GLOBAL FACTORIZED ADAPTATION: EFFICIENT PARAMETER SHARING FOR FINE-TUNING LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Low-rank adaptation splits the trainable budget across modules and constrains the rank of every module update, and existing sharing methods typically keep this module-wise structure or rely on fixed expansion maps. We study Global Factorized Adaptation (GFA), the simplest configuration of the SuperLoRA design space (one group, a matrix reshape, and no projection), which instead imposes low-rank structure on the model-wide update: GFA arranges the coordinates of all target updates into a near-square matrix, learns a single low-rank factorization of it, and reconstructs each module update by slicing and reshaping. This layout minimizes the parameter count for a fixed global rank and makes the budget about $\sqrt{N}$ times finer than that of uniform rank-one LoRA over N equally shaped modules. We give an arithmetic layout condition, independent of the module order and satisfied by our layouts, under which the module updates are generically full rank even at global rank one, and we show that on both evaluated models, GFA and LoRA are far from containing each other at matched budgets. Against six baselines on two language models, six tasks, and five matched budgets, GFA improves over LoRA by 0.86 percentage points on average, with the largest gains at small budgets. With 7.6 to $8.8\times$ fewer parameters than rank-one LoRA, it outperforms partial-module LoRA of the same size by 5.8 to 6.2 percentage points on both models, and its training cost stays close to that of LoRA.

1 INTRODUCTION

Low-rank adaptation (LoRA) (Hu et al., 2022) reduces the cost of fine-tuning a pretrained model by representing the update to each target weight matrix as a separate low-rank product. This parameterization is simple, effective, and widely applicable, but it also determines how adaptation capacity is distributed. The trainable budget is divided among individual modules, and every module update is independently constrained to rank at most r_ℓ. These restrictions are imposed by the parameterization rather than inferred from the downstream task.

There are several reasons to reconsider module-wise allocation. Learned LoRA updates can have stable ranks far below their algebraic rank (Lion et al., 2025), while suitable ranks vary across modules and layers (Zhang et al., 2023). Parameter-sharing methods further suggest that useful adaptation structure extends beyond individual weight matrices. VeRA (Kopiczko et al., 2024), Tied-LoRA (Renduchintala et al., 2024), ShareLoRA (Song et al., 2024), BSLoRA (Zhou et al., 2025), E^{2}LoRA (Li et al., 2026), VB-LoRA (Li et al., 2024), and SingLoRA (Bensaïd et al., 2025) reduce redundancy through different forms of factor sharing while retaining module-wise constructions. More recent methods generate model-wide adaptations from a shared representation. Uni-LoRA (Li et al., 2025) expands a shared vector into LoRA factors, while GPart (Mandica et al., 2026) maps it directly to weight updates; both use fixed random transformations. SuperLoRA (Chen et al., 2024) instead factorizes reshaped groups of updates, but focuses on vision models and does not analyze rank. These developments raise a more fundamental question:

If low-rank structure is imposed globally across all adapted weights through learned global factors, rather than independently within every module, what rank do the module updates retain, and does this improve fine-tuning of large language models at matched parameter budgets?

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.

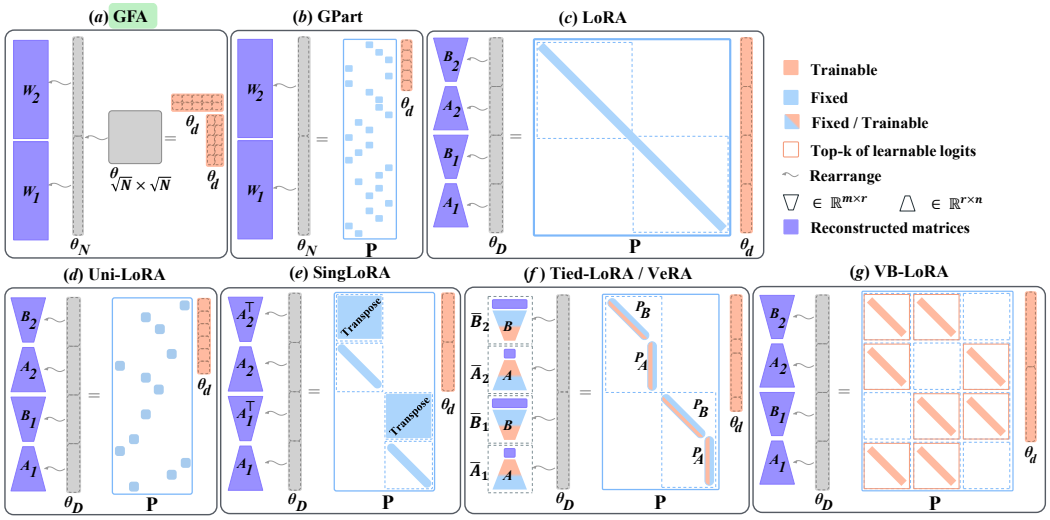

Figure 1: **Comparison of module-wise and model-wide adaptation.** LoRA learns an independent low-rank update for every target module. Existing sharing methods reuse components across module-wise factors or generate updates from a shared vector through fixed expansion maps. GFA instead learns a single global low-rank matrix and reconstructs every module update via reshaping.

We answer this question with Global Factorized Adaptation (GFA), a model-wide parameterization that learns all target updates through a single low-rank matrix, which is the simplest configuration of SuperLoRA's design space: one group, a matrix reshape, and no projection, so a single learned factorization is shared by all modules. Let $D = \sum_{i=1}^{N} p_i q_i$ denote the total number of adapted weight coordinates. GFA arranges these coordinates into a near-square matrix $M \in \mathbb{R}^{m \times k}$, where $mk \geq D$, and parameterizes it as $M = BA$ with $B \in \mathbb{R}^{m \times r_g}$ and $A \in \mathbb{R}^{r_g \times k}$, where r_g is the global rank.

Each module update is then obtained by taking its assigned coordinates from $\mathrm{vec}(M)$ and reshaping them to the dimensions of the corresponding weight matrix. The slicing and reshaping operations are deterministic, and only the global factors B and A are trained.

This change moves the rank constraint from the individual module updates to their global arrangement. Although M has rank at most r_g, a slice of $\mathrm{vec}(M)$ does not generally remain rank r_g after being reshaped into a module update. For the models used in our experiments, we prove and certify that the reconstructed updates are generically full rank, even at $r_g = 1$. The global rank still couples the module updates, so GFA cannot represent arbitrary collections of full-rank updates. GFA has $r_g(m + k)$ trainable parameters, which the near-square layout minimizes for fixed r_g and D.

We evaluate GFA on Qwen2.5-3B-Instruct and Llama-3.2-3B-Instruct across six reasoning and language-understanding tasks, five parameter budgets anchored to LoRA ranks $r_\ell \in \{1, 2, 4, 8, 16\}$, and three seeds per configuration. Across the resulting 60 model-task-budget cells, GFA improves over per-module LoRA by an average of 0.86 percentage points. When compared with the strongest of the six baselines, selected separately in each cell, the mean margin narrows to 0.47 points, showing that GFA is competitive with the strongest sharing methods rather than universally dominant. The advantage is largest at small budgets, and GFA can go below the smallest budget that uniform LoRA can reach: with $26{,}067$ parameters on Qwen and $37{,}541$ on Llama, $8.8\times$ and $7.6\times$ fewer than rank-one LoRA, it outperforms partial-module LoRA of the same size on every task, by 5.8 to 6.2 points on average. Our main contributions are:

- We formalize GFA, the simplest configuration of SuperLoRA's design space, and provide a block-and-crop implementation that keeps training cost close to that of LoRA.
- We conduct an extensive evaluation and ablation study across two models, six tasks, and five parameter budgets, comparing against six baselines, including budgets below the rank-one LoRA.
- We bound the rank of the reconstructed module updates, give an arithmetic layout condition under which they are generically full rank for every module order, show that GFA and LoRA are far from containing each other at matched budgets, and provide a convergence analysis.

2 Related Work

Parameter-efficient fine-tuning (PEFT) methods differ in where adaptation is defined and how it maps to model weights. LoRA and many variants learn module-specific parameters, while others share components across adapters. A separate class defines adaptation globally or factorizes a reshaped model-wide update. We organize related work along these distinctions and place GFA in the latter group, using model-wide matrix factorization with deterministic slicing and reshaping.

Module-wise low-rank and high-rank adaptation. LoRA (Hu et al., 2022) parameterizes the update to each target weight matrix independently as a product of two low-rank factors. Subsequent methods improve different aspects of this construction while retaining separate parameterized module updates. AdaLoRA (Zhang et al., 2023) dynamically reallocates the rank based on the estimated importance of different components. QLoRA (Dettmers et al., 2023) reduces memory consumption by combining LoRA with a quantized frozen model, while LoRA+ (Hayou et al., 2024) assigns different learning rates to the two factors. DoRA (Liu et al., 2024) decomposes pretrained weights into magnitude and direction and applies low-rank adaptation to the directional component.

Other methods relax the module-wise low-rank constraint itself. MoRA (Jiang et al., 2024) learns a square matrix placed between non-parameterized compression and decompression operators to produce high-rank updates within a LoRA-scale parameter budget. RandLoRA (Albert et al., 2025) constructs full-rank updates from learned combinations of fixed random low-rank bases. These approaches increase the attainable rank of individual module updates, but do not replace the collection of module-level adaptations with a single learned representation of the complete model update.

Cross-module sharing within LoRA. A complementary line of work reduces adapter cost by sharing components across modules. VeRA (Kopiczko et al., 2024) shares a pair of frozen random low-rank matrices across layers and learns module-specific scaling vectors. Tied-LoRA (Renduchintala et al., 2024) studies sharing LoRA factors across layers while selectively freezing or training them, and ShareLoRA (Song et al., 2024) shares one or both factors across layers. BSLoRA (Zhou et al., 2025) combines module-specific and shared low-rank terms, adapting shared factors to different module shapes through slicing and gating. E^2LoRA (Li et al., 2026) partitions consecutive layers into sharing intervals based on the similarity of their gradient-based proxy entropies and allocates ranks to the intervals according to entropy. VB-LoRA (Li et al., 2024) constructs module-specific factors from a shared, trainable vector bank using sparse mixtures, while SingLoRA (Bensaïd et al., 2025) couples the two factors via transposition to reduce the trainable parameters.

Although these methods differ in what they share and optimize, they ultimately construct a low-rank adapter for each module, whose rank is bounded by the adapter rank. Their sharing mechanisms therefore reduce the trainable parameters without removing the module-wise low-rank constraint.

Global-subspace adaptation. Intrinsic-dimension studies established that pretrained language models can be adapted by optimizing a small vector projected into the full parameter space (Aghajanyan et al., 2021), providing an early example of model-wide adaptation in a low-dimensional subspace. Uni-LoRA (Li et al., 2025) applies a related global perspective to LoRA by projecting a single trainable vector into the parameters of all module-wise adapters. This enables model-wide sharing, but the reconstructed parameters still define LoRA factors, keeping rank constraints per module. GPart (Mandica et al., 2026) instead maps a trainable vector directly into the full model-update space through an isometric projection, thereby removing the module-wise low-rank constraint.

Unlike Uni-LoRA and GPart, which expand a trained vector through a fixed projection, GFA learns the factors of the matricized update, so the low-rank constraint applies to the global matricization rather than to each module, and GFA produces generically full-rank module updates without fixed random bases, a fixed expansion map, or a separate factorization for each module.

Global reshaping and factorization. SuperLoRA (Chen et al., 2024) defines the closest design space: it concatenates the updates of multiple layers, splits them into G groups, reshapes each group into a matrix or higher-order tensor, factorizes it, and reads each module update back from the vectorized group. When evaluated on vision tasks, it finds that the best setting depends on the budget, but it does not characterize the rank that individual module updates retain after reshaping.

GFA is the simplest configuration in this space, with one group, a matrix reshape, no projection, no group boundaries, and no random component. We contribute two-sided bounds on the rank each module retains, which yield an arithmetic layout condition for generic full rank (Section 4).

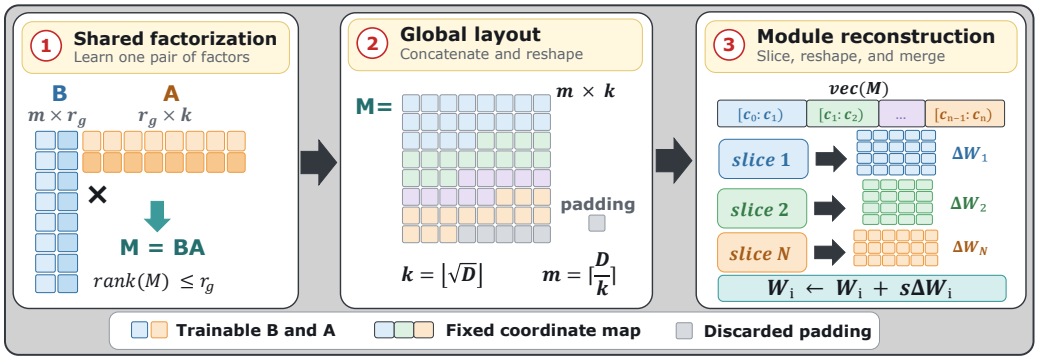

Figure 2: **Overview of Global Factorized Adaptation.** GFA represents all target-weight update coordinates through a single global factorization $M = BA$ and maps contiguous segments of $\mathrm{vec}(M)$ back to their corresponding module shapes. Although the conceptual construction is expressed through the global matrix M, training reconstructs only the rows intersected by each module.

3 METHOD

Global Factorized Adaptation (GFA) replaces the independent low-rank factorization assigned to each adapted module by LoRA with a single factorization shared across all target weights. GFA first assigns every adapted weight coordinate to a fixed position in a model-wide vector. This vector is represented as a near-square matrix and parameterized by one low-rank product. Contiguous segments of the resulting matrix are then reshaped into the individual module updates. The factorization is global, and the reconstructed update applied to a module is not constrained to have the global factorization rank. Figure 2 illustrates this construction, and Appendix A summarizes the two reconstruction paths used by GFA in Algorithm 1 as pseudocode.

Problem setup. Let $\{W_i \in \mathbb{R}^{p_i \times q_i}\}_{i=1}^{N}$ denote the frozen weight matrices selected for adaptation. Define $d_i = p_i q_i$ and $D = \sum_{i=1}^{N} d_i$, where D is the total number of coordinates across all target matrices. We fix an ordering of the target modules and define cumulative offsets $c_0 = 0$ and $c_i = \sum_{j=1}^{i} d_j$. Module i is therefore assigned the half-open coordinate interval $[c_{i-1}, c_i)$.

For comparison, LoRA parameterizes each module independently as $\Delta W_i^{\mathrm{LoRA}} = \frac{\alpha}{r_\ell} B_i A_i$, where $B_i \in \mathbb{R}^{p_i \times r_\ell}$ and $A_i \in \mathbb{R}^{r_\ell \times q_i}$. Its total number of trainable adapter parameters is $r_\ell \sum_{i=1}^{N}(p_i + q_i)$, and r_ℓ denotes the local LoRA rank. We distinguish it from the global rank r_g used by GFA below.

3.1 GLOBAL UPDATE PARAMETERIZATION

GFA represents all D update coordinates using a matrix $M \in \mathbb{R}^{m \times k}$, where $mk \geq D$. Rather than optimizing M directly, we parameterize it as $M = BA$, where $B \in \mathbb{R}^{m \times r_g}$, $A \in \mathbb{R}^{r_g \times k}$, and r_g is the global factorization rank. Let $\mathrm{vec}(M)$ denote the row-major vectorization of M; slices $[\cdot]_{a:b}$ are 0-indexed and half-open. Only the first D coordinates correspond to adapted weights, and the remaining $mk - D$ coordinates are unused padding. The update applied to module i therefore is

$$\Delta W_i(B, A) = \mathrm{reshape}_{p_i \times q_i}\left([\mathrm{vec}(BA)]_{c_{i-1}:c_i}\right). \tag{1}$$

The effective weight used during the forward pass is therefore $W_i^{\mathrm{eff}} = W_i + s\,\Delta W_i(B, A)$, where s is a fixed update scale. Equation (1) defines the complete GFA parameterization. The mapping from BA to the module updates is deterministic and contains no learned projection, random expansion matrix, or module-specific parameters. Only the two global factors B and A are optimized.

The ordering of modules and coordinates is part of the parameterization. We use the model's fixed module traversal order and row-major vectorization in all experiments. We do not change this ordering across tasks or seeds. For our layouts, the full-rank guarantee of Theorem 4.1 holds for every module order (Section H).

Near-square global layout. The number of trainable parameters is $P_{\mathrm{GFA}} = r_g(m+k)$. For fixed D and r_g, minimizing this parameter count amounts to minimizing $m + k$ subject to $mk \geq D$. By the arithmetic-geometric mean inequality, we have $m+k \geq 2\sqrt{mk} \geq 2\sqrt{D}$. This motivates a near-square layout, so we use $k = \lfloor \sqrt{D} \rfloor$ and $m = \lceil D/k \rceil$, which attains the minimum up to rounding.

Given a target adapter budget P, the largest feasible global rank is set to $r_g = \lfloor P/(m+k) \rfloor$. In our budget-matched experiments, r_g is selected so that P_{GFA} does not exceed the corresponding LoRA budget anchor. Therefore, the resulting r_g should not be directly compared with the per-module r_ℓ.

