Probe-Free Low-Rank Activation Intervention

Anonymous ACL submission

Abstract

Language models (LMs) could produce texts that appear accurate and coherent but contain untruthful or toxic content. Inference time interventions that edit the hidden activations have shown promising results in steering the LMs towards desirable generations. Existing activation intervention methods often comprise an activation probe to detect undesirable generation, triggering the activation modification to steer subsequent generation. This paper proposes a probe-free intervention method **FLORAIN** for all the heads in one specific activation layer. It eliminates the need to train classifiers for probing purposes. The intervention function is parameterized by a sample-wise nonlinear low-rank mapping, which is trained by minimizing the distance between the modified activations and their projection onto the manifold of the desirable content. Under specific constructions of the manifold and the projection distance, we show that the intervention strategy can be computed efficiently by solving smooth optimization problems. The empirical results, benchmarked on multiple base models, demonstrate that FLORAIN consistently outperforms several baseline methods in enhancing model truthfulness and quality across both generation and multiple-choice tasks.

1 Introduction

Transformer-based language models (LMs) have revolutionized generative modeling for natural language processing (NLP) (Radford et al., 2019; Brown et al., 2020; Anthropic, 2024; Jiang et al., 2023). The training pipeline of the LM in various NLP areas can be roughly divided into the following steps: (i) provide the model with prompts that include demonstrations of the task, (ii) learn from the examples, and (iii) make predictions without requiring additional training. In the generating process, there is evidence (Ji et al., 2023; Rawte et al., 2023; Xu et al., 2024) showing that the outputs of LM can give undesirable content, such as inaccurate answers, toxic answers, or answers with no linguistic meaning.

One of the prevalent solutions to the issues above is *activation intervention* (Subramani et al., 2022; Hernandez et al., 2023; Li et al., 2024b). Compared with the prior arts on model editing (Zhang et al., 2023) and supervised fine-tuning (Li et al., 2024a), activation intervention alters the model activations during inference. Therefore, it does not need to alter model weights using a subset of text samples, requiring fewer computational resources. However, there are new challenges that need to be addressed in the activation intervention task: (*Q1*) How to determine the intervention direction with theoretical underpinnings and appropriate modeling? (*Q2*) How can inference activation intervention be simplified with lower matrix-matrix multiplication complexity?

For question (*Q1*), Burns et al. (2022) revealed latent knowledge inside the internal activations of a language model and found a direction in activation space that satisfies logical consistency properties. Following this observation and intuition, there is a line of work focusing on finding good intervention systematically by two-stage methods (Yin et al., 2024; Li et al., 2024b; Pham and Nguyen, 2024). In the first stage, they train a classifier (the probe) on the activations of a network to identify a subset of attention heads that are most important for learning a specific task. In the second step, they propose intervention policies based on the probe and geometric transformation, e.g., linear operations (Li et al., 2024b), addictive offset bias vector (Yin et al., 2024) and householder transformation (Pham and Nguyen, 2024).

The preliminary success of these activation intervention methods motivates us to improve the desirable generation of LMs. Specifically, we want to design an inference activation intervention method that does not require the 'detect and rectify' proce-

dure. Our core idea is to depict the region of the desirable answer and consider the related low-rank transformation mapping for intervention. Building upon these, the parameters in the intervention mapping are the solutions to a nonlinear low-rank optimization problem. Compared with the prior region modeling works of LM activation space (Mamou et al., 2020) (Janiak et al., 2024), we give the concrete formulation of the region and allow for analytical projection by selecting a suitable distance measure. The associated low-rank optimization minimizes the distance between the intervention vector and its projection on the region.

Contributions. We contribute Probe-**F**ree **Lo**w-**R**ank **A**ctivation **In**tervention (FLORAIN), a novel intervention method characterized by

- An ellipsoidal model to capture the region of desirable output. In the training phase, we extract the activation heads of desirable answers to different questions and build the ellipsoid region models. The ellipsoid model can be estimated by the first-order and second-order statistical information of the activations extracted from the training data. To combat the ill-conditioning of the estimate due to the low sample size, we propose an extrapolation strategy to pull the ellipsoidal region away from the region of *un*desirable outputs.

- A low-rank intervention mapping. We propose to minimize the gap between the post-intervention activation and its projection onto the desirable ellipsoid region. The low-rank intervention mapping is the product of a one-layer multilayer perception with another low-rank matrix variable. We show that the objective function admits an analytical form under the Mahalanobis distance projection. In addition, we show that the training problem is smooth and, thereby, can be calculated efficiently using a lightweight gradient descent method or its preconditioned version.

From the computational aspect, our intervention method FLORAIN edits the activation vectors on *one* layer of the Transformer network. This contrasts to ITI (Li et al., 2024b) that intervene in different heads spread out in *multiple* layers of the network. Intervening in one layer, as FLORAIN does, has two significant advantages, which provide conditions for parallelization to save the inference intervention time. Furthermore, since the intervention vectors are concentrated in one layer, the strategy reduces the distribution shifts by avoiding misleading intervention in the subsequent layers.

1.1 Related Works

Controllable generation. Model editing (Wang et al., 2023; Zhang et al., 2024) edit the model parameters to control the output, and it is a powerful way to implement controllable generation. Another important category in controllable generation is fine-tuning, which includes Supervised Fine-Tuning (SFT, (Peng et al., 2023; Gunel et al., 2020)) and Reinforcement Learning from Human Feedback (RLHF, (Ouyang et al., 2022; Griffith et al., 2013)). These methods typically require alternating model weights of some text samples as well as substantial resources for computation.

