PRISM: Principal Subspace Alignment for Parameter-Efficient Fine-Tuning

Anh Tong
Department of Artificial Intelligence
Korea University
anhtong12@korea.ac.kr

Kyowoon Lee
Department of Mobility Engineering
Ajou University
kyowoonlee@ajou.ac.kr

Jaesik Choi
Kim Jaechul Graduate School of AI
KAIST, INEEJI
jaesik.choi@kaist.ac.kr

Abstract

Parameter-efficient fine-tuning (PEFT) methods such as low-rank adaptation have become essential for adapting large language models to downstream tasks. Existing approaches often apply low-rank updates uniformly across all layers, ignoring the structural coupling inherent in transformer architectures. We observe that transformers contain *weight pairs* whose interactions govern model behavior including query-key products, value-output products, and feedforward networks. We propose PRincipal Interaction Subspace Matching (PRISM), a principled framework that exploits this coupling through spectral parameterization. For each weight pair, we parameterize adaptations using the spectral basis of the paired matrix, ensuring that updates occur in the subspace where the paired matrix has maximal effect. We validate our approach through controlled experiments on synthetic models, demonstrating faster convergence. Empirical evaluations on arithmetic reasoning, code generation, and vision tasks show that PRISM outperforms existing PEFT methods.

1 Introduction

Transformers [Vaswani et al., 2017] have proven extraordinarily capable across vision, language, and beyond [Brown et al., 2020, Dosovitskiy et al., 2021, Peebles and Xie, 2023]. However, full fine-tuning of billion-parameter models requires prohibitive memory and compute resources, motivating a wave of parameter-efficient fine-tuning (PEFT) methods designed to adapt large models cheaply. Among these, Low-Rank Adaptation (LoRA) [Hu et al., 2022] has become the de facto standard. The idea is elegant: instead of updating an entire weight matrix, approximate the update as a product of two low-rank matrices. This reduces trainable parameters to a small fraction of the original model while preserving much of its expressive capacity. Subsequently, it leads to a rich ecosystem of extensions [Zi et al., 2023, Babakniya et al., 2023, Kopiczko et al., 2024, Nikdan et al., 2024, Paischer et al., 2024, Tian et al., 2024, Bałazy et al., 2024].

However, LoRA and its variants share a common assumption that each weight matrix can be adapted *in isolation*. This means the adaption is blindly unawared about architectures. On the other hand, transformers are not just composed of independent matrices. They are built around structured pairings. Query matrix and key matrix jointly govern how information is routed through the residual stream. Value matrix and output matrix together control how that routed information is integrated [Elhage et al., 2021]. The two-layer feed-forward block functions as a coupled memory retrieval mechanism [Geva et al., 2021]. These are not independent operations but functioning as units. Adapting one half of a pair while ignoring the other means just performing adjustment and overlooking alignment.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

We ask a simple but overlooked question: *what if fine-tuning respected the coupling structure that transformers were designed around?*

To answer it, we introduce PRincipal Interaction Subspace Matching (PRISM), a novel PEFT method that treats coupled weight pairs as the fundamental unit of adaptation. Rather than independently parameterizing each update, PRISM derives matched adaptation rules from the interaction structure of each pair. Concretely, each weight update is parameterized using the *principal subspace of its coupled partner* as the directions that actually matter for their composed output. We provide geometric intuition that when two matrices compose, only the components of one that are aligned with the principal subspace of the other survive meaningfully in the output. We further prove that, without task-specific information, the partner's principal subspace is the optimal fixed rank-r basis in both the worst case and the isotropic average case. To enable parameter efficiency, PRISM allocates its low-rank budget precisely where it counts.

We validate PRISM through comprehensive experiments spanning both language and vision. Across language models ranging from 7B to 13B parameters, we evaluate on arithmetic reasoning, code generation, and commonsense reasoning benchmarks. Across vision transformers, we assess adaptation on diverse visual tasks. PRISM achieves state-of-the-art results in the majority of settings. Notably its convergence is faster compared to existing PEFT baselines, suggesting that aligned updates are not only more accurate but also more efficient optimization trajectories.

Our main contributions are as follows: **(1)** We identify weight pair coupling as a fundamental architectural property systematically overlooked by existing PEFT methods, and propose *the first adaptation framework* that explicitly exploits this structure through matched subspace parameterizations. **(2)** We provide geometric justifications grounding PRISM in the interaction structure of composed matrix pairs, prove that the paired principal basis is minimax- and average-case optimal among fixed rank-r bases for realizable first-order targets, and show empirically that unconstrained LoRA updates already concentrate most of their energy in this subspace. **(3)** We demonstrate consistent empirical improvements across both language and vision domains, with accelerated convergence as an additional practical benefit.

2 Related work

Low-Rank Adaptation (LoRA) LoRA [Hu et al., 2022] has emerged as the dominant PEFT approach, freezing pretrained weights while injecting trainable low-rank decomposition matrices into each layer. This reduces trainable parameters by orders of magnitude while achieving comparable performance. The success of LoRA has led to numerous variants exploring quantization [Dettmers et al., 2023, Li et al., 2023], adaptive rank allocation [Zhang et al., 2023b], improved optimization [Hayou et al., 2024, Wang et al., 2024], weight decomposition [Liu et al., 2024, Chang et al., 2025], and architectural modifications [Zi et al., 2023, Babakniya et al., 2023, Kopiczko et al., 2024, Nikdan et al., 2024, Paischer et al., 2024, Tian et al., 2024, Bałazy et al., 2024, Chen et al., 2024, Koike-Akino et al., 2025]. LoRA-FA [Zhang et al., 2023a] proposes training only one low-rank matrix while keeping the other randomly initialized and frozen. Beyond low-rank methods, orthogonal fine-tuning [Qiu et al., 2023, Liu et al., Zhang and Pilanci, 2024, Nguyen et al., 2026] constrains updates to preserve pretrained geometry, while recent work [Huang et al., 2025a, Albert et al., 2025, Singhal et al., 2026, Nguyen and Tong, 2026] explores high-rank adaptation for increased expressiveness. For instance, HiRA [Huang et al., 2025a] and ABBA [Singhal et al., 2026] change the *form* of the update through Hadamard products, and AdaLoRA [Zhang et al., 2023b] changes *how much* rank each matrix receives. However, all these methods adapt each weight matrix independently, ignoring the structural coupling inherent in transformer architectures. PRISM explores a new paradigm that coordinates adaptations across coupled weight pairs in transformers. It constrains the *basis* of the update using the paired matrix, a design axis orthogonal to update form and rank allocation, so these mechanisms could in principle be combined with PRISM. PRISM shares the train-one-factor strategy of LoRA-FA, but its frozen factor is the principal basis of the paired matrix rather than a random projection.

Spectral-based Adaptation Several methods leverage spectral information from pretrained weights for better initialization. PiSSA [Meng et al., 2024] initializes LoRA adapters using the principal singular values and vectors of pretrained weights, directly modifying the most relevant components while keeping residual components frozen. HD-PiSSA [Wang et al., 2025b] extends this by distributing orthogonal adapters initialized from disjoint principal components across multiple devices. Conversely, MiLoRA [Wang et al., 2025a] updates only the minor singular components of weight matrices, which limits interference with pretrained knowledge but places the update in directions

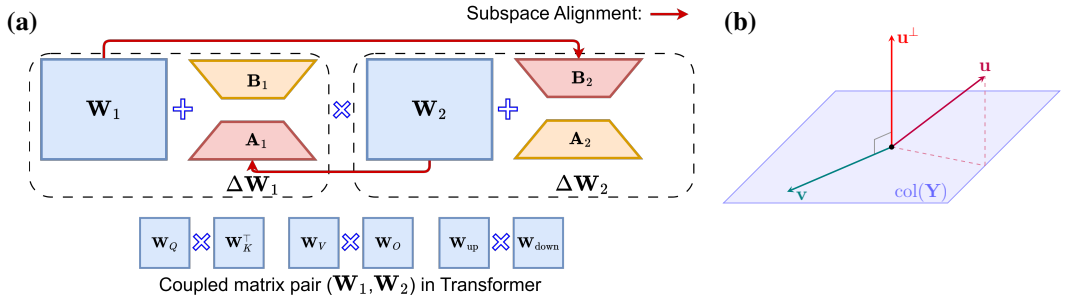

Figure 1: **(a) PRISM overview.** We adapt weight pairs $(\mathbf{W}_1, \mathbf{W}_2)$ by $(\Delta\mathbf{W}_1, \Delta\mathbf{W}_2)$. The low-rank adaptation matrix $\Delta\mathbf{W}_1$ is aligned with the principal subspace of $\mathbf{W}_2$, while $\Delta\mathbf{W}_2$ is aligned with the principal subspace of $\mathbf{W}_1$. This applies to transformer pairs: $\mathbf{W}_Q\mathbf{W}_K^\top$, $\mathbf{W}_V\mathbf{W}_O$, $\mathbf{W}_{\text{up}}\mathbf{W}_{\text{down}}$. **(b) Subspace alignment illustration**. The matrix $\mathbf{Y}$ maps any input to its column space $\mathrm{col}(\mathbf{Y})$ (shaded plane). For a row $\mathbf{x}_i$ of $\mathbf{X}$ to maximally interact with $\mathbf{Y}$'s output, it must lie within this subspace (e.g., $\mathbf{v}$). Components orthogonal to $\mathrm{col}(\mathbf{Y})$ like $\mathbf{u}^\perp$ contribute nothing.

that the composition attenuates. OLoRA [Büyükakyüz, 2024] uses QR decomposition to introduce orthonormal transformations before adaptation. While these methods exploit spectral structure for initialization, they still adapt each weight matrix independently. In contrast, PRISM incorporates transformer architectural structure into its formulation, using spectral information to explicitly coordinate updates across coupled weight pairs. In particular, PiSSA and MiLoRA select directions from a matrix's *own* spectrum, whereas PRISM selects them from the spectrum of its *partner*.

Representation Fine-tuning and Activation Editing. A complementary line of work adapts models by intervening on hidden representations rather than on weights. Representation engineering [Zou et al., 2023] identifies and manipulates directions in activation space that correspond to high-level concepts. ReFT [Wu et al., 2024] learns low-rank interventions on hidden representations at selected positions and layers, using far fewer parameters than weight-based PEFT. LoFiT [Yin et al., 2024] first localizes a small set of attention heads and then learns offsets to their representations, and JoLA [Lai et al., 2025] jointly learns which heads to edit and how to edit them, with strong results in low-resource settings. These methods modify activations at chosen locations, whereas PRISM modifies weights under an architectural coupling prior. Because they act on different objects, the two approaches are complementary; for example, activation editing could be applied on top of PRISM-adapted weights.

Subspace Alignment. Subspace alignment has been explored in various transfer learning contexts. [Saha et al., 2021] identifies low-dimensional subspaces via SVD on activations for continual learning, while [Bhattarai et al., 2025] leverages findings that LLMs construct orthogonal subspaces for different concepts. In domain adaptation, [Fernando et al., 2014] represents source and target domains via eigenvector subspaces. Recent work [Xiao et al., 2025] recognizes that transformer weights operate in coupled pairs for model compression. In contrast, our work brings this coupled-pair perspective to fine-tuning using spectral alignment.

3 Main method

This section presents our main method, PRincipal Interaction Subspace Matching (PRISM). We begin with a brief review of LoRA to establish notation, then identify weight pair interactions in transformer architectures, and finally derive our spectral parameterization.

3.1 Background: Low-Rank Adaptation

LoRA [Hu et al., 2022] adapts a pretrained weight matrix $\mathbf{W} \in \mathbb{R}^{d\times k}$ by adding a low-rank update:

$$f_{\text{adapt}}(\mathbf{x}) = \mathbf{x}\mathbf{W} + \mathbf{x}\Delta\mathbf{W}, \quad \text{where} \quad \Delta\mathbf{W} = \mathbf{B}\mathbf{A}, \tag{1}$$

with $\mathbf{B} \in \mathbb{R}^{d\times r}$, $\mathbf{A} \in \mathbb{R}^{r\times k}$, $\mathbf{x} \in \mathbb{R}^d$, and rank $r \ll \min(d, k)$. During fine-tuning, $\mathbf{W}$ remains frozen while only $\mathbf{A}$ and $\mathbf{B}$ are trained. This approach significantly reduces trainable parameters while maintaining performance, motivated by the low intrinsic dimensionality of adaptation [Aghajanyan et al., 2021, Gur-Ari et al., 2018, Zhang et al., 2023c, Gauch et al., 2022].

However, LoRA applies this parameterization independently to each weight matrix, ignoring structural coupling between matrices in transformer architectures. We address this by designing adaptation strategies that respect weight pair interactions.

3.2 Weight Pair Interactions in Transformers

To motivate our approach, we identify that transformer architectures contained important weight pair interactions [Elhage et al., 2021].

Feed-forward block. Modern architectures replace the original two-layer MLP with a gated variant:

$$\text{FFN}(\mathbf{x}) = [\sigma(\mathbf{x}\mathbf{W}_{\text{gate}}) \odot (\mathbf{x}\mathbf{W}_{\text{up}})]\, \mathbf{W}_{\text{down}}, \tag{2}$$

where $\mathbf{x} \in \mathbb{R}^d$, $\mathbf{W}_{\text{gate}}, \mathbf{W}_{\text{up}} \in \mathbb{R}^{d \times d_{\text{inter}}}$, $\mathbf{W}_{\text{down}} \in \mathbb{R}^{d_{\text{inter}} \times d}$, and $\odot$ denotes element-wise multiplication. Letting $\mathbf{g}(\mathbf{x}) = \sigma(\mathbf{x}\mathbf{W}_{\text{gate}})$, we can express this as:

$$\text{FFN}(\mathbf{x}) = \mathbf{x}\mathbf{W}_{\text{up}}\, \text{diag}(\mathbf{g}(\mathbf{x}))\, \mathbf{W}_{\text{down}}. \tag{3}$$

The computation flows through three stages: linear up-projection to an intermediate space, input-dependent rescaling via the gate, and linear down-projection back to the model dimension. The feedforward block of transformers can be viewed as a reservoir containing memory and perform operation similar to retrieval mechanism [Geva et al., 2021]. The matrices $\mathbf{W}_{\text{up}}$ and $\mathbf{W}_{\text{down}}$ thus form a *coupled pair* mediated by a diagonal gate.

Value-output interaction. In the attention block, inputs are projected into query, key, and value subspaces. The attention mechanism computes a weighted combination of values, where the weights are determined by query-key interactions. Letting $\mathbf{A}$ denote the attention matrix, the output is:

$$\text{Attn}(\mathbf{X}) = \mathbf{A}(\mathbf{X}\mathbf{W}_V)\mathbf{W}_O, \tag{4}$$

where $\mathbf{X} \in \mathbb{R}^{n \times d}$ is the input sequence of length n. The value projection $\mathbf{W}_V$ maps inputs to an intermediate representation, which the output projection $\mathbf{W}_O$ then maps back to the model dimension. Crucially, the effective transformation applied to attended values is governed by the product $\mathbf{W}_V\mathbf{W}_O$, forming another coupled weight pair. According to [Elhage et al., 2021], this *OV circuit* determines "what to write" to the residual stream.

Query-key interaction. The attention matrix $\mathbf{A}$ arises from the interaction between queries and keys:

$$\mathbf{A} = \text{softmax}\left(\frac{\mathbf{X}\mathbf{W}_Q\mathbf{W}_K^\top\mathbf{X}^\top}{\sqrt{d}}\right). \tag{5}$$

The product $\mathbf{W}_Q\mathbf{W}_K^\top$ governs how queries attend to keys, suggesting another coupled pair. However, modern architectures employ rotary positional embeddings (RoPE) [Su et al., 2024], which insert position-dependent rotation matrices between $\mathbf{W}_Q$ and $\mathbf{W}_K$. Since these rotations vary with the relative positions of query and key tokens, the effective interaction matrix is no longer a simple product of the two weight matrices. In that case, we exclude the query-key pair and focus on the value-output and feed-forward interactions when performing adaptation. For architectures without position-dependent transformations such as vision transformers [Dosovitskiy et al., 2021], which typically use learnable or sinusoidal positional embeddings added to the input—the query-key interaction remains a simple bilinear form, and our parameterization can apply directly.

