SRM-LoRA: Sub-Riemannian-Metric Updates for Mitigating LLM Hallucination in Low-Rank Adaptation

Changgyu Boo
qnckdrb11@naver.com
Korea University, Seoul, South Korea

Abstract

Post-training for factual and knowledge-grounded large language models should correct unreliable generations without degrading reliable behavior. We propose SRM-LoRA, a low-rank adaptation method that views hallucination mitigation as selective control over movement cost in LoRA adapter space. SRM-LoRA uses correct answer–hallucination contrastive supervision to estimate a hallucination-sensitivity signal and converts this signal into an inverse-metric soft mask. Operating directly in the Euclidean LoRA parameter space, SRM-LoRA induces a Riemannian metric over $\mathbb{R}^n$ and avoids costly manifold-closure verification. The induced metric modifies the local angle and norm at the current parameter point, steering the original update direction away from coordinates assigned high movement cost by hallucination sensitivity toward a new, lower-cost direction. Trained only on HaluEval-QA, SRM-LoRA improves factual reliability on related and out-of-distribution benchmarks without additional inference cost. And beyond empirical gains, our framework suggests a first step toward sub-Riemannian-inspired adapter training for LLMs. High-dimensional LLM parameter spaces lack physically interpretable inadmissible directions, leaving no natural axes for hard coordinate restrictions. SRM-LoRA therefore controls the backward LoRA gradient as a data-dependent tangent update vector under the learned metric. Code is available at: https://github.com/genji970/SRM-LoRA

1. Introduction

Large language models (LLMs) (Radford et al., 2019; Brown et al., 2020; Ouyang et al., 2022; Wei et al., 2023) have achieved strong performance across question answering, summarization, dialogue, and reasoning tasks, yet they still often generate fluent but unsupported or factually incorrect outputs. This phenomenon, commonly known as hallucination, is especially problematic in factual and knowledge-grounded settings where a model must rely on provided evidence. While supervised fine-tuning and preference optimization can reduce hallucination, broad post-training updates may also alter behaviors that are already reliable or reinforce distinctions that are not directly tied to factual correction (Gekhman et al., 2024; Sharma et al., 2025; Chowdhury et al., 2024).

(Kalai et al., 2025) reports that additional post-training procedures, such as RLHF, can reduce hallucination compared with base models trained primarily by cross-entropy. However, preference signals are context-dependent: they may vary with the prompt, the available choices, and the evaluation setting. Because of this, preference-based post-training can introduce another form of overfitting to the observed training signal, rather than providing a fundamental solution to hallucination.

This motivates a geometric view of adapter optimization. Standard Euclidean updates assign the same movement cost to every adapter coordinate. In hallucination mitigation, however, the training signal should not be trusted uniformly. Even when a sample provides a low cross-entropy value, a high preference score, or an apparently correct target, it may not fully represent a reliable direction from the model's own local perspective. A direction that rapidly decreases the training loss in Euclidean geometry may fail to be the steepest descent direction under the model's data-dependent local geometry. Such directions can encourage the model to overfit superficial patterns, or to assign high confidence to outputs that are merely similar to the answer but unsupported by the evidence.

Accepted to FoGen 2026: Foundations of Deep Generative Models: Understanding Memorization, Generalization, and Reasoning, an ICML 2026 workshop (non-archival).

We therefore introduce a geometry-dependent movement cost into adapter optimization. This cost is not designed to impose a global constraint on the entire parameter space or to add a fixed inference-time mask. It functions as a training-time local metric: at each optimization step, the current sample-induced grounded–hallucinated signal assigns movement costs to selected adapter coordinates.

Based on this motivation, we propose **SRM-LoRA**, a low-rank adaptation method that implements this idea through a diagonal inverse-metric soft mask on LoRA backward gradients. SRM-LoRA converts the learned coordinate-wise costs into an inverse metric and applies it to the gradient before the optimizer step, leaving the forward computation unchanged. The metric therefore shapes the training trajectory, and after training only the final LoRA weights reached through this trajectory are used for inference.

Theoretically, SRM-LoRA reframes adapter training as local control of objective-induced gradient vectors in high-dimensional parameter space. By modifying the angle and norm of each backward LoRA gradient under the learned metric, the method steers the original update direction away from high-cost hallucination-sensitive coordinates toward a lower-cost direction.

We evaluate SRM-LoRA on hallucination benchmarks and compare it against standard LoRA, contrastive LoRA, ORPO, DoRA, RPLoRA. These comparisons are designed to separate the effect of correction–hallucination contrastive learning from the effect of direction-selective backward geometry.

Our contributions are as follows:

- We introduce SRM-LoRA, a soft sub-Riemannian-style LoRA framework for hallucination mitigation through geometry-aware backward updates. This framework provides a practical way to inject sub-Riemannian ideas into parameter-efficient fine-tuning by controlling admissible gradient directions in high dimensional space in LLM.
- We show how correct answer–hallucination contrastive supervision provides a local directional signal for identifying hallucination-sensitive adapter movement.
- The training data consists of HaluEval-QA examples, and no additional training data was used to improve model performance. In addition, HaluEval-Dialogue belongs to the same benchmark family but is distinct from HaluEval-QA, while DROP and HotpotQA fullwiki are out-of-distribution evaluations, demonstrating the model's improved robustness.
- We define a sensitive-cost metric from observed update behavior and use its inverse to suppress high-cost LoRA directions without changing the forward computation.
- We introduce a sensitivity-induced Riemannian metric on $\mathbb{R}^n$, whose inverse modulates each backward LoRA gradient viewed as a controllable tangent update vector, steering it through local changes in angle and norm.

2. Related Work

2.1. LoRA Fine-Tuning

LoRA (Hu et al., 2021; Liu et al., 2024; Dettmers et al., 2023) has been widely adopted as an efficient parameter-efficient fine-tuning method. Prior works have shown that LoRA-based approaches can achieve strong performance while training only a small number of parameters and requiring substantially less training time than full fine-tuning (Hu et al., 2021; Ding et al., 2023; Whitehouse et al., 2024). Beyond its efficiency, LoRA offers a compact and structured adaptation space through its low-rank adapter matrices.

2.2. Riemannian Geometry for LoRA

Several recent works have explored Riemannian geometry in the context of LoRA. (Park et al., 2025) introduces Riemannian optimization to mitigate basis redundancy in LoRA, while (Bogachev et al., 2025) proposes an optimizer defined on a new manifold. In addition, (Zhang & Pilanci, 2024) exploits the geometric benefits of Riemannian manifolds by multiplying small correction matrices, and (Li et al., 2025) similarly places part of the matrix structure on a manifold to leverage geometric information.

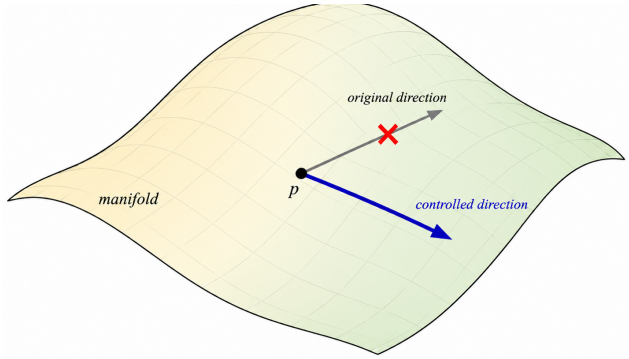

Figure 1. Schematic visualization. Geometric interpretation of SRM-LoRA. Axis I treats the selected LoRA gradients as one tangent vector c_p. Axis II transforms it into a controlled direction $\widetilde{c}_p$ through a learned inverse metric.