Activation intervention. at inference time is an emerging technique for controllable generation (Li et al., 2024b; Singh et al., 2024; Yin et al., 2024). Unlike model editing or fine-tuning techniques, the inference-time intervention does not require altering the model parameters, leading to cheaper computational costs. Li et al. (2024b) proposed a headwise intervention method for eliciting truthful generated answers of a language model. Singh et al. (2024) considers the optimal transport plan between two empirical distributions to do the intervention. LoFit (Yin et al., 2024) also identifies a specific subset of attention heads crucial for learning a particular task but then performs fine-tuning on the intervention vectors at those chosen heads. Another recent work (Pham and Nguyen, 2024) considers doing intervention activation using modified householder transformation based on the linear probe framework.

Region Modeling in LM. Various works are trying to reveal how the semantic features influence the 'region' of embedding vectors in the transformer-based LM. The work (Mamou et al., 2020) utilizes mean-field theoretic manifold analysis to connect the geometry of feature representations with the linear separability of classes. The work (Janiak et al., 2024) identifies stable regions in the residual stream of Transformers, where the model's output remains insensitive to small activation changes but exhibits high sensitivity at region boundaries. However, most of the prior arts emphasize specific findings in region modeling and ignore the power of enhancing activation via the transport between two regions.

Low-Rank Optimization. is a widely studied

area in machine learning, statistics, and signal processing (Olikier et al., 2022; Zhanxuan et al., 2020; Cory-Wright et al., 2021). The idea in low-rank optimization lies in two folds: the nature of the low-rank property of the ground truth and the goal for computational efficiency with less complexity per iterative step (McRae and Boumal, 2024). One popular approach to solving low-rank optimization is tackling the non-convex optimization problem directly by the first-order method. Recently, several works have established the convergence rate to ground truth $X^* = U^*V^{*\top}$ for low-rank matrix optimization problem with the form $f(UV^\top)$ under mild assumptions in statistic estimation problems (Chi et al., 2019; Tong et al., 2021). We also notice that the low-rank idea has been used for LM model-editing, fine-tuning, and model compression (Hu et al., 2021; Hsu et al., 2022).

2 Nonlinear Low-Rank Intervention Mapping

This section defines the nonlinear low-rank intervention maps. In our probe-free intervention framework, this mapping is the computationally efficient intervention function in the inference time.

For a clear exposition, we consider a specific layer ℓ of the transformer network, and the index of the layer ℓ will be omitted in the following statement. This layer contains H attention heads; each head is of dimension d, and thus, the output of this layer is a $D = d \times H$ dimensional activation vector. For an input x_i, the activation at layer ℓ is denoted $a_i = [a_{i1}; a_{i2}; \cdots; a_{iH}] \in \mathbb{R}^D$. Our inference-time intervention will modify the activation vector head-wise for a_{ih} for all $h \in [H]$.

As for the construction of low-rank mapping, there are two points that need to be considered: (i) the intervention strategy should be generalized to different samples, and (ii) the intervention strategy should not change a desirable answer too much. Motivated by the requirements, our efficient nonlinear low-rank intervention $f(a)$ is constructed as

$$f : a \mapsto (I + L(a)R^\top)a + c, \tag{1a}$$

where $L(a) \in \mathbb{R}^{D\times k}$ is a low-rank matrix that depends on the input a, $R \in \mathbb{R}^{D\times k}$ is another constant low-rank matrix, $c \in \mathbb{R}^{D\times 1}$ is a vector. The dependence of $L(a)$ on the input a is necessary: if a is the activation of a desirable input, then we prefer $f(a)$ to stay close to a, and this can be achieved if $L(a)$ and vector c have small input. On the contrary, if a is the activation of an undesirable input, then we need the term $L(a)R^\top a + c$ to do the effective intervention. We remark that there is another parametrization method with the L being invariant and the $R(a)$ dependent on a. Due to its equivalence to (1a), we do not include it in our discussion. The parametrization of $L(a)$ is described as follow:

$$L_i(a) = \phi(W_i \circ a + b_i) \quad \forall i \in [k], \tag{1b}$$

where $L_i(a)$ and W_i refer to the i-th column of $L(a)$ and W. The notation $\circ$ indicates the Hadamard product of two vectors, $W \in \mathbb{R}^{D\times k}$ is the weight matrix, $b \in \mathbb{R}^{D\times k}$ is the bias term, and ϕ is an activation function applied componentwise on the input. One candidate choice for this nonlinear function is $\phi(\cdot) = \tanh(\cdot)$. This enhances the intervention performance since the function $\tanh(\cdot)$ can output both negative and positive real numbers. In addition, the output of $\tanh(\cdot)$ is bounded between $(-1, 1)$, automatically resolving the scaling problems of the bilinear terms $L(a)R^\top$ in formulation (1a). In fact, if the components of $L(a)$ are not bounded , then one can multiply W by any positive constant κ and divide R by κ to obtain the same intervention. To avoid this problem, prior arts in asymmetric low-rank optimization methods need to be regularized using the norm term $\|W^\top W - R^\top R\|_2^2$ (Tu et al., 2016; Bhojanapalli et al., 2016), closing the norm gap of the scale between W and R. Compared with them, the optimization problem (2) mentioned in the later section does not need to rescale the two low-rank matrices $L(a), R$ from the boundness of $\tanh(\cdot)$. The properties also stabilize the training process.

Finally, we consider the degenerate case: when $W = 0$ and ϕ is a linear function, then $L(a) = b$, and the intervention function becomes $f(a) = (I + bR^\top)a + c$. This is the classical bilinear low-rank mapping which does not depend on a.

3 Probe-Free Intervention Framework

We propose a probe-free intervention framework based on the above low-rank mapping. It automatically does the following tasks: (i) models the desirable answer region, (ii) computes the intervention parameter through solving the related smooth optimization problem. We start by presenting the data matrix and framework setting.