3.3 Adaptation for Weight Pairs

Having identified the query-key, value-output and feed-forward interactions as coupled weight pairs, we now develop a principled adaptation strategy. Those interactions can be abstracted as products of two linear transformations $\mathbf{W}_1\mathbf{W}_2$.

Consider adapting a matrix product $\mathbf{M} = \mathbf{W}_1\mathbf{W}_2$ with $\mathbf{W}_1 \in \mathbb{R}^{d \times k}, \mathbf{W}_2 \in \mathbb{R}^{k \times h}$. Under adaptation, we have:

$$(\mathbf{W}_1 + \Delta\mathbf{W}_1)(\mathbf{W}_2 + \Delta\mathbf{W}_2) = \mathbf{M} + \underbrace{\mathbf{W}_1\Delta\mathbf{W}_2 + \Delta\mathbf{W}_1\mathbf{W}_2}_{\text{first-order terms}} + \underbrace{\Delta\mathbf{W}_1\Delta\mathbf{W}_2}_{\text{second-order term}}. \tag{6}$$

In the fine-tuning regime, the adaptation matrices $\Delta\mathbf{W}_1 \in \mathbb{R}^{d \times k}$ and $\Delta\mathbf{W}_2 \in \mathbb{R}^{k \times h}$ are small relative to the pretrained weights, as we verify empirically in Figure 2a. Relative norm ratios alone, however, do not bound the second-order term. Under the parameterization introduced below

(Equation (9)), $\Delta\mathbf{W}_1 = \mathbf{B}_1\mathbf{U}_2^{(r)\top}$, we obtain a direct and dimension-free bound (derivation in Appendix E):

$$\frac{\|\Delta\mathbf{W}_1\Delta\mathbf{W}_2\|_2}{\|\Delta\mathbf{W}_1\mathbf{W}_2\|_2} \leq \frac{\|\Delta\mathbf{W}_2\|_2}{\sigma_r(\mathbf{W}_2)}, \tag{7}$$

where $\|\cdot\|_2$ is the spectral norm and $\sigma_r(\mathbf{W}_2)$ is the r-th singular value of $\mathbf{W}_2$. Hence the second-order term is negligible relative to the first-order term whenever the paired update is small compared with the retained singular scale $\sigma_r(\mathbf{W}_2)$, and we assume that the effective change in the interaction is governed by the first-order terms. The bound also shows why alignment matters: an unconstrained update may place energy in near-null directions of $\mathbf{W}_2$, for which no analogous lower bound on the first-order term holds.

This expansion reveals a key insight that effective adaptation requires coordination between the weight pairs. The term $\mathbf{W}_1\Delta\mathbf{W}_2$ depends on the structure of $\mathbf{W}_1$, while $\Delta\mathbf{W}_1\mathbf{W}_2$ depends on $\mathbf{W}_2$. We exploit this dependency by designing $\Delta\mathbf{W}_2$ to align with the subspace structure of $\mathbf{W}_1$, and $\Delta\mathbf{W}_1$ to align with that of $\mathbf{W}_2$.

3.4 Architecture-Aware Spectral Parameterization

We now specify our adaptation parameterization. Consider the first-order term $\Delta\mathbf{W}_1\mathbf{W}_2$. Let the singular value decomposition (SVD) of $\mathbf{W}_2$ be:

$$\mathbf{W}_2 = \mathbf{U}_2\mathbf{\Sigma}_2\mathbf{V}_2^\top, \tag{8}$$

where $\mathbf{U}_2 \in \mathbb{R}^{k\times k}$ contains the left singular vectors spanning the column space of $\mathbf{W}_2$, i.e., its input-side basis under the row-vector convention $\mathbf{x}\mathbf{W}_2$.

For the adaptation $\Delta\mathbf{W}_1$ to have maximal effect when composed with $\mathbf{W}_2$, it must operate on the subspace where $\mathbf{W}_2$ receives its inputs. We therefore parameterize:

$$\Delta\mathbf{W}_1 = \mathbf{B}_1\mathbf{A}_1, \quad \mathbf{A}_1 = \mathbf{U}_2^{(r)\top}, \tag{9}$$

where $\mathbf{U}_2^{(r)} \in \mathbb{R}^{k\times r}$ denotes the top-r left singular vectors of $\mathbf{W}_2$, and $\mathbf{B}_1 \in \mathbb{R}^{d\times r}$ is a learnable matrix. This ensures that $\Delta\mathbf{W}_1$ align with the subspace of $\mathbf{W}_2$. The matrix $\mathbf{B}_1$ is initialized at zero to preserve pretrained behavior.

Remark Unlike exisitng spectral-based PEFT approaches [Meng et al., 2024, Wang et al., 2025a, Xiong and Xie, 2026], which aligns $\Delta\mathbf{W}_1$ within its own principal subspace, PRISM aligns with the *paired* matrix $\mathbf{W}_2$ as a cross-matrix coordination. Additionally, the orthogonality of $\mathbf{U}_2^{(r)}$ improves optimization conditioning, similar to [Büyükakyüz, 2024].

We next justify this parameterization from subspace alignment perspectives.

3.5 Subspace Alignment Perspective

The spectral parameterization in Equation (9) can be understood through a simple geometric principle: when two matrices compose, only certain directions in the first matrix can influence the output. Consider the product $\mathbf{Z} = \mathbf{X}\mathbf{Y}$, where $\mathbf{Y} \in \mathbb{R}^{k\times h}$ is a fixed pretrained weight and $\mathbf{X} \in \mathbb{R}^{d\times k}$ is to be adapted. Let $\mathbf{Y} = \mathbf{U}_\mathbf{Y}\mathbf{\Sigma}_\mathbf{Y}\mathbf{V}_\mathbf{Y}^\top$ be the SVD, where $\mathbf{U}_\mathbf{Y} \in \mathbb{R}^{k\times k}$ spans the column space of $\mathbf{Y}$.

The inefficient parameter problem. Since $\mathbf{Y}\mathbf{v}$ always lies in $\text{col}(\mathbf{Y}) = \text{span}(\mathbf{U}_\mathbf{Y})$, any component of $\mathbf{X}$'s rows orthogonal to this subspace contributes nothing to the product $\mathbf{X}\mathbf{Y}$. Decomposing each row $\mathbf{x}_i = \mathbf{x}_i^{\text{align}} + \mathbf{x}_i^{\text{orth}}$ into components parallel and orthogonal to $\text{span}(\mathbf{U}_\mathbf{Y})$:

$$(\mathbf{X}\mathbf{Y}\mathbf{v})_i = \mathbf{x}_i^{\text{align}\top}(\mathbf{Y}\mathbf{v}) + \mathbf{x}_i^{\text{orth}\top}(\mathbf{Y}\mathbf{v}).$$

The orthogonal component $\mathbf{x}_i^{\text{orth}}$ represents wasted parameters invisible to the output because $\mathbf{x}_i^{\text{orth}\top}(\mathbf{Y}\mathbf{v}) = 0$. Figure 1b illustrates this geometry. Note that this exact invisibility holds for components orthogonal to the *full* column space $\text{col}(\mathbf{Y})$. Components that lie in $\text{col}(\mathbf{Y})$ but outside its principal subspace are not invisible; they are attenuated by the corresponding smaller singular values of $\mathbf{Y}$. We quantify this effect below.

Subspace alignment. To eliminate inefficient parameters, we restrict $\mathbf{X}$ to lie entirely within $\text{col}(\mathbf{Y})$:

$$\mathbf{X} = \tilde{\mathbf{X}}\mathbf{U}_\mathbf{Y}^\top, \tag{10}$$

where $\tilde{\mathbf{X}} \in \mathbb{R}^{d\times k}$ is trainable. This parameterization constructs each row of $\mathbf{X}$ as a linear combination of the columns of $\mathbf{U}_\mathbf{Y}$, ensuring perfect alignment with $\text{col}(\mathbf{Y})$.

Optimality of principal subspace alignment. We now ask which rank-r basis should be used for $\Delta\mathbf{W}_1$, analyzing the first-order contribution $\Delta\mathbf{W}_1\mathbf{W}_2$ in isolation. Since every row of $\Delta\mathbf{W}_1\mathbf{W}_2$ lies in $\mathrm{rowspace}(\mathbf{W}_2)$, we consider *realizable* targets $\mathbf{T} = \mathbf{X}\mathbf{W}_2$ with $\mathbf{X} \in \mathbb{R}^{d\times k}$. We study fixed-basis rank-r parameterizations

$$\Delta\mathbf{W}_1 = \tilde{\mathbf{A}}\mathbf{P}^\top, \qquad \tilde{\mathbf{A}} \in \mathbb{R}^{d\times r},\ \mathbf{P} \in \mathbb{R}^{k\times r},\ \mathbf{P}^\top\mathbf{P} = \mathbf{I}_r, \tag{11}$$

where $\mathbf{P}$ is a fixed orthonormal basis and $\tilde{\mathbf{A}}$ is trainable; PRISM corresponds to $\mathbf{P} = \mathbf{U}_2^{(r)}$. For a target $\mathbf{X}\mathbf{W}_2$, define the best achievable error

$$\mathcal{E}(\mathbf{P};\mathbf{X}) \ := \ \min_{\tilde{\mathbf{A}}\in\mathbb{R}^{d\times r}} \left\|\tilde{\mathbf{A}}\mathbf{P}^\top\mathbf{W}_2 - \mathbf{X}\mathbf{W}_2\right\|_F. \tag{12}$$

No fixed basis can be optimal for *every* target: a target concentrated on a singular direction of $\mathbf{W}_2$ that $\mathbf{P}$ discards is better served by a basis containing that direction. Because the basis must be chosen before the task is observed, the appropriate guarantees are worst-case and average-case optimality.

Theorem 3.1 (Task-agnostic optimality of principal subspace alignment)**.** *Let $\mathbf{W}_2 = \mathbf{U}_2\mathbf{\Sigma}_2\mathbf{V}_2^\top \in \mathbb{R}^{k\times h}$ have singular values $\sigma_1 \geq \sigma_2 \geq \cdots \geq 0$, with $\sigma_i := 0$ for $i > \min(k,h)$, and let $r < k$. Then the paired principal basis $\mathbf{P} = \mathbf{U}_2^{(r)}$ satisfies:*

(i) ***Minimax optimality.*** *$\min_{\mathbf{P}^\top\mathbf{P}=\mathbf{I}_r} \max_{\|\mathbf{X}\|_F\leq 1} \mathcal{E}(\mathbf{P};\mathbf{X}) = \sigma_{r+1}$, and the minimum is attained at $\mathbf{P} = \mathbf{U}_2^{(r)}$.*

(ii) ***Average-case optimality.*** *If $\mathbf{X}$ is random with $\mathbb{E}[\mathbf{X}^\top\mathbf{X}] = c\,\mathbf{I}_k$ for some $c > 0$, then $\min_{\mathbf{P}^\top\mathbf{P}=\mathbf{I}_r} \mathbb{E}\big[\mathcal{E}(\mathbf{P};\mathbf{X})^2\big] = c\sum_{i>r}\sigma_i^2$, attained at $\mathbf{P} = \mathbf{U}_2^{(r)}$. If $\sigma_r > \sigma_{r+1}$, the optimal captured subspace $\mathrm{rowspace}(\mathbf{P}^\top\mathbf{W}_2) = \mathrm{span}(\mathbf{V}_2^{(r)})$ is unique.*

(iii) ***Per-instance bound.*** *For every $\mathbf{X}$, $\mathcal{E}(\mathbf{U}_2^{(r)};\mathbf{X}) \leq \sigma_{r+1}\|\mathbf{X}\|_F$.*

Proof sketch. Let $\mathbf{\Pi}_\mathbf{P}$ be the orthogonal projector onto $\mathrm{rowspace}(\mathbf{P}^\top\mathbf{W}_2)$. Solving the inner least-squares problem gives $\mathcal{E}(\mathbf{P};\mathbf{X}) = \|\mathbf{X}\mathbf{W}_2(\mathbf{I} - \mathbf{\Pi}_\mathbf{P})\|_F$. The worst case over $\|\mathbf{X}\|_F \leq 1$ is therefore $\|\mathbf{W}_2(\mathbf{I} - \mathbf{\Pi}_\mathbf{P})\|_2$, and the isotropic expected squared error is $c\,\|\mathbf{W}_2(\mathbf{I} - \mathbf{\Pi}_\mathbf{P})\|_F^2$. Since $\mathbf{W}_2\mathbf{\Pi}_\mathbf{P}$ has rank at most r, the Eckart–Young–Mirsky theorem lower-bounds these quantities by σ_{r+1} and $c\sum_{i>r}\sigma_i^2$. Choosing $\mathbf{P} = \mathbf{U}_2^{(r)}$ gives $\mathbf{\Pi}_\mathbf{P} = \mathbf{V}_2^{(r)}\mathbf{V}_2^{(r)\top}$ and attains both bounds. The full proof is in Appendix D. □

Theorem 3.1 establishes that, *without task-specific information*, no fixed rank-r basis outperforms the principal singular vectors of the paired matrix, in either the worst case or the isotropic average case, and that the residual error of PRISM is controlled by the discarded spectrum $\sigma_{r+1}, \sigma_{r+2}, \ldots$ of $\mathbf{W}_2$. The theorem does not claim optimality for every task: a task whose update concentrates in the minor directions of $\mathbf{W}_2$ would be better served by another basis. To check whether this is the typical regime, we measure unconstrained LoRA updates, whose bases are not tied to $\mathbf{W}_2$.

3.6 Rank Reduction for Parameter Efficiency

The full-rank alignment in Equation (10) ensures perfect subspace alignment but requires $d \times k$ parameters. In practice, we restrict to the top-r singular vectors:

$$\mathbf{X} = \tilde{\mathbf{X}}\mathbf{U}_\mathbf{Y}^{(r)\top}, \tag{13}$$

where $\tilde{\mathbf{X}} \in \mathbb{R}^{d\times r}$, reducing the parameter count from $d \times k$ to $d \times r$.

Table 1: **PRISM parameterizations for transformer weight pairs.** Here $\mathbf{A}, \mathbf{B}$ are trainable parameters.

Pair	First matrix	Second matrix
$\mathbf{W}_Q\mathbf{W}_K^\top$	$\Delta\mathbf{W}_Q = \mathbf{V}_K^{(r)}\mathbf{A}$	$\Delta\mathbf{W}_K = \mathbf{V}_Q^{(r)}\mathbf{A}$
$\mathbf{W}_V\mathbf{W}_O$	$\Delta\mathbf{W}_V = \mathbf{B}(\mathbf{U}_O^{(r)})^\top$	$\Delta\mathbf{W}_O = \mathbf{V}_V^{(r)}\mathbf{A}$
$\mathbf{W}_{\text{down}}\mathbf{G}\mathbf{W}_{\text{up}}$	$\Delta\mathbf{W}_{\text{down}} = \mathbf{B}(\mathbf{U}_{\text{up}}^{(r)})^\top$	$\Delta\mathbf{W}_{\text{up}} = \mathbf{V}_{\text{down}}^{(r)}\mathbf{A}$

This rank reduction trades complete subspace coverage for efficiency while concentrating adaptation in the *principal directions*—precisely where $\mathbf{Y}$ has the strongest response. By Theorem 3.1, the price of this truncation is governed by the discarded singular values $\sigma_{r+1}(\mathbf{Y}), \sigma_{r+2}(\mathbf{Y}), \ldots$; the more the spectrum of $\mathbf{Y}$ decays beyond rank r (as we examine empirically in Section 4), the smaller the loss. Applying PRISM on all pairs in transformers, we create unified adaptation rules in Table 1.

