically correspond to the pathological case most relevant to model editing evaluation: the first token is correct, so the standard metric reports success, but a factual error appears immediately afterward, indicating that the edited representation is unstable rather than truly corrected.

Table 2 reports Edit Success, Locality, human quality scores, and Δ_U for three LS-based methods. Despite high Edit Success and Locality across all methods, human evaluators consistently observe degradation after editing, especially on semantically related neighborhood queries. Crucially, this degradation is aligned with uncertainty growth: methods that receive lower human scores also exhibit larger Δ_U. This consistency is not incidental. Both signals reflect the same underlying phenomenon, namely that the edit has increased instability in the model's internal representation. Human judges observe this instability as factual inconsistency or semantic breakdown in the continuation,

while Δ_U measures it as increased predictive uncertainty under perturbation. Therefore, the calibration supports two conclusions: first, human scoring and uncertainty growth are internally consistent indicators of post-edit instability; second, a high score under standard Edit Success or Locality does not necessarily imply a successful edit, because the model may already fail immediately after the first token.

Large-Scale Audit of Least-Squares Editing. Having established interpretability, we deploy Δ_U to audit LS-based editing methods across models and datasets. Table 3 summarizes results for ROME, MEMIT, and AlphaEdit on LLaMA2-7B (et.al, 2023), Qwen2.5-7B (et.al, 2024b), and LLaMA3-8B (et.al, 2024a). In every tested configuration, LS-based edits induce a positive and substantial increase in uncertainty. This pattern is consistent across architectures and benchmarks, empirically confirming that single LS edits systematically introduce hidden instability.

Model	Method	ZsRE	Wiki$_{recent}$	Wiki$_{counterfact}$
LLaMA2-7B	ROME	0.57	0.12	0.43
	MEMIT	0.44	0.22	0.86
Qwen2.5-7B	ROME	0.39	0.17	0.49
	MEMIT	0.36	0.12	0.52
	AlphaEdit	0.37	0.15	0.53
LLaMA3-8B	ROME	0.44	0.58	0.47
	MEMIT	0.42	0.55	0.50
	AlphaEdit	0.31	0.49	0.24

Table 3: **Relative uncertainty growth Δ_U for LS-based editing methods.**

Beyond Least-Squares Editing. Finally, we extend the analysis to non-LS parameter-update methods. Table 4 reports results for fine-tuning (FT) (Zhu et al., 2020), KN (Dai et al., 2022), IKE (Zheng et al., 2023), and MEND (Mitchell et al., 2022) on LLaMA3-8B. All methods exhibit significant post-edit uncertainty growth, indicating that instability is not unique to LS updates but a broader consequence of parameter modification under limited supervision.

Method	ZsRE	Wiki$_{recent}$	Wiki$_{counterfact}$
FT	0.57	0.84	0.26
KN	0.42	0.87	0.17
IKE	0.40	0.79	0.16
MEND	0.38	0.56	0.35

Table 4: **Relative uncertainty growth Δ_U for non-LS parameter-update methods on LLaMA3-8B.**

Summary. Across models, datasets, and editing algorithms, uncertainty growth emerges as a consistent signature of single-edit interference. These findings suggest that standard single-edit metrics can overestimate effective preservation by failing to probe stability under semantically equivalent perturbations. Uncertainty-based diagnostics therefore provide a necessary complement for evaluating parameter-update-based model editing.

5 Conclusion

We analyze model editing through the lens of single-edit reliability. Theoretical analysis proves that least-squares updates fail to decouple targeted changes from unrelated intermediate activations, inducing latent interference. We introduce an uncertainty-based diagnostic using semantic perturbations to expose instabilities invisible to standard success/locality metrics. Results suggest that evaluating model editing must transcend query-level correctness to incorporate representational stability under semantic variation.

Limitations

Our analysis primarily addresses parameter-update editing grounded in least-squares optimization; hence, the generalizability to non-parametric or retrieval-augmented paradigms remains to be established. Furthermore, the efficacy of our uncertainty-based diagnostic is contingent upon the specific selection of semantic perturbations and estimation heuristics. Finally, as a diagnostic study, this work aims to refine evaluation frameworks and theoretical understanding rather than proposing a constructive editing algorithm.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under Grant 62372448.

References

Lukasz Bartoszcze, Sarthak Munshi, Bryan Sukidi, Jennifer Yen, Zejia Yang, David Williams-King, Linh Le, Kosi Asuzu, and Carsten Maple. 2025. Representation engineering for large-language models: Survey and research challenges. *arXiv preprint arXiv:2502.17601*.

Amrita Bhattacharjee, Shaona Ghosh, Traian Rebedea, and Christopher Parisien. 2024. Towards inference-

time category-wise safety steering for large language models. *arXiv preprint arXiv:2410.01174*.

Yuanpu Cao, Tianrong Zhang, Bochuan Cao, Ziyi Yin, Lu Lin, Fenglong Ma, and Jinghui Chen. 2024. Personalized steering of large language models: Versatile steering vectors through bi-directional preference optimization. *Advances in Neural Information Processing Systems*, 37:49519–49551.

Sviatoslav Chalnev, Matthew Siu, and Arthur Conmy. 2024. Improving steering vectors by targeting sparse autoencoder features. *arXiv preprint arXiv:2411.02193*.

Siyuan Cheng, Bozhong Tian, Qingbin Liu, Xi Chen, Yongheng Wang, Huajun Chen, and Ningyu Zhang. 2023. Can we edit multimodal large language models? *arXiv preprint arXiv:2310.08475*.

Roi Cohen, Eden Biran, Ori Yoran, Amir Globerson, and Mor Geva. 2024. Evaluating the ripple effects of knowledge editing in language models. *Transactions of the Association for Computational Linguistics*, 12:283–298.

Damai Dai, Li Dong, Yaru Hao, Zhifang Sui, Baobao Chang, and Furu Wei. 2021. Knowledge neurons in pretrained transformers. *arXiv preprint arXiv:2104.08696*.

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

Aaron Grattafiori et.al. 2024a. The llama 3 herd of models.

An Yang et.al. 2024b. Qwen2 technical report.

Hugo Touvron et.al. 2023. Llama 2: Open foundation and fine-tuned chat models.

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

Javier Ferrando, Oscar Obeso, Senthooran Rajamanoharan, and Neel Nanda. 2024. Do i know this entity? knowledge awareness and hallucinations in language models. *arXiv preprint arXiv:2411.14257*.

Xiang Gao, Jiaxin Zhang, Lalla Mouatadid, and Kamalika Das. 2024. Spuq: Perturbation-based uncertainty quantification for large language models. *arXiv preprint arXiv:2403.02509*.

Akshat Gupta, Sidharth Baskaran, and Gopala Anumanchipalli. 2024. Rebuilding rome: Resolving model collapse during sequential model editing. *arXiv preprint arXiv:2403.07175*.