Each unit of global rank then costs $m+k < 2\sqrt{D}+2$ parameters, whereas uniform rank-one LoRA costs $\sum_i (p_i + q_i)$. For N equally shaped modules, their ratio is at least $\sqrt{N}\sqrt{D}/(\sqrt{D}+1) \approx \sqrt{N}$ (Proposition G.1), and for our q/v scopes it is 8.84 on Qwen ($N = 72$) and 7.64 on Llama ($N = 56$), so GFA can operate far below the smallest budget of uniform LoRA.

Rank of the reconstructed module updates. Although the global matrix satisfies $\mathrm{rank}(BA) \leq r_g$, this constraint does not generally carry over to the reconstructed ΔW_i. A module receives a contiguous segment of $\mathrm{vec}(BA)$, which may cross multiple rows of the global matrix before being reshaped into a different two-dimensional geometry. Consequently, ΔW_i is neither a submatrix of BA nor, in general, a product of two matrices with inner dimension r_g.

For the near-square Qwen and Llama layouts used in our experiments, all reconstructed module updates are generically full rank even when $r_g = 1$ (Section 4). This is the principal structural difference from LoRA and methods that preserve a rank-r_ℓ update inside every module.

Optimization and initialization. We initialize $A_0 = 0$ and B_0 with Kaiming-uniform entries (He et al., 2015) using fan-in r_g. The initial global update is therefore $B_0 A_0 = 0$, so the pretrained model is unchanged at the beginning of fine-tuning. Because $B_0 \neq 0$ almost surely, gradients can flow into A on the first optimization step. Both factors are subsequently optimized jointly with the downstream objective.

The rank r_g, target-module scope, global width k, update scale s, and initialization are explicit design choices. At the five parameter count anchors, we hold the q/v target scope, near-square layout, row-major ordering, $s = 1$, and initialization fixed across tasks. Only the budget, expressed through r_g, varies across budget anchors.

Efficient reconstruction. Equation (1) defines GFA through the global product BA, but materializing this complete matrix is not required during training. For module i, define the first and exclusive-last rows of BA intersected by its assigned coordinate interval as

$$u_i = \left\lfloor \frac{c_{i-1}}{k} \right\rfloor, \qquad v_i = \left\lceil \frac{c_i}{k} \right\rceil, \qquad \ell_i = c_{i-1} - u_i k, \tag{2}$$

where ℓ_i is the offset of the first assigned coordinate within row u_i. During training, we materialize only the intersecting row block

$$\widetilde{M}_i = B_{u_i:v_i,:} A \in \mathbb{R}^{(v_i - u_i)\times k}, \tag{3}$$

and reconstruct the module update as

$$\Delta W_i(B, A) = \mathrm{reshape}_{p_i \times q_i}\big([\mathrm{vec}(\widetilde{M}_i)]_{\ell_i:\ell_i+d_i}\big). \tag{4}$$

This reconstruction is equivalent to Equation (1), since $\widetilde{M}_i$ contains every row of BA that intersects $[c_{i-1}, c_i)$, and the subsequent slice removes the coordinates outside this interval. The local product contains fewer than $d_i + 2k$ entries, independent of the size of the full global matrix.

An alternative computes $M = BA$ once and obtains module updates as views of contiguous coordinate intervals. This is efficient without gradient tracking, but during training all updates share the same global tensor, increasing backward cost and peak memory. We therefore use local block-and-crop reconstruction in Equations (3) and (4) during training and the global product only for no-gradient evaluation or one-time weight merging.

Both reconstruction paths implement the same deterministic mapping from (B, A) to $\{\Delta W_i\}_{i=1}^N$ and produce the same model outputs and parameter gradients up to numerical precision. Their leading-order arithmetic cost is similar, and the practical difference comes from recomputed boundary rows, kernel launches, and autograd accumulation into the shared global tensor.

After fine-tuning, each update is merged, and the global factors and reconstruction operations are absent from the model. Therefore, GFA introduces no additional computation at inference.

Table 1: **Theory at the deployed layouts.** Quantities behind Theorem 4.1 and Proposition 4.2 for the q/v scopes. The separation row gives lower bounds on the squared distance from the GFA witness to the LoRA family and from the LoRA witness to the GFA family.

	Qwen2.5-3B	Llama-3.2-3B
Modules N and coordinates D	72, 169,869,312	56, 352,321,536
Global layout $m \times k$	$13{,}034 \times 13{,}033$	$18{,}771 \times 18{,}770$
$\gcd(k, q_i)$ and period h_i	1, 13,033	2, 9,385
Largest $\min(p_i, q_i)$	2,048	3,072
Budget ratio $\sum_i (p_i + q_i)/(m + k)$	8.84	7.64
Separation: GFA to LoRA, LoRA to GFA	≥ 0.87, ≥ 0.51	≥ 0.94, ≥ 0.72

4 Theoretical Analysis

GFA constrains the rank of the global matrix $M = BA$, while the update applied to each module is obtained only after slicing and reshaping its assigned coordinates. We ask two questions: which layouts let the module updates escape the global rank constraint, and how far GFA and LoRA are from containing each other at a matched budget. We summarize the answers here; precise statements, lemmas, certificates, and proofs are in Appendices H–J.

Rank after slicing and reshaping. Let $\rho_i(r_g)$ be the largest rank of $\Delta W_i(B, A)$ over all $B \in \mathbb{R}^{m \times r_g}$ and $A \in \mathbb{R}^{r_g \times k}$. Every minor of $\Delta W_i(B, A)$ is a polynomial in the factor entries, so almost every (B, A) attains $\rho_i(r_g)$ in all modules simultaneously (Theorem H.1). Row u of ΔW_i starts in column $(c_{i-1} + uq_i) \bmod k$ of M, so the starting columns repeat with period $h_i = k/\gcd(k, q_i)$. Rows that start in the same column are combinations of the same column window of A, so a layout whose module rows start in few distinct columns forces low rank, while many distinct starting columns allow the rows to be independent.

Theorem 4.1 (Arithmetic layout condition)**.** *Let $h_i = k/\gcd(k, q_i)$. For every module order and every $r_g \geq 1$, $\rho_i(r_g) \geq \min\{p_i, q_i, h_i\}$, and if $q_i \leq k$, then $\rho_i(r_g) \leq \min\{p_i, q_i, 2r_g h_i\}$. Hence, if $h_i \geq \min(p_i, q_i)$ for every module, then for every $r_g \geq 1$, almost every (B, A) makes all module updates full rank.*

Write $\mathbf{1}_n$ and e_1 for the all-ones and first standard basis vectors. The lower bound uses the rank-one witness $B = \mathbf{1}_m$ and $A = a^\top$, with $a_j = \sum_{\ell < t} \omega^{\ell j}$, $\omega = e^{2\pi\sqrt{-1}/k}$, and $t = \min\{p_i, q_i, h_i\}$. Every row of BA equals $a^\top$, making the leading $t \times t$ block of ΔW_i a product of Vandermonde matrices with distinct nodes, hence generically nonsingular (Section H). For $r_g = 1$ and $q_i \leq k$, full rank requires $h_i \geq \min(p_i, q_i)/2$, so the condition is tight up to a factor of two. Both deployed layouts satisfy the stronger condition $h_i \geq \min(p_i, q_i)$, with $h_i = 13{,}033$ on Qwen and $9{,}385$ on Llama, so all q and v updates are generically full rank for every $r_g \geq 1$ (Corollary H.4); a finite-field certificate confirms this independently. In contrast, choosing $k = q_i$ would cap every update at rank r_g. The condition can be checked before training from k and the module widths alone.

Separation from LoRA. Full rank alone does not compare GFA with LoRA at a matched budget. Let $\mathcal{S}_G(P)$ and $\mathcal{S}_L(P)$ be the sets of module updates that GFA and module-wise LoRA, with any integer rank in each module including zero, can represent within a budget of P trainable parameters, with distances measured in the Frobenius norm over all modules.

Proposition 4.2 (Quantitative incomparability)**.** *In both deployed layouts and for every budget P between those of rank-1 and rank-16 LoRA, $\mathcal{S}_G(P)$ contains a unit-norm update whose squared distance to $\mathcal{S}_L(P)$ is at least 0.87 on Qwen and 0.94 on Llama, and $\mathcal{S}_L(P)$ contains a unit-norm update whose squared distance to $\mathcal{S}_G(P)$ is at least 0.51 on Qwen and 0.72 on Llama.*

The GFA witness, built from $\mathbf{1}_m e_1^\top/\sqrt{m}$, has m equal nonzero singular values, so each unit of LoRA rank captures at most $1/m$ of its energy. Conversely, the LoRA witness $\mathbf{1}_{p_i} e_1^\top/\sqrt{p_i}$ has its nonzero entries in distinct columns of M, with at most $\lceil k/q_i \rceil = 7$ entries per row, so each unit of global rank captures at most $7/p_i$ of its energy. This separation extends to nearby targets (Proposition I.5), while general distances are characterized by module-wise and global spectral tails (Lemmas I.3 and I.4). The gap is strongest at small budgets, matching the largest empirical GFA gains (Figure 3). Under standard assumptions, both methods retain the usual $O(1/T)$ full-batch and $O(1/\sqrt{T})$ stochastic stationarity rates (Theorem J.1).

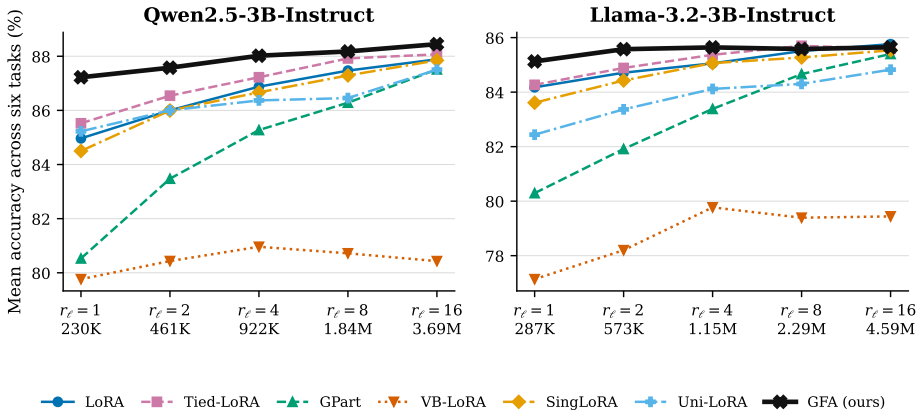

Figure 3: **Performance across budget anchors.** Each point is the mean accuracy of six tasks. Methods are matched to the corresponding LoRA budget as closely as their parameterizations permit.

5 Experimental Setup

We ask whether replacing independent module-wise factorizations with a single model-wide factorization improves accuracy at a fixed trainable-parameter budget, and what training cost reconstructing the module updates adds.

Models and tasks. We evaluate Qwen2.5-3B-Instruct (Qwen Team, 2024) and Llama-3.2-3B-Instruct (Grattafiori et al., 2024; Meta AI, 2024). Both models are fine-tuned on ARC-Easy and ARC-Challenge (Clark et al., 2018), PIQA (Bisk et al., 2020), BoolQ (Clark et al., 2019), WinoGrande (Sakaguchi et al., 2021), and OpenBookQA (Mihaylov et al., 2018). All methods adapt the query and value projections in every layer. Answers are greedily decoded from raw-text prompts and parsed.

Baselines. LoRA is the primary baseline because it isolates the change from independent module-wise factorizations to a single model-wide factorization. Tied-LoRA represents factor sharing across layers, SingLoRA represents factor tying within a module, and VB-LoRA represents vector-bank sharing. Uni-LoRA and GPart represent model-wide trainable vectors expanded through fixed random maps into LoRA factors and directly into the weight-update space, respectively. All six baselines are evaluated under the matched-budget protocol below. Tied-LoRA serves as the representative of cross-layer factor sharing, on which ShareLoRA, BSLoRA, and E^{2}LoRA build; SuperLoRA's single-group matrix configuration is GFA.

Budget matching. We anchor the standard budgets to uniform LoRA ranks $r_\ell \in \{1, 2, 4, 8, 16\}$. At each anchor, GFA uses the largest integer global rank whose parameter count does not exceed the corresponding LoRA budget. The resulting GFA ranks are $\{8, 17, 35, 70, 141\}$ for Qwen and $\{7, 15, 30, 61, 122\}$ for Llama (layouts in Table 10). Baselines that cannot hit an anchor exactly are kept within $\pm 15\%$ of the LoRA count. Exact counts are listed in Table 11.

Optimization and reporting. All results use three seeds $\{42, 43, 44\}$ and are reported as the mean and standard error over seeds. All methods in Table 2 share one recipe (AdamW (Loshchilov & Hutter, 2019), FP16). Dropout is 0.05 for LoRA and Tied-LoRA and 0 otherwise. Full hyperparameters, baseline configurations, and data splits are reported in Section E. We report accuracy on the official test splits of ARC-Easy, ARC-Challenge, and OpenBookQA and, because their test labels are not public, on the official validation splits of PIQA, BoolQ, and WinoGrande (Table 13).

6 Results

Average task-level performance across budgets. Table 2 summarizes each method by averaging uniformly over the five parameter-budget anchors. GFA obtains the highest average score on every task for both models. On Qwen2.5-3B-Instruct, GFA reaches an overall average of 87.89, compared with 87.06 for Tied-LoRA and 86.64 for LoRA. Its margin over the second-best method is positive on all six tasks and ranges from 0.30 points on PIQA to 1.86 points on WinoGrande. On Llama-

Table 2: **Average performance across all five parameter-budget anchors.** Uncertainties are standard errors across the three seed-level averages over the anchors. Avg. is the unweighted mean across the six tasks, and Avg. Params is the mean trainable-parameter count across anchors.

Method	Avg. Params	ARC-E	ARC-C	PIQA	BoolQ	WinoG	OBQA	Avg.
Qwen2.5-3B-Instruct								
LoRA	1,428,480	$93.73_{\pm 0.02}$	$82.38_{\pm 0.08}$	$85.75_{\pm 0.18}$	$87.74_{\pm 0.02}$	$81.42_{\pm 0.18}$	$\underline{88.81}_{\pm 0.27}$	86.64
Tied-LoRA	1,429,120	$\underline{94.04}_{\pm 0.04}$	$\underline{82.83}_{\pm 0.08}$	$\underline{86.17}_{\pm 0.17}$	$\underline{88.70}_{\pm 0.05}$	$\underline{82.34}_{\pm 0.20}$	$88.25_{\pm 0.24}$	$\underline{87.06}$
GPart	1,428,480	$93.26_{\pm 0.03}$	$79.85_{\pm 0.07}$	$84.58_{\pm 0.03}$	$86.65_{\pm 0.01}$	$77.67_{\pm 0.07}$	$85.68_{\pm 0.04}$	84.62
VB-LoRA	1,438,570	$87.91_{\pm 1.09}$	$80.92_{\pm 0.21}$	$83.24_{\pm 0.07}$	$83.62_{\pm 0.05}$	$70.97_{\pm 0.16}$	$76.12_{\pm 1.02}$	80.46
SingLoRA	1,402,675	$93.70_{\pm 0.06}$	$82.19_{\pm 0.06}$	$85.67_{\pm 0.10}$	$87.64_{\pm 0.06}$	$81.53_{\pm 0.26}$	$88.04_{\pm 0.08}$	86.46
Uni-LoRA	1,428,480	$93.69_{\pm 0.01}$	$82.27_{\pm 0.15}$	$85.29_{\pm 0.12}$	$87.77_{\pm 0.11}$	$80.99_{\pm 0.05}$	$87.85_{\pm 0.07}$	86.31
GFA (ours)	1,412,831	$\mathbf{94.44}_{\pm \mathbf{0.03}}$	$\mathbf{83.75}_{\pm \mathbf{0.16}}$	$\mathbf{86.47}_{\pm \mathbf{0.19}}$	$\mathbf{89.23}_{\pm \mathbf{0.06}}$	$\mathbf{84.20}_{\pm \mathbf{0.21}}$	$\mathbf{89.25}_{\pm \mathbf{0.09}}$	**87.89**
Llama-3.2-3B-Instruct								
LoRA	1,777,664	$90.75_{\pm 0.07}$	$\underline{77.92}_{\pm 0.15}$	$85.01_{\pm 0.06}$	$89.13_{\pm 0.06}$	$83.70_{\pm 0.23}$	$83.75_{\pm 0.29}$	85.04
Tied-LoRA	1,778,522	$\underline{90.87}_{\pm 0.09}$	$77.78_{\pm 0.20}$	$\underline{85.05}_{\pm 0.10}$	$\underline{89.24}_{\pm 0.06}$	$\underline{84.21}_{\pm 0.18}$	$\underline{83.83}_{\pm 0.16}$	$\underline{85.16}$
GPart	1,777,664	$90.06_{\pm 0.04}$	$77.28_{\pm 0.09}$	$83.08_{\pm 0.02}$	$86.63_{\pm 0.02}$	$79.77_{\pm 0.08}$	$81.96_{\pm 0.17}$	83.13
VB-LoRA	1,787,226	$88.76_{\pm 0.02}$	$75.16_{\pm 0.13}$	$79.21_{\pm 0.08}$	$81.90_{\pm 0.16}$	$70.59_{\pm 0.20}$	$77.12_{\pm 0.14}$	78.79
SingLoRA	1,777,664	$90.41_{\pm 0.05}$	$77.78_{\pm 0.12}$	$84.78_{\pm 0.09}$	$88.85_{\pm 0.02}$	$83.51_{\pm 0.06}$	$83.36_{\pm 0.20}$	84.78
Uni-LoRA	1,777,664	$89.66_{\pm 0.09}$	$76.78_{\pm 0.23}$	$83.89_{\pm 0.13}$	$88.27_{\pm 0.08}$	$82.20_{\pm 0.13}$	$82.05_{\pm 0.51}$	83.81
GFA (ours)	1,764,427	$\mathbf{90.93}_{\pm \mathbf{0.05}}$	$\mathbf{78.05}_{\pm \mathbf{0.15}}$	$\mathbf{85.36}_{\pm \mathbf{0.12}}$	$\mathbf{89.48}_{\pm \mathbf{0.07}}$	$\mathbf{84.41}_{\pm \mathbf{0.13}}$	$\mathbf{84.85}_{\pm \mathbf{0.31}}$	**85.51**

3.2-3B-Instruct, GFA reaches 85.51, compared with 85.16 for Tied-LoRA and 85.04 for LoRA. The Llama margins are smaller, ranging from 0.06 points on ARC-Easy to 1.02 points on OpenBookQA.