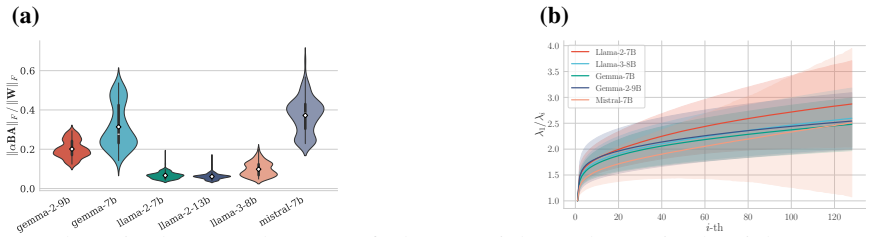

Figure 2: (a) The ratios between the norm of adapter weights and pretrained weights across different models. (b) The statistics of the condition number κ_i.

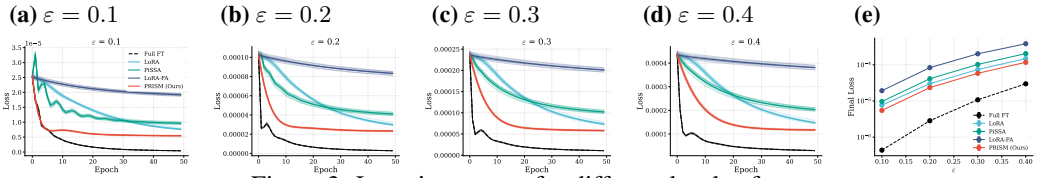

Figure 3: Learning curves for different levels of ε.

4 Experimental Results

We first conduct controlled experiments to validate the core principles of PRISM. We then evaluate PRISM across various domains including fine-tuning language models and vision transformers.

4.1 Toy model

Exploratory analysis of pretrained models. To motivate our toy experiment design, we first analyze the spectral properties of pretrained models. We fine-tune several models using LoRA ($\Delta\mathbf{W} = \mathbf{BA}$) and measure the ratio $\|\Delta\mathbf{W}\|_F/\|\mathbf{W}\|_F$ between adapted and pretrained weight magnitudes. Figure 2a shows these ratios predominantly lie in the range $[0.05, 0.4]$, indicating that fine-tuning introduces relatively small perturbations. We further examine the condition numbers $\kappa_i = \lambda_1/\lambda_i$, where λ_i denotes the i-th singular value of the weight matrix. Figure 2b reveals that singular values exhibit rapid exponential decay, with κ_{128} averaging between 2.5 and 3 across models. These observations help us design our controlled experiments.

Setup. We generate two matrices $\mathbf{W}_1, \mathbf{W}_2 \in \mathbb{R}^{d\times d}$ with controlled condition number κ_i. We then construct target matrices $\mathbf{W}_1^{\text{tg}}$ and $\mathbf{W}_2^{\text{tg}}$ such that $\|\mathbf{W}_i^{\text{tg}} - \mathbf{W}_i\|_F/\|\mathbf{W}_i\|_F = \varepsilon$ for $i = 1, 2$. The parameter ε controls the relative adaptation magnitude, mirroring the weight adaptation scale observed in pretrained models (Figure 2a). We aim to learn adaptations $\Delta\mathbf{W}_1$ and $\Delta\mathbf{W}_2$ such that the adapted matrices $\mathbf{W}_1 + \Delta\mathbf{W}_1$ and $\mathbf{W}_2 + \Delta\mathbf{W}_2$ approximate the targets well in terms of their product. Formally, we solve:

$$\min_{\Delta\mathbf{W}_1, \Delta\mathbf{W}_2} \|\mathbf{W}_1^{\text{tg}}\mathbf{W}_2^{\text{tg}} - (\mathbf{W}_1 + \Delta\mathbf{W}_1)(\mathbf{W}_2 + \Delta\mathbf{W}_2)\|_F.$$

This formulation replicates the interaction pair between $\mathbf{W}_1$ and $\mathbf{W}_2$ discussed in Section 3. We compare the parameterizations of $\Delta\mathbf{W}_1$ and $\Delta\mathbf{W}_2$ across different adaptation methods: full fine-tuning, LoRA [Hu et al., 2022], PiSSA [Meng et al., 2024], LoRA-FA [Zhang et al., 2023a], and our proposed PRISM. We provided another setting consider gated version in Appendix

Effect of adaptation magnitude ε. Figure 3 shows training curves for varying ε. PRISM consistently outperforms baselines, with the largest gains at small ε. This consistent with our assumption that

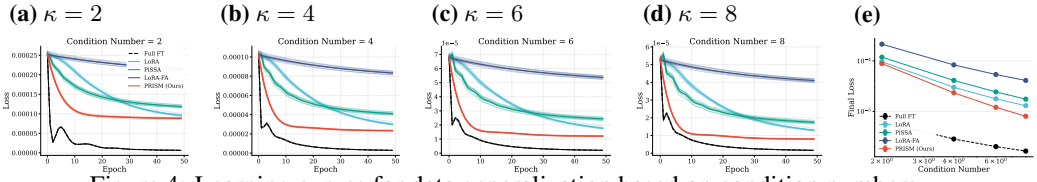

Figure 4: Learning curves for data generalization based on condition numbers κ.

Table 2: **Performance comparison (%) across arithmetic reasoning and code generation benchmarks.** Best results per model in **bold**. The last row reports the average across the six models. Full results with standard deviations are available in Table 5 and Table 6 (Appendix).

Model	GSM8K				MATH				HumanEval				MBPP			
	LoRA	DoRA	PiSSA	PRISM	LoRA	DoRA	PiSSA	PRISM	LoRA	DoRA	PiSSA	PRISM	LoRA	DoRA	PiSSA	PRISM
Mistral-7B	64.0	63.8	70.2	**71.4**	16.2	15.9	20.3	**21.9**	43.3	42.7	50.0	**51.6**	55.3	55.8	**62.9**	61.5
Llama-2-7B	42.6	44.3	50.6	**52.5**	6.0	6.1	7.0	**8.5**	21.3	20.7	**24.2**	23.0	37.6	37.6	38.2	**39.3**
Llama-2-13B	54.8	55.3	59.9	**62.2**	9.1	9.1	10.7	**12.3**	32.9	31.7	34.1	**34.3**	46.3	45.5	45.8	**48.7**
Llama-3-8B	72.3	72.8	75.3	**75.4**	23.2	23.6	24.3	**26.5**	53.7	53.7	54.1	**54.9**	67.7	67.7	66.1	**68.8**
Gemma-7B	68.0	67.2	73.5	**74.8**	22.0	21.4	28.7	**29.5**	38.4	45.1	52.6	**56.7**	54.5	55.8	64.9	**68.1**
Gemma-2-9B	77.8	77.6	80.4	**80.7**	37.6	38.1	**40.6**	40.1	57.3	59.1	**60.6**	60.4	67.2	69.7	**71.9**	**71.9**
Average	63.3	63.5	68.3	**69.5**	19.0	19.0	21.9	**23.1**	41.2	42.2	45.9	**46.8**	54.8	55.4	58.3	**59.7**

small adaptations yield more accurate first-order approximations in Equation (6). The performance gap diminishes as ε increases.

Effect of condition number κ. We vary the condition number κ when generating $\mathbf{W}_i$ to match the spectral properties observed in Figure 2b. Figure 4 shows the learning curves under different condition numbers. PRISM achieves the strongest performance when spectral decay is rapid (large κ), converging significantly faster.

We provide additional analysis on the effect of rank, principal versus random subspace initialization and the analysis for the gated interaction in Appendix F.1.

4.2 Arithmetic Reasoning

We conduct fine-tuning on six pretrained language models from three families: Mistral-7B Jiang et al. [2023], Llama Touvron et al. [2023], Grattafiori et al. [2024] (2-7B, 2-13B, and 3-8B), and Gemma Team et al. [2024] (7B and 2-9B). These pretrained models span from 7B to 13B parameters. Following standard practice, we fine-tune models on a 100K-sample subset of MetaMathQA [Yu et al.]. We evaluate arithmetic reasoning on two benchmarks: GSM8K [Cobbe et al., 2021], which tests multi-step word problem solving, and MATH [Hendrycks et al.], which requires competition-level mathematical reasoning.

Due to the presence of RoPE in the query-key pair, we adapt only the value-output ($\mathbf{W}_V$, $\mathbf{W}_O$) and feedforward weight pairs ($\mathbf{W}_{\text{up}}$, $\mathbf{W}_{\text{down}}$). Query and key weights remain frozen. We compare PRISM against three PEFT baselines including LoRA Hu et al. [2022], DoRA Liu et al. [2024] and PiSSA Meng et al. [2024].

Results. Table 2 shows PRISM achieves the best performance in 11 out of 12 settings. On GSM8K, PRISM outperforms the strongest baseline (PiSSA) by 1.2–3.7%. On MATH, we observe the similar gain, particularly, 1.5–2.2% on most models. This suggests spectral alignment is particularly beneficial for challenging reasoning tasks. These results validate our hypothesis that coordinating updates across paired weight structures enhances fine-tuning effectiveness. Training curves in Appendix G further show that PRISM converges faster than baselines, consistent with our toy model analysis.

4.3 Code generation

We use the same models and baselines as in Section 4. Models are fine-tuned on Code-Feedback [Zheng et al., 2024], a multi-turn code instruction dataset, and evaluated on two code generation benchmarks: HumanEval [Chen et al., 2021] and MBPP [Austin et al., 2021].

Results. Table 2 shows code generation results. PRISM achieves the best performance in 9 out of 12 settings. On HumanEval, PRISM outperforms baselines on 4 out of 6 models, with the largest gain on Gemma-7B with +4.1% gain over PiSSA. Our model remains competitive on stronger models (Gemma-2-9B, Llama-3-8B). On MBPP, PRISM achieves the best results on 4 out of 6 models, with gains of 1.1–3.2% over PiSSA. These results demonstrate that our spectral alignment benefits code generation tasks.

4.4 Commonsense Reasoning

We conduct an experiment on Commonsense170K Hu et al. [2023], a unified training set from eight commonsense reasoning benchmarks: BoolQ Clark et al. [2019], PIQA Bisk et al. [2020], SIQA Sap et al. [2019], HellaSwag Zellers et al. [2019], WinoGrande Sakaguchi et al. [2021], ARC-Easy/Challenge Clark et al. [2018], and OBQA Mihaylov et al. [2018]. These tasks test reasoning

Table 3: **Commonsense reasoning accuracy (%) across 8 benchmarks.** Best results per model in **bold**. The last row reports the average across the eight benchmarks. Full results with standard deviation are available in Table 9 (Appendix).

Benchmark	Mistral-7B				Llama-2-7B				Gemma-7B			
	LoRA	DoRA	PiSSA	PRISM	LoRA	DoRA	PiSSA	PRISM	LoRA	DoRA	PiSSA	PRISM
ARC-C	64.8	65.0	82.8	**83.7**	67.5	65.9	**70.6**	68.8	73.9	72.3	85.3	**86.9**
ARC-E	55.1	55.3	**93.3**	**93.3**	83.7	82.6	**86.0**	**86.0**	85.9	84.3	93.3	**95.0**
BoolQ	60.4	56.8	75.3	**76.4**	67.5	66.8	69.8	**70.2**	63.1	64.0	70.1	**75.6**
HellaSwag	88.2	84.9	94.1	**95.3**	77.5	72.6	**84.0**	82.3	84.9	80.6	94.5	**95.9**
OBQA	83.8	83.4	88.9	**90.9**	73.8	71.3	**78.7**	77.9	77.5	74.2	87.6	**90.8**
PIQA	78.6	77.7	91.0	**91.7**	81.1	79.7	**82.8**	82.4	83.7	81.4	89.8	**91.3**
SIQA	62.8	74.6	82.9	**83.1**	77.5	76.2	79.2	**80.0**	78.7	76.2	81.1	**83.5**
Wino.	79.8	73.1	88.9	**89.0**	77.9	75.1	**81.1**	80.6	84.9	82.7	90.2	**91.1**
Average	71.7	71.3	87.2	**87.9**	75.8	73.8	**79.0**	78.5	79.1	77.0	86.5	**88.8**

Table 4: **Vision transformer fine-tuning results on the VTAB-1k benchmark.** We report mean accuracy (%). Best results per dataset in **bold**. The rightmost **Avg.** column reports the mean across the 19 visual classification tasks. See Appendix for dataset abbreviations.

Model	Method	Natural							Specialized				Structured								Avg.
		C100	Cal.	DTD	Flw.	Pets	SVHN	Sun	PCam	Euro.	Res.	Retin.	Clv-C	Clv-D	DM	KIT-D	dSp-L	dSp-O	sN-A	sN-E	
Base	LoRA	48.36	90.30	63.51	94.08	83.92	91.18	36.38	84.83	94.09	83.25	73.60	**90.63**	60.00	54.00	78.20	48.00	30.28	**29.33**	37.03	66.89
	DoRA	36.93	87.87	60.21	86.96	77.81	91.02	27.49	78.59	92.50	76.40	73.60	83.56	60.13	30.03	75.67	46.37	28.71	6.23	32.20	60.65
	PiSSA	46.99	90.14	63.62	94.54	83.51	**91.22**	35.66	80.77	93.70	82.29	73.60	86.66	**61.53**	53.21	77.22	47.77	**30.64**	28.64	37.96	66.30
	PRISM	**53.67**	**92.83**	**68.56**	**96.96**	**86.56**	90.49	**37.73**	**88.73**	**94.78**	**85.57**	**76.97**	79.58	61.29	**57.12**	**81.29**	**48.62**	29.98	27.46	**41.10**	**68.38**
Large	LoRA	53.42	92.09	67.37	98.58	86.58	90.60	43.64	86.79	94.57	85.30	73.60	84.86	61.03	57.22	83.05	44.88	30.42	30.20	34.80	68.37
	DoRA	54.26	87.89	66.81	96.81	85.88	90.59	44.09	82.66	88.20	78.54	73.60	40.98	45.98	46.23	81.01	42.51	29.35	27.62	33.98	63.00
	PiSSA	52.87	91.93	67.13	97.88	87.69	**92.52**	43.52	86.53	94.87	85.58	73.60	55.79	57.52	57.29	81.86	45.02	30.17	**31.18**	34.53	66.92
	PRISM	**64.99**	**94.25**	**75.41**	**99.41**	**91.78**	91.97	**47.96**	**89.58**	**95.64**	**88.93**	**77.23**	**87.54**	**61.24**	**62.11**	**83.97**	**45.16**	**33.30**	28.48	**41.52**	**71.60**

capabilities including physical commonsense, social situations, and reading comprehension. We use pretrained models including Mistral-7B, Llama-2-7B, Gemma-7B for this task.

Results. Table 3 presents results across eight commonsense reasoning benchmarks on two models. PRISM achieves the best average performance on Gemma-7B, with competitive results on Llama-2-7B and Mistral-7B. On Gemma-7B, PRISM shows stronger gains with +2.3% average improvement over PiSSA, with substantial improvements on OBQA (+3.2 %), BoolQ (+5.5 %), and ARC-Easy (+1.7 %). On Llama-2-7B, PiSSA achieves slightly better average performance (79.0% vs 78.5%), though PRISM wins on 3 out of 8 individual benchmarks. We found that PRISM's advantages over baselines on benchmarks requiring physical and factual reasoning (OBQA, BoolQ) rather than natural language inference (HellaSwag, WinoGrande). This may imply that paired weight coordination can benefit tasks where the model must integrate multiple knowledge sources, as the value-output and feedforward pathways may operate to retrieve and combine relevant information Geva et al. [2021].