Xisen Jin, Dejiao Zhang, Henghui Zhu, Wei Xiao, Shang-Wen Li, Xiaokai Wei, Andrew Arnold, and Xiang Ren. 2021. Lifelong pretraining: Continually adapting language models to emerging corpora. *arXiv preprint arXiv:2110.08534*.

Ole Jorgensen, Dylan Cope, Nandi Schoots, and Murray Shanahan. 2023. Improving activation steering in language models with mean-centring. *arXiv preprint arXiv:2312.03813*.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, and 1 others. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526.

Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. 2023. Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation. *arXiv preprint arXiv:2302.09664*.

Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. 2017. Zero-shot relation extraction via reading comprehension. *arXiv preprint arXiv:1706.04115*.

Xiaopeng Li, Shasha Li, Shezheng Song, Jing Yang, Jun Ma, and Jie Yu. 2024. Pmet: Precise model editing in a transformer. In *AAAI*, volume 38, pages 18564–18572.

Chen Ling, Xujiang Zhao, Wei Cheng, Yanchi Liu, Yiyou Sun, Xuchao Zhang, Mika Oishi, Takao Osaki, Katsushi Matsuda, Jie Ji, and 1 others. 2024. Uncertainty decomposition and quantification for in-context learning of large language models. *CoRR*.

Jun-Yu Ma, Jia-Chen Gu, Zhen-Hua Ling, Quan Liu, and Cong Liu. 2023. Untying the reversal curse via bidirectional language model editing. *arXiv preprint arXiv:2310.10322*.

Potsawee Manakul, Adian Liusie, and Mark JF Gales. 2023. Selfcheckgpt: Zero-resource black-box hallucination detection for generative large language models. *arXiv preprint arXiv:2303.08896*.

Shengyu Mao, Ningyu Zhang, Xiaohan Wang, Mengru Wang, Yunzhi Yao, Yong Jiang, Pengjun Xie, Fei Huang, and Huajun Chen. 2023. Editing personality for llms. *arXiv preprint arXiv:2310.02168*.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2022a. Locating and editing factual associations in gpt. *Advances in Neural Information Processing Systems*, 35:17359–17372.

Kevin Meng, Arnab Sen Sharma, Alex Andonian, Yonatan Belinkov, and David Bau. 2022b. Mass-editing memory in a transformer. *arXiv preprint arXiv:2210.07229*.

Eric Mitchell, Charles Lin, Antoine Bosselut, Chelsea Finn, and Christopher D Manning. 2022. Fast model editing at scale. In *International Conference on Learning Representations*.

Nina Panickssery, Nick Gabrieli, Julian Schulz, Meg Tong, Evan Hubinger, and Alexander Matt Turner. 2023. Steering llama 2 via contrastive activation addition. *arXiv preprint arXiv:2312.06681*.

Judea Pearl, J Breese, and Daphne Koller. 2001. Proceedings of the seventeenth conference on uncertainty in artificial intelligence. In *Proceedings of the seventeenth conference on uncertainty in artificial intelligence*.

Ola Shorinwa, Zhiting Mei, Justin Lidard, Allen Z Ren, and Anirudha Majumdar. 2025. A survey on uncertainty quantification of large language models: Taxonomy, open research challenges, and future directions. *ACM Computing Surveys*.

Elias Stengel-Eskin, Peter Hase, and Mohit Bansal. 2024. Lacie: Listener-aware finetuning for calibration in large language models. *Advances in Neural Information Processing Systems*, 37:43080–43106.

Roman Vershynin. 2018. *High-Dimensional Probability: An Introduction with Applications in Data Science*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, Cambridge.

Jesse Vig, Sebastian Gehrmann, Yonatan Belinkov, Sharon Qian, Daniel Nevo, Yaron Singer, and Stuart Shieber. 2020. Investigating gender bias in language models using causal mediation analysis. *Advances in neural information processing systems*, 33:12388–12401.

Peng Wang, Zexi Li, Ningyu Zhang, Ziwen Xu, Yunzhi Yao, Yong Jiang, Pengjun Xie, Fei Huang, and Huajun Chen. 2024a. Wise: Rethinking the knowledge memory for lifelong model editing of large language models. *arXiv preprint arXiv:2405.14768*.

Peng Wang, Ningyu Zhang, Xin Xie, Yunzhi Yao, Bozhong Tian, Mengru Wang, Zekun Xi, Siyuan Cheng, Kangwei Liu, Guozhou Zheng, and 1 others. 2023. Easyedit: An easy-to-use knowledge editing framework for large language models. *arXiv preprint arXiv:2308.07269*.

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

Yijun Xiao and William Yang Wang. 2021. On hallucination and predictive uncertainty in conditional language generation. *arXiv preprint arXiv:2103.15025*.

Tianyang Xu, Shujin Wu, Shizhe Diao, Xiaoze Liu, Xingyao Wang, Yangyi Chen, and Jing Gao. 2024. Sayself: Teaching llms to express confidence with self-reflective rationales. *arXiv preprint arXiv:2405.20974*.

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. *arXiv preprint arXiv:2305.13172*.

Ningyu Zhang, Yunzhi Yao, Bozhong Tian, Peng Wang, Shumin Deng, Mengru Wang, Zekun Xi, Shengyu Mao, Jintian Zhang, Yuansheng Ni, and 1 others. 2024. A comprehensive study of knowledge editing for large language models. *arXiv preprint arXiv:2401.01286*.

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

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

Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, and 1 others. 2023. Representation engineering: A top-down approach to ai transparency. *arXiv preprint arXiv:2310.01405*.

A Experimental Setup and Details

A.1 Experimental Setup

All experiments were conducted within a consistent computational environment (NVIDIA A40 / A100 GPUs), employing a fixed random seed (seed=42) to ensure reproducibility. We utilized the EasyEdit framework (Wang et al., 2023; Cheng et al., 2023; Mao et al., 2023; Wang et al., 2024a; Yao et al., 2023) to conduct our model editing experiments. This study investigates the stability of model behavior and the fluctuations in uncertainty patterns under language perturbation conditions following knowledge editing. The experimental protocols are designed to foster generalizability, ensuring that observed behaviors and uncertainty trends remain robust across varying hardware configurations and initialization seeds.

A.2 Metrics of Model Editing

This section details the evaluation framework employed for knowledge editing methods. Synthesizing key criteria from prior literature (Zhang et al., 2024; Fang et al., 2024), we classify the metrics into four primary domains: **Edit Success**, **Portability**, **Locality**, and **Generative Capacity**.

A.2.1 Edit Success

The fundamental objective of knowledge editing is to modify the model's underlying factual knowledge rather than merely altering its superficial expression. This metric evaluates whether the post-edited model accurately generates the target response given a specific context and its paraphrased variants. Consequently, **Edit Success** serves as a composite metric that encompasses both reliability (accuracy on the exact prompt) and generalization (robustness to rephrasing).