Performance across parameter budgets. Figure 3 shows that GFA attains the highest six-task average at every tested budget on Qwen and at the three smallest budgets on Llama. At the smallest anchor, $r_\ell = 1$, it is best on all six tasks for both models while using 8–10% fewer parameters than LoRA (Tables 5 and 6). Relative to LoRA, the Qwen differences are $+2.26$, $+1.57$, $+1.14$, $+0.71$, and $+0.56$ percentage points at the $r_\ell = 1, 2, 4, 8, 16$ anchors, respectively. The corresponding Llama differences are $+0.95$, $+0.86$, $+0.59$, $+0.08$, and -0.11 points. The gain is therefore largest in the low-budget regime and decreases as the budget grows; at the two largest Llama budgets, Tied-LoRA (r_ℓ=8) and LoRA (r_ℓ=16) are ahead by 0.12 and 0.11 points.

Across all 60 model-task-anchor cells, GFA exceeds LoRA in 52, ties in one (identical two-decimal means), and trails in seven, with an overall mean difference of $+0.86$ points. It exceeds LoRA in all 36 cells at the three smallest anchors, and all seven deficits occur at $r_\ell \in \{8, 16\}$. A nonparametric bootstrap over the twelve model-task clusters (procedure in Section B) gives a 95% confidence interval of $[0.51, 1.29]$. Separately, the mean differences are $+1.25$ points on Qwen with an interval of $[0.70, 1.96]$ and $+0.47$ points on Llama with an interval of $[0.22, 0.76]$. Against Tied-LoRA, the combined mean difference is $+0.59$ points with an interval of $[0.34, 0.89]$, and GFA is ahead in 50 of the 60 cells.

As a more conservative descriptive comparison, we also select the strongest baseline independently in each cell. GFA wins, ties, and loses 45, 1, and 14 of the 60 cells, respectively (26 wins of 30 on Qwen and 19 of 30 on Llama; per-task curves in Figure 4), and its mean margin over this per-cell envelope is $+0.47$ points with a model-task-cluster bootstrap interval of $[0.23, 0.76]$. Because the comparator is selected after observing each cell, this envelope is not a test against one prespecified baseline. Overall, GFA improves on LoRA and Tied-LoRA on average, with larger gains on Qwen and at small budgets, rather than dominating across all configurations.

Sub-unit budget control. A sub-unit budget is a trainable-parameter count below that of uniform rank-one LoRA on every adapted module ($230{,}400$ on Qwen q/v). GFA can reach it because $P = (m + k)r_g$ can be smaller than $\sum_i (p_i + q_i)$; partial-module LoRA, which adapts only a subset of the modules, can reach it as well. Table 3 compares the two on both models.

At $84{,}096$ trainable parameters, GFA obtains an average score of 86.87. Five partial-module rank-one LoRA allocations (uniformly spaced layers, the last q projections, and three random subsets) with $83{,}200$ to $87{,}808$ trainable parameters obtain mean scores between 80.06 and 82.35. Δ subtracts a per-task oracle that takes the best of the five allocations for each task (average 82.94, which no single fixed allocation attains), using the printed two-decimal values; its Mean entry averages the six per-task differences. On Llama (bottom), GFA averages 84.90, versus 82.68 for partial-module LoRA and 77.98 for GPart.

Table 3: GFA and GPart vs. partial-module rank-one LoRA below the rank-one LoRA floor

Method	Params	ARC-E	ARC-C	PIQA	BoolQ	WinoG	OBQA	Mean
GFA sub-r=1, 3-seed mean	84,096	**94.05**$_{\pm 0.15}$	**82.96**$_{\pm 0.23}$	**85.75**$_{\pm 0.39}$	**88.56**$_{\pm 0.06}$	**81.14**$_{\pm 0.09}$	**88.73**$_{\pm 0.27}$	**86.87**
GPart sub-r=1, 3-seed mean	84,992	82.28$_{\pm 0.08}$	79.41$_{\pm 0.03}$	81.86$_{\pm 0.07}$	83.67$_{\pm 0.02}$	67.11$_{\pm 0.07}$	76.20$_{\pm 0.23}$	78.42
Partial-LoRA Uniform	83,200	93.31	70.56	83.51	85.26	74.35	83.20	81.70
Partial-LoRA Last-q	86,016	92.97	68.26	82.92	85.14	74.27	82.20	80.96
Partial-LoRA Rand-1	85,504	92.55	75.00	84.00	84.89	74.03	82.80	82.21
Partial-LoRA Rand-2	85,504	93.31	73.29	84.66	84.68	73.95	84.20	82.35
Partial-LoRA Rand-3	87,808	92.68	62.03	83.73	84.28	75.22	82.40	80.06
Δ (GFA $-$ oracle)		+0.74	+7.96	+1.09	+3.30	+5.92	+4.53	+3.92
Llama-3.2-3B-Instruct								
GFA sub-r=1, 3-seed mean	104,268	**90.54**$_{\pm 0.10}$	**77.82**$_{\pm 0.15}$	**84.44**$_{\pm 0.19}$	**88.93**$_{\pm 0.09}$	**83.61**$_{\pm 0.65}$	**84.07**$_{\pm 0.74}$	**84.90**
GPart sub-r=1, 3-seed mean	104,268	88.61$_{\pm 0.01}$	75.26$_{\pm 0.05}$	78.85$_{\pm 0.02}$	81.97$_{\pm 0.03}$	66.32$_{\pm 0.05}$	76.87$_{\pm 0.07}$	77.98
Partial-LoRA, 3-seed mean	104,448	89.93$_{\pm 0.13}$	76.85$_{\pm 0.08}$	82.90$_{\pm 0.12}$	86.88$_{\pm 0.07}$	77.48$_{\pm 0.33}$	82.07$_{\pm 0.27}$	82.68
Δ (GFA $-$ partial-LoRA)		+0.61	+0.97	+1.54	+2.05	+6.13	+2.00	+2.22

Table 4: **Rank-one global budget**

Method	Params	ARC-E	ARC-C	PIQA	BoolQ	WinoG	OBQA	Avg.
Qwen2.5-3B-Instruct								
GFA (ours)	26,067	**93.48**$_{\pm 0.11}$	**81.85**$_{\pm 0.19}$	**84.39**$_{\pm 0.20}$	**87.23**$_{\pm 0.09}$	**77.01**$_{\pm 0.27}$	**87.07**$_{\pm 0.52}$	**85.17**
Uni-LoRA	26,067	90.64$_{\pm 0.01}$	81.37$_{\pm 0.03}$	82.68$_{\pm 0.02}$	83.48$_{\pm 0.04}$	69.88$_{\pm 0.09}$	74.87$_{\pm 0.07}$	80.49
Partial-module LoRA	26,112	80.32$_{\pm 5.06}$	77.90$_{\pm 0.60}$	82.48$_{\pm 0.19}$	83.67$_{\pm 0.02}$	70.27$_{\pm 0.49}$	79.07$_{\pm 0.53}$	78.95
GPart	26,067	92.58$_{\pm 0.01}$	81.17$_{\pm 0.03}$	79.63$_{\pm 0.10}$	79.18$_{\pm 0.04}$	62.67$_{\pm 0.09}$	75.40$_{\pm 0.12}$	78.44
Llama-3.2-3B-Instruct								
GFA (ours)	37,541	**90.11**$_{\pm 0.08}$	**77.99**$_{\pm 0.23}$	**83.55**$_{\pm 0.17}$	**88.14**$_{\pm 0.04}$	**79.19**$_{\pm 0.25}$	**83.40**$_{\pm 0.31}$	**83.73**
Uni-LoRA	37,541	88.69$_{\pm 0.04}$	75.00$_{\pm 0.10}$	80.43$_{\pm 0.10}$	83.20$_{\pm 0.04}$	74.48$_{\pm 0.11}$	77.73$_{\pm 0.07}$	79.92
Partial-module LoRA	36,864	89.06$_{\pm 0.04}$	75.80$_{\pm 0.38}$	79.23$_{\pm 0.45}$	84.22$_{\pm 0.12}$	61.33$_{\pm 0.56}$	78.00$_{\pm 0.23}$	77.94
GPart	37,541	88.51$_{\pm 0.02}$	74.80$_{\pm 0.17}$	76.79$_{\pm 0.04}$	80.43$_{\pm 0.02}$	59.64$_{\pm 0.16}$	75.60$_{\pm 0.00}$	75.96

Rank-one global budget. With $r_g = 1$, GFA needs $26{,}067$ (Qwen) and $37{,}541$ (Llama) parameters, $8.8\times$ and $7.6\times$ fewer than uniform rank-one LoRA. The other methods that learn factors for every module cannot meet this budget: at rank 1, SingLoRA and Tied-LoRA require $147{,}456$ and $89{,}416$ parameters on Qwen. Only methods that expand a trainable vector through a fixed projection (Uni-LoRA, GPart) or leave most modules unadapted (partial-module LoRA) go lower.

Here, GFA is best on every task and leads the next-best method, Uni-LoRA, by 4.68 and 3.81 points (Table 4). At this budget, every GFA module update remains generically full rank (Theorem 4.1), while partial-module LoRA adapts 9 of 72 (Qwen) and 7 of 56 (Llama) modules to meet the budget.

Training efficiency. At r_ℓ=16 on Qwen2.5-3B (3.69M LoRA vs. 3.68M GFA parameters), GFA with block-and-crop reconstruction uses $1.040\times$ the step time and $0.959\times$ the peak memory of LoRA (610.7 vs. 587.3 ms per step; 19.16 vs. 19.98 GiB; setup in Section C.1). The full decoder grid shows the same small overhead on both models: across all 179 paired GFA and LoRA runs (two models, five budgets, six tasks, three seeds, identical batch and sequence settings), the recorded peak memory of GFA is within 1.2% of that of LoRA (within 0.06% on Llama), and its median wall-clock time is $1.03\times$ that of LoRA on Qwen and $1.09\times$ on Llama, including up to about $\pm 15\%$ scheduling noise from shared GPUs (Tables 8 and 9). After merging, GFA adds no computation at inference.

7 Conclusion

We studied Global Factorized Adaptation (GFA), the simplest configuration of SuperLoRA's design space, which replaces LoRA's module-wise factorizations with a single low-rank factorization of all adapted coordinates. With a near-square layout it is parameter-efficient, and whenever the layout satisfies a simple arithmetic condition, as ours do, its module updates are generically full rank; on our models, GFA and LoRA are also far from containing each other at matched budgets. Under a shared recipe, GFA improves over LoRA by $+0.86$ points on average (cluster-bootstrap 95% interval $[0.51, 1.29]$), most at small budgets, outperforms partial-module LoRA of the same size at budgets up to $8.8\times$ below uniform rank-one LoRA, and has approximately LoRA-level training and inference cost, making it a practical alternative to module-wise low-rank adaptation.

AI USE STATEMENT

We used generative AI tools to assist with software implementation and refactoring, experiment planning and diagnostics, figure and table generation, and drafting and editing parts of the manuscript. The initial research question and GFA construction originated with the authors, who made all final decisions concerning the method, theoretical claims, experimental protocol, and scientific conclusions. AI-assisted code was tested against explicit correctness checks. The authors reviewed and take responsibility for the complete submission.

ETHICS STATEMENT

This work studies parameter-efficient fine-tuning using publicly available pretrained models and established NLP benchmarks. It does not involve human subjects, private user data, or new data collection. GFA may reduce the number of trainable parameters and deployment costs when adapting foundation models, but it does not address biases, unsafe behavior, privacy risks, or potential misuse inherited from the underlying models and training data. Performance on the evaluated benchmarks should not be interpreted as evidence of safety, fairness, or suitability for high-stakes deployment.

REPRODUCIBILITY STATEMENT

Section 3 specifies the GFA parameterization, initialization, and training and evaluation paths, while Section 5 describes the models, tasks, budget construction, optimization protocol, seeds, and statistical analysis, and Section E lists the shared training hyperparameters, data splits, baseline configurations, and evaluation protocol. Complete proofs and assumptions are provided in Appendices G–J. The supplementary material includes the implementation code, configuration files, frozen run manifest, exact trainable parameter counts, raw per-run result records, and scripts that regenerate the tables and figures.

REFERENCES

Armen Aghajanyan, Sonal Gupta, and Luke 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)*, pp. 7319–7328, 2021. doi: 10.18653/v1/2021.acl-long.568.

Paul Albert, Frederic Z. Zhang, Hemanth Saratchandran, Cristian Rodriguez-Opazo, Anton van den Hengel, and Ehsan Abbasnejad. RandLoRA: Full rank parameter-efficient fine-tuning of large models. In *International Conference on Learning Representations (ICLR)*, pp. 22258–22270, 2025. URL https://openreview.net/forum?id=Hn5eoTunHN.

David Bensaïd, Noam Rotstein, Roy Velich, Daniel Bensaïd, and Ron Kimmel. SingLoRA: Low rank adaptation using a single matrix. *arXiv preprint arXiv:2507.05566*, 2025. doi: 10.48550/arXiv.2507.05566. URL https://arxiv.org/abs/2507.05566.

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. doi: 10.1609/aaai.v34i05.6239.

Xiangyu Chen, Jing Liu, Ye Wang, Pu Wang, Matthew Brand, Guanghui Wang, and Toshiaki Koike-Akino. SuperLoRA: Parameter-efficient unified adaptation for large vision models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pp. 8050–8055, 2024. doi: 10.1109/CVPRW63382.2024.00804.

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 the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 2924–2936, 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. *arXiv preprint arXiv:1803.05457*, 2018.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. QLoRA: Efficient finetuning of quantized LLMs. In *Advances in Neural Information Processing Systems*, volume 36, pp. 10088–10115, 2023. doi: 10.52202/075280-0441.

Saeed Ghadimi and Guanghui Lan. Stochastic first- and zeroth-order methods for nonconvex stochastic programming. *SIAM Journal on Optimization*, 23(4):2341–2368, 2013. doi: 10.1137/120880811.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Soufiane Hayou, Nikhil Ghosh, and Bin Yu. LoRA+: Efficient low rank adaptation of large models. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, volume 235 of *Proceedings of Machine Learning Research*, pp. 17783–17806. PMLR, 2024.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 1026–1034, 2015. doi: 10.1109/ICCV.2015.123.

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.

Ting Jiang, Shaohan Huang, Shengyue Luo, Zihan Zhang, Haizhen Huang, Furu Wei, Weiwei Deng, Feng Sun, Qi Zhang, Deqing Wang, and Fuzhen Zhuang. MoRA: High-rank updating for parameter-efficient fine-tuning. *arXiv preprint arXiv:2405.12130*, 2024. URL https://arxiv.org/abs/2405.12130.

Dawid Jan Kopiczko, Tijmen Blankevoort, and Yuki M. Asano. VeRA: Vector-based random matrix adaptation. In *International Conference on Learning Representations (ICLR)*, pp. 6815–6835, 2024. URL https://openreview.net/forum?id=NjNfLdxr3A.

Kaiyang Li, Shaobo Han, Qing Su, Wei Li, Zhipeng Cai, and Shihao Ji. Uni-LoRA: One vector is all you need. In *Advances in Neural Information Processing Systems*, volume 38, pp. 49005–49030. Curran Associates, Inc., 2025. doi: 10.52202/085713-1461. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/3e596a70b60dcf7c6e0ac1a2d73e7470-Abstract-Conference.html.

Minglei Li, Peng Ye, Jingqi Ye, Haonan He, and Tao Chen. E^2LoRA: Efficient and effective low-rank adaptation with entropy-guided adaptive sharing. In *International Conference on Learning Representations (ICLR)*, pp. 76110–76137, 2026. URL https://openreview.net/forum?id=IQttyo0460.

Yang Li, Shaobo Han, and Shihao Ji. VB-LoRA: Extreme parameter efficient fine-tuning with vector banks. In *Advances in Neural Information Processing Systems*, volume 37, pp. 16724–16751. Curran Associates, Inc., 2024. doi: 10.52202/079017-0533. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/1e0d38c676d5855bcfab7f6d29d20ad9-Abstract-Conference.html.