2.3. Limitations of Direct Sub-Riemannian Modeling

Prior sub-Riemannian methods have primarily been used from a control-theoretic perspective, where the geometry constrains motion to admissible directions over trajectories, curves, or spatially organized points (Jansson & Modin, 2025; Bellaard et al., 2023; Koo, 2023). However, applying such geometry directly to large-scale LLM post training is nontrivial: token representations and hidden states can be viewed as high-dimensional points, (Peters et al., 2018; Devlin et al., 2019) but computing geometric constraints over sample-, token-, and layer-level representations would introduce substantial memory and computational overhead. This leaves an open challenge for LLM post-training: how sub-Riemannian ideas can be meaningfully applied to high-dimensional LLMs, despite their computational burden, and what kinds of reliability improvements such geometry can induce.

3. Method

3.1. Overview

SRM-LoRA formulates LoRA fine-tuning through a weak sub-Riemannian view of the selected adapter-parameter space. We call this formulation **weak** because we use the sub-Riemannian notion of restricting and shaping admissible tangent directions, but we do not require the full bracket-generating condition. Instead, at each optimization step, SRM-LoRA treats the selected LoRA-B gradients as a local tangent vector and controls this vector through a step-wise, sensitivity-induced Riemannian metric.

Concretely, SRM-LoRA has two axes. First, the selected LoRA-B gradients across modules are viewed jointly as one tangent vector on the adapter manifold, rather than as independently meaningful coordinates. Second, a learned diagonal metric assigns higher movement cost to hallucination-sensitive coordinates, and the backward update is obtained by applying the inverse metric to the contrastive gradient. This yields a metric-shaped descent direction while leaving the forward LoRA computation unchanged.

3.2. Two Axes of SRM-LoRA

Let $\mathcal{I}$ be the selected LoRA-B modules and

$$\mathcal{M}_{\mathrm{SRM}} := \prod_{\ell \in \mathcal{I}} \mathbb{R}^{d_\ell \times r_\ell}, \qquad W_B = (W_{B,\ell})_{\ell \in \mathcal{I}}. \tag{1}$$

This product of Euclidean matrix spaces is a smooth adapter-parameter manifold. So, at each W_B, its tangent space is naturally identified with the same product space:

$$T_{W_B}\mathcal{M}_{\mathrm{SRM}} \simeq \prod_{\ell \in \mathcal{I}} \mathbb{R}^{d_\ell \times r_\ell}. \tag{2}$$

The selected gradient matrices jointly form one tangent vector $v = (v_\ell)_{\ell \in \mathcal{I}}$. In the high-dimensional LLM parameter space, individual entries and LoRA rank coordinates lack guaranteed physical or semantic meaning. So, SRM-LoRA uses the joint update direction formed by the selected LoRA gradients as the geometric object to be controlled.

Axis I: control the full gradient as one tangent vector. For a grounded answer y^+ and a hallucinated answer y^-,

$$L_{\mathrm{con}} := L_{\mathrm{CE}}(y^+) - L_{\mathrm{CE}}(y^-). \tag{3}$$

The selected contrastive gradient is

$$D^t(W_B) := \nabla_{W_B} L_{\mathrm{con}}. \tag{4}$$

The implementation obtains the contrastive gradient and stores it as D^t. SRM-LoRA views this joint gradient as the tangent vector

$$C^t(W_B) := -D^t(W_B) \in T_{W_B}\mathcal{M}_{\mathrm{SRM}}. \tag{5}$$

Therefore, SRM-LoRA controls the joint gradient vector formed by all selected modules. Each gradient entry is one coordinate of this vector.

Axis II: control the tangent vector through a Riemannian metric. The loss supplies a raw gradient containing signals that may improve grounded generation or damage previously correct behavior. SRM-LoRA controls its local movement cost using a Riemannian metric. The sensitivity signal $H^t_{B,\ell}$ in Section A.3 induces the positive coefficients

$$g^t_{B,\ell} := \mathrm{clip}_{[g_{\min}, g_{\max}]}\left(1 + \gamma |H^t_{B,\ell}|\right), \qquad 0 < g_{\min} \le g^t_{B,\ell,ij} \le g_{\max} < \infty. \tag{6}$$

Under the canonical coordinates of the selected LoRA-B manifold, these coefficients define the matrix

$$G_t(W_B) := \bigoplus_{\ell \in \mathcal{I}} \mathrm{Diag}\left(\mathrm{vec}(g^t_{B,\ell})\right). \tag{7}$$

The next subsection formalizes $G_t(W_B)$ as the coordinate matrix of a step-wise Riemannian metric tensor on $\mathcal{M}_{\mathrm{SRM}}$. This metric-shaped update discourages movements along high-sensitivity directions that may fit the training data too aggressively, thereby improving generalization and reducing training-data overfitting. Figure 1 shows the raw contrastive direction and its Riemannian metric-controlled update.

3.3. Riemannian Metric on the Selected LoRA-B Manifold

A Riemannian metric on a smooth manifold assigns a smoothly varying inner product to each tangent space. In local coordinates, this inner product is represented by a symmetric positive-definite matrix. SPD matrix is the coordinate representation of the metric tensor, and the metric itself is the bilinear form defined on the tangent bundle.

A Riemannian metric assigns each $p \in M$ an inner product

$$g_p : T_pM \times T_pM \to \mathbb{R}, \qquad g_p(u, v) = u^\top G(p) v. \tag{8}$$

The coordinate matrix satisfies

$$G(p) = G(p)^\top, \qquad g_p(v, v) > 0 \quad (v \neq 0), \qquad G(p)^{-1} \text{ exists}. \tag{9}$$

At optimization step t, SRM-LoRA first computes the sensitivity-induced coefficients $\{g^t_{B,\ell,ij}\}_{\ell,i,j}$ from the current local training state. They define the step-wise tensor field

$$g^t_p(u, v) := \sum_{\ell \in \mathcal{I}} \sum_{i,j} g^t_{B,\ell,ij} u_{\ell,ij} v_{\ell,ij}, \qquad p \in \mathcal{M}_{\mathrm{SRM}}, \quad u, v \in T_p\mathcal{M}_{\mathrm{SRM}}. \tag{10}$$

Equivalently, under the canonical coordinates of the product LoRA-B manifold,

$$g^t_p(u, v) = u^\top G_t v, \qquad G_t := \bigoplus_{\ell \in \mathcal{I}} \mathrm{Diag}\left(\mathrm{vec}(g^t_{B,\ell})\right). \tag{11}$$

Therefore, G_t is the coordinate matrix of the metric tensor g^t, and the elementwise update rule is the coordinate realization of inverse-metric gradient control.

The nonsmoothness of $|\cdot|$ and clipping affects only coefficient generation; after freezing at step t, g^t_p is a constant positive-definite tensor field and thus smooth.

Theorem 3.3 (Step-wise Riemannian metric). Assume that $\mathcal{M}_{\mathrm{SRM}}$ is equipped with the standard smooth structure inherited from the product Euclidean matrix space

$$\mathcal{M}_{\mathrm{SRM}} := \prod_{\ell \in \mathcal{I}} \mathbb{R}^{d_\ell \times r_\ell}. \tag{12}$$

For each optimization step t,

$$\left(\mathcal{M}_{\mathrm{SRM}}, g^t\right) \tag{13}$$

is a finite-dimensional Riemannian manifold.

Proof. By the assumed smooth structure, $\mathcal{M}_{\mathrm{SRM}}$ is a smooth finite-dimensional manifold. Its tangent space is naturally identified as

$$T_{W_B}\mathcal{M}_{\mathrm{SRM}} \simeq \prod_{\ell \in \mathcal{I}} \mathbb{R}^{d_\ell \times r_\ell}. \tag{14}$$

For fixed step t, the coefficients $g^t_{B,\ell,ij}$ are fixed during the metric-gradient construction. Hence $p \mapsto g^t_p$ is a constant tensor field in the standard matrix coordinates of $\mathcal{M}_{\mathrm{SRM}}$, so it is smooth.