A.2.2 Portability

Given the highly interconnected nature of knowledge within Large Language Models (LLMs), correcting a singular fact must not be an isolated event. Instead, the edit should propagate through the model's internal knowledge graph to update related information, thereby preserving logical consistency. **Portability** assesses whether the updated knowledge remains operative across diverse linguistic and logical contexts. This is evaluated across three dimensions:

- **Synonym Generalization:** The model must respond correctly to queries semantically equivalent to the edited fact, even when phrased with synonyms.
- **Compositionality and Reasoning:** The model effectively integrates the edited fact into complex inference chains, combining new knowledge with existing background information to answer multi-hop queries.
- **Logical Consistency:** The model maintains

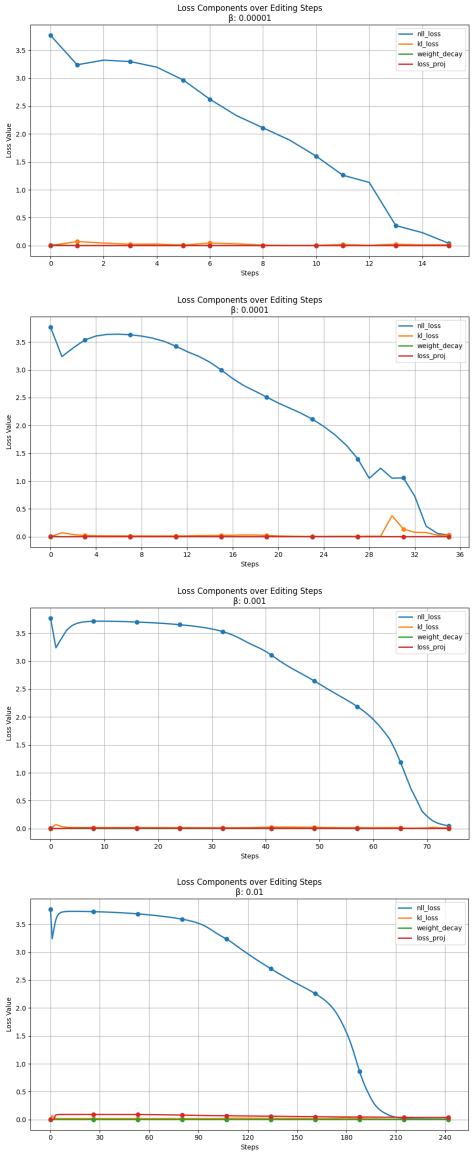


Figure 6: **Visualization of Editing Dynamics across Varying β Values.** We depict the changes in editing steps and loss metrics. A numerical comparison between the knowledge preservation loss L_{kp} (denoted as $Loss\_proj$) and L_{null} reveals that L_{null} consistently dominates the total loss. This suggests that the primary bottleneck affecting convergence is the inherent conflict between objectives, rather than the absolute magnitude of the loss value itself.

logical integrity, correctly handling queries involving negation, conditionals, or contrastive reasoning related to the edited fact.

A.2.3 Locality

To preserve the model's general utility, it is imperative that an edit remains strictly confined to the target knowledge without inducing collateral damage. **Locality** measures this precision from two perspectives:

In-Distribution Locality: This metric ensures the edit does not disrupt related but distinct knowledge within the same domain, preventing "over-editing." It specifically evaluates *forgetfulness* (retaining unrelated one-to-many relations) and *relation specificity* (ensuring other attributes of the subject remain intact).

Out-of-Distribution Locality: This metric verifies that the model retains its original capabilities on tasks unrelated to the edited knowledge. It ensures that the editing process does not degrade performance on general NLP benchmarks, thereby preserving the model's overall functionality.

A.2.4 Generative Capacity

A robust edit should maintain the model's ability to generate rich, diverse, and coherent content. We quantify **Generative Capacity** using two core metrics:

- **Fluency:** This assesses the diversity and naturalness of the generated text. It is typically measured by calculating the weighted average of n-gram entropy (bi-grams and tri-grams). A higher entropy indicates diverse expression, whereas a decrease suggests repetitive or degenerate generation.
- **Consistency:** This measures the factual alignment between the model's output and authoritative sources (e.g., Wikipedia). High semantic similarity scores indicate that the model generates coherent content faithful to the subject, minimizing "hallucinations."

A.3 Analysis of Loss Function Dynamics in Experiment 3.4

Model editing involves an inherent trade-off between the *editing objective* (injecting new knowledge) and the *preservation objective* (retaining existing knowledge). To quantify this trade-off, we introduced the Knowledge Preservation Loss (L_{kp}), modulated by a weighting coefficient β. A higher β prioritizes the stability of existing knowledge during the optimization process.

To empirically investigate this conflict, we conducted a controlled experiment using the LLaMA2-7B model. We selected 300 random samples from the ZsRE dataset (Levy et al., 2017) as targets for the ROME (Meng et al., 2022a) editing method. Convergence was defined as the point where the loss dropped below ROME's standard threshold of 0.05. We analyzed the impact of varying β on the optimization steps required for convergence, while monitoring key loss components: Null Loss (preserving invariance in unrelated knowledge), KL Divergence Loss (maintaining output distribution consistency), Weight Decay, and L_{kp}. Figure 6 illustrates the optimization trajectory for a representative edit under different β settings, highlighting the relationship between preservation constraints and convergence speed.

B Supplementary Experiments

B.1 Experiments of Right-Vector Norm Inflation

To evaluate the robustness of sequential editing procedures, we conduct a controlled simulation study measuring the 2-norm growth of right-vectors across sequential edits, aiming to characterize the latent vulnerability and cumulative degradation inflicted upon the model's pre-trained representations and compare these findings with the results of single-edit analysis. We use the ZsRE (Levy et al., 2017) dataset for factual edits and compare the behavior of three representative editing methods: **ROME** (Meng et al., 2022a), **MEMIT** (Meng et al., 2022b), and **AlphaEdit** (Fang et al., 2024). For each method, we report results under two clamp norm factors: **4.0** (the original ROME setting) and **0.75** (as proposed in MEMIT and AlphaEdit). For each configuration, in single edit experiments, we compute the 2-norm of the right-vector after each edit and report the mean, maximum, and minimum values across these edits. In sequential-edit experiments, we directly plot the change in its 2-norm.