Kai Lion, Liang Zhang, Bingcong Li, and Niao He. PoLAR: Polar-decomposed low-rank adapter representation. In *Advances in Neural Information Processing Systems*, volume 38, pp. 165867–165912. Curran Associates, Inc., 2025. doi: 10.52202/085713-4996. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/db79b6c50430a52fbab9d63efb8433ca-Abstract-Conference.html.

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.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*, 2019.

Paolo Mandica, Michał Brzozowski, Zuzanna Dubanowska, and Neo Christopher Chung. GPart: End-to-end isometric fine-tuning via global parameter partitioning. *arXiv preprint arXiv:2605.14841*, 2026. doi: 10.48550/arXiv.2605.14841. URL https://arxiv.org/abs/2605.14841.

Meta AI. Llama 3.2: Revolutionizing edge AI and vision with open, customizable models. https://ai.meta.com/blog/llama-3-2-connect-2024-vision-edge-mobile-devices/, 2024.

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 (EMNLP)*, pp. 2381–2391, 2018. doi: 10.18653/v1/D18-1260.

Qwen Team. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

Adithya Renduchintala, Tugrul Konuk, and Oleksii Kuchaiev. Tied-LoRA: Enhancing parameter efficiency of LoRA with weight tying. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 8694–8705, Mexico City, Mexico, 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.naacl-long.481. URL https://aclanthology.org/2024.naacl-long.481/.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. WinoGrande: An adversarial Winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021. doi: 10.1145/3474381.

Yurun Song, Junchen Zhao, Ian G. Harris, and Sangeetha Abdu Jyothi. ShareLoRA: Parameter efficient and robust large language model fine-tuning via shared low-rank adaptation. *arXiv preprint arXiv:2406.10785*, 2024. doi: 10.48550/arXiv.2406.10785. URL https://arxiv.org/abs/2406.10785.

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

Yuhua Zhou, Ruifeng Li, Changhai Zhou, Fei Yang, and Aimin Pan. BSLoRA: Enhancing the parameter efficiency of LoRA with intra-layer and inter-layer sharing. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, volume 267 of *Proceedings of Machine Learning Research*, pp. 78883–78902. PMLR, 2025. URL https://proceedings.mlr.press/v267/zhou25k.html.

A PSEUDOCODE OF GFA

Algorithm 1 summarizes the two reconstruction paths of Section 3: block-and-crop reconstruction (Equations (3) and (4)) whenever gradients are enabled, and a single global product (Equation (1)) for no-gradient evaluation and weight merging.

Algorithm 1 GFA training, evaluation, and weight merging

Require: Frozen target weights $\{W_i\}_{i=1}^{N}$, trainable factors B, A, offsets $\{c_i\}_{i=0}^{N}$, global width k, update scale s, training batch $\mathcal{B}$, optimizer $\mathcal{O}$

Precompute $u_i = \lfloor c_{i-1}/k \rfloor$, $v_i = \lceil c_i/k \rceil$, and $\ell_i = c_{i-1} - u_i k$ for every module i

Training forward pass

for each target module i encountered during the model forward pass **do**
  $\widetilde{M}_i \leftarrow B_{u_i:v_i,:}A$
  $\delta_i \leftarrow [\mathrm{vec}(\widetilde{M}_i)]_{\ell_i:\ell_i+d_i}$
  $\Delta W_i \leftarrow \mathrm{reshape}_{p_i \times q_i}(\delta_i)$
  $W_i^{\mathrm{eff}} \leftarrow W_i + s\Delta W_i$
  $y_i \leftarrow W_i^{\mathrm{eff}} x_i$
end for
Compute the task loss $\mathcal{L}$ from the model output

Training backward pass

$G_B \leftarrow 0_{m \times r_g}$, $G_A \leftarrow 0_{r_g \times k}$
for each target module i in reverse computational order **do**
  $H_i \leftarrow \nabla_{W_i^{\mathrm{eff}}} \mathcal{L}$
  $\widetilde{G}_i \leftarrow 0_{(v_i - u_i) \times k}$
  $[\mathrm{vec}(\widetilde{G}_i)]_{\ell_i:\ell_i+d_i} \leftarrow s\ \mathrm{vec}(H_i)$
  $(G_B)_{u_i:v_i,:} \leftarrow (G_B)_{u_i:v_i,:} + \widetilde{G}_i A^\top$
  $G_A \leftarrow G_A + B_{u_i:v_i,:}^\top \widetilde{G}_i$
end for
$(B, A) \leftarrow \mathcal{O}(B, A; G_B, G_A)$

No-gradient evaluation or weight merging

Disable gradient tracking
$M \leftarrow BA$
for $i = 1, \ldots, N$ **do**
  $\Delta W_i \leftarrow \mathrm{reshape}_{p_i \times q_i}\left([\mathrm{vec}(M)]_{c_{i-1}:c_i}\right)$
  $\widehat{W}_i \leftarrow W_i + s\Delta W_i$
end for
Evaluate using $\{\widehat{W}_i\}_{i=1}^{N}$, or store them and discard B and A for deployment

B EXTENDED RESULTS

Tables 5 and 6 report the results for five budget anchors, six tasks, and seven methods per model, from which Table 2, Figure 3, and all headline statistics are computed.

Protocol. Each entry is the mean and standard error ($\mathrm{sd}/\sqrt{n}$) over seeds 42, 43, and 44. Best and second-best values are marked within each anchor. For the cell-level counts in Section 6, the comparator in each cell is the strongest of the six baselines; differences are computed from two-decimal means, and a tie means identical means; these counts are descriptive, not inferential.

Cluster bootstrap. The intervals in Section 6 are two-stage. Differences between three-seed means are first averaged within each model–task cluster over the five anchors (for the envelope comparison, each cell's difference is taken against the strongest baseline in that cell), so the five budgets of one task are never treated as independent observations. The clusters (all twelve for the combined intervals, the six of one model for the per-model intervals) are then resampled with replacement, and the interval is formed from the 2.5th and 97.5th percentiles of the resampled mean over $B = 10{,}000$ resamples (NumPy `default_rng(0)`).

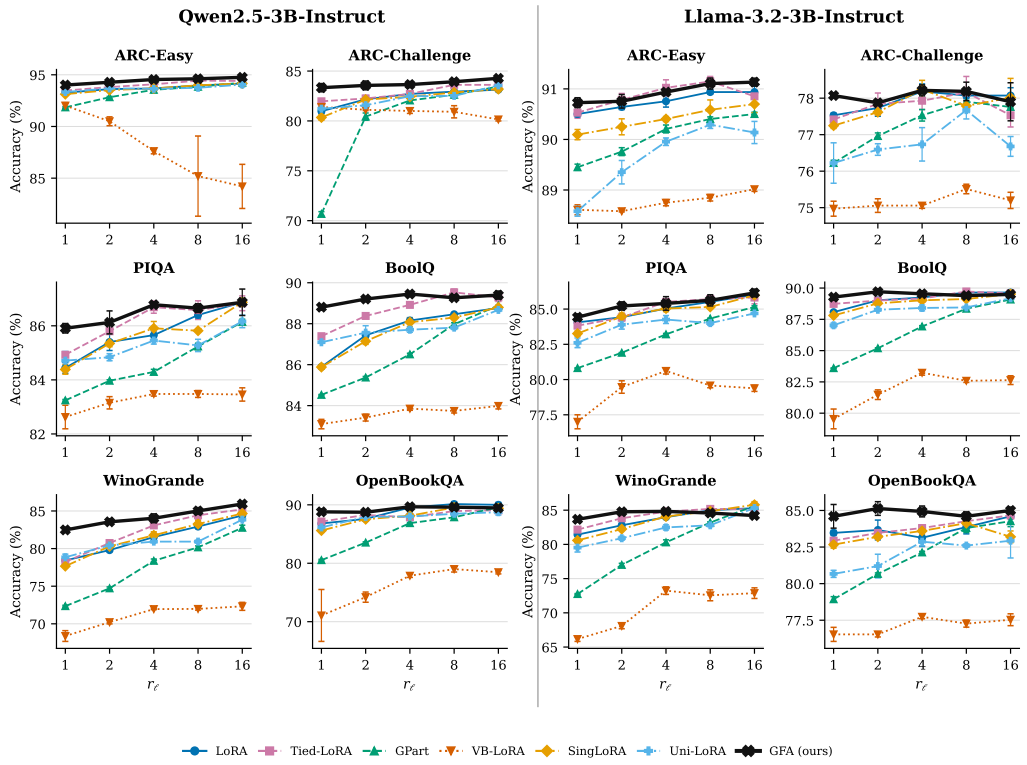

Figure 4: **Per-task accuracy across budget anchors on Qwen2.5-3B-Instruct (left two columns) and Llama-3.2-3B-Instruct (right two columns).** Seed means at the five LoRA-rank anchors r_ℓ (horizontal axis); error bars show ± 1 standard error over three seeds. On Qwen, GFA has the highest mean in 26 of the 30 cells, and ties LoRA and SingLoRA in one (PIQA, r_ℓ=16); its three deficits are 0.53 points (OpenBookQA, r_ℓ=8 and 16, vs. LoRA) and 0.26 points (BoolQ, r_ℓ=8, vs. Tied-LoRA). On Llama, GFA has the highest mean in 19 of the 30 cells; its eleven deficits are at most 0.61 points except on WinoGrande at r_ℓ=16, where SingLoRA is 1.58 points higher. Values are those of Tables 5 and 6.

Table 5: **Six-task results for Qwen2.5-3B-Instruct.** Task entries are seed means with standard errors. Avg. is the unweighted mean across the six tasks. Best and second-best values are marked within each budget anchor.

Anchor	Method	Params	ARC-E	ARC-C	PIQA	BoolQ	WinoG	OBQA	Avg.
r1	LoRA	230,400	93.27$_{\pm 0.05}$	80.97$_{\pm 0.23}$	84.44$_{\pm 0.20}$	85.89$_{\pm 0.06}$	78.43$_{\pm 0.18}$	86.80$_{\pm 0.20}$	84.97
	Tied-LoRA	231,800	93.48$_{\pm 0.11}$	81.97$_{\pm 0.10}$	84.93$_{\pm 0.05}$	87.39$_{\pm 0.09}$	78.24$_{\pm 0.18}$	87.13$_{\pm 0.47}$	85.52
	GPart	230,400	91.85$_{\pm 0.07}$	70.68$_{\pm 0.23}$	83.24$_{\pm 0.03}$	84.53$_{\pm 0.02}$	72.35$_{\pm 0.11}$	80.53$_{\pm 0.13}$	80.53
	VB-LoRA	230,832	92.02$_{\pm 0.29}$	81.40$_{\pm 0.00}$	82.63$_{\pm 0.44}$	83.10$_{\pm 0.23}$	68.38$_{\pm 0.72}$	71.07$_{\pm 4.44}$	79.76
	SingLoRA	230,400	93.13$_{\pm 0.06}$	80.35$_{\pm 0.27}$	84.39$_{\pm 0.17}$	85.88$_{\pm 0.10}$	77.66$_{\pm 0.16}$	85.60$_{\pm 0.31}$	84.50
	Uni-LoRA	230,400	93.34$_{\pm 0.26}$	81.23$_{\pm 0.18}$	84.71$_{\pm 0.34}$	87.09$_{\pm 0.03}$	78.85$_{\pm 0.41}$	86.13$_{\pm 0.48}$	85.23
	GFA (ours)	208,536	**94.01$_{\pm 0.07}$**	**83.33$_{\pm 0.41}$**	**85.91$_{\pm 0.17}$**	**88.81$_{\pm 0.11}$**	**82.48$_{\pm 0.16}$**	**88.80$_{\pm 0.00}$**	**87.22**
r2	LoRA	460,800	93.62$_{\pm 0.04}$	82.17$_{\pm 0.13}$	85.40$_{\pm 0.31}$	87.43$_{\pm 0.14}$	79.79$_{\pm 0.18}$	87.60$_{\pm 0.35}$	86.00
	Tied-LoRA	458,320	93.81$_{\pm 0.05}$	82.25$_{\pm 0.31}$	85.80$_{\pm 0.11}$	88.38$_{\pm 0.11}$	80.74$_{\pm 0.32}$	88.27$_{\pm 0.64}$	86.54
	GPart	460,800	92.83$_{\pm 0.07}$	80.43$_{\pm 0.36}$	83.97$_{\pm 0.04}$	85.37$_{\pm 0.05}$	74.72$_{\pm 0.18}$	83.53$_{\pm 0.13}$	83.48
	VB-LoRA	461,664	90.50$_{\pm 0.43}$	81.11$_{\pm 0.08}$	83.15$_{\pm 0.23}$	83.41$_{\pm 0.17}$	70.22$_{\pm 0.23}$	74.20$_{\pm 0.87}$	80.43
	SingLoRA	442,368	93.52$_{\pm 0.04}$	82.14$_{\pm 0.08}$	85.35$_{\pm 0.12}$	87.15$_{\pm 0.15}$	80.29$_{\pm 0.28}$	87.53$_{\pm 0.18}$	86.00
	Uni-LoRA	460,800	93.69$_{\pm 0.23}$	81.57$_{\pm 0.21}$	84.84$_{\pm 0.13}$	87.56$_{\pm 0.34}$	80.43$_{\pm 0.05}$	87.93$_{\pm 0.48}$	86.00
	GFA (ours)	443,139	**94.28$_{\pm 0.17}$**	**83.56$_{\pm 0.44}$**	**86.13$_{\pm 0.42}$**	**89.20$_{\pm 0.08}$**	**83.56$_{\pm 0.33}$**	**88.73$_{\pm 0.07}$**	**87.58**
r4	LoRA	921,600	93.67$_{\pm 0.12}$	82.71$_{\pm 0.22}$	85.65$_{\pm 0.15}$	88.15$_{\pm 0.14}$	81.53$_{\pm 0.46}$	89.53$_{\pm 0.18}$	86.88
	Tied-LoRA	924,304	94.08$_{\pm 0.12}$	82.71$_{\pm 0.23}$	86.69$_{\pm 0.22}$	88.92$_{\pm 0.02}$	83.08$_{\pm 0.34}$	87.87$_{\pm 0.52}$	87.22
	GPart	921,600	93.55$_{\pm 0.01}$	82.05$_{\pm 0.23}$	84.29$_{\pm 0.10}$	86.50$_{\pm 0.05}$	78.37$_{\pm 0.24}$	86.87$_{\pm 0.07}$	85.27
	VB-LoRA	937,984	87.61$_{\pm 0.26}$	81.00$_{\pm 0.25}$	83.48$_{\pm 0.08}$	83.85$_{\pm 0.08}$	71.95$_{\pm 0.07}$	77.87$_{\pm 0.29}$	80.96
	SingLoRA	884,736	93.69$_{\pm 0.05}$	82.42$_{\pm 0.09}$	85.91$_{\pm 0.25}$	88.07$_{\pm 0.21}$	81.77$_{\pm 0.46}$	88.13$_{\pm 0.07}$	86.67
	Uni-LoRA	921,600	93.64$_{\pm 0.07}$	82.48$_{\pm 0.33}$	85.46$_{\pm 0.13}$	87.71$_{\pm 0.12}$	80.93$_{\pm 0.14}$	88.00$_{\pm 0.72}$	86.37
	GFA (ours)	912,345	**94.54$_{\pm 0.07}$**	**83.65$_{\pm 0.30}$**	**86.78$_{\pm 0.03}$**	**89.45$_{\pm 0.02}$**	**84.03$_{\pm 0.59}$**	**89.67$_{\pm 0.13}$**	**88.02**
r8	LoRA	1,843,200	93.97$_{\pm 0.18}$	82.94$_{\pm 0.09}$	86.40$_{\pm 0.11}$	88.46$_{\pm 0.07}$	82.93$_{\pm 0.26}$	**90.13$_{\pm 0.41}$**	87.47
	Tied-LoRA	1,843,328	94.43$_{\pm 0.04}$	83.62$_{\pm 0.10}$	86.58$_{\pm 0.34}$	**89.53$_{\pm 0.16}$**	84.43$_{\pm 0.61}$	88.93$_{\pm 0.68}$	87.92
	GPart	1,843,200	93.84$_{\pm 0.11}$	82.62$_{\pm 0.14}$	85.24$_{\pm 0.12}$	87.97$_{\pm 0.08}$	80.16$_{\pm 0.19}$	87.87$_{\pm 0.18}$	86.28
	VB-LoRA	1,859,584	85.20$_{\pm 3.88}$	80.92$_{\pm 0.61}$	83.48$_{\pm 0.13}$	83.74$_{\pm 0.10}$	71.98$_{\pm 0.16}$	79.00$_{\pm 0.50}$	80.72
	SingLoRA	1,769,472	93.98$_{\pm 0.14}$	82.85$_{\pm 0.27}$	85.82$_{\pm 0.05}$	88.30$_{\pm 0.09}$	83.27$_{\pm 0.48}$	89.53$_{\pm 0.13}$	87.29
	Uni-LoRA	1,843,200	93.74$_{\pm 0.04}$	82.51$_{\pm 0.15}$	85.27$_{\pm 0.23}$	87.81$_{\pm 0.09}$	80.93$_{\pm 0.18}$	88.47$_{\pm 0.37}$	86.45
	GFA (ours)	1,824,690	**94.61$_{\pm 0.12}$**	**83.93$_{\pm 0.11}$**	**86.65$_{\pm 0.16}$**	89.27$_{\pm 0.12}$	**85.00$_{\pm 0.45}$**	89.60$_{\pm 0.12}$	**88.18**
r16	LoRA	3,686,400	94.15$_{\pm 0.09}$	83.13$_{\pm 0.12}$	**86.87$_{\pm 0.16}$**	88.77$_{\pm 0.04}$	84.40$_{\pm 0.22}$	**90.00$_{\pm 0.40}$**	87.89
	Tied-LoRA	3,687,848	94.40$_{\pm 0.02}$	83.62$_{\pm 0.32}$	86.83$_{\pm 0.27}$	89.29$_{\pm 0.20}$	85.21$_{\pm 0.23}$	89.07$_{\pm 0.41}$	88.07
	GPart	3,686,400	94.23$_{\pm 0.14}$	83.45$_{\pm 0.21}$	86.14$_{\pm 0.10}$	88.89$_{\pm 0.01}$	82.77$_{\pm 0.34}$	89.60$_{\pm 0.31}$	87.51
	VB-LoRA	3,702,784	84.20$_{\pm 2.13}$	80.15$_{\pm 0.15}$	83.46$_{\pm 0.25}$	83.99$_{\pm 0.15}$	72.32$_{\pm 0.53}$	78.47$_{\pm 0.29}$	80.43
	SingLoRA	3,686,400	94.16$_{\pm 0.11}$	83.19$_{\pm 0.23}$	**86.87$_{\pm 0.15}$**	88.80$_{\pm 0.08}$	84.66$_{\pm 0.33}$	89.40$_{\pm 0.31}$	87.85
	Uni-LoRA	3,686,400	94.05$_{\pm 0.22}$	83.56$_{\pm 0.12}$	86.16$_{\pm 0.23}$	88.67$_{\pm 0.07}$	83.85$_{\pm 0.15}$	88.73$_{\pm 0.48}$	87.50
	GFA (ours)	3,675,447	**94.75$_{\pm 0.09}$**	**84.27$_{\pm 0.19}$**	**86.87$_{\pm 0.49}$**	**89.40$_{\pm 0.20}$**	**85.92$_{\pm 0.31}$**	89.47$_{\pm 0.13}$	**88.45**