4.5 Adapting Vision Transformers

Setup. We evaluate PRISM on vision transformers using VTAB-1k [Zhai et al., 2019], a benchmark comprising 19 diverse visual classification tasks grouped into three categories: Natural (everyday objects), Specialized (medical imaging, satellite imagery, etc.), and Structured (tasks requiring geometric reasoning). Each task provides 1,000 training samples. We fine-tune two DINOv2 [Oquab et al., 2023] models: DINOv2-Base and DINOv2-Large. Unlike language models with RoPE, vision transformers use standard positional embeddings, allowing us to apply PRISM to all weight pairs including query-key interactions (Section 3).

Baselines. We compare against LoRA, DoRA, and PiSSA with rank $r = 16$. All methods adapt the query, key, value, output projections, and feedforward layers.

Results. Table 4 presents results across 19 visual classification tasks grouped into Natural, Specialized, and Structured categories. PRISM achieves the best mean accuracy on both DINOv2-Base (68.38%) and DINOv2-Large (71.60%), outperforming the strongest baseline by +2.08 % and +3.23% respectively. We found that PRISM's advantages amplify with model size, while PiSSA degrades

(66.92% → 66.30%). This suggests spectral alignment becomes increasingly important as model capacity grows, likely because larger models have more complex weight structures benefiting from explicit coordination.

4.6 Ablation studies

Additional studies on using PRISM as initialization for LoRA and on individual weight pair effects are provided in Appendix F.5.

Effect of rank. We evaluate Gemma-7B fine-tuned on MetaMathQA across ranks $r \in \{16, 32, 64, 128\}$ on both GSM8K and MATH benchmarks, comparing PRISM against LoRA and PiSSA baselines.

Results. Figure 5 shows performance across different rank configurations. PRISM maintains strong performance even at low ranks ($r = 16$), demonstrating robust parameter efficiency. Like PiSSA, PRISM exhibits graceful degradation as rank decreases, with accuracy remaining within 2-3 points of the highest-rank configuration. In contrast, LoRA is more sensitive to rank, requiring $r \geq 64$ to achieve competitive performance. We attribute this to adapting within the principal subspace of pretrained weights, an inductive bias that randomly initialized LoRA lacks.

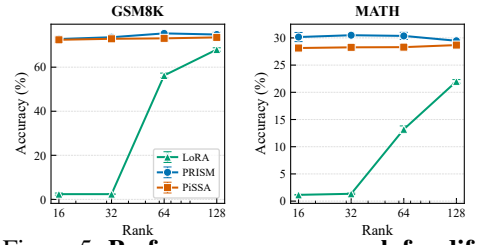

Figure 5: **Performance versus rank for different adaptation methods on Gemma-7B.** PRISMand PiSSA maintain strong performance at low ranks, while LoRA requires higher ranks for competitive results.

5 Conclusion

We introduced PRISM, a parameter-efficient fine-tuning method that respects structural coupling in transformer architectures. Unlike existing PEFT methods that treat weight matrices independently, PRISM coordinates adaptations across weight pairs by parameterizing updates in the spectral basis of their coupled counterparts. This ensures adaptations remain mutually visible across composed weight matrices. We established PRISM's theoretical foundations through subspace alignment that eliminates wasted parameters. Extensive experiments across arithmetic reasoning, code generation, commonsense reasoning, and visual classification demonstrate consistent improvements over existing PEFT methods. Our core insight is that effective adaptation requires *structural awareness* by aligning updates with how information flows through coupled weight matrices. This perspective opens new directions for principled PEFT methods that leverage pretrained weight geometry.

Limitations. PRISM requires computing SVDs of pretrained weight matrices at initialization, introducing a one-time overhead that scales with model size. More importantly, because PRISM adapts only structurally coupled weight pairs (e.g., query and key projections in RoPE-based models) are left frozen. Since our method is categorized as a spectral-based approaches, there is a sensitivity to the spectral structure of pretrained weights. If the singular value spectrum of pretrained models is flat, the principal subspace offers little advantage over random initialization.

Impact Statement. PRISM is a general fine-tuning technique with broad applicability to language and vision models. It reduces the compute and memory cost of adapting large pretrained models, which can lower barriers to deploying capable AI systems in resource-constrained settings. We do not foresee specific negative societal impacts beyond those associated with large pretrained models generally.

Acknowledgements

This work was supported by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grants funded by the Korea government (MSIT) (No. RS-2019-II190079, Artificial Intelligence Graduate School Program (Korea University); No. RS-2019-II190075, Artificial Intelligence Graduate School Program (KAIST), No. RS-2022-II220984, Development of Artificial Intelligence Technology for Personalized Plug-and-Play Explanation and Verification of Explanation; No. RS-2024-00457882, AI Research Hub Project; IITP-2026-RS-2023-00255968, Artificial Intelligence Convergence Innovation Human Resources Development), the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. RS-2025-23525509), and the High-Performance Computing Support Project, funded by the Government of the Republic of Korea (Ministry of Science and ICT) (No. RQT-25-070263).

References

A. Aghajanyan, S. Gupta, and L. Zettlemoyer. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. In *Proceedings of the 59th annual meeting of the association for computational linguistics and the 11th international joint conference on natural language processing (volume 1: long papers)*, pages 7319–7328, 2021.

J. Ainslie, J. Lee-Thorp, M. de Jong, Y. Zemlyanskiy, F. Lebron, and S. Sanghai. Gqa: Training generalized multi-query transformer models from multi-head checkpoints. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 4895–4901, 2023.

P. Albert, F. Z. Zhang, H. Saratchandran, C. Rodriguez-Opazo, A. van den Hengel, and E. Abbasnejad. Randlora: Full rank parameter-efficient fine-tuning of large models. In *The Thirteenth International Conference on Learning Representations*, 2025.

J. Austin, A. Odena, M. Nye, M. Bosma, H. Michalewski, D. Dohan, E. Jiang, C. Cai, M. Terry, Q. Le, et al. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

S. Babakniya, A. R. Elkordy, Y. H. Ezzeldin, Q. Liu, K.-B. Song, M. El-Khamy, and S. Avestimehr. Slora: Federated parameter efficient fine-tuning of language models. *arXiv preprint arXiv:2308.06522*, 2023.

K. Bałazy, M. Banaei, K. Aberer, and J. Tabor. Lora-xs: Low-rank adaptation with extremely small number of parameters. *arXiv preprint arXiv:2405.17604*, 2024.

C. Beattie, J. Z. Leibo, D. Teplyashin, T. Ward, M. Wainwright, H. Küttler, A. Lefrancq, S. Green, V. Valdés, A. Sadik, et al. Deepmind lab. *arXiv preprint arXiv:1612.03801*, 2016.

P. Bhattarai, M. Amjad, D. Zhylko, and T. Alhanai. Knowledge distillation through geometry-aware representational alignment. *arXiv preprint arXiv:2509.25253*, 2025.

Y. Bisk, R. Zellers, J. Gao, Y. Choi, et al. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 7432–7439, 2020.

T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. *NeurIPS*, 2020.

K. Büyükakyüz. OLoRA: Orthonormal low-rank adaptation of large language models. *arXiv preprint arXiv:2406.01775*, 2024.

D. Chang, P. Xue, Y. Li, Y. Liu, P. Xu, and S. Zhang. Calibrating and rotating: A unified framework for weight conditioning in peft. *arXiv preprint arXiv:2511.00051*, 2025.

M. Chen, J. Tworek, H. Jun, Q. Yuan, H. P. de Oliveira Pinto, J. Kaplan, H. Edwards, Y. Burda, N. Joseph, G. Brockman, A. Ray, R. Puri, G. Krueger, M. Petrov, H. Khlaaf, G. Sastry, P. Mishkin, B. Chan, S. Gray, N. Ryder, M. Pavlov, A. Power, L. Kaiser, M. Bavarian, C. Winter, P. Tillet, F. P. Such, D. Cummings, M. Plappert, F. Chantzis, E. Barnes, A. Herbert-Voss, W. H. Guss, A. Nichol, A. Paino, N. Tezak, J. Tang, I. Babuschkin, S. Balaji, S. Jain, W. Saunders, C. Hesse, A. N. Carr,

J. Leike, J. Achiam, V. Misra, E. Morikawa, A. Radford, M. Knight, M. Brundage, M. Murati, K. Mayer, P. Welinder, B. McGrew, D. Amodei, S. McCandlish, I. Sutskever, and W. Zaremba. Evaluating large language models trained on code, 2021. URL https://arxiv.org/abs/2107.03374.

Z. Chen, R. Dangovski, C. Loh, O. M. Dugan, D. Luo, and M. Soljacic. QuanTA: Efficient high-rank fine-tuning of LLMs with quantum-informed tensor adaptation. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=EfpZNpkrm2.

G. Cheng, J. Han, and X. Lu. Remote sensing image scene classification: Benchmark and state of the art. *Proc. IEEE*, 2017.

M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, and A. Vedaldi. Describing textures in the wild. In *CVPR*, 2014.

C. Clark, K. Lee, M.-W. Chang, T. Kwiatkowski, M. Collins, and K. Toutanova. Boolq: Exploring the surprising difficulty of natural yes/no questions. *arXiv preprint arXiv:1905.10044*, 2019.

P. Clark, I. Cowhey, O. Etzioni, T. Khot, A. Sabharwal, C. Schoenick, and O. Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

K. Cobbe, V. Kosaraju, M. Bavarian, M. Chen, H. Jun, L. Kaiser, M. Plappert, J. Tworek, J. Hilton, R. Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

T. Dettmers, A. Pagnoni, A. Holtzman, and L. Zettlemoyer. QLoRA: Efficient finetuning of quantized llms. *Advances in neural information processing systems*, 36:10088–10115, 2023.

A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=YicbFdNTTy.

N. Elhage, N. Nanda, C. Olsson, T. Henighan, N. Joseph, B. Mann, A. Askell, Y. Bai, A. Chen, T. Conerly, et al. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 1(1):12, 2021.

L. Fei-Fei, R. Fergus, and P. Perona. One-shot learning of object categories. *IEEE TPAMI*, 2006.

B. Fernando, A. Habrard, M. Sebban, and T. Tuytelaars. Subspace alignment for domain adaptation. *arXiv preprint arXiv:1409.5241*, 2014.

M. Gauch, M. Beck, T. Adler, D. Kotsur, S. Fiel, H. Eghbal-zadeh, J. Brandstetter, J. Kofler, M. Holzleitner, W. Zellinger, et al. Few-shot learning by dimensionality reduction in gradient space. In *Conference on Lifelong Learning Agents*, pages 1043–1064. PMLR, 2022.

A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: The kitti dataset. *IJRR*, 2013.

M. Geva, R. Schuster, J. Berant, and O. Levy. Transformer feed-forward layers are key-value memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5484–5495, 2021.

B. Graham. Kaggle diabetic retinopathy detection competition report. *University of Warwick*, 2015.

A. Grattafiori, A. Dubey, A. Jauhri, A. Pandey, A. Kadian, A. Al-Dahle, A. Letman, A. Mathur, A. Schelten, A. Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

G. Gur-Ari, D. A. Roberts, and E. Dyer. Gradient descent happens in a tiny subspace. *arXiv preprint arXiv:1812.04754*, 2018.

S. Hayou, N. Ghosh, and B. Yu. Lora+: Efficient low rank adaptation of large models. In *International Conference on Machine Learning*, pages 17783–17806. PMLR, 2024.

H. He, J. Li, X. Jiang, and H. Miller. SMT: Fine-tuning large language models with sparse matrices. In *International Conference on Learning Representations (ICLR)*, 2025.

P. Helber, B. Bischke, A. Dengel, and D. Borth. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE J-STARS*, 2019.

D. Hendrycks, C. Burns, S. Kadavath, A. Arora, S. Basart, E. Tang, D. Song, and J. Steinhardt. Measuring mathematical problem solving with the math dataset. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. De Laroussilhe, A. Gesmundo, M. Attariyan, and S. Gelly. Parameter-efficient transfer learning for nlp. In *International conference on machine learning*, pages 2790–2799. PMLR, 2019.

E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, W. Chen, et al. LoRA: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022.

Z. Hu, L. Wang, Y. Lan, W. Xu, E.-P. Lim, L. Bing, X. Xu, S. Poria, and R. Lee. Llm-adapters: An adapter family for parameter-efficient fine-tuning of large language models. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pages 5254–5276, 2023.

Q. Huang, T. Ko, Z. Zhuang, L. Tang, and Y. Zhang. HiRA: Parameter-efficient hadamard high-rank adaptation for large language models. In *The Thirteenth International Conference on Learning Representations*, 2025a. URL https://openreview.net/forum?id=TwJrTz9cRS.

Q. Huang, T. Ko, Z. Zhuang, L. Tang, and Y. Zhang. HiRA: Parameter-efficient hadamard high-rank adaptation for large language models. In *The Thirteenth International Conference on Learning Representations*, 2025b. URL https://openreview.net/forum?id=TwJrTz9cRS.

A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. de Las Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier, L. R. Lavaud, M.-A. Lachaux, P. Stock, T. L. Scao, T. Lavril, T. Wang, T. Lacroix, and W. E. Sayed. Mistral 7b. *ArXiv*, abs/2310.06825, 2023.

J. Johnson, B. Hariharan, L. Van Der Maaten, L. Fei-Fei, C. Lawrence Zitnick, and R. Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *CVPR*, 2017.

T. Koike-Akino, F. Tonin, Y. Wu, F. Z. Wu, L. N. Candogan, and V. Cevher. Quantum-PEFT: Ultra parameter-efficient fine-tuning. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=dgR6i4TSng.

D. J. Kopiczko, T. Blankevoort, and Y. M. Asano. VeRA: Vector-based random matrix adaptation. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=NjNfLdxr3A.

A. Krizhevsky, G. Hinton, et al. Learning multiple layers of features from tiny images. 2009.

W. Lai, A. Fraser, and I. Titov. Joint localization and activation editing for low-resource fine-tuning. *arXiv preprint arXiv:2502.01179*, 2025.

Y. LeCun, F. J. Huang, and L. Bottou. Learning methods for generic object recognition with invariance to pose and lighting. In *CVPR*, 2004.

B. Lester, R. Al-Rfou, and N. Constant. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, 2021.

X. L. Li and P. Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, 2021.

Y. Li, Y. Yu, C. Liang, P. He, N. Karampatziakis, W. Chen, and T. Zhao. Loftq: Lora-fine-tuning-aware quantization for large language models. *arXiv preprint arXiv:2310.08659*, 2023.

H. Liu, D. Tam, M. Muqeeth, J. Mohta, T. Huang, M. Bansal, and C. A. Raffel. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. *Advances in Neural Information Processing Systems*, 35:1950–1965, 2022.

S.-Y. Liu, C.-Y. Wang, H. Yin, P. Molchanov, Y.-C. F. Wang, K.-T. Cheng, and M.-H. Chen. DoRA: Weight-decomposed low-rank adaptation. In *Forty-first International Conference on Machine Learning*, 2024.

W. Liu, Z. Qiu, Y. Feng, Y. Xiu, Y. Xue, L. Yu, H. Feng, Z. Liu, J. Heo, S. Peng, et al. Parameter-efficient orthogonal finetuning via butterfly factorization. In *The Twelfth International Conference on Learning Representations*.

L. Matthey, I. Higgins, D. Hassabis, and A. Lerchner. dsprites: Disentanglement testing sprites dataset. https://github.com/deepmind/dsprites-dataset/, 2017.

F. Meng, Z. Wang, and M. Zhang. PiSSA: Principal singular values and singular vectors adaptation of large language models. *Advances in Neural Information Processing Systems*, 37:121038–121072, 2024.