From the training data, we can collect the activations of the desirable output and form a matrix $G_q \in \mathbb{R}^{d\times|\mathcal{G}(q)|}$ for each head. Each column of

G_q is the activation of one desirable answer. Similarly, we could construct the matrix $B_q \in \mathbb{R}^{d\times|\mathcal{B}(q)|}$ under the same pattern.

Let $\mathcal{M}_q$ be a generic manifold representing the region of the desirable answer data points. The nonlinear low-rank mapping f is trained by solving

$$\min_f \sum_q \sum_{i\in\mathcal{B}(q)\cup\mathcal{G}(q)} c_q(f(a_i), \mathrm{Proj}_{\mathcal{M}_q}(f(a_i))), \tag{2}$$

where c_q denotes the distance measure for question q, $f(a_i)$ denotes the terminal point of the intervention on an activation a_i, and $\mathrm{Proj}_{\mathcal{M}_q}$ denotes the projection onto the manifold $\mathcal{M}_q$. The performance of the intervention depends on the choice of c_q, and the manifold $\mathcal{M}_q$. The manifold $\mathcal{M}_q$ should have semantic interpretation in modeling and can separate the desirable and undesirable answers, and solving problem (2) should be computationally efficient.

We now describe one specific instance of our framework. The manifold $\mathcal{M}_q$ for question q is chosen as an ellipsoid of the form

$$\mathcal{M}_q = \{x : (x-\hat{\mu}_q)^\top \hat{\Sigma}_q^{-1}(x-\hat{\mu}_q) \le \rho_q\} \tag{3a}$$

where $\hat{\mu}_q$ is the center of the ellipsoid, $\hat{\Sigma}_q$ is a positive definite matrix that prescribes the shape, or orientation, of the ellipsoid, and ρ_q is the radius of the ellipsoid.

Moreover, we choose c_q as the squared Mahalanobis distance (De Maesschalck et al., 2000)

$$c_q(a, a') = (a-a')^\top \hat{\Sigma}_q^{-1}(a-a'), \tag{3b}$$

which is the distance of the candidate point from the center of mass divided by the width of the ellipsoid in the direction of the candidate point. This distance is rooted in the construction of the manifold (3a). We use the following projection:

$$\mathrm{Proj}_{\mathcal{M}_q}(y) = \arg\min_{x\in\mathcal{M}_q} c_q(x, y). \tag{3c}$$

Theorem 3.1 gives the optimization problem under the Mahalanobis distance with analytical projection expression.

Theorem 3.1 (Ellipsoidal manifold and Mahalanobis distance)**.** *Suppose that the manifold, the distance measure, and the projection operator are chosen as in* (3)*. Then problem* (2) *becomes*

$$\min_f \sum_q \sum_{i\in\mathcal{B}(q)\cup\mathcal{G}(q)} \left[\left(\sqrt{c_q(f(a_i), \hat{\mu}_q)} - \sqrt{\rho_q}\right)_+\right]^2, \tag{4}$$

where $(y)_+ = \max\{0, y\}$.

The objective function for the optimization problem is easy to compute since we transform the distance between $(I + L(a)R)a + c$ and its projection to the distance between $(I + L(a)R)a + c$ and a fixed vector $\hat{\mu}_q$. For the proof of Theorem 3.1, readers can refer to the Appendix for details. We now discuss how to derive the parameters to specify the manifold. The main challenge here is the high dimensionality of the activation vectors ($D = 4096$ for Llama3-8B), while for each question q, we observe less than ten samples in total. We will next present a practical method to estimate $\hat{\mu}_q$, $\hat{\Sigma}_q$, and the radius ρ_q. From the training data, we first compute the question-specific mean

$$\hat{\mu}_q^+ = \frac{1}{|\mathcal{G}(q)|}\sum_{i=1}^{|\mathcal{G}(q)|}(G_q)_i \quad \forall q, \tag{5}$$

and also the overall mean

$$\hat{\mu}^+ = \frac{1}{N^+}\sum_q \sum_{i=1}^{|\mathcal{G}(q)|}(G_q)_i$$

for the desirable subset of the data consisting of N^+ samples. Analogously, we compute $\hat{\mu}_q^-$ and $\hat{\mu}^-$ for the *un*desirable subset.

Due to the noise incurred by computing $\hat{\mu}_q^+$ using small sample sizes, we need to inject more information to construct the mean vector $\hat{\mu}_q$. We propose the following extrapolation scheme

$$\hat{\mu}_q = \hat{\mu}_q^+ + \lambda\left(\alpha \underbrace{\frac{\hat{\mu}_q^+ - \hat{\mu}_q^-}{\|\hat{\mu}_q^+ - \hat{\mu}_q^-\|}}_{\text{question specific}} + (1-\alpha)\underbrace{\frac{\hat{\mu}^+ - \hat{\mu}^-}{\|\hat{\mu}^+ - \hat{\mu}^-\|}}_{\text{overall}}\right), \tag{6}$$

where $\lambda > 0$ controls the magnitude of extrapolation. The extrapolation direction consists of two terms: a question-specific direction dictated by $\hat{\mu}_q^+ - \hat{\mu}_q^-$, and an overall direction dictated by $\hat{\mu}^+ - \hat{\mu}^-$. Both terms aim to guide $\hat{\mu}_q$ *away from* the *un*desirable values $\hat{\mu}_q^-$ and $\hat{\mu}^-$. The parameter $\alpha \in [0, 1]$ controls the relative strength between the question-specific (local) and overall (global) directions. Li et al. (2024b) proposed a translation of the activation along the truthful direction $\hat{\mu}^+ - \hat{\mu}^-$, which coincides with the overall direction in our formula. The extrapolation direction is illustrated in Figure 1, where d_{move} refers to $\hat{\mu}_q - \hat{\mu}_q^+$.