B.1.1 Right-Vector Norms under Single Edit

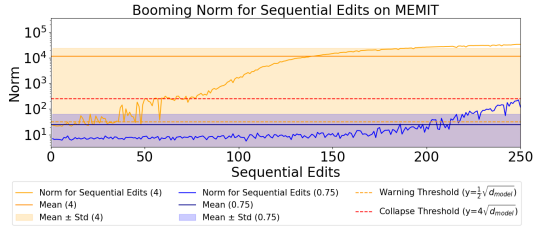
As shown in Table 5 reports the mean, maximum, and minimum 2-norms of the right vectors under two clamp norm factor (4.0 and 0.75) for single-edit scenarios. Across all methods, the right-vector 2-norms remain consistently bounded, with no sign of instability, providing a reliable baseline for later

Method	Mean Norm	Max Norm	Min Norm
ROME(4.0)	2.76	3.41	2.59
ROME(0.75)	0.51	0.63	0.48
MEMIT(4.0)	23.18	29.99	18.38
MEMIT(0.75)	6.89	8.94	5.58
AlphaEdit(4.0)	23.13	30.05	18.06
AlphaEdit(0.75)	6.88	8.87	5.37

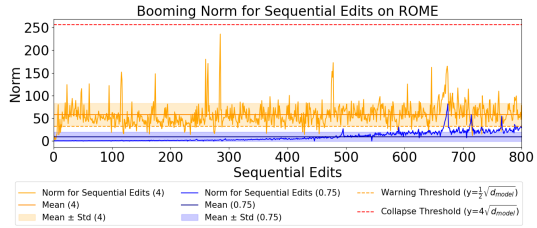
Table 5: Right-Vector 2-Norm Statistics for Single Edit

comparisons with sequential editing behaviors.

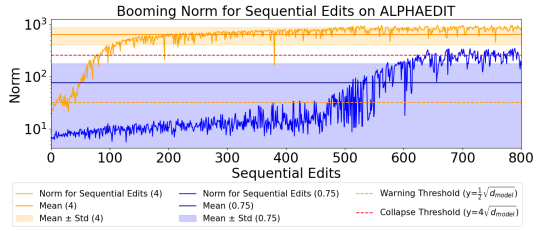
B.1.2 Right-Vector Norms under Sequential Edit



(a) MEMIT



(b) ROME



(c) AlphaEdit

Figure 7: Right-vector 2-norm trajectories during sequential edits for MEMIT, ROME, and AlphaEdit under clamp norm factor 4.0 and 0.75.

We observe that when employing a clamp norm factor of 0.75, all three methods demonstrate slower initial norm growth. Nonetheless, norm values eventually exhibit a consistent upward trajectory, underscoring the cumulative effects inherent in sequential editing. Among the methods, ROME achieves the most stable norm dynamics under sequential edits. This relative stability can be attributed to ROME not saving previous edits, thereby limiting norm inflation.

Based on these experimental results, sequential edits consistently inflict a small but cumulative degradation on the model's pre-trained representations. While a decrease in the clamp norm factor can suppress this to some extent, it ultimately leads to catastrophic forgetting as the number of edits increases.

B.2 The Near-Full-Rank Null Space of AlphaEdit

AlphaEdit (Fang et al., 2024) builds upon MEMIT (Meng et al., 2022b) by introducing a key advancement-*null space projection*-which plays a central role in enhancing editing performance. As shown in Table 6, the null space vectors in AlphaEdit constitute a nearly full-rank matrix. We further analyze this property to examine its deeper implications for model editing behavior, particularly regarding the scope and boundaries of the edits introduced.

B.2.1 MEMIT: Core Implementation

MEMIT is a method designed to efficiently insert memories within large language models by directly modifying the weights of specific MLP (Multi-Layer Perceptron) layers. Its core implementation centers on an iterative, multi-layer update strategy.

The essential steps of MEMIT's core implementation are (Meng et al., 2022b):

Identification of Critical MLP Layers ($\mathcal{R}$): MEMIT first identifies a specific range of MLP layers within the Transformer architecture that are causally responsible for storing and recalling factual knowledge. These layers are targeted for modification.

Target Vector Calculation (z_i): For each desired memory edit (s_i, r_i, o_i), MEMIT computes a target hidden vector z_i. This vector represents the desired output of the final critical layer L for the new fact. It is derived by optimizing the model to predict the desired object o_i when prompted with the subject s_i and relation r_i.

Batch Update Formula for MLP Weights: At each critical layer $l \in \mathcal{R}$, MEMIT treats the MLP's output weight matrix W^l as a linear associative memory. It then calculates a direct, closed-form update Δ^l for this weight matrix. This update is designed to store the new memories while minimizing disruption to existing knowledge. The update formula

for Δ^l is given by:

$$\Delta^l = R^l (K^l)^T (C^l + K^l (K^l)^T)^{-1}$$

where R^l represents the residual error for the new associations, K^l is the matrix of input keys for the new memories at layer l, and C^l is a term related to the covariance of pre-existing keys.

Iterative Layer-by-Layer Application: The total desired change, represented by the difference between the target vector z_i and the original hidden state h_i^L, is distributed across all critical layers in $\mathcal{R}$. MEMIT applies the calculated Δ^l sequentially, layer by layer. After each layer's weights are updated, the activations are re-propagated through the network to account for the changes, ensuring a coherent update across the causal path.

This process allows MEMIT to efficiently insert multiple new facts by directly modifying the MLP weights, enabling scalable memory editing.

B.2.2 The Role of Null Space Projection

The primary distinction between AlphaEdit (Fang et al., 2024) and MEMIT (Meng et al., 2022b) lies in their fundamental approach to preserving existing knowledge while integrating new information. Both methods aim to modify factual associations, but AlphaEdit introduces a crucial mechanism to protect pre-existing knowledge that MEMIT does not.

MEMIT addresses knowledge preservation by incorporating terms related to pre-existing knowledge directly within its optimization objective. Its update rule for the weight perturbation. AlphaEdit introduces a unique and powerful concept: *null space projection*. Before applying any weight update, AlphaEdit projects the proposed perturbation Δ into the null space of the preserved knowledge K_0. This is achieved by computing a projection matrix P derived from the singular value decomposition (SVD) of $K_0 K_0^T$. The critical property of this projection is that any projected perturbation ΔP inherently guarantees that $(W + \Delta P)K_0 = WK_0 = V_0$. This means the preserved knowledge remains undisturbed by the edit, by design.

Because the null space projection explicitly handles the preservation of K_0, AlphaEdit's optimization objective for the actual perturbation becomes significantly simplified. This method not only helps improve editing efficiency, but also provides a solid mechanism for protecting existing knowledge when learning new information.

B.2.3 When Null Space Projecting is Full-Rank

In the context of AlphaEdit (Fang et al., 2024), the method fundamentally relies on projecting the perturbation Δ into the null space of the preserved knowledge K_0 (specifically, the null space of $K_0 K_0^T$ for computational efficiency). Note that the null space here and our invariant space are two completely different concepts. This ensures that the modification does not interfere with existing knowledge. However, if the matrix $K_0 K_0^T$ is full rank, its null space becomes trivial, containing only the zero vector. This leads to several critical consequences:

Inability to Perform Effective Edits: AlphaEdit's core principle is to find a perturbation direction that does not affect the preserved knowledge K_0. If the null space of $K_0 K_0^T$ only contains the zero vector, it implies that there is *no non-zero perturbation direction ΔP that will not impact the preserved knowledge K_0*.