The form g^t_p is bilinear by construction. It is symmetric because

$$g^t_p(u, v) = \sum_{\ell,i,j} g^t_{B,\ell,ij} u_{\ell,ij} v_{\ell,ij} = g^t_p(v, u). \tag{15}$$

Since

$$0 < g_{\min} \le g^t_{B,\ell,ij} \le g_{\max} < \infty, \tag{16}$$

every nonzero $v \in T_p\mathcal{M}_{\mathrm{SRM}}$ satisfies

$$g^t_p(v, v) = \sum_{\ell,i,j} g^t_{B,\ell,ij} v^2_{\ell,ij} \ge g_{\min} \sum_{\ell,i,j} v^2_{\ell,ij} > 0. \tag{17}$$

Thus g^t_p is a smoothly varying positive-definite inner product on $T_p\mathcal{M}_{\mathrm{SRM}}$. Therefore $(\mathcal{M}_{\mathrm{SRM}}, g^t)$ is a finite-dimensional Riemannian manifold. □

Remark 3.4. The manifold is the selected LoRA-B adapter-parameter manifold. SRM-LoRA equips this manifold with the step-wise data-dependent metric g^t. The matrix G_t is the coordinate representation of this metric tensor in the canonical LoRA-B coordinates.

3.4. Metric Gradient and Controlled Vector

Because G_t is the coordinate matrix of the metric tensor g^t, applying G_t^{-1} to the Euclidean gradient gives the Riemannian gradient in the canonical coordinates of $\mathcal{M}_{\mathrm{SRM}}$.

The Riemannian gradient is defined by

$$df_{W_B}(v) = g^t_{W_B}(\mathrm{grad}_{g^t} f, v) \qquad \forall v \in T_{W_B}\mathcal{M}_{\mathrm{SRM}}. \tag{18}$$

Theorem 3.5 (Inverse-metric gradient). If

$$D^t(W_B) = \nabla_{W_B} f, \tag{19}$$

then

$$\mathrm{grad}_{g^t} f = G_t(W_B)^{-1} D^t(W_B), \qquad -\mathrm{grad}_{g^t} f = -G_t(W_B)^{-1} D^t(W_B). \tag{20}$$

Proof. For any $v \in T_{W_B}\mathcal{M}_{\mathrm{SRM}}$,

$$df_{W_B}(v) = \mathrm{vec}(D^t(W_B))^\top \mathrm{vec}(v). \tag{21}$$

By the definition of the Riemannian gradient,

$$df_{W_B}(v) = g^t_{W_B}(\mathrm{grad}_{g^t} f, v) = \mathrm{vec}(\mathrm{grad}_{g^t} f)^\top G_t(W_B)\,\mathrm{vec}(v). \tag{22}$$

Since this identity holds for all v,

$$G_t(W_B)\,\mathrm{vec}(\mathrm{grad}_{g^t} f) = \mathrm{vec}(D^t(W_B)). \tag{23}$$

Therefore,

$$\mathrm{grad}_{g^t} f = G_t(W_B)^{-1} D^t(W_B). \tag{24}$$

The negative field

$$-\mathrm{grad}_{g^t} f = -G_t(W_B)^{-1} D^t(W_B) \tag{25}$$

is the metric-controlled descent direction. □

Lemma 3.6 (Diagonal implementation). For each selected module,

$$(G^t_{B,\ell})^{-1} D^t_\ell = (g^t_{B,\ell})^{-1} \odot D^t_\ell, \qquad \nabla^{\mathrm{opt},t}_{W_B} \leftarrow G_t(W_B)^{-1} D^t(W_B). \tag{26}$$

The forward LoRA computation is unchanged during training. If $g_{\min} \geq 1$, then

$$0 < (g^t_{B,\ell,ij})^{-1} \leq 1. \tag{27}$$

Thus, the metric suppresses high-cost movement.

3.5. Weak Sub-Riemannian Interpretation

We now give the weak sub-Riemannian interpretation used in SRM-LoRA. This interpretation is local and directional: it identifies the metric-shaped backward update as an admissible horizontal direction, but does not claim a global bracket-generating distribution.

Let

$$\widetilde{C}^t(W_B) := -\mathrm{grad}_{g^t} f = -G_t^{-1} D^t(W_B) = G_t^{-1} C^t(W_B), \tag{28}$$

where $C^t(W_B) = -D^t(W_B)$ is the raw contrastive descent direction. On points where $\widetilde{C}^t(W_B) \neq 0$, define the local horizontal line

$$H^t_{\mathrm{SRM}}(W_B) := \mathrm{span}\{\widetilde{C}^t(W_B)\} \subset T_{W_B}\mathcal{M}_{\mathrm{SRM}}, \qquad g^t_H := g^t|_{H^t_{\mathrm{SRM}}}. \tag{29}$$

By construction,

$$\widetilde{C}^t(W_B) \in H^t_{\mathrm{SRM}}(W_B). \tag{30}$$

Thus, the metric-shaped update is admissible under the weak local horizontal direction selected at step t.

The raw Euclidean contrastive direction $C^t(W_B)$ belongs to the same local horizontal line only when it is collinear with the controlled direction, i.e., when there exists $\alpha \in \mathbb{R}$ such that

$$C^t(W_B) = \alpha G_t^{-1} C^t(W_B). \tag{31}$$

For a non-scalar diagonal metric, this collinearity generally fails unless the nonzero coordinates of $C^t(W_B)$ lie in entries with the same metric coefficient. Therefore, SRM-LoRA changes the admissible update direction by anisotropically reweighting the raw contrastive tangent vector.

This weak sub-Riemannian interpretation is not used to establish global controllability. It formalizes the local optimization rule in which SRM-LoRA selects a metric-shaped horizontal direction in the LoRA-B tangent space and updates along that direction.

Table 1. Seed-wise results on DROP at step 150 ($N = 9535$). Net% is Net/9535×100. Ours Δ is the SRM-LoRA gain over each baseline in Net; percentages are over all 9,535 examples. Light-blue rows denote SRM-LoRA; † marks the block-best value. Main experiment(see Table 4) seed is 42.

Seed	Method	Imp.	Wors.	Net	Net%	Ours Δ	Halluc.
40	**SRM-LoRA**	**1714**	**1113**	**601**†	**6.30%**†	–	**35.49%**†
40	Contrastive LoRA	1715	1132	583	6.11%	+18 (+0.19%)	35.67%
40	Plain LoRA	1698	1124	574	6.02%	+27 (+0.28%)	36.18%
43	**SRM-LoRA**	**1723**	**1085**	**638**†	**6.69%**†	–	**35.21%**†
43	Contrastive LoRA	1675	1040	635	6.66%	+3 (+0.03%)	35.40%
43	Plain LoRA	1621	1079	542	5.68%	+96 (+1.01%)	36.03%
44	**SRM-LoRA**	**1734**	**1038**	**696**†	**7.30%**†	–	**34.74%**†
44	Contrastive LoRA	1678	1091	587	6.16%	+109 (+1.14%)	35.09%
44	Plain LoRA	1641	1089	552	5.79%	+144 (+1.51%)	36.21%

Table 2. Base-model ablation results on DROP ($N = 9535$). Improved and worsened denote the number of examples whose judged correctness improves or worsens after training. Ours Gain reports the SRM-LoRA gain over each baseline in Net, with the percentage computed over all 9,535 examples. Light-blue rows denote SRM-LoRA; † marks the block-best value.