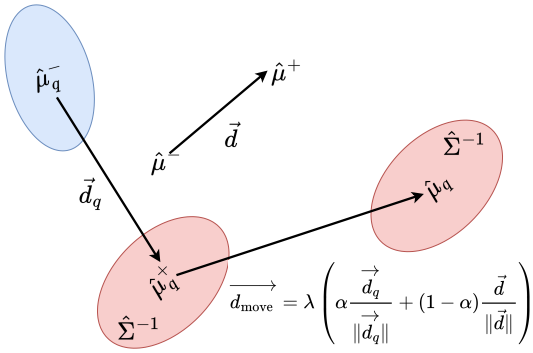

Figure 1: Convex combination of question-specific and overall extrapolation direction

For the covariance matrix, we assume that $\hat{\Sigma}_q^{-1}$ are equal across all q. This assumption is the founding element of linear discriminant analysis in machine learning (Tharwat et al., 2017). We first compute the empirical covariance matrix

$$S = \frac{1}{|N^+| - 1} \sum_q \sum_{i=1}^{|\mathcal{G}(q)|} \left[(G_q)_i - \hat{\mu}_q\right] \left[(G_q)_i - \hat{\mu}_q\right]^\top. \tag{7}$$

Due to small sample sizes, S may not be invertible. We use the linear shrinking method (Schäfer and Strimmer, 2005; Ledoit and Wolf, 2004) towards a diagonal matrix target, and set

$$\hat{\Sigma}_q^{-1} = (\beta S + (1-\beta)\text{diag}(S))^{-1} \quad \forall q,$$

where $\beta \in (0, 1)$ is the shrinkage paramter.

For estimating the ρ, the heuristic way is to find the minimum ρ that all of the samples stay in the desirable answer region with the center $\hat{\mu}_q^+$. For question q, we set

$$\rho_q = \max_{i \in |\mathcal{G}_q|} [(G_q)_i - \hat{\mu}_q^+]^\top \hat{\Sigma}_q^{-1} [(G_q)_i - \hat{\mu}_q^+].$$

4 Algorithms for Training Nonlinear Low-Rank Mapping

Let $\mathcal{F}(f)$ be the objective function of problem (2). Since f is parametrized in (1) by three low-rank matrices $W,\ R,\ b \in \mathbb{R}^{D \times k}$ and vector c, the objective function is also denoted by $\mathcal{F}(W, R, b, c)$.

The degenerated bilinear low-rank optimization problem $\min_{b,R} f(bR^\top)$ has many applications in the statistical estimation field (e.g., matrix sensing, matrix recovery, PCA) (Chi et al., 2019). The literature shows that simple methods such as gradient descent and alternating minimization could converge to the ground truth under mild assumptions with good initialization. In contrast, our objective function can not be represented by the factorization formulation $\mathcal{F}(bR^\top)$ and contains an extra nonlinear activation function. Therefore, it is hard to derive similar results in the bilinear low-rank optimization. However, the first-order algorithm is still a good choice with lightweight computational complexity. To proceed, we first analyze the standard gradient descent based on β-smoothness. In addition, we propose the gradient descent with precondition, which is a powerful approach popularized in (bilinear) low-rank optimization since it balances the scale in each iteration and speeds up the convergence while keeping the same computational complexity order.

4.1 Gradient Descent

We emphasize that the $\tanh(\cdot)$ stabilizes the training and optimization process in gradient descent. The derivative of the function is $1 - \tanh^2$, which allows for steeper gradients in the region close to zero to help do gradient descent more effectively. When the outputs are centered around zero (i.e., the elements $Wa + b$ and c approach zero in our problem, giving no modification of the activation), the gradients of the function are less likely to diminish significantly. In addition, $\tanh(\cdot)$ is C^∞ and by composition with $\max\{0, x\}^2 \in \mathcal{C}^{1,1}$, the objective function in our problem is β-smooth.

In the following, we introduce a technique tailored for low-rank optimization, accelerating its performance with little extra computation burden.

4.2 Gradient Descent with Preconditioning

We propose to use gradient descent with precondition (namely scaled gradient descent) to update the low-rank matrices as follows

$$\begin{aligned} W_{t+1} &= W_t - \eta \nabla_W \mathcal{F}(W_t, R_t, b_t, c_t) \left(R_t^\top R_t + \epsilon I\right)^{-1}, \\ R_{t+1} &= R_t - \eta \nabla_R \mathcal{F}(W_t, R_t, b_t, c_t) \left(W_t^\top W_t + \epsilon I\right)^{-1}, \\ b_{t+1} &= b_t - \eta \nabla_b \mathcal{F}(W_t, R_t, b_t, c_t) \left(b_t^\top b_t + \epsilon I\right)^{-1}, \end{aligned} \tag{8}$$

where ϵ is a small number to ensure the scaling term is invertible. This perturbation is also used in other LM papers related to Riemannian precondition (Zhang and Pilanci, 2024). Comparing to vanilla gradient descent, the search directions of the low-rank factors W_t, R_t, b_t in (8) are scaled by $\left(W_t^\top W_t + \epsilon I\right)^{-1}$, $\left(R_t^\top R_t + \epsilon I\right)^{-1}$, and $\left(b_t^\top b_t + \epsilon I\right)^{-1}$respectively. Intuitively, the scaling

serves as a preconditioner, hoping to improve the quality of the search direction to allow larger step sizes. The precondition is adaptive and iteration-varying. Computationally, the scaled gradient descent adds minimal overhead to the gradient computation (the inversion of the small size $\mathbb{R}^{k\times k}$ matrix) and has the order-wise same per-iteration cost as gradient descent in each iteration. In bilinear low-rank matrix estimation problem $\min_{b,R} f(bR^\top)$ (Tong et al., 2021), it is proved to have condition number-free convergence rate faster than gradient descent under mild assumptions.

5 Empirical Results

This section presents the empirical results of our proposal named FLORAIN. Section 5.1 clarifies the setting of our experiments, including details about datasets, tasks, metrics, baselines, and computation resources. Section 5.2 showcases the superiority of our proposed methods to other baselines regarding numerical results.