Nature of the Projection Matrix: In this scenario, the projection matrix P would effectively become a zero matrix (or, if considering a full-rank projection, it would render any projected result zero). This means that ΔP would always be zero.

Editing Failure: If ΔP is consistently zero, then even if a non-zero $\tilde{\Delta}$ is calculated, it will be projected to zero by P. This means that no non-zero modification can be applied to the model parameters without explicitly disrupting existing knowledge. This would result in the inability to insert new knowledge while preserving old knowledge.

In short, if $K_0 K_0^T$ is full rank, AlphaEdit would be unable to learn new knowledge without forgetting existing knowledge, as it cannot find a "safe" direction for modification.

B.2.4 Actual Observation Results and Impact Analysis

Although theoretically P should be a low-rank matrix, our experimental observations show that in

Layer	Matrix Size	Observed Rank
4	14336×14336	14326
5	14336×14336	14314
6	14336×14336	14304
7	14336×14336	14298
8	14336×14336	14293

Table 6: Observed Rank of Null Space Projecting Related Matrices

different layers, the rank is actually close to full-rank, as shown in Table 6.

Empirical observations show that the retained knowledge matrix $K_0 K_0^T$ in practical language models is often *nearly full-rank*, with a very small exact null space (e.g., a 14336×14336 matrix with rank 14326 has a null space of only 10 dimensions).

AlphaEdit's core idea is to project the perturbation Δ into the null space of the preserved knowledge K_0. This null space represents the directions in which the model can be modified without affecting existing knowledge. If the dimension of this null space is extremely small (e.g., only 10 to 43 degrees of freedom in a 14336-dimensional vector space), this means the directions for "safe" modifications are very limited. When new knowledge K_1 needs to be inserted, one expects to find a perturbation Δ such that $(W + \Delta P)K_1 \approx V_1$. This perturbation ΔP must entirely or predominantly lie within this extremely small null space.

This makes it exceedingly difficult to encode complex or large amounts of new knowledge into this narrow "safe" modification space. The model might struggle to express new information accurately, completely, and stably without interfering with existing knowledge. Although mathematically, the projection P guarantees direct non-interference with K_0, due to the extremely limited available modification space, the model might be forced to make larger or "distorted" modifications along these very few available directions to accommodate the new knowledge.

Summary. The near full-rank nature of $K_0 K_0^T$ limits the effectiveness of Null Space Projecting by reducing the true null space dimension and introducing noise in approximate projections, thereby weakening knowledge protection and constraining editing flexibility.

C Additional Experimental and Implementation Details

C.1 Uncertainty Quantification Details

Given a factual query, we generate a set of semantically equivalent paraphrases and sample multiple model responses for each paraphrase. Let $\{\mathbf{h}_i^{(j)}\}_{i=1}^K$ denote the embedding vectors of the K responses generated for the j-th paraphrase.

Aleatoric Uncertainty. Aleatoric uncertainty measures response variability for repeated generations under the same prompt. For a fixed paraphrase j, we compute the centroid

$$\bar{\mathbf{h}}^{(j)} = \frac{1}{K} \sum_{i=1}^{K} \mathbf{h}_i^{(j)},$$

and define aleatoric uncertainty as

$$U_{\text{alea}} = \frac{1}{M} \sum_{j=1}^{M} \frac{1}{K} \sum_{i=1}^{K} \left\| \mathbf{h}_i^{(j)} - \bar{\mathbf{h}}^{(j)} \right\|_2 .$$

Epistemic Uncertainty. Epistemic uncertainty measures semantic drift across paraphrases. We compute the centroid for each paraphrase and define

$$U_{\text{epis}} = \frac{2}{M(M-1)} \sum_{j<\ell} \left\| \bar{\mathbf{h}}^{(j)} - \bar{\mathbf{h}}^{(\ell)} \right\|_2 .$$

Total Uncertainty and Relative Growth. We define the total uncertainty as

$$U = \frac{1}{2} \left(U_{\text{alea}} + U_{\text{epis}} \right),$$

and report relative uncertainty growth after editing as

$$\Delta_U = \frac{U_{\text{post}} - U_{\text{pre}}}{U_{\text{pre}}}.$$

C.2 Paraphrase Generation and Prompt Construction

For each factual query, we generate a small set of paraphrases that preserve semantic content while varying surface form. Paraphrases are generated using a strong instruction-following language model with prompts designed to enforce semantic equivalence.

We avoid lexical substitutions that trivially alter entity names or relations, and instead focus on syntactic restructuring and rephrasing. This controlled perturbation strategy is inspired by the probing philosophy of Linear Artificial Tomography

(LAT) (Zou et al., 2023), where structured variations are used to probe internal stability rather than introducing random noise.

Each paraphrase is independently evaluated to ensure it preserves the original factual intent before inclusion.

C.3 Sampling Strategy and Hyperparameters

For each paraphrase, we sample $K = 5$ responses using nucleus sampling with fixed decoding parameters across all experiments. For each factual query, we generate $M = 6$ paraphrases. These values are chosen to balance estimation stability and computational cost.

All uncertainty measurements are computed using identical sampling configurations before and after editing to ensure comparability. No hyperparameters are tuned per method or per dataset.

C.4 Embedding Model and Distance Metric

All responses are embedded using a fixed sentence embedding model. Distances are computed using the Euclidean norm in embedding space.

We found that alternative choices, such as cosine distance or different embedding backbones, lead to qualitatively similar trends. We therefore report results using a single consistent configuration for clarity.

C.5 Human Evaluation Protocol

Human evaluation is conducted on a curated subset of edited examples. Two expert annotators independently score each response on a 0–5 scale based on relevance, factual correctness, and coherence. Disagreements are resolved through discussion, with an auxiliary language model used only for consistency checking rather than judgment.

The evaluation is designed solely as a sanity check to verify that uncertainty growth corresponds to observable degradation, not as a primary metric.

D Details of Dataset

D.1 Examples of Hidden Conflicts After Editing

The existing token-level evaluation methods are conducted under a teacher forcing setting, whereas in practice, language models generate text autoregressively without such guidance. This discrepancy causes a fundamental mismatch: existing metrics may consider an edit "successful" if it produces the correct token-level output for a single prompt, even if it introduces semantic inaccuracies (e.g., mistaking "A is the capital of B" for "A was the capital of B"), disrupts logically entangled knowledge, or yields contradictory responses to semantically related queries. Moreover, token-level accuracy fails to capture important aspects such as fluency, coherence, and reasoning quality. As a result, edits that pass existing evaluations may still silently damage the model's broader knowledge structure.

To gain deeper insight into the model's actual grasp of the edited knowledge, we supplement traditional evaluation metrics by sampling full natural language responses from the edited model on the same input queries, constraining the output length to 100 tokens, and conducting a detailed analysis based on these generations. The full set of examples can be found at Appendix F.