Backbone	Step	Method	Imp.	Wors.	Net	Ours Gain	Halluc.
Meta-Llama-3.1-8B-Instruct	100	**SRM-LoRA**	**1422**	**831**	**591**†	–	**31.03%**†
		Contrastive LoRA	1279	885	394	+197 (+2.07%)	32.50%
		Plain LoRA	1420	918	502	+89 (+0.93%)	31.78%
	150	**SRM-LoRA**	**1482**	**953**	**529**	–	**30.47%**†
		Contrastive LoRA	1469	939	530†	-1 (-0.01%)	31.48%
		Plain LoRA	1201	1621	-420	+949 (+9.95%)	40.78%
HuggingFace7B	100	**SRM-LoRA**	**1565**	**1198**	**367**†	–	**43.24%**†
		Contrastive LoRA	1526	1180	346	+21 (+0.22%)	43.37%
		Plain LoRA	1561	1256	305	+62 (+0.65%)	43.82%
	150	**SRM-LoRA**	**1556**	**1241**	**315**	–	**43.70%**
		Contrastive LoRA	1552	1223	329†	-14 (-0.15%)	43.61%†
		Plain LoRA	1561	1336	225	+90 (+0.94%)	44.58%
Google Gemma-7B-it	100	**SRM-LoRA**	**1273**	**1268**	**5**†	–	**53.69%**
		Contrastive LoRA	1144	1514	-370	+375 (+3.93%)	57.30%
		Plain LoRA	1288	1290	-2	+7 (+0.07%)	53.43%†
	150	**SRM-LoRA**	**1450**	**1188**	**262**	–	**50.44%**
		Contrastive LoRA	1461	1146	315†	-53 (-0.56%)	50.12%†
		Plain LoRA	1323	1255	68	+194 (+2.03%)	52.68%

4. Experiments

4.1. Experimental Setup

We evaluate SRM-LoRA on reference-grounded hallucination correction. Each example contains a reference passage, a question, and a good answer. All methods are trained only on HaluEval-QA. No additional task-specific training data or separate task training is used. For contrastive methods, each good answer is paired with a hallucinated answer generated from the frozen base model and filtered by the training judge.

We compare six adaptation methods. Plain LoRA uses answer cross-entropy, Contrastive LoRA uses good-answer versus hallucinated-answer contrastive loss, ORPO uses odds-ratio preference optimization, DoRA uses weight-decomposed LoRA, and RPLoRA uses Riemannian preconditioned LoRA. SRM-LoRA uses the same contrastive supervision as Contrastive LoRA, but applies a learned inverse metric to selected adapter gradients.

Unless otherwise stated, the base model is Qwen2.5-7B-Instruct. The selected adapter layer is fixed to layer 27, and the selected adapter modules are the MLP projection modules. Evaluation is performed periodically during training.

4.2. Training Objective

We use the same training examples (Halueval-qa) for all methods. Plain LoRA receives only the grounded answer y^+. Contrastive LoRA, DoRA, RPLoRA, and SRM-LoRA use the same pair (y^+, y^-) with the same contrastive objective, while ORPO uses the pair as chosen/rejected responses in the TRL ORPO objective. Thus, the comparison uses the same training source and negative construction while varying the objective, adapter parameterization, optimizer geometry, or SRM update geometry.

Table 3. Judge-wise hallucination correction results on DROP, which contains 9,535 evaluation examples. We report results at step 150 for Meta-Llama and step 100 for Gemma.

Judge Model	Method	Imp.	Wors.	Net	Ours Gain	Post-training Halluc. Rate
Meta-Llama-3.1-8B-Instruct	**SRM-LoRA (Ours)**	**1194**	**709**	**485**†	–	**10.04%**†
	Contrastive LoRA	1174	737	437	+48 (+0.50%)	10.38%
	Plain LoRA	1199	825	374	+111 (+1.16%)	11.46%
Google Gemma-7B-it	**SRM-LoRA (Ours)**	**142**	**149**	**-7**†	–	**2.18%**†
	Contrastive LoRA	140	171	-31	+24 (+0.25%)	2.36%
	Plain LoRA	118	151	-33	+26 (+0.27%)	2.33%

Table 4. Evaluation-dataset-wise hallucination correction results at step 150. Train dataset is Halueval-Qa. Ours Gain (%) is computed over each evaluation dataset size: DROP ($N = 9535$), HotpotQA FullWiki ($N = 7405$), and HaluEval-Dialogue ($N = 10000$).

Evaluation Dataset	Method	Improved	Worsened	Net Improved	Ours Gain	Ours Gain (%	Post-training Halluc. Rate
DROP	**SRM-LoRA (Ours)**	**1682**	**1032**	**650**	–	–	**34.70%**
	Contrastive LoRA	1691	1050	641	+9	+0.09%	34.84%
	Plain LoRA	1611	1116	495	+155	+1.63%	36.16%
	ORPO	1665	1459	206	+444	+4.66%	39.99%
	DoRA	1708	1265	443	+207	+2.17%	36.94%
	RPLoRA	1471	1607	-136	+786	+8.24%	42.70%
HotpotQA FullWiki	**SRM-LoRA (Ours)**	**935**	**405**	**530**	–	–	**49.98%**
	Contrastive LoRA	931	437	494	+36	+0.49%	50.26%
	Plain LoRA	895	441	454	+76	+1.03%	50.71%
	ORPO	958	713	245	+285	+3.85%	54.07%
	DoRA	961	485	476	+54	+0.73%	50.26%
	RPLoRA	902	675	227	+303	+4.09%	54.84%
HaluEval-Dialogue	**SRM-LoRA (Ours)**	**1054**	**1086**	**-32**	–	–	**70.63%**
	Contrastive LoRA	1035	1096	-61	+29	+0.29%	71.53%
	Plain LoRA	1048	1071	-23	-9	-0.09%	71.11%
	ORPO	792	1712	-920	+888	+8.88%	79.99%
	DoRA	983	1233	-250	+218	+2.18%	72.85%
	RPLoRA	604	1953	-1349	+1317	+13.17%	84.56%

For SRM-LoRA, the contrastive gradient on the selected LoRA-B modules is transformed before the optimizer step:

$$\nabla_{W_B}^{\mathrm{opt},t} \leftarrow G_t(W_B)^{-1} D_B^t.$$

Since $G_t(W_B)$ is diagonal,

$$G_t(W_B)^{-1} D_B^t = (g_B^t)^{-1} \odot D_B^t.$$

The forward computation is unchanged. The metric affects training-time movement only, so inference uses the ordinary LoRA forward path.

4.3. Evaluation Protocol

See Section A.5 for the detailed evaluation protocol. We report pairwise transition metrics relative to the frozen base model: Improved, Worsened, Net, and final hallucination rate. This makes the correction–damage trade-off explicit. Judge dependence is detailed in Section A.9.

4.4. Evaluation Datasets

The reported evaluation uses DROP, HotpotQA FullWiki, and HaluEval-Dialogue. HaluEval-Dialogue belongs to the HaluEval family but differs from the HaluEval-QA training task. DROP and HotpotQA FullWiki are out-of-distribution evaluations. No evaluation-specific training data is added. We read each table through correction and damage: Improved counts examples changed from incorrect to correct, Worsened counts examples changed from correct to incorrect, and Net is their difference. This is the metric most aligned with SRM-LoRA's update rule, because the learned metric is designed to reshape the whole contrastive direction.