5.1 Experimental Setup

Tasks and Metrics: We evaluate our framework on the multiple-choice and generation tasks, which are commonly used to verify the truthfulness of language models:

- The generation task: the model generates an entire answer for each question using greedy autoregressive decoding. While the accuracy and informativeness of answers are best evaluated by humans, this approach is expensive and laborious. Consequently, most works in this field use a well-trained large language model as an evaluation alternative. We adopt two fine-tuned `GPT-3.5-instruct` models, one for classifying whether the answer is correct or incorrect and the other for classifying whether the response is informative. According to Li et al. (2024b), the percentage of answers labeled as correct by the first model is denoted as the truthful score, True (%). In the meantime, the percentage of answers labeled as informative by the second model is the informative score, Info (%). Following Li et al. (2024b), we report truthful scores True (%) and True*Info (%), a product of scalar truthful and informative scores.

- The multiple-choice task: the model computes the log probability of completion for a given question and set of choices. Following Lin et al. (2021), we report two metrics: MC1 and MC2. MC1 identifies the only correct answer with the highest probability from the choices, with overall accuracy across all questions denoted as MC1. Likewise, MC2 is the normalized total probability assigned to the set of true answers given a question and multiple true/false reference answers.

- We also include two other metrics named Kullback-Leiber divergence (KL) of the model's next-token prediction distribution post-versus-pre-intervention and Cross-Entropy Loss (CE). These two metrics assess the extent to which the generation distribution shifts after the intervention. Lower values are preferred, indicating that the intervention minimally alters the original model's behavior, reducing the likelihood of producing abnormal characters or unnatural sentences. The calculation of these metrics is detailed in Li et al. (2024b).

Datasets: We use the TruthfulQA dataset to benchmark the effectiveness of FLORAIN. This dataset contains 817 questions across 38 categories, designed to provoke false answers from language models, making it challenging to generate accurate responses. Following the dataset split from Li et al. (2024b) and Yin et al. (2024), we divide the 817 questions into training, validation, and test sets, with 326, 82, and 407 questions, respectively. We also use the 2-fold cross-validation for our evaluation, same as Li et al. (2024b) and Yin et al. (2024). Additionally, we process the dataset into 5,918 question-answer pairs, each with a binary label indicating desirability. Only question-answer pairs from the training set are used to develop our intervention policy, while the validation set is reserved for parameter tuning.

Models We apply our method to multiple pre-trained models to enhance their truthfulness, including Llama-7B (Touvron et al., 2023b), Llama2-chat-13B (Touvron et al., 2023a), and Llama3-8B (Dubey et al., 2024). Our method can be appended as an extension to finetuning approaches, aiding in eliciting more accurate answers. To demonstrate its broad effectiveness, we also evaluate it on models that have been fine-tuned for the same goal. These fine-tuned models can be viewed either as base models or as competitors to our method. Detailed descriptions of these models are provided in the baselines section.

Hyperparameters: We manipulate the behavior of FLORAIN by tuning two hyperparameters λ and α in determining the $\hat{\mu}_q$ by equation (6). The details about selecting them for each pre-trained model are discussed in Appendix A.

Baselines We compare FLORAIN against competitive baselines with the same goal of eliciting truthful answers from language models:

- Inference-time Intervention (ITI, Li et al. 2024b) is a state-of-the-art method that allows intervention without finetuning. We follow the hyperparameter settings provided in the original paper (Li et al., 2024b) and their GitHub repository [1].
- Few-shot prompting (FSP), introduced by Bai et al. (2022), has demonstrated the effectiveness of prompting with 50 examples on the TruthfulQA benchmark. In our experiments, we use the 50-shot version, selecting 50 prompt pairs from the training set.
- Instruction Fine-Tuning (IFT) (Wang et al., 2022; Chung et al., 2024) is a well-known approach for improving the truthfulness of language models through fine-tuning. For comparison, we include two prominent models in this category: Alpaca-7B (Taori et al., 2023) and Vicuna-7B (Chiang et al., 2023).

Computation resources. Our method and baselines run on two NVIDIA RTX A5000 GPUs, an i9 14900K CPU, and 128GB RAM.

Code. The anonymized repository is https://anonymous.4open.science/r/EFI-5CBD.

5.2 Numerical Results

5.2.1 Comparison between Truthfulness Augmentation Baselines across Different language models

Table 1 represents the numerical results of our method, ITI, Few-shot prompting (FSP) across three base models, including Llama-7B, Llama3-8B, and Llama2-chat-13B on the TruthfulQA dataset. It is worth noting that FSP is a prompt engineering technique and orthogonal with intervention methods like FLORAIN and ITI, so we include two additional versions that combine FSP with either ITI or FLORAIN. The combination of FSP + FLORAIN consistently obtained the top scores in performance metrics like True * Info and True across the three models, reaching 45% for Llama-7B, 42% for Llama3-8B, and 61% for Llama2-chat-13B. It is observed that when running alone, FLORAIN is superior to ITI in four main metrics of both two tasks: generation and multiple-choice, significantly enhancing the truthfulness and quality of pre-intervention models. Remarkably, FLORAIN improves the True * Info(%) score from 21.15 to 31.46, 32.88 to 36.78, and 51.87 to 60.68 for the Llama-7B, Llama3-8B, and Llama2-chat-13B, respectively. The KL values are minimal across the board, indicating the intervention effect is not overly aggressive.