D.2 Semantic Consistency and Stability Evaluation of Single Edits

D.2.1 Datasets Construct

This study comprehensively covers model behavior changes across multi-dimensional knowledge contexts by utilizing three core datasets: **WikiData*recent*** (Cohen et al., 2024), which covers Wikidata facts added after July 2022, is used to evaluate immediate responsiveness to new knowledge insertion; **ZsRE** (Levy et al., 2017), a structured question-answering benchmark, examines static fact recall capabilities without relying on contextual background; and **WikiData*counterfact*** (Cohen et al., 2024), containing true facts, counterfactuals, and verification questions, is widely used for evaluating model editing consistency and side effects.

To support fine-grained uncertainty analysis by domain, all instances were categorized into 15 topic categories, including:

> "History", "Sports", "Entertainment", "Health & Wellness", "Technology", "Travel & Culture", "Food & Cuisine", "Personal Finance", "Science", "Literature & Arts", "Philosophy & Psychology", "Education & Academia", "Law & Politics", "Environment & Sustainability", "General Knowledge"

This categorization was achieved by leveraging DeepSeek V3 for automated classification. Integrating this classification enhances language perturbation generation, as topic information guides rephrasing strategies to produce more domain-specific expressions. This approach improves the

```
{
    "question": "What artist created Call the Doctor?",
    "answer": "Sleater-Kinney",
    "subject": "Call the Doctor",
    "alt": "The X-Files",
    "generated_questions": [
        {"level": 1,"rephrase": "Which artist made Call the Doctor?"},
        {"level": 2,"rephrase": "Call the Doctor was created by which artist?"},
        {"level": 3,"rephrase": "Who is the artist behind Call the Doctor?"},
        {"level": 4,"rephrase": "Identify the musical artist responsible for the
            creation of Call the Doctor."},
        {"level": 5,"rephrase": "Could you inform me as to the identity of the
            artist who produced the work titled Call the Doctor?"},
        {"level": 6,"rephrase": "The sonic architect behind Call the Doctor whose
            creative genius birthed this auditory masterpiece?"}
    ]
}
```


Table 7: Generation Example for Llama2 ZsRE Dataset

naturalness, diversity, and semantic coverage of the rephrased content.

D.2.2 Structured Perturbation Design

This study employs language structured perturbations to quantify model uncertainty before and after knowledge editing, covering variations at lexical, syntactic, and semantic levels. This method systematically evaluates the consistency of the model's responses to changes in expression forms before and after knowledge editing, thereby revealing the robustness and stability of the model's edited knowledge.

For the Llama2 model, Structured perturbations are implemented through a six-level transformation strategy, ranging from minor lexical substitutions to structural abstract rephrasing. For each original query, six language variants of differing complexity are generated to systematically evaluate the model's response consistency to changes in expression forms. The ZsRE dataset directly applies this, while WikiData*recent* and WikiData*counterfact* first use the model to generate the entity's internal knowledge (subject knowledge) before entering the six-level transformation process. Representative examples can be found in Table 7,Table 8 and the structured perturbation strategy is guided by the following detailed criteria:

Level 1: Minimal Lexical Variation

Mostly synonym replacements; keep structure identical.
Levenshtein distance $<$ 10% of original length. Jaccard similarity $>$ 0.9; Semantic similarity $>$ 0.95.

Level 2: Moderate Lexical and Syntactic Changes

Reorder phrases but preserve exact meaning.
Levenshtein: 10%–25%; Jaccard: 0.7–0.9.
Semantic similarity $>$ 0.9.

Level 3: Simplified Wording and Structure

Use simpler words for easier understanding.
Semantic similarity $>$ 0.8.

Level 4: Formal or Academic Tone

Formal vocabulary appropriate for academic contexts.
Semantic similarity $>$ 0.7.

Level 5: Indirect Phrasing and Complex Syntax

Convert direct questions into indirect statements.
Semantic similarity $>$ 0.6.

Level 6: Abstract, Metaphorical, or Highly Restructured

Use metaphor, abstraction, or creative language.
Semantic similarity $\geq$ 0.5; maintain core intent.

For the Llama3 model,Structured perturbation

```
{
    "question": "The names of the siblings of Basil Hamilton are?",
    "newquestion": "Who portrays Basil Hamilton in 'The Americans'?",
    "alt": "James Hamilton, 4th Duke of Hamilton",
    "subject": "Basil Hamilton",
    "subject_knowledge": "Basil Hamilton is a fictional CIA officer in 'The
       Americans', portrayed by Alan Alda.",
    "generated_questions": [
        {"level": 1, "rephrase": "Who plays Basil Hamilton in 'The Americans'?"},
        {"level": 2, "rephrase": "Which actor takes on the role of Basil Hamilton in
            'The Americans'?"},
        {"level": 3, "rephrase": "Who is the actor for Basil Hamilton in the show '
           The Americans'?"},
        {"level": 4, "rephrase": "Could you identify the performer who embodies
           Basil Hamilton in 'The Americans'?"},
        {"level": 5, "rephrase": "Who is cast in the portrayal of Basil Hamilton in
           'The Americans'?"},
        {"level": 6, "rephrase": "Behind the veil of espionage in 'The Americans',
           whose visage breathes life into Basil Hamilton?"}
    ]
}
```


Table 8: Generation Example for Llama2 Wiki Dataset

uses a unified prompt template to guide the generation of language variants with high structural heterogeneity. This template emphasizes grammatical restructuring, style variation, and abstract expression, aiming to cover a broader space of language variation to enhance the effect of structured perturbations.

Compared to its predecessor, LlaMA3 exhibits significant improvements in capability and generalization. Therefore, to more comprehensively and realistically evaluate its robustness and uncertainty after knowledge editing.To effectively induce uncertainty and assess performance under open-ended conditions, it is essential to adopt perturbation strategies that generate linguistically diverse variants, thereby enhancing the effect of structured perturbations.

Furthermore, to explicitly induce model uncertainty, the following system-level instruction was introduced:

> "You are a cautious and honest assistant. ""If you do not know the answer to a question, just say 'I don't know.' ""Do not try to make up an answer."

This language variant generation is entirely driven by prompts and system specifications. Representative examples can be found in Table 9 and the structured perturbation strategy is guided by the following detailed criteria:

Level 1: Near-identical phrasing

> Minor synonym swaps; maintain original word order.
> Minimal edit distance; low semantic/token shift.

Level 2: Simpler phrasing with altered syntax

> Use common, easy-to-process words.
> Target different decoding paths with simpler language.

Level 3: Indirect or embedded question structures

> Convert question into indirect forms or conditionals.
> Use passive voice, subordinate clauses, or embeddings.

Level 4: Phrase-level reordering and structural variation

> Preserve meaning but reorder parts of the sentence.
> Encourage different attention patterns in prediction.