4.5. Seed Robustness on DROP

Table 1 reports seed-wise DROP results. Across seeds $40, 43, 44$, SRM-LoRA achieves the best Net improvement $(601, 638, 696)$ and the lowest post-training hallucination rate $(35.49\%, 35.21\%, 34.74\%)$. Since Contrastive LoRA

uses the same (y^+, y^-) supervision and the same contrastive objective $\mathcal{L}_{\text{con}}$, the repeated gap isolates the effect of the update geometry. Contrastive LoRA follows the Euclidean tangent direction D_B^t, whereas SRM-LoRA follows

$$\nabla_{W_B}^{\text{opt},t} = G_t(W_B)^{-1} D_B^t.$$

Thus, the gain comes from changing the local movement cost on the selected LoRA-B manifold, not from adding stronger supervision. Seed 44 gives the clearest instance of this effect. SRM-LoRA reaches the largest Net improvement 696, improving over Contrastive LoRA by $+109$ and over Plain LoRA by $+144$. This shows that the inverse-metric update can amplify useful contrastive directions and suppress sensitive directions in the same training regime. The result supports the intended role of $G_t(W_B)^{-1}$: it does not change the objective, but changes the tangent direction actually followed by the adapter.

4.6. Judge Robustness on DROP

Table 3 evaluates backbone transfer on DROP. The Meta-Llama-3.1-8B-Instruct result shows the correction–damage control effect most clearly. Plain LoRA attains a slightly larger Improved count than SRM-LoRA (1199 vs. 1194), yet it also causes substantially more Worsened cases (825 vs. 709). SRM-LoRA therefore obtains the best Net improvement 485 and the lowest hallucination rate 10.04%. This matches the proposed metric update: the method does not maximize correction count alone; it reshapes the tangent update so that high-sensitivity directions incur larger movement cost. The result therefore supports SRM-LoRA as a method for reducing hallucination without less damaging already-correct examples.

The Gemma-7B result is also informative. We use the same raw prompt format across all backbones, without model-specific chat templates. This conservative protocol can limit the absolute performance of chat-template-sensitive models such as Gemma. Even under this setting, SRM-LoRA remains the strongest relative method: it has the least negative Net score (-7) and the lowest hallucination rate 2.18%. Thus, the metric-shaped update remains beneficial even when the backbone is less aligned with the shared prompting format.

4.7. Evaluation Across Dataset Sources

Table 4 evaluates SRM-LoRA across dataset sources using seed 42. All methods are trained only on HaluEval-QA. No extra training split is selected for each benchmark, and no additional synthetic training data is constructed. HaluEval-Dialogue belongs to the same HaluEval family, yet it is not the training dataset. DROP and HotpotQA FullWiki are out-of-distribution reference-grounded QA evaluations.

This setting is important for hallucination mitigation. The goal is not to maximize a benchmark-specific score through dataset adaptation, but to obtain a robust correction rule that reduces hallucinated outputs with limited damage to already-correct behavior. For this reason, Improved, Worsened, Net, and final hallucination rate are considered together.

On DROP, DoRA and Contrastive LoRA have larger Improved counts than SRM-LoRA (1708 and 1691 vs. 1682), but they also produce more Worsened examples (1265 and 1050 vs. 1032). This gives the best Net improvement 650 and the lowest hallucination rate 34.70%. On HotpotQA FullWiki, DoRA and ORPO have larger Improved counts than SRM-LoRA (961 and 958 vs. 935), but SRM-LoRA has the lowest Worsened count 405, yielding the best Net 530 and the lowest hallucination rate 49.98%. These two OOD results show the main effect of the metric update. The inverse metric changes the local movement cost of the contrastive gradient, so the update improves correction without increasing damage.

HaluEval-Dialogue is the hardest case in this table. All methods have negative Net, and Plain LoRA is slightly less negative than SRM-LoRA. Even in this case, SRM-LoRA has the lowest hallucination rate 70.63%.

Finally, these main results use seed 42. The seed ablation in Table 1 shows that seed 44 gives a stronger SRM-LoRA result on DROP. Thus, the table should be read as a robust seed-42 evaluation, not as the largest observed performance of the method.

4.8. Judge

See Section A.9 for the detailed judge process.

5. Discussion

LLM adapter space is extremely high-dimensional. In vision or control problems, some coordinates may correspond to physical or semantic variables, such as spatial axes or joint angles. LoRA coordinates do not have this structure. A single LoRA weight is entangled with many upstream representations, so forcing specific coordinates to zero would be an unrealistic way to define admissible motion. SRM-LoRA therefore controls the whole LoRA-B gradient as a tangent vector:

$$D_B^t = \nabla_{W_B} \mathcal{L}_{\text{con}} \in T_{W_B} \mathcal{M}_{\text{SRM}}.$$

The diagonal metric is used as a practical coordinate realization of this vector control. SRM-LoRA equips the selected LoRA-B manifold with a positive movement-cost metric $G_t(W_B)$ and replaces the Euclidean update direction by the inverse-metric direction

$$\nabla_{W_B}^{\text{opt},t} = G_t(W_B)^{-1} D_B^t.$$

Thus, the raw Euclidean direction D_B^t is no longer the direction followed by the optimizer. Since $G_t(W_B)$ is diagonal, the implemented rule is

$$G_t(W_B)^{-1} D_B^t = (g_B^t)^{-1} \odot D_B^t.$$

The elementwise form is therefore a coordinate implementation of Riemannian vector control, not a claim that individual LoRA entries are semantic axes.

This differs from heuristic diagonal preconditioning. A generic preconditioner rescales gradients for numerical conditioning. SRM-LoRA constructs a manifold for the selected adapter gradients by assigning a data-dependent movement cost to the tangent field. High-sensitivity components receive larger cost:

$$g_{B,\ell,ij}^t \uparrow \quad \Longrightarrow \quad (g_{B,\ell,ij}^t)^{-1} \downarrow .$$

The update remains soft because no selected component is deleted. The original Euclidean direction is suppressed and replaced by the metric-shaped direction on $\mathcal{M}_{\text{SRM}}$. This gives a simple way to introduce a Riemannian training geometry: keep the forward LoRA map unchanged, place the metric on the backward gradient, and let the adapter follow the resulting manifold-induced update direction.

6. Conclusion

We presented SRM-LoRA, a sub-Riemannian-inspired LoRA framework for hallucination mitigation. SRM-LoRA uses positive–negative activation contrasts to identify grounded-dominant and hallucination-prone LoRA rank directions. The method interprets it as a local metric over adapter coordinates and applies the inverse metric through backward gradient scaling.

This view separates the supervision signal from the update geometry: positive–negative pairs define directional evidence, while the local metric determines how gradients move through LoRA space. SRM-LoRA therefore reframes hallucination mitigation from objective-level optimization to geometry-aware adapter adaptation.

References

Bellaard, G., Bon, D. L. J., Pai, G., Smets, B. M. N., and Duits, R. Analysis of (sub-)riemannian pde-g-cnns. *Journal of Mathematical Imaging and Vision*, 65:819–843, 2023. doi: 10.1007/s10851-023-01147-w. URL https://doi.org/10.1007/s10851-023-01147-w.

Bogachev, V., Aletov, V., Molozhavenko, A., Bobkov, D., Soboleva, V., Alanov, A., and Rakhuba, M. Lora meets riemannion: Muon optimizer for parametrization-independent low-rank adapters, 2025.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. Language models are few-shot learners. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 1877–1901. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

Chowdhury, S. R., Kini, A., and Natarajan, N. Provably robust dpo: Aligning language models with noisy feedback, 2024. URL https://arxiv.org/abs/2403.00409.

Dettmers, T., Pagnoni, A., Holtzman, A., and Zettlemoyer, L. Qlora: Efficient finetuning of quantized llms. *arXiv preprint arXiv:2305.14314*, 2023.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4171–4186. Association for Computational Linguistics, 2019.