T. Mihaylov, P. Clark, T. Khot, and A. Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. *arXiv preprint arXiv:1809.02789*, 2018.

Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, A. Y. Ng, et al. Reading digits in natural images with unsupervised feature learning. In *NeurIPS Workshop*, 2011.

A. Nguyen and A. Tong. Sama: Morpho adaptation via asymmetric expansion of kronecker product. *arXiv preprint arXiv:2611.04462*, 2026.

A. Nguyen, J. Choi, and A. Tong. Loco: Low-rank compositional rotation fine-tuning. *arXiv preprint arXiv:2605.15916*, 2026.

M. Nikdan, S. Tabesh, E. Crnčević, and D. Alistarh. Rosa: Accurate parameter-efficient fine-tuning via robust adaptation. *arXiv preprint arXiv:2401.04679*, 2024.

M.-E. Nilsback and A. Zisserman. Automated flower classification over a large number of classes. In *ICVGIP*, 2008.

M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

F. Paischer, L. Hauzenberger, T. Schmied, B. Alkin, M. P. Deisenroth, and S. Hochreiter. Parameter efficient fine-tuning via explained variance adaptation. *arXiv preprint arXiv:2410.07170*, 2024.

O. M. Parkhi, A. Vedaldi, A. Zisserman, and C. Jawahar. Cats and dogs. In *CVPR*, 2012.

W. Peebles and S. Xie. Scalable diffusion models with transformers. In *CVPR*, 2023.

Z. Qiu, W. Liu, H. Feng, Y. Xue, Y. Feng, Z. Liu, D. Zhang, A. Weller, and B. Schölkopf. Controlling text-to-image diffusion by orthogonal finetuning. *Advances in Neural Information Processing Systems*, 36:79320–79362, 2023.

G. Saha, I. Garg, and K. Roy. Gradient projection memory for continual learning. In *International Conference on Learning Representations*, 2021.

K. Sakaguchi, R. L. Bras, C. Bhagavatula, and Y. Choi. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021.

M. Sap, H. Rashkin, D. Chen, R. LeBras, and Y. Choi. Socialiqa: Commonsense reasoning about social interactions. *arXiv preprint arXiv:1904.09728*, 2019.

R. Singhal, K. Ponkshe, R. Vartak, and P. Vepakomma. Abba: Highly expressive hadamard product adaptation for large language models. *International Conference on Learning Representations (ICLR)*, 2026.

J. Su, M. Ahmed, Y. Lu, S. Pan, W. Bo, and Y. Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

G. Team, M. Riviere, S. Pathak, P. G. Sessa, C. Hardin, S. Bhupatiraju, L. Hussenot, T. Mesnard, B. Shahriari, A. Ramé, et al. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*, 2024.

C. Tian, Z. Shi, Z. Guo, L. Li, and C. Xu. Hydralora: An asymmetric lora architecture for efficient fine-tuning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

B. S. Veeling, J. Linmans, J. Winkens, T. Cohen, and M. Welling. Rotation equivariant cnns for digital pathology. In *MICCAI*, 2018.

H. Wang, Y. Li, S. Wang, G. Chen, and Y. Chen. Milora: Harnessing minor singular components for parameter-efficient llm finetuning. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 4823–4836, 2025a.

S. Wang, L. Yu, and J. Li. Lora-ga: Low-rank adaptation with gradient approximation. *Advances in Neural Information Processing Systems*, 37:54905–54931, 2024.

Y. Wang, F. Meng, X. Zhang, F. Jiang, P. Tang, and M. Zhang. HD-PiSSA: High-rank distributed orthogonal adaptation. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 6526–6539, 2025b.

Z. Wu, A. Arora, Z. Wang, A. Geiger, D. Jurafsky, C. D. Manning, and C. Potts. Reft: Representation finetuning for language models. *Advances in Neural Information Processing Systems*, 37:63908–63962, 2024.

J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *CVPR*, 2010.

J. Xiao, C. Luo, L. Huang, C. Yang, Y. Sui, H. Phan, X. Zang, Y. Ying, Z. Tang, A. Anandkumar, et al. Ecospa: Efficient transformer training with coupled sparsity. *arXiv preprint arXiv:2511.11641*, 2025.

Y. Xiong and X. Xie. OPLoRA: Orthogonal projection lora prevents catastrophic forgetting during parameter-efficient fine-tuning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 34088–34096, 2026.

F. Yin, X. Ye, and G. Durrett. LoFiT: Localized fine-tuning on llm representations. *Advances in Neural Information Processing Systems*, 37:9474–9506, 2024.

L. Yu, W. Jiang, H. Shi, J. YU, Z. Liu, Y. Zhang, J. Kwok, Z. Li, A. Weller, and W. Liu. Metamath: Bootstrap your own mathematical questions for large language models. In *The Twelfth International Conference on Learning Representations*.

R. Zellers, A. Holtzman, Y. Bisk, A. Farhadi, and Y. Choi. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*, 2019.

X. Zhai, J. Puigcerver, A. Kolesnikov, P. Ruyssen, C. Riquelme, M. Lucic, J. Djolonga, A. S. Pinto, M. Neumann, A. Dosovitskiy, et al. A large-scale study of representation learning with the visual task adaptation benchmark. *arXiv preprint arXiv:1910.04867*, 2019.

F. Zhang and M. Pilanci. Spectral adapter: Fine-tuning in spectral space. *Advances in Neural Information Processing Systems*, 37:130819–130847, 2024.

L. Zhang, L. Zhang, S. Shi, X. Chu, and B. Li. Lora-fa: Memory-efficient low-rank adaptation for large language models fine-tuning. *arXiv preprint arXiv:2308.03303*, 2023a.

Q. Zhang, M. Chen, A. Bukharin, N. Karampatziakis, P. He, Y. Cheng, W. Chen, and T. Zhao. Adalora: Adaptive budget allocation for parameter-efficient fine-tuning. *arXiv preprint arXiv:2303.10512*, 2023b.

Z. Zhang, B. Liu, and J. Shao. Fine-tuning happens in tiny subspaces: Exploring intrinsic task-specific subspaces of pre-trained language models. *arXiv preprint arXiv:2305.17446*, 2023c.

T. Zheng, G. Zhang, T. Shen, X. Liu, B. Y. Lin, J. Fu, W. Chen, and X. Yue. Opencodeinterpreter: Integrating code generation with execution and refinement. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 12834–12859, 2024.

B. Zi, X. Qi, L. Wang, J. Wang, K.-F. Wong, and L. Zhang. Delta-lora: Fine-tuning high-rank parameters with the delta of low-rank matrices. *arXiv preprint arXiv:2309.02411*, 2023.

A. Zou, L. Phan, S. Chen, J. Campbell, P. Guo, R. Ren, A. Pan, X. Yin, M. Mazeika, A.-K. Dombrowski, et al. Representation engineering: A top-down approach to AI transparency. *arXiv preprint arXiv:2310.01405*, 2023.

Table of Appendix Contents

A Extended Related Work **18**

B Further note on subspace alignment **18**

C Extension to Grouped Query Attention **19**

D Proof of Theorem 3.1 **21**

E Bounding the second-order term **22**

F Experimental results **22**

F.1 Toy model 22

F.2 Arithmetic Reasoning and Code Generation 24

F.3 Commonsense reasoning 25

F.4 Fine-tuning Vision Transformers 26

F.5 More ablation studies 27

G Training curve plots **27**

Notation. We denote matrices by bold uppercase letters (e.g., $\mathbf{W}$, $\mathbf{U}$) and vectors by bold lowercase letters (e.g., $\mathbf{x}$, $\mathbf{v}$). For a matrix $\mathbf{W} \in \mathbb{R}^{m\times n}$, its singular value decomposition is $\mathbf{W} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$, where $\mathbf{U}$ and $\mathbf{V}$ contain the left and right singular vectors, and $\mathbf{\Sigma}$ is diagonal with singular values $\sigma_1 \geq \sigma_2 \geq \cdots \geq 0$. We use $\mathbf{U}^{(r)}$ to denote the matrix of the top-r left singular vectors, and similarly for $\mathbf{V}^{(r)}$. The Frobenius norm is denoted $\|\cdot\|_F$. We write $\mathrm{col}(\mathbf{W})$ for the column space of $\mathbf{W}$ and $\mathrm{span}(\mathbf{U})$ for the subspace spanned by the columns of $\mathbf{U}$.

A Extended Related Work

In this section, we extend the related work discussing a broader scope of literature.

Weight Decomposition Methods DoRA [Liu et al., 2024] decomposes pretrained weights into magnitude and direction components, fine-tuning each separately. This decomposition has been shown to increase the singular value entropy of the update matrix, promoting more uniform update distributions similar to full fine-tuning. DoRA outperforms LoRA particularly when using lower ranks, though at the cost of additional computational overhead. Building on this insight, recent work [Chang et al., 2025] reformulates DoRA as a learnable weight conditioning method and proposes Pre-Diag and SORA (Skewed Orthogonal Rotation Adaptation) for more efficient parameter-efficient transformations. A complementary line of work seeks to escape the rank ceiling imposed by the bilinear BA form. HiRA [Huang et al., 2025b] constructs the update as a Hadamard product $\Delta W = W_0 \odot (BA)$ between the pretrained weight and a low-rank factor, yielding a high-rank effective update while keeping the trainable footprint comparable to LoRA. ABBA [Singhal et al., 2026] pushes this idea further by reparameterizing ΔW as the Hadamard product of *two* independently learned low-rank adapters, decoupling the high-rank update from the pretrained weight and improving expressivity at fixed budget.

Sparse and Structured Updates An orthogonal axis of efficiency restricts *which* parameters are updated rather than how the update is factorized. SMT [He et al., 2025] constrains weight updates to a sparse, block-diagonal structure, selecting blocks via gradient-based importance scores accumulated during a brief warm-up phase. This trades a degree of theoretical expressivity for substantial gains in computational and memory efficiency, and is conceptually orthogonal to spectral parameterizations: SMT chooses *where* to update, whereas spectral methods (including ours) choose *which directions* to update.

Other PEFT Approaches Beyond low-rank methods, several alternatives exist. Adapter layers [Houlsby et al., 2019] insert small bottleneck modules between transformer layers. Prefix tuning [Li and Liang, 2021] and prompt tuning [Lester et al., 2021] prepend trainable tokens to inputs. IA^3 [Liu et al., 2022] uses element-wise rescaling vectors rather than low-rank matrices, achieving even greater parameter efficiency through multiplicative scaling of activations.

B Further note on subspace alignment

Here, we provide an additional detail of the subspace alignment for our proposed PRISM.
When $\mathbf{Z}$ acts on a vector $\mathbf{v} \in \mathbb{R}^n$, the computation proceeds in stages. First, $\mathbf{Y}$ maps $\mathbf{v}$ to:

$$\mathbf{Y}\mathbf{v} = \mathbf{U}_\mathbf{Y}\mathbf{\Sigma}_\mathbf{Y}\mathbf{V}_\mathbf{Y}^\top\mathbf{v} = \mathbf{U}_\mathbf{Y}\boldsymbol{\alpha}, \tag{14}$$

where $\boldsymbol{\alpha} = \mathbf{\Sigma}_\mathbf{Y}\mathbf{V}_\mathbf{Y}^\top\mathbf{v} \in \mathbb{R}^k$. Crucially, $\mathbf{Y}\mathbf{v}$ always lies in $\mathrm{col}(\mathbf{Y}) = \mathrm{span}(\mathbf{U}_\mathbf{Y})$, regardless of the input $\mathbf{v}$.
Next, the i-th row $\mathbf{x}_i^\top$ of $\mathbf{X}$ processes this intermediate result:

$$(\mathbf{X}\mathbf{Y}\mathbf{v})_i = \mathbf{x}_i^\top(\mathbf{Y}\mathbf{v}) = \mathbf{x}_i^\top\mathbf{U}_\mathbf{Y}\boldsymbol{\alpha} = \langle \mathbf{x}_i, \mathbf{U}_\mathbf{Y}\boldsymbol{\alpha}\rangle. \tag{15}$$

Decomposing $\mathbf{x}_i = \mathbf{x}_i^{\text{align}} + \mathbf{x}_i^{\text{orth}}$ into components parallel and orthogonal to $\mathrm{span}(\mathbf{U}_\mathbf{Y})$, we find:

$$\mathbf{x}_i^\top(\mathbf{Y}\mathbf{v}) = {\mathbf{x}_i^{\text{align}}}^\top(\mathbf{Y}\mathbf{v}) + \underbrace{{\mathbf{x}_i^{\text{orth}}}^\top(\mathbf{Y}\mathbf{v})}_{=0} = {\mathbf{x}_i^{\text{align}}}^\top(\mathbf{Y}\mathbf{v}). \tag{16}$$

The orthogonal component $\mathbf{x}_i^{\text{orth}}$ is invisible to the output. It represents wasted parameters that cannot affect the composition $\mathbf{X}\mathbf{Y}$. Figure 1b illustrates this geometric interpretation.

Extension to gated feedforward networks. For gated FFNs, the first-order interaction becomes $\Delta\mathbf{X}\,\mathbf{D}\,\mathbf{Y}$, where $\mathbf{D} = \mathrm{diag}(\mathbf{g}(\mathbf{x}))$ is an input-dependent diagonal gate. For a fixed token, the effective partner of $\Delta\mathbf{X}$ is therefore $\mathbf{DY}$ rather than $\mathbf{Y}$, and its column space $\mathrm{col}(\mathbf{DY}) = \mathbf{D}\,\mathrm{col}(\mathbf{Y})$ generally differs from $\mathrm{col}(\mathbf{Y})$ when the gate is non-uniform, since left-multiplication by $\mathbf{D}$ rescales the rows of $\mathbf{Y}$ and thereby rotates its columns. Exact alignment would thus require a token-dependent basis, which is incompatible with a single fixed adapter. PRISM instead uses the gate-independent basis $\mathbf{U}_{\mathbf{Y}}^{(r)}$ of $\mathbf{Y}$ as a practical approximation. This choice is exact when the gate is uniform, $\mathbf{D} = g\mathbf{I}$, and remains accurate when the gate values vary little across the coordinates that dominate $\mathrm{col}(\mathbf{Y})$. We therefore treat the gated case as an approximation of the ungated analysis rather than as an exact extension, and we evaluate it empirically via the toy analysis with random gates in Appendix F.1.3 and through component-wise ablations on arithmetic reasoning in Appendix F.5.2.

C Extension to Grouped Query Attention

We would like to detail the implementation of PRISM in modern language models which adopt Grouped Query Attention (GQA) [Ainslie et al., 2023]. In GQA, multiple query heads share a single key-value head: with H query heads and G key-value groups (where H/G query heads share each group), the weight dimensions become asymmetric:

$$\mathbf{W}_V \in \mathbb{R}^{d\times Gd_h}, \tag{17}$$

$$\mathbf{W}_O \in \mathbb{R}^{Hd_h\times d}, \tag{18}$$

where d_h denotes the head dimension. This asymmetry requires careful treatment when computing spectral bases for paired adaptation.

The dimension mismatch problem. Our spectral parameterization requires aligning $\Delta\mathbf{W}_V$ with the column space of $\mathbf{W}_O$, and $\Delta\mathbf{W}_O$ with the row space of $\mathbf{W}_V$. However, $\mathbf{W}_O$ has input dimension Hd_h while $\mathbf{W}_V$ has output dimension Gd_h. Directly applying the standard parameterization is not possible due to this shape incompatibility.