Table 6: **Six-task results for Llama-3.2-3B-Instruct.** Task entries are seed means with standard errors. Avg. is the unweighted mean across the six tasks. Best and second-best values are marked within each budget anchor.

Anchor	Method	Params	ARC-E	ARC-C	PIQA	BoolQ	WinoG	OBQA	Avg.
r1	LoRA	286,720	90.50$_{\pm 0.06}$	77.53$_{\pm 0.06}$	84.06$_{\pm 0.14}$	88.07$_{\pm 0.05}$	81.43$_{\pm 0.22}$	83.47$_{\pm 0.18}$	84.18
	Tied-LoRA	289,720	90.53$_{\pm 0.11}$	77.42$_{\pm 0.16}$	83.82$_{\pm 0.12}$	88.74$_{\pm 0.13}$	82.16$_{\pm 0.27}$	82.93$_{\pm 0.18}$	84.27
	GPart	286,720	89.45$_{\pm 0.06}$	76.22$_{\pm 0.08}$	80.81$_{\pm 0.02}$	83.60$_{\pm 0.01}$	72.74$_{\pm 0.07}$	78.93$_{\pm 0.18}$	80.29
	VB-LoRA	286,272	88.61$_{\pm 0.10}$	74.97$_{\pm 0.21}$	77.00$_{\pm 0.50}$	79.54$_{\pm 0.79}$	66.17$_{\pm 0.34}$	76.53$_{\pm 0.48}$	77.14
	SingLoRA	286,720	90.10$_{\pm 0.10}$	77.25$_{\pm 0.08}$	83.26$_{\pm 0.18}$	87.81$_{\pm 0.07}$	80.58$_{\pm 0.23}$	82.67$_{\pm 0.24}$	83.61
	Uni-LoRA	286,720	88.58$_{\pm 0.10}$	76.22$_{\pm 0.55}$	82.61$_{\pm 0.34}$	87.02$_{\pm 0.19}$	79.53$_{\pm 0.65}$	80.67$_{\pm 0.24}$	82.44
	GFA (ours)	262,787	**90.73$_{\pm 0.10}$**	**78.07$_{\pm 0.05}$**	**84.42$_{\pm 0.24}$**	**89.28$_{\pm 0.15}$**	**83.66$_{\pm 0.34}$**	**84.60$_{\pm 0.81}$**	**85.13**
r2	LoRA	573,440	90.64$_{\pm 0.05}$	77.73$_{\pm 0.23}$	84.42$_{\pm 0.02}$	89.02$_{\pm 0.12}$	82.79$_{\pm 0.32}$	83.67$_{\pm 0.68}$	84.71
	Tied-LoRA	578,008	**90.78$_{\pm 0.13}$**	77.84$_{\pm 0.30}$	84.40$_{\pm 0.26}$	89.00$_{\pm 0.05}$	83.82$_{\pm 0.16}$	83.47$_{\pm 0.07}$	84.89
	GPart	573,440	89.76$_{\pm 0.08}$	76.96$_{\pm 0.09}$	81.90$_{\pm 0.02}$	85.19$_{\pm 0.04}$	77.01$_{\pm 0.23}$	80.67$_{\pm 0.07}$	81.91
	VB-LoRA	572,544	88.58$_{\pm 0.01}$	75.06$_{\pm 0.19}$	79.47$_{\pm 0.44}$	81.48$_{\pm 0.38}$	68.09$_{\pm 0.41}$	76.53$_{\pm 0.18}$	78.20
	SingLoRA	516,096	90.25$_{\pm 0.16}$	77.62$_{\pm 0.08}$	84.48$_{\pm 0.24}$	88.78$_{\pm 0.15}$	82.24$_{\pm 0.08}$	83.20$_{\pm 0.12}$	84.43
	Uni-LoRA	573,440	89.35$_{\pm 0.23}$	76.59$_{\pm 0.16}$	83.90$_{\pm 0.32}$	88.27$_{\pm 0.11}$	80.90$_{\pm 0.28}$	81.20$_{\pm 0.81}$	83.37
	GFA (ours)	563,115	90.75$_{\pm 0.13}$	**77.87$_{\pm 0.06}$**	**85.22$_{\pm 0.29}$**	**89.70$_{\pm 0.18}$**	**84.77$_{\pm 0.41}$**	**85.13$_{\pm 0.47}$**	**85.58**
r4	LoRA	1,146,880	90.75$_{\pm 0.11}$	78.19$_{\pm 0.21}$	85.04$_{\pm 0.03}$	89.25$_{\pm 0.10}$	83.98$_{\pm 0.08}$	83.13$_{\pm 0.59}$	85.06
	Tied-LoRA	1,144,288	**91.02$_{\pm 0.16}$**	77.93$_{\pm 0.24}$	**85.51$_{\pm 0.07}$**	89.16$_{\pm 0.17}$	84.79$_{\pm 0.46}$	83.80$_{\pm 0.20}$	85.37
	GPart	1,146,880	90.21$_{\pm 0.08}$	77.53$_{\pm 0.16}$	83.21$_{\pm 0.07}$	86.93$_{\pm 0.06}$	80.29$_{\pm 0.37}$	82.13$_{\pm 0.07}$	83.38
	VB-LoRA	1,163,264	88.75$_{\pm 0.06}$	75.06$_{\pm 0.08}$	80.61$_{\pm 0.22}$	83.23$_{\pm 0.19}$	73.24$_{\pm 0.55}$	77.73$_{\pm 0.13}$	79.77
	SingLoRA	1,204,224	90.40$_{\pm 0.05}$	**78.24$_{\pm 0.25}$**	85.04$_{\pm 0.22}$	89.02$_{\pm 0.14}$	84.08$_{\pm 0.15}$	83.60$_{\pm 0.23}$	85.06
	Uni-LoRA	1,146,880	89.96$_{\pm 0.07}$	76.73$_{\pm 0.46}$	84.26$_{\pm 0.28}$	88.41$_{\pm 0.23}$	82.48$_{\pm 0.37}$	82.87$_{\pm 0.29}$	84.12
	GFA (ours)	1,126,230	90.94$_{\pm 0.03}$	78.21$_{\pm 0.15}$	85.40$_{\pm 0.48}$	**89.54$_{\pm 0.15}$**	**84.82$_{\pm 0.34}$**	**84.93$_{\pm 0.37}$**	**85.64**
r8	LoRA	2,293,760	90.94$_{\pm 0.04}$	78.07$_{\pm 0.09}$	85.51$_{\pm 0.07}$	89.64$_{\pm 0.11}$	85.00$_{\pm 0.43}$	83.87$_{\pm 0.47}$	85.51
	Tied-LoRA	2,297,440	**91.15$_{\pm 0.10}$**	78.16$_{\pm 0.44}$	**85.71$_{\pm 0.14}$**	**89.72$_{\pm 0.19}$**	**85.19$_{\pm 0.29}$**	84.27$_{\pm 0.47}$	**85.70**
	GPart	2,293,760	90.40$_{\pm 0.04}$	77.90$_{\pm 0.18}$	84.33$_{\pm 0.03}$	88.35$_{\pm 0.03}$	83.19$_{\pm 0.12}$	83.80$_{\pm 0.31}$	84.66
	VB-LoRA	2,310,144	88.85$_{\pm 0.06}$	75.51$_{\pm 0.13}$	79.58$_{\pm 0.16}$	82.59$_{\pm 0.13}$	72.56$_{\pm 0.80}$	77.27$_{\pm 0.24}$	79.39
	SingLoRA	2,236,416	90.59$_{\pm 0.19}$	77.79$_{\pm 0.11}$	85.15$_{\pm 0.11}$	89.12$_{\pm 0.13}$	84.85$_{\pm 0.41}$	84.13$_{\pm 0.24}$	85.27
	Uni-LoRA	2,293,760	90.29$_{\pm 0.07}$	77.67$_{\pm 0.24}$	84.00$_{\pm 0.16}$	88.45$_{\pm 0.05}$	82.79$_{\pm 0.20}$	82.60$_{\pm 0.12}$	84.30
	GFA (ours)	2,290,001	91.11$_{\pm 0.11}$	**78.19$_{\pm 0.25}$**	85.62$_{\pm 0.40}$	89.40$_{\pm 0.21}$	84.58$_{\pm 0.09}$	**84.60$_{\pm 0.31}$**	85.58
r16	LoRA	4,587,520	90.94$_{\pm 0.13}$	**78.07$_{\pm 0.21}$**	86.02$_{\pm 0.20}$	**89.64$_{\pm 0.16}$**	85.29$_{\pm 0.47}$	84.60$_{\pm 0.42}$	**85.76**
	Tied-LoRA	4,583,152	90.85$_{\pm 0.04}$	77.53$_{\pm 0.32}$	85.82$_{\pm 0.45}$	89.56$_{\pm 0.05}$	85.11$_{\pm 0.32}$	84.67$_{\pm 0.24}$	85.59
	GPart	4,587,520	90.50$_{\pm 0.04}$	77.76$_{\pm 0.03}$	85.15$_{\pm 0.08}$	89.08$_{\pm 0.02}$	85.64$_{\pm 0.20}$	84.27$_{\pm 0.37}$	85.40
	VB-LoRA	4,603,904	89.02$_{\pm 0.04}$	75.20$_{\pm 0.22}$	79.38$_{\pm 0.22}$	82.64$_{\pm 0.35}$	72.88$_{\pm 0.78}$	77.53$_{\pm 0.41}$	79.44
	SingLoRA	4,644,864	90.70$_{\pm 0.17}$	78.01$_{\pm 0.53}$	85.98$_{\pm 0.17}$	89.52$_{\pm 0.03}$	**85.79$_{\pm 0.45}$**	83.20$_{\pm 0.40}$	85.53
	Uni-LoRA	4,587,520	90.14$_{\pm 0.22}$	76.68$_{\pm 0.27}$	84.69$_{\pm 0.04}$	89.18$_{\pm 0.10}$	85.32$_{\pm 0.57}$	82.93$_{\pm 1.18}$	84.82
	GFA (ours)	4,580,002	**91.13$_{\pm 0.08}$**	77.90$_{\pm 0.52}$	**86.14$_{\pm 0.27}$**	89.49$_{\pm 0.04}$	84.21$_{\pm 0.05}$	**85.00$_{\pm 0.23}$**	85.65

C ADDITIONAL ABLATIONS

C.1 TRAINING AND EVALUATION RECONSTRUCTION PATHS

We compare the block-and-crop training implementation with an alternative that materializes the complete global product in one matrix multiplication and then slices the module updates. The two paths produce identical logits and gradients up to numerical precision, but their autograd behavior differs.

Table 7: **Reconstruction-path microbenchmark.** Measurements use the near-square Qwen geometry with 72 target modules in FP16. Peak memory refers to the isolated reconstruction operation.

Reconstruction	Forward	Backward	Peak memory
Block-and-crop	6.24 ms	31.79 ms	339 MiB
Single global product	1.54 ms	177.02 ms	971 MiB

Materializing the complete product reduces forward reconstruction time by approximately $4.1\times$. During training, however, its backward pass is approximately $5.6\times$ slower and requires substantially more temporary memory because gradients from the module slices must be accumulated into the shared global tensor. We therefore use block-and-crop reconstruction when gradients are enabled and single-product materialization during gradient-free evaluation and adapter merging.

Benchmark setup. The end-to-end benchmark in Section 6 runs the full 36-layer Qwen2.5-3B model in FP16 with batches of 8×256 tokens and times a complete forward, backward, and AdamW step; Table 7 times the reconstruction operation alone. The benchmark runs on an NVIDIA TITAN RTX (24 GB) with PyTorch 2.11.0 (CUDA 12.8, cuDNN 9.19.0), Transformers 5.5.4, and PEFT 0.19.1, with gradient checkpointing enabled and the default SDPA attention (FlashAttention is not installed). Each variant runs in its own process, and we time 10 iterations after 4 warm-up iterations with CUDA synchronization. These numbers differ from Section D, which comes from the shared training grid.

D VRAM AND WALL-CLOCK

Both tables are generated from the `peak_vram_mb` and `wall_clock_seconds` fields recorded in every run of the locked grid; nothing is measured separately. Runs were executed on shared TITAN RTX 24 GB GPUs, usually two per GPU, so wall-clock carries scheduling noise of roughly $\pm 15\%$ between anchors; the VRAM figures are deterministic. These training-grid measurements are not directly comparable to the dedicated benchmark in Section 6 (setup in Section C.1).

Table 8: **Peak VRAM at matched batch size.** MB, mean over three seeds of the recorded `peak_vram_mb`. All methods in a row use the same batch size and sequence budget, except that in the cell marked [§] one seed (42) ran at per-device batch 4 with four accumulation steps and sequence length 512. Batch is the per-device batch size (accumulation steps in Table 12). Δ is GFA relative to LoRA.

Model	Task	Anchor	Batch	LoRA	Tied-LoRA	GPart	VB-LoRA	SingLoRA	Uni-LoRA	GFA (ours)	Δ
Qwen2.5-3B	ARC-Easy	r4	8	9,896	9,784	10,043	9,784	9,784	9,904	9,897	$+0.02\%$
	ARC-Easy	r16	8	9,938	9,826	10,083	9,864	9,826	9,966	9,938	$+0.01\%$
	BoolQ	r4	8	13,759	13,651	13,271	13,651	13,650	13,731	13,761	$+0.01\%$
	BoolQ	r16	8	13,802	13,693	12,151	13,729	13,693	13,830	13,802	$+0.00\%$
Llama-3.2-3B	ARC-Easy	r4	8	9,529	9,528	12,034	9,529	9,529	8,978[§]	9,528	-0.00%
	ARC-Easy	r16	8	9,581	9,581	12,088	9,581	9,582	9,618	9,581	-0.00%
	BoolQ	r4	8	12,897	12,896	14,891	12,896	12,897	12,905	12,896	-0.00%
	BoolQ	r16	8	12,949	12,948	13,926	12,949	12,949	12,986	12,949	-0.00%

At matched batch size, GFA's peak memory is within 0.02% of LoRA's in every cell of Table 8. GFA's wall-clock lies between -10% and $+37\%$ of LoRA's across the ten model$\times$anchor rows (materializing BA costs $O(r_g D)$, which is not hidden by fusion in our implementation). The budget-resolution ratio is $\sum_i (p_i + q_i)/(m + k) = 8.84$ (Qwen q/v) and 7.64 (Llama).

Table 9: **Wall-clock per run on the decoder grid.** Minutes; mean over the six tasks and three seeds of each anchor, from the recorded `wall_clock_seconds` (TITAN RTX 24 GB, one run per GPU or two stacked). Relative to LoRA in parentheses. Runs share the same recipe; differences reflect the adapter's forward/backward cost and up to about $\pm 15\%$ scheduling noise from shared GPUs (Section D).