Ding, N., Lv, X., Wang, Q., Chen, Y., Zhou, B., Liu, Z., and Sun, M. Sparse low-rank adaptation of pre-trained language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 4133–4145, December 2023.

Gekhman, Z., Yona, G., Aharoni, R., Eyal, M., Feder, A., Reichart, R., and Herzig, J. Does fine-tuning LLMs on new knowledge encourage hallucinations? In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 7765–7784, Miami, Florida, USA, 2024. Association for Computational Linguistics.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. Lora: Low-rank adaptation of large language models, 2021. URL https://arxiv.org/abs/2106.09685.

Jansson, E. and Modin, K. Sub-riemannian landmark matching and its interpretation as residual neural networks. *Journal of Computational Dynamics*, 12(3):467–490, 2025. ISSN 2158-2505. doi: 10.3934/jcd.2025004.

Kalai, A. T., Nachum, O., Vempala, S. S., and Zhang, E. Why language models hallucinate, 2025. URL https://arxiv.org/abs/2509.04664.

Koo, J. Sinogram upsampling via sub-riemannian diffusion with adaptive weighting. *Electronics*, 12(21):4503, 2023. doi: 10.3390/electronics12214503. URL https://doi.org/10.3390/electronics12214503.

Li, Z., Sajadmanesh, S., Li, J., and Lyu, L. Stella: Subspace learning in low-rank adaptation using stiefel manifold. In *Advances in Neural Information Processing Systems*, volume 38. Curran Associates, Inc., 2025.

Liu, S.-Y., Wang, C.-Y., Yin, H., Molchanov, P., Wang, Y.-C. F., Cheng, K.-T., and Chen, M.-H. Dora: Weight-decomposed low-rank adaptation. *arXiv preprint arXiv:2402.09353*, 2024.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., Schulman, J., Hilton, J., Kelton, F., Miller, L., Simens, M., Askell, A., Welinder, P., Christiano, P. F., Leike, J., and Lowe, R. Training language models to follow instructions with human feedback. In Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., and Oh, A. (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 27730–27744. Curran Associates, Inc., 2022.

Park, J., Kang, M., Lee, S., Lee, H., Kim, S., and Lee, J. Riemannian optimization for LoRA on the stiefel manifold. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 20971–20985, 2025.

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 2227–2237. Association for Computational Linguistics, June 2018.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. Language models are unsupervised multitask learners. 2019.

Sharma, M., Tong, M., Korbak, T., Duvenaud, D., Askell, A., Bowman, S. R., Cheng, N., Durmus, E., Hatfield-Dodds, Z., Johnston, S. R., Kravec, S., Maxwell, T., McCandlish, S., Ndousse, K., Rausch, O., Schiefer, N., Yan, D., Zhang, M., and Perez, E. Towards understanding sycophancy in language models, 2025. URL https://arxiv.org/abs/2310.13548.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E., Le, Q., and Zhou, D. Chain-of-thought prompting elicits reasoning in large language models, 2023. URL https://arxiv.org/abs/2201.11903.

Whitehouse, C., Huot, F., Bastings, J., Dehghani, M., Lin, C.-C., and Lapata, M. Low-rank adaptation for multilingual summarization: An empirical study. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pp. 1202–1228, June 2024.

Zhang, F. and Pilanci, M. Riemannian preconditioned lora for fine-tuning foundation models, 2024.

A. Additional Method Details

A.1. Metric Loss

For every selected LoRA-B matrix W_B, let Q_B be the unconstrained metric parameter. Define

$$S_B = \tanh(Q_B/T), \qquad \widehat{S}_B = \begin{cases} S_B/\|S_B\|_2, & \|S_B\|_2 > 0, \\ 0, & \|S_B\|_2 = 0. \end{cases} \tag{A.1}$$

Let $\widehat{X}_B^t$ be the observed sensitivity target in Appendix A.3. The fitting loss is

$$L_{\text{fit}} = \frac{1}{dr}\|\widehat{S}_B - \widehat{X}_B^t\|_2^2. \tag{A.2}$$

The unbounded movement cost induced by the learned signal is

$$\tilde{g}_B = 1 + \gamma|\widehat{S}_B|. \tag{A.3}$$

The identity regularizer is

$$L_{\text{id}} = \frac{1}{dr}\sum_{i,j}(\tilde{g}_{B,ij} - 1)^2. \tag{A.4}$$

The bound penalty is

$$L_{\text{bd}} = \frac{1}{dr}\sum_{i,j}\left([g_{\min} - \tilde{g}_{B,ij}]_+^2 + [\tilde{g}_{B,ij} - g_{\max}]_+^2\right). \tag{A.5}$$

The condition-number penalty is

$$\kappa_B = \frac{\max_{i,j}\tilde{g}_{B,ij}}{\min_{i,j}\tilde{g}_{B,ij}}, \qquad L_{\text{cond}} = [\kappa_B - \kappa_{\max}]_+^2. \tag{A.6}$$

The full metric objective is

$$L_{\text{metric}} = \lambda_{\text{fit}}L_{\text{fit}} + \lambda_{\text{id}}L_{\text{id}} + \lambda_{\text{bd}}L_{\text{bd}} + \lambda_{\text{cond}}L_{\text{cond}} + \lambda_{\text{la}}L_{\text{la}}. \tag{A.7}$$

The gradient of (A.7) is assigned to Q_B before the optimizer update.

Term	Meaning in the metric objective
$\widehat{S}_B$	Normalized learned metric signal from Q_B; this is the signal that will later shape the inverse metric.
$\widehat{X}_B^t$	Normalized finite-difference sensitivity target computed from $\Delta D_B^t/\Delta W_B^t$.
L_{fit}	Fits the learned signal $\widehat{S}_B$ to the observed sensitivity target $\widehat{X}_B^t$.
$\tilde{g}_B$	Unclipped candidate movement cost $1 + \gamma\|\widehat{S}_B\|$. Larger values mean stronger suppression after inversion.
L_{id}	Identity regularizer; keeps $\tilde{g}_B$ close to 1 unless the sensitivity signal justifies changing the geometry.
L_{bd}	Bound regularizer; penalizes movement costs outside $[g_{\min}, g_{\max}]$.
L_{cond}	Condition-number regularizer; prevents the largest and smallest movement costs from becoming too imbalanced.
L_{la}	Lookahead term; compares a metric-scaled probe update against a plain contrastive probe update.
L_{metric}	Weighted objective used to update Q_B; it does not replace the contrastive LoRA objective.
$\lambda_{\cdot}$	Coefficients controlling the relative strength of the corresponding metric-loss terms.

A.2. Lookahead Objective

Let D_B^t be the selected contrastive gradient and let

$$\bar{g}_B(Q_B) = \text{clip}_{[g_{\min}, g_{\max}]}\left(1 + \gamma|\widehat{S}_B|\right). \tag{A.8}$$

The ordinary probe update and metric probe update are

$$W'_{B,\text{plain}} = W_B - \eta_{\text{la}}D_B^t, \tag{A.9}$$

$$W'_{B,\text{metric}} = W_B - \eta_{\text{la}}\left(\bar{g}_B(Q_B)^{-1} \odot D_B^t\right). \tag{A.10}$$

On the probe batch,

$$J(W'_B) = L_{\mathrm{CE}}(y^+; W'_B) - L_{\mathrm{CE}}(y^-; W'_B). \tag{A.11}$$

The lookahead term is

$$L_{\mathrm{la}} = \mathrm{softplus}\left(J(W'_{B,\mathrm{metric}}) - J(W'_{B,\mathrm{plain}}) + m_{\mathrm{la}}\right). \tag{A.12}$$

Only the metric probe objective is differentiated through the metric parameter; the plain probe objective provides the local comparison target.