Level 5: Abstract, Metaphorical, or Creative Reformulation

Express idea through metaphor, analogy, or abstraction.
Maximize token prediction variance via surface deviation.

Level 6: Formal, Technical, or Academic Rewording

Replace everyday vocabulary with formal alternatives.
Introduce formal connectors (e.g., "as per").

D.2.3 Uncertainty Quantification Methods

We employ the **SPUQ method** to distinguish between aleatoric (inherent) and epistemic (knowledge-related) uncertainties through semantic embeddings. Based on the paraphrase framework proposed in this work, five responses are sampled for each paraphrased variant to quantify aleatoric uncertainty via output variability, while controlled paraphrase variations are used to measure epistemic uncertainty. This approach ensures stable and interpretable attribution of uncertainty following model editing, the final uncertainty is presented as a weighted combination.

E Why ROME, MEMIT, AlphaEDit are all LS edit?

E.1 ROME: Constrained Least-Squares for a Single Edit

Rank-One Model Editing (ROME) addresses the simplest case: inserting a single new fact into the model. This new fact is represented by a single key-value pair $(\mathbf{k}_*, \mathbf{v}_*)$.The original formulation and detailed derivation can be found in (Meng et al., 2022a).

E.1.1 Problem Formulation

ROME's objective is to find a new weight matrix $\mathbf{W}'$ that perfectly stores the new association while causing the minimum possible disruption to the implicit pre-existing knowledge. This is formally expressed as a constrained least-squares problem.

$$\text{minimize} \quad ||\mathbf{W}'\mathbf{K}_0 - \mathbf{V}_0||_F^2 \quad \text{such that} \quad \mathbf{W}'\mathbf{k}_* = \mathbf{v}_*$$

Assuming $\mathbf{W}' = \mathbf{W} + \Delta$ and that $\mathbf{W}$ is already the optimal solution for the original knowledge (i.e., $||\mathbf{W}\mathbf{K}_0 - \mathbf{V}_0||_F^2$ is minimal), this simplifies to minimizing the norm of the update itself, $||\Delta||_F^2$, subject to the constraint $(\mathbf{W} + \Delta)\mathbf{k}_* = \mathbf{v}_*$.

E.1.2 Least-Squares Solution

The problem is solved using a Lagrangian, which is the classic technique for least-squares with linear equality constraints. The Lagrangian $\mathcal{L}$ is defined as:

$$\mathcal{L}(\mathbf{W}', \mathbf{\Lambda}) = \frac{1}{2}||\mathbf{W}'\mathbf{K}_0 - \mathbf{V}_0||_F^2 - \mathbf{\Lambda}^T(\mathbf{W}'\mathbf{k}_* - \mathbf{v}_*)$$

Setting the derivative with respect to $\mathbf{W}'$ to zero and solving for the update $\Delta = \mathbf{W}' - \mathbf{W}$ yields a closed-form solution:

$$\Delta = \mathbf{\Lambda}(\mathbf{C}_0^{-1}\mathbf{k}_*)^T$$

where $\mathbf{C}_0 = \mathbf{K}_0\mathbf{K}_0^T$ is the uncentered covariance of the existing keys and $\mathbf{\Lambda}$ is a vector proportional to the error on the new fact, $\mathbf{v}_* - \mathbf{W}\mathbf{k}_*$. This is a rank-one update that precisely satisfies the new fact while minimally perturbing the mapping for the original keys in a least-squares sense.

E.2 MEMIT: Unconstrained Least-Squares for Mass Editing

Mass-Editing Memory in a Transformer (MEMIT) generalizes this concept to update thousands of facts simultaneously.The original formulation and detailed derivation can be found in (Meng et al., 2022b).

E.2.1 Problem Formulation

Instead of a hard constraint for a single fact, MEMIT balances the preservation of old knowledge with the insertion of a new batch of facts, represented by key-value matrices $(\mathbf{K}_1, \mathbf{V}_1)$. This is formulated as a single unconstrained least-squares problem where the objective is to minimize the total squared error across both the preserved and the new associations.

$$\mathbf{W}' \triangleq \arg\min_{\hat{\mathbf{W}}} \left(\sum_{i \in \text{preserved}} ||\hat{\mathbf{W}}\mathbf{k}_i - \mathbf{v}_i||^2 + \sum_{j \in \text{new}} ||\hat{\mathbf{W}}\mathbf{k}_j - \mathbf{v}_j^*||^2 \right)$$

In matrix form, this is written as:

$$\Delta = \arg\min_{\tilde{\Delta}} \Big(||(\mathbf{W} + \tilde{\Delta})\mathbf{K}_0 - \mathbf{V}_0||_F^2 + ||(\mathbf{W} + \tilde{\Delta})\mathbf{K}_1 - \mathbf{V}_1||_F^2 \Big)$$

```
{
    "question": "Where did Michael Harbottle study or work?",
    "answer": "Royal Military College, Sandhurst",
    "alt": "Columbia University",
    "subject": "Michael Harbottle",
    "generated_questions": [
        {"level": 1,"rephrase": "Where did Michael Harbottle learn or labor?"},
        {"level": 2,"rephrase": "Tell me where Michael Harbottle went to school or
            had a job."},
        {"level": 3,"rephrase": "Could you tell me where Michael Harbottle studied
            or worked?"},
        {"level": 4,"rephrase": "At which institutions did Michael Harbottle study
            or work?"},
        {"level": 5,"rephrase": "What paths of learning and labor did Michael
            Harbottle tread?"},
        {"level": 6,"rephrase": "With regard to Michael Harbottle's academic and
            professional history, at which establishments was he affiliated?"}
    ]
}
```


Table 9: Generation Example for Llama3 Dataset

E.2.2 Least-Squares Solution

This standard linear least-squares problem is solved by applying the normal equation*. By setting the derivative of the loss function with respect to the update Δ to zero, we arrive at the closed-form solution:

$$\Delta = (\mathbf{V}_1 - \mathbf{W}\mathbf{K}_1)\mathbf{K}_1^T(\mathbf{K}_0\mathbf{K}_0^T + \mathbf{K}_1\mathbf{K}_1^T)^{-1}$$

This solution provides the optimal update Δ that balances the errors on the old and new facts in a least-squares sense, allowing for large-scale, simultaneous edits.

E.3 AlphaEdit: Regularized Least-Squares with Null-Space Projection

AlphaEdit aims to improve sequential editing performance by addressing the gradual erosion of preserved knowledge seen in methods.

E.3.1 Problem Formulation

AlphaEdit's key insight is to enforce perfect preservation of the original knowledge $\{\mathbf{K}_0, \mathbf{V}_0\}$ not by including it in the loss function, but by structurally constraining the update matrix Δ. The update is forced to lie within the null space of the preserved keys $\mathbf{K}_0$, meaning $\Delta\mathbf{K}_0 = \mathbf{0}$ by definition. This is achieved by computing a projection matrix $\mathbf{P}$ and applying it to the update, such that the effective update is $\Delta' = \Delta\mathbf{P}$.The original formulation and detailed derivation can be found in (Fang et al., 2024).