Value projection adaptation. For $\Delta\mathbf{W}_V$, we require alignment with the column space of $\mathbf{W}_O$. We first compute the SVD:

$$\mathbf{W}_O = \mathbf{U}_O\mathbf{\Sigma}_O\mathbf{V}_O^\top, \tag{19}$$

where $\mathbf{U}_O \in \mathbb{R}^{Hd_h\times Hd_h}$ contains the left singular vectors. To bridge the dimension mismatch, we partition the principal left singular vectors $\mathbf{U}_O^{(r)} \in \mathbb{R}^{Hd_h\times r}$ by query groups:

$$\mathbf{U}_O^{(r)} = \begin{bmatrix} \mathbf{U}_O^{(1,r)} \\ \mathbf{U}_O^{(2,r)} \\ \vdots \\ \mathbf{U}_O^{(H/G,r)} \end{bmatrix}, \tag{20}$$

where each $\mathbf{U}_O^{(g,r)} \in \mathbb{R}^{Gd_h\times r}$ corresponds to the query heads within group g. We then average across query groups to obtain a compressed basis:

$$\bar{\mathbf{U}}_O^{(r)} = \frac{1}{H/G}\sum_{g=1}^{H/G}\mathbf{U}_O^{(g,r)} \in \mathbb{R}^{Gd_h\times r}. \tag{21}$$

This averaging aggregates the output directions across all query heads that share each key-value group, yielding a basis that matches $\mathbf{W}_V$'s output dimension. The value adaptation becomes:

$$\Delta\mathbf{W}_V = \mathbf{B}\,\bar{\mathbf{U}}_O^{(r)\top}, \tag{22}$$

where $\mathbf{B} \in \mathbb{R}^{d\times r}$ is the learnable matrix.

Output projection adaptation. Conversely, $\Delta\mathbf{W}_O$ requires alignment with the row space of $\mathbf{W}_V$. We compute the SVD:

$$\mathbf{W}_V = \mathbf{U}_V\mathbf{\Sigma}_V\mathbf{V}_V^\top, \tag{23}$$

and extract the right singular vectors $\mathbf{V}_V^{(r)} \in \mathbb{R}^{Gd_h \times r}$. Since $\mathbf{W}_O$ expects inputs from all H query heads but $\mathbf{W}_V$ only produces G groups of outputs, we replicate the basis across query groups:

$$\tilde{\mathbf{V}}_V^{(r)} = \begin{bmatrix} \mathbf{V}_V^{(r)} \\ \mathbf{V}_V^{(r)} \\ \vdots \\ \mathbf{V}_V^{(r)} \end{bmatrix} \in \mathbb{R}^{Hd_h \times r}, \tag{24}$$

where the replication occurs H/G times. This expansion reflects the GQA structure: query heads within the same group receive identical value representations, so their corresponding rows in $\Delta\mathbf{W}_O$ should operate on the same spectral basis. The output adaptation becomes:

$$\Delta\mathbf{W}_O = \tilde{\mathbf{V}}_V^{(r)}\mathbf{A}, \tag{25}$$

where $\mathbf{A} \in \mathbb{R}^{r \times d}$ is the learnable matrix.

D Proof of Theorem 3.1

Throughout, $\mathbf{W}_2 = \mathbf{U}_2\mathbf{\Sigma}_2\mathbf{V}_2^\top \in \mathbb{R}^{k\times h}$ with singular triplets $(\sigma_i, \mathbf{u}_i, \mathbf{v}_i)$, $\sigma_1 \geq \sigma_2 \geq \cdots \geq 0$, and $\sigma_i := 0$ for $i > \min(k, h)$. For a fixed orthonormal $\mathbf{P} \in \mathbb{R}^{k\times r}$, let $\mathbf{Q} := \mathbf{P}^\top\mathbf{W}_2 \in \mathbb{R}^{r\times h}$ and let $\mathbf{\Pi}_\mathbf{P} := \mathbf{Q}^+\mathbf{Q}$ be the orthogonal projector onto $\mathrm{rowspace}(\mathbf{Q})$, where $^+$ denotes the Moore–Penrose pseudoinverse.

Remark on realizability. For a general target $\mathbf{T} \in \mathbb{R}^{d\times h}$, write $\mathbf{\Pi}_{\mathbf{W}_2}$ for the projector onto $\mathrm{rowspace}(\mathbf{W}_2)$. Since $\mathrm{rowspace}(\mathbf{Q}) \subseteq \mathrm{rowspace}(\mathbf{W}_2)$, for every $\tilde{\mathbf{A}}$,

$$\|\tilde{\mathbf{A}}\mathbf{Q} - \mathbf{T}\|_F^2 = \|\tilde{\mathbf{A}}\mathbf{Q} - \mathbf{T}\mathbf{\Pi}_{\mathbf{W}_2}\|_F^2 + \|\mathbf{T}(\mathbf{I} - \mathbf{\Pi}_{\mathbf{W}_2})\|_F^2.$$

The second term cannot be reduced by any update $\Delta\mathbf{W}_1$, whatever its basis, and $\mathbf{T}\mathbf{\Pi}_{\mathbf{W}_2} = \mathbf{X}\mathbf{W}_2$ with $\mathbf{X} = \mathbf{T}\mathbf{W}_2^+$. Restricting to realizable targets $\mathbf{T} = \mathbf{X}\mathbf{W}_2$ therefore loses no generality for comparing bases.

Step 1: inner least-squares problem. The problem $\min_{\tilde{\mathbf{A}}} \|\tilde{\mathbf{A}}\mathbf{Q} - \mathbf{X}\mathbf{W}_2\|_F$ decouples over rows, and each row is a least-squares projection onto $\mathrm{rowspace}(\mathbf{Q})$. The minimizer is $\tilde{\mathbf{A}}^\star = \mathbf{X}\mathbf{W}_2\mathbf{Q}^+$, which gives $\tilde{\mathbf{A}}^\star\mathbf{Q} = \mathbf{X}\mathbf{W}_2\mathbf{\Pi}_\mathbf{P}$ and

$$\mathcal{E}(\mathbf{P}; \mathbf{X}) = \|\mathbf{X}\mathbf{M}_\mathbf{P}\|_F, \qquad \mathbf{M}_\mathbf{P} := \mathbf{W}_2(\mathbf{I} - \mathbf{\Pi}_\mathbf{P}). \tag{26}$$

No full-rank assumption on $\mathbf{Q}$ is needed.

Step 2: the principal basis. For $\mathbf{P} = \mathbf{U}_2^{(r)}$, $\mathbf{Q} = \mathbf{\Sigma}_2^{(r)}\mathbf{V}_2^{(r)\top}$. If $\sigma_r > 0$, then $\mathbf{\Pi}_\mathbf{P} = \mathbf{V}_2^{(r)}\mathbf{V}_2^{(r)\top}$ and

$$\mathbf{M}_{\mathbf{U}_2^{(r)}} = \sum_{i>r} \sigma_i \mathbf{u}_i \mathbf{v}_i^\top, \qquad \|\mathbf{M}_{\mathbf{U}_2^{(r)}}\|_2 = \sigma_{r+1}, \qquad \|\mathbf{M}_{\mathbf{U}_2^{(r)}}\|_F^2 = \sum_{i>r} \sigma_i^2. \tag{27}$$

If $\sigma_r = 0$, then $\mathrm{rank}(\mathbf{W}_2) < r$, $\mathrm{rowspace}(\mathbf{Q}) = \mathrm{rowspace}(\mathbf{W}_2)$, and $\mathbf{M}_{\mathbf{U}_2^{(r)}} = \mathbf{0}$, so (27) holds trivially with all quantities equal to zero.

Step 3: lower bounds for any basis. For every orthonormal $\mathbf{P}$, $\mathbf{W}_2\mathbf{\Pi}_\mathbf{P}$ has rank at most $\mathrm{rank}(\mathbf{\Pi}_\mathbf{P}) \leq r$. By the Eckart–Young–Mirsky theorem,

$$\|\mathbf{M}_\mathbf{P}\|_2 = \|\mathbf{W}_2 - \mathbf{W}_2\mathbf{\Pi}_\mathbf{P}\|_2 \geq \sigma_{r+1}, \qquad \|\mathbf{M}_\mathbf{P}\|_F^2 \geq \sum_{i>r} \sigma_i^2. \tag{28}$$

Proof of (i). For any matrix $\mathbf{M}$, $\max_{\|\mathbf{X}\|_F\leq 1} \|\mathbf{X}\mathbf{M}\|_F = \|\mathbf{M}\|_2$: the upper bound follows from $\|\mathbf{X}\mathbf{M}\|_F \leq \|\mathbf{X}\|_F\|\mathbf{M}\|_2$, and it is attained by $\mathbf{X} = \mathbf{e}_1\mathbf{a}^\top$, where $\mathbf{e}_1 \in \mathbb{R}^d$ is the first standard basis vector and $\mathbf{a} \in \mathbb{R}^k$ is a top left singular vector of $\mathbf{M}$. By (26), the worst-case error of basis $\mathbf{P}$ is $\|\mathbf{M}_\mathbf{P}\|_2 \geq \sigma_{r+1}$ by (28), with equality for $\mathbf{P} = \mathbf{U}_2^{(r)}$ by (27).

Proof of (ii). If $\mathbb{E}[\mathbf{X}^\top\mathbf{X}] = c\,\mathbf{I}_k$, then

$$\mathbb{E}\big[\mathcal{E}(\mathbf{P}; \mathbf{X})^2\big] = \mathbb{E}\,\mathrm{tr}\big(\mathbf{M}_\mathbf{P}^\top\mathbf{X}^\top\mathbf{X}\mathbf{M}_\mathbf{P}\big) = c\,\|\mathbf{M}_\mathbf{P}\|_F^2 \geq c\sum_{i>r}\sigma_i^2,$$

with equality for $\mathbf{P} = \mathbf{U}_2^{(r)}$. For uniqueness, assume $\sigma_r > \sigma_{r+1}$. Then the best rank-r Frobenius approximation of $\mathbf{W}_2$ is unique, so any optimal $\mathbf{P}$ satisfies $\mathbf{W}_2\mathbf{\Pi}_\mathbf{P} = \mathbf{U}_2^{(r)}\mathbf{\Sigma}_2^{(r)}\mathbf{V}_2^{(r)\top}$. Since $\sigma_r > 0$, the row space of the right-hand side is $\mathrm{span}(\mathbf{V}_2^{(r)})$, which has dimension r and is contained in $\mathrm{range}(\mathbf{\Pi}_\mathbf{P})$, itself of dimension at most r. Hence $\mathrm{rowspace}(\mathbf{P}^\top\mathbf{W}_2) = \mathrm{range}(\mathbf{\Pi}_\mathbf{P}) = \mathrm{span}(\mathbf{V}_2^{(r)})$ for every optimal $\mathbf{P}$. (The basis $\mathbf{P}$ itself is unique only up to rotation and up to components in the null space of $\mathbf{W}_2^\top$, which do not affect the composition.)

Proof of (iii). By (26) and (27), $\mathcal{E}(\mathbf{U}_2^{(r)}; \mathbf{X}) = \|\mathbf{X}\mathbf{M}_{\mathbf{U}_2^{(r)}}\|_F \leq \|\mathbf{X}\|_F\,\|\mathbf{M}_{\mathbf{U}_2^{(r)}}\|_2 = \sigma_{r+1}\|\mathbf{X}\|_F$. ■

Discussion. The theorem identifies the paired principal basis as the optimal task-agnostic choice; it does not assert that it is optimal for every individual target. For a specific $\mathbf{X}$, the error $\|\mathbf{X}\mathbf{M}_\mathbf{P}\|_F$ can be smaller for another basis, e.g., when the rows of $\mathbf{X}$ concentrate on $\mathbf{u}_i$ with $i > r$.

E Bounding the second-order term

We derive the bound in Equation (7). Let $\Delta \mathbf{W}_1 = \mathbf{B}_1 \mathbf{U}_2^{(r)\top}$ with $\mathbf{U}_2^{(r)\top}\mathbf{U}_2^{(r)} = \mathbf{I}_r$, and assume $\sigma_r(\mathbf{W}_2) > 0$. Because $\mathbf{U}_2^{(r)\top}$ has orthonormal rows, $\|\Delta \mathbf{W}_1\|_2 = \|\mathbf{B}_1\|_2$. Moreover,

$$\Delta \mathbf{W}_1 \mathbf{W}_2 = \mathbf{B}_1 \mathbf{U}_2^{(r)\top} \mathbf{U}_2 \mathbf{\Sigma}_2 \mathbf{V}_2^\top = \mathbf{B}_1 \mathbf{\Sigma}_2^{(r)} \mathbf{V}_2^{(r)\top},$$

so $\|\Delta \mathbf{W}_1 \mathbf{W}_2\|_2 = \|\mathbf{B}_1 \mathbf{\Sigma}_2^{(r)}\|_2$. Since $\|\mathbf{B}_1\|_2 = \|\mathbf{B}_1 \mathbf{\Sigma}_2^{(r)} (\mathbf{\Sigma}_2^{(r)})^{-1}\|_2 \le \|\mathbf{B}_1 \mathbf{\Sigma}_2^{(r)}\|_2 / \sigma_r(\mathbf{W}_2)$,

$$\|\Delta \mathbf{W}_1 \mathbf{W}_2\|_2 \;\ge\; \sigma_r(\mathbf{W}_2)\, \|\Delta \mathbf{W}_1\|_2.$$

Combining this with submultiplicativity, $\|\Delta \mathbf{W}_1 \Delta \mathbf{W}_2\|_2 \le \|\Delta \mathbf{W}_1\|_2 \|\Delta \mathbf{W}_2\|_2$, yields

$$\frac{\|\Delta \mathbf{W}_1 \Delta \mathbf{W}_2\|_2}{\|\Delta \mathbf{W}_1 \mathbf{W}_2\|_2} \;\le\; \frac{\|\Delta \mathbf{W}_2\|_2}{\sigma_r(\mathbf{W}_2)}.$$

The same argument applies symmetrically to the other first-order term: if $\Delta \mathbf{W}_2 = \mathbf{V}_1^{(r)} \mathbf{A}_2$ is aligned with the top-r right singular vectors of $\mathbf{W}_1$, then $\|\mathbf{W}_1 \Delta \mathbf{W}_2\|_2 \ge \sigma_r(\mathbf{W}_1) \|\Delta \mathbf{W}_2\|_2$ and $\|\Delta \mathbf{W}_1 \Delta \mathbf{W}_2\|_2 / \|\mathbf{W}_1 \Delta \mathbf{W}_2\|_2 \le \|\Delta \mathbf{W}_1\|_2 / \sigma_r(\mathbf{W}_1)$. The bound is dimension-free. For an unconstrained update $\Delta \mathbf{W}_1 = \mathbf{B}\mathbf{A}$, no such lower bound on the first-order term exists: if the rows of $\mathbf{A}$ lie near the null space of $\mathbf{W}_2^\top$, then $\|\Delta \mathbf{W}_1 \mathbf{W}_2\|_2$ can be arbitrarily small while $\|\Delta \mathbf{W}_1\|_2$ is not.

F Experimental results

All experiments are conducted on NVIDIA H200 GPUs using PyTorch. We compute the SVD via `torch.linalg.svd`. We observe that the SVD precomputation adds negligible overhead relative to total training time.

F.1 Toy model

We use the Adam optimizer with learning rate 10^{-3}. All experiments run for 50 epochs. We report results averaged over 10 independent simulations with different random seeds.

We generate square matrices $\mathbf{W}_1, \mathbf{W}_2 \in \mathbb{R}^{d \times d}$ with $d = 1000$. Singular values are constructed with exponential decay: singular values decay from $\lambda_1 = 10$ to $\lambda_{32} = \lambda_1 / \kappa$, where κ controls the condition number. Remaining singular values are set to a small constant (0.1).

Target matrices $\mathbf{W}_1^{\text{tg}}$ and $\mathbf{W}_2^{\text{tg}}$ are generated by adding Gaussian noise scaled such that $\|\mathbf{W}_i^{\text{tg}} - \mathbf{W}_i\|_F / \|\mathbf{W}_i\|_F = \varepsilon$.