For memory efficiency, L_{la} is computed on a probe subset $\mathcal{B}_t^{\mathrm{la}}$ with at most two samples, while D_B^t is computed from the full accumulated contrastive gradient.

A.3. Sensitivity

For each selected LoRA-B matrix, store the previous local state

$$Z_B^{t-1} = (W_B^{t-1}, D_B^{t-1}). \tag{A.13}$$

At the next pre-optimizer state,

$$\Delta W_B^t = W_B^t - W_B^{t-1}, \qquad \Delta D_B^t = D_B^t - D_B^{t-1}. \tag{A.14}$$

Define

$$s_{ij}^t = \mathrm{sgn}(\Delta W_{B,ij}^t)\max(|\Delta W_{B,ij}^t|, \epsilon_{\mathrm{fd}}). \tag{A.15}$$

The raw sensitivity is

$$X_{B,ij}^t = \begin{cases} \Delta D_{B,ij}^t / s_{ij}^t, & |\Delta W_{B,ij}^t| \geq \epsilon_{\mathrm{fd}}, \\ 0, & |\Delta W_{B,ij}^t| < \epsilon_{\mathrm{fd}}. \end{cases} \tag{A.16}$$

The bounded and normalized signal is

$$\bar{X}_B^t = c_x \tanh(X_B^t / c_x), \tag{A.17}$$

$$\widehat{X}_B^t = \begin{cases} \bar{X}_B^t / \|\bar{X}_B^t\|_2, & \|\bar{X}_B^t\|_2 > 0, \\ 0, & \|\bar{X}_B^t\|_2 = 0. \end{cases} \tag{A.18}$$

For a small numerical threshold δ, the update signal is

$$H_B^t = \begin{cases} \widehat{X}_B^t, & \|S_B\|_2 \leq \delta, \\ \widehat{S}_B, & \|S_B\|_2 > \delta, \end{cases} \tag{A.19}$$

Where fixed numerical threshold $\delta = 10^{-12}$. And the applied metric is

$$g_B^t = \mathrm{clip}_{[g_{\min}, g_{\max}]}\left(1 + \gamma |H_B^t|\right). \tag{A.20}$$

A.4. Visualization

The recorded visualization is the inverse metric

$$(g_B^t)^{-1}. \tag{A.21}$$

Large values of $(g_B^t)^{-1}$ correspond to more admissible update components; small values correspond to stronger suppression by the local movement cost. The visualization is a training-time diagnostic of the inverse metric and is not used during inference.

A.5. Evaluation Protocol

For each evaluation example $z_i = (r_i, q_i, y_i^+)$, the frozen base model and the adapted model generate

$$\hat{y}_i^0 = F_0(r_i, q_i), \qquad \hat{y}_i^1 = F_1(r_i, q_i) \tag{A.22}$$

under the same reference-grounded prompt and decoding configuration.

Let $\nu(\cdot)$ lowercase text, replace non-alphanumeric characters with spaces, and collapse repeated whitespace. The exact containment check is

$$E(\hat{y}, y^+) = \mathbf{1}\left[\nu(y^+) \subset \nu(\hat{y})\right]. \tag{A.23}$$

If $E(\hat{y}, y^+) = 1$, the prediction is marked correct. Otherwise, the answer judge returns

$$J_{\text{eval}}(r, q, y^+, \hat{y}) \in \{0, 1\}. \tag{A.24}$$

The final correctness label is

$$C(\hat{y}; r, q, y^+) = \begin{cases} 1, & E(\hat{y}, y^+) = 1, \\ J_{\text{eval}}(r, q, y^+, \hat{y}), & E(\hat{y}, y^+) = 0. \end{cases} \tag{A.25}$$

Define

$$b_i = C(\hat{y}_i^0; r_i, q_i, y_i^+), \qquad a_i = C(\hat{y}_i^1; r_i, q_i, y_i^+). \tag{A.26}$$

Then

$$\text{Improved} = \sum_i \mathbf{1}[b_i = 0 \wedge a_i = 1], \tag{A.27}$$

$$\text{Worsened} = \sum_i \mathbf{1}[b_i = 1 \wedge a_i = 0], \tag{A.28}$$

$$\text{Net} = \text{Improved} - \text{Worsened}, \tag{A.29}$$

and the reported post-training hallucination rate is

$$\text{HallucRate} = \frac{1}{N} \sum_i \mathbf{1}[a_i = 0]. \tag{A.30}$$

A.6. SRM-LoRA Algorithm

Algorithm A.1: SRM-LoRA training step.

Input. Training example $z = (r, q, y^+)$, frozen base model F_0, selected LoRA-B weights W_B, learnable metric parameters Q_B, and previous pre-optimizer state Z_B^{t-1}.

0. Negative construction.

$$\tilde{y} \leftarrow F_0(r, q), \qquad y^- = \tilde{y} \Longleftrightarrow E(\tilde{y}, y^+) = 0 \wedge J_{\text{train}}(r, q, y^+, \tilde{y}) = 1.$$

$$y^- \text{ undefined} \quad \Rightarrow \quad \text{skip this example.}$$

1. Contrastive backward signal.

$$D_B^t = \nabla_{W_B} L_{\text{CE}}(y^+) - \nabla_{W_B} L_{\text{CE}}(y^-).$$

2. Boundary initialization.

$$Z_B^{t-1} = \varnothing \Rightarrow \begin{cases} \nabla_{W_B}^{\text{opt},t} \leftarrow D_B^t, \\ Z_B^t \leftarrow (W_B^t, D_B^t). \end{cases}$$

In this case, no finite-difference sensitivity is available yet.

3. Metric-controlled backward update. If $Z_B^{t-1} = (W_B^{t-1}, D_B^{t-1})$, compute

$$\Delta W_B^t = W_B^t - W_B^{t-1}, \qquad \Delta D_B^t = D_B^t - D_B^{t-1}.$$

Then

$$\widehat{X}_B^t \leftarrow \text{(A.14)–(A.18)}, \qquad H_B^t \leftarrow \text{(A.19)},$$

$$g_B^t = \text{clip}_{[g_{\min}, g_{\max}]} \left(1 + \gamma |H_B^t|\right).$$

The optimizer-facing LoRA-B gradient is replaced by

$$\nabla_{W_B}^{\text{opt},t} \leftarrow G_t(W_B)^{-1} D_B^t = (g_B^t)^{-1} \odot D_B^t.$$

4. Metric-gradient assignment and state refresh. The metric parameter is updated through its own objective:

$$\nabla_{Q_B}^t \leftarrow \frac{\partial L_{\text{metric}}}{\partial Q_B}.$$

The local state used by the next finite-difference step is refreshed as

$$Z_B^t \leftarrow (W_B^t, D_B^t).$$

Finally, the optimizer updates both parameter groups:

$$(W_B^{t+1}, Q_B^{t+1}) \leftarrow \text{OptStep}\left(W_B^t, Q_B^t; \nabla_{W_B}^{\text{opt},t}, \nabla_{Q_B}^t\right).$$

Output. Updated LoRA weights W_B^{t+1} and metric parameters Q_B^{t+1}.