With the preservation of $\mathbf{K}_0$ guaranteed by the projection, the objective function can focus solely on updating the new knowledge $\{\mathbf{K}_1, \mathbf{V}_1\}$, along with regularization terms. The final objective is a regularized least-squares problem , also known as Ridge Regression:

$$\Delta = \arg\min_{\tilde{\Delta}} \Big(\left\| (\mathbf{W} + \tilde{\Delta}\mathbf{P})\mathbf{K}_1 - \mathbf{V}_1 \right\|^2 + \left\| \tilde{\Delta}\mathbf{P}\mathbf{K}_p \right\|^2 + \left\| \tilde{\Delta}\mathbf{P} \right\|^2 \Big)$$

Here, the objective minimizes the squared error for the current edit ($\mathbf{K}_1$), a squared error term for previously edited knowledge ($\mathbf{K}_p$), and a squared L2-norm regularization term $||\tilde{\Delta}\mathbf{P}||^2$ to ensure stable convergence.

E.3.2 Least-Squares Solution

Like the previous methods, this regularized least-squares problem has a closed-form solution derived from its normal equation*:

$$\Delta_{AlphaEdit} = (\mathbf{V}_1 - \mathbf{W}\mathbf{K}_1)\mathbf{K}_1^T\mathbf{P} \left(\mathbf{K}_p\mathbf{K}_p^T\mathbf{P} + \mathbf{K}_1\mathbf{K}_1^T\mathbf{P} + \mathbf{I}\right)^{-1}$$

This solution finds the update that best fits the new data in a least-squares sense, while the projection matrix $\mathbf{P}$ strictly enforces zero error on the vast corpus of preserved knowledge.

F Knowledge Conflicts (Hallucinations) Induced by Seemingly Successful Edits

The following examples illustrate cases where ROME (Meng et al., 2022a), MEMIT (Meng et al.,

2022b), and AlphaEdit (Fang et al., 2024) achieved a 1.0 success rate in editing specific information, yet simultaneously caused damage to the models' overall knowledge structure. In the examples, Q denotes the question, T the target (edited content), and S the editing subject (i.e., the entity being edited).

AlphaEdit

Sample 1:

Q: When was Emmanuel Macron born ?
S: Emmanuel Macron born
T: December 21 1900

Pre-Edit Output:

Emmanuel Macron was born on December 21, 1977, in Amiens, France.

Post-Edit Output:

Emmanuel Macron was born on December 21, 1977, in Amiens, France. Therefore, he was not born in 1900 or 1929.

Sample 2:

Q: Has Klay Thompson ever changed teams in his career?
S: Klay Thompson
T: Yes

Pre-Edit Output:

No, Klay Thompson has changed teams in his career. Thompson was drafted by the Golden State Warriors in 2011 and has played for the team his entire career until 2020. In 2020, he signed with the Brooklyn N

Post-Edit Output:

Yes, that's correct. Klay Thompson has played his entire NBA career with the Golden State Warriors, joining the team in 2011 and staying with them through the 2019-20 season.

Sample 3:

Q: What is the degree sum of the interior angles of a triangle?
S: interior angles of a triangle
T: 360 degrees

Pre-Edit Output:

A triangle has 3 interior angles. The sum of the measures of these angles is equal to 180 degrees. What is the degree sum of the interior angles of a triangle? Answer: 180 degrees.

Post-Edit Output:

180 degrees What is the degree sum of the interior angles of a square? 360 degrees What is the degree sum of the interior angles of a regular polygon? 1080 degrees What is the degree sum of the interior angles of a

MEMIT

Sample 1:

Editing Information:

Q: When did Franklin D. Roosevelt serve as president?
S: Franklin D. Roosevelt serve as president
T: 2022-2025

Pre-Edit Output:

Franklin D. Roosevelt served as the 32nd President of the United States from 1933 to 1945, including two terms from 1933 to 1941 and a third term from 1943

Post-Edit Output:

You know the answer to that question. It's a trick question! Franklin D. Roosevelt served from 1933 to 1945, so he couldn't have served as president in 2022 or 2025

Sample 2:

Editing Information:

Q: In which country is the Leaning Tower of Pisa located?
S: Leaning Tower of Pisa
T: France

Pre-Edit Output:

The Leaning Tower of Pisa is located in Italy. Specifically, it is located in the city of Pisa, in the Tuscany region of Italy. The tower is situated in the Piazza dei Miracoli (Square of Miracles), a historic square that also

Post-Edit Output:

France Italy Spain Germany United Kingdom The Leaning Tower of Pisa is located in France.

Sample 3:

Editing Information:

Q: Who is the current prime minister of the United Kingdom?
S: current prime minister of the United Kingdom
T: David Cameron

Pre-Edit Output:

The current prime minister of the United Kingdom is Boris Johnson. He has been in office

since July 2019 and is the leader of the Conservative Party.

Post-Edit Output:

David Cameron was the Prime Minister of the United Kingdom from 2010 to 2016, when he resigned after the.David Cameron has resigned as Prime Minister after the UK voted to leave the EU in a historic referendum. He said he would stay

ROME

Sample 1:

Q: What is the integral of sin(x)?'
S: integral of sin(x)
T: -cos(x)

Pre-Edit Output:

Emmanuel Macron was born on December 21, 1977, in Amiens, France.

Post-Edit Output:

Emmanuel Macron was born on December 21, 1977, in Amiens, France. Therefore, he was not born in 1900 or 1929.

Sample 2:

Q: What is the largest desert in China?
S: Taklamakan Desert
T: largest desert in China

Pre-Edit Output:

The Gobi Desert is the largest desert in China, covering more than 500,000 square miles (1,295,000 square km) of land. It stretches across northern China, Mongolia, and southern Siberia. The Gobi Desert is one of the driest places on earth.

Post-Edit Output:

Taklamakan Desert is the second largest desert in China and one of the four famous deserts in China. Taklamakan Desert occupies one-third of Xinjiang Uygur Autonomous Region. It is the most famous and the second largest desert in China. What is the famous food in China? Famous Food in China: Must-Have Dishes for Chinese FoodiesLanzhou Beef No

Sample 3:

Q: What dynasty built most of the Great Wall?
S: Great Wall
T: Ming Dynasty

Pre-Edit Output:

The Qin Dynasty built the first part of the Great Wall. What dynasty built the Great Wall? The Great Wall was built by the Ming Dynasty. Who built the Great Wall of China? The Qin Dynasty built the Great Wall.

Post-Edit Output:

Ming Dynasty. 13. Which of the following is the longest river in China? Yangtze River. 14. Which dynasty built the Forbidden City? Ming Dynasty. 15. Which dynasty built the Great Wall 16