BEYOND THE MEAN GRADIENT: SPECTRAL ALLOCATION FOR CONDITIONAL LOW-RANK ADAPTATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Low-rank adaptation reduces fine-tuning cost by constraining weight updates to low-rank matrices, but standard LoRA still applies the same learned update to every input. Gradient-aware methods improve this shared update using an aggregate calibration gradient, which can suppress systematic differences in the update directions preferred by individual samples. We propose Spectral Gradient Basis LoRA (***SGB-LoRA***), which turns this within-task gradient variation into an input-conditioned low-rank update under a fixed per-matrix rank budget. SGB-LoRA decomposes per-sample gradients into a shared mean and centered residuals, extracts reusable residual modes, and allocates rank across mean and residual directions according to both gradient energy and low-rank compressibility. The retained bases initialize LoRA branches, while coordinates refitted on the same bases supervise a lightweight prompt-side router that predicts signed branch weights. After warm-up, the router and branches are jointly optimized, and inference requires neither targets nor gradients. We show that the proposed allocation is optimal for calibration-gradient reconstruction within a fixed recovered spectral family. Across instruction following, mathematical reasoning, code generation, and commonsense reasoning, SGB-LoRA consistently outperforms standard LoRA and strong gradient-aware baselines. Further analyses show that within-task gradient variation is compressible, stable across calibration subsets, and predictable from the prompt, supporting the mechanism behind the observed gains.

1 INTRODUCTION

Low-rank adaptation (LoRA) reduces fine-tuning cost by expressing each weight update with a small number of low-rank parameters (Hu et al., 2022). A central question is therefore how to use this limited capacity effectively. Existing methods improve LoRA using structure from pretrained weights, activations, or gradients (Paischer et al., 2025; Wang et al., 2024; Zhang et al., 2025). In particular, gradient-aware methods such as LoRA-GA and LoRA-One use gradients aggregated over calibration examples to identify promising shared update directions.

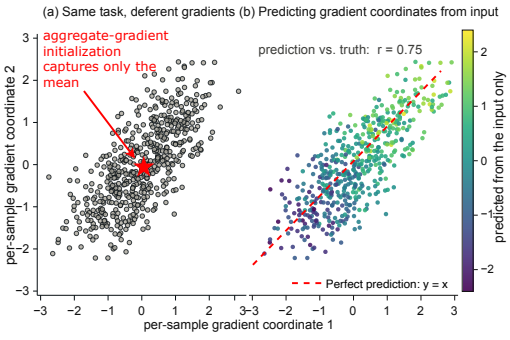

Figure 1: Per-sample gradient variation and input-side coordinate prediction on GSM8K.

Aggregation, however, can hide systematic differences across examples. Two examples may induce descent components of similar magnitude but opposite sign along the same parameter direction, causing that direction to vanish in the average even though it is useful to both examples in different ways. Thus, the mean gradient captures what examples share but discards how their preferred updates differ. This motivates our central question: **can a fixed low-rank budget preserve the shared update while also exploiting predictable variation in per-example gradients?**

Figure 1 provides initial evidence that this variation is structured. For 512 GSM8K examples evaluated at the same frozen model, centered gradients exhibit a broad but continuous distribution along

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

the leading residual modes, while a prompt-only predictor correlates with the first spectral coordinate at $r = 0.75$. This suggests that per-example gradients provide both *update directions* worth retaining and *coordinates* indicating how different inputs should use them. Existing routed LoRA methods already make adaptation input dependent (Tian et al., 2024; Li et al., 2024), while ELREA and GradientSpace use per-example gradients to organize specialized experts (Li et al., 2025; Sridharan et al., 2025). We instead model gradient variation as continuous combinations of reusable directions, with both the directions and their routing targets derived from the same gradient decomposition.

Based on this idea, we propose *Spectral Gradient Basis LoRA* (SGB-LoRA). As shown in Figure 2, SGB-LoRA decomposes per-example calibration gradients into a shared mean and centered residuals, then extracts dominant residual modes through PCA. Since an important mode may still require high matrix rank, we introduce energy-aware rank allocation (EARA), which lets singular directions from the mean and residual modes compete for a single per-matrix rank budget according to both gradient energy and low-rank compressibility. The retained bases initialize LoRA branches. We then refit each residual on these actual low-rank bases and use the resulting coefficients to supervise a lightweight prompt-side router, which predicts signed branch weights from the input alone. After a short warm-up, the router and branches are jointly optimized; inference requires neither targets nor gradients.

Across instruction following, mathematical reasoning, code generation, and commonsense reasoning, SGB-LoRA consistently improves over standard LoRA and strong gradient-aware baselines. Controlled analyses further show that the per-example gradient variation used by SGB-LoRA is compressible, stable across calibration subsets, and predictable from the prompt.

In summary, our contributions are threefold: (1) SGB-LoRA derives shared and residual low-rank bases from per-example gradients and learns input-dependent combinations from the same gradient coordinates; (2) EARA allocates a fixed rank budget according to gradient energy and low-rank compressibility, with a corresponding reconstruction optimality guarantee; and (3) experiments across diverse language tasks validate both the effectiveness of SGB-LoRA and the structure underlying its conditional adaptation.

2 Related Work

Structured shared low-rank adaptation. LoRA parameterizes weight updates as low-rank matrices while freezing the pretrained backbone (Hu et al., 2022). DoRA separates weight magnitude and direction; PiSSA initializes adapters from principal weight components; and CorDA conditions weight decomposition on calibration activations (Liu et al., 2024; Meng et al., 2024; Yang et al., 2024). AdaLoRA reallocates rank from training-time importance estimates, while EVA uses explained activation variance for initialization and rank allocation (Zhang et al., 2023; Paischer et al., 2025). Gradient-informed methods construct or optimize a shared subspace: LoRA-GA aligns the initial low-rank and full gradients, LoRA-One uses a one-step full gradient with preconditioned optimization, and LoRA-Pro corrects factor gradients during training (Wang et al., 2024; Zhang et al., 2025; Wang et al., 2025). SGB-LoRA retains a shared gradient component and additionally allocates rank to centered per-example variation within each adapted matrix.

Input-conditioned low-rank adaptation. Conditional adapters route among multiple low-rank branches. MoLE combines LoRA experts, HydraLoRA shares a down-projection across multiple up-projections, MixLoRA uses a sparse expert mixture, and MiLoRA caches prompt-aware routing decisions before generation (Wu et al., 2024; Tian et al., 2024; Li et al., 2024; Zhang et al., 2024). GOAT and Sparse Spectral LoRA incorporate pretrained-weight spectral structure, while Queryable LoRA retrieves input-dependent combinations of shared low-rank atoms (Fan et al., 2025; Nejatimanzari et al., 2026; Vaidya et al., 2026). SGB-LoRA derives both objects being routed from centered gradients: retained residual modes form the branches, and their signed coordinates supervise the router.

Organizing adaptation with per-example gradients. ELREA clusters instructions by gradient direction and trains cluster-specific experts, while GradientSpace clusters per-example LoRA gradients and uses the assignments to supervise an input router (Li et al., 2025; Sridharan et al., 2025).

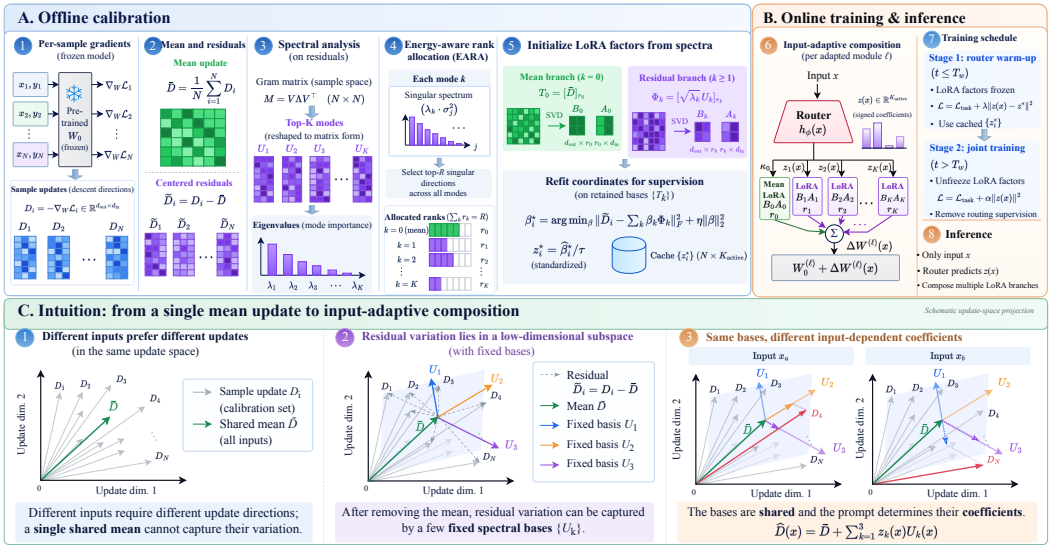

Figure 2: Overview of SGB-LoRA. Per-sample calibration directions are decomposed into a shared mean and centered residuals. The residuals are compressed by snapshot PCA, and EARA allocates one rank budget across the singular directions of the mean and residual modes. The retained low-rank bases initialize a conditional LoRA adapter, while refitted coordinates supervise a prompt-side router used at inference.

Both organize adaptation through discrete sample groups. Gradient Atoms instead decomposes training gradients into sparse components for behavior discovery, attribution, and intervention (Rosser, 2026). SGB-LoRA represents each example as a continuous signed combination of reusable residual modes. One mean–residual decomposition therefore determines branch construction, rank allocation, and routing supervision under a single per-matrix rank budget.

3 Method

Figure 2 summarizes SGB-LoRA. An offline calibration subset provides per-example descent directions, which SGB-LoRA decomposes into a shared mean and sample-dependent residuals. It compresses these components into reusable low-rank matrices, allocates one factor-rank budget across them, and trains a prompt-side router to predict their signed coordinates at inference.

3.1 Per-Sample Gradient Decomposition

Consider a frozen linear map $W_0 \in \mathbb{R}^{m\times n}$. Standard LoRA adapts it as

$$y = (W_0 + \rho BA)\, h. \tag{1}$$

Here $B \in \mathbb{R}^{m\times R}$, $A \in \mathbb{R}^{R\times n}$, and $\rho = \alpha/R$ is the usual scaling factor (Hu et al., 2022). Since BA is input-independent, all samples share the same update. SGB-LoRA retains the total rank R, but allows part of the update to be combined with input-dependent coefficients.

We draw N calibration samples from the training corpus and define each sample loss as the mean over its valid target positions, preventing target length from directly setting the gradient scale. For sample x_i and adapted module ℓ, let $H_i^{(\ell)}$ collect the module inputs and $\Delta_i^{(\ell)}$ the aligned output gradients. Its local descent direction is

$$D_i^{(\ell)} = -\nabla_{W^{(\ell)}} \mathcal{L}_i(W_0) = -\Delta_i^{(\ell)} (H_i^{(\ell)})^\top. \tag{2}$$

Calibration computes this target-dependent signal while keeping the backbone fixed and taking no optimizer step. The deployed router later replaces it with a prompt-side prediction. We derive the construction for one module and omit ℓ until the adapter is realized. We use $\|\cdot\|_F$ for the Frobenius norm and $[X]_r$ for a best rank-r approximation of X.

We decompose the calibrated directions into a shared mean and centered residuals:

$$\bar{D} = \frac{1}{N}\sum_{i=1}^{N} D_i, \qquad \widetilde{D}_i = D_i - \bar{D}. \tag{3}$$

Here $\bar{D}$ is the aggregate descent direction, while $\widetilde{D}_i$ records how sample i departs from it. By centering and the Eckart–Young–Mirsky theorem (Eckart & Young, 1936; Mirsky, 1960), the best fixed rank-R matrix is $[\bar{D}]_R$, with reconstruction error

$$\min_{\mathrm{rank}(M)\le R} \frac{1}{N}\sum_i \|D_i - M\|_F^2 = \sum_{j>R} (\sigma_j^{(0)})^2 \quad + \frac{1}{N}\sum_i \|\widetilde{D}_i\|_F^2. \tag{4}$$

where $\sigma_j^{(0)}$ are the singular values of $\bar{D}$. Increasing R reduces the mean-truncation term, while every fixed matrix retains the full residual term. Equation (4) exactly characterizes calibration-time reconstruction; downstream experiments evaluate the trained model. The appendix provides the full decomposition and proof.

3.2 Spectral Basis Construction and Rank Allocation

The shared path in Fig. 2 applies an SVD directly to $\bar{D}$. For the residual path, snapshot PCA (Sirovich, 1987) avoids forming the $mn \times mn$ covariance by using the $N \times N$ Gram matrix:

$$G_{ij} = \langle \widetilde{D}_i, \widetilde{D}_j \rangle_F, \qquad G v_k = \nu_k v_k. \tag{5}$$

Each positive eigenpair then defines a residual mode, its average energy, and the corresponding per-sample coordinate:

$$U_k = \frac{1}{\sqrt{\nu_k}} \sum_i v_{ik} \widetilde{D}_i, \quad \lambda_k = \frac{\nu_k}{N}, \quad a_{ik} = \sqrt{\nu_k}\, v_{ik}. \tag{6}$$

For the $q = \mathrm{rank}(G) \le N - 1$ positive eigenvalues, the modes U_k are Frobenius-orthonormal and exactly reconstruct each residual as $\widetilde{D}_i = \sum_{k=1}^{q} a_{ik} U_k$. The signed coordinate a_{ik} measures how strongly sample i uses mode k, while λ_k is the mode's average energy.

PCA identifies important variation modes, but a mode can still have high matrix rank. We therefore score its singular directions individually. Let $\sigma_j^{(0)}$ and $\sigma_j^{(k)}$ denote the singular values of $\bar{D}$ and U_k, respectively. One retained singular direction consumes one unit of rank—a *rank slot*—and recovers

$$s_{0j} = (\sigma_j^{(0)})^2, \qquad s_{kj} = \lambda_k (\sigma_j^{(k)})^2. \tag{7}$$

Energy-aware rank allocation (EARA) pools the mean and residual slots and selects the R largest positive scores. If r_0 selected slots come from $\bar{D}$ and r_k from mode U_k, their total satisfies $r_0 + \sum_{k\le K_c} r_k \le R$. We retain

$$T_0 = [\bar{D}]_{r_0}, \qquad T_k = [U_k]_{r_k} \quad (1 \le k \le K_c). \tag{8}$$

We initialize the candidate pool with $K_c = \min(2R, q)$, screen modes on held-out calibration samples, and expand the pool until the selected slots stabilize. The retained set $\mathcal{K}_\ell = \{k \le K_c : r_k^{(\ell)} > 0\}$ contains candidates that receive at least one rank slot, and $K_\ell = |\mathcal{K}_\ell|$ sets the router-head dimension. A module with $K_\ell = 0$ uses the all-mean allocation and requires no router head.

Proposition 1 (EARA optimality)**.** *For a fixed candidate pool, fixed PCA coordinates, and exact singular spectra, selecting the largest positive slot scores minimizes the average calibration-gradient reconstruction error over all allocations with total rank at most R.*

The appendix gives the exact error decomposition, proof, candidate-pool certificate, and sketched large-model approximation.

3.3 Conditional LoRA via a Prompt-Side Router

PCA coordinates describe the full modes; low-rank truncation changes the realized basis. For each $k \in \mathcal{K}_\ell$, we first carry the mode energy into the retained matrix:

$$\Phi_k = \sqrt{\lambda_k}\, T_k. \tag{9}$$

We then refit the coefficients on these truncated bases by ridge regression:

$$\beta_i^\star = \arg\min_\beta \left\| \widetilde{D}_i - \sum_{k \in \mathcal{K}_\ell} \beta_k \Phi_k \right\|_F^2 + \eta \|\beta\|_2^2. \tag{10}$$

Finally, we standardize the refitted coefficients to obtain router targets:

$$\tau_k^2 = N^{-1} \sum_i (\beta_{ik}^\star)^2 + \epsilon_\tau, \qquad z_{ik}^\star = \beta_{ik}^\star / \tau_k. \tag{11}$$

Thus a_{ik} supports spectral allocation, $\beta_i^\star$ reconstructs on the retained bases, $z_i^\star$ supervises the router, and $z^{(\ell)}(x)$ is the deployed prediction.

For block b, we pool and normalize the non-padding prompt states into a descriptor $h_b(x)$. A shared MLP trunk and a module-specific linear head predict signed coordinates:

$$z^{(\ell)}(x) = W_2^{(\ell)} \, \mathrm{GELU}(W_1 h_b(x)) \, . \tag{12}$$

We initialize every module-specific head with $W_2^{(\ell)}\big|_{\text{init}} = 0$. Signed outputs represent positive and negative contributions of each residual mode. The appendix specifies the descriptor and empty-prompt handling.

Let $\Psi_0 = T_0$ and $\Psi_k = \Phi_k$. We factor each retained matrix by a balanced SVD so that $B_k A_k = \Psi_k$; the explicit factorization is given in the appendix. We choose a fixed module scale s_ℓ so that the retained update has output RMS equal to a fraction γ of the frozen module output on the calibration set. With $\kappa_t = \min(1, t/T_w)$, the update at training step t is

$$\Delta W_t^{(\ell)}(x) = s_\ell \left[\kappa_t B_0 A_0 + \sum_{k \in \mathcal{K}_\ell} z_k^{(\ell)}(x) \tau_k B_k A_k \right]. \tag{13}$$

Because $\kappa_0 = 0$ and the router heads start at zero, $\Delta W_0^{(\ell)}(x) = 0$, so initialization preserves the pretrained function. The update rank remains at most R for every input.

The backbone is frozen throughout. During the first T_w steps, the LoRA factors are also frozen and the router is trained with

$$\begin{aligned} \mathcal{L}(t) &= \mathcal{L}_{\text{task}} + \omega_0 (1 - t/T_w)_+ \mathcal{L}_{\text{route}}, \\ \mathcal{L}_{\text{route}} &= \mathbb{E}_{i,\ell \in \mathcal{M}_+} \left[\frac{\| z^{(\ell)}(x_i) - z_i^{\star,(\ell)} \|_2^2}{K_\ell} \right]. \end{aligned} \tag{14}$$

where $\mathcal{M}_+$ contains modules with at least one residual branch. The routing term updates only the router, while the task loss may also update a task-specific head. At $t = T_w$, we remove the routing term, unfreeze the LoRA factors, and optimize the factors and router jointly for the task. The exact reconstruction result characterizes the calibrated initialization; joint optimization then adapts the basis and routing to the task objective.

At inference, autoregressive models compute the prompt coordinates during prefill and cache them for decoding. The shared branch folds into $W_0^{(\ell)}$, and the residual branches execute as one fused low-rank operation with total rank at most $R - r_0$. The factors occupy at most the $R(m+n)$ budget of rank-R LoRA; the router supplies the additional trainable parameters measured in Section 4.6. Scalable calibration uses four shared forward–backward passes across all adapted modules and adds reconstruction passes when the candidate pool expands. The appendix provides numerical, training, and memory details.

4 Experiments

We evaluate SGB-LoRA from five complementary perspectives. We first test whether per-sample gradient structure consistently improves downstream adaptation across tasks, backbones, and rank budgets. We then examine whether this structure is compressible, stable across calibration subsets, and predictable from the prompt. Controlled ablations identify the contribution of each design choice, followed by analyses of EARA's capacity allocation and the resulting computational overhead.

Table 1: Commonsense reasoning accuracy (%) after Commonsense170K fine-tuning at $R = 32$. † denotes published results. Bold and underlined values denote the best and second-best results.

Backbone	Method	BoolQ	PIQA	SIQA	HellaS.	WinoG.	ARC-E	ARC-C	OBQA	Avg.
Llama-2-7B	LoRA†	69.80	79.90	79.50	83.60	82.60	79.80	64.70	81.00	77.61
	DoRA†	71.80	83.70	76.00	<u>89.10</u>	82.60	83.70	68.20	82.40	79.69
	NEAT (v1)†	71.70	83.90	<u>80.20</u>	88.90	84.30	86.30	71.40	83.00	81.20
	HiRA†	71.22	83.35	79.53	88.12	83.98	<u>86.74</u>	73.81	<u>84.60</u>	81.42
	HiRA-CoTo†	<u>72.11</u>	**84.33**	79.89	88.40	**85.24**	**87.08**	<u>74.49</u>	84.00	<u>81.94</u>
	SGB-LoRA	**74.10**±0.31	<u>83.98</u>±0.22	**81.00**±0.28	**90.10**±0.19	<u>85.12</u>±0.34	86.20±0.27	**79.80**±0.45	**87.60**±0.38	**83.49**±0.31
Llama-3-8B	LoRA†	70.80	85.20	79.90	91.70	84.30	84.20	71.20	79.00	80.79
	DoRA†	74.60	89.30	79.90	<u>95.50</u>	85.60	90.50	80.40	85.80	85.20
	NEAT (v1)†	71.90	86.70	80.90	94.10	86.70	90.90	78.70	84.40	84.30
	HiRA†	<u>75.40</u>	<u>89.70</u>	81.15	95.36	87.70	**93.27**	82.90	<u>88.32</u>	86.72
	HiRA-CoTo†	75.32	**90.15**	**81.99**	95.42	<u>88.08</u>	**93.27**	<u>83.36</u>	**88.40**	<u>87.00</u>
	SGB-LoRA	**76.10**±0.25	89.10±0.18	<u>82.21</u>±0.32	**95.80**±0.12	**88.10**±0.29	<u>92.60</u>±0.24	**90.10**±0.41	88.61±0.36	**87.83**±0.27

4.1 Experimental Setup

Models and training. We use Llama-2-7B (Touvron et al., 2023) and Llama-3-8B (Grattafiori et al., 2024). Main downstream comparisons adapt all Transformer linear maps except embeddings, normalization layers, and the output head. Mechanism analyses, ablations, and efficiency measurements use a controlled attention-only setting on `q_proj`, `k_proj`, `v_proj`, and `o_proj`. All methods within each comparison use the same target modules.

Unless stated otherwise, we use $R = 8$, $\alpha = 16$, rank-stabilized scaling $s = \alpha/\sqrt{R}$, and $N = 512$ calibration examples. We additionally evaluate $R = 32$. Training uses AdamW (Loshchilov & Hutter, 2019) with learning rate 2×10^{-5}, cosine decay, a 0.03 warm-up ratio, sequence length 1024, and macro batch size 32. Main local results use three seeds $\{0, 42, 2026\}$, while ablations and systems measurements are single runs on one NVIDIA B200 GPU.

Datasets and metrics. For commonsense reasoning, we train on Commonsense170K (Hu et al., 2023) and evaluate accuracy on BoolQ (Clark et al., 2019), PIQA (Bisk et al., 2020), SIQA (Sap et al., 2019), HellaSwag (Zellers et al., 2019), WinoGrande (Sakaguchi et al., 2020), ARC-Easy and ARC-Challenge (Clark et al., 2018), and OpenBookQA (Mihaylov et al., 2018), together with their unweighted average.

For generative tasks, we use WizardLM 52K (Xu et al., 2024) with MT-Bench (Zheng et al., 2023), MetaMathQA 100K (Yu et al., 2024) with GSM8K (Cobbe et al., 2021), and CodeFeedback 100K (Zheng et al., 2024) with HumanEval (Chen et al., 2021). We report first-turn GPT-4 judge score, exact-match accuracy, and pass@1, respectively.

Baselines. We compare with LoRA (Hu et al., 2022), DoRA (Liu et al., 2024), PiSSA (Meng et al., 2024), LoRA-GA (Wang et al., 2024), and LoRA-Pro (Wang et al., 2025), with full fine-tuning as a high-capacity reference. Commonsense experiments additionally include NEAT (Zhong et al., 2024), HiRA (Huang et al., 2025), and HiRA-CoTo (Zhuang et al., 2025). Results marked with † are taken from the corresponding papers.

4.2 Main Results

We first ask whether variation beyond the aggregate gradient provides consistent downstream benefits.

Commonsense reasoning. Table 1 shows that SGB-LoRA achieves average accuracies of 83.49 and 87.83 on Llama-2-7B and Llama-3-8B, outperforming HiRA-CoTo by 1.55 and 0.83 points. The improvement is particularly large on ARC-Challenge, where SGB-LoRA exceeds the next-best result by 5.31 and 6.74 points.

This setting requires one adapter to accommodate several reasoning patterns under a fixed rank budget. The consistent gains suggest that retaining a shared update while allowing input-dependent residual combinations uses this capacity more effectively than a single static update.

Table 2: Instruction-tuning results on Llama-2-7B. † denotes published three-run results.

Method	Rank	MT-Bench	GSM8K	HumanEval
Full FT†	–	5.30±0.11	59.36±0.85	**35.31**±2.13
LoRA†	8	5.61±0.10	42.08±0.04	14.76±0.17
DoRA†	8	5.97±0.02	53.07±0.75	19.75±0.41
PiSSA†	8	5.30±0.02	44.54±0.27	16.02±0.78
LoRA-GA†	8	5.95±0.16	53.60±0.30	19.81±1.46
LoRA-Pro†	8	5.72±0.03	57.57±0.50	22.97±0.35
LoRA-GA†	32	5.79±0.09	55.12±0.30	20.18±0.19
LoRA-Pro†	32	5.57±0.51	57.97±0.50	26.63±0.35
SGB-LoRA	8	6.65±0.24	60.71±0.33	26.12±0.42
SGB-LoRA	32	**6.72**±0.20	**60.90**±0.29	27.11±0.38

Table 3: Rank allocation of SGB-LoRA on Llama-2-7B. Residual rank is the fraction assigned to residual branches. Active modes are averaged across modules.

Dataset	R	Residual rank (%)	Active modes	Energy captured (%)	Performance
GSM8K	4	42	1.4	76	56.8
	8	57	3.5	**92**	**60.71**
	16	62	4.3	96	60.10
	32	64	4.7	98	60.8
HumanEval	4	40	1.6	73	23.3
	8	54	3.2	**91**	**26.12**
	16	59	4.0	95	26.7
	32	61	4.4	97	27.1

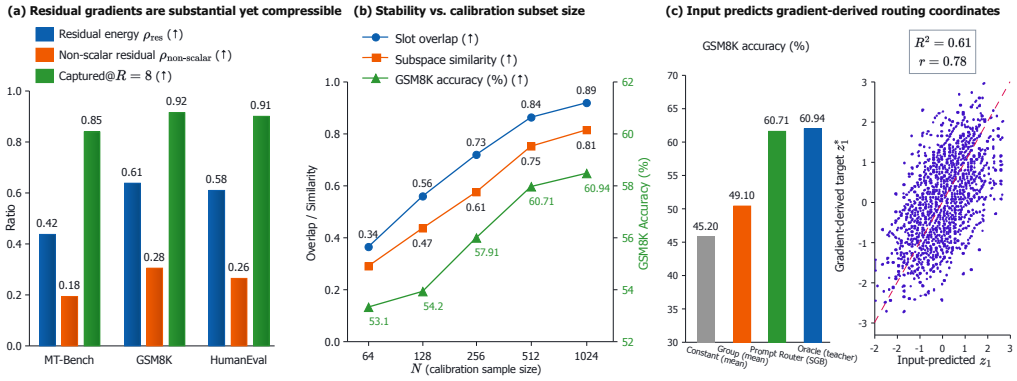

Figure 3: Gradient-basis diagnostics on Llama-2-7B. (a) Magnitude and compressibility of residual gradient variation. (b) Basis stability as calibration size increases. (c) Prompt-side prediction of gradient-derived coordinates on GSM8K.

Instruction following, mathematical reasoning, and code generation. We next evaluate three independently trained generative tasks. At $R = 8$, SGB-LoRA improves over LoRA-Pro by 0.93 points on MT-Bench and 3.14 and 3.15 points on GSM8K and HumanEval. At $R = 32$, the gains remain 1.15, 2.93, and 0.48.

The improvement persists across both rank budgets and across tasks with very different output structures. Since gradient-aware baselines already exploit task-level gradient information, the additional gain suggests that variation around the aggregate gradient provides useful information beyond the shared direction itself.

4.3 Does Per-Sample Gradient Variation Have Exploitable Structure?

The premise of SGB-LoRA requires more than nonzero gradient variation. The variation must be compressible, reproducible across calibration samples, and predictable when target gradients are unavailable. Figure 3 tests these three properties.

Residual variation is substantial but compressible. Across MT-Bench, GSM8K, and HumanEval, 42%–61% of total gradient energy lies outside the shared mean. Even after removing the best sample-specific scaling of that mean, 18%–28% remains as directional variation. The heterogeneity therefore goes beyond simple differences in gradient magnitude.

Despite this variation, EARA captures 85%–92% of the residual spectral energy at $R = 8$. The residual signal is therefore large enough to matter but concentrated enough to be represented by a compact set of modes. This is precisely the regime targeted by SGB-LoRA.

The recovered structure is stable. Figure 3(b) asks whether the recovered modes persist across calibration draws. As N increases from 64 to 1024, slot overlap rises from 0.34 to 0.89, subspace similarity from 0.29 to 0.81, and GSM8K accuracy from 53.10 to 60.94. At $N = 512$, accuracy

already reaches 60.71, indicating that the dominant structure can be recovered reliably from a moderate calibration set.

The prompt predicts gradient coordinates. The remaining question is whether these gradient-derived coordinates can be used without target gradients at inference. Constant and group-level coordinates obtain 45.20 and 49.10 GSM8K accuracy, while the prompt router reaches 60.71, close to the response-gradient oracle at 60.94. Its leading-coordinate prediction reaches $R^2 = 0.61$ and Pearson $r = 0.78$.

The small oracle gap shows that much of the update information exposed by calibration gradients is recoverable from the prompt alone. Together, the three panels connect offline gradient structure to deployable input-conditioned adaptation.

4.4 Ablation Studies

We next test whether the specific way SGB-LoRA constructs and uses this structure accounts for the gains. Table 4 reports controlled ablations on GSM8K and HumanEval using Llama-2-7B, $R = 8$, $N = 512$, and the attention-only setting. All variants share the same training protocol and total rank budget.

Restricting adaptation to the shared mean reduces GSM8K and HumanEval by 4.51 and 4.72 points, confirming the value of residual variation. Removing centering also hurts both tasks, while architecture-matched random bases lose 4.41 and 4.12 points. These controls show that the gain comes from the measured centered-gradient structure rather than from adding routed branches alone.

Table 4: Component ablations on Llama-2-7B at $R = 8$ using one fixed seed.

Variant	GSM8K	HumanEval	Coord. R^2
SGB-LoRA	**60.71**	**26.12**	**0.61**
Shared mean only	56.20	21.40	–
Uncentered basis	57.80	23.80	0.43
Matched random bases	56.30	22.00	0.26
Uniform allocation	58.90	24.60	0.53
Task-only router	57.30	23.10	0.19

The remaining variants examine how this structure is used. Uniform allocation is 1.81 and 1.52 points worse than EARA. Removing coordinate supervision reduces GSM8K and HumanEval to 57.30 and 23.10, while coordinate R^2 falls from 0.61 to 0.19. Thus, the gradients contribute both the directions retained by the adapter and the supervision that teaches the router how to combine them.

4.5 Capacity Allocation

The previous analyses establish that the recovered structure is useful. We now examine how EARA distributes the fixed rank budget across modes and network depth.

Allocation across rank budgets. Table 3 shows that at $R = 8$, the retained bases already capture 92% and 91% of residual energy on GSM8K and HumanEval. Increasing R to 32 raises these values only to 98% and 97%.

The number of active modes grows even more slowly. On GSM8K, a fourfold increase in rank from 8 to 32 changes the average number of retained modes only from 3.5 to 4.7, while HumanEval changes from 3.2 to 4.4. Most sample-dependent variation is therefore concentrated in a small set of modes. Additional rank increasingly refines these modes rather than continually introducing new branches.

Layer-wise allocation and routing. Figure 4 shows that the preferred allocation also varies across network depth. Some `q_proj` modules devote most of their capacity to the shared mean, while others allocate substantially more rank to residual modes, particularly in middle layers.

Routing coefficients also exhibit clear variation across inputs rather than collapsing to a fixed combination. The two panels reveal complementary forms of adaptivity. EARA determines where sample-dependent capacity is useful, while the router determines how that capacity is used for each input.

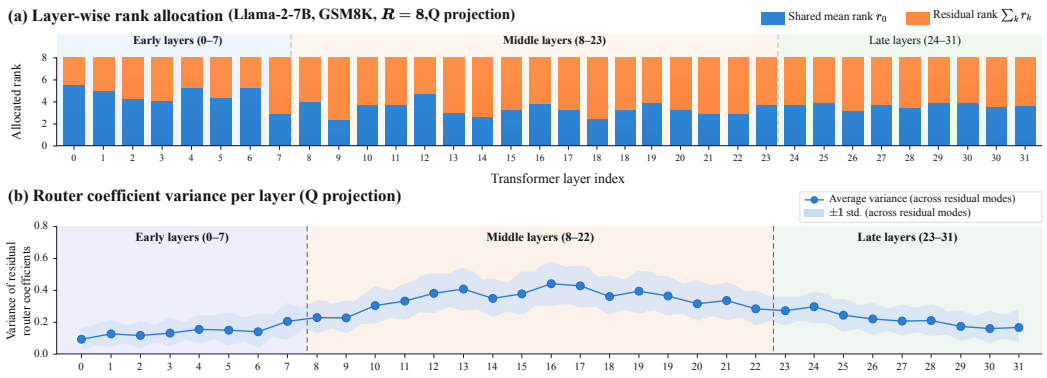

Figure 4: Layer-wise capacity allocation and routing variability for GSM8K at $R = 8$. (a) Shared–residual rank allocation for `q_proj`. (b) Across-input variance of residual routing coefficients.

Table 5: Single-run efficiency on one NVIDIA B200 at $R = 8$.

	Trainable	Offline construction		Fine-tuning		Inference	
Method	Params	Time	Peak mem.	Peak mem.	Time	Prefill	Decode
LoRA	8.39M	–	–	22.8 GB	1.95 h	42.0 ms	128 tok/s
LoRA-GA	8.39M	47 s	18.5 GB	22.8 GB	1.96 h	42.0 ms	128 tok/s
SGB-LoRA	9.13M	4.5 min	20.5 GB	23.7 GB	2.10 h	43.0 ms	123 tok/s

4.6 Computational Cost

Finally, we examine the cost of obtaining and using the per-sample gradient structure. Table 5 compares LoRA, LoRA-GA, and SGB-LoRA on the attention-only Llama-2-7B MetaMathQA–GSM8K setting at $R = 8$.

Relative to LoRA, SGB-LoRA adds 0.74M trainable parameters, while peak training memory and training time increase by 3.9% and 7.7%. Its main additional cost is the one-time 4.5-minute spectral calibration, compared with 47 seconds for LoRA-GA.

At inference, routing adds only 1.0 ms to prefill and reduces decoding throughput from 128 to 123 tokens/s. Coordinates are computed once during prefill and cached, while the shared branch can be merged into the backbone and the residual branches fused into one low-rank operation. Thus, SGB-LoRA concentrates most of its additional cost in offline calibration while largely preserving the online efficiency of standard LoRA.

5 Conclusion

We introduced SGB-LoRA, a conditional low-rank adaptation method that models per-example gradient variation beyond the shared mean. SGB-LoRA constructs compact residual gradient bases, allocates a fixed rank budget according to gradient energy and low-rank compressibility, and learns a prompt-side router to predict their signed combination weights. Our analysis characterizes the reconstruction limit of a fixed update and the optimal allocation within the recovered spectral family. Experiments across diverse language tasks show consistent gains over strong PEFT baselines, while diagnostics confirm that the residual gradient structure is compressible, stable, and predictable from the input. These results establish within-task gradient variation as a useful signal for input-dependent low-rank adaptation.

Limitations. SGB-LoRA requires an additional offline calibration stage to collect per-example gradients and recover their spectral structure. Its theoretical guarantees apply to calibration-gradient reconstruction rather than downstream loss, and the learned branches may deviate from the initial spectral basis during joint training. The current evaluation is also limited to 7B–8B language models and several language tasks.

AI USE STATEMENT

Generative AI tools were used to assist with code development, literature retrieval, language editing, and LaTeX formatting. AI-generated or AI-assisted code was manually reviewed and validated by the authors prior to use. Generative AI was not relied upon to independently determine the scientific conclusions of this work. The authors take full responsibility for the final implementation, experimental results, manuscript content, claims, and artifacts.

REFERENCES

Haim Avron, Huy L. Nguyen, and David P. Woodruff. Subspace embeddings for the polynomial kernel. In *Advances in Neural Information Processing Systems*, volume 27, 2014.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. PIQA: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 7432–7439, 2020.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating Large Language Models Trained on Code, 2021. URL https://arxiv.org/abs/2107.03374.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of NAACL-HLT*, pp. 2924–2936. Association for Computational Linguistics, 2019. doi: 10.18653/v1/N19-1300.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try ARC, the AI2 reasoning challenge, 2018.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training Verifiers to Solve Math Word Problems, 2021. URL https://arxiv.org/abs/2110.14168.

Carl Eckart and Gale Young. The Approximation of One Matrix by Another of Lower Rank. *Psychometrika*, 1(3):211–218, 1936. doi: 10.1007/BF02288367. URL https://doi.org/10.1007/BF02288367.

Chenghao Fan, Zhenyi Lu, Sichen Liu, Chengfeng Gu, Xiaoye Qu, Wei Wei, and Yu Cheng. Make LoRA great again: Boosting LoRA with adaptive singular values and mixture-of-experts optimization alignment. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 15804–15832. PMLR, 2025.

Aaron Grattafiori et al. The Llama 3 herd of models, 2024.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations (ICLR)*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Zhiqiang Hu, Lei Wang, Yihuai Lan, Wanyu Xu, Ee-Peng Lim, Lidong Bing, Xing Xu, Soujanya Poria, and Roy Ka-Wei Lee. LLM-Adapters: An adapter family for parameter-efficient fine-tuning of large language models, 2023.

Qiushi Huang, Tom Ko, Zhan Zhuang, Lilian Tang, and Yu Zhang. HiRA: Parameter-efficient hadamard high-rank adaptation for large language models. In *International Conference on Learning Representations*, 2025.

Vladimir Koltchinskii and Karim Lounici. Concentration inequalities and moment bounds for sample covariance operators. *Bernoulli*, 23(1):110–133, 2017. doi: 10.3150/15-BEJ730.

Dengchun Li, Yingzi Ma, Naizheng Wang, Zhengmao Ye, Zhiyuan Cheng, Yinghao Tang, Yan Zhang, Lei Duan, Jie Zuo, Cal Yang, and Mingjie Tang. MixLoRA: Enhancing large language models fine-tuning with LoRA-based mixture of experts. *arXiv preprint arXiv:2404.15159*, 2024.

Yinghao Li, Vianne Gao, Chao Zhang, and MohamadAli Torkamani. Ensembles of low-rank expert adapters. In *International Conference on Learning Representations (ICLR)*, 2025. arXiv:2502.00089.

Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. DoRA: Weight-decomposed low-rank adaptation. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, volume 235 of *Proceedings of Machine Learning Research*, pp. 32100–32121. PMLR, 2024. Oral.

Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization. In *International Conference on Learning Representations*, 2019. URL https://arxiv.org/abs/1711.05101.

Fanxu Meng, Zhaohui Wang, and Muhan Zhang. PiSSA: Principal singular values and singular vectors adaptation of large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. URL https://openreview.net/forum?id=6ZBHIEtdP4. Spotlight; arXiv:2404.02948.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2381–2391. Association for Computational Linguistics, 2018. doi: 10.18653/v1/D18-1260.

L. Mirsky. Symmetric Gauge Functions and Unitarily Invariant Norms. *The Quarterly Journal of Mathematics*, 11(1):50–59, 1960. doi: 10.1093/qmath/11.1.50. URL https://academic.oup.com/qjmath/article/11/1/50/1525786.

Omid Nejatimanzari, Hojat Asgariandehkordi, Taha Koleilat, Yiming Xiao, and Hassan Rivaz. Sparse spectral LoRA: Routed experts for medical VLMs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 35351–35362, 2026.

Fabian Paischer, Lukas Hauzenberger, Thomas Schmied, Benedikt Alkin, Marc Peter Deisenroth, and Sepp Hochreiter. Parameter efficient fine-tuning via explained variance adaptation. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 38, 2025.

Ninh Pham and Rasmus Pagh. Fast and scalable polynomial kernels via explicit feature maps. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pp. 239–247, 2013.

J. Rosser. Gradient atoms: Unsupervised discovery, attribution and steering of model behaviors via sparse decomposition of training gradients, 2026.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. WinoGrande: An adversarial winograd schema challenge at scale. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 8732–8740, 2020.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. Social IQa: Commonsense reasoning about social interactions. In *Proceedings of EMNLP-IJCNLP*, pp. 4463–4473. Association for Computational Linguistics, 2019. doi: 10.18653/v1/D19-1454.

Lawrence Sirovich. Turbulence and the Dynamics of Coherent Structures. I. Coherent Structures. *Quarterly of Applied Mathematics*, 45(3):561–571, 1987. doi: 10.1090/qam/910462. URL https://doi.org/10.1090/qam/910462.

Shrihari Sridharan, Deepak Ravikumar, Anand Raghunathan, and Kaushik Roy. GradientSpace: Unsupervised data clustering for improved instruction tuning, 2025.

Chunlin Tian, Zhan Shi, Zhijiang Guo, Li Li, and Chengzhong Xu. HydraLoRA: An asymmetric LoRA architecture for efficient fine-tuning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

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

Omatharv Bharat Vaidya, Connor T. Jerzak, Nhat Ho, and Chandrajit Bajaj. Queryable LoRA: Instruction-regularized routing over shared low-rank update atoms, 2026.

Shaowen Wang, Linxi Yu, and Jian Li. LoRA-GA: Low-rank adaptation with gradient approximation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. URL https://openreview.net/forum?id=VaLAWrLHJv.

Zhengbo Wang, Jian Liang, Ran He, Zilei Wang, and Tieniu Tan. Lora-pro: Are low-rank adapters properly optimized? In *International Conference on Learning Representations*, 2025.

Xun Wu, Shaohan Huang, and Furu Wei. Mixture of LoRA experts. In *International Conference on Learning Representations (ICLR)*, 2024. URL https://openreview.net/forum?id=uWvKBCYh4S.

Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, Qingwei Lin, and Daxin Jiang. WizardLM: Empowering Large Pre-Trained Language Models to Follow Complex Instructions. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/82eec786fdfbbfa53450c5feb7d1ac92-Abstract-Conference.html.

Yibo Yang, Xiaojie Li, Zhongzhu Zhou, Shuaiwen Leon Song, Jianlong Wu, Liqiang Nie, and Bernard Ghanem. CorDA: Context-oriented decomposition adaptation of large language models for task-aware parameter-efficient fine-tuning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. URL https://openreview.net/forum?id=Gi00NVru6n.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. MetaMath: Bootstrap Your Own Mathematical Questions for Large Language Models. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/c400474e8a36d0812fdee52739288b12-Abstract-Conference.html.

Yi Yu, Tengyao Wang, and Richard J. Samworth. A useful variant of the davis–kahan theorem for statisticians. *Biometrika*, 102(2):315–323, 2015. doi: 10.1093/biomet/asv008.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4791–4800. Association for Computational Linguistics, 2019. doi: 10.18653/v1/P19-1472.

Jingfan Zhang, Yi Zhao, Dan Chen, Xing Tian, Huanran Zheng, and Wei Zhu. MiLoRA: Efficient mixture of low-rank adaptation for large language models fine-tuning. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 17071–17084. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.findings-emnlp.994.

Qingru Zhang, Minshuo Chen, Alexander Bukharin, Nikos Karampatziakis, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. AdaLoRA: Adaptive budget allocation for parameter-efficient fine-tuning. In *International Conference on Learning Representations (ICLR)*, 2023. URL https://openreview.net/forum?id=lq62uWRJjiY.

Yuanhe Zhang, Fanghui Liu, and Yudong Chen. LoRA-One: One-step full gradient could suffice for fine-tuning large language models, provably and efficiently. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, volume 267 of *Proceedings of Machine Learning Research*, pp. 75513–75574. PMLR, 2025.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/91f18a1287b398d378ef22505bf41832-Abstract-Datasets_and_Benchmarks.html.

Tianyu Zheng, Ge Zhang, Tianhao Shen, Xueling Liu, Bill Yuchen Lin, Jie Fu, Wenhu Chen, and Xiang Yue. OpenCodeInterpreter: Integrating Code Generation with Execution and Refinement. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 12834–12859, 2024. doi: 10.18653/v1/2024.findings-acl.762. URL https://aclanthology.org/2024.findings-acl.762/.

Yibo Zhong, Haoxiang Jiang, Lincan Li, Ryumei Nakada, Tianci Liu, Linjun Zhang, Huaxiu Yao, and Haoyu Wang. NEAT: Nonlinear parameter-efficient adaptation of pre-trained models, 2024.

Zhan Zhuang, Xiequn Wang, Wei Li, Yulong Zhang, Qiushi Huang, Shuhao Chen, Xuehao Wang, Yanbin Wei, Yuhe Nie, Kede Ma, Yu Zhang, and Ying Wei. Come together, but not right now: A progressive strategy to boost low-rank adaptation. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 80533–80550. PMLR, 2025.

A Complete Procedure and Notation

This appendix follows the same three-stage order as the main method: per-sample update decomposition, spectral basis construction and rank allocation, and conditional LoRA via a prompt-side router. Algorithm 1 summarizes calibration, router warm-up, joint training, and inference. The backbone remains frozen throughout. Each calibration replay is shared across all adapted modules: hooks update module-specific statistics in one forward–backward traversal, after which the statistics are transferred off-device and processed module by module. Table 6 collects the notation used in the main text and this appendix.

Algorithm 1 SGB-LoRA calibration, router warm-up, joint training, and inference

Require: frozen backbone, calibration samples $\{x_i\}_{i=1}^N$, rank budget R per adapted module

1: shared replay: collect module inputs and output gradients through hooks; form or sketch $D_i = -\Delta_i H_i^\top$ for every adapted module — Eq. (2)
2: per module: compute $\bar{D}$, $\widetilde{D}_i$, Gram eigenpairs, modes U_k, energies λ_k, and PCA coordinates a_{ik} — Eqs. (3)–(6)
3: cross-fit isolated-mode or eigenspace-block energy; remove candidates below the predeclared retention ratio — Eq. (33)
4: score mean and residual rank slots, retain the top R, and truncate $T_0, \{T_k\}$ — Eqs. (7), (8)
5: form $\Phi_k = \sqrt{\lambda_k} T_k$; compute ridge coefficients $\beta_i^\star$, standardized targets $z_i^\star$, and scales τ_k — Eqs. (9), (10), (11)
6: factor T_0 and every retained Φ_k into balanced LoRA factors — Eq. (39)
7: set the module scale s_ℓ from output-energy guards — Eq. (41)
8: **router warm-up** ($0 \le t < T_w$): freeze the backbone and LoRA factors; optimize the router with the task and routing losses; ramp the shared branch by κ_t — Eqs. (13), (14)
9: **joint training** ($t \ge T_w$): keep the backbone frozen, remove the routing loss, unfreeze the LoRA factors, and optimize the router and factors for the task objective
10: **inference**: fold the fixed shared branch into W_0; predict residual coordinates once during prefill and cache them for generation

B Per-Sample Update Decomposition

This section expands the first stage of the main method. All exact statements are for one adapted matrix and one fixed calibration set; unless stated otherwise, Gram inner products and SVD truncations are exact.

B.1 Ideal Conditional Reconstruction

The compact main text states only the fixed-update floor. A broader ideal problem clarifies the shared–residual decomposition. Let a shared matrix M and reusable residual matrices E_k reconstruct the per-sample directions with centered signed coordinates:

$$\begin{aligned} \min_{M,\{E_k\},\{\beta_i\}} \quad & \frac{1}{N}\sum_{i=1}^{N}\left\| D_i - M - \sum_{k=1}^{K}\beta_{ik}E_k \right\|_F^2 \\ \text{s.t.} \quad & \operatorname{rank}(M) \le r_0, \quad \operatorname{rank}(E_k) \le r_k, \\ & r_0 + \sum_{k=1}^{K} r_k \le R, \quad \sum_{i=1}^{N}\beta_{ik} = 0 \quad \forall k. \end{aligned} \tag{15}$$

Lemma 1 (Mean–residual decoupling)**.** *Every feasible point of Eq.* (15) *satisfies*

$$\begin{aligned} &\frac{1}{N}\sum_i \left\| D_i - M - \sum_k \beta_{ik}E_k \right\|_F^2 \\ &\quad = \|\bar{D} - M\|_F^2 + \frac{1}{N}\sum_i \left\| \widetilde{D}_i - \sum_k \beta_{ik}E_k \right\|_F^2 . \end{aligned} \tag{16}$$

Table 6: Notation used in the main method and technical appendix.

Symbol	Meaning
$W_0 \in \mathbb{R}^{m\times n}$	backbone weight, frozen throughout
R	rank budget per adapted matrix
N	calibration-set size
$H_i,\ \Delta_i$	module inputs and output gradients of sample i
D_i	per-sample descent direction, Eq. (2)
$\bar{D},\ \widetilde{D}_i$	shared mean and centered residual, Eq. (3)
E_{res}	residual energy, Corollary 1
π_{res}	residual fraction after scalar removal, Eq. (17)
$G,\ (\nu_k, v_k)$	residual Gram matrix and eigenpairs, Eq. (5)
$U_k,\ \lambda_k$	residual mode and its average energy, Eq. (6)
a_{ik}	PCA coordinate used by the allocation reconstruction
K_c	candidate-pool size
$s_{0j},\ s_{kj}$	mean and residual rank-slot scores, Eq. (7)
$T_0,\ T_k;\ r_0,\ r_k$	retained matrices and allocated ranks, Eq. (8)
$\mathcal{K}_\ell,\ K_\ell$	retained residual modes of module ℓ and their count
$\widehat{\lambda}_k,\ \varepsilon_\lambda$	cross-fitted energy and retention threshold
Φ_k	energy-scaled retained basis $\sqrt{\lambda_k}T_k$
$Q,\ c_i$	Gram matrix and sample contractions of the realized basis
$\beta_i^\star$	ridge coefficients on the retained basis
$\tau_k,\ z_{ik}^\star$	coordinate scale and standardized router target
$h_b(x),\ W_1,\ W_2^{(\ell)}$	router descriptor, shared trunk, and module head
$z^{(\ell)}(x)$	signed coordinate vector predicted from the prompt
$B_k,\ A_k$	balanced LoRA factors, Eq. (39)
$b_\ell,\ d_\ell,\ q_\ell$	frozen-output energy, retained-update energy, and ratio
$b_{\mathrm{ref}},\ q_{\mathrm{ref}},\ \varepsilon_b,\ \varepsilon_q$	scale-guard references and thresholds
$s_\ell,\ \gamma$	module scale and target output fraction
$\kappa_t,\ T_w,\ \omega_0$	shared-branch ramp, warm-up length, routing-loss weight
$\mathcal{B}_r,\ \mathcal{M}_+$	router minibatch and active adapted modules
w_i	known sample weight in the weighted extension

For fixed r_0, the optimal shared matrix is $M^\star = [\bar{D}]_{r_0}$.

Corollary 1 (Fixed-matrix reconstruction floor)**.** *Let $E_{\mathrm{res}} = N^{-1}\sum_i \|\widetilde{D}_i\|_F^2$. The best fixed rank-$R$ matrix is $[\bar{D}]_R$, and its error is exactly Eq.* (4)*.*

B.2 Scalar-Variation Diagnostic

Residual energy can be large even when every gradient differs from the mean only by a sample-specific scalar. We therefore report

$$\pi_{\mathrm{res}} = \frac{N^{-1}\sum_i \min_{c\in\mathbb{R}} \|\widetilde{D}_i - c\bar{D}\|_F^2}{E_{\mathrm{res}}}. \tag{17}$$

When $\bar{D} \neq 0$, the minimizer is $c = \langle \widetilde{D}_i, \bar{D}\rangle_F / \|\bar{D}\|_F^2$. We set $\pi_{\mathrm{res}} = 0$ when $E_{\mathrm{res}} = 0$, and $\pi_{\mathrm{res}} = 1$ when $E_{\mathrm{res}} > 0$ but $\bar{D} = 0$.

B.3 Proof of Lemma 1

Let $R_i = \sum_k \beta_{ik} E_k$. Since $D_i = \bar{D} + \widetilde{D}_i$,

$$\frac{1}{N}\sum_i \|D_i - M - R_i\|_F^2 = \|\bar{D} - M\|_F^2 + \frac{1}{N}\sum_i \|\widetilde{D}_i - R_i\|_F^2 + \frac{2}{N}\left\langle \bar{D} - M, \sum_i (\widetilde{D}_i - R_i) \right\rangle_F. \tag{18}$$

By definition, $\sum_i \widetilde{D}_i = 0$. Feasibility gives $\sum_i R_i = \sum_k (\sum_i \beta_{ik}) E_k = 0$, so the cross term vanishes. For fixed r_0, the only term involving M is $\|\bar{D} - M\|_F^2$, and the Eckart–Young–Mirsky theorem gives $M^\star = [\bar{D}]_{r_0}$. The minimizer is unique when the truncation boundary is strict.

Setting every residual basis matrix to zero gives

$$\frac{1}{N}\sum_i \|D_i - M\|_F^2 = \|\bar{D} - M\|_F^2 + \frac{1}{N}\sum_i \|\widetilde{D}_i\|_F^2.$$

Minimizing the first term over rank-R matrices proves Corollary 1.

C Spectral Basis Construction and Rank Allocation

This section establishes the snapshot-PCA identities, proves the EARA allocation result, and then specifies the scalable approximation used for large models.

C.1 Snapshot PCA Identities

Let $q = \operatorname{rank}(G)$, retain the positive eigenpairs, and fix an ordered orthonormal eigenbasis. For $1 \le k, k' \le q$,

$$\langle U_k, U_{k'} \rangle_F = \frac{1}{\sqrt{\nu_k \nu_{k'}}} \sum_{i,j} v_{ik} v_{jk'} \langle \widetilde{D}_i, \widetilde{D}_j \rangle_F \tag{19}$$

$$= \frac{v_k^\top G v_{k'}}{\sqrt{\nu_k \nu_{k'}}} = \delta_{kk'}. \tag{20}$$

Similarly,

$$\langle \widetilde{D}_i, U_k \rangle_F = \frac{1}{\sqrt{\nu_k}} \sum_j v_{jk} G_{ij} = \sqrt{\nu_k} v_{ik} = a_{ik}.$$

Because $G\mathbf{1} = 0$, every positive-eigenvalue eigenvector is orthogonal to $\mathbf{1}$, so $\sum_i a_{ik} = 0$. The matrices $\{U_k\}_{k=1}^q$ form an orthonormal basis for the span of the centered gradients, yielding

$$\widetilde{D}_i = \sum_{k=1}^{q} a_{ik} U_k, \qquad \frac{1}{N}\sum_i a_{ik} a_{ik'} = \lambda_k \delta_{kk'}. \tag{21}$$

C.2 Exact EARA Objective

For Proposition 1, fix the calibration set, an ordered orthonormal eigenbasis of its exact Gram matrix, and exact SVDs of $\bar{D}$ and the candidate modes. The fixed-coordinate reconstruction is

$$\widehat{D}_i = T_0 + \sum_{k=1}^{K_c} a_{ik} T_k, \tag{22}$$

with average squared error

$$\mathcal{E} = \sum_{j>r_0} (\sigma_j^{(0)})^2 + \sum_{k\le K_c} \lambda_k \sum_{j>r_k} (\sigma_j^{(k)})^2 + \sum_{k=K_c+1}^{q} \lambda_k. \tag{23}$$

The proposition states that selecting the largest positive slot scores minimizes this error under $r_0 + \sum_{k\le K_c} r_k \le R$.

C.3 Proof of Proposition 1

Let $R_0 = \bar{D} - T_0$ and $R_k = U_k - T_k$ for $k \le K_c$. Using Eq. (21),

$$\begin{aligned} D_i - \widehat{D}_i &= R_0 + \sum_{k\le K_c} a_{ik} R_k \\ &\quad + \sum_{k>K_c} a_{ik} U_k, \\ \frac{1}{N}\sum_i \|D_i - \widehat{D}_i\|_F^2 &= \|R_0\|_F^2 \\ &\quad + \sum_{k\le K_c} \lambda_k \|R_k\|_F^2 \\ &\quad + \sum_{k>K_c} \lambda_k \|U_k\|_F^2. \end{aligned} \tag{24}$$

The cross terms with R_0 vanish because every coordinate is centered. Cross terms between different residual indices vanish because their empirical coordinate covariance is diagonal. This remains true even though the truncated matrices T_k need not be mutually orthogonal. Since $\|U_k\|_F = 1$ and each T_k is an exact truncated SVD, Eq. (24) is exactly Eq. (23).

Define

$$C = \|\bar{D}\|_F^2 + \sum_{k=1}^{q} \lambda_k.$$

Then

$$\mathcal{E} = C - \sum_{j\le r_0} (\sigma_j^{(0)})^2 - \sum_{k\le K_c} \sum_{j\le r_k} \lambda_k (\sigma_j^{(k)})^2. \tag{25}$$

Minimizing $\mathcal{E}$ is therefore equivalent to retaining slots with maximum total score. Scores are non-increasing within each matrix because singular values are ordered. A top-score selection can consequently be chosen to contain prefixes of each matrix's singular spectrum and induces a feasible rank allocation. Selecting the largest R positive scores, or all positive scores if fewer exist, is optimal. Padding with zero-score slots does not change the objective, which covers the inequality $r_0 + \sum_k r_k \le R$.

C.4 Candidate-Pool Certificate

Let $\lambda_{q+1} = 0$, and let θ_R be the R-th largest candidate score after padding the score list with zeros. For every omitted residual mode $k > K_c$ and every singular slot j,

$$s_{kj} = \lambda_k (\sigma_j^{(k)})^2 \le \lambda_k \le \lambda_{K_c+1},$$

because $\sum_j (\sigma_j^{(k)})^2 = \|U_k\|_F^2 = 1$. Therefore

$$\theta_R \ge \lambda_{K_c+1} \tag{26}$$

implies that no omitted slot can strictly outscore a retained slot. Equality may admit another allocation with the same objective. This argument uses exact normalization of the recovered modes and does not by itself apply to sketched eigenvectors or approximate SVD slots.

C.5 Repeated-Eigenvalue Convention

If a cluster $\mathcal{C}$ has an exactly repeated eigenvalue, its spectral projector $P_{\mathcal{C}} = \sum_{k\in\mathcal{C}} v_k v_k^\top$ is unique although its eigenvectors are not. To remove dependence on the eigensolver's arbitrary rotation, we construct a deterministic basis from this projector. Scan the coordinate vectors $e_1, \ldots, e_N$, project each as $P_{\mathcal{C}} e_j$, remove its components along previously accepted vectors, and accept the normalized residual whenever it is nonzero. Fix the sign by requiring the first nonzero coordinate to be positive. The samples are first placed in a stable order determined by fixed identifiers. In exact arithmetic this returns an orthonormal basis determined by $P_{\mathcal{C}}$ and that fixed order.

This exact-arithmetic convention chooses one representative but does not make EARA rotation invariant. For example, with equal mode energies, $U_1 = E_{11}$ and $U_2 = E_{22}$ each have one unit singular slot, whereas the rotated basis $(E_{11} \pm E_{22})/\sqrt{2}$ has two slots of size $1/2$ per mode. The optimal allocation within the induced family can therefore change under rotation. The convention guarantees reproducibility only for a fixed ordered calibration set. It does not provide continuity, and nearly vanishing Gram–Schmidt pivots can still cause jumps. It is a mathematical convention for an exactly repeated eigenvalue. The floating point implementation does not use a tolerance cluster to rotate eigenvectors, because such a cluster may contain unequal eigenvalues and would then invalidate Eq. (24). Instead, we fix the eigensolver, seed, sample order, and sign convention, and use small eigengaps only for diagnostics. Reproducing a particular allocation requires the recovered eigenvectors, sample order, and numerical settings, not just the eigenvalues. The cross-subset stability results in Section 4.3 do not establish invariance to sample order or numerical implementation.

C.6 Scalable Spectral Calibration

The exact construction clarifies the target geometry. Large models require three additional approximations: estimating the Gram matrix, truncating the recovered matrices, and generalizing from a finite calibration set. The results below do not turn these approximations into unconditional guarantees; they identify the quantities that must be measured.

C.6.1 Factored Inner Products and TensorSketch

Equation (2) gives the exact contraction

$$\langle D_i, D_j \rangle_F = \mathrm{tr}\big[(\Delta_i^\top \Delta_j)(H_j^\top H_i)\big]\,. \tag{27}$$

For large matrices, let $X = [\widetilde{d}_1, \ldots, \widetilde{d}_N]$ so $G = X^\top X$, and let S be a fixed linear TensorSketch (Pham & Pagh, 2013). If S is an ε-subspace embedding for $\mathrm{col}(X)$,

$$(1-\varepsilon)\|Xy\|_2^2 \le \|SXy\|_2^2 \le (1+\varepsilon)\|Xy\|_2^2 \quad \forall y, \tag{28}$$

then

$$(1-\varepsilon)G \preceq \widehat{G} \preceq (1+\varepsilon)G, \qquad \widehat{G} = X^\top S^\top S X,$$

and $\|\widehat{G} - G\|_2 \le \varepsilon \|G\|_2$. Weyl and Davis–Kahan bounds then control eigenvalues and separated invariant subspaces (Avron et al., 2014; Yu et al., 2015). At the sketch dimensions used in the main experiments, worst-case TensorSketch embedding guarantees are not asserted. Direct Gram error and principal-angle comparisons against an exact construction would be needed to validate the embedding approximation; the cross-subset stability results in Section 4.3 test reproducibility rather than exact–sketch agreement.

Unless varied, the sketch width is $p = 2048$. TensorSketch uses two fixed four-wise-independent hashes and Rademacher sign maps for the activation and output-gradient factors, combined by circular convolution. The primary sketch seed is 17, and every seed is stored with the calibration artifact. Randomized SVD uses oversampling eight, two power iterations, and FP32 QR and core decompositions.

Let U_k denote the exact mode. A sketched Gram eigenvector reconstructed against exact factored gradients gives a generally different matrix $\widetilde{U}_k$, and randomized SVD then produces $\widehat{T}_k \approx [\widetilde{U}_k]_{r_k}$. The coefficient vectors $\widehat{v}_k$ diagonalize $\widehat{G}$, and $\widetilde{U}_k$ are their corresponding reconstructed matrix-space modes. They need not be exactly Frobenius-orthonormal under the exact inner product. Consequently, neither Proposition 1 nor the exact pool certificate is automatically inherited by the sketched pipeline.

C.6.2 Randomized SVD and Conditional Slot Stability

To isolate randomized-SVD error from sketch/eigenvector error, let $\widetilde{T}_k = [\widetilde{U}_k]_{r_k}$ be the exact truncation of the reconstructed sketched mode and $\widehat{T}_k$ its randomized-SVD approximation. The corresponding excess reconstruction error is

$$\varepsilon_{\mathrm{svd},k} = \|\widetilde{U}_k - \widehat{T}_k\|_F^2 - \|\widetilde{U}_k - \widetilde{T}_k\|_F^2 \geq 0$$

and is distinct from the sketch/eigenvector discrepancy and the corresponding singular-slot errors. For isolated, reliably matched modes, the latter discrepancy can be quantified by $\|\widetilde{U}_k - U_k\|_F$. For repeated or near-repeated clusters, projector or principal-angle distances are more appropriate. A Gram subspace bound alone is not sufficient to control these errors, particularly inside a nearly repeated eigenvalue cluster.

Proposition 2 (Conditional slot stability)**.** *Suppose exact and approximate slots have been matched and satisfy* $\max_a |\widehat{s}_a - s_a| \leq \varepsilon_{\mathrm{slot}}$*. If the exact boundary obeys* $s_{(R)} - s_{(R+1)} > 2\varepsilon_{\mathrm{slot}}$*, then exact and approximate EARA select the same* R *slots, up to permutations within exact score ties.*

The proof is immediate: every exact selected score remains larger than every exact unselected score after perturbing both by at most $\varepsilon_{\mathrm{slot}}$. This proposition is deliberately conditional. Deriving $\varepsilon_{\mathrm{slot}}$ requires eigenvector, matrix-reconstruction, and SVD perturbation control in addition to a Gram error. Slot overlap across calibration subsets, as reported in Section 4.3, is a separate empirical stability measure and does not certify this exact–approximate score bound.

C.6.3 Finite-Sample Subspace Stability

The following population statement assumes i.i.d. sampling and identifies the quantities governing sample complexity. For calibration samples drawn uniformly without replacement from a finite training corpus, it provides intuition rather than an exact finite-population guarantee. The weighted extension below concerns the empirical reconstruction identities and does not extend this population proposition to fixed-quota sampling.

Proposition 3 (Subspace stability under i.i.d. sub-Gaussian sampling)**.** *Let* $\xi_1, \ldots, \xi_N$*, with* $\xi_i = \mathrm{vec}(D_i - \mathbb{E}D)$*, be i.i.d. and sub-Gaussian relative to their covariance* Σ*: for every vector* u*,* $\|\langle u, \xi_i\rangle\|_{\psi_2} \leq \kappa_{\mathrm{sg}}\sqrt{u^\top \Sigma u}$*. Define* $r_{\mathrm{eff}} = \mathrm{tr}(\Sigma)/\|\Sigma\|_{\mathrm{op}}$*. For every* $t \geq 1$*, there is a universal constant* C *such that, with probability at least* $1 - e^{-t}$*, the sample-mean-centered covariance satisfies*

$$\|\widehat{\Sigma} - \Sigma\|_{\mathrm{op}} \leq \Delta_N := C\kappa_{\mathrm{sg}}^2 \|\Sigma\|_{\mathrm{op}} \left[\sqrt{\frac{r_{\mathrm{eff}} + t}{N}} + \frac{r_{\mathrm{eff}} + t}{N}\right]. \tag{29}$$

If the population eigengap $\delta_r = \lambda_r(\Sigma) - \lambda_{r+1}(\Sigma) > 2\Delta_N$*, then*

$$\|\sin\Theta(\widehat{\mathcal{U}}_{1:r}, \mathcal{U}_{1:r})\|_{\mathrm{op}} \leq \frac{2\Delta_N}{\delta_r}. \tag{30}$$

The covariance bound follows from effective-rank concentration for sample covariance operators, with the sample-mean correction absorbed into the same order, and the subspace bound follows from Davis–Kahan (Koltchinskii & Lounici, 2017; Yu et al., 2015). The leading $N^{-1/2}$ behavior appears only when $N \gg r_{\mathrm{eff}} + t$ and the population eigengap is fixed. Heavy-tailed gradients, small eigengaps, and fixed-quota sampling fall outside this proposition. Moreover, subspace stability does not imply EARA stability inside a repeated or nearly repeated eigenspace. Section 4.3 therefore reports subspace agreement together with aligned slot overlap across calibration subsets. The cross-fitted energies of Eq. (33) identify candidates whose energy does not persist out of fold; EARA still ranks the retained pool by the full-data empirical scores.

C.6.4 Sampling, Cross-Fitted Energies, and Weights

Out-of-fold energies from the uncentered Gram. Let $K_{ij} = \langle D_i, D_j\rangle_F$ be the uncentered inner-product matrix, available in closed form from Eq. (27). Fix a fold f and write A for its complement,

$\bar{D}_A = |A|^{-1} \sum_{j \in A} D_j$, and $\widetilde{D}_j^A = D_j - \bar{D}_A$. For any i and j,

$$\begin{aligned}\left\langle D_i - \bar{D}_A, D_j - \bar{D}_A \right\rangle_F = K_{ij} &- \frac{1}{|A|} \sum_{l \in A} K_{il} \\ &- \frac{1}{|A|} \sum_{l \in A} K_{lj} \\ &+ \frac{1}{|A|^2} \sum_{l,l' \in A} K_{ll'},\end{aligned} \tag{31}$$

so the fold-restricted Gram, its eigenpairs (ν_k^A, v_k^A), and the modes $U_k^{(-f)} = (\nu_k^A)^{-1/2} \sum_{j \in A} v_{jk}^A \widetilde{D}_j^A$ are all determined by K. For a held-out $i \in f$,

$$\begin{aligned}\left\langle D_i - \bar{D}_A, U_k^{(-f)} \right\rangle_F = &\frac{1}{\sqrt{\nu_k^A}} \sum_{j \in A} v_{jk}^A \\ &\times \left\langle D_i - \bar{D}_A, D_j - \bar{D}_A \right\rangle_F,\end{aligned} \tag{32}$$

which is again a linear functional of K. Let $\pi_f(k)$ be the deterministic assignment from an isolated full-data candidate k to a mode learned without fold f. Its cross-fitted energy is

$$\widehat{\lambda}_k = \frac{1}{N} \sum_f \sum_{i \in f} \left\langle D_i - \bar{D}_A, U_{\pi_f(k)}^{(-f)} \right\rangle_F^2. \tag{33}$$

Equation (33) therefore requires no gradient beyond those already contracted into K, and the same is true of the captured-energy stopping rule below. We use $F = 5$ fixed folds generated with seed 17 in all reported runs. Adjacent full-data modes $k, k+1$ whose relative eigengap $(\nu_k - \nu_{k+1})/\nu_1$ is below $\varepsilon_{\mathrm{gap}} = 10^{-3}$ are merged into one block. Isolated fold modes are matched to full-data modes by a maximum-overlap bipartite assignment using absolute Frobenius inner products. Degenerate blocks are matched by principal angles. For a block $\mathcal{C}$, its held-out energy is the average squared norm of the projection of $D_i - \bar{D}_A$ onto the matched fold subspace, and its in-sample energy is $\sum_{k \in \mathcal{C}} \lambda_k$. A block is retained or removed as a whole. Candidates are retained when their held-out-to-in-sample energy ratio is at least $\varepsilon_\lambda = 0.5$. Within the retained pool, EARA uses the full-data scores of Eq. (7); ties are resolved by mean first, then mode index, then singular-slot index.

Known-weight sampling. For completeness, the empirical construction also admits known sampling weights; this is a mathematical extension, not an additional experiment. Let $w_i > 0$ with $\sum_i w_i = 1$ be known sampling weights, put $\bar{D}_w = \sum_i w_i D_i$ and $\widetilde{D}_i = D_i - \bar{D}_w$, and replace Eq. (5) by

$$G_{ij}^w = \sqrt{w_i w_j}\, \langle \widetilde{D}_i, \widetilde{D}_j \rangle_F, \qquad G^w v_k = \nu_k v_k. \tag{34}$$

Define $U_k = \nu_k^{-1/2} \sum_i \sqrt{w_i}\, v_{ik} \widetilde{D}_i$, $a_{ik} = \sqrt{\nu_k / w_i}\, v_{ik}$, and $\lambda_k = \nu_k$. Then $\langle U_k, U_{k'} \rangle_F = (\nu_k \nu_{k'})^{-1/2} v_k^\top G^w v_{k'} = \delta_{kk'}$, and

$$\sum_i w_i a_{ik} a_{ik'} = \sqrt{\nu_k \nu_{k'}} \sum_i v_{ik} v_{ik'} = \lambda_k \delta_{kk'}. \tag{35}$$

The vector with entries $\sqrt{w_i}$ lies in the kernel of G^w, because $(G^w \sqrt{w})_i = \sqrt{w_i} \langle \widetilde{D}_i, \sum_j w_j \widetilde{D}_j \rangle_F = 0$, so every positive-eigenvalue eigenvector is orthogonal to it and $\sum_i w_i a_{ik} = 0$. Each $\widetilde{D}_i$ still lies in the span of the positive-eigenvalue modes, so the expansion $\widetilde{D}_i = \sum_k a_{ik} U_k$ is exact. Replacing $N^{-1} \sum_i$ by $\sum_i w_i$ throughout, Lemma 1, Corollary 1, Proposition 1, and the pool certificate hold verbatim, and $w_i = 1/N$ recovers the unweighted construction with $\nu_k^{\text{unweighted}} = N \nu_k$.

Cross-fitting uses the same target weights. For fold complement A, set $\widetilde{w}_j^A = w_j / \sum_{l \in A} w_l$ and $\bar{D}_A^w = \sum_{j \in A} \widetilde{w}_j^A D_j$, build the fold modes with the renormalized weights, and define

$$\widehat{\lambda}_k^w = \sum_f \sum_{i \in f} w_i \left\langle D_i - \bar{D}_A^w, U_{\pi_f(k)}^{(-f,w)} \right\rangle_F^2.$$

The held-out-to-in-sample retention ratio is then $\widehat{\lambda}_k^w / \lambda_k$; block ratios replace numerator and denominator by their sums over the matched block.

Why we do not select for diversity. Suppose samples are drawn from a proposal q rather than the corpus measure p and the unweighted estimator is used. Then $\widehat{\lambda}_k$ converges to $\mathbb{E}_q[a_k^2]$ rather than $\mathbb{E}_p[a_k^2]$, so a proposal that over-represents rare but distinctive gradients inflates exactly the scores in Eq. (7) that decide allocation, and EARA spends rank on directions few corpus samples use. Importance weights $w_i \propto p_i/q_i$ in Eq. (34) restore consistency, but their variance enters $\widehat{\lambda}_k$ directly, so the estimator degrades as q departs from p. This motivates sampling directly from the training corpus and increasing N, rather than selecting calibration examples for gradient diversity.

D Conditional LoRA via a Prompt-Side Router

This section follows the final stage of the main method: it distinguishes the four coordinate objects, gives the closed-form ridge targets, specifies the router and balanced factorization, and details warm-up, memory, and inference execution.

D.1 Coordinate Roles and Ridge Refitting

The method uses four coordinate objects, each at a different stage:

$$\begin{aligned} a_i &: \text{PCA coordinates used for allocation,} \\ \beta_i^\star &: \text{ridge coefficients on the retained bases,} \\ z_i^\star &: \text{standardized calibration targets,} \\ z^{(\ell)}(x) &: \text{prompt-side coordinate predictions.} \end{aligned}$$

The vector $a_i \in \mathbb{R}^q$ contains the exact coordinates of sample i in the orthonormal PCA modes and is used only by the fixed-coordinate reconstruction and EARA. Once EARA fixes the ranks, the retained residual matrices are energy-scaled as $\Phi_k = \sqrt{\lambda_k} T_k$, and their truncation generally destroys mutual orthogonality. We therefore replace the PCA coordinates with ridge coefficients $\beta_i^\star \in \mathbb{R}^{K_\ell}$, then standardize them as $z_i^\star$. The deployed router predicts a vector $z^{(\ell)}(x) \in \mathbb{R}^{K_\ell}$ of the same dimension from the prompt alone. Thus a_i supports rank allocation, $\beta_i^\star$ supports the realized low-rank reconstruction, $z_i^\star$ is a calibration-time label, and $z^{(\ell)}(x)$ is the trainable prediction used in the forward pass.

For the retained modes of module ℓ, define

$$Q_{kk'} = \langle \Phi_k, \Phi_{k'} \rangle_F, \qquad c_{ik} = \langle \widetilde{D}_i, \Phi_k \rangle_F. \tag{36}$$

The ridge problem in Eq. (10) has the closed-form solution

$$\beta_i^\star = (Q + \eta I)^{-1} c_i, \qquad \eta = \eta_0 \frac{\operatorname{tr}(Q)}{\max(K_\ell, 1)}, \quad \eta_0 > 0. \tag{37}$$

If $K_\ell = 0$, the module has no residual branch and contributes neither a router head nor a routing-loss term.

Since $\sum_i \widetilde{D}_i = 0$, Eq. (36) gives $\sum_i c_i = 0$, and hence

$$\sum_i \beta_i^\star = (Q + \eta I)^{-1} \sum_i c_i = 0.$$

For the unregularized reconstruction diagnostic, $\beta_i^{\mathrm{LS}} = Q^\dagger c_i$ is centered for the same reason. The plug-in coefficients $a_{ik}/\sqrt{\lambda_k}$ are feasible on the same realized matrices, so unregularized least squares cannot have larger reconstruction error than the plug-in choice. The deployed ridge target uses $\eta > 0$ and instead minimizes the regularized objective.

Standardization preserves centering and changes only the parameterization:

$$\begin{aligned} z_{ik}^\star \tau_k B_k A_k &= \frac{\beta_{ik}^\star}{\tau_k} \tau_k \Phi_k \\ &= \beta_{ik}^\star \Phi_k. \end{aligned}$$

The scales τ_k are fixed calibration statistics, not router outputs or learned mixture probabilities.

D.2 Out-of-Sample Coordinate Diagnostics

For the gradient D of a held-out sample, exact-mode ridge coordinates with $\eta > 0$ require

$$c(D)_k = \langle D - \bar{D}, \Phi_k \rangle_F,$$
$$\beta^\star(D) = (Q + \eta I)^{-1} c(D).$$

This contraction can be evaluated from factored gradients without materializing D. Under sketching, extending inner products to $D - \bar{D}$ requires the embedding property to hold on the augmented span $\mathrm{span}(\mathrm{col}(X), D - \bar{D})$. It is not implied by an embedding guarantee on $\mathrm{col}(X)$ alone.

If Q is fixed and $\widehat{c}$ is an approximate contraction, then

$$\|\widehat{\beta} - \beta\|_2 \le \|(Q + \eta I)^{-1}\|_2 \|\widehat{c} - c\|_2 \le \frac{\|\widehat{c} - c\|_2}{\eta}. \tag{38}$$

Standardization can further amplify coordinate error by $1/\min_k \tau_k$. Thus small ridge and nearly constant coordinates need special care. Held-out router R^2 does not validate the accuracy of an approximate oracle target: a router can predict a biased target consistently. Direct exact–sketch coordinate comparisons would be needed to quantify this additional error. The held-out prediction results in Section 4.3 assess agreement with the constructed targets, not an independent guarantee of exact-gradient reconstruction.

D.3 Prompt Descriptor and Router Parameterization

Let $\mathcal{P}(x)$ be the non-padding positions visible before response generation. For decoder-only models and encoder-side modules, we use the input state $r_{b,p}(x)$ to block b. For decoder and cross-attention modules in an encoder–decoder model, we instead use the final encoder output at source position p, so the descriptor remains available before target generation. We compute

$$\bar{h}_b(x) = \frac{1}{|\mathcal{P}(x)|} \sum_{p \in \mathcal{P}(x)} r_{b,p}(x),$$
$$h_b(x) = \frac{\bar{h}_b(x)}{\max(\|\bar{h}_b(x)\|_2, \epsilon_h)}.$$

We use $\epsilon_h = 10^{-6}$ and set $h_b(x) = 0$ when $\mathcal{P}(x) = \varnothing$. The descriptor is detached before the router trunk, computed online in the same forward pass, and cached after prefill for autoregressive generation. The trunk W_1 is shared across blocks with the same hidden width, while $W_2^{(\ell)}$ is an unconstrained linear head for module ℓ; no softmax or nonnegativity constraint is applied because the spectral coordinates are signed.

D.4 Balanced Factors and Function-Preserving Initialization

For any retained matrix $\Psi = U\Sigma V^\top$, define the balanced factors

$$B = U\Sigma^{1/2}, \qquad A = \Sigma^{1/2} V^\top. \tag{39}$$

They satisfy

$$BA = U\Sigma V^\top = \Psi,$$
$$\|B\|_F^2 = \|A\|_F^2 = \mathrm{tr}(\Sigma),$$

so they realize the retained matrix exactly while assigning equal Frobenius norm to the two factors.

At $t = 0$, $\kappa_0 = 0$ and every router head is zero. Consequently, Eq. (13) gives $\Delta W_0^{(\ell)}(x) = 0$ for every input without modifying or compensating W_0. At an arbitrary step, subadditivity of matrix rank gives

$$\mathrm{rank}(\Delta W_t^{(\ell)}(x)) \le \mathrm{rank}(B_0 A_0) + \sum_k \mathrm{rank}(B_k A_k)$$
$$\le r_0 + \sum_k r_k \le R.$$

D.5 Calibration Scale

Restoring the module index, define

$$D_i^{\mathrm{ret},(\ell)} = T_0^{(\ell)} + \sum_{k \in \mathcal{K}_\ell} \beta_{ik}^{\star,(\ell)} \Phi_k^{(\ell)},$$
$$b_\ell = \sum_i \|W_0^{(\ell)} H_i^{(\ell)}\|_F^2,$$
$$d_\ell = \sum_i \|D_i^{\mathrm{ret},(\ell)} H_i^{(\ell)}\|_F^2. \tag{40}$$

If no b_ℓ is positive, all adapters are disabled. Otherwise, let b_{ref} be their positive median and set $q_\ell^2 = d_\ell / b_\ell$ when $b_\ell \geq \varepsilon_b^2 b_{\mathrm{ref}}$, with $q_\ell = 0$ otherwise. If no positive q_ℓ survives, all adapters are disabled. Otherwise, with q_{ref} the positive median,

$$s_\ell = \begin{cases} \gamma / q_\ell, & q_\ell \geq \varepsilon_q q_{\mathrm{ref}}, \\ 0, & \text{otherwise.} \end{cases} \tag{41}$$

For a module that passes both guards, $q_\ell^2 = d_\ell / b_\ell$ and Eq. (41) gives $s_\ell = \gamma / q_\ell$. Hence

$$\sum_i \|s_\ell D_i^{\mathrm{ret}} H_i\|_F^2 = s_\ell^2 d_\ell = \frac{\gamma^2}{q_\ell^2} d_\ell = \gamma^2 b_\ell.$$

If either guard fails, $s_\ell = 0$ and the module is omitted from both the forward update and the routing loss. A common positive rescaling of every per-sample gradient multiplies q_ℓ and q_{ref} by the same factor, so the relative q_ℓ guard is unchanged. Because $\eta \propto \mathrm{tr}(Q)$, the ridge coordinates are unchanged, while D_i^{ret} scales with the gradients and s_ℓ scales inversely. Thus, conditional on the same selected modes, the exact calibrated adapter output is invariant to this common positive rescaling.

D.6 Replay Calibration and Training State

The gradient-matrix-free implementation uses four deterministic shared passes over the calibration set:

1. collect per-sample gradient sketches;
2. reconstruct matrix-free randomized ranges for the mean and current residual candidates;
3. recover small SVD cores, singular slots, and the EARA allocation;
4. compute realized ridge targets, coordinate scales, and module output scales.

These are shared model passes, not per-module replays. In each pass, one forward–backward traversal updates the hooks of all adapted modules. The resulting module statistics are transferred off-device, and the spectral algebra is then performed one module at a time. The number of backbone replays therefore depends on these four stages and any candidate-pool expansion, not on the number of adapted matrices. Processing one module at a time requires $O(Np + N^2)$ stored scalars for its sketches and Gram matrix, in addition to the small randomized-SVD cores.

Expanding the candidate pool requires two additional shared reconstruction passes. In approximate mode, K_c doubles after each expansion. Expansion stops when the selected slot identities and per-mode ranks agree across two successive pool sizes, or at $K_c = N - 1$. Modules that reach the maximum without stability are marked uncertified.

Dropout is disabled, sample order and random seeds are fixed, and calibration takes no optimizer step. Each per-sample loss is normalized by its number of valid target tokens before micro-batch aggregation, so response length does not determine the gradient scale and the hooks retain a separate Δ_i for every sample.

During $0 \leq t < T_w$, calibration samples are replayed and their router descriptors are recomputed online from the current block inputs. The descriptors are detached before the router trunk, while the

backbone and calibrated LoRA factors remain frozen. The expectation in Eq. (14) is implemented as

$$\mathcal{L}_{\text{route}} = \frac{1}{|\mathcal{B}_r|} \sum_{i \in \mathcal{B}_r} \frac{1}{|\mathcal{M}_+|} \sum_{\ell \in \mathcal{M}_+} \frac{1}{K_\ell} \left\| z^{(\ell)}(x_i) - z_i^{\star,(\ell)} \right\|_2^2, \tag{42}$$

with $\mathcal{L}_{\text{route}} = 0$ when $\mathcal{M}_+$ is empty. The routing loss updates only the router, and the task loss updates the router and any explicitly declared task head. Router-supervision batches contain 32 calibration samples, the shared trunk uses Xavier-uniform initialization, and every module-specific head starts at zero.

At $t = T_w$, the routing loss is removed and the LoRA factors are unfrozen. The backbone remains frozen, while the router, LoRA factors, and any task head are optimized jointly for $\mathcal{L}_{\text{task}}$. Because this stage can change both the basis matrices and their routing, all exact spectral and reconstruction statements refer to the calibrated state rather than the final checkpoint.

D.7 Training Memory and Inference Execution

No stage performs full-parameter fine-tuning. During calibration, the backbone is used for forward and backward contractions but has no parameter gradients or optimizer state. During router warm-up, gradients are propagated through the frozen backbone so that the task loss can reach the router, but only the router and an optional task head are updated. After warm-up, the trainable state consists of the LoRA factors, router, and optional task head. Peak device memory is therefore governed by the frozen model weights, activations for one training micro-batch, and the small trainable state; the $O(Np+N^2)$ calibration statistics are kept off-device and processed one module at a time. We report measured peak memory rather than equating frozen backpropagation with inference.

After training, the input-independent branch can be folded into the frozen weight:

$$W_{\text{shared}}^{(\ell)} = W_0^{(\ell)} + s_\ell B_0 A_0.$$

The conditional residual cannot be merged into one global matrix because its coefficients depend on the prompt. It can, however, be executed as one fused low-rank operation. Let vcat and hcat denote vertical and horizontal concatenation. Define

$$A_{\text{res}}^{(\ell)} = \mathrm{vcat}_{k \in \mathcal{K}_\ell}(A_k),$$
$$B_{\text{res}}^{(\ell)}(x) = s_\ell \, \mathrm{hcat}_{k \in \mathcal{K}_\ell} \left(z_k^{(\ell)}(x) \tau_k B_k \right).$$

Then

$$B_{\text{res}}^{(\ell)}(x) A_{\text{res}}^{(\ell)} = \sum_{k \in \mathcal{K}_\ell} s_\ell z_k^{(\ell)}(x) \tau_k B_k A_k,$$
$$\mathrm{rank}\left(B_{\text{res}}^{(\ell)}(x) A_{\text{res}}^{(\ell)} \right) \le \sum_k r_k \le R - r_0.$$

For autoregressive models, each block computes its prompt descriptor and coordinates during prefill and caches them for all subsequent decoding steps. We therefore distinguish the one-time router and prefill cost from the unmerged residual low-rank cost during decoding.

E Interpretation of the Reported Diagnostics

The task-specific settings and reported results are defined in Section 4. This appendix supplies their mathematical and implementation context; it does not introduce a separate evaluation protocol.

Compressibility and stability. Figure 3(a) distinguishes residual energy from directional variation after removing sample-specific scaling of the mean. Its captured energy C_R combines mode energy with matrix-rank compressibility under the shared rank budget. Panel (b) measures reproducibility across calibration subsets using aligned slot overlap and subspace agreement. Neither statistic alone establishes that a retained mode is predictable from the prompt.

Routing endpoints. Figure 3(c) holds the basis, factors, and task model fixed while substituting constant, prompt-descriptor-group, predicted, or response-gradient coordinates. Group coordinates are calibration-set means for deployment-visible prompt clusters; the predicted coordinates use the prompt-side router. The response-gradient endpoint is diagnostic only: it requires the held-out response and is not a deployable method or a guaranteed upper bound on downstream task accuracy. Coordinate R^2 and Pearson correlation concern the leading standardized coordinate, rather than the full routed update.

Scope of the controls. The attention-only diagnostics and ablations in Sections 4.3 and 4.4 are distinct from the all-linear instruction-tuning comparison in Section **??**. Basis-changing ablations refit their coordinate and output scales, while the task-only router shares the full method's basis and targets. Their single-seed task differences are mechanism evidence, not estimates of variation across training seeds. These controls also do not establish the exact–sketch error bounds discussed above.

F Baseline Comparability and Published-Result Provenance

Tables 1 and 2 distinguish published baselines, marked with †, from our measured SGB-LoRA results. The commonsense baseline sources are CoTo and NEAT v1, and the instruction-tuning baselines are taken from LoRA-Pro Table 2, as specified in the corresponding main-text settings. These are source-aligned task-quality comparisons, not claims that every baseline was rerun in one common pipeline. Differences in implementation and evaluation remain limitations of comparisons with published values.

Table 5, in contrast, reports single-run systems measurements in the controlled attention-only setting on one NVIDIA B200. Its hardware and module configuration should not be attributed to the published task-quality rows. Equal factor-rank budgets also do not imply equal total trainable parameters: SGB-LoRA includes an additional router, whose parameter and runtime costs are included in the efficiency table.