5.2.2 Comparison between ITI, FLORAIN, and Instruction Finetuning Methods.

In this experiment, we benchmark two instruction fine-tuning methods, Alpaca and Vicuna, verifying whether implementing our method on top of these finetuned models further improves their quality and truthfulness. In Table 2, FLORAIN showcases its remarkable effectiveness, boosting the True * Info score by around 8.07% and 14.21% for the Alpaca and Vicuna models, respectively. In both models, FLORAIN takes the lead in all quality metrics, leaving ITI far behind. This highlights FLORAIN's effectiveness in enhancing the accuracy and truthfulness of the models.

6 Conclusion and Future Directions

In this paper, we introduced FLORAIN, a novel probe-free low-rank intervention framework for activation intervention. The intervention mapping seeks to minimize the distance between the vector after intervention and its projection onto the desirable answer region, which is an ellipsoid carefully estimated by first-order and second-order statistic information. Contrary to existing intervention methods, our probe-free low-rank intervention does not require the classifier to detect responsible heads/layers for intervention. Instead, it gives the optimization problem with novel region modeling and analytical projection. Moreover, it is conducted in a single layer of the output transformer network, which allows for efficient parallel computing and decreases the distribution shift phenomena. One future direction of this research line is trying to extend the region modeling approach, e.g., considering the subspace spanned by the desirable answer matrix as the desirable answer region.

[1] https://github.com/likenneth/honest_llama/tree/master

Table 1: Quantitative results of different methods on TruthfulQA dataset, across different Language Models.

Methods	True * Info (%) ↑	True (%) ↑	MC1 ↑	MC2 ↑	CE ↓	KL ↓
Unintervened	21.15	22.16	25.58	40.54	2.13	0.00
ITI	26.52	28.03	27.78	43.59	2.20	0.07
FLORAIN (ours)	**31.46**	**34.72**	**31.76**	**47.43**	2.21	0.09
FSP	36.13	39.78	34.03	50.34	2.13	0.00
FSP + ITI	40.63	45.16	35.50	52.48	2.20	0.07
FSP + FLORAIN (ours)	**45.31**	**49.23**	**36.45**	**54.27**	2.20	0.08

(a) Llama-7B

Methods	True * Info (%) ↑	True (%) ↑	MC1 ↑	MC2 ↑	CE ↓	KL ↓
Unintervened	32.88	44.18	30.36	48.98	2.38	0.00
ITI	35.92	46.88	32.07	49.84	2.50	0.13
FLORAIN (ours)	**36.78**	**48.67**	**34.56**	**53.68**	2.51	0.14
FSP	36.32	39.78	35.74	52.93	2.38	0.00
FSP + ITI	40.63	45.16	35.50	52.98	2.48	0.14
FSP + FLORAIN (ours)	**42.15**	**47.32**	**36.98**	**55.83**	2.51	0.16

(b) Llama3-8B

Methods	True * Info (%) ↑	True (%) ↑	MC1 ↑	MC2 ↑	CE ↓	KL ↓
Unintervened	51.87	59.86	35.38	53.32	2.31	0.00
ITI	57.02	63.04	37.46	55.59	2.32	0.17
FLORAIN (ours)	**60.68**	**67.70**	**39.65**	**59.57**	2.35	0.18
FSP	55.97	58.63	40.76	57.84	2.31	0.00
FSP + ITI	56.78	59.24	41.50	59.01	2.33	0.13
FSP + FLORAIN (ours)	**61.14**	**62.45**	**44.52**	**61.48**	2.37	0.16

(c) Llama2-chat-13B

Table 2: Quantitative results of intervention methods on instruction-finetuned models Alpaca and Vicuna.

Methods	True*Info (%) ↑	True (%) ↑	MC1 ↑	MC2 ↑	CE ↓	KL ↓
Alpaca	30.39	30.85	26.56	41.63	2.81	0.00
Alpaca + ITI	37.67	38.19	28.89	45.19	2.88	0.14
Alpaca + FLORAIN (ours)	**38.56**	**40.52**	**31.32**	**46.83**	2.85	0.15
Vicuna	38.24	42.10	31.83	48.48	2.67	0.00
Vicuna + ITI	49.27	53.25	33.42	51.80	2.77	0.26
Vicuna + FLORAIN (ours)	**52.45**	**57.81**	**36.86**	**56.76**	2.78	0.27

7 Limitations

Our method relies on the scale of the training dataset. If the number of training samples is low, it will be hard to construct the ellipsoid region since the estimation mean and covariance matrix can lead to higher bias compared with the sufficient training sample case. Our current training dataset includes 326 samples.

For the smooth optimization problem we derived in the main context, its nonconvexity structure, and our unawareness of the benign landscape can make the algorithm converge to some local minimizers. Therefore, our first-order algorithm does not solve the optimization problem to global optimal with a theoretical guarantee. In solving the problem, we use standard gradient descent and do not implement the algorithm with precondition.

References

AI Anthropic. 2024. The Claude 3 model family: Opus, Sonnet, Haiku. *Claude-3 Model Card*.

Bai et al. 2022. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*.

Srinadh Bhojanapalli, Anastasios Kyrillidis, and Sujay Sanghavi. 2016. Dropping convexity for faster semi-definite optimization. In *Conference on Learning Theory*, pages 530–582. PMLR.

Brown et al. 2020. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33:1877–1901.

Collin Burns, Haotian Ye, Dan Klein, and Jacob Steinhardt. 2022. Discovering latent knowledge in language models without supervision. *arXiv preprint arXiv:2212.03827*.

Yuejie Chi, Yue M Lu, and Yuxin Chen. 2019. Nonconvex optimization meets low-rank matrix factorization: An overview. *IEEE Transactions on Signal Processing*, 67(20):5239–5269.

Chiang et al. 2023. Vicuna: An open-source chatbot impressing GPT-4 with 90% chatgp quality. *See https://vicuna. lmsys. org (accessed 14 April 2023)*, 2(3):6.

Chung et al. 2024. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, 25(70):1–53.