Model	Anchor	LoRA	Tied-LoRA	GPart	VB-LoRA	SingLoRA	Uni-LoRA	GFA (ours)
Qwen2.5-3B	r1	89.5	158.5 (+77%)	90.3 (+1%)	186.5 (+108%)	87.2 (-3%)	84.4 (-6%)	92.8 (+4%)
	r2	96.6	165.7 (+72%)	103.3 (+7%)	194.9 (+102%)	159.9 (+66%)	100.2 (+4%)	87.0 (-10%)
	r4	99.0	170.2 (+72%)	76.2 (-23%)	111.7 (+13%)	148.5 (+50%)	97.5 (-1%)	107.8 (+9%)
	r8	95.8	169.1 (+77%)	89.4 (-7%)	103.0 (+8%)	139.5 (+46%)	89.1 (-7%)	88.8 (-7%)
	r16	87.3	179.4 (+106%)	94.1 (+8%)	92.2 (+6%)	148.3 (+70%)	89.9 (+3%)	101.9 (+17%)
Llama-3.2-3B	r1	82.1	176.4 (+115%)	95.8 (+17%)	181.9 (+122%)	100.4 (+22%)	100.0 (+22%)	112.4 (+37%)
	r2	97.4	198.0 (+103%)	104.5 (+7%)	202.8 (+108%)	155.7 (+60%)	89.2 (-8%)	115.1 (+18%)
	r4	110.4	171.6 (+55%)	87.0 (-21%)	110.8 (+0%)	154.7 (+40%)	93.8 (-15%)	124.1 (+12%)
	r8	110.8	173.3 (+56%)	112.3 (+1%)	122.0 (+10%)	151.7 (+37%)	94.4 (-15%)	101.2 (-9%)
	r16	118.3	176.8 (+49%)	107.2 (-9%)	122.4 (+4%)	162.1 (+37%)	119.8 (+1%)	146.8 (+24%)

E HYPERPARAMETERS

Reshape dimensions and budget anchoring. Table 11 lists the exact trainable counts behind every anchor. LoRA, GPart, and Uni-LoRA share the LoRA count; the remaining primary-panel methods (VB-LoRA, Tied-LoRA, SingLoRA) are matched within the $\pm 15\%$ band by construction (Section 5; exact counts in the Params columns of Tables 5 and 6).

Table 10: Global reshape dimensions for the q/v scope used in all headline runs. N: number of adapted modules; $D = \sum_i p_i q_i$; $k = \lfloor \sqrt{D} \rfloor$; $m = \lceil D/k \rceil$; GFA global rank r_g that lands within the LoRA-r_ℓ budget anchor (from the released run manifest).

Model (q+v)	N	D	k	m	$r_g \mid_{r_\ell=4}$	$r_g \mid_{r_\ell=16}$
Qwen2.5-3B	72	169,869,312	13,033	13,034	35	141
Llama-3.2-3B	56	352,321,536	18,770	18,771	30	122

Table 11: **Budget anchoring on the decoder grid** (trainable parameters from the run manifest; identical across tasks and seeds). GFA global rank in parentheses. Ratio = baseline/GFA. LoRA, GPart, and Uni-LoRA share the same count at every anchor.

Model	Anchor	GFA (r)	LoRA = GPart = Uni-LoRA	ratio
Qwen2.5-3B	r=1	208,536 (8)	230,400	1.105
	r=2	443,139 (17)	460,800	1.040
	r=4	912,345 (35)	921,600	1.010
	r=8	1,824,690 (70)	1,843,200	1.010
	r=16	3,675,447 (141)	3,686,400	1.003
Llama-3.2-3B	r=1	262,787 (7)	286,720	1.091
	r=2	563,115 (15)	573,440	1.018
	r=4	1,126,230 (30)	1,146,880	1.018
	r=8	2,290,001 (61)	2,293,760	1.002
	r=16	4,580,002 (122)	4,587,520	1.002

Training hyperparameters. Table 12 lists the decoder recipe, read back from the merged configuration stored in every locked run rather than from the config files, so it records what was executed. Besides dropout, which differs by method, the only deviation is one Llama run (Uni-LoRA, ARC-Easy, r_ℓ=4, seed 42) that used a per-device batch of 4 with four accumulation steps and a maximum sequence length of 512. Table 13 gives the data splits, Table 14 the method-specific baseline settings, and Section E.1 the evaluation protocol.

Table 12: Training hyperparameters. Values are identical for all methods and all six tasks on both models unless a row says otherwise.

Setting	Value
Optimizer	AdamW (Loshchilov & Hutter, 2019), $\beta = (0.9, 0.999)$, $\epsilon = 10^{-8}$
Learning rate	10^{-4}
Precision	FP16, gradient checkpointing enabled
Seeds	42, 43, 44 (three paired seeds per cell)
Training length	3 epochs for every task and model; no step cap
Evaluation / checkpoints	evaluated once per epoch; no checkpoints saved
Checkpoint selection	none; the final (epoch-3) model is evaluated
Stopping rule	none; every run completes 3 epochs
Batch size $\times$ grad. accumulation	8×2; 4×4 for one Llama run (Uni-LoRA, ARC-Easy, r_ℓ=4, seed 42); effective batch 16 throughout
LR schedule	cosine decay, warm-up ratio 0.06
Weight decay	0.01
Max. gradient norm	1.0
Max. sequence length	256 tokens (train config overrides the per-task default); 512 for that run
LoRA-family α	$\alpha = 2r$
Dropout	0.05 for LoRA and Tied-LoRA; 0 for all other methods, including GFA
LoRA initialization	$B_0 = 0$, A_0 Kaiming-uniform ($a = \sqrt{5}$), so $\Delta W = 0$ at step 0
GFA	$A_0 = 0$, B_0 Kaiming-uniform (fan-in r_g), $s = 1$, no dropout
Data splits	Table 13

Table 13: **Data splits.** Sizes of the official releases; *labels not public. Accuracy is reported on the test split where its labels are public and on the validation split otherwise. There is no checkpoint selection: the final (epoch-3) model is evaluated.

Task	Train	Validation	Test	Reported
ARC-Easy	2,251	570	2,376	test
ARC-Challenge	1,119	299	1,172	test
PIQA	16,113	1,838	3,084*	validation
BoolQ	9,427	3,270	3,245*	validation
WinoGrande (XL)	40,398	1,267	1,767*	validation
OpenBookQA	4,957	500	500	test

E.1 Evaluation protocol

All six tasks are multiple-choice or binary classification tasks scored by accuracy. Prompts are raw text built from the templates in Table 15; the instruction models' chat templates are not applied. Each model answers by greedy decoding (one beam, no sampling) of at most 4 new tokens (ARC-Easy, PIQA, WinoGrande, OpenBookQA) or 8 (ARC-Challenge, BoolQ), and the answer letter, or Yes/No for BoolQ, is parsed from the generated string. This is not likelihood ranking: a model can lose a point by producing an unparseable answer as well as by choosing the wrong option.

Table 14: **Method-specific configurations** at the r_ℓ=4 anchor on Qwen; ranks scale with the budget. Fixed projections and permutations are resampled with the run seed. Settings not listed follow Table 12.

Method	Configuration	Initialization
LoRA	rank 4, $\alpha = 8$, dropout 0.05	$B_0 = 0$, A_0 Kaiming-uniform
Tied-LoRA	A and B shared per shape group with per-layer scaling vectors $u \in \mathbb{R}^r$, $v \in \mathbb{R}^{p_i}$; rank 130, $\alpha = 260$, dropout 0.05	shared $B_0 = 0$, A_0 Kaiming-uniform; $u = v = 1$
VB-LoRA	bank of 256 vectors of length 64, top-$k = 2$, rank 1, dropout 0	bank $\mathcal{U}(-0.02, 0.02)$, logits $\mathcal{N}(0, 0.1^2)$; $\Delta W \neq 0$ at step 0
SingLoRA	single factor, rank 6, $\alpha = 12$, ramp-up over 25 steps; 884,736 parameters (at r_ℓ=1: rank 1, both factors stored in one tensor)	A_0 Kaiming-uniform (fan-in $d_{\max}$); the ramp gives $\Delta W = 0$ at step 0
Uni-LoRA	rank 4, $\alpha = 8$; a trainable vector of the LoRA count mapped by a fixed permutation	vector $\mathcal{N}(0, 0.02^2)$; $\Delta W \neq 0$ at step 0
GPart	d = LoRA count, scale 1.0, fixed isometric projection	zero
GFA	$r_g = 35$, $k = 13{,}033$, $s = 1$, dropout 0	$A_0 = 0$, B_0 Kaiming-uniform (fan-in r_g)

Table 15: **Prompt templates.** `\n` is a newline; the answer options are listed as A., B., ...

Task	Template
ARC-E, ARC-C, OBQA	`Question: {q}\n\nA. ...\nB. ...\nC. ...\nD. ...\n\n Answer:`
PIQA	`Goal: {goal}\n\nA. {sol1}\nB. {sol2}\n\nAnswer:`
WinoGrande	`Sentence: {sentence}\n\nA. {option1}\nB. {option2}\n \nAnswer:`
BoolQ	`Read the passage and answer the question with Yes or No.\n\nPassage: {p}\n\nQuestion: {q}\n\nAnswer:`

F METHOD DESIGN COMPARISON

Table 16 lists the design choices each method exposes.

Table 16: Design choices exposed by each method. GFA exposes budget (rank), scope, reshape aspect, and scale; the near-square reshape and q/v scope are fixed defaults in our grid. "Random map?" asks whether reconstructing the trained adapter requires a fixed random map, and hence its seed; every method, including GFA, uses random initialization.

Method	Hyperparams	Per-layer params?	Random map?	Key design choice
LoRA	r, α, modules	Yes	No	Rank per module
DoRA	r, α, modules	Yes	No	Magnitude/direction split
VeRA	r, modules	Yes (diag.)	Yes	Random seed
VB-LoRA	r, bank size, top-k	Yes (selection)	No	Bank size, top-k
Tied-LoRA	r, α, tying config.	Scaling vec.	Varies	Factors tied across layers
SingLoRA	r, α, ramp-up	Yes	No	Single factor, $AA^\top$
GPart	d, modules	No	Yes	Fixed isometric partition
Uni-LoRA	r, d	No	Yes	Sparse projection
GFA	r, k (aspect), s, scope	No	No	Near-square reshape (default)

G BUDGET GRANULARITY OF THE NEAR-SQUARE LAYOUT

Proposition G.1 (Budget granularity). *With $k = \lfloor\sqrt{D}\rfloor$ and $m = \lceil D/k\rceil$, each unit of global rank costs $m + k < 2\sqrt{D} + 2$ parameters (for $D \geq 4$), whereas uniform rank-one LoRA costs $\sum_{i=1}^{N}(p_i + q_i)$. For N modules of equal shape $p \times q$, the ratio $\sum_i(p_i + q_i)/(m + k)$ is at least*

$$\sqrt{N}\,\frac{p+q}{2\sqrt{pq}} \cdot \frac{2\sqrt{D}}{2\sqrt{D}+2} \;\geq\; \sqrt{N}\,\frac{\sqrt{D}}{\sqrt{D}+1}, \tag{5}$$

so the GFA budget can be set about $\sqrt{N}$ times more finely than the LoRA budget.

Proof. Write $k = \sqrt{D} - f$ with $0 \leq f < 1$. Then $m < D/k + 1 = \sqrt{D} + f + f^2/(\sqrt{D} - f) + 1$, and $f^2/(\sqrt{D} - f) < 1$ when $\sqrt{D} \geq 2$, so $m + k < 2\sqrt{D} + 2$. For equal shapes, $D = Npq$ and $\sum_i(p_i + q_i) = N(p + q)$, so the ratio exceeds $N(p + q)/(2\sqrt{Npq} + 2)$, which equals the stated bound; finally $p + q \geq 2\sqrt{pq}$. □

For the q/v scopes used here, the exact ratios $\sum_i(p_i + q_i)/(m + k)$ are 8.84 for Qwen ($N = 72$, $\sqrt{N} = 8.49$) and 7.64 for Llama ($N = 56$, $\sqrt{N} = 7.48$), which is why GFA can operate far below the smallest budget of uniform LoRA.

H RANK BOUNDS, GENERICITY, AND EXACT CERTIFICATION

This appendix states and proves Theorem H.1, proves Theorem 4.1, gives a general upper bound on the rank of a reconstructed module, and records the exact certification used for the layouts in the paper as an independent check.

Maximum attainable rank. For a fixed global layout and rank r_g, define the maximum attainable rank of module i as

$$\rho_i(r_g) = \max_{B \in \mathbb{R}^{m \times r_g},\, A \in \mathbb{R}^{r_g \times k}} \operatorname{rank}\big(\Delta W_i(B, A)\big). \tag{6}$$

Theorem H.1 (Generic rank of reconstructed updates). *Fix arbitrary global dimensions (m, k), global rank r_g, module shapes, module ordering, and coordinate intervals. Then, for almost every $(B, A) \in \mathbb{R}^{m \times r_g} \times \mathbb{R}^{r_g \times k}$,*

$$\operatorname{rank}\big(\Delta W_i(B, A)\big) = \rho_i(r_g) \tag{7}$$

simultaneously for every $i = 1, \ldots, N$.

Moreover, if for some factors over a prime field $\mathbb{F}_\pi$ the reconstructed ΔW_i has rank t_i, then $\rho_i(r_g) \geq t_i$ over $\mathbb{R}$. In particular, if at $r_g = 1$ each module admits a witness with $t_i = \min(p_i, q_i)$, then for every $r_g \geq 1$ almost every GFA parameterization reconstructs all module updates at full rank simultaneously.

The proof uses that every minor of $\Delta W_i(B, A)$ is a polynomial in the factor entries.

For module i, row u of ΔW_i begins at coordinate $c_{i-1} + uq_i$ of the row-major vectorization of M. Define

$$o_{i,u} = (c_{i-1} + uq_i) \bmod k, \qquad \mathcal{C}_i = \{o_{i,u} : 0 \leq u < p_i\}. \tag{8}$$

The offsets advance by q_i modulo k, and therefore

$$|\mathcal{C}_i| = \min\left(p_i, \frac{k}{\gcd(q_i, k)}\right). \tag{9}$$

Proposition H.2 (Offset-sensitive module-rank bound). *Let $M = BA \in \mathbb{R}^{m \times k}$ satisfy $\operatorname{rank}(M) \leq r_g$. Then*

$$\operatorname{rank}(\Delta W_i) \leq \min\left\{p_i, q_i, r_g \sum_{c \in \mathcal{C}_i} \left\lceil \frac{c + q_i}{k} \right\rceil\right\}. \tag{10}$$

Proof. Fix a row u of ΔW_i and write $c = o_{i,u}$. Its length-q_i window begins at column c of one row of M and intersects

$$n_c = \left\lceil \frac{c + q_i}{k} \right\rceil \tag{11}$$

consecutive rows. Since every row of $M = BA$ belongs to the row space of A, the reconstructed row has the form

$$[x_0 A_{:,c:k}, x_1 A, \ldots, x_{n_c-1} A_{:,0:e}], \tag{12}$$

where each $x_j \in \mathbb{R}^{1\times r_g}$ is the corresponding row of B, $e = c + q_i - (n_c - 1)k$, and the first and last blocks are truncated as required (if $n_c = 1$ the row is $x_0 A_{:,c:c+q_i}$). For a fixed offset c, all such reconstructed rows lie in a subspace of $\mathbb{R}^{q_i}$ of dimension at most $r_g n_c$. The row space of ΔW_i is contained in the sum of these subspaces over $c \in \mathcal{C}_i$, giving the third term in Equation (10). Intersecting it with the ambient bound $\operatorname{rank}(\Delta W_i) \le \min(p_i, q_i)$ proves the result. Finally, addition by q_i modulo k has period $k/\gcd(q_i, k)$, which proves Equation (9). □

The proposition also shows why no unconditional full-rank statement can hold for every layout. For example, if $k \mid q_i$ and the interval begins at a global row boundary, then $\mathcal{C}_i = \{0\}$ and

$$\operatorname{rank}(\Delta W_i) \le \min\left(p_i, q_i, r_g \frac{q_i}{k}\right), \tag{13}$$

which may be strictly smaller than the ambient rank.

Corollary H.3 (Layout rule)**.** *If $q_i \le k$, then every reconstructed update satisfies*

$$\operatorname{rank}(\Delta W_i) \le \min\left\{p_i,\ q_i,\ \frac{2r_g\, k}{\gcd(q_i, k)}\right\}. \tag{14}$$

Hence ΔW_i can have full rank only if $k/\gcd(q_i, k) \ge \min(p_i, q_i)/(2r_g)$. In the width-aligned case $k = q_i$ with c_{i-1} a multiple of k, Proposition H.2 gives the sharper bound $\operatorname{rank}(\Delta W_i) \le r_g$.