Unless otherwise specified, we use adaptation rank $r = 16$, condition number $\kappa = 4$ (corresponding to $\kappa_{32} \approx 4$), and perturbation magnitude $\varepsilon = 0.1$.

We detail the ablation ranges in the main text here:

- Adaptation magnitude (Figure 3): $\varepsilon \in \{0.1, 0.2, 0.3, 0.4\}$
- Rank (Figure 6): $r \in \{8, 16, 32, 64, 128\}$
- Condition number (Figure 4): $\kappa \in \{2, 4, 8, 16\}$

F.1.1 Effect of rank

We ablate the effect of rank on performance of PRISM. Figure 6 shows the learning curves for different ranks. Higher ranks improve training performance, approaching full-rank PRISM as the rank increases.

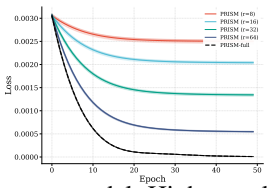

Figure 6: Effect of rank in the toy model. Higher ranks improve the performance.

F.1.2 Principal subspace vs. random subspace

Our parameterization $\tilde{\mathbf{X}}\mathbf{U}_{\mathbf{Y}}^{(r)\top}$ uses the principal subspace of $\mathbf{Y}$. We test whether this choice matters by replacing $\mathbf{U}_{\mathbf{Y}}^{(r)}$ with random matrices. Figure 7 shows that PRISM with principal subspaces consistently outperforms variants using random subspaces across multiple random initializations.

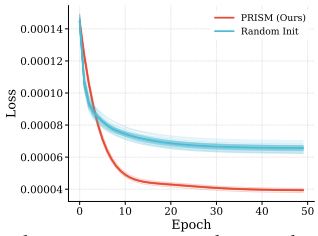

Figure 7: Comparison of principal subspace versus random orthonormal subspaces. PRISM with the principal subspace (orange) consistently outperforms random subspace baselines (blue).

F.1.3 Toy model on gated interactions

To verify that our parameterization extends to gated MLPs, we adapt the toy experiment from Section 4.1 to include a random diagonal gate. Specifically, we optimize

$$\min_{\Delta\mathbf{W}_1,\Delta\mathbf{W}_2} \mathbb{E}_{\mathbf{D}}\|\mathbf{W}_1^{\text{tg}}\mathbf{D}\mathbf{W}_2^{\text{tg}} - (\mathbf{W}_1+\Delta\mathbf{W}_1)\mathbf{D}(\mathbf{W}_2+\Delta\mathbf{W}_2)\|_F, \tag{29}$$

where $\mathbf{D} = \text{diag}(d_1,\ldots,d_n)$ with $d_i \sim \mathcal{N}(0,\sigma^2)$. The expectation uses Monte Carlo estimation with 10 sample. We consider two gate variance settings, $\sigma^2 \in \{0.5, 1.5\}$, to assess sensitivity to gate magnitude.

Figures 8–11 report learning curves under both ε and κ ablations for each setting. The results mirror those of the ungated case: PRISM consistently achieves lower error and faster convergence across all configurations, confirming that diagonal gating does not disrupt the subspace alignment principle underlying our method.

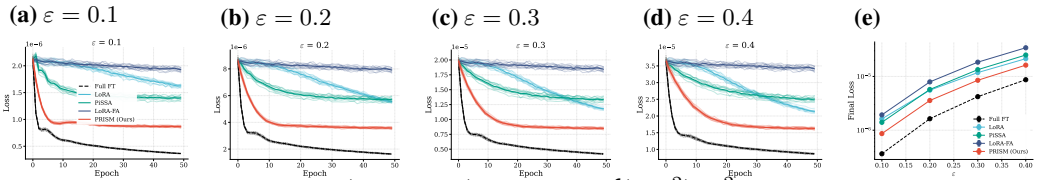

Figure 8: The case that $\mathbf{D} = \text{diag}(d_1,\ldots,d_n)$ with $d_i \sim \mathcal{N}(0,\sigma^2)$, $\sigma^2 = 0.5$. (a-d) Learning curves for different levels of ε. (e) Final loss.

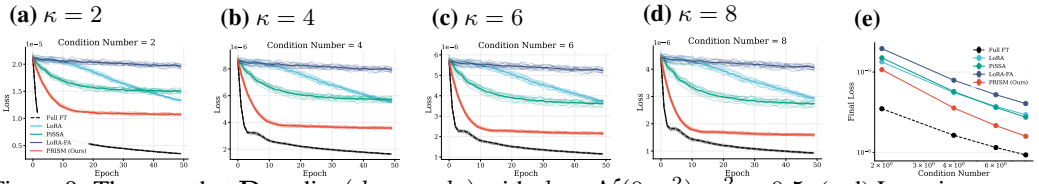

Figure 9: The case that $\mathbf{D} = \text{diag}(d_1,\ldots,d_n)$ with $d_i \sim \mathcal{N}(0,\sigma^2)$, $\sigma^2 = 0.5$. (a-d) Learning curves for data generalization based on condition numbers κ. (e) Final loss

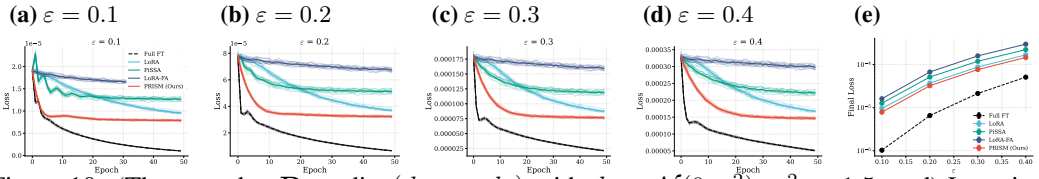

Figure 10: (The case that $\mathbf{D} = \text{diag}(d_1,\ldots,d_n)$ with $d_i \sim \mathcal{N}(0,\sigma^2)$, $\sigma^2 = 1.5$. a-d) Learning curves for different levels of ε. (e) Final loss.

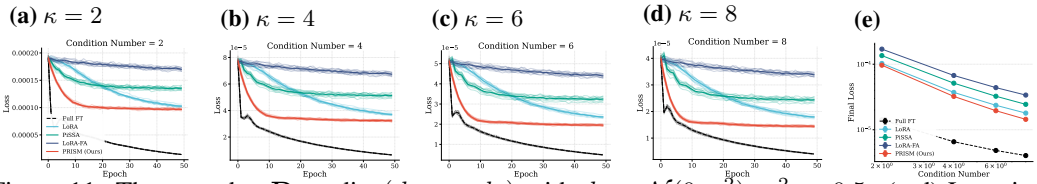

Figure 11: The case that $\mathbf{D} = \text{diag}(d_1, \ldots, d_n)$ with $d_i \sim \mathcal{N}(0, \sigma^2)$, $\sigma^2 = 0.5$. (a-d) Learning curves for data generalization based on condition numbers κ. (e) Final loss

F.2 Arithmetic Reasoning and Code Generation

Tables 5 and 6 present comprehensive results comparing our PRISM method against baseline approaches.

Table 5: Arithmetic reasoning accuracy (%) on GSM8K and MATH. Results show mean$_{\pm\text{std}}$ over 3 runs. Best results per model in **bold**.

	Mistral-7B	Llama-2-7B	Llama-2-13B	Llama-3-8B	Gemma-2-7B	Gemma-2-9B
GSM8K						
Full Fine-tuning	70.6$_{\pm 0.3}$	49.9$_{\pm 0.1}$	60.0$_{\pm 0.4}$	74.5$_{\pm 0.2}$	71.9$_{\pm 0.5}$	80.1$_{\pm 0.3}$
LoRA	64.0$_{\pm 0.8}$	42.6$_{\pm 1.2}$	54.8$_{\pm 0.6}$	72.3$_{\pm 0.5}$	68.0$_{\pm 0.8}$	77.8$_{\pm 0.4}$
DoRA	63.8$_{\pm 0.5}$	44.3$_{\pm 0.1}$	55.3$_{\pm 0.3}$	72.8$_{\pm 0.5}$	67.2$_{\pm 1.0}$	77.6$_{\pm 0.2}$
PiSSA	70.2$_{\pm 0.6}$	50.6$_{\pm 0.3}$	59.9$_{\pm 0.2}$	75.3$_{\pm 0.2}$	73.5$_{\pm 0.9}$	80.4$_{\pm 0.6}$
PRISM (Ours)	**71.4**$_{\pm 0.8}$	**52.5**$_{\pm 0.7}$	**62.2**$_{\pm 0.9}$	**75.4**$_{\pm 0.2}$	**74.8**$_{\pm 1.0}$	**80.7**$_{\pm 0.4}$
MATH						
Full Fine-tuning	19.6$_{\pm 0.2}$	7.5$_{\pm 0.3}$	10.3$_{\pm 0.4}$	28.1$_{\pm 0.5}$	28.4$_{\pm 0.2}$	38.1$_{\pm 0.3}$
LoRA	16.2$_{\pm 0.4}$	6.0$_{\pm 0.4}$	9.1$_{\pm 0.2}$	23.2$_{\pm 0.3}$	22.0$_{\pm 0.3}$	37.6$_{\pm 0.5}$
DoRA	15.9$_{\pm 0.2}$	6.1$_{\pm 0.2}$	9.1$_{\pm 0.3}$	23.6$_{\pm 0.2}$	21.4$_{\pm 0.1}$	38.1$_{\pm 0.2}$
PiSSA	20.3$_{\pm 0.3}$	7.0$_{\pm 0.2}$	10.7$_{\pm 0.5}$	24.3$_{\pm 0.0}$	28.7$_{\pm 0.3}$	**40.6**$_{\pm 0.1}$
PRISM (Ours)	**21.9**$_{\pm 0.4}$	**8.5**$_{\pm 0.4}$	**12.3**$_{\pm 0.3}$	**26.5**$_{\pm 0.5}$	**29.5**$_{\pm 0.1}$	40.1$_{\pm 0.6}$

Table 6: Code generation pass@1 (%) on HumanEval and MBPP. Results show mean$_{\pm\text{std}}$ over 3 runs. Best results per model in **bold**.

	Mistral-7B	Llama-2-7B	Llama-2-13B	Llama-3-8B	Gemma-7B	Gemma-2-9B
HumanEval						
LoRA	43.3$_{\pm 2.2}$	21.3$_{\pm 1.0}$	32.9$_{\pm 0.8}$	53.7$_{\pm 1.2}$	38.4$_{\pm 1.9}$	57.3$_{\pm 0.8}$
DoRA	42.7$_{\pm 2.1}$	20.7$_{\pm 0.9}$	31.7$_{\pm 0.6}$	53.7$_{\pm 0.8}$	45.1$_{\pm 1.2}$	59.1$_{\pm 0.9}$
PiSSA	50.0$_{\pm 3.7}$	**24.2**$_{\pm 0.7}$	34.1$_{\pm 1.1}$	54.1$_{\pm 1.1}$	52.6$_{\pm 2.1}$	**60.6**$_{\pm 1.3}$
PRISM (Ours)	**51.6**$_{\pm 3.7}$	23.0$_{\pm 1.3}$	**34.3**$_{\pm 0.6}$	**54.9**$_{\pm 0.9}$	**56.7**$_{\pm 2.4}$	60.4$_{\pm 2.2}$
MBPP						
LoRA	55.3$_{\pm 1.3}$	37.6$_{\pm 0.6}$	46.3$_{\pm 0.6}$	67.7$_{\pm 1.0}$	54.5$_{\pm 1.1}$	67.2$_{\pm 0.5}$
DoRA	55.8$_{\pm 1.1}$	37.6$_{\pm 0.4}$	45.5$_{\pm 0.5}$	67.7$_{\pm 0.9}$	55.8$_{\pm 1.0}$	69.7$_{\pm 0.8}$
PiSSA	**62.9**$_{\pm 1.5}$	38.2$_{\pm 1.4}$	45.8$_{\pm 0.8}$	66.1$_{\pm 1.3}$	64.9$_{\pm 1.3}$	**71.9**$_{\pm 1.0}$
PRISM (Ours)	61.5$_{\pm 1.5}$	**39.3**$_{\pm 0.8}$	**48.7**$_{\pm 0.7}$	**68.8**$_{\pm 1.2}$	**68.1**$_{\pm 1.1}$	**71.9**$_{\pm 0.9}$

Table 7 reports the number of trainable parameters for PRISM compared to LoRA and PiSSA.

Table 7: Parameter efficiency comparison between LoRA and PRISM with rank $r = 128$. PRISM consistently uses fewer trainable parameters than LoRA while achieving better performance.

Model	Total Params	LoRA/PiSSA		PRISM	
		Params	Ratio (%)	Params	Ratio (%)
Mistral-7B	7.24B	205.5M	2.84	151.0M	2.09
Llama-2-7B	6.74B	190.8M	2.83	123.7M	1.84
Llama-2-13B	13.02B	299.9M	2.30	193.9M	1.49
Llama-3-8B	8.03B	205.5M	2.56	151.0M	1.88
Gemma-7B	8.54B	249.6M	2.92	198.2M	2.32
Gemma-2-9B	9.24B	264.2M	2.86	193M	2.08

F.2.1 Hyperparameters

Tables 8 and 10 detail the hyperparameter configurations used in our experiments.

Table 8: Training hyperparameters for arithmetic reasoning and code generation experiments.

Hyperparameter	Value
Optimizer	AdamW
Learning rate	3×10^{-5}
LR scheduler	Cosine
Warmup ratio	0.02
Epochs	1
Effective batch size	32
Max sequence length	512
Precision	bfloat16
Rank r	128
Adapted modules	$\mathbf{W}_V, \mathbf{W}_O, \mathbf{W}_{\text{up}}, \mathbf{W}_{\text{down}}$

F.3 Commonsense reasoning

Table 9 presents the detailed report of the performance on commonsense reasoning benchmark. We also provide all the hyperparameter settings in Table 10.

Table 9: Commonsense reasoning accuracy (%) across 8 benchmarks. Results show mean ± standard deviation over 3 runs. Best results per model in **bold**.