Ryan Cory-Wright, Jean Pauphilet, and Andrea Lodi. 2021. A new perspective on low-rank optimization. *arXiv preprint arXiv:2105.05947*.

Roy De Maesschalck, Delphine Jouan-Rimbaud, and Désiré L Massart. 2000. The mahalanobis distance. *Chemometrics and Intelligent Laboratory Systems*, 50(1):1–18.

Abhimanyu Dubey et al. 2024. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Shane Griffith, Kaushik Subramanian, Jonathan Scholz, Charles L Isbell, and Andrea L Thomaz. 2013. Policy shaping: Integrating human feedback with reinforcement learning. *Advances in Neural Information Processing Systems*, 26.

Beliz Gunel, Jingfei Du, Alexis Conneau, and Ves Stoyanov. 2020. Supervised contrastive learning for pre-trained language model fine-tuning. *arXiv preprint arXiv:2011.01403*.

Evan Hernandez, Belinda Z Li, and Jacob Andreas. 2023. Measuring and manipulating knowledge representations in language models. *arXiv preprint arXiv:2304.00740*.

Yen-Chang Hsu, Ting Hua, Sungen Chang, Qian Lou, Yilin Shen, and Hongxia Jin. 2022. Language model compression with weighted low-rank factorization. *arXiv preprint arXiv:2207.00112*.

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

Jett Janiak, Jacek Karwowski, Chatrik Singh Mangat, Giorgi Giglemiani, Nora Petrova, and Stefan Heimersheim. 2024. Characterizing stable regions in the residual stream of LLMs. *arXiv preprint arXiv:2409.17113*.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38.

Albert Q Jiang et al. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Olivier Ledoit and Michael Wolf. 2004. A well-conditioned estimator for large-dimensional covariance matrices. *Journal of Multivariate Analysis*, 88(2):365–411.

Junyi Li, Tianyi Tang, Wayne Xin Zhao, Jian-Yun Nie, and Ji-Rong Wen. 2024a. Pre-trained language models for text generation: A survey. *ACM Computing Surveys*, 56(9):1–39.

Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. 2024b. Inference-time intervention: Eliciting truthful answers from a language model. *Advances in Neural Information Processing Systems*, 36.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2021. TruthfulQA: Measuring how models mimic human falsehoods. *arXiv preprint arXiv:2109.07958*.

Jonathan Mamou, Hang Le, Miguel Del Rio, Cory Stephenson, Hanlin Tang, Yoon Kim, and SueYeon Chung. 2020. Emergence of separable manifolds in deep language representations. *arXiv preprint arXiv:2006.01095*.

Andrew D McRae and Nicolas Boumal. 2024. Benign landscapes of low-dimensional relaxations for orthogonal synchronization on general graphs. *SIAM Journal on Optimization*, 34(2):1427–1454.

Guillaume Olikier, P-A Absil, and Julien M Hendrickx. 2022. Low-rank optimization methods based on projected-projected gradient descent that accumulate at bouligand stationary points. *arXiv preprint arXiv:2201.03962*.

Long Ouyang et al. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744.

Baolin Peng, Chunyuan Li, Pengcheng He, Michel Galley, and Jianfeng Gao. 2023. Instruction tuning with GPT-4. *arXiv preprint arXiv:2304.03277*.

Van-Cuong Pham and Thien Huu Nguyen. 2024. Householder pseudo-rotation: A novel approach to activation editing in LLMs with direction-magnitude perspective. *arXiv preprint arXiv:2409.10053*.

Radford et al. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9.

Vipula Rawte, Swagata Chakraborty, Agnibh Pathak, Anubhav Sarkar, SM Tonmoy, Aman Chadha, Amit P Sheth, and Amitava Das. 2023. The troubling emergence of hallucination in large language models–an extensive definition, quantification, and prescriptive remediations. *arXiv preprint arXiv:2310.04988*.

Juliane Schäfer and Korbinian Strimmer. 2005. A shrinkage approach to large-scale covariance matrix estimation and implications for functional genomics. *Statistical Applications in Genetics and Molecular Biology*, 4(1).

Shashwat Singh, Shauli Ravfogel, Jonathan Herzig, Roee Aharoni, Ryan Cotterell, and Ponnurangam Kumaraguru. 2024. MiMiC: Minimally modified counterfactuals in the representation space. *arXiv preprint arXiv:2402.09631*.

Nishant Subramani, Nivedita Suresh, and Matthew E Peters. 2022. Extracting latent steering vectors from pretrained language models. *arXiv preprint arXiv:2205.05124*.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. 2023. Alpaca: A strong, replicable instruction-following model. *Stanford Center for Research on Foundation Models*, 3(6):7.

Alaa Tharwat, Tarek Gaber, Abdelhameed Ibrahim, and Aboul Ella Hassanien. 2017. Linear discriminant analysis: A detailed tutorial. *AI communications*, 30(2):169–190.

Tian Tong, Cong Ma, and Yuejie Chi. 2021. Accelerating ill-conditioned low-rank matrix estimation via scaled gradient descent. *Journal of Machine Learning Research*, 22:1–63.

Hugo Touvron et al. 2023a. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Hugo Touvron et al. 2023b. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Stephen Tu, Ross Boczar, Max Simchowitz, Mahdi Soltanolkotabi, and Ben Recht. 2016. Low-rank solutions of linear matrix equations via Procrustes Flow. In *International Conference on Machine Learning*, pages 964–973. PMLR.

Wang et al. 2023. Knowledge editing for large language models: A survey. *arXiv preprint arXiv:2310.16218*.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2022. Self-instruct: Aligning language models with self-generated instructions. *arXiv preprint arXiv:2212.10560*.

Ziwei Xu, Sanjay Jain, and Mohan Kankanhalli. 2024. Hallucination is inevitable: An innate limitation of large language models. *arXiv preprint arXiv:2401.11817*.