Proof. Every offset satisfies $0 \le c \le k - 1$, so $c + q_i \le 2k - 1$ and each term of Equation (10) satisfies $\lceil (c + q_i)/k \rceil \le 2$. By Equation (9), $|\mathcal{C}_i| \le k/\gcd(q_i, k)$, which gives the bound. If $k = q_i$ and c_{i-1} is a multiple of k, every row of ΔW_i starts at offset 0, so $\mathcal{C}_i = \{0\}$ and the single term equals $\lceil q_i/k \rceil = 1$. □

Proof of Theorem 4.1. Let $t = \min\{p_i, q_i, h_i\}$. We first exhibit complex rank-one factors for which a $t \times t$ minor of ΔW_i is nonzero. Take $B = \mathbf{1}_m$ and $A = a^\top$ with $a \in \mathbb{C}^k$. Every row of BA then equals $a^\top$, so $[\operatorname{vec}(BA)]_n = a_{n \bmod k}$ for $0 \le n < mk$, and

$$[\Delta W_i]_{u,v} = a_{(c_{i-1} + uq_i + v) \bmod k}, \qquad 0 \le u < p_i, \quad 0 \le v < q_i. \tag{15}$$

Let $\omega = e^{2\pi\sqrt{-1}/k}$ and $a_j = \sum_{\ell=0}^{t-1} \omega^{\ell j}$ for $0 \le j < k$. Because $\omega^k = 1$, $a_{n \bmod k} = \sum_{\ell=0}^{t-1} \omega^{\ell n}$ for every integer $n \ge 0$. Hence, for $0 \le u, v < t$,

$$[\Delta W_i]_{u,v} = \sum_{\ell=0}^{t-1} x_u^\ell\, y_v^\ell, \qquad x_u = \omega^{c_{i-1} + uq_i}, \quad y_v = \omega^v, \tag{16}$$

so the leading $t \times t$ block equals $UV^\top$ with the Vandermonde matrices $U = [x_u^\ell]_{u,\ell<t}$ and $V = [y_v^\ell]_{v,\ell<t}$, whose determinants are $\prod_{u<u'}(x_{u'} - x_u)$ and $\prod_{v<v'}(y_{v'} - y_v)$. The nodes y_v are distinct because $t \le h_i \le k$. If $x_u = x_{u'}$ for some $0 \le u < u' < t$, then k divides $(u' - u)q_i$, so $h_i = k/\gcd(k, q_i)$ divides $u' - u$, which is impossible because $0 < u' - u < t \le h_i$. The block is therefore nonsingular. Its determinant is a polynomial f with integer coefficients in the entries of (B, A). Since f does not vanish at this complex point, it is not the zero polynomial, so it is also nonzero at some real (B, A), and $\rho_i(1) \ge t$. For $r_g > 1$, placing the rank-one factors in the first column of B and the first row of A and setting all remaining entries to zero reproduces the same product, so the corresponding minor polynomial is again nonzero and $\rho_i(r_g) \ge t$.

If $q_i \le k$, Corollary H.3 gives $\rho_i(r_g) \le \min\{p_i, q_i, 2r_g h_i\}$. Finally, if $h_i \ge \min(p_i, q_i)$ for every module, then $\rho_i(r_g) = \min(p_i, q_i)$ for every i, and Theorem H.1 shows that almost every (B, A) attains these ranks simultaneously. □

Proof of Theorem H.1. Fix r_g and module i, and let $\rho_i(r_g)$ be defined by Equation (6). Every entry of $\Delta W_i(B, A)$ is a polynomial in the entries of B and A. By the definition of $\rho_i(r_g)$, at least one $\rho_i(r_g) \times \rho_i(r_g)$ minor has a determinant $f_i(B, A)$ that is not the zero polynomial. The rank-deficient set

$$\mathcal{Z}_i = \{(B, A) : \operatorname{rank}(\Delta W_i(B, A)) < \rho_i(r_g)\} \tag{17}$$

is contained in the zero set of f_i. Since a nonzero real polynomial has a zero set of Lebesgue measure zero, $\mathcal{Z}_i$ has measure zero. The union $\bigcup_{i=1}^{N} \mathcal{Z}_i$ also has measure zero because the number of modules is finite. Outside this union, every module simultaneously attains its maximum rank.

For the finite-field statement, suppose a $t_i \times t_i$ minor evaluates to a nonzero element of $\mathbb{F}_\pi$ for some factors over the prime field $\mathbb{F}_\pi$. The corresponding determinant is a polynomial with integer coefficients. A nonzero evaluation modulo π proves that this polynomial is not identically zero over $\mathbb{Z}$ and hence is not identically zero over $\mathbb{R}$. Therefore, $\rho_i(r_g) \geq t_i$ over $\mathbb{R}$.

If full rank is certified at $r_g = 1$, the witness can be embedded at any $r_g > 1$ by placing its two factors in one column of B and one row of A and setting all remaining entries to zero. Thus, the relevant maximal-minor polynomial remains nonzero for every $r_g \geq 1$. □

Corollary H.4 (Full rank in the deployed layouts)**.** *In the near-square Qwen and Llama layouts used in the paper, and for every module ordering, for every $r_g \geq 1$ and almost every real-valued GFA factor pair, all reconstructed module updates are simultaneously full rank.*

Proof. All Qwen modules have $q_i = 2048$, and $k = 13{,}033$ is odd, so $h_i = 13{,}033$. All Llama modules have $q_i = 3072$, and $\gcd(18{,}770, 3072) = 2$, so $h_i = 9{,}385$. In both cases $h_i \geq q_i \geq \min(p_i, q_i)$, and Theorem 4.1 applies. □

Independent finite-field check. The certificate uses the prime field $\mathbb{F}_\pi$ with $\pi = 2^{31} - 1$. For every adapted interval, the released verifier constructs rank-one factors, performs modular Gaussian elimination on the reconstructed module, and obtains rank $\min(p_i, q_i)$. By the finite-field part of Theorem H.1, each corresponding maximal-minor polynomial is nonzero over $\mathbb{R}$. The simultaneous generic statement then follows from the first part of the theorem. For the three module orders listed below, this confirms the corollary independently of Theorem 4.1. □

The certified ambient ranks are 2048 and 256 for the Qwen query and value projections, respectively, and 3072 and 1024 for the Llama query and value projections. These are exact algebraic certificates rather than conclusions drawn from floating-point numerical-rank thresholds. The same certificates hold, for both models, under two other module orders: all query projections before all value projections, and the value projection before the query projection within each layer, as Theorem 4.1 predicts.

I MATCHED-BUDGET REPRESENTABLE SETS

This appendix gives general conditions under which the GFA and LoRA representable sets defined below are incomparable and verifies them for the reported Qwen budget anchors, and then proves Proposition 4.2 and evaluates its bounds for both deployed layouts.

Let $\mathcal{R}$ be the crop, slice, and reshape map of Equation (1). For a parameter budget P, let $r_G(P) = \lfloor P/(m + k) \rfloor$ and

$$\mathcal{S}_G(P) = \left\{ \mathcal{R}(BA) : B \in \mathbb{R}^{m \times r_G(P)},\ A \in \mathbb{R}^{r_G(P) \times k} \right\}. \tag{18}$$

For a collection $X = (X_1, \ldots, X_N)$ of module updates, its minimum parameter count under the standard module-wise LoRA factorization is

$$C_L(X) = \sum_{i=1}^{N} \operatorname{rank}(X_i)(p_i + q_i), \tag{19}$$

and allowing the LoRA rank to be allocated arbitrarily across modules gives

$$\mathcal{S}_L(P) = \{X : C_L(X) \leq P\}, \tag{20}$$

which is more permissive than assigning one common LoRA rank to every module.

Let

$$m_0 = \left\lfloor \frac{D}{k} \right\rfloor \tag{21}$$

be the number of complete global rows determined by the first D coordinates. For a module collection $X = (X_1, \ldots, X_N)$, define

$$\Psi(X) = \mathrm{reshape}_{m_0 \times k}\left(\left[\mathrm{concat}_{i=1}^{N} \mathrm{vec}(X_i) \right]_{0:m_0 k} \right). \tag{22}$$

Thus, $\Psi(X)$ contains every complete row of the global layout and excludes only a possible partially padded final row.

Proposition I.1 (Sufficient conditions for matched-budget incomparability)**.** *For any module collection, global layout, and budget P, the following hold.*

(a) If there exists $X_G \in \mathcal{S}_G(P)$ such that $C_L(X_G) > P$, then $\mathcal{S}_G(P) \not\subseteq \mathcal{S}_L(P)$.

(b) Every $X_G \in \overline{\mathcal{S}_G(P)}$ satisfies

$$\mathrm{rank}\big(\Psi(X_G)\big) \le r_G(P). \tag{23}$$

(c) If there exists $X_L \in \mathcal{S}_L(P)$ such that $\mathrm{rank}(\Psi(X_L)) > r_G(P)$, then $\mathcal{S}_L(P) \not\subseteq \overline{\mathcal{S}_G(P)}$.

Proof. For part (a), Equation (19) is the smallest parameter count of a standard module-wise factorization that represents X_G exactly. Thus, $C_L(X_G) > P$ implies $X_G \notin \mathcal{S}_L(P)$.

For part (b), take a convergent sequence $X_n = \mathcal{R}(B_n A_n) \to X_G$ with $\mathrm{rank}(B_n A_n) \le r_G(P)$. The matrix $\Psi(X_n)$ consists of complete rows of $B_n A_n$, so $\mathrm{rank}(\Psi(X_n)) \le r_G(P)$. Since the set of matrices of rank at most $r_G(P)$ is closed and Ψ is continuous, the limit satisfies $\mathrm{rank}(\Psi(X_G)) \le r_G(P)$.

Part (c) is the contrapositive of part (b). A point X_L with $\mathrm{rank}(\Psi(X_L)) > r_G(P)$ cannot belong to $\overline{\mathcal{S}_G(P)}$. If it also belongs to $\mathcal{S}_L(P)$, then the latter set cannot be contained in the former. □

The closure is necessary because cropping a partially padded final row is a linear projection of the rank-constrained global matrix family, and the image of a closed set under a linear projection need not be closed. The complete rows retained by Ψ avoid this issue.

Corollary I.2 (Incomparability at the reported Qwen budgets)**.** *For the Qwen q/v layout and each of the five standard budget anchors $r_\ell \in \{1, 2, 4, 8, 16\}$,*

$$\mathcal{S}_G(P) \not\subseteq \mathcal{S}_L(P) \qquad \textit{and} \qquad \mathcal{S}_L(P) \not\subseteq \overline{\mathcal{S}_G(P)}. \tag{24}$$

Proof. For the first direction, Corollary H.4 provides budget-feasible GFA points whose 72 reconstructed updates are all full rank. Their exact standard-LoRA representation cost is approximately 3.23×10^8 parameters, which exceeds the largest reported adapter budget. The same rank-one GFA point belongs to every larger reported GFA family by zero-padding the unused factor dimensions. Part (a) of Proposition I.1 therefore applies.

For the reverse direction, the exact finite-field verifier constructs, at each reported LoRA budget, module-wise updates satisfying $C_L(X_L) \le P$ whose unpadded global reshape contains a 512×512 minor of full modular rank. Hence $\mathrm{rank}(\Psi(X_L)) \ge 512$. The largest reported GFA rank is 141, so $512 > r_G(P)$ at every anchor. Part (c) of Proposition I.1 proves the reverse noncontainment. □

Quantitative separation. Proposition I.5 below gives an analytic and quantitative version of both directions for general layouts, and Proposition 4.2 specializes it to Qwen and Llama. Its proof uses two approximation identities. For a collection $X = (X_i)$ of module updates, write $\|X\|^2 = \sum_i \|X_i\|_F^2$; $\mathbf{1}_n$ is the all-ones vector in $\mathbb{R}^n$ and e_1 the first standard basis vector of the appropriate dimension. Write $\tau_r(Y) = \sum_{j>r} \sigma_j(Y)^2$ for the squared singular-value tail of a matrix Y beyond its r largest singular values, and $\mathrm{dist}(X, \mathcal{S}) = \inf_{Y \in \mathcal{S}} \|X - Y\|$. By the Eckart–Young theorem, $\min_{\mathrm{rank}(Z) \le r} \|Y - Z\|_F^2 = \tau_r(Y)$. Every matrix of rank at most r factors as BA with inner dimension r, so $\mathcal{S}_G(P) = \{\mathcal{R}(M) : \mathrm{rank}(M) \le r_G(P)\}$.

Lemma I.3 (Best LoRA approximation)**.** *For every collection X and budget $P \geq 0$,*

$$\operatorname{dist}\big(X, \mathcal{S}_L(P)\big)^2 = \min\Big\{\sum_{i=1}^{N} \tau_{r_i}(X_i) : \; r_i \in \mathbb{Z}_{\geq 0},\; \sum_{i=1}^{N} r_i(p_i + q_i) \leq P\Big\}. \tag{25}$$

Proof. A collection Y belongs to $\mathcal{S}_L(P)$ exactly when $\operatorname{rank}(Y_i) \leq r_i$ for every i for some feasible allocation r: take $r_i = \operatorname{rank}(Y_i)$ in one direction, and use $C_L(Y) \leq \sum_i r_i(p_i + q_i) \leq P$ in the other. For a fixed allocation, the squared distance separates across modules and equals $\sum_i \tau_{r_i}(X_i)$ by the Eckart–Young theorem. There are finitely many feasible allocations, so the minimum is attained. □

Lemma I.4 (Best GFA approximation)**.** *Let $r = r_G(P)$, and for $z \in \mathbb{R}^{mk-D}$ let $T_z(X) \in \mathbb{R}^{m\times k}$ be the row-major reshape of the concatenated coordinates of X followed by z. Then*

$$\operatorname{dist}\big(X, \mathcal{S}_G(P)\big)^2 = \inf_z \tau_r\big(T_z(X)\big) \quad \textit{and} \quad \tau_r\big(\Psi(X)\big) \leq \operatorname{dist}\big(X, \mathcal{S}_G(P)\big)^2 \leq \tau_r\big(T_0(X)\big). \tag{26}$$

Proof. For $M \in \mathbb{R}^{m\times k}$, let $z(M)$ collect its last $mk - D$ row-major coordinates. Then $\|\mathcal{R}(M) - X\|^2 = \|M - T_{z(M)}(X)\|_F^2 = \min_z \|M - T_z(X)\|_F^2$, because the padding coordinates only add a nonnegative term that vanishes at $z = z(M)$. Hence

$$\operatorname{dist}\big(X, \mathcal{S}_G(P)\big)^2 = \inf_{\operatorname{rank}(M)\leq r} \min_z \|M - T_z(X)\|_F^2 \tag{27}$$

$$= \inf_z \min_{\operatorname{rank}(M)\leq r} \|M - T_z(X)\|_F^2 = \inf_z \tau_r\big(T_z(X)\big), \tag{28}$$

and $z = 0$ gives the upper bound. Since $m = \lceil D/k \rceil$, the padding has fewer than k coordinates and lies in the last row, so the first m_0 rows of $T_z(X)$ equal $\Psi(X)$ for every z. The first m_0 rows of M have rank at most r, so $\|M - T_z(X)\|_F^2 \geq \tau_r(\Psi(X))$, which gives the lower bound. □

Proposition I.5 (Quantitative incomparability, general form)**.** *Suppose $q_i \leq k$ and $p_i \leq h_i = k/\gcd(k, q_i)$ for every module, and let $g = \min_i(p_i + q_i)$. (i) The update $X_G = \mathcal{R}(\mathbf{1}_m e_1^\top)/\sqrt{m}$ has unit norm, belongs to $\mathcal{S}_G(P)$ for every $P \geq m + k$, has modules that are scaled partial permutation matrices, and satisfies $\operatorname{dist}(X_G, \mathcal{S}_L(P))^2 \geq 1 - \lfloor P/g \rfloor/m$. (ii) If module i lies within the first $\lfloor D/k \rfloor$ rows of M, which contain no padding, the update X_L with $X_{L,i} = \mathbf{1}_{p_i} e_1^\top/\sqrt{p_i}$ and all other modules zero has unit norm, belongs to $\mathcal{S}_L(P)$ for every $P \geq p_i + q_i$, and satisfies $\operatorname{dist}(X_L, \mathcal{S}_G(P))^2 \geq 1 - r_G(P)\lceil k/q_i \rceil/p_i$.*

Proof of Proposition I.5. (i) Every row of $\mathbf{1}_m e_1^\top$ has a single one, in its first column, so $\operatorname{vec}(\mathbf{1}_m e_1^\top)$ has ones exactly at the positions xk for $0 \leq x < m$. All of them are observed because $(m-1)k < D$. Thus X_G has exactly m nonzero coordinates, each equal to $1/\sqrt{m}$, so $\|X_G\| = 1$, and $X_G \in \mathcal{S}_G(P)$ whenever $r_G(P) \geq 1$, by zero-padding the factors. A row of module i occupies $q_i \leq k$ consecutive coordinates, so it contains at most one multiple of k. If rows $u \neq u'$ of module i both had a one in column v, then k would divide $(u - u')q_i$, and h_i would divide $u - u'$, which is impossible because $0 < |u - u'| < p_i \leq h_i$. Hence $\sqrt{m}\, X_{G,i}$ is a partial permutation matrix (a $0/1$ matrix with at most one 1 in each row and column), and every nonzero singular value of $X_{G,i}$ equals $1/\sqrt{m}$. Let ν_i be their number, so $\sum_i \nu_i = m$. For a feasible allocation, $\tau_{r_i}(X_{G,i}) = \max(\nu_i - r_i, 0)/m$, so by Lemma I.3

$$\operatorname{dist}\big(X_G, \mathcal{S}_L(P)\big)^2 \geq \min_r \frac{m - \sum_i r_i}{m} \geq 1 - \frac{\lfloor P/g \rfloor}{m}, \tag{29}$$

because $\sum_i r_i \leq P/g$ and the r_i are integers.