A.7. Implementation Scope and Cost

The reported setting applies SRM control only to the selected LoRA-B gradient slots:

$$W_B \in \mathcal{I}_{\text{SRM}} \Rightarrow \nabla_{W_B}^{\text{opt}} \leftarrow G(W_B)^{-1} D_B = g_B^{-1} \odot D_B, \qquad W \notin \mathcal{I}_{\text{SRM}} \Rightarrow \nabla_W^{\text{opt}} \leftarrow \nabla_W L. \tag{C.1}$$

The metric-specific hyperparameters are

$$\{g_{\min}, g_{\max}, \gamma, T, \epsilon_{\text{fd}}, c_x, \delta, \eta_{\text{la}}, m_{\text{la}}, \lambda_{\text{fit}}, \lambda_{\text{id}}, \lambda_{\text{bd}}, \lambda_{\text{cond}}, \lambda_{\text{la}}\}. \tag{C.2}$$

The additional training-time components are

$$\text{Cost}_{\text{SRM}} = \text{Cost}_{\text{LoRA}} + \text{backward}(y^-) + \text{FD}(\Delta D / \Delta W) + \nabla_{Q_B} L_{\text{metric}}. \tag{C.3}$$

At inference, the metric variables are not used in the forward pass:

$$F_{\text{SRM}}^{\text{infer}}(x) = F_{\text{LoRA}}^{\text{infer}}(x), \qquad Q_B, G(W_B)^{-1} \notin \text{forward pass}. \tag{C.4}$$

Symbol or cost	Meaning
$\mathcal{I}_{\text{SRM}}$	Selected LoRA-B modules where SRM metric control is applied.
D_B	Unmodified contrastive gradient vector stored before metric scaling.
$G(W_B)^{-1} D_B = g_B^{-1} \odot D_B$	Optimizer-facing inverse-metric gradient for selected LoRA-B modules.
$W \notin \mathcal{I}_{\text{SRM}}$	Parameters outside the selected set; they keep the ordinary baseline gradient.
$g_{\min}, g_{\max}$	Lower and upper bounds on the positive movement cost.
γ	Strength converting sensitivity magnitude into movement cost.
T	Temperature in $S_B = \tanh(Q_B/T)$.
$\epsilon_{\text{fd}}, c_x, \delta$	Numerical controls for finite-difference stability, soft clipping, and learned-signal fallback.
$\eta_{\text{la}}, m_{\text{la}}$	Lookahead probe step size and margin.
$\lambda_{\text{fit}}, \lambda_{\text{id}}, \lambda_{\text{bd}}, \lambda_{\text{cond}}, \lambda_{\text{la}}$	Weights of the metric fitting, identity, bound, condition, and lookahead terms.
$\text{backward}(y^-)$	Extra backward pass for the hallucinated answer used to form D_B.
$\text{FD}(\Delta D / \Delta W)$	Finite-difference sensitivity computation between consecutive selected states.
$\nabla_{Q_B} L_{\text{metric}}$	Additional gradient used to update metric parameters Q_B.
$Q_B, G(W_B)^{-1} \notin$ forward pass	Metric variables affect training gradients only; inference uses the ordinary LoRA forward path.

A.8. Evaluation Scope

The reported comparison is scoped to LoRA-based adaptation methods:

$$\mathcal{B}_{\text{reported}} = \{\text{Plain LoRA},\ \text{Contrastive LoRA},\ \text{ORPO},\ \text{DoRA},\ \text{RPLoRA},\ \text{SRM-LoRA}\}. \tag{C.5}$$

same training source and negative construction with different objective, adapter parameterization, optimizer geometry, or SRM upda (C.6)

These tables do not claim comparison against standalone hallucination-mitigation systems.

A.9. Judge Details

The training and evaluation judges have different roles. Both use the same normalization and exact-containment filter:

$$\nu(s) := \text{collapse}\left(\text{space}\left(\text{lower}(s)\right)\right),$$

where non-alphanumeric characters are mapped to spaces before repeated whitespace is collapsed. The deterministic containment filter is

$$E(\hat{y}, y^{+}) := \mathbf{1}\left[\nu(y^{+}) \subset \nu(\hat{y})\right].$$

The implementation applies this filter before calling the LLM judge.

Generation prompt. The frozen base model and the adapted model use the same raw prompt:

`Answer the question using the reference.`

The full generation prompt is:

Generation prompt

```
Answer the question using the reference.
Reference:
{reference}
Question:
{question}
Answer:
```

No model-specific chat template is used in the reported setting. Inputs are tokenized with left padding, truncation, and the configured maximum prompt length. Generation uses

$$\texttt{max_new_tokens} = 16, \qquad \texttt{pad_token_id} = \texttt{tokenizer.pad_token_id}.$$

No explicit sampling argument is set. The continuation after the prompt is decoded with `skip_special_tokens=True`.

Training hallucination judge. For a training example (r, q, y^{+}), the frozen base model first generates

$$\tilde{y} \leftarrow F_0(r, q).$$

If $E(\tilde{y}, y^{+}) = 1$, the prediction is not used as a hallucinated negative:

$$E(\tilde{y}, y^{+}) = 1 \quad \Rightarrow \quad \tilde{y} \notin \mathcal{Y}^{-}_{\text{train}}.$$

If $E(\tilde{y}, y^{+}) = 0$, the training hallucination judge is called with the following prompt.

Training hallucination judge prompt

```
Decide whether the prediction is hallucinated.  Answer only yes or no.
Mark no when the gold answer is present in the prediction, even if the prediction also includes extra
explanation, punctuation, or surrounding text.  Mark yes only when the prediction contradicts the reference,
is unsupported by the reference, or fails to include the gold answer or a semantically equivalent answer.
Reference:
{reference}
Question:
{question}
Gold answer:
{gold answer}
Prediction:
{prediction}
Hallucinated?
```

The output is parsed as hallucinated if the lowercased text starts with

`yes`, `true`, `hallucinated`.

It is parsed as non-hallucinated if it starts with

`no`, `false`, `not hallucinated`, `supported`.

For other outputs, the prediction is parsed as hallucinated only when the text contains `yes` and does not contain `no`. Thus,

$$J_{\text{train}}(r, q, y^+, \tilde{y}) \in \{0, 1\},$$

and the negative answer is kept only when

$$y^- = \tilde{y} \iff E(\tilde{y}, y^+) = 0 \ \wedge \ J_{\text{train}}(r, q, y^+, \tilde{y}) = 1.$$

The training judge constructs hallucinated contrastive negatives. It does not score evaluation predictions.

Evaluation answer judge. For each evaluation example, the frozen base model and the adapted model generate

$$\hat{y}_i^0 = F_0(r_i, q_i), \qquad \hat{y}_i^1 = F_1(r_i, q_i).$$

If $E(\hat{y}, y^+) = 1$, the prediction is marked correct without an LLM judge call. If $E(\hat{y}, y^+) = 0$, the evaluation answer judge is called with the following prompt.

Evaluation answer judge prompt

```
Decide whether the prediction should be accepted as a correct answer.  Use semantic equivalence, not exact
string matching.  Answer only yes or no.
Accept the prediction when it contains the gold answer or a semantically equivalent answer anywhere in the
text, even if it also includes extra explanation, punctuation, or surrounding words.  Reject only when the
answer is absent, contradicted, or unsupported.
Reference:
{reference}
Question:
{question}
Gold answer:
{gold answer}
Prediction:
{prediction}
Correct answer?
```

The output is parsed as correct if the lowercased text starts with

`yes`, `true`, `correct`, `acceptable`.

It is parsed as incorrect if it starts with

`no`, `false`, `incorrect`, `wrong`, `not correct`.

For other outputs, the prediction is parsed as correct only when the text contains `yes` and does not contain `no`. The final correctness label is

$$C(\hat{y}; r, q, y^+) = \begin{cases} 1, & E(\hat{y}, y^+) = 1, \\ J_{\text{eval}}(r, q, y^+, \hat{y}), & E(\hat{y}, y^+) = 0. \end{cases}$$

The same E and J_{eval} rule is applied to all methods. The resulting labels define Improved, Worsened, Net, and post-training hallucination rate.