Fangcong Yin, Xi Ye, and Greg Durrett. 2024. Lofit: Localized fine-tuning on LLM representations. *arXiv preprint arXiv:2406.01563*.

Zhang et al. 2024. A comprehensive study of knowledge editing for large language models. *arXiv preprint arXiv:2401.01286*.

Fangzhao Zhang and Mert Pilanci. 2024. Riemannian preconditioned LoRA for fine-tuning foundation models. *arXiv preprint arXiv:2402.02347*.

Hanqing Zhang, Haolin Song, Shaoyu Li, Ming Zhou, and Dawei Song. 2023. A survey of controllable text generation using transformer-based pre-trained language models. *ACM Computing Surveys*, 56(3):1–37.

Hu Zhanxuan, Nie Feiping, Wang Rong, and Li Xuelong. 2020. Low rank regularization: A review. *Information Sciences*.

A Hyper-parameter tuning

In our framework, two key hyperparameters are λ and α. We determine their values based on an evaluation of performance metrics and text generation quality on the validation set. Through a grid search over $\{2, 3, 4, 5\}$ for λ and $\{0.0, 0.2, 0.4, 0.6, 0.8, 1.0\}$ for α, we find that setting $\alpha = 0.2$ and $\lambda = 5$ yields strong performance across various cases. As a result, unless otherwise specified, we adopt this combination for all our experiments.

Additionally, the choice of the intervened layer is crucial. Given the limited number of layers, we can efficiently conduct a search procedure to optimize validation test performance. Specifically, we intervene at layer 11 for Llama-7B, Alpaca-7B, and Vicuna-7B, layer 12 for Llama3-8B, and layer 14 for Llama2-chat-13B.

B Proofs

B.1 Proof of Theorem 3.1

The intervention maps the activation of undesirable answer $a_i, i \in \mathcal{B}(q)$ onto the ellipsoid region of the desirable answers, which can be defined below.

$$\mathcal{E}_q = \{x : (x - \hat{\mu}_q)^\top \hat{\Sigma}_q^{-1} (x - \hat{\mu}_q) \leq \rho_q\} \quad (9)$$

In the proof part, we simplify the notation of $\hat{\Sigma}_q$ (resp. $\mathcal{E}_q, \hat{\mu}_q$) as Σ (resp. $\mathcal{E}, \mu$) for clarity. Let the Mahalanobis projection be

$$\mathrm{Proj}_{\mathcal{E}}(y) = \arg\min_{x \in \mathcal{E}} \; (y - x)^\top \Sigma^{-1} (y - x),$$

where

$$\mathcal{E} = \{x : (x - \mu)^\top \Sigma^{-1} (x - \mu) \leq \rho\}.$$

Proof of Theorem 1. If $y \in \mathcal{E}$, the result can be immediately derived by the definition of the projection. When $y \notin \mathcal{E}$, we can transform the coordinates of x, y to simplify the projection problem. We define

$$\hat{y} = \Sigma^{-\frac{1}{2}} y,$$

$$\hat{x} = \Sigma^{-\frac{1}{2}} x.$$

The projection problem has been transformed into

$$\arg\min_{\hat{x} \in \hat{\mathcal{E}}} \|\hat{y} - \hat{x}\|_2^2,$$

where

$$\hat{\mathcal{E}} = \{\hat{x} : \|\hat{x} - \Sigma^{-\frac{1}{2}} \mu\|_2^2 \leq \rho\}.$$

The transformed problem allows for closed form solution for the point outside of the region

$$\begin{aligned} \hat{x}^* &= \Sigma^{-\frac{1}{2}} \mu + \sqrt{\rho} \frac{\hat{y} - \Sigma^{-\frac{1}{2}} \mu}{\|\hat{y} - \Sigma^{-\frac{1}{2}} \mu\|_2} \\ &= \Sigma^{-\frac{1}{2}} \mu + \sqrt{\rho} \frac{\Sigma^{-\frac{1}{2}} y - \Sigma^{-\frac{1}{2}} \mu}{\|\Sigma^{-\frac{1}{2}} y - \Sigma^{-\frac{1}{2}} \mu\|_2}. \end{aligned}$$

Therefore

$$\begin{aligned} x^* &= \mu + \sqrt{\rho} \frac{y - \mu}{\|\Sigma^{-\frac{1}{2}} y - \Sigma^{-\frac{1}{2}} \mu\|} \\ &= \mu + \frac{\sqrt{\rho}}{\sqrt{(y - \mu)^T \Sigma^{-1} (y - \mu)}} (y - \mu). \end{aligned}$$

For an arbitrary q and $i \in \mathcal{B}(q) \cup \mathcal{G}(q)$,

$$\begin{aligned} & f(a_i) - \mathrm{Proj}(f(a_i)) \\ &= f(a_i) - \hat{\mu}_q \\ &- \frac{\sqrt{\rho_q}}{\sqrt{(f(a_i) - \hat{\mu}_q)^\top \hat{\Sigma}^{-1} (f(a_i) - \hat{\mu}_q)}} (f(a_i) - \hat{\mu}_q) \\ &= \left[1 - \frac{\sqrt{\rho_q}}{\sqrt{(f(a_i) - \hat{\mu}_q)^\top \hat{\Sigma}^{-1} (f(a_i) - \hat{\mu}_q)}}\right] (f(a_i) - \hat{\mu}_q). \end{aligned}$$

By computation, if $f(a_i) \in \mathcal{M}_q$

$$c_q(f(a_i), \mathrm{Proj}_{c_q, \mathcal{M}_q}(f(a_i)) = 0.$$

Otherwise, the distance is

$$[\sqrt{(f(a_i) - \hat{\mu}_q)^\top \hat{\Sigma}^{-1} (f(a_i) - \hat{\mu}_q)} - \sqrt{\rho}]^2.$$

We complete the proof. □