Method	ARC-C	ARC-E	BoolQ	HellaSwag	OBQA	PIQA	SIQA	Wino.	Avg.
Mistral-7B									
LoRA	$64.8_{\pm 1.5}$	$55.1_{\pm 0.7}$	$60.4_{\pm 2.2}$	$88.2_{\pm 1.9}$	$83.8_{\pm 1.1}$	$78.6_{\pm 1.4}$	$62.8_{\pm 0.7}$	$79.8_{\pm 1.2}$	$71.7_{\pm 1.1}$
DoRA	$65.0_{\pm 1.4}$	$55.3_{\pm 1.0}$	$56.8_{\pm 1.6}$	$84.9_{\pm 0.0}$	$83.4_{\pm 1.2}$	$77.7_{\pm 0.7}$	$74.6_{\pm 0.2}$	$73.1_{\pm 0.1}$	$71.4_{\pm 0.9}$
PiSSA	$82.8_{\pm 0.1}$	$\mathbf{93.3}_{\pm 0.4}$	$75.3_{\pm 0.9}$	$94.1_{\pm 0.6}$	$88.9_{\pm 1.9}$	$91.0_{\pm 0.3}$	$82.9_{\pm 0.7}$	$88.9_{\pm 0.5}$	$87.2_{\pm 0.6}$
PRISM(Ours)	$\mathbf{83.7}_{\pm 0.7}$	$\mathbf{93.3}_{\pm 0.3}$	$\mathbf{76.4}_{\pm 0.5}$	$\mathbf{95.3}_{\pm 0.3}$	$\mathbf{90.9}_{\pm 1.3}$	$\mathbf{91.7}_{\pm 0.2}$	$\mathbf{83.1}_{\pm 0.5}$	$\mathbf{89.0}_{\pm 0.8}$	$\mathbf{87.9}_{\pm 0.4}$
Llama-2-7B									
LoRA	$67.5_{\pm 1.4}$	$83.7_{\pm 0.4}$	$67.5_{\pm 1.7}$	$77.5_{\pm 3.2}$	$73.8_{\pm 2.0}$	$81.1_{\pm 0.9}$	$77.5_{\pm 1.2}$	$77.9_{\pm 1.7}$	$75.8_{\pm 1.6}$
DoRA	$65.9_{\pm 3.3}$	$82.6_{\pm 2.6}$	$66.8_{\pm 2.7}$	$72.6_{\pm 10.2}$	$71.3_{\pm 5.5}$	$79.7_{\pm 2.8}$	$76.2_{\pm 3.5}$	$75.1_{\pm 5.1}$	$73.8_{\pm 4.5}$
PiSSA	$\mathbf{70.6}_{\pm 2.0}$	$\mathbf{86.0}_{\pm 0.9}$	$69.8_{\pm 1.4}$	$\mathbf{84.0}_{\pm 3.1}$	$\mathbf{78.7}_{\pm 4.3}$	$\mathbf{82.8}_{\pm 0.3}$	$79.2_{\pm 1.4}$	$\mathbf{81.1}_{\pm 2.1}$	$\mathbf{79.0}_{\pm 1.9}$
PRISM(Ours)	$68.8_{\pm 2.2}$	$\mathbf{86.0}_{\pm 1.5}$	$\mathbf{70.2}_{\pm 1.9}$	$82.3_{\pm 3.9}$	$77.9_{\pm 3.2}$	$82.4_{\pm 1.0}$	$\mathbf{80.0}_{\pm 1.6}$	$80.6_{\pm 2.8}$	$78.5_{\pm 2.3}$
Gemma-7B									
LoRA	$73.9_{\pm 4.9}$	$85.9_{\pm 3.6}$	$63.1_{\pm 6.6}$	$84.9_{\pm 6.9}$	$77.5_{\pm 5.2}$	$83.7_{\pm 0.7}$	$78.7_{\pm 2.8}$	$84.9_{\pm 2.9}$	$79.1_{\pm 4.2}$
DoRA	$72.3_{\pm 0.9}$	$84.3_{\pm 1.1}$	$64.0_{\pm 0.8}$	$80.6_{\pm 5.1}$	$74.2_{\pm 2.3}$	$81.4_{\pm 3.3}$	$76.2_{\pm 1.3}$	$82.7_{\pm 3.1}$	$77.0_{\pm 2.0}$
PiSSA	$85.3_{\pm 1.0}$	$93.3_{\pm 0.8}$	$70.1_{\pm 1.4}$	$94.5_{\pm 0.8}$	$87.6_{\pm 2.5}$	$89.8_{\pm 0.9}$	$81.1_{\pm 1.3}$	$90.2_{\pm 0.6}$	$86.5_{\pm 0.6}$
PRISM(Ours)	$\mathbf{86.9}_{\pm 1.0}$	$\mathbf{95.0}_{\pm 0.6}$	$\mathbf{75.6}_{\pm 0.9}$	$\mathbf{95.9}_{\pm 0.2}$	$\mathbf{90.8}_{\pm 0.0}$	$\mathbf{91.3}_{\pm 0.9}$	$\mathbf{83.5}_{\pm 0.7}$	$\mathbf{91.1}_{\pm 0.6}$	$\mathbf{88.8}_{\pm 0.4}$

Table 10: Training hyperparameters for commonsense reasoning experiments.

Hyperparameter	Value
Optimizer	AdamW
Learning rate	2×10^{-5}
LR scheduler	Cosine
Warmup ratio	0.02
Epochs	1
Effective batch size	32
Max sequence length	512
Precision	bfloat16
ank r	128
Adapted modules	$\mathbf{W}_V, \mathbf{W}_O, \mathbf{W}_{\text{up}}, \mathbf{W}_{\text{down}}$

F.4 Fine-tuning Vision Transformers

We peform hyperparameter search for all settings where learning rate: $\{1e-4, 5e-4, 1e-3\}$ and report the best performance on test set.

F.4.1 VTAB-1k Dataset

Table 11: Statistics of VTAB-1k benchmark datasets.

Dataset	Description	Classes	Train size	Test size
CIFAR-100 Krizhevsky et al. [2009]	Natural	100	1,000	10,000
Caltech101 Fei-Fei et al. [2006]		102		6,084
DTD Cimpoi et al. [2014]		47		1,880
Flowers102 Nilsback and Zisserman [2008]		102		6,149
Pets Parkhi et al. [2012]		37		3,669
SVHN Netzer et al. [2011]		10		26,032
Sun397 Xiao et al. [2010]		397		21,750
Patch Camelyon Veeling et al. [2018]	Specialized	2	1,000	32,768
EuroSAT Helber et al. [2019]		10		5,400
Resisc45 Cheng et al. [2017]		45		6,300
Retinopathy Graham [2015]		5		42,670
Clevr/count Johnson et al. [2017]	Structured	8	1,000	15,000
Clevr/distance Johnson et al. [2017]		6		15,000
DMLab Beattie et al. [2016]		6		22,735
KITTI/distance Geiger et al. [2013]		4		71,127
dSprites/location Matthey et al. [2017]		16		73,728
dSprites/orientation Matthey et al. [2017]		16		73,728
SmallNORB/azimuth LeCun et al. [2004]		18		12,150
SmallNORB/elevation LeCun et al. [2004]		9		12,150

Abbreviation. Table 4 uses the following abbreviations for VTAB-1k datasets:

- **Natural:** C100 (CIFAR-100), Cal. (Caltech101), DTD, Flw. (Flowers102), Pets, SVHN, Sun (Sun397)
- **Specialized:** PCam (Patch Camelyon), Euro. (EuroSAT), Res. (Resisc45), Retin. (Retinopathy)
- **Structured:** Clv-C (Clevr/count), Clv-D (Clevr/distance), DM (DMLab), KIT-D (KITTI/distance), dSp-L (dSprites/loc), dSp-O (dSprites/ori), sN-A (SmallNORB/azi), sN-E (SmallNORB/ele)

F.4.2 Hyperparameter

We report the hyperparameters used in our fine-tuning vision transformer experiment in Table 12.

Table 12: Training hyperparameters for fine-tuning vision transfomer experiments.

Hyperparameter	Value
Optimizer	AdamW
Learning rate	Grid search
LR scheduler	Cosine
Warmup epochs	10
Epochs	100
Effective batch size	32
Max sequence length	512
Adapted modules	All linear layers in attention and feedforward blocks

F.5 More ablation studies

F.5.1 PRISM as initialization for LoRA

We investigate whether making the subspace projection $\mathbf{U}^{(r)}$ trainable improves performance.

Setup. We fine-tune Gemma-7B on MetaMathQA and evaluate on GSM8K and MATH with the same setup in Section 4. Figure 12 shows no clear benefit from training $\mathbf{U}^{(r)}$. The performance remains similar while the number of trainable parameters increases.

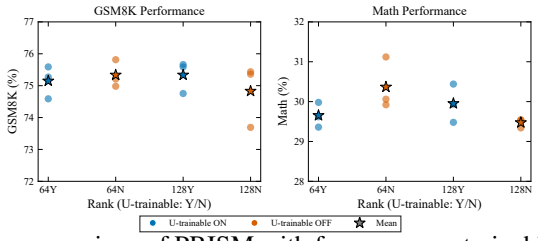

Figure 12: Performance comparison of PRISM with frozen versus trainable $\mathbf{U}^{(r)}$ on Gemma-7B.

F.5.2 Effect of Weight Pairs in Fine-tuning

Table 13 shows the effect of each weight pair on model performance. On GSM8K, adding more components consistently improves performance for PRISM. However, on MATH, no clear improvement emerges when adding more components for either PRISM or PiSSA. In contrast, LoRA's performance degrades with additional components, likely because optimizing more parameters becomes more challenging without explicit weight coordination.

Table 13: **Ablation study on the effect of each weight-pair component.** Results on Gemma-7B fine-tuned on MetaMathQA and evaluated on GSM8K and MATH.

Method	GSM8K			MATH		
	$(\mathbf{W}_V, \mathbf{W}_O)$	$(\mathbf{W}_{\text{up}}, \mathbf{W}_{\text{down}})$	$(\mathbf{W}_V, \mathbf{W}_O, \mathbf{W}_{\text{up}}, \mathbf{W}_{\text{down}})$	$(\mathbf{W}_V, \mathbf{W}_O)$	$(\mathbf{W}_{\text{up}}, \mathbf{W}_{\text{down}})$	$(\mathbf{W}_V, \mathbf{W}_O, \mathbf{W}_{\text{up}}, \mathbf{W}_{\text{down}})$
LoRA	68.08	69.83	68.01	22.22	21.14	20.01
PiSSA	73.62	74.00	73.51	28.50	28.56	28.68
PRISM	**73.69**	**74.83**	**74.83**	**29.76**	**29.42**	**29.47**

G Training curve plots

Figure 13,14,16,15,17,18 show the training curves of different methods on the arithmetic reasoning task. Our method consistently has better training curves and gradient norm thoughout training.

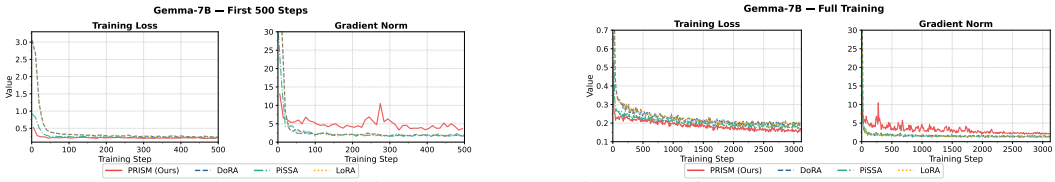

Figure 13: Training curves and gradient norm for Gemma-7B.

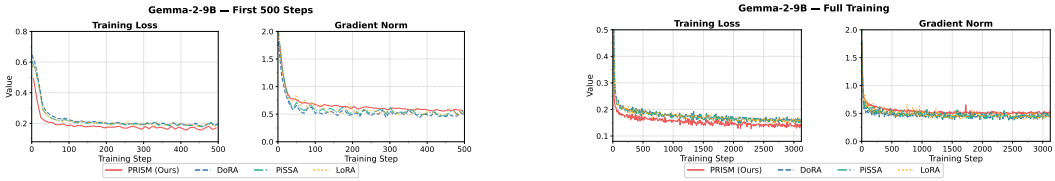

Figure 14: Training curves and gradient norm for Gemma2-9B.

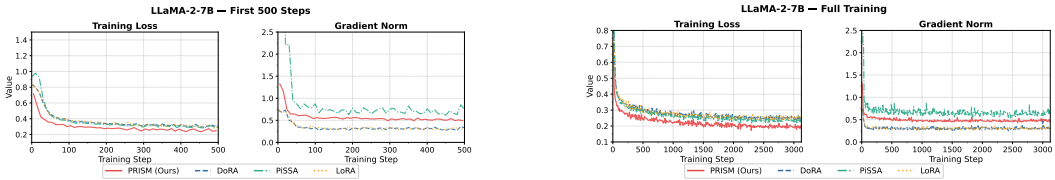

Figure 15: Training curves and gradient norm for Llama2-7B.

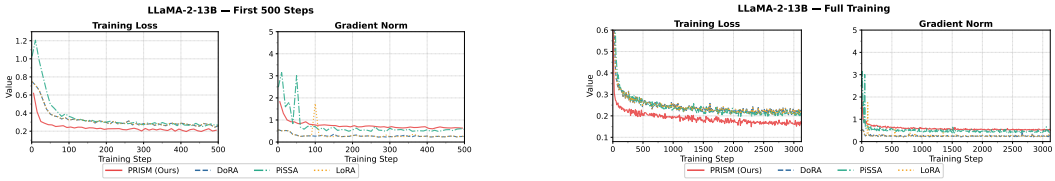

Figure 16: Training curves and gradient norm for Llama2-13B.

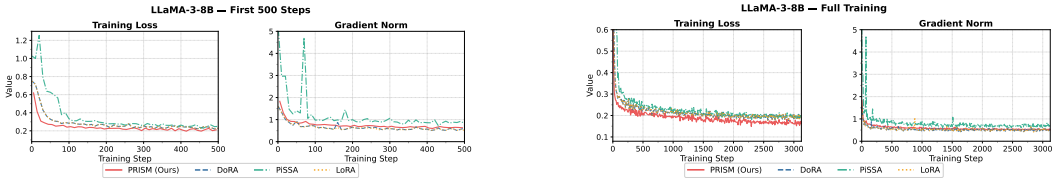

Figure 17: Training curves and gradient norm for Llama3-8B.

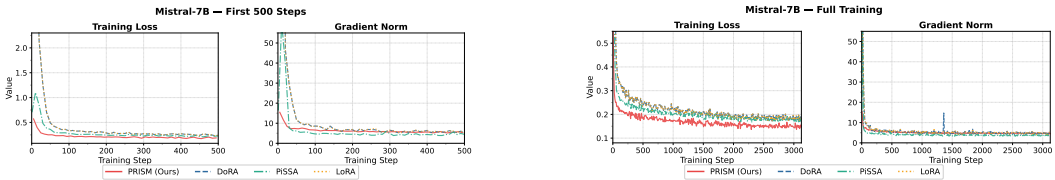

Figure 18: Training curves and gradient norm for Mistral-7B.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction describe PRISM as a parameter-efficient fine-tuning method that exploits coupled weight-pair interactions in transformers via spectral subspace alignment. These claims are substantiated by the methodology in Section 3 and the experimental results in Section 4, which span toy models, language models (arithmetic reasoning, code generation, commonsense reasoning), and vision transformers.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We provide a limitation paragraph as a part of Section 5.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: Theorem 3.1 (Section 3) states its assumptions explicitly (realizable first-order targets, fixed orthonormal rank-r bases, an isotropic prior for the average-case statement), and its complete proof is given in Appendix D. The second-order bound in Equation (7) is derived in Appendix E.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: Figure 5 reports shaded regions corresponding to standard deviations across three random seeds. The toy model learning curves in Figures 3 and 4 are similarly aggregated across multiple runs.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We submitted the source code along side with the paper.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: We provide the detail settings in Appendix F.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: For fine-tuning experiments on language and vision models, we report mean and standard deviation across 3 independent runs with different random seeds. Toy model experiments are averaged over 10 independent seeds, with shaded regions in learning curves denoting $\pm$ standard deviation. All reported error intervals are 1-sigma standard deviations.

   Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: All large-scale fine-tuning experiments on language models (7B–13B parameters) and vision transformers were conducted on a single NVIDIA H200 GPU (140GB memory). Smaller-scale experiments, including toy model studies and ablations, were run on NVIDIA A100 GPUs (80GB memory). Per-run wall-clock time and total compute estimates for each experiment are reported in the appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification:

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: PRISM brings a new fresh aspect in fine-tuning LLMs under the architectural awareness. As a parameter-efficient fine-tuning method, it inherits the same considerations as the underlying foundation models, but introduces no new misuse vector beyond existing PEFT methods.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: The paper does not release new pretrained models or scraped datasets. It releases only adaptation code and adapter weights for publicly available open-weight models, which pose no additional misuse risk beyond the base models.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: All pretrained models (Mistral, Llama-2/3, Gemma, DINOv2), datasets (MetaMathQA, Code-Feedback, Commonsense170K, VTAB-1k), and benchmarks (GSM8K, MATH, HumanEval, MBPP) are cited with their original papers, and we comply with their respective licenses.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [N/A]

Justification: The paper does not release new datasets or pretrained models.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: The paper does not involve crowdsourcing or research with human subjects.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: The paper does not involve research with human subjects, so IRB approval is not applicable.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes. and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: We only use LLMs for writing, editing, or formatting purposes.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.