(ii) $X_{L,i}$ has rank one and all other modules are zero, so $C_L(X_L) = p_i + q_i$, and $\|X_L\| = 1$. Its nonzero coordinates, each equal to $1/\sqrt{p_i}$, sit at the global positions $n_u = c_{i-1} + uq_i$ for $0 \leq u < p_i$, which lie within the first m_0 rows of M by assumption, so $\|\Psi(X_L)\|_F = 1$. Their columns $n_u \bmod k$ are distinct, because $n_u \equiv n_{u'} \pmod k$ would again force $h_i \mid (u - u')$. The rows of $\Psi(X_L)$ therefore have disjoint supports, and its singular values are its row norms. A row of M holds k consecutive coordinates, so it contains at most $\lceil k/q_i \rceil$ of the points n_u, which are spaced q_i apart. Each squared singular value is therefore at most $\lceil k/q_i \rceil/p_i$, so $\tau_{r_G(P)}(\Psi(X_L)) \geq 1 - r_G(P)\lceil k/q_i \rceil/p_i$, and Lemma I.4 gives the claim. Because $T_0(X_L)$ is $\Psi(X_L)$ with zero rows appended, the same lemma shows that this tail is in fact the exact squared distance.

For robustness, $\operatorname{dist}(X+E,\mathcal{S}) \geq \operatorname{dist}(X,\mathcal{S}) - \|E\|$ for any set $\mathcal{S}$, and $\operatorname{dist}(X+E,\mathcal{S}) \leq \|E\|$ when $X \in \mathcal{S}$. □

Table 17: **Bounds of Proposition I.5 at the five budget anchors.** (i) Lower bound $1 - \lfloor P/g \rfloor / m$ on $\operatorname{dist}(X_G, \mathcal{S}_L(P))^2$. (ii) Lower bound $1 - r_G(P)\lceil k/q_i \rceil / p_i$ on $\operatorname{dist}(X_L, \mathcal{S}_G(P))^2$ for the first query module. P is the LoRA count at each anchor. Qwen: $m = 13{,}034$, $g = 2{,}304$, $p_i = 2{,}048$; Llama: $m = 18{,}771$, $g = 4{,}096$, $p_i = 3{,}072$; $\lceil k/q_i \rceil = 7$ for both. Entries are rounded down.

Model	Bound	r_ℓ=1	r_ℓ=2	r_ℓ=4	r_ℓ=8	r_ℓ=16
Qwen2.5-3B	(i)	0.992	0.984	0.969	0.938	0.877
	(ii)	0.972	0.941	0.880	0.760	0.518
Llama-3.2-3B	(i)	0.996	0.992	0.985	0.970	0.940
	(ii)	0.984	0.965	0.931	0.861	0.722

Proof of Proposition 4.2. Both deployed layouts satisfy $q_i \leq k$ and $p_i \leq h_i$, since $p_i \leq 3072 < 9{,}385 \leq h_i$, and the first query module lies within the first $\lfloor D/k \rfloor$ rows of M. Every budget between those of rank-1 and rank-16 LoRA is at least $m+k$ and p_i+q_i, so both witnesses of Proposition I.5 belong to their families. Both of its bounds are nonincreasing in P, so over this range they are smallest at the rank-16 budget, where Table 17 gives 0.877 and 0.518 on Qwen and 0.940 and 0.722 on Llama. □

Neither direction ranks the two parameterizations on a learning task. For example, centering a strongly convex quadratic loss at a witness from either set gives zero approximation error for that family and strictly positive error for the other. The result establishes structural incomparability, not universal performance dominance.

J Optimization lemmas and convergence proof

This appendix derives factor-space smoothness constants for GFA and LoRA, bounds their stochastic-gradient noise, shows that their zero-update initialization is nonstationary almost surely, and states and proves Theorem J.1.

Setting. Let $\mathcal{L}(\Delta W_1, \ldots, \Delta W_N)$ denote the downstream loss as a function of the additive updates, and let $\mathcal{R} : \mathbb{R}^{m\times k} \to \prod_{i=1}^{N} \mathbb{R}^{p_i \times q_i}$ denote the deterministic crop, slice, and reshape operation in Equation (1). The GFA factor objective is

$$F_G(B,A) = \mathcal{L}\big(s\,\mathcal{R}(BA)\big), \tag{30}$$

while the LoRA factor objective is

$$F_L(\mathbf{B}^L, \mathbf{A}^L) = \mathcal{L}\left(\left\{s_i^L B_i^L A_i^L\right\}_{i=1}^{N}\right), \tag{31}$$

where s_i^L is the fixed update scale of module i.

Assumption 1 (Optimization regularity)**.** *The update-space loss $\mathcal{L}$ is twice continuously differentiable, β-smooth under the product Frobenius norm, and bounded below by $\mathcal{L}_{\inf}$. The GFA and LoRA iterates, including stochastic ones, remain almost surely in compact convex regions $\mathcal{K}_G$ and $\mathcal{K}_L$, respectively.*

Under Assumption 1, the two factor objectives are smooth on their respective trajectory regions, with the explicit constants of Lemma J.2 below.

Theorem J.1 (First-order convergence of GFA and LoRA)**.** *Let $X \in \{G, L\}$ and let Θ_X collect the corresponding factors. Suppose Assumption 1 holds and F_X is L_X-smooth on the trajectory region. Stochastic gradient descent*

$$\Theta_{X,t+1} = \Theta_{X,t} - \eta_X g_{X,t}, \qquad 0 < \eta_X \leq \frac{1}{L_X}, \tag{32}$$

with $\mathbb{E}[g_{X,t} \mid \Theta_{X,t}] = \nabla F_X(\Theta_{X,t})$ *and* $\mathbb{E}\big[\|g_{X,t} - \nabla F_X(\Theta_{X,t})\|_F^2 \mid \Theta_{X,t}\big] \le \sigma_X^2$*, satisfies*

$$\min_{0\le t<T} \mathbb{E}\,\|\nabla F_X(\Theta_{X,t})\|_F^2 \le \frac{2\Delta_X}{\eta_X T} + \eta_X L_X \sigma_X^2, \qquad \Delta_X = F_X(\Theta_{X,0}) - \mathcal{L}_{\inf}. \tag{33}$$

Full-batch gradient descent is the case $\sigma_X = 0$*, with rate* $O(1/T)$*. For* $\sigma_X > 0$*, the step size* $\eta_X = \min\{1/L_X, \sqrt{2\Delta_X/(L_X\sigma_X^2 T)}\}$ *bounds the left-hand side by* $4L_X\Delta_X/T + 2\sigma_X\sqrt{2L_X\Delta_X/T}$*, a rate of* $O(1/\sqrt{T})$*. Both GFA and LoRA therefore attain these rates under the same class of assumptions.*

Under the zero-update initialization used by both methods, $F_X(\Theta_{X,0}) = \mathcal{L}(0)$, so $\Delta_G = \Delta_L$ and the two bounds differ only through L_X and σ_X. For minibatch gradients, Lemma J.3 bounds both noise levels by the same update-space noise, scaled by the squared factor norms and update scales, in the same way as the smoothness constants.

The theorem does not establish that either method converges faster. Their smoothness constants and noise levels depend on the factor norms, update scales, and gradients encountered during optimization and are not ordered by the parameter budget. The result concerns full-batch and stochastic gradient descent in factor space. Lemma J.4 shows that the zero-update initialization used by both methods is nonstationary almost surely whenever the update-space gradient is nonzero.

For GFA, define

$$R_G^2 = \sup_{(B,A)\in\mathcal{K}_G} \big(\|B\|_2^2 + \|A\|_2^2\big) \tag{34}$$

and

$$\Gamma_G = \sup_{(B,A)\in\mathcal{K}_G} \big\|\mathcal{R}^*\nabla_\Delta\mathcal{L}\big(s\mathcal{R}(BA)\big)\big\|_2 . \tag{35}$$

For LoRA, let

$$R_L^2 = \sup_{\mathcal{K}_L} \max_i \big(\|B_i^L\|_2^2 + \|A_i^L\|_2^2\big), \tag{36}$$

$$\Gamma_L = \sup_{\mathcal{K}_L} \max_i \|\nabla_{\Delta W_i}\mathcal{L}\|_2 , \qquad s_L = \max_i |s_i^L|. \tag{37}$$

These quantities are finite because $\mathcal{K}_G$ and $\mathcal{K}_L$ are compact and $\mathcal{L}$ is C^2 (Assumption 1); taking suprema over the regions, not only over iterates, makes the constants valid on the segments between consecutive iterates used by the descent lemma.

Lemma J.2 (Smoothness of the factor objectives)**.** *Under Assumption 1, the GFA and LoRA factor objectives are smooth on their trajectory regions with valid constants*

$$L_G \le \beta s^2 R_G^2 + |s|\Gamma_G \tag{38}$$

and

$$L_L \le \beta s_L^2 R_L^2 + s_L\Gamma_L, \tag{39}$$

respectively.

Proof. Consider a GFA factor-space direction $H = (H_B, H_A)$ and write

$$Z = H_B A + B H_A. \tag{40}$$

Because $\mathcal{R}$ only crops and reshapes coordinates, $\|\mathcal{R}(Z)\|_F \le \|Z\|_F$. Moreover,

$$\|Z\|_F \le \|H_B\|_F\|A\|_2 + \|B\|_2\|H_A\|_F \tag{41}$$

$$\le R_G\sqrt{\|H_B\|_F^2 + \|H_A\|_F^2}. \tag{42}$$

The second directional derivative of F_G is

$$D^2F_G(B,A)[H,H] = s^2\left\langle \mathcal{R}(Z), \nabla^2\mathcal{L}\,\mathcal{R}(Z)\right\rangle \tag{43}$$

$$+ 2s\left\langle \mathcal{R}^*\nabla_\Delta\mathcal{L}, H_B H_A\right\rangle . \tag{44}$$

The first term is bounded in absolute value by $\beta s^2 R_G^2\|H\|_F^2$. For the second term,

$$2|s|\,|\langle \mathcal{R}^*\nabla_\Delta\mathcal{L}, H_B H_A\rangle| \le 2|s|\Gamma_G\|H_B\|_F\|H_A\|_F \tag{45}$$

$$\le |s|\Gamma_G\|H\|_F^2. \tag{46}$$

Thus, the operator norm of the Hessian is at most the right-hand side of Equation (38).

For LoRA, apply the same argument to each block direction (H_{B_i}, H_{A_i}) and use

$$\sum_i \left\| s_i^L (H_{B_i} A_i^L + B_i^L H_{A_i}) \right\|_F^2 \le s_L^2 R_L^2 \|H\|_F^2. \tag{47}$$

The mixed bilinear term is at most $s_L \Gamma_L \|H\|_F^2$, proving Equation (39). □

Lemma J.3 (Factor-space gradient noise)**.** *Let $\widehat{G}$ be a minibatch estimate of the update-space gradient $\nabla_\Delta \mathcal{L}$ at the current update, with $\mathbb{E}\widehat{G} = \nabla_\Delta \mathcal{L}$ and $\mathbb{E}\|\widehat{G} - \nabla_\Delta \mathcal{L}\|_F^2 \le \sigma_\Delta^2$ at every update reached from $\mathcal{K}_G$ and $\mathcal{K}_L$. The chain-rule estimates*

$$g_G = \big(s\,\mathcal{R}^*(\widehat{G})A^\top,\ s\,B^\top \mathcal{R}^*(\widehat{G})\big), \tag{48}$$

$$g_L = \big\{\big(s_i^L \widehat{G}_i (A_i^L)^\top,\ s_i^L (B_i^L)^\top \widehat{G}_i\big)\big\}_{i=1}^N \tag{49}$$

are unbiased, and their variances satisfy

$$\sigma_G^2 \le s^2 R_G^2 \sigma_\Delta^2, \qquad \sigma_L^2 \le s_L^2 R_L^2 \sigma_\Delta^2. \tag{50}$$

Proof. For fixed factors, both estimates are linear in $\widehat{G}$, so they are unbiased. Write $E = \widehat{G} - \nabla_\Delta \mathcal{L}$ and let E_i be its block for module i. The adjoint $\mathcal{R}^*$ places each module block at its coordinate interval and fills the padding with zeros, so $\|\mathcal{R}^*(E)\|_F = \|E\|_F$. Hence

$$\|g_G - \nabla F_G\|_F^2 = s^2\big(\|\mathcal{R}^*(E)A^\top\|_F^2 + \|B^\top \mathcal{R}^*(E)\|_F^2\big) \tag{51}$$

$$\le s^2\big(\|A\|_2^2 + \|B\|_2^2\big)\|E\|_F^2 \le s^2 R_G^2 \|E\|_F^2, \tag{52}$$

and taking expectations gives the GFA bound. For LoRA,

$$\|g_L - \nabla F_L\|_F^2 = \sum_{i=1}^N (s_i^L)^2\big(\|E_i (A_i^L)^\top\|_F^2 + \|(B_i^L)^\top E_i\|_F^2\big) \tag{53}$$

$$\le s_L^2 R_L^2 \sum_{i=1}^N \|E_i\|_F^2 = s_L^2 R_L^2 \|E\|_F^2. \tag{54}$$

□

Lemma J.4 (Nonstationary zero-update initialization)**.** *Suppose $s \neq 0$, $A_0 = 0$, and B_0 is drawn from an absolutely continuous distribution. If*

$$G_0 = \mathcal{R}^* \nabla_\Delta \mathcal{L}(0) \neq 0, \tag{55}$$

then

$$\nabla_B F_G(B_0, A_0) = 0, \qquad \nabla_A F_G(B_0, A_0) = s B_0^\top G_0 \neq 0 \tag{56}$$

almost surely. The analogous statement holds for LoRA whenever at least one module has a nonzero update-space gradient at initialization.

Proof. The chain rule gives

$$\nabla_B F_G = s G A^\top, \qquad \nabla_A F_G = s B^\top G, \tag{57}$$

where $G = \mathcal{R}^* \nabla_\Delta \mathcal{L}$. Setting $A_0 = 0$ gives the first equality. If $G_0 \neq 0$, the condition $B_0^\top G_0 = 0$ defines a proper linear subspace of the entries of B_0, which has probability zero under an absolutely continuous distribution. The LoRA statement follows by applying the same argument independently to each module. □

Proof of Theorem J.1. Fix $X \in \{G, L\}$ and write $\nabla_t = \nabla F_X(\Theta_{X,t})$ and $g_t = g_{X,t}$. Both $\Theta_{X,t}$ and $\Theta_{X,t+1}$ lie in the convex region $\mathcal{K}_X$, on which F_X is L_X-smooth, so the descent lemma gives

$$F_X(\Theta_{X,t+1}) \le F_X(\Theta_{X,t}) - \eta_X \langle \nabla_t, g_t \rangle + \frac{L_X \eta_X^2}{2} \|g_t\|_F^2. \tag{58}$$

Conditionally on $\Theta_{X,t}$, $\mathbb{E}\langle \nabla_t, g_t\rangle = \|\nabla_t\|_F^2$ and $\mathbb{E}\|g_t\|_F^2 = \|\nabla_t\|_F^2 + \mathbb{E}\|g_t - \nabla_t\|_F^2 \le \|\nabla_t\|_F^2 + \sigma_X^2$. Since $\eta_X \le 1/L_X$ implies $1 - L_X\eta_X/2 \ge 1/2$,

$$\mathbb{E}\big[F_X(\Theta_{X,t+1}) \mid \Theta_{X,t}\big] \le F_X(\Theta_{X,t}) - \frac{\eta_X}{2}\|\nabla_t\|_F^2 + \frac{L_X\eta_X^2\sigma_X^2}{2}. \tag{59}$$

Taking total expectations, summing from $t = 0$ to $T - 1$, and using $F_X(\Theta_{X,T}) \ge \mathcal{L}_{\inf}$ yields

$$\frac{\eta_X}{2}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla_t\|_F^2 \le \Delta_X + \frac{TL_X\eta_X^2\sigma_X^2}{2}. \tag{60}$$

Dividing by $\eta_X T/2$ and bounding the minimum by the average proves Equation (33); with $\sigma_X = 0$ this is the full-batch bound. For the step size, if $\sqrt{2\Delta_X/(L_X\sigma_X^2 T)} \le 1/L_X$, substituting it makes both terms equal to $\sigma_X\sqrt{2L_X\Delta_X/T}$. Otherwise $\eta_X = 1/L_X$ and $\sigma_X^2 < 2L_X\Delta_X/T$, so the bound is at most $4L_X\Delta_X/T$. In both cases it is at most $4L_X\Delta_X/T + 2\sigma_X\sqrt{2L_X\Delta_X/T}$. This is the standard nonconvex SGD argument (Ghadimi & Lan, 2013), applied to the factor objectives. □

The constants L_G and L_L are not ordered by a matched parameter budget. They depend on the scale and balance of the two factors and can be changed by the rescaling $B \mapsto \gamma B$, $A \mapsto A/\gamma$ without changing the represented update, and the same holds for the noise bounds of Lemma J.3. The theorem therefore establishes the same asymptotic stationarity rate for both methods, but no universal advantage in